Problems 4 - Character codings, Dictionary coders

Coding Technology

Illés Horváth

2025/10/31

Reminder: character encodings

Character codings: prefix codes, code tree for decoding.

Fixed length encoding.

Shannon–Fano code: codeword lengths $\ell_i = \lceil -\log_2 p_i \rceil$, greedy tree construction.

Huffman code: in each step, add the two smallest probabilities. Huffman is optimal among character codings.

Entropy as a measure of information. Theoretical lower bound for average codeword length.

$$H(X) = \sum_i p_i \log_2(1/p_i)$$

Arithmetic coding: partition $[0, 1]$ according to character probabilities, repeat for subintervals in each step. Not a character encoding, but asymptotically optimal.

Problem 1

We have a source with the following distribution and code table:

Source symbol	Probability	Codeword
A	0.4	0
B	0.2	10
C	0.2	110
D	0.2	1111

(a) Is this a prefix code?

(b) What is the average codelength?

(c) How far is the average codelength from the theoretical lower bound of compressibility?

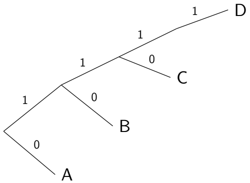
(d) Is this an optimal character encoding?

Problem 1

Solution.

(a) In order to check if the code is prefix, we can draw the code tree:

A	0
B	10
C	110
D	1111



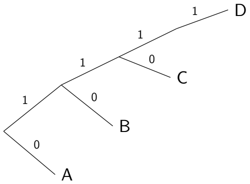
Characters are only in the leaves, so this is a prefix code.

Problem 1

Solution.

(a) In order to check if the code is prefix, we can draw the code tree:

A	0
B	10
C	110
D	1111



Characters are only in the leaves, so this is a prefix code.

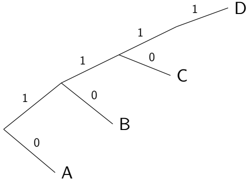
(b) $L = \sum_{i=1}^{4} p_i \ell_i = 0.4 \cdot 1 + 0.2 \cdot 2 + 0.2 \cdot 3 + 0.2 \cdot 4 = 2.2.$

Problem 1

Solution.

(a) In order to check if the code is prefix, we can draw the code tree:

A	0
B	10
C	110
D	1111



Characters are only in the leaves, so this is a prefix code.

(b) $L = \sum_{i=1}^{4} p_i \ell_i = 0.4 \cdot 1 + 0.2 \cdot 2 + 0.2 \cdot 3 + 0.2 \cdot 4 = 2.2.$

(c) Theoretical lower bound for average codeword length:

$$H(X) = \sum_{i=1}^{4} p_i \log_2 \left(\frac{1}{p_i} \right) = 0.4 \cdot 1.31 + 3 \cdot 0.2 \cdot 2.322 = 1.922,$$

so

$$L - H(X) = 0.278$$

Problem 1

(d) Is this an optimal character encoding?

Problem 1

(d) Is this an optimal character encoding?

For the character D, the codeword 111 is sufficient instead of 1111 (and the code is still prefix), so the code is not optimal.

Problem 1

(d) Is this an optimal character encoding?

For the character D, the codeword 111 is sufficient instead of 1111 (and the code is still prefix), so the code is not optimal.

What if we use D $\rightarrow$ 111, is the code optimal then?

Problem 1

(d) Is this an optimal character encoding?

For the character D, the codeword 111 is sufficient instead of 1111 (and the code is still prefix), so the code is not optimal.

What if we use D $\rightarrow$ 111, is the code optimal then?

We can check by comparing it to Huffman:

Problem 1

(d) Is this an optimal character encoding?

For the character D, the codeword 111 is sufficient instead of 1111 (and the code is still prefix), so the code is not optimal.

What if we use D $\rightarrow$ 111, is the code optimal then?

We can check by comparing it to Huffman:

$$p_1 = 0.4$$
$$p_2 = 0.2$$
$$p_3 = 0.2$$
$$p_4 = 0.2$$

Problem 1

(d) Is this an optimal character encoding?

For the character D, the codeword 111 is sufficient instead of 1111 (and the code is still prefix), so the code is not optimal.

What if we use D $\rightarrow$ 111, is the code optimal then?

We can check by comparing it to Huffman:

$p_1 = 0.4$	0.4
$p_2 = 0.2$	0.2
$p_3 = 0.2$	0.4
$p_4 = 0.2$	

Problem 1

(d) Is this an optimal character encoding?

For the character D, the codeword 111 is sufficient instead of 1111 (and the code is still prefix), so the code is not optimal.

What if we use D $\rightarrow$ 111, is the code optimal then?

We can check by comparing it to Huffman:

$$
\begin{array}{lll}
p_1 = 0.4 & 0.4 & 0.4 \\
p_2 = 0.2 & 0.2 & 0.6 \\
p_3 = 0.2 & 0.4 & \\
p_4 = 0.2 & &
\end{array}
$$

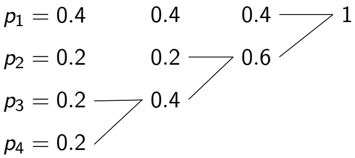
Problem 1

(d) Is this an optimal character encoding?

For the character D, the codeword 111 is sufficient instead of 1111 (and the code is still prefix), so the code is not optimal.

What if we use D $\rightarrow$ 111, is the code optimal then?

We can check by comparing it to Huffman:



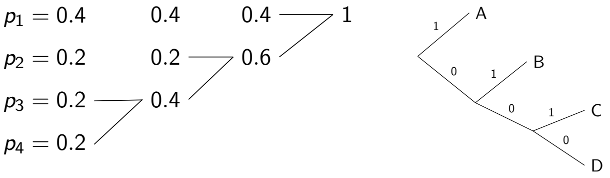
Problem 1

(d) Is this an optimal character encoding?

For the character D, the codeword 111 is sufficient instead of 1111 (and the code is still prefix), so the code is not optimal.

What if we use D $\rightarrow$ 111, is the code optimal then?

We can check by comparing it to Huffman:



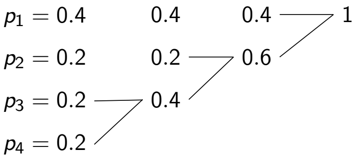
Problem 1

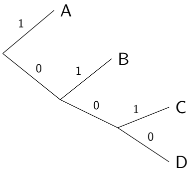
(d) Is this an optimal character encoding?

For the character D, the codeword 111 is sufficient instead of 1111 (and the code is still prefix), so the code is not optimal.

What if we use D $\rightarrow$ 111, is the code optimal then?

We can check by comparing it to Huffman:





A	1
B	01
C	001
D	000

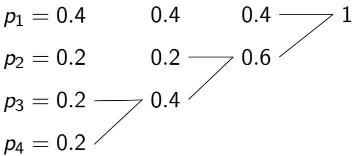
Problem 1

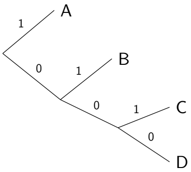
(d) Is this an optimal character encoding?

For the character D, the codeword 111 is sufficient instead of 1111 (and the code is still prefix), so the code is not optimal.

What if we use D $\rightarrow$ 111, is the code optimal then?

We can check by comparing it to Huffman:





A	1
B	01
C	001
D	000

Same codeword lengths $\rightarrow$ the code is equivalent to Huffman, both are optimal, and the average codeword length is

$$L = 0.4 \cdot 1 + 0.2 \cdot 2 + 0.2 \cdot 3 + 0.2 \cdot 3 = 2.0.$$

Problem 1

(d) During Huffman, in the second column we picked the bottom 0.4.

$p_1 = 0.4$	0.4	0.4	1
$p_2 = 0.2$	0.2	0.6	
$p_3 = 0.2$	0.4		
$p_4 = 0.2$			

Problem 1

(d) During Huffman, in the second column we picked the bottom 0.4.

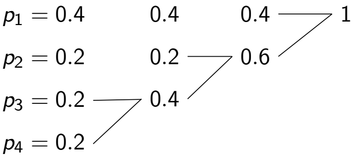
$$
\begin{array}{llll}
p_1 = 0.4 & 0.4 & 0.4 & 1 \\
p_2 = 0.2 & 0.2 & 0.6 & \\
p_3 = 0.2 & 0.4 & & \\
p_4 = 0.2 & & &
\end{array}
$$

What if we pick the other 0.4?

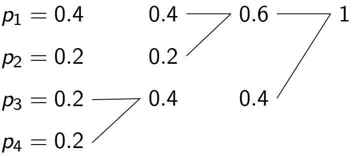
$$
\begin{array}{llll}
p_1 = 0.4 & 0.4 & 0.6 & 1 \\
p_2 = 0.2 & 0.2 & & \\
p_3 = 0.2 & 0.4 & 0.4 & \\
p_4 = 0.2 & & &
\end{array}
$$

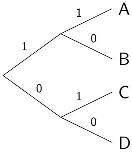
Problem 1

(d) During Huffman, in the second column we picked the bottom 0.4.



What if we pick the other 0.4?

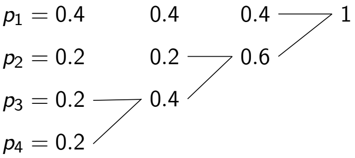




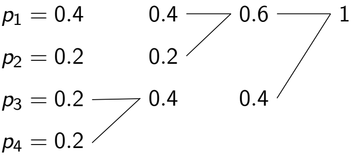
A	11
B	10
C	01
D	00

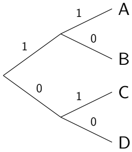
Problem 1

(d) During Huffman, in the second column we picked the bottom 0.4.



What if we pick the other 0.4?





A	11
B	10
C	01
D	00

$L = 2$, same as for the previous choice. This coding is also an optimal Huffman code, just using a different tree.

Problem 2

Encode the following distribution using Fixed length coding, Shannon–Fano coding and Huffman coding. Compute the average codeword length for each coding. Compute the theoretical lower bound.

$$p_1 = 0.49$$
$$p_2 = 0.14$$
$$p_3 = 0.14$$
$$p_4 = 0.07$$
$$p_5 = 0.07$$
$$p_6 = 0.04$$
$$p_7 = 0.02$$
$$p_8 = 0.02$$
$$p_9 = 0.01$$

Problem 2

Solution. For fixed length encoding, the required codeword length is $\lceil \log_2 9 \rceil = 4$, and

Symbol	Codeword
X_1	0000
X_2	0001
X_3	0010
X_4	0011
X_5	0100
X_6	0101
X_7	0110
X_8	0111
X_9	1000

Average codeword length is $L = 4$.

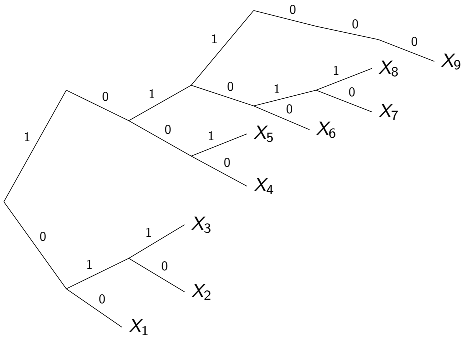
Problem 2

For the Shannon–Fano coding, the codeword lengths are $\ell_i = \lceil \log_2 1/p_i \rceil$, so

$$\begin{aligned}
\ell_1 &= \lceil \log_2 1/0.49 \rceil = \lceil 1.029 \rceil = 2, \\
\ell_2 &= \lceil \log_2 1/0.14 \rceil = \lceil 2.836 \rceil = 3, \\
\ell_3 &= \lceil \log_2 1/0.14 \rceil = \lceil 2.836 \rceil = 3, \\
\ell_4 &= \lceil \log_2 1/0.07 \rceil = \lceil 3.836 \rceil = 4, \\
\ell_5 &= \lceil \log_2 1/0.07 \rceil = \lceil 3.836 \rceil = 4, \\
\ell_6 &= \lceil \log_2 1/0.04 \rceil = \lceil 4.644 \rceil = 5, \\
\ell_7 &= \lceil \log_2 1/0.02 \rceil = \lceil 5.644 \rceil = 6, \\
\ell_8 &= \lceil \log_2 1/0.02 \rceil = \lceil 5.644 \rceil = 6, \\
\ell_9 &= \lceil \log_2 1/0.01 \rceil = \lceil 6.644 \rceil = 7.
\end{aligned}$$

Problem 3

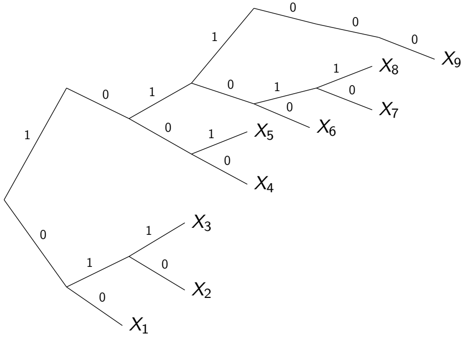
Code tree and codeword table for Shannon–Fano:



Symbol	Codeword
X_1	00
X_2	010
X_3	011
X_4	1000
X_5	1001
X_6	10100
X_7	101010
X_8	101011
X_9	1011000

Problem 3

Code tree and codeword table for Shannon–Fano:



Symbol	Codeword
X_1	00
X_2	010
X_3	011
X_4	1000
X_5	1001
X_6	10100
X_7	101010
X_8	101011
X_9	1011000

The average codeword length is

$$\begin{aligned} L_{\text{S–F}} =& 0.49 \cdot 2 + 2 \times 0.14 \cdot 3 + 2 \times 0.07 \cdot 4 + \\ & 0.04 \cdot 5 + 2 \times 0.02 \cdot 6 + 0.01 \cdot 7 = 2.89. \end{aligned}$$

Problem 2

For Huffman coding, we add the two smallest probabilities in each step.

$p_1 = 0.49$

$p_2 = 0.14$

$p_3 = 0.14$

$p_4 = 0.07$

$p_5 = 0.07$

$p_6 = 0.04$

$p_7 = 0.02$

$p_8 = 0.02$

$p_9 = 0.01$

Problem 2

For Huffman coding, we add the two smallest probabilities in each step.

$p_1 = 0.49$	0.49
$p_2 = 0.14$	0.14
$p_3 = 0.14$	0.14
$p_4 = 0.07$	0.07
$p_5 = 0.07$	0.07
$p_6 = 0.04$	0.04
$p_7 = 0.02$	0.02
$p_8 = 0.02$	0.03
$p_9 = 0.01$	

Problem 2

For Huffman coding, we add the two smallest probabilities in each step.

$p_1 = 0.49$	0.49	0.49
$p_2 = 0.14$	0.14	0.14
$p_3 = 0.14$	0.14	0.14
$p_4 = 0.07$	0.07	0.07
$p_5 = 0.07$	0.07	0.07
$p_6 = 0.04$	0.04	0.04
$p_7 = 0.02$	0.02	0.05
$p_8 = 0.02$	0.03	
$p_9 = 0.01$		

Problem 2

For Huffman coding, we add the two smallest probabilities in each step.

$p_1 = 0.49$	0.49	0.49	0.49
$p_2 = 0.14$	0.14	0.14	0.14
$p_3 = 0.14$	0.14	0.14	0.14
$p_4 = 0.07$	0.07	0.07	0.07
$p_5 = 0.07$	0.07	0.07	0.07
$p_6 = 0.04$	0.04	0.04	0.09
$p_7 = 0.02$	0.02	0.05	
$p_8 = 0.02$	0.03		
$p_9 = 0.01$			

Problem 2

For Huffman coding, we add the two smallest probabilities in each step.

$p_1 = 0.49$	0.49	0.49	0.49	0.49
$p_2 = 0.14$	0.14	0.14	0.14	0.14
$p_3 = 0.14$	0.14	0.14	0.14	0.14
$p_4 = 0.07$	0.07	0.07	0.07	0.14
$p_5 = 0.07$	0.07	0.07	0.07	0.09
$p_6 = 0.04$	0.04	0.04	0.09	
$p_7 = 0.02$	0.02	0.05		
$p_8 = 0.02$	0.03			
$p_9 = 0.01$				

Problem 2

For Huffman coding, we add the two smallest probabilities in each step.

$p_1 = 0.49$	0.49	0.49	0.49	0.49	0.49
$p_2 = 0.14$	0.14	0.14	0.14	0.14	0.14
$p_3 = 0.14$	0.14	0.14	0.14	0.14	0.14
$p_4 = 0.07$	0.07	0.07	0.07	0.14	0.23
$p_5 = 0.07$	0.07	0.07	0.07	0.09	
$p_6 = 0.04$	0.04	0.04	0.09		
$p_7 = 0.02$	0.02	0.05			
$p_8 = 0.02$	0.03				
$p_9 = 0.01$					

Problem 2

For Huffman coding, we add the two smallest probabilities in each step.

$p_1 = 0.49$	0.49	0.49	0.49	0.49	0.49	0.49
$p_2 = 0.14$	0.14	0.14	0.14	0.14	0.14	0.28
$p_3 = 0.14$	0.14	0.14	0.14	0.14	0.14	0.23
$p_4 = 0.07$	0.07	0.07	0.07	0.14	0.23	
$p_5 = 0.07$	0.07	0.07	0.07	0.09		
$p_6 = 0.04$	0.04	0.04	0.09			
$p_7 = 0.02$	0.02	0.05				
$p_8 = 0.02$	0.03					
$p_9 = 0.01$						

Problem 2

For Huffman coding, we add the two smallest probabilities in each step.

$p_1 = 0.49$	0.49	0.49	0.49	0.49	0.49	0.49	0.49
$p_2 = 0.14$	0.14	0.14	0.14	0.14	0.14	0.28	0.51
$p_3 = 0.14$	0.14	0.14	0.14	0.14	0.14	0.23	
$p_4 = 0.07$	0.07	0.07	0.07	0.14	0.23		
$p_5 = 0.07$	0.07	0.07	0.07	0.09			
$p_6 = 0.04$	0.04	0.04	0.09				
$p_7 = 0.02$	0.02	0.05					
$p_8 = 0.02$	0.03						
$p_9 = 0.01$							

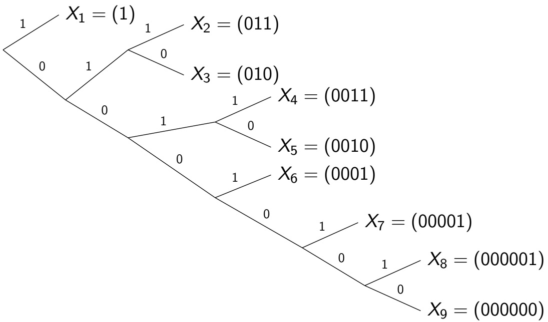
Problem 2

For Huffman coding, we add the two smallest probabilities in each step.

$p_1 = 0.49$	0.49	0.49	0.49	0.49	0.49	0.49	0.49	1
$p_2 = 0.14$	0.14	0.14	0.14	0.14	0.14	0.28	0.51	
$p_3 = 0.14$	0.14	0.14	0.14	0.14	0.14	0.23		
$p_4 = 0.07$	0.07	0.07	0.07	0.14	0.23			
$p_5 = 0.07$	0.07	0.07	0.07	0.09				
$p_6 = 0.04$	0.04	0.04	0.09					
$p_7 = 0.02$	0.02	0.05						
$p_8 = 0.02$	0.03							
$p_9 = 0.01$								

Problem 2

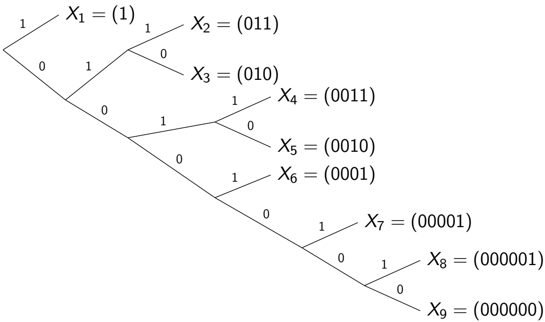
Then the code tree and codeword table can be obtained:



Symbol	Codeword
X_1	1
X_2	011
X_3	010
X_4	0011
X_5	0010
X_6	0001
X_7	00001
X_8	000001
X_9	000000

Problem 2

Then the code tree and codeword table can be obtained:



Symbol	Codeword
X_1	1
X_2	011
X_3	010
X_4	0011
X_5	0010
X_6	0001
X_7	00001
X_8	000001
X_9	000000

$$L_{\text{Huff}} = 0.49 \cdot 1 + 0.14 \cdot 3 + 0.14 \cdot 3 + 0.07 \cdot 4 + 0.07 \cdot 4 + \\ + 0.04 \cdot 4 + 0.02 \cdot 5 + 0.02 \cdot 6 + 0.01 \cdot 6 = 2.33$$

Problem 2

Entropy of the source:

$$H(X) = \sum_{i=1}^{9} p_i \log_2 \left(\frac{1}{p_i} \right) = 2.314.$$

Problem 2

Entropy of the source:

$$H(X) = \sum_{i=1}^{9} p_i \log_2 \left(\frac{1}{p_i} \right) = 2.314.$$

Comparison with the average codeword length for the various codings:

$$\begin{aligned} L_{\text{fixed}} &= 4 \\ L_{\text{S–F}} &= 2.89 \\ L_{\text{Huff}} &= 2.33 \end{aligned}$$

Problem 3

A source has alphabet {A,B,C,D}, with distribution

$$P(A) = 0.4, \quad P(B) = 0.3, \quad P(C) = 0.2, \quad P(D) = 0.1.$$

For each of Fixed length coding, Huffman coding and Arithmetic coding, what is the length of the codeword corresponding to the message AAAAA?

And for the messages BBBBB, CCCCC and DDDDD?

Problem 3

A source has alphabet {A,B,C,D}, with distribution

$$P(A) = 0.4, \quad P(B) = 0.3, \quad P(C) = 0.2, \quad P(D) = 0.1.$$

For each of Fixed length coding, Huffman coding and Arithmetic coding, what is the length of the codeword corresponding to the message AAAAA?

And for the messages BBBBB, CCCCC and DDDDD?

Solution. For Fixed length encoding, all characters are 2 bits, so the length of each of the four messages is 10 bits.

Problem 3

For Huffman coding,

$$p_1 = 0.4$$

$$p_2 = 0.3$$

$$p_3 = 0.2$$

$$p_4 = 0.1$$

Problem 3

For Huffman coding,

$$\begin{array}{ll}
p_1 = 0.4 & 0.4 \\
p_2 = 0.3 & 0.3 \\
p_3 = 0.2 & 0.3 \\
p_4 = 0.1 &
\end{array}$$

Problem 3

For Huffman coding,

$p_1 = 0.4$	0.4	0.4
$p_2 = 0.3$	0.3	0.6
$p_3 = 0.2$	0.3	
$p_4 = 0.1$		

Problem 3

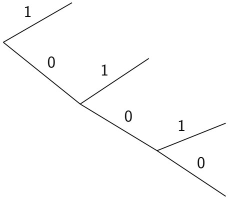
For Huffman coding,

$p_1 = 0.4$	0.4	0.4	1
$p_2 = 0.3$	0.3	0.6	
$p_3 = 0.2$	0.3		
$p_4 = 0.1$			

Problem 3

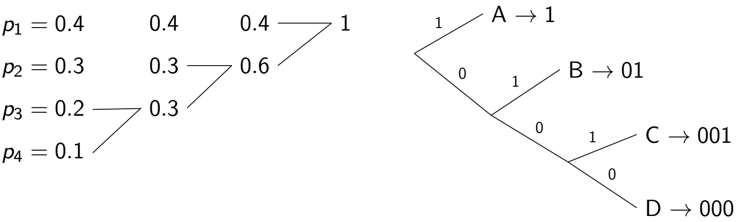
For Huffman coding,

$$\begin{array}{llll}
p_1 = 0.4 & 0.4 & 0.4 & 1 \\
p_2 = 0.3 & 0.3 & 0.6 & \\
p_3 = 0.2 & 0.3 & & \\
p_4 = 0.1 & & &
\end{array}$$



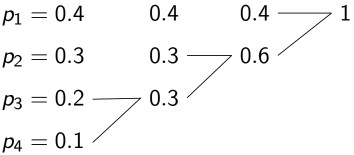
Problem 3

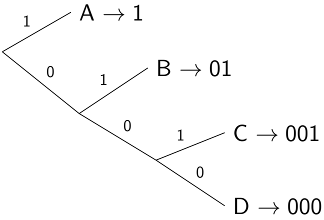
For Huffman coding,



Problem 3

For Huffman coding,





The codewords are:

- AAAAA $\to$ 11111, length 5
- BBBBB $\to$ 0101010101, length 10
- CCCCC $\to$ 001001001001001, length 15
- DDDDD $\to$ 000000000000000, length 15

Problem 3

For Arithmetic coding, the length of the messages:

$$\text{AAAAA} \quad \rightarrow \quad \left\lceil -\log_2\left(0.4^5\right)\right\rceil + 1 = 7$$

$$\text{BBBBB} \quad \rightarrow \quad \left\lceil -\log_2\left(0.3^5\right)\right\rceil + 1 = 10$$

$$\text{CCCCC} \quad \rightarrow \quad \left\lceil -\log_2\left(0.2^5\right)\right\rceil + 1 = 13$$

$$\text{DDDDD} \quad \rightarrow \quad \left\lceil -\log_2\left(0.1^5\right)\right\rceil + 1 = 18$$

Problem 4

Using the Fixed length encoding, Huffman and Arithmetic coding from the previous problem, compute the codeword length for a message containing 40 A's, 30 B's, 20 C's and 10 D's total.

Problem 4

Using the Fixed length encoding, Huffman and Arithmetic coding from the previous problem, compute the codeword length for a message containing 40 A's, 30 B's, 20 C's and 10 D's total.

Solution. For Fixed length encoding, the codeword length is

$$100 \times 2 = 200 \text{ bits.}$$

Problem 4

Using the Fixed length encoding, Huffman and Arithmetic coding from the previous problem, compute the codeword length for a message containing 40 A's, 30 B's, 20 C's and 10 D's total.

Solution. For Fixed length encoding, the codeword length is

$$100 \times 2 = 200 \text{ bits.}$$

Using Huffman, it is

$$40 \times 1 + 30 \times 2 + 20 \times 3 + 10 \times 3 = 190 \text{ bits.}$$

Problem 4

Using the Fixed length encoding, Huffman and Arithmetic coding from the previous problem, compute the codeword length for a message containing 40 A's, 30 B's, 20 C's and 10 D's total.

Solution. For Fixed length encoding, the codeword length is

$$100 \times 2 = 200 \text{ bits.}$$

Using Huffman, it is

$$40 \times 1 + 30 \times 2 + 20 \times 3 + 10 \times 3 = 190 \text{ bits.}$$

With Arithmetic coding, it is

$$\left\lceil -\log_2(0.4^{40} \cdot 0.3^{30} \cdot 0.2^{20} \cdot 0.1^{10}) \right\rceil + 1 = 186 \text{ bits.}$$

Problem 5

We have a source with 7 characters and probabilities

$$p_1 = 0.5, \quad p_2 = p_3 = 0.125, \quad p_4 = p_5 = p_6 = p_7 = 0.0625.$$

Compute the codeword lengths for this source for both Shannon–Fano coding and Huffman coding. Also compute the entropy of the source.

Problem 5

We have a source with 7 characters and probabilities

$$p_1 = 0.5, \quad p_2 = p_3 = 0.125, \quad p_4 = p_5 = p_6 = p_7 = 0.0625.$$

Compute the codeword lengths for this source for both Shannon–Fano coding and Huffman coding. Also compute the entropy of the source.

Solution. For the Shannon–Fano coding, the codeword lengths are $\ell_i = \lceil \log_2 1/p_i \rceil$, that is,

$$\ell_1 = 1, \quad \ell_2 = \ell_3 = 3, \quad \ell_4 = \ell_5 = \ell_6 = \ell_7 = 4,$$

and the average codeword length is

$$L_{\text{S–F}} = 0.5 \times 1 + 2 \times 0.125 \times 3 + 4 \times 0.0625 \times 4 = 2.25.$$

Problem 5

For Huffman coding,

0.5

0.125

0.125

0.0625

0.0625

0.0625

0.0625

Problem 5

For Huffman coding,

0.5	1
0.125	0.125
0.125	0.125
0.0625	0.0625
0.0625	0.0625
0.0625	0.125
0.0625	

Problem 5

For Huffman coding,

0.5	1	0.5
0.125	0.125	0.125
0.125	0.125	0.125
0.0625	0.0625	0.125
0.0625	0.0625	0.125
0.0625	0.125	
0.0625		

Problem 5

For Huffman coding,

0.5	1	0.5	0.5
0.125	0.125	0.125	0.125
0.125	0.125	0.125	0.125
0.0625	0.0625	0.125	0.25
0.0625	0.0625	0.125	
0.0625	0.125		
0.0625			

Problem 5

For Huffman coding,

0.5	1	0.5	0.5	0.5
0.125	0.125	0.125	0.125	0.25
0.125	0.125	0.125	0.125	0.25
0.0625	0.0625	0.125	0.25	
0.0625	0.0625	0.125		
0.0625	0.125			
0.0625				

Problem 5

For Huffman coding,

0.5	1	0.5	0.5	0.5	0.5
0.125	0.125	0.125	0.125	0.25	0.5
0.125	0.125	0.125	0.125	0.25	
0.0625	0.0625	0.125	0.25		
0.0625	0.0625	0.125			
0.0625	0.125				
0.0625					

Problem 5

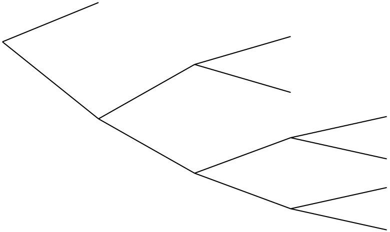
For Huffman coding,

0.5	1	0.5	0.5	0.5	0.5	1
0.125	0.125	0.125	0.125	0.25	0.5	
0.125	0.125	0.125	0.125	0.25		
0.0625	0.0625	0.125	0.25			
0.0625	0.0625	0.125				
0.0625	0.125					
0.0625						

Problem 5

For Huffman coding,

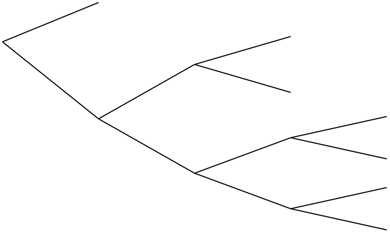
0.5	1	0.5	0.5	0.5	0.5	1
0.125	0.125	0.125	0.125	0.25	0.5	
0.125	0.125	0.125	0.125	0.25		
0.0625	0.0625	0.125	0.25			
0.0625	0.0625	0.125				
0.0625	0.125					
0.0625						



Problem 5

For Huffman coding,

0.5	1	0.5	0.5	0.5	0.5	1
0.125	0.125	0.125	0.125	0.25	0.5	
0.125	0.125	0.125	0.125	0.25		
0.0625	0.0625	0.125	0.25			
0.0625	0.0625	0.125				
0.0625	0.125					
0.0625						



Codeword lengths:
1, 3, 3, 4, 4, 4, 4

Problem 5

For Huffman coding, the average codeword length is

$$L_{\mathsf{Huff}} = 0.5 \times 1 + 2 \times 0.125 \times 3 + 4 \times 0.0625 \times 4 = 2.25.$$

Problem 5

For Huffman coding, the average codeword length is

$$L_{\mathrm{Huff}} = 0.5 \times 1 + 2 \times 0.125 \times 3 + 4 \times 0.0625 \times 4 = 2.25.$$

The entropy of the source is

$$\begin{aligned} H(X) = -\big(&0.5 \times \log_2(0.5) + 2 \times 0.125 \cdot \log_2(0.125) \\ &+ 4 \times 0.0625 \cdot \log_2(0.0625)\big) = 2.25. \end{aligned}$$

Problem 5

For Huffman coding, the average codeword length is

$$L_{\text{Huff}} = 0.5 \times 1 + 2 \times 0.125 \times 3 + 4 \times 0.0625 \times 4 = 2.25.$$

The entropy of the source is

$$\begin{aligned} H(X) = -\big(&0.5 \times \log_2(0.5) + 2 \times 0.125 \cdot \log_2(0.125) \\ &+ 4 \times 0.0625 \cdot \log_2(0.0625)\big) = 2.25. \end{aligned}$$

Both Shannon–Fano and Huffman reach the theoretical lower bound for this source.

Problem 5

For Huffman coding, the average codeword length is

$$L_{\text{Huff}} = 0.5 \times 1 + 2 \times 0.125 \times 3 + 4 \times 0.0625 \times 4 = 2.25.$$

The entropy of the source is

$$H(X) = -\big(0.5 \times \log_2(0.5) + 2 \times 0.125 \cdot \log_2(0.125) \\ + 4 \times 0.0625 \cdot \log_2(0.0625)\big) = 2.25.$$

Both Shannon–Fano and Huffman reach the theoretical lower bound for this source.

Actually, Huffman and Shannon–Fano both reach the entropy for the same sources in general: when all probabilities are negative integer powers of 2.

Adaptive Huffman code

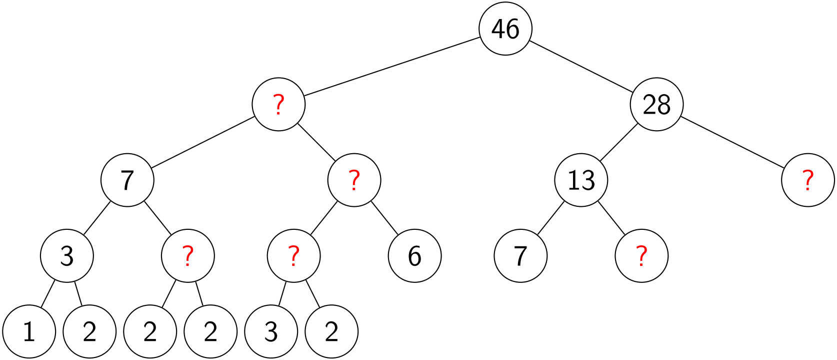
Adaptive Huffman code: the code tree is based on the number of occurrences of each character seen so far. Internal nodes get the sum of their children.

1. Initialize the code tree.
2. Code the next character according to the current state of the code tree.
3. Add the character to the tree.
4. Check the sibling property, and restore it if necessary by exchanging two nodes with their entire subtrees.
5. Go to next character and repeat from step 2.

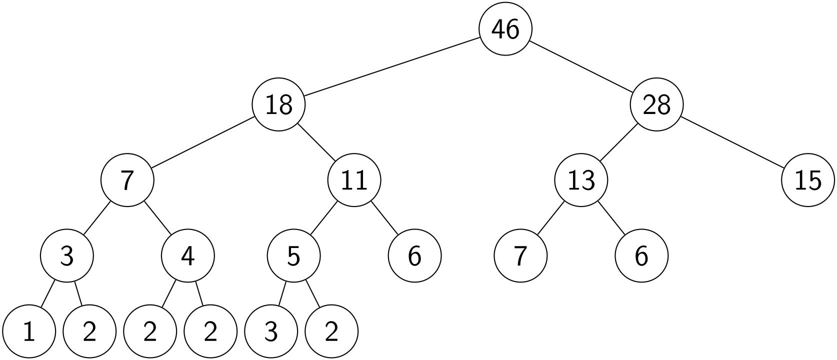
Sibling property: listing the nodes from bottom level to top level, going from left to right within each level, the nodes are increasing, except possibly for sibling pairs.

Problem 6

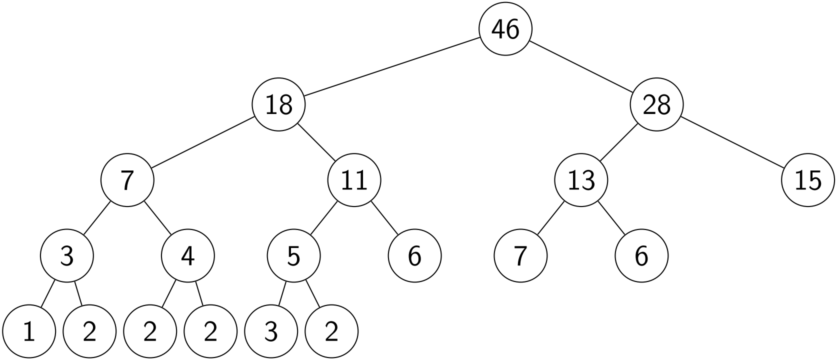
Determine the missing values. Does the sibling property hold for this tree?



Problem 6



Problem 6



Starting from the bottom left, going left to right first, then up one level and repeat, the sequence (1, 2), (2, 2), (3, 2), (3, 4), (5, 6), (7, 6), (7, 11), (13, 15), (18, 28), 46 is decreasing only for two sibling pairs, so the sibling property holds for this tree, this is a valid Huffman-tree.

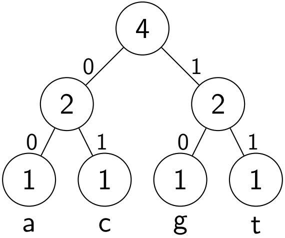
Problem 7

Over the alphabet $\{a,c,g,t\}$, apply adaptive Huffman coding to the message “aagacaa”.

Problem 7

Over the alphabet $\{a,c,g,t\}$, apply adaptive Huffman coding to the message "aagacaa".

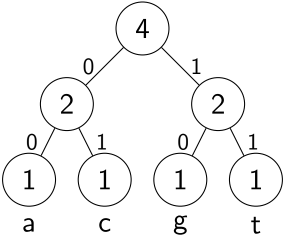
Solution. Initialization:



Problem 7

Over the alphabet {a,c,g,t}, apply adaptive Huffman coding to the message "aagacaa".

Solution. Initialization:

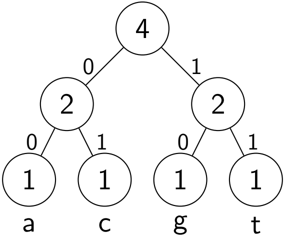


a $\rightarrow$ 00

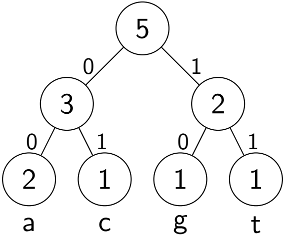
Problem 7

Over the alphabet {a,c,g,t}, apply adaptive Huffman coding to the message "aagacaa".

Solution. Initialization:



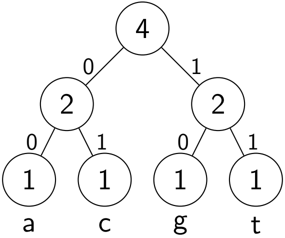
a $\rightarrow$ 00



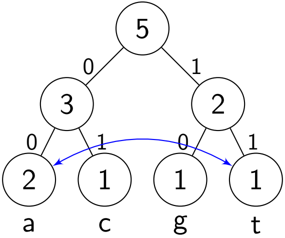
Problem 7

Over the alphabet {a,c,g,t}, apply adaptive Huffman coding to the message "aagacaa".

Solution. Initialization:

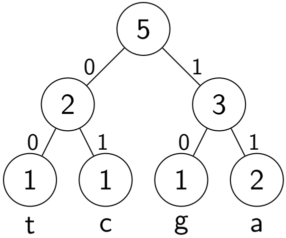


a $\rightarrow$ 00



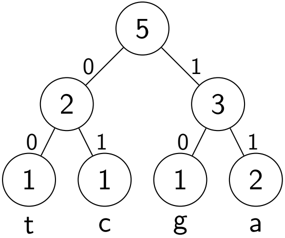
Problem 7

aagacaa



Problem 7

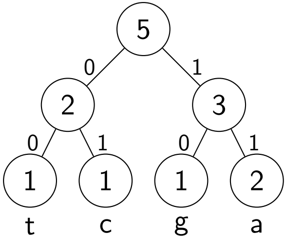
aagacaa

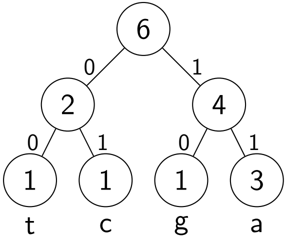


aa $\rightarrow$ 0011

Problem 7

aagacaa

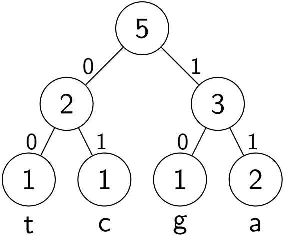


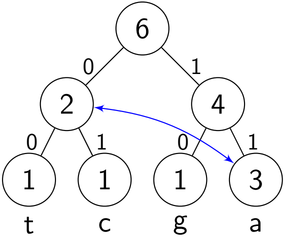


aa $\rightarrow$ 0011

Problem 7

aagacaa

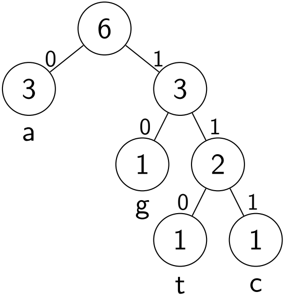




aa $\rightarrow$ 0011

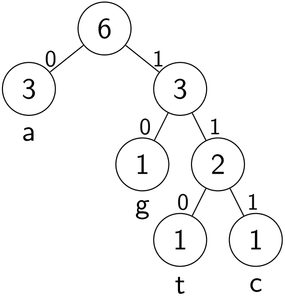
Problem 7

aagacaa



Problem 7

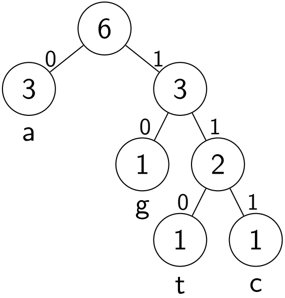
aagacaa



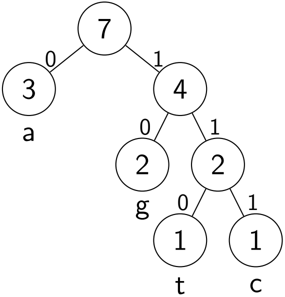
aag $\rightarrow$ 001110

Problem 7

aagacaa

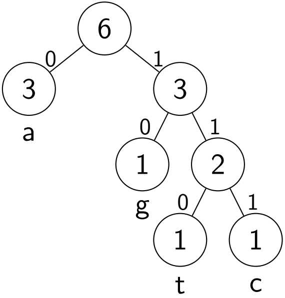


aag $\rightarrow$ 001110

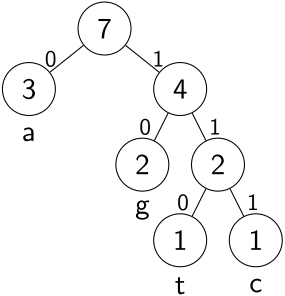


Problem 7

aagacaa



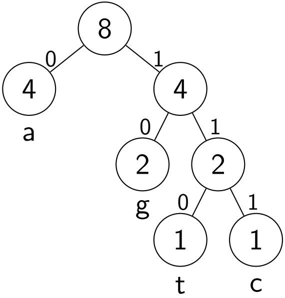
aag $\rightarrow$ 001110



aaga $\rightarrow$ 0011100

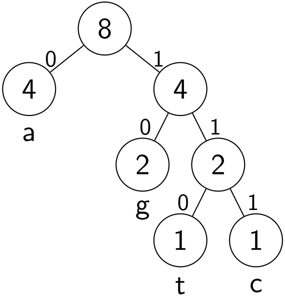
Problem 7

aagacaa



Problem 7

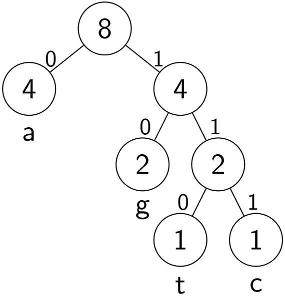
aagacaa



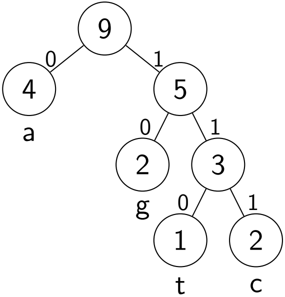
aagac $\rightarrow$ 001110111

Problem 7

aagacaa

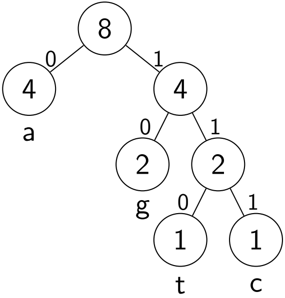


aagac $\rightarrow$ 001110111

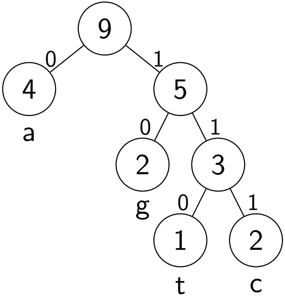


Problem 7

aagacaa



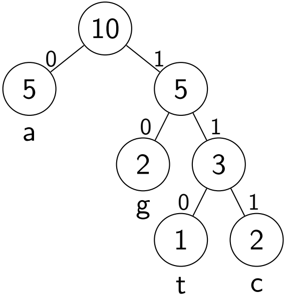
aagac → 001110111



aagaca → 0011101110

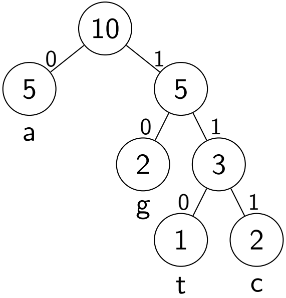
Problem 7

aagacaa



Problem 7

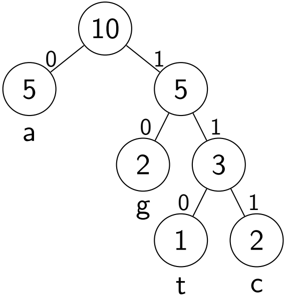
aagacaa



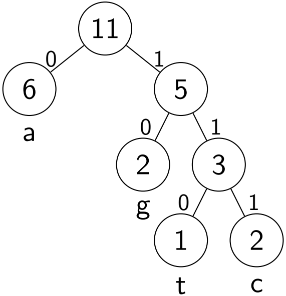
aagacaa $\rightarrow$ 00111011100

Problem 7

aagacaa

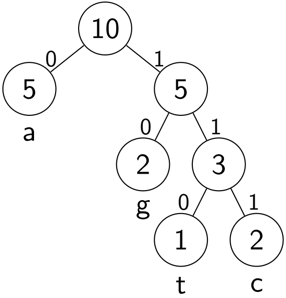


aagacaa $\rightarrow$ 00111011100

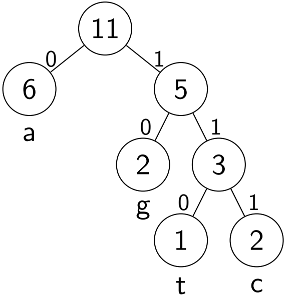


Problem 7

aagacaa



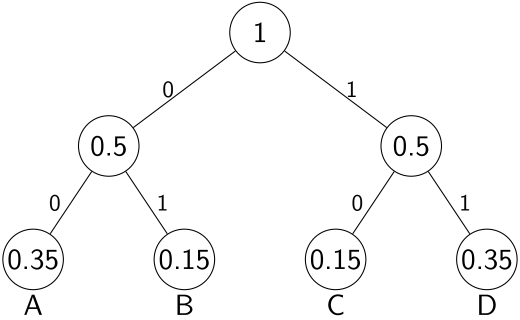
aagacaa → 00111011100



aagacaa → 0011101110

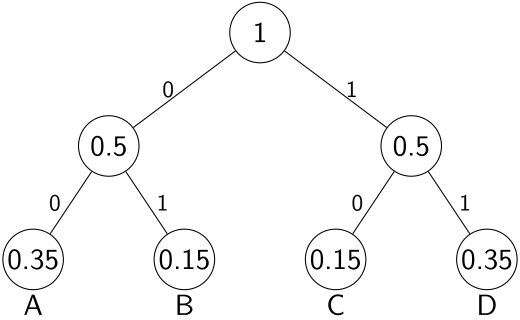
Problem 8

Can we obtain the following code tree as the result of a Huffman algorithm for code tree construction?



Problem 8

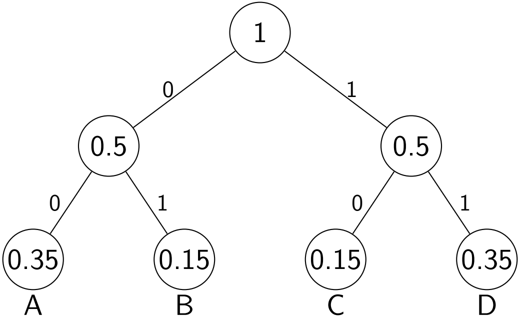
Can we obtain the following code tree as the result of a Huffman algorithm for code tree construction?



Solution 1. No, the sibling property does not hold.

Problem 8

Can we obtain the following code tree as the result of a Huffman algorithm for code tree construction?

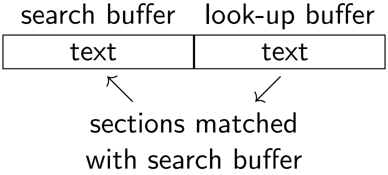


Solution 1. No, the sibling property does not hold.

Solution 2. For the Huffman tree construction, in the first step of the algorithm, the two smallest probabilities are the two 0.15's, so they would have to be matched to each other, not to the 0.35's.

Reminder: Dictionary coders

LZ77: search ahead for a section of text seen recently.



Output is a sequence of (p, ℓ, c) records (position of match, length of match, next character).

LZ78: parse the text into sections 1 character longer than seen before. The output is a sequence of records (i, c) (address of old section, new character).

Problem 9

Compute the next two records of the LZ77 algorithm, starting from the following position.

b | a b l a k b a r n a | a b l a k b a a h a r | a r r a d a. . .

Problem 9

Compute the next two records of the LZ77 algorithm, starting from the following position.

b | a b l a k b a r n a | a b l a k b a a h a r | a r r a d a. . .

Solution.

- first step:

10

| **a b l a k** b a r n a | **a b l a k** b a a h a r | a r r a d a. . .

5

Problem 9

Compute the next two records of the LZ77 algorithm, starting from the following position.

b $\boxed{\text{a b l a k b a r n a}}\boxed{\text{a b l a k b a a h a r}}$ a r r a d a. . .

Solution.

- first step:

  10

  $\boxed{\text{a b l a k b a r n a}}\boxed{\text{a b l a k b a a h a r}}$ a r r a d a. . . (10,5,b)

  5

Problem 9

Compute the next two records of the LZ77 algorithm, starting from the following position.

b | a b l a k b a r n a | a b l a k b a a h a r | a r r a d a . . .

Solution.

- first step:

  10

  | a b l a k b a r n a | a b l a k b a a h a r | a r r a d a . . .   (10,5,b)

  5

- next step:

  7

  a b l a k b | a r n a a b l a k b | a a h a r a r r a d a | . . .

  2

Problem 9

Compute the next two records of the LZ77 algorithm, starting from the following position.

b | a b l a k b a r n a | a b l a k b a a h a r | a r r a d a . . .

Solution.

- first step:

  10

  | a b l a k b a r n a | a b l a k b a a h a r | a r r a d a . . . (10,5,b)

  5

- next step:

  7

  a b l a k b | a r n a a b l a k b | a a h a r a r r a d a | . . . (7,2,h)

  2

Problem 10

An LZ77 coder has parameters $h_s = h_\ell = 4$. Decode the following sequence of records: (0,0,C), (0,0,A), (2,1,D), (3,2,B), (2,5,A).

Problem 10

An LZ77 coder has parameters $h_s = h_\ell = 4$. Decode the following sequence of records: (0,0,C), (0,0,A), (2,1,D), (3,2,B), (2,5,A).

Solution. (0,0,X)-type records code a single character (X), so the first two records are decoded to CA.

Problem 10

An LZ77 coder has parameters $h_s = h_\ell = 4$. Decode the following sequence of records: (0,0,C), (0,0,A), (2,1,D), (3,2,B), (2,5,A).

Solution. (0,0,X)-type records code a single character (X), so the first two records are decoded to CA.

The numbers in the record (2,1,D) are decoded as a copy command: move back 2 positions and copy 1 character (C). Then the D is added to the end $\rightarrow$ CACD.

Problem 10

An LZ77 coder has parameters $h_s = h_\ell = 4$. Decode the following sequence of records: (0,0,C), (0,0,A), (2,1,D), (3,2,B), (2,5,A).

Solution. (0,0,X)-type records code a single character (X), so the first two records are decoded to CA.

The numbers in the record (2,1,D) are decoded as a copy command: move back 2 positions and copy 1 character (C). Then the D is added to the end $\rightarrow$ CACD.

Similarly, (3,2,B) results in CACDACB.

Problem 10

An LZ77 coder has parameters $h_s = h_\ell = 4$. Decode the following sequence of records: (0,0,C), (0,0,A), (2,1,D), (3,2,B), (2,5,A).

Solution. (0,0,X)-type records code a single character (X), so the first two records are decoded to CA.

The numbers in the record (2,1,D) are decoded as a copy command: move back 2 positions and copy 1 character (C). Then the D is added to the end $\rightarrow$ CACD.

Similarly, (3,2,B) results in CACDACB.

For the record (2,5,A), we execute a similar copy command, but we only have 2 characters (CB) to copy; in this case, we keep copying those 2 characters cyclically until we get 5 characters (CBCBC), then add an A to the end $\rightarrow$ CACDACBCBCBCA.

Problem 11

Using LZ78, encode the message AABABBBABBABB. Also convert the dictionary to binary.

Problem 11

Using LZ78, encode the message AABABBBABBABB. Also convert the dictionary to binary.

Solution.

A A B A B B B A B B A B B

Problem 11

Using LZ78, encode the message AABABBBABBABB. Also convert the dictionary to binary.

Solution.

A|A B A B B B A B B A B B

1	(0,A)	1

Problem 11

Using LZ78, encode the message AABABBBABBABB. Also convert the dictionary to binary.

Solution.

A|A B|A B B B A B B A B B

1	(0,A)	1
2	(1,B)	111

Problem 11

Using LZ78, encode the message AABABBBABBABB. Also convert the dictionary to binary.

Solution.

$$A|A\,B|A\,B\,B|B\,A\,B\,B\,A\,B\,B$$

1	(0,A)	1
2	(1,B)	111
3	(2,B)	111101

Problem 11

Using LZ78, encode the message AABABBBABBABB. Also convert the dictionary to binary.

Solution.

A|A B|A B B|B|A B B A B B

1	(0,A)	1
2	(1,B)	111
3	(2,B)	111101
4	(0,B)	11110001

Problem 11

Using LZ78, encode the message AABABBBABBABB. Also convert the dictionary to binary.

Solution.

A|A B|A B B|B|A B B A|B B

1	(0,A)	1
2	(1,B)	111
3	(2,B)	111101
4	(0,B)	11110001
5	(3,A)	11110000110

Problem 11

Using LZ78, encode the message AABABBBABBABB. Also convert the dictionary to binary.

Solution.

$$A|A\,B|A\,B\,B|B|A\,B\,B\,A|B\,B$$

1	(0,A)	1
2	(1,B)	111
3	(2,B)	111101
4	(0,B)	11110001
5	(3,A)	11110000110
6	(4,B)	111100001101001