Podstawy Teorii Informacji

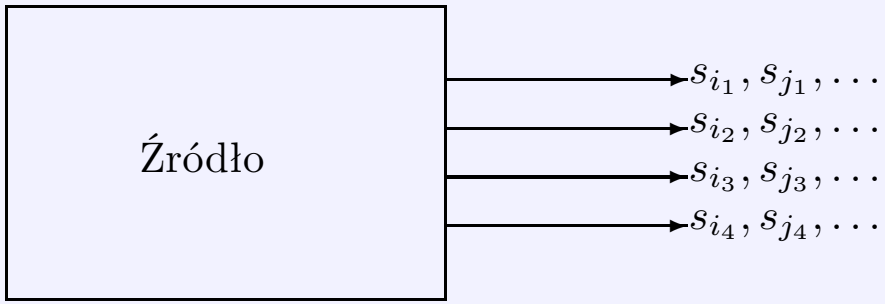
Wykład 2
Źródła informacji

Wojciech Kordecki
wojciech.kordecki@collegiumwitelona.pl

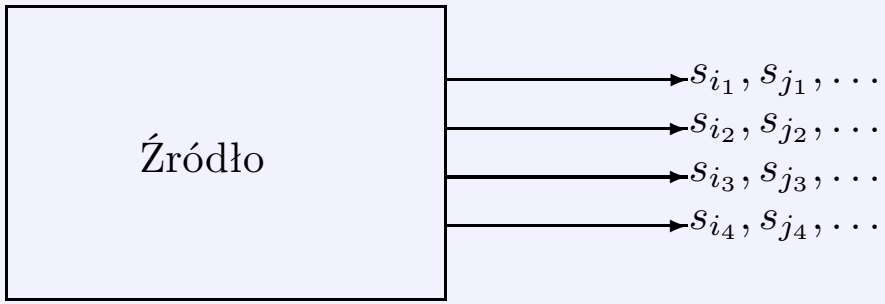
Collegium Witelona
Wydział Nauk Technicznych i Ekonomicznych
Zakład Informatyki

Semestr zimowy 2025/26

Rozszerzenie źródła

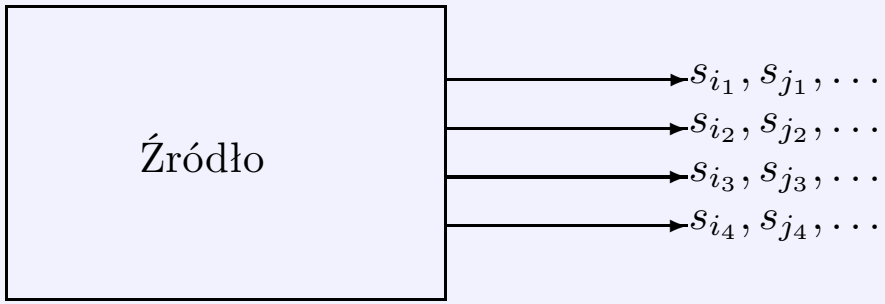


Rozszerzenie źródła

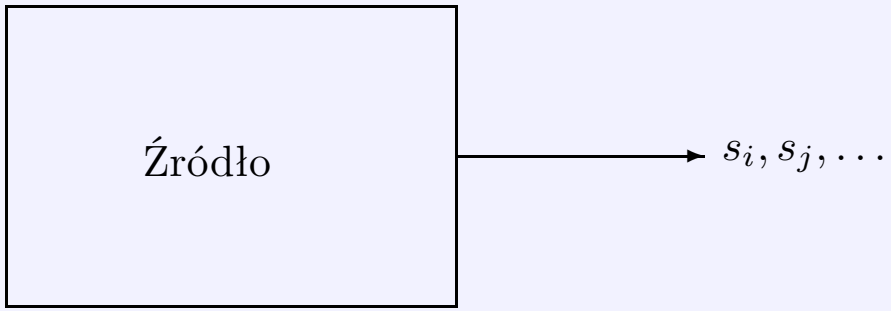


$$s_k = \left(s_{k_1}, s_{k_2}, s_{k_3}, s_{k_4}\right).$$

Rozszerzenie źródła



$$s_k = \left(s_{k_1}, s_{k_2}, s_{k_3}, s_{k_4}\right).$$



Źródło binarne

Dwa wyjścia:

$$S = \{00, 01, 10, 11\},$$

Źródło binarne

Dwa wyjścia:

$$S = \{00, 01, 10, 11\},$$

Trzy wyjścia:

$$S = \{000, 001, 010, 011, 100, 101, 110, 111\},$$

Źródło binarne

Dwa wyjścia:

$$S = \{00, 01, 10, 11\},$$

Trzy wyjścia:

$$S = \{000, 001, 010, 011, 100, 101, 110, 111\},$$

$$\ldots$$

itd.

Jeżeli dane jest źródło o zbiorze elementów

$$S = \{s_1, s_2, \ldots, s_n\},$$

i k wyjściach, to jako jedno wyjście źródła S można traktować ciąg k wyjść wziętych łącznie.

Jeżeli dane jest źródło o zbiorze elementów

$$S = \{s_1, s_2, \ldots, s_n\},$$

i k wyjściach, to jako jedno wyjście źródła S można traktować ciąg k wyjść wziętych łącznie.

Mamy wtedy n^k takich ciągów wyjściowych.

Definicja

Niech S będzie bezpamięciowym źródłem wiadomości

$$S = \{s_1, s_2, \ldots, s_n\}$$

oraz niech prawdopodobieństwo wiadomości s_i będzie równe p_i.

Definicja

Niech S będzie bezpamięciowym źródłem wiadomości

$$S = \{s_1, s_2, \ldots, s_n\}$$

oraz niech prawdopodobieństwo wiadomości s_i będzie równe p_i.

k-krotnym rozszerzeniem źródła S jest bezpamięciowe źródło S^k o zbiorze zawierającym n^k elementów

$$\{\sigma_1, \sigma_2, \ldots, \sigma_{n^k}\},$$

przy czym każdemu z elementów σ_i odpowiada k-elementowy ciąg wiadomości s_i wytwarzanych przez S

Definicja c.d.

Prawdopodobieństwo wiadomości σ_i: $\Pr(\sigma_i)$ jest prawdopodobieństwem wytworzenia przez S^k, określonego k-elementowego ciągu

$$\sigma_i = (s_{i_1}, s_{i_2}, \ldots, s_{i_k}),$$

czyli

$$\Pr(\sigma_i) = p_{i_1} p_{i_2} \cdots p_{i_k}.$$

Entropia $H\left(S^k\right)$

Przypomnienie (repetitio est mater studiorum):

$$\begin{aligned} H(S) &= p_1 I(s_1) + p_2 I(s_2) + \cdots + p_n I(s_n) \\ &= -\left(p_1 \log p_1 + p_2 \log p_2 + \cdots + p_n \log p_n\right) \\ &= -\sum_{i=1}^{n} p_i \log p_i = -\sum_{S} p_i \log p_i \end{aligned}$$

Entropia $H\left(S^k\right)$

Przypomnienie (repetitio est mater studiorum):

$$\begin{aligned} H(S) &= p_1 I(s_1) + p_2 I(s_2) + \cdots + p_n I(s_n) \\ &= -\left(p_1 \log p_1 + p_2 \log p_2 + \cdots + p_n \log p_n\right) \\ &= -\sum_{i=1}^{n} p_i \log p_i = -\sum_{S} p_i \log p_i \end{aligned}$$

Twierdzenie

$$H\left(S^k\right) = kH(S).$$

Entropia $H\left(S^k\right)$

Przypomnienie (repetitio est mater studiorum):

$$\begin{aligned} H(S) &= p_1 I(s_1) + p_2 I(s_2) + \cdots + p_n I(s_n) \\ &= -\left(p_1 \log p_1 + p_2 \log p_2 + \cdots + p_n \log p_n\right) \\ &= -\sum_{i=1}^{n} p_i \log p_i = -\sum_{S} p_i \log p_i \end{aligned}$$

Twierdzenie

$$H\left(S^k\right) = kH(S).$$

Dowód łatwy, ale rachunkowy i dość długi.

Entropia $H\left(S^k\right)$

Przypomnienie (repetitio est mater studiorum):

$$\begin{aligned} H(S) &= p_1 I(s_1) + p_2 I(s_2) + \cdots + p_n I(s_n) \\ &= -\left(p_1 \log p_1 + p_2 \log p_2 + \cdots + p_n \log p_n\right) \\ &= -\sum_{i=1}^{n} p_i \log p_i = -\sum_{S} p_i \log p_i \end{aligned}$$

Twierdzenie

$$H\left(S^k\right) = kH(S).$$

Dowód łatwy, ale rachunkowy i dość długi. Pominiemy go.

Przykład

Rozszerzymy źródło z wykładu 1.

Przykład

Rozszerzymy źródło z wykładu 1.

$$S = \{s_1, s_2, s_3\}$$

oraz

$$p_1 = \frac{1}{2}\,,\ p_2 = p_3 = \frac{1}{4}\,.$$

Entropia

$$H(S) = \frac{1}{2}\log_2 2 + \frac{1}{4}\log_2 4 + \frac{1}{4}\log_2 4 = \frac{1}{2} + \frac{1}{4}\cdot 2 + \frac{1}{4}\cdot 2 = \frac{3}{2}\,.$$

Przykład c.d.

S^2	σ_1	σ_2	σ_3	σ_4	σ_5	σ_6	σ_7	σ_8	σ_9
S	s_1s_1	s_1s_2	s_1s_3	s_2s_1	s_2s_2	s_2s_3	s_3s_1	s_3s_2	s_3s_3
$\Pr(\sigma_i)$	$\frac{1}{4}$	$\frac{1}{8}$	$\frac{1}{8}$	$\frac{1}{8}$	$\frac{1}{16}$	$\frac{1}{16}$	$\frac{1}{8}$	$\frac{1}{16}$	$\frac{1}{16}$

Przykład c.d.

S^2	σ_1	σ_2	σ_3	σ_4	σ_5	σ_6	σ_7	σ_8	σ_9
S	s_1s_1	s_1s_2	s_1s_3	s_2s_1	s_2s_2	s_2s_3	s_3s_1	s_3s_2	s_3s_3
$\Pr(\sigma_i)$	$\frac{1}{4}$	$\frac{1}{8}$	$\frac{1}{8}$	$\frac{1}{8}$	$\frac{1}{16}$	$\frac{1}{16}$	$\frac{1}{8}$	$\frac{1}{16}$	$\frac{1}{16}$

$$H(S^2) = \frac{1}{4}\log_2 4 + 4 \cdot \frac{1}{8}\log_2 8 + 4 \cdot \frac{1}{16}\log_2 16$$
$$= 3 = 2H(s).$$

Przykład c.d.

S^2	σ_1	σ_2	σ_3	σ_4	σ_5	σ_6	σ_7	σ_8	σ_9
S	s_1s_1	s_1s_2	s_1s_3	s_2s_1	s_2s_2	s_2s_3	s_3s_1	s_3s_2	s_3s_3
$\Pr(\sigma_i)$	$\frac{1}{4}$	$\frac{1}{8}$	$\frac{1}{8}$	$\frac{1}{8}$	$\frac{1}{16}$	$\frac{1}{16}$	$\frac{1}{8}$	$\frac{1}{16}$	$\frac{1}{16}$

$$H(S^2) = \frac{1}{4}\log_2 4 + 4 \cdot \frac{1}{8}\log_2 8 + 4 \cdot \frac{1}{16}\log_2 16$$
$$= 3 = 2H(s).$$

3 bity na wiadomość.

Przykład – liczba π

Liczba

$$\pi = 3.141592653589793\cdots \approx 3.14.$$

jest niewymierna.
Szesnastkowo:

$$\pi = 3.243F6A88\ldots$$

Cyfry dwójkowo, po pominięciu kropki:

$$11001010000111111011010101000100 0$$

Przykład – liczba π

Liczba

$$\pi = 3.141592653589793\cdots \approx 3.14.$$

jest niewymierna.
Szesnastkowo:

$$\pi = 3.243F6A88\ldots$$

Cyfry dwójkowo, po pominięciu kropki:

$$11001010000111111011010101000100 0$$

Liczba π nie jest losowa, więc ciąg bitów jest deterministyczny.

Przykład – liczba π c.d.

Czy źródło produkujące taki ciąg cyfr (bitów) jest bezpamięciowe?

Przykład – liczba π c.d.

Czy źródło produkujące taki ciąg cyfr (bitów) jest bezpamięciowe?

Czy można taki ciąg traktować jako losowy?

Przykład – liczba π c.d.

Czy źródło produkujące taki ciąg cyfr (bitów) jest bezpamięciowe?

Czy można taki ciąg traktować jako losowy?

Ponieważ taki ciąg cyfr jest rozwinięciem (dwójkowym) liczby niewymiernej, to po jedynce w dowolnie wybranym, ale nieznanym miejscu, może wystąpić albo zero albo jedynka.
Podobnie po zerze.

Przykład – liczba π c.d.

Czy źródło produkujące taki ciąg cyfr (bitów) jest bezpamięciowe?

Czy można taki ciąg traktować jako losowy?

Ponieważ taki ciąg cyfr jest rozwinięciem (dwójkowym) liczby niewymiernej, to po jedynce w dowolnie wybranym, ale nieznanym miejscu, może wystąpić albo zero albo jedynka.
Podobnie po zerze.
Po jedynce (zerze) w losowo wybranym miejscu występuje jedynka (zero) z takim samym prawdopodobieństwem 1/2.

Przykład – liczba π c.d.

Czy źródło produkujące taki ciąg cyfr (bitów) jest bezpamięciowe?

Czy można taki ciąg traktować jako losowy?

Ponieważ taki ciąg cyfr jest rozwinięciem (dwójkowym) liczby niewymiernej, to po jedynce w dowolnie wybranym, ale nieznanym miejscu, może wystąpić albo zero albo jedynka.
Podobnie po zerze.
Po jedynce (zerze) w losowo wybranym miejscu występuje jedynka (zero) z takim samym prawdopodobieństwem $1/2$.
Można więc taki ciąg uważać za pochodzący ze źródła bezpamięciowego, w którym $p_0 = p_1 = 1/2$.

Przykład – flip–flop

Załóżmy, że źródło informacji ma tylko dwa stany: 0 i 1.
Każdy kolejny impuls na wejściu powoduje zmianę stanu na wyjściu.
Inaczej mówiąc, ciąg impulsów na wejściu produkuje ciąg zer i jedynek na wyjściu:

$$01010101\ldots$$

albo

$$10101010\ldots$$

Przykład – flip–flop

Załóżmy, że źródło informacji ma tylko dwa stany: 0 i 1.
Każdy kolejny impuls na wejściu powoduje zmianę stanu na wyjściu.
Inaczej mówiąc, ciąg impulsów na wejściu produkuje ciąg zer i jedynek na wyjściu:

$$01010101\ldots$$

albo

$$10101010\ldots$$

Źródło takie na pewno nie jest bezpamięciowe, bo po 1 zawsze następuje 0, a po 0 zawsze następuje 1.

Przykład – flip–flop z losowością

Flip-flop czasem się zacina.

Przykład – flip–flop z losowością

Flip-flop czasem się zacina.

Założenia:

- Po 1 następuje 0 z prawdopodobieństwem 0.99 i 1 z prawdopodobieństwem 0.01.
- Po 0 następuje 1 z prawdopodobieństwem 0.98 i 0 z prawdopodobieństwem 0.02.

Przykład – flip–flop z losowością

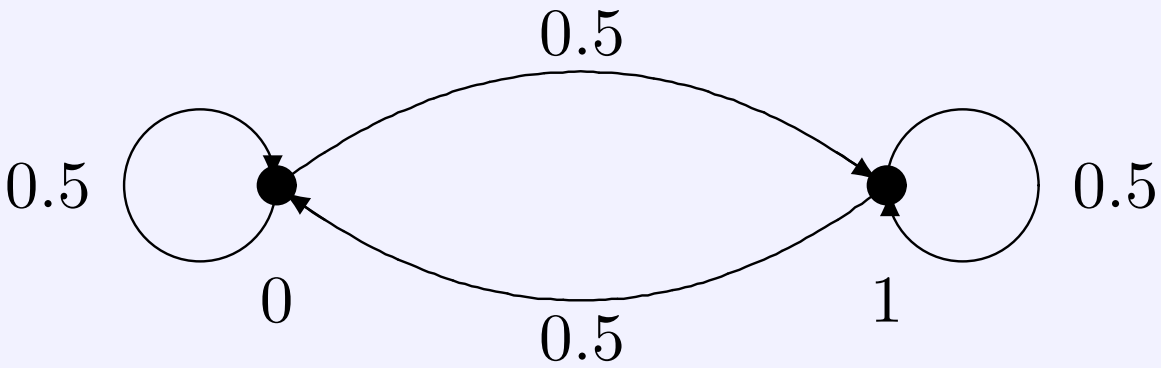
Flip-flop czasem się zacina.

Założenia:

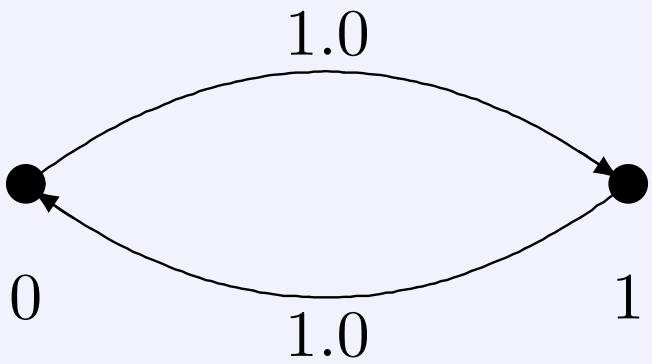
- Po 1 następuje 0 z prawdopodobieństwem 0.99 i 1 z prawdopodobieństwem 0.01.
- Po 0 następuje 1 z prawdopodobieństwem 0.98 i 0 z prawdopodobieństwem 0.02.

Nie jest to źródło bezpamięciowe.

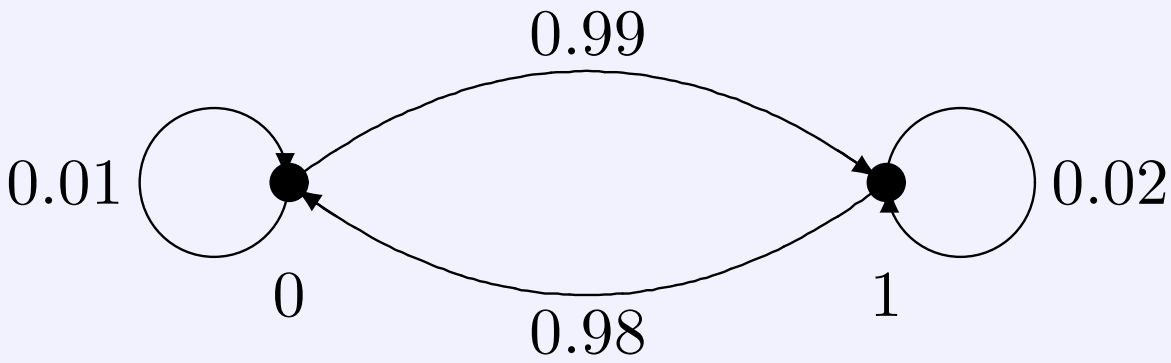
Graf dla liczby π



Graf dla flip–flop



Graf dla flip–flop z losowością



Markow

Andriej Andriejewicz Markow
Андрей Андреевич Марков
Andrey Andreyevich Markov

Matematyk rosyjski.

ur. 14 czerwca 1856 w guberni Riazanskiej,

zm. 20 lipca 1922 w Petersburgu.

Markow

Andriej Andriejewicz Markow
Андрей Андреевич Марков
Andrey Andreyevich Markov

Matematyk rosyjski.

ur. 14 czerwca 1856 w guberni Riazanskiej,

zm. 20 lipca 1922 w Petersburgu.

Od 1886 profesor uniwersytetu w Petersburgu; członek zwyczajny Petersburskiej Akademii Nauk (od 1896). Był uczniem znanego matematyka Pafnutija Czebyszewa. Jego najważniejsze dokonania to prace z teorii prawdopodobieństwa – zapoczątkował w niej m.in. badania nad ważną w zastosowaniach klasą procesów stochastycznych, zwanych procesami Markowa.

Ciągi Markowa

Niech S będzie źródłem tworzącym wiadomości w taki sposób, że prawdopodobieństwo utworzenia wiadomości x_i zależy od wiadomości poprzedniej x_{i-1}, ale nie zależy wiadomości x_{n-2} i jeszcze wcześniejszych.

Ciągi Markowa

Niech S będzie źródłem tworzącym wiadomości w taki sposób, że prawdopodobieństwo utworzenia wiadomości x_i zależy od wiadomości poprzedniej x_{i-1}, ale nie zależy wiadomości x_{n-2} i jeszcze wcześniejszych.

Ciąg takich wiadomości jest ciągiem Markowa.

Ciągi Markowa

Niech S będzie źródłem tworzącym wiadomości w taki sposób, że prawdopodobieństwo utworzenia wiadomości x_i zależy od wiadomości poprzedniej x_{i-1}, ale nie zależy wiadomości x_{n-2} i jeszcze wcześniejszych.

Ciąg takich wiadomości jest ciągiem Markowa.

Ciąg „flip–flop z losowością" jest ciągiem Markowa.

Ciągi Markowa

Niech S będzie źródłem tworzącym wiadomości w taki sposób, że prawdopodobieństwo utworzenia wiadomości x_i zależy od wiadomości poprzedniej x_{i-1}, ale nie zależy wiadomości x_{n-2} i jeszcze wcześniejszych.

Ciąg takich wiadomości jest ciągiem Markowa.

Ciąg „flip–flop z losowością" jest ciągiem Markowa.

Prawdopodobieństwo pojawienia się jedynki zależy od poprzedniej wartości.
Jeśli było to zero, to jedynka pojawi się z prawdopodobieństwem 0.98,
Jeśli była to jedynka, to jedynka pojawi się z prawdopodobieństwem 0.02,

Ciągi Markowa k-tego rzędu

Niech S będzie źródłem tworzącym wiadomości w taki sposób, że prawdopodobieństwo utworzenia wiadomości x_i zależy od k wiadomości poprzednich $x_{i-1}, \ldots, x_{i-k}$, ale nie zależy wiadomości x_{n-k-1} i jeszcze wcześniejszych.

Ciągi Markowa k-tego rzędu

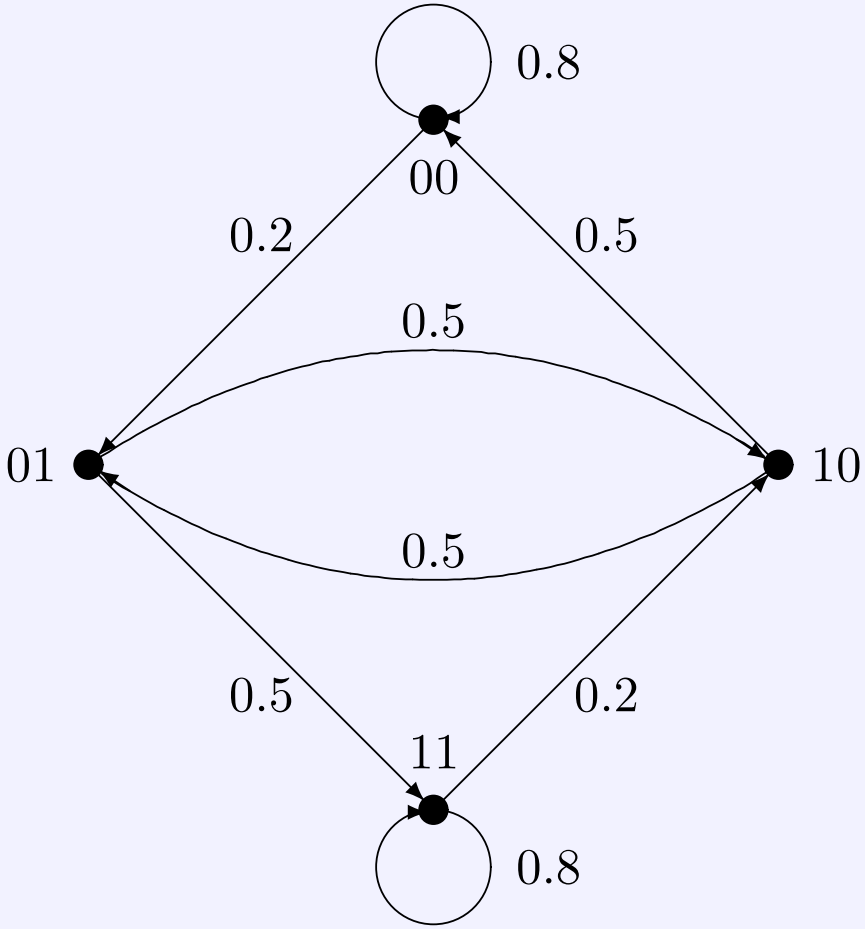
Niech S będzie źródłem tworzącym wiadomości w taki sposób, że prawdopodobieństwo utworzenia wiadomości x_i zależy od k wiadomości poprzednich $x_{i-1}, \ldots, x_{i-k}$, ale nie zależy wiadomości x_{n-k-1} i jeszcze wcześniejszych.

Ciąg takich wiadomości jest ciągiem Markowa k-tego rzędu.

Źródła ergodyczne

Źródło ergodyczne to takie źródło, które po obserwacji przez długi okres czasu, będzie wytwarzać „typowy” ciąg wiadomości.

Źródła ergodyczne drugiego rzędu – przykład



Rozkład stacjonarny

Częstości (prawdopodobieństwa) zaobserwowane w długim okresie czasu.

Rozkład stacjonarny

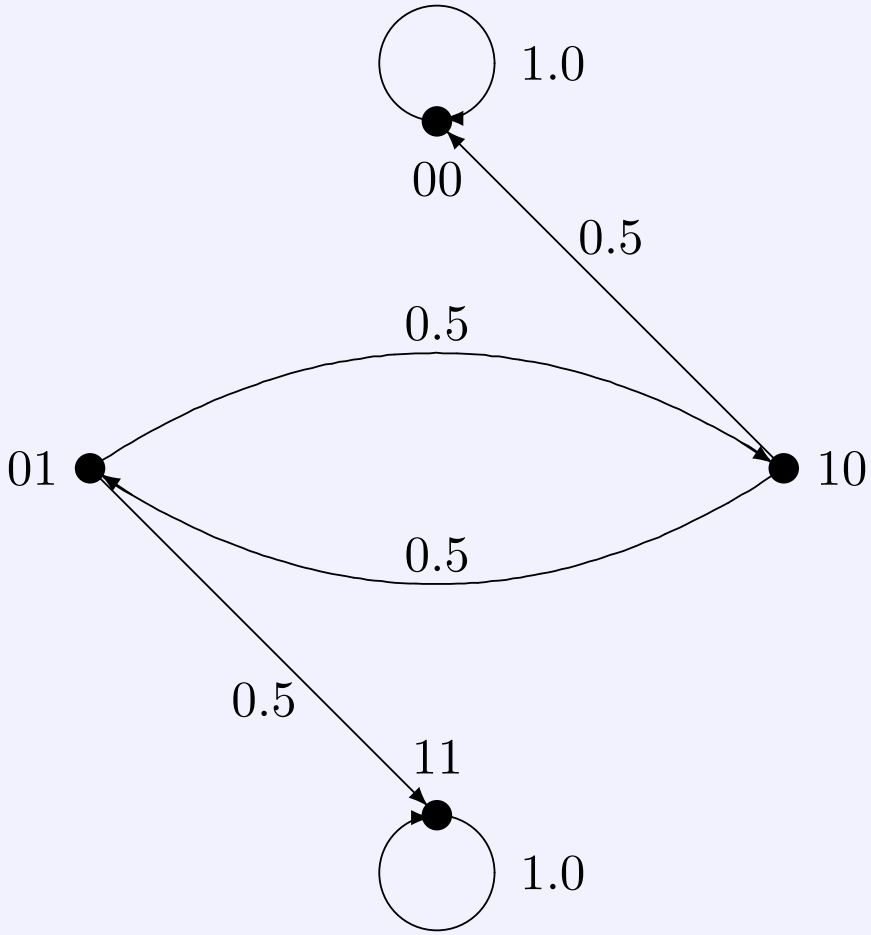
Częstości (prawdopodobieństwa) zaobserwowane w długim okresie czasu.

Numeracja stanów:

Stan	00	01	11	10
Numer	0	1	2	3

$$p_0 = p_2 = \frac{5}{14} \approx 0.3571,$$
$$p_1 = p_3 = \frac{2}{14} \approx 0.1429.$$

Źródła nieergodyczne drugiego rzędu – przykład



Symulacje – program w Pascalu

```
procedure next;
var p:double;
  tstate:integer;
begin
  p:=random;
  tstate:=currstate;
  if p<prob[currstate] then
  begin
    inc(tstate);
    tstate:=tstate mod 4;
  end else
  begin
    if currstate=1 then tstate:=3;
    if currstate=3 then tstate:=1;
  end;
  currstate:=tstate;
  inc(state[currstate]);
end;
```

Zadanie 1

Przetłumaczyć program na inny język (inne języki), np. Python, C++ itp.

Zadanie 1

Przetłumaczyć program na inny język (inne języki), np. Python, C++ itp.

Wykonać symulacje.

Symulacje – wyniki

Start ze stanu 00.

Symulacje – wyniki

Start ze stanu 00.

Stan	$n = 100$	$n = 1000$	$n = 10000$	p_i
00	0.340	0.385	0.355	0.3571
01	0.140	0.131	0.139	0.1429
11	0.390	0.354	0.367	0.3571
10	0.130	0.130	0.139	0.1429

Symulacje – wyniki

Start ze stanu 00.

Stan	$n = 100$	$n = 1000$	$n = 10000$	p_i
00	0.340	0.385	0.355	0.3571
01	0.140	0.131	0.139	0.1429
11	0.390	0.354	0.367	0.3571
10	0.130	0.130	0.139	0.1429

Powtórzenie symulacji:

Stan	$n = 100$	$n = 1000$	$n = 10000$	p_i
00	0.390	0.318	0.358	0.3571
01	0.140	0.149	0.143	0.1429
11	0.336	0.384	0.357	0.3571
10	0.140	0.149	0.143	0.1429

Start z innych stanów

Start z innych stanów daje wyniki analogiczne dla dużych n.

Dla małych n wyniki symulacji będą istotnie zależeć od punktu startu.

Start z innych stanów

Start z innych stanów daje wyniki analogiczne dla dużych n.

Dla małych n wyniki symulacji będą istotnie zależeć od punktu startu.

Ważne

Do samodzielnych prób!

Stany N i F (1)

Urządzenie może być w stanie normalnym N i w stanie awaryjnym F. Urządzenie przechodzi ze stanu do stanu z prawdopodobieństwami podanymi wzorem:

$$p_{NN} = 0.98$$
$$p_{NF} = 0.02$$
$$p_{FN} = 0.98$$
$$p_{FF} = 0.02$$

Inaczej mówiąc, w pracującym urządzeniu rzadko następuje awaria, to znaczy że jeśli otrzymamy wiadomość, że urządzenie pracuje, to prawdopodobieństwo, że następną będzie informacja o awarii jest mała i wynosi 0.02. Jeśli wiadomo, że urządzenie nie pracuje, to prawdopodobieństwo, że następną będzie informacja o pozostawaniu w stanie awarii również jest mała i wynosi 0.02.

Stany N i F (2)

Przyjmijmy teraz

$$p_{NN} = 0.99$$
$$p_{NF} = 0.01$$
$$p_{FN} = 0.49$$
$$p_{FF} = 0.51$$

Jest to sytuacja taka, że urządzenie rzadziej się psuje, ale popsute, z prawdopodobieństwem mniejszym (bo około połowy), przechodzi do stanu normalnego w momencie wysłania następnej wiadomości.

Entropia dla źródeł z pamięcią

Entropię określa się również dla ciągów Markowa i dla źródeł z pamięcią.

Są to jednak zagadnienia znacznie bardziej zaawansowane i muszą być tutaj pominięte. Zainteresowanym tym zagadnieniem można polecić książkę:
N. Abramson, *Teoria informacji i kodowania*, PWN 1969.

Zadanie 2

Wymyślić bliski rzeczywistości model źródła bez pamięci, które może wysłać n różnych wiadomości z danymi prawdopodobieństwami.

Podać dla nich kilka różnych sposobów kodowania.

Obliczyć entropię źródła.

Podać średnią długość słowa kodowego.

Zadanie 3

Rzucamy monetą i kostką do gry w następujący sposób:

- Jeżeli przy rzucie monetą wypadnie orzeł, to następny raz też rzucamy monetą. Jeżeli wypadnie reszka, to następny raz rzucamy kostką do gry.
- Jeżeli przy rzucie kostką wypadnie szóstka, to następny raz też rzucamy kostką. Jeżeli wypadnie inna liczba oczek, to następny raz rzucamy monetą.

Stanami są:

$$\{O, R, 1, 2, 3, 4, 5, 6\}$$

Narysować graf przejść ze stanu do stanu i podać prawdopodobieństwa przejść.

Zadanie 3

Rzucamy monetą i kostką do gry w następujący sposób:

- Jeżeli przy rzucie monetą wypadnie orzeł, to następny raz też rzucamy monetą. Jeżeli wypadnie reszka, to następny raz rzucamy kostką do gry.
- Jeżeli przy rzucie kostką wypadnie szóstka, to następny raz też rzucamy kostką. Jeżeli wypadnie inna liczba oczek, to następny raz rzucamy monetą.

Stanami są:

$$\{O, R, 1, 2, 3, 4, 5, 6\}$$

Narysować graf przejść ze stanu do stanu i podać prawdopodobieństwa przejść.

Napisać procedurę w dowolnym języku, symulującą taki ciąg rzutów i wyznaczającą częstości wystąpień stanów w długich seriach rzutów.

Rozwiązania

Zadania są nieobowiązkowe, ale ich rozwiązanie jest premiowane dodatkowymi punktami przy zaliczeniu.

Rozwiązane zadania: rysunek, kod źródłowy programu i wyniki liczbowe, proszę przesłać e-mailem na adres:

wojciech.kordecki@collegiumwiteloma.pl

Rozwiązania

Zadania są nieobowiązkowe, ale ich rozwiązanie jest premiowane dodatkowymi punktami przy zaliczeniu.

Rozwiązane zadania: rysunek, kod źródłowy programu i wyniki liczbowe, proszę przesłać e-mailem na adres:

wojciech.kordecki@collegiumwiteloma.pl

Termin do końca 6 listopada.

Kod nie do odkodowania

Wiadomość	Kod
s_1	0
s_2	01
s_3	001
s_4	111

Kod nie do odkodowania

Wiadomość	Kod
s_1	0
s_2	01
s_3	001
s_4	111

Odkodowanie ciągu 111001:

$$(111)(001) \Rightarrow s_4 s_3$$

albo

$$(111)(0)(01) \Rightarrow s_4 s_1 s_2 \,.$$

Przykład z wykładu 1

Cztery stany: stan normalny N i jeden z trzech stanów awaryjnych A, B, C.

Kodowanie 1: zawsze dwa bity:

N	00
A	01
B	10
C	11

Kodowanie 2: jeden lub trzy bity:

N	0
A	101
B	110
C	111

Przykład z wykładu 1

Cztery stany: stan normalny N i jeden z trzech stanów awaryjnych A, B, C.

Kodowanie 1: zawsze dwa bity:

N	00
A	01
B	10
C	11

Kodowanie 2: jeden lub trzy bity:

N	0
A	101
B	110
C	111

Żaden kod znaku w kodowaniu 1 i 2 nie jest *prefixem*, czyli początkową częścią kodu żadnego innego znaku.

Przykład z wykładu 1

Cztery stany: stan normalny N i jeden z trzech stanów awaryjnych A, B, C.

Kodowanie 1: zawsze dwa bity:

N	00
A	01
B	10
C	11

Kodowanie 2: jeden lub trzy bity:

N	0
A	101
B	110
C	111

Żaden kod znaku w kodowaniu 1 i 2 nie jest *prefixem*, czyli początkową częścią kodu żadnego innego znaku.

Jest to więc kod prefiksowy

Przykład

Kodowanie 3: jeden, dwa lub trzy bity:

N	0
A	01
B	001
C	111

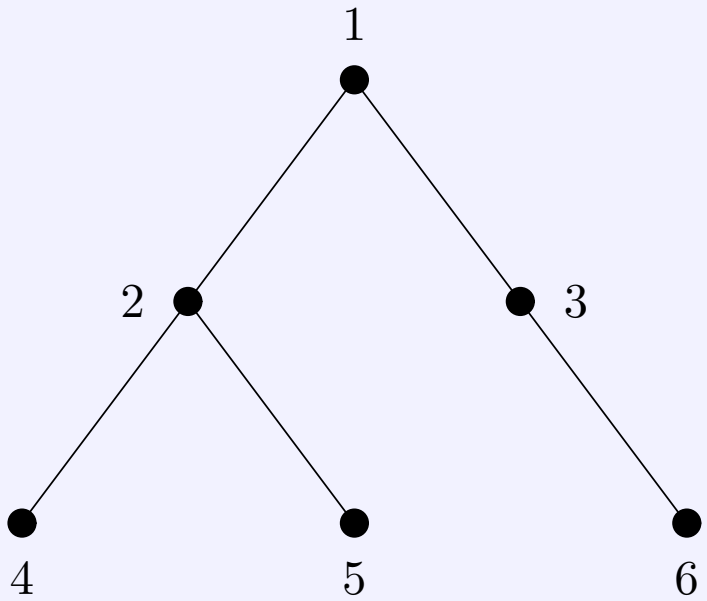
Przykład

Kodowanie 3: jeden, dwa lub trzy bity:

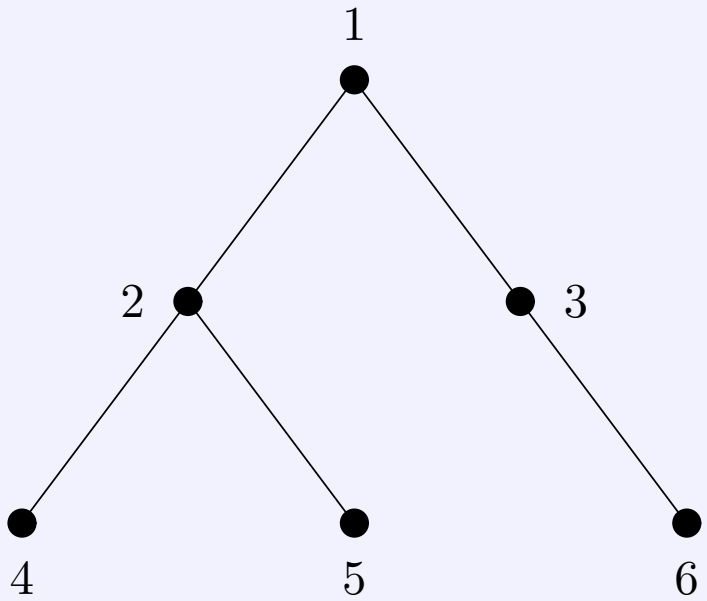
N	0
A	01
B	001
C	111

W kodowaniu 3, kod znaku N jest prefiksem znaków A i B: **0**1 i **0**01,

Drzewo binarne



Drzewo binarne



- $\{1\}$ – korzeń,
- $\{2, 3\}$ – potomkowie (lewy, prawy) węzła $\{1\}$,
- $\{4, 5\}$ – potomkowie (lewy, prawy) węzła $\{2\}$,
- $\{6\}$ – (prawy) potomek węzła $\{3\}$,
- $\{4, 5, 6\}$ – liście.

Definicja kodu Huffmana

Z symboli utworzony jest ciąg skończony tekst. Słowa kodowe o długości k bitów każde, wystarczają do zakodowania 2^k symboli.
Każdemu symbolowi przypisujemy $s_i \in S$ przyporządkowujemy prawdopodobieństwo p_i.
Niech b_i będzie liczbą bitów słowa kodującego symbol s_i.
Kodowanie Huffmana polega na utworzeniu słów kodowych o niejednakowej długości w taki sposób, aby średnia

$$W = \sum_{i=1}^{n} b_i p_i$$

była najmniejsza.

Drzewo Huffmana (1)

1. Tworzymy listę drzew binarnych, które w wierzchołkach przechowują pary: (s_i, p_i). Na początku drzewa składają się wyłącznie z korzenia.

Drzewo Huffmana (1)

1. Tworzymy listę drzew binarnych, które w wierzchołkach przechowują pary: (s_i, p_i). Na początku drzewa składają się wyłącznie z korzenia.
2. Jeśli na liście jest więcej niż jedno drzewo, powtarzamy: usuwamy z listy dwa drzewa o najmniejszym prawdopodobieństwie zapisanym w korzeniu. Wstawiamy nowe drzewo, w którego korzeniu jest suma prawdopodobieństw usuniętych drzew, natomiast one same stają się jego lewym i prawym poddrzewem. Korzeń drzewa nie przechowuje symbolu.

Drzewo Huffmana (1)

1. Tworzymy listę drzew binarnych, które w wierzchołkach przechowują pary: (s_i, p_i). Na początku drzewa składają się wyłącznie z korzenia.
2. Jeśli na liście jest więcej niż jedno drzewo, powtarzamy: usuwamy z listy dwa drzewa o najmniejszym prawdopodobieństwie zapisanym w korzeniu. Wstawiamy nowe drzewo, w którego korzeniu jest suma prawdopodobieństw usuniętych drzew, natomiast one same stają się jego lewym i prawym poddrzewem. Korzeń drzewa nie przechowuje symbolu.
3. Drzewo, które pozostanie na liście, jest nazywane drzewem Huffmana – prawdopodobieństwo zapisane w korzeniu jest równe 1, natomiast w liściach drzewa zapisane są symbole.

Drzewo Huffmana (2)

Algorytm Huffmana nie określa, w jakiej kolejności wybierać drzewa z listy, jeśli mają takie samo prawdopodobieństwo.
Nie jest również określone, które z usuwanych drzew ma stać się lewym bądź prawym poddrzewem.
Bez względu na przyjęte rozwiązanie, średnia długośći kodu pozostaje taka sama.

Algorytm

1. Każdej lewej krawędzi drzewa przypisujemy 0, prawej 1 (można oczywiście odwrotnie).

Algorytm

1. Każdej lewej krawędzi drzewa przypisujemy 0, prawej 1 (można oczywiście odwrotnie).
2. Przechodzimy w głąb drzewa od korzenia do każdego liścia (symbolu): jeśli idziemy w prawo, dopisujemy do kodu bit 1; jeśli idziemy w lewo, dopisujemy do kodu bit 0.

Algorytm

1. Każdej lewej krawędzi drzewa przypisujemy 0, prawej 1 (można oczywiście odwrotnie).
2. Przechodzimy w głąb drzewa od korzenia do każdego liścia (symbolu): jeśli idziemy w prawo, dopisujemy do kodu bit 1; jeśli idziemy w lewo, dopisujemy do kodu bit 0.

Kod Huffmana jest kodem prefiksowym.
Jest przykładem kompresji danych.

Odkodowanie

Dekodowanie jest procesem odwrotnym. Od korzenia do liścia idziemy w lewo gdy 0 i w prawo gdy 1.

Odkodowanie

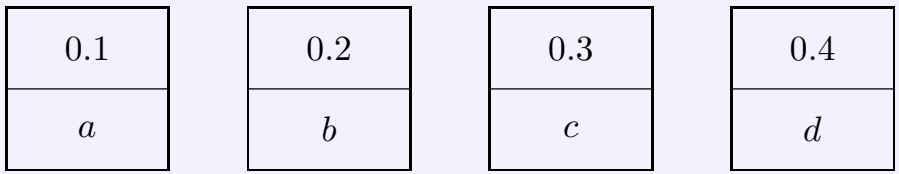
Dekodowanie jest procesem odwrotnym. Od korzenia do liścia idziemy w lewo gdy 0 i w prawo gdy 1.

Do odkodowania konieczna jest znajomość drzewa Huffmana.

Przykład – cztery znaki

Symbol	a	b	c	d
Prawdopodobieństwo	0.1	0.2	0.3	0.4

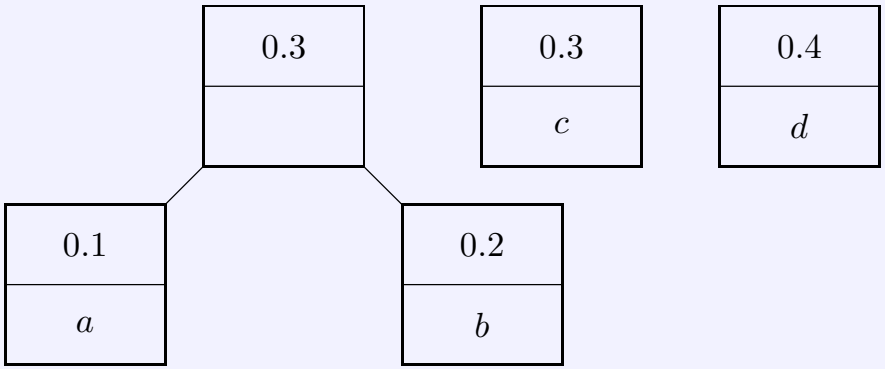
Krok 1



Cztery drzewa – tylko korzenie

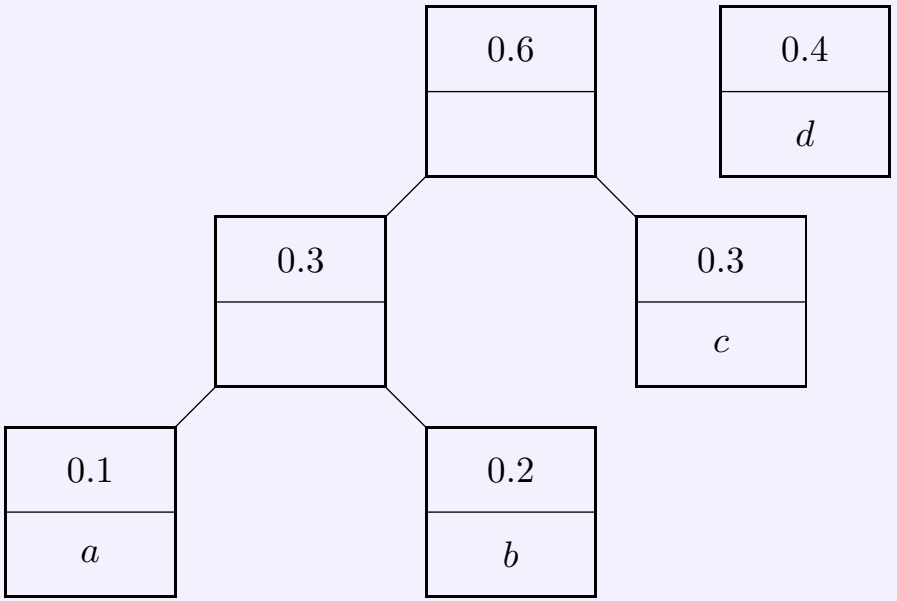
Krok 2

Łączymy korzenie z symbolami a i b.



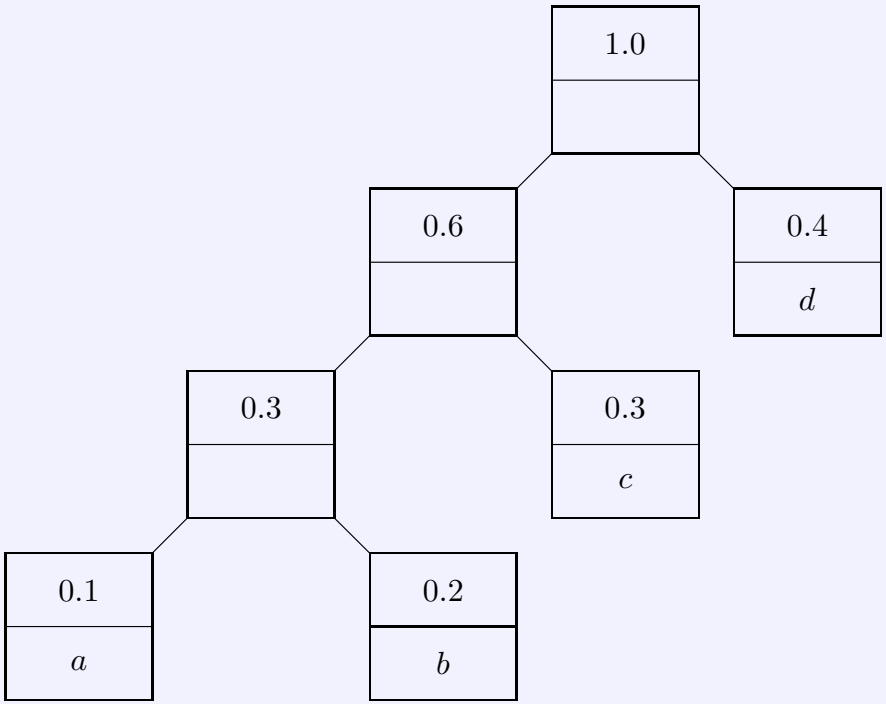
Krok 3

Łączymy korzenie (a, b) i c.



Krok 4

Łączymy korzenie $((a, b), c)$ i d



Kody znaków

Symbol	a	b	c	d
Kod	000	001	01	1

Kody znaków

Symbol	a	b	c	d
Kod	000	001	01	1

Średnia długość słowa kodowego wynosi:

$$L = 3 \cdot 0.1 + 3 \cdot 0.2 + 2 \cdot 0.3 + 1 \cdot 0.4 = 1.9.$$

Kody znaków

Symbol	a	b	c	d
Kod	000	001	01	1

Średnia długość słowa kodowego wynosi:

$$L = 3 \cdot 0.1 + 3 \cdot 0.2 + 2 \cdot 0.3 + 1 \cdot 0.4 = 1.9.$$

Bez kodowania, długość słowa kodowego wynosi dokładnie 2, na przykład:

Symbol	a	b	c	d
Kod	00	01	10	11

Kody znaków

Symbol	a	b	c	d
Kod	000	001	01	1

Średnia długość słowa kodowego wynosi:

$$L = 3 \cdot 0.1 + 3 \cdot 0.2 + 2 \cdot 0.3 + 1 \cdot 0.4 = 1.9.$$

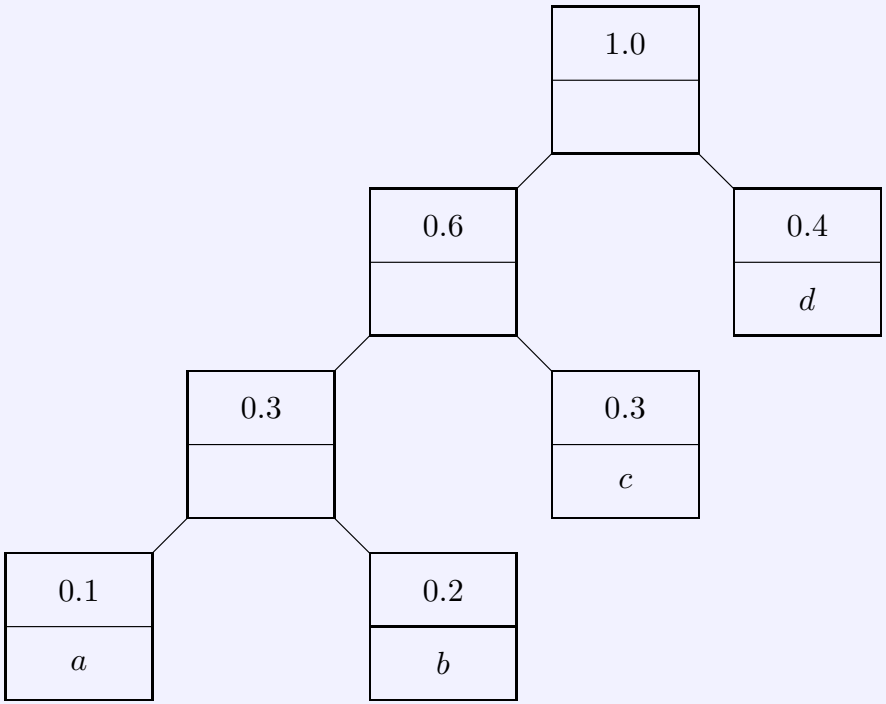
Bez kodowania, długość słowa kodowego wynosi dokładnie 2, na przykład:

Symbol	a	b	c	d
Kod	00	01	10	11

Kompresja do 95%.

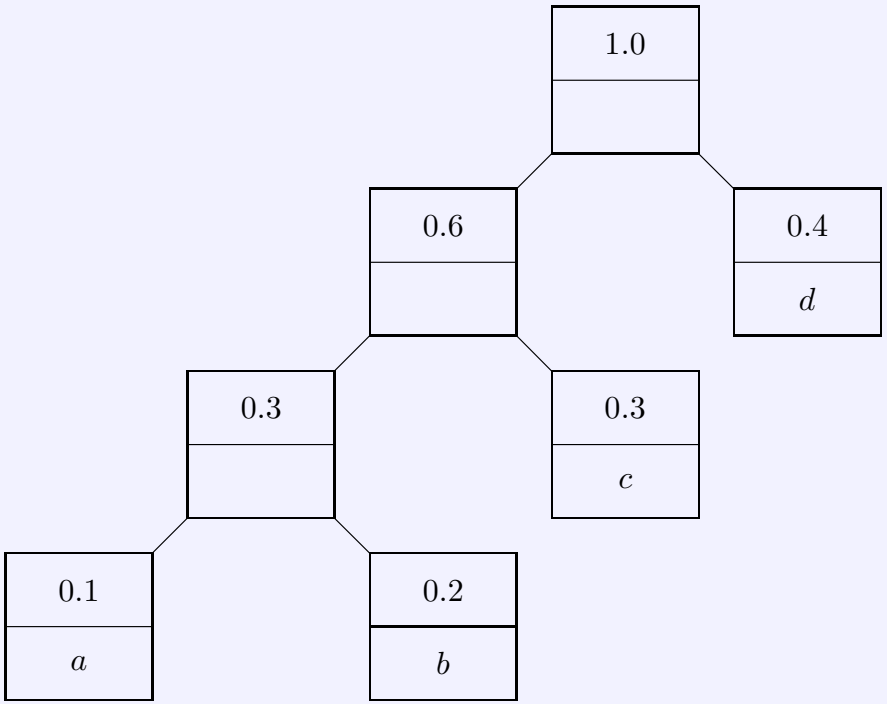
Kodowanie tekstu

cddcabbac



Kodowanie tekstu

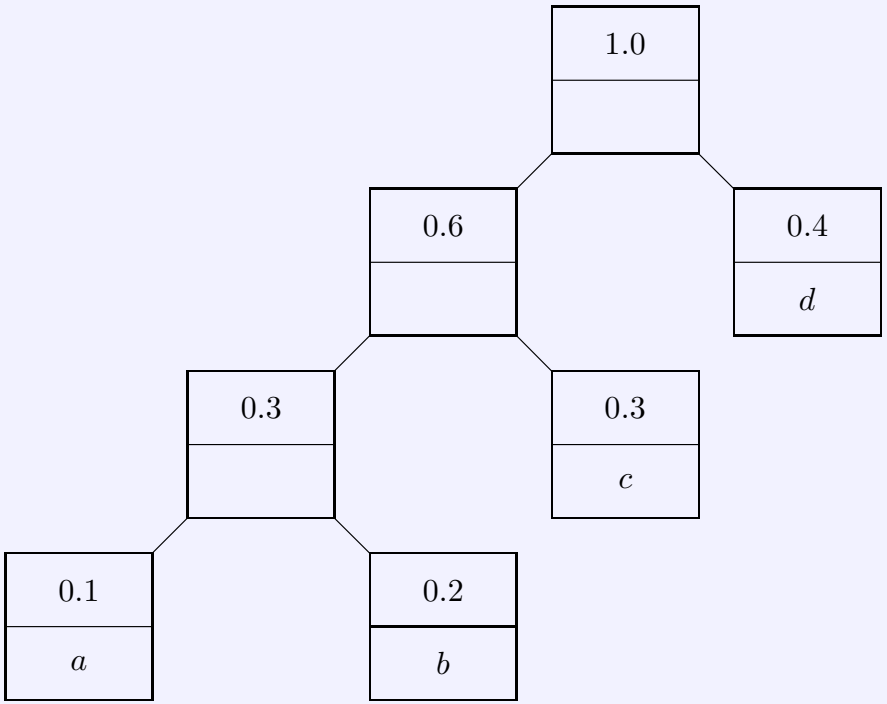
cddcabbac



c 01 |

Kodowanie tekstu

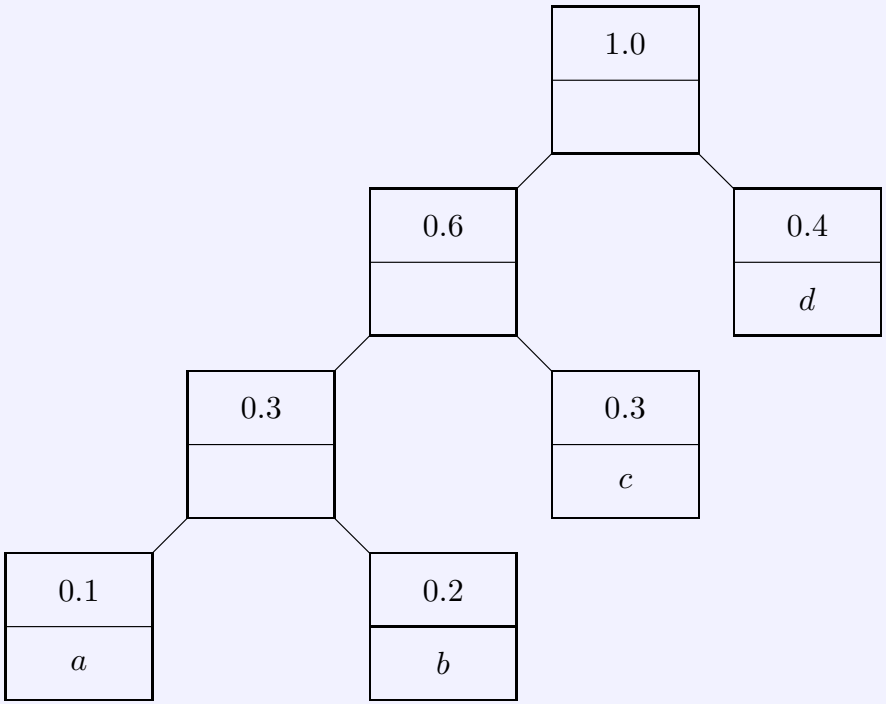
cddcabbac



c 01 | d 1 |

Kodowanie tekstu

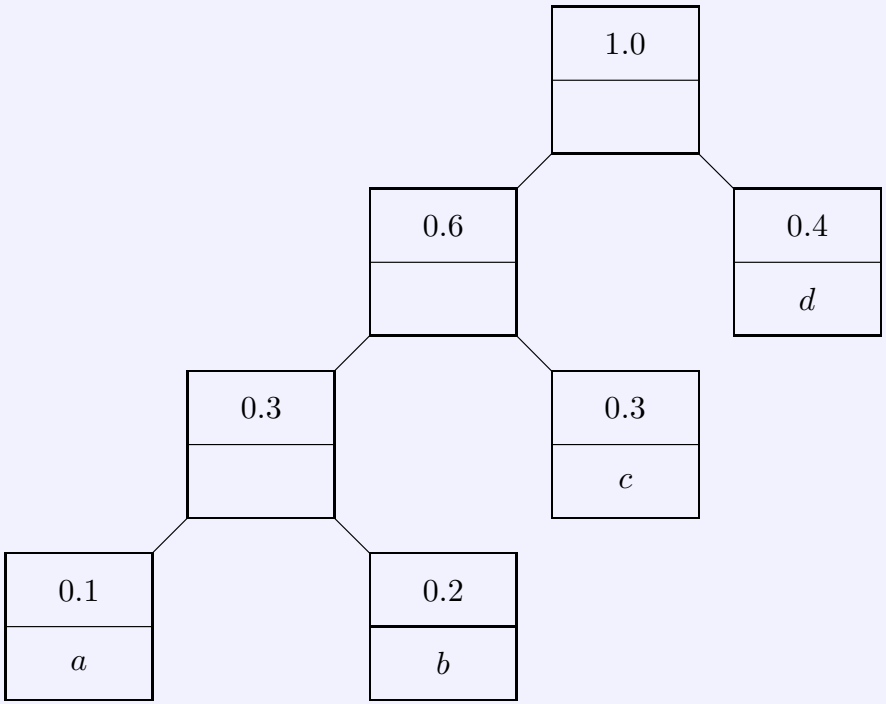
cddcabbac



c 01 | d 1 | d 1 |

Kodowanie tekstu

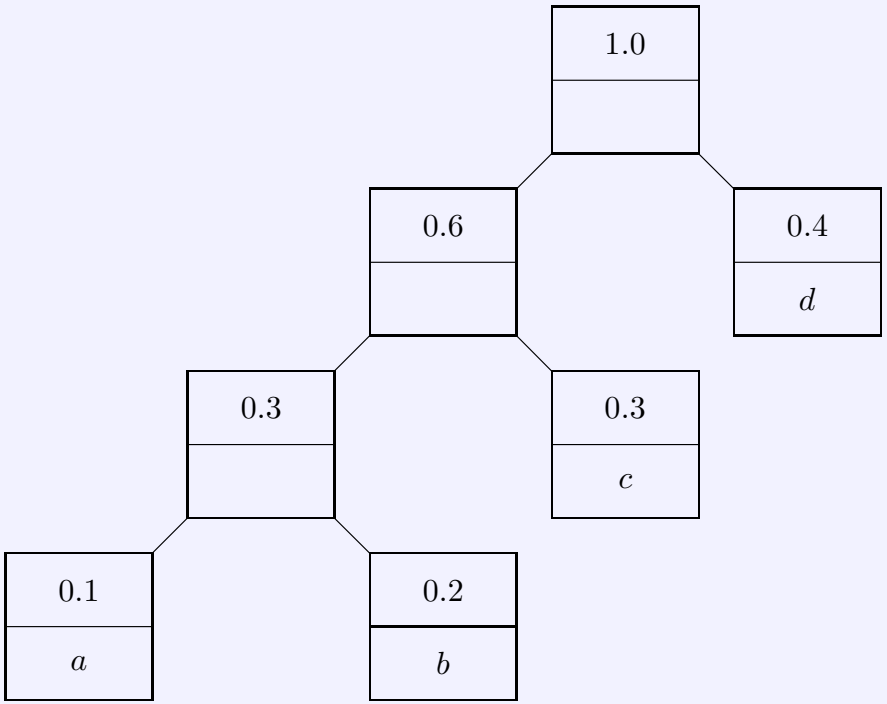
cddcabbac



c 01 | d 1 | d 1 | c 01 |

Kodowanie tekstu

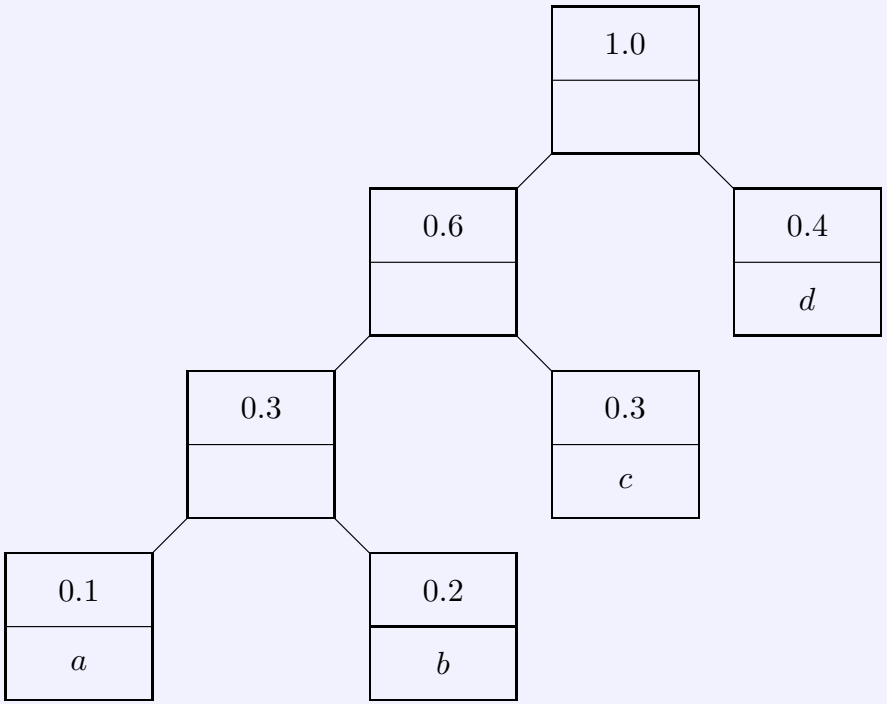
cddcabbac



c 01 | d 1 | d 1 | c 01 | a 000 |

Kodowanie tekstu

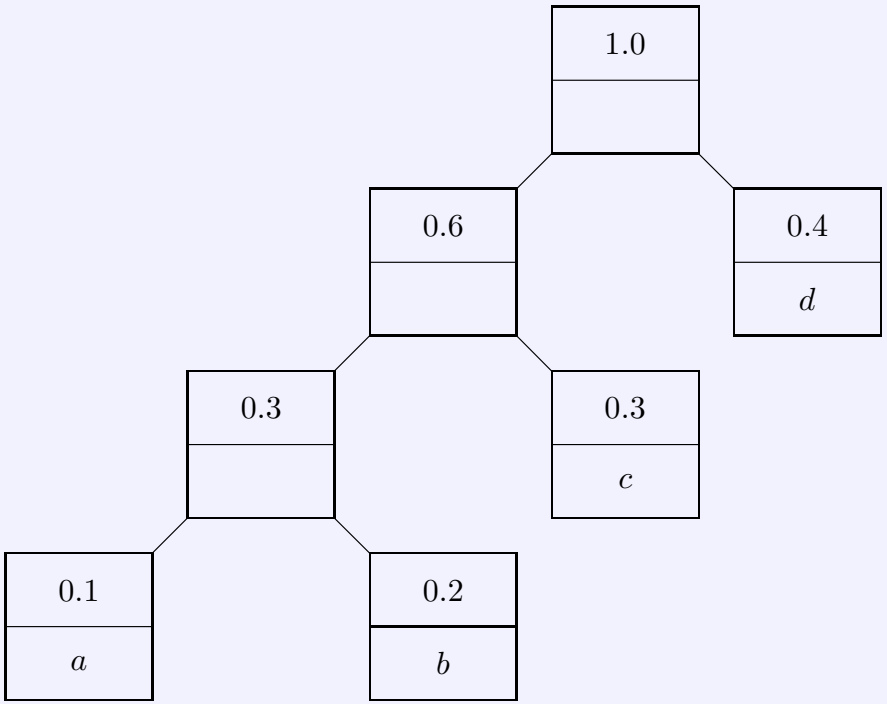
cddcabbac



c 01 | d 1 | d 1 | c 01 | a 000 | b 001 |

Kodowanie tekstu

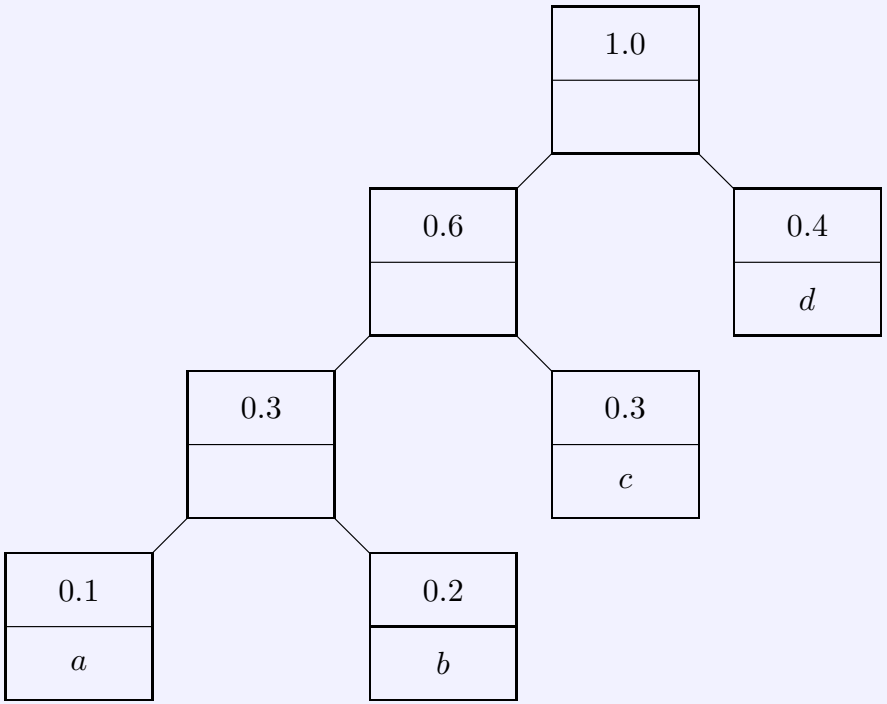
cddcabbac



c 01 | d 1 | d 1 | c 01 | a 000 | b 001 | b 001 |

Kodowanie tekstu

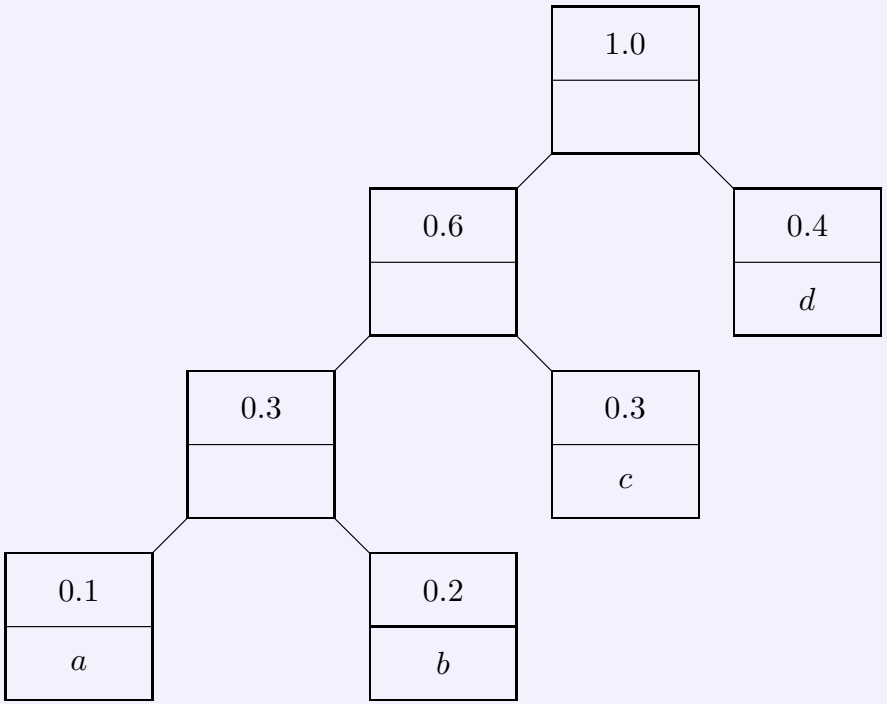
cddcabbac



c 01 | d 1 | d 1 | c 01 | a 000 | b 001 | b 001 | a 000 |

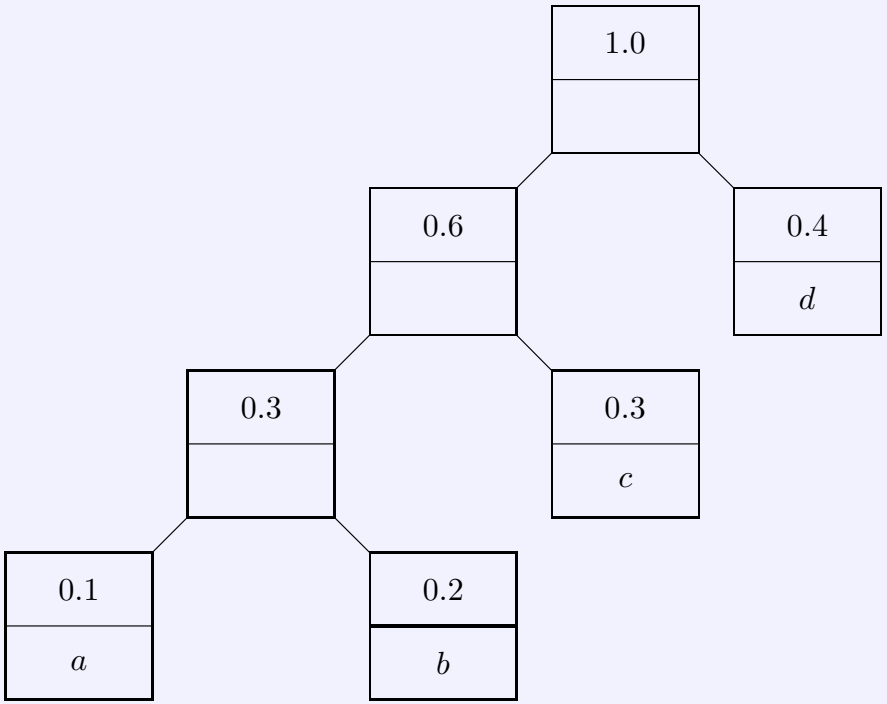
Kodowanie tekstu

cddcabbac



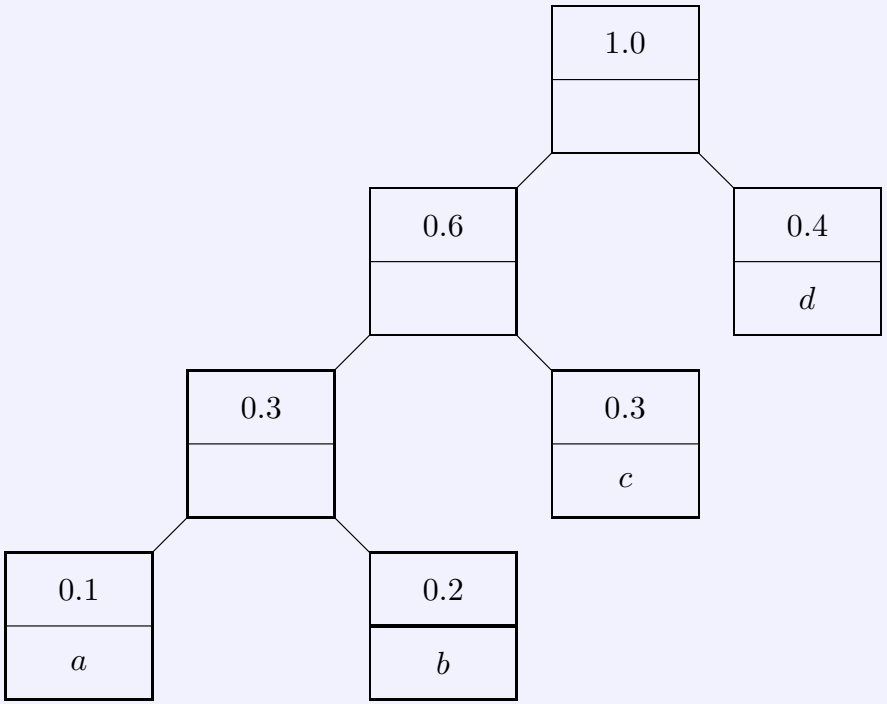
c 01 | d 1 | d 1 | c 01 | a 000 | b 001 | b 001 | a 000 | c 01

Odkodowywanie tekstu



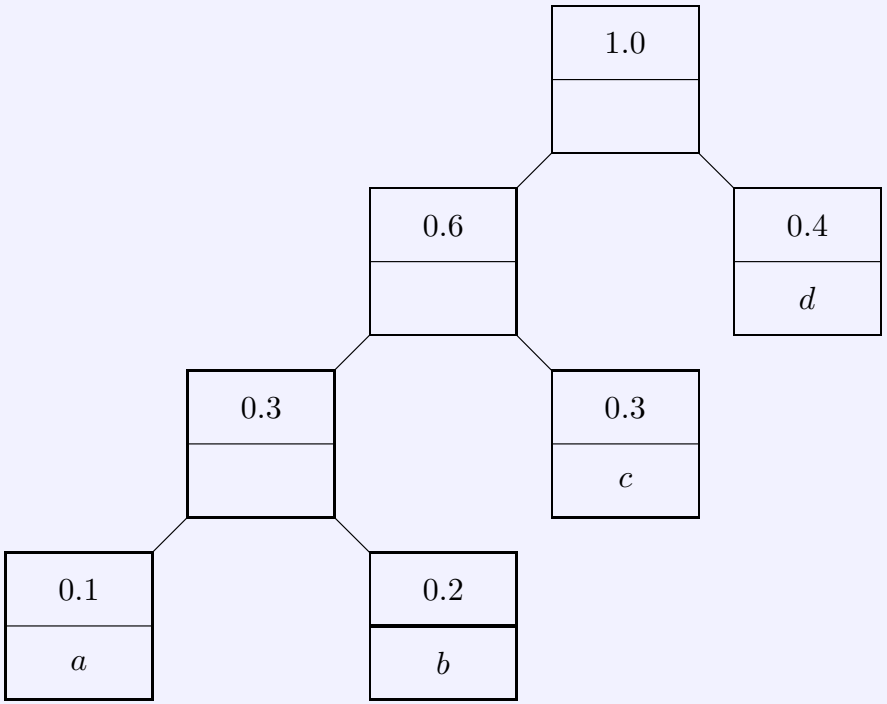
Tekst zakodowany 01110100000100100001

Odkodowywanie tekstu



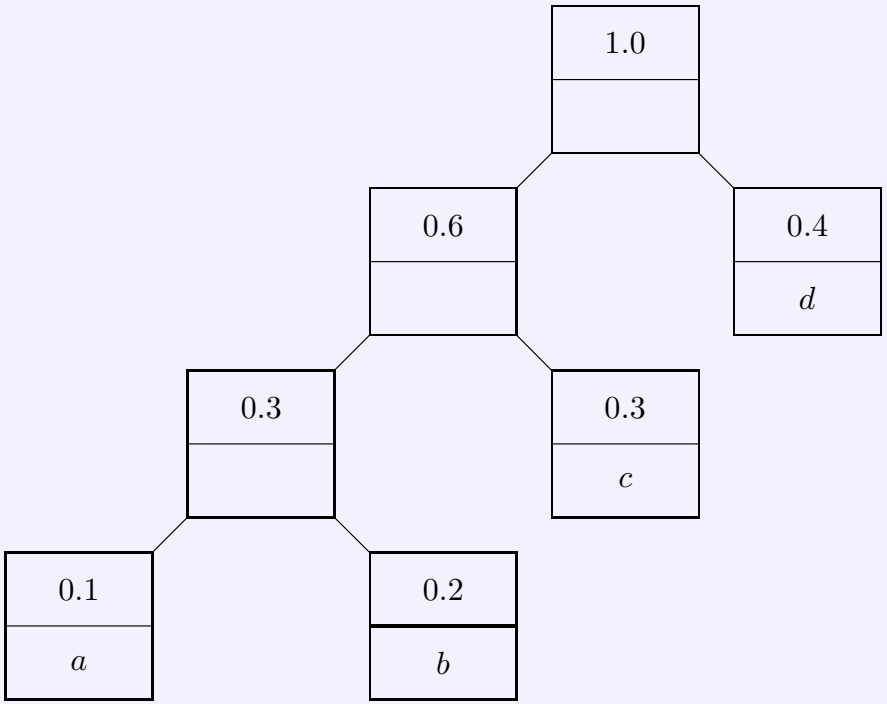
Tekst zakodowany 011101000001001000 01
01 c |

Odkodowywanie tekstu



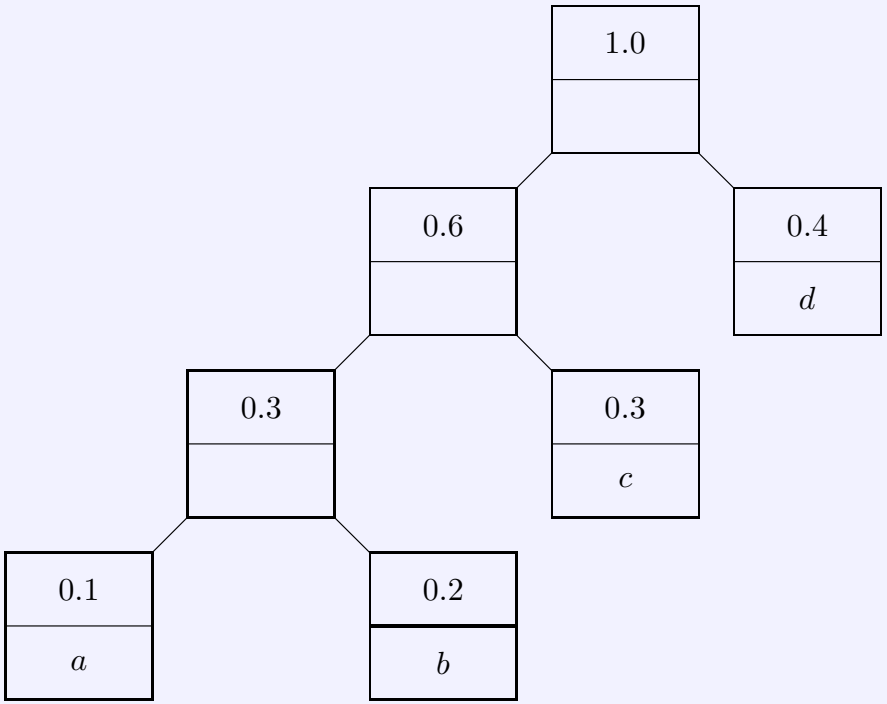
Tekst zakodowany 011101000001001000 01
01 c | 1 d |

Odkodowywanie tekstu



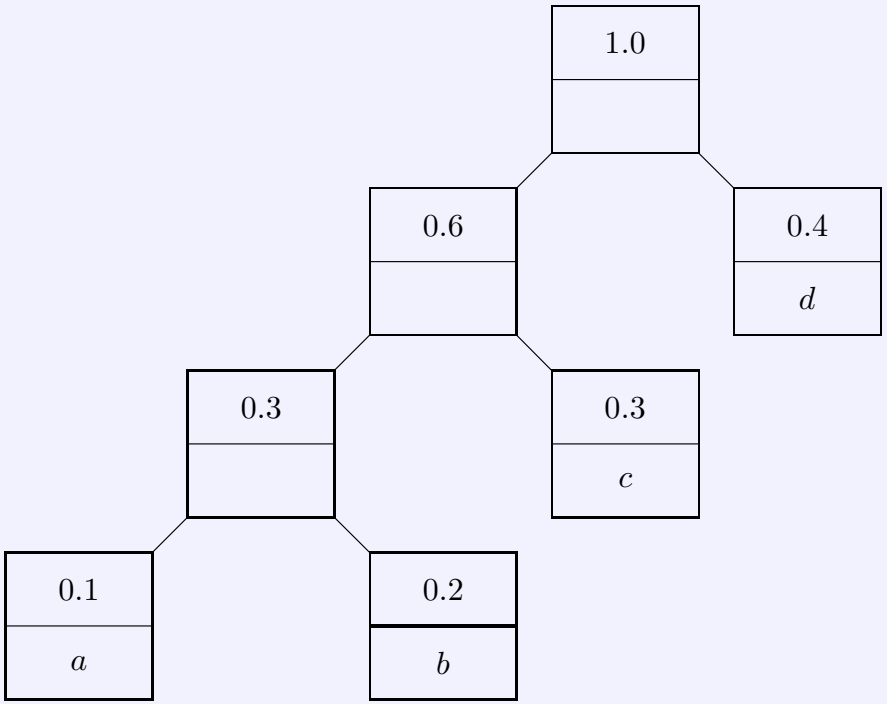
Tekst zakodowany 011101000001001000 01
01 c | 1 d | 1 d |

Odkodowywanie tekstu



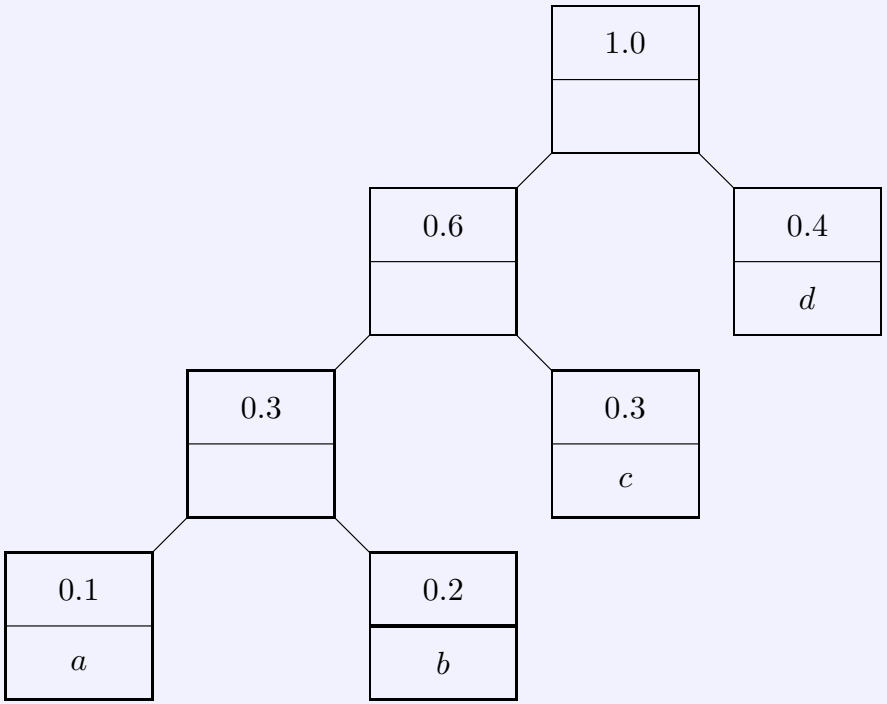
Tekst zakodowany 011101000001001000 01
01 c | 1 d | 1 d | 01 c |

Odkodowywanie tekstu



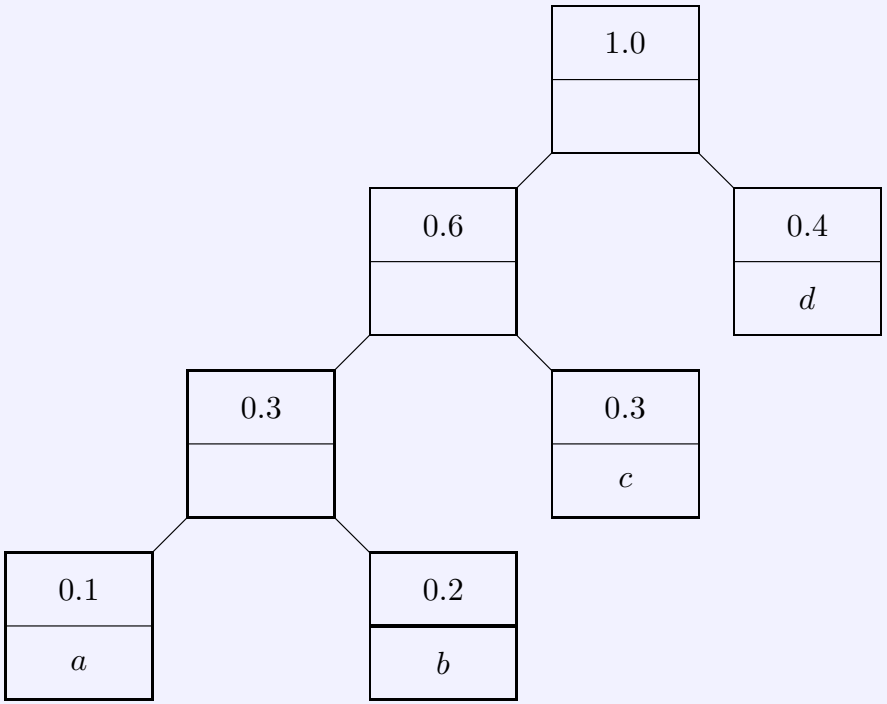
Tekst zakodowany 011101000001001000 01
01 c | 1 d | 1 d | 01 c | 000 a |

Odkodowywanie tekstu



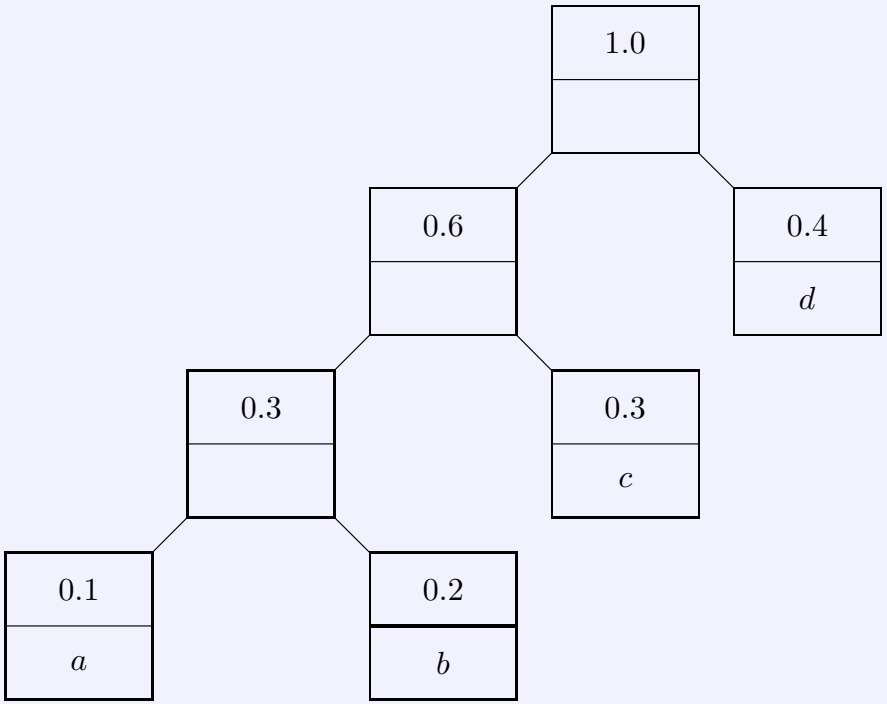
Tekst zakodowany 011101000001001000 01
01 c | 1 d | 1 d | 01 c | 000 a | 001 b |

Odkodowywanie tekstu



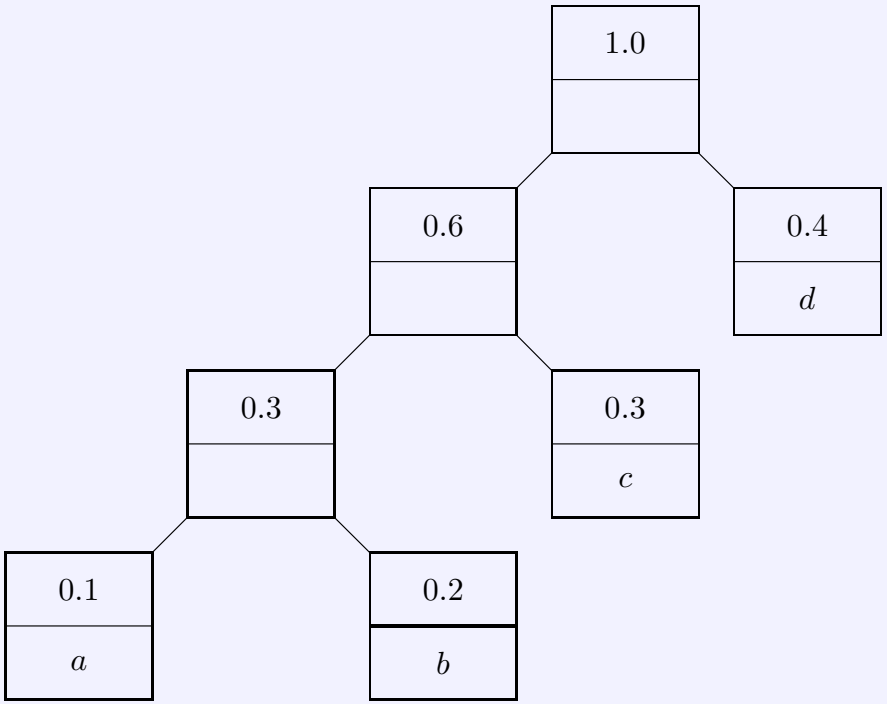
Tekst zakodowany 011101000001001000 01
01 c | 1 d | 1 d | 01 c | 000 a | 001 b | 001 b|

Odkodowywanie tekstu



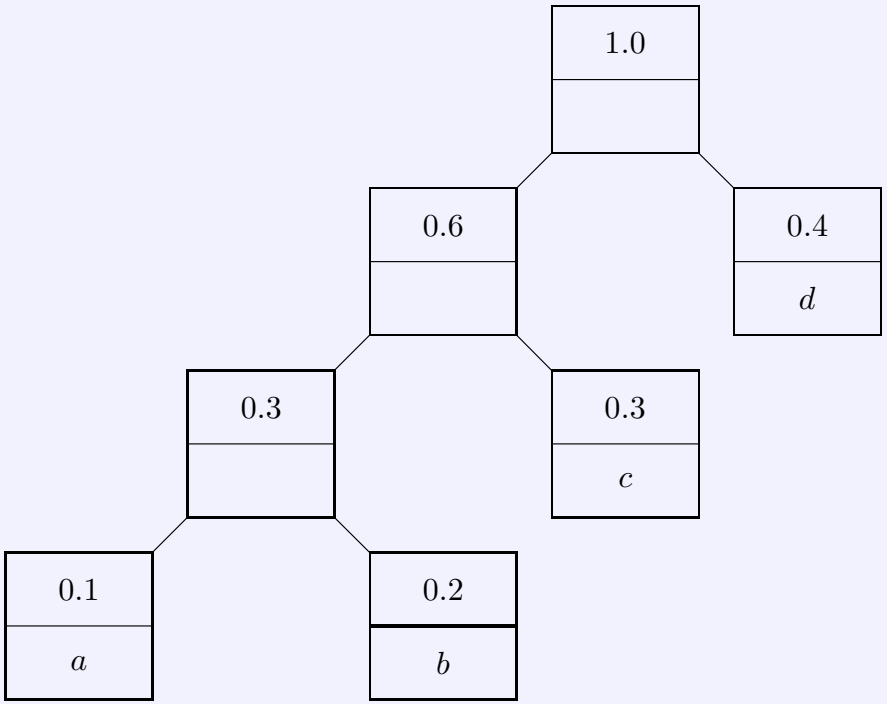
Tekst zakodowany 01110100000100100001
01 c | 1 d | 1 d | 01 c | 000 a | 001 b | 001 b| 000 a |

Odkodowywanie tekstu



Tekst zakodowany 01110100000100100001
01 c | 1 d | 1 d | 01 c | 000 a | 001 b | 001 b| 000 a | 01 c

Odkodowywanie tekstu



Tekst zakodowany 01110100000100100001
01 c | 1 d | 1 d | 01 c | 000 a | 001 b | 001 b| 000 a | 01 c

Tekst odkodowany cddcabbac

Optymalność

Entropia

$$\begin{aligned} H(S) &= -\left(0.1 \log_2 0.1 + 0.2 \log_2 0.2 + 0.3 \log_2 0.3 + 0.4 \log_2 0.4\right) \\ &= 1.8464. \end{aligned}$$

Taka jest optymalna długość kodu.

Optymalność

Entropia

$$\begin{aligned} H(S) &= -\left(0.1 \log_2 0.1 + 0.2 \log_2 0.2 + 0.3 \log_2 0.3 + 0.4 \log_2 0.4\right) \\ &= 1.8464. \end{aligned}$$

Taka jest optymalna długość kodu.

Dla kodu Huffmana

$$\frac{H(S)}{L} = \frac{1.8464}{1.9} \approx 0.972$$

Optymalność

Entropia

$$\begin{aligned} H(S) &= -\left(0.1 \log_2 0.1 + 0.2 \log_2 0.2 + 0.3 \log_2 0.3 + 0.4 \log_2 0.4\right) \\ &= 1.8464. \end{aligned}$$

Taka jest optymalna długość kodu.

Dla kodu Huffmana

$$\frac{H(S)}{L} = \frac{1.8464}{1.9} \approx 0.972$$

Efektywność w tym przypadku wynosi 97.2%,

Przykład z wykładu 1 (1)

Załóżmy, że urządzenie jest

- w stanie normalnym z prawdopodobieństwem 0.98,
- stanie A z prawdopodobieństwem 0.01,
- w każdym ze stanów B i C, z prawdopodobieństwami po 0.005.

Wtedy

$$\begin{aligned} H &= -\left(0.98 \log_2 0.98 + 0.01 \cdot \log_2 0.01 + 2 \cdot 0.005 \cdot \log_2 0.005\right) \\ &= 0.1714405425418207. \end{aligned}$$

Przykład z wykładu 1 (2)

Kod:

N	0
A	101
B	110
C	111

Odczyt: 0 – stan N, jeśli 1, to odczytujemy dwa następne bity.
Średnia długość słowa kodowego:

$$L = 0.98 + 3 \cdot 0.01 + 2 \cdot 3 \cdot 0.005 = 1.04.$$

Efektywność

$$\frac{H}{L} = \frac{0.1714}{1.04} = 0.1648.$$

Przykład z wykładu 1 (3)

Kod Huffmana

N	1
A	01
B	000
C	001

Średnia długość słowa kodowego:

$$L = 1 \cdot 0.98 + 2 \cdot 0.01 + 2 \cdot 3 \cdot 0.005 = 1.03.$$

Efektywność

$$\frac{H}{L} = \frac{0.1714}{1.03} = 0.1664.$$