Mixed Huffman codes for on-line and off-line applications

Ryszard Stasinski*, and Grzegorz Ulacha†

March 24, 2022

* Poznan University of Technology, Poznan
†West Pomeranian University of Technology, Szczecin
Poland

Idea

Symbol merging					Coding				
0.4	0.4	0.4	0.4	*0.6*	0	1	1	1	1
0.3	0.3	0.3	0.3	0.4	1	00	00	00	00
0.1	0.1	*0.2*	*0.3*			01	010	011	011
0.1	0.1	0.1					011	0100	0100
0.06	*0.1*							0101	01010
0.04									01011
	0.4	0.4	0.4	*0.6*	0	1	1	1	
	0.3	0.3	0.3	0.4	1	00	00	00	
	0.1	0.1	*0.3*			01	$01\,0_3$	$01\,0_3$	
	0.1	0.1					$01\,1_3$	$01\,1_3$	
	0.06	*0.1*					$01\,2_3$	$01\,2_3\,0$	
	0.04							$01\,2_3\,1$	

Table 1: Binary (above) and mixed Huffman code $\{2, 3, 2, 2\}$ derivation.

Codes comparison

S	Binary	$\{2,2,2,3\}$	$\{2,3,2,2\}$	$\{2,3,3\}$
0.4	1	0_3	1	0_3
0.3	00	1_3	00	1_3
0.1	011	$2_3\,1$	$01\,0_3$	2_30_3
0.1	0100	$2_3\,00$	$01\,1_3$	2_31_3
0.06	01010	$2_3\,010$	$01\,2_3\,0$	$2_32_3\,0$
0.04	01011	$2_3\,011$	$01\,2_3\,1$	$2_32_3\,1$

Table 2: Best Huffman codes for the source above.

Binary	$\{2,2,2,3\}$	$\{2,3,2,2\}$	$\{2,3,3\}$
2.2	2.185	2.1755	2.1605
97.43%	98.1%	98.47%	99.22%

Table 3: Code lengths and efficiencies.

Element coding

Length $b(r)$ of number r - number of bits representing it.

Usually:

$$m = b(r^q) < q \cdot b(r),$$

e.g.: $b(5) = 3$, $b(5^3) = b(125) = 7 < 3 \cdot b(5) = 9$.

Best m, q:

$$m/q \approx \log r$$

When r^q coded using Huffman code:

$$q \cdot \underline{b}_q(r) = b(r^q) - \frac{2^{b(r^q)}}{r^q} + 1$$

$\underline{b}_q(r)$ - average number of bits used for coding of r.

Element coding 2

r	q	m	m/q	$\underline{b}_q(r)$	$\log r$
3	5	8	1.6	1.58930	1.58496
3	17	27	1.58824	1.58592	
3	41	65	1.58537	1.58509	
5	3	7	2.33333	2.32533	2.3219
5	31	72	2.32258	2.32213	
7	6	17	2.83333	2.8143	2.80735
7	11	31	2.81818	2.8104	
11	13	45	3.46154	3.4601	3.4594

Table 4: Binary coding of r-nary symbols, $m = b(r^q)$.

Experimental data sources

Source size	Uniform distribution	Linear distribution
3	82.62	90.31
4	47.85	40.04
5	39.07	38.00
6	25.96	27.05
7	13.33	11.02
8	5.57	2.72
9	2.48	0.45
10	1.64	0.14
11	1.26	0.11
12	0.83	0.04
20	$< 1/3 \cdot 10^{-3}$	$< 1/3 \cdot 10^{-3}$

Table 5: Percentage of binary Huffman codes that are optimal.

Experimental data sources 2

Coder	File 1	File 2	File 3	File 4
Uncoded	230103	1000000	340000	262144
Binary Huffman code	82292	300691	175083	122868
Mixed Huffman code	82081	293560	173929	121987
Arithmetic code	81483	292138	173784	121367
Entropy limit	81423.4	292069.6	173717.2	121301.9

Table 6: Length of compressed files in bytes.

File 1: 12 letters with decreasing probabilities, $p_{i+1} = 0.7p_i$.

File 2: 14-element alphabet.

File 3: 17 equiprobable symbols.

File 4: Lenna image quantized to 16 levels.

Coding delay

r^q coded $\rightarrow$

q element buffer needed $\rightarrow$

$d \geq q - 1$ element delay.

Shortest maximum delay code: $d = q - 1$.

It contains element(s) r in each codeword.

Many mixed Huffman codes better than the binary one (37 in the example on the next slide).

Notation:

- d_m - mean code delay
- d_c - delay which probability is 10^{-c}

Probability distribution, p - probability that a codeword contains r:

$$P(= d) = \binom{d}{q-1} p^q (1-p)^{d-q+1}$$

Coding delay 2

Code	q	d_m	d_2	d_6	d_∞	Length	$\eta\,[\%]$
Binary	1	0	0	0	0	3.26	99.00
$\{2,2,2,2,2,2,5\}$	3	2	2	2	2	3.2553	99.14
$\{2,3,2,3,2,2,3\}$	5	2.60	4	4	4	3.2522	99.24
$\{2,3,2,3,2,2,3\}$	17	10.71	13	16	16	3.2472	99.39
$\{2,3,2,3,2,2,3\}$	41	26.92	31	36	40	3.2459	99.43
$\{2,2,2,5,2,2,2\}$	3	4.88	12	27	∞	3.2427	99.53
$\{2,2,2,2,2,2,3,2,2\}$	5	8.43	17	33	∞	3.2423	99.54
$\{2,2,2,2,2,2,3,2,2\}$	17	31.08	45	67	∞	3.2405	99.59
$\{2,2,2,2,2,2,3,2,2\}$	∞	∞	∞	∞	∞	3.2400	99.61

Table 7: Delays and efficiencies of Huffman codes for the source:

$$S = \{0.22, 0.15, 0.12, 0.1, 0.1, 0.08, 0.06, 0.05, 0.05, 0.04, 0.03\}.$$

Thank you, questions?

gulacha@wi.zut.edu.pl
ryszard.stasinski@put.poznan.pl