Levenshtein coding

From Wikipedia, the free encyclopedia

Levenstein coding, or **Levenshtein coding**, is a universal code encoding the non-negative integers developed by Vladimir Levenshtein.[1][2]

Contents [hide]

Encoding [edit]

The code of zero is "0"; to code a positive number:

1. Initialize the step count variable *C* to 1.
2. Write the binary representation of the number without the leading "1" to the beginning of the code.
3. Let *M* be the number of bits written in step 2.
4. If *M* is not 0, increment *C*, repeat from step 2 with M as the new number.
5. Write *C* "1" bits and a "0" to the beginning of the code.

The code begins:

Number	Encoding	Implied probability
0	`0`	1/2
1	`10`	1/4
2	`110 0`	1/16
3	`110 1`	1/16
4	`1110 0 00`	1/128
5	`1110 0 01`	1/128
6	`1110 0 10`	1/128
7	`1110 0 11`	1/128
8	`1110 1 000`	1/256
9	`1110 1 001`	1/256
10	`1110 1 010`	1/256
11	`1110 1 011`	1/256
12	`1110 1 100`	1/256
13	`1110 1 101`	1/256
14	`1110 1 110`	1/256
15	`1110 1 111`	1/256
16	`11110 0 00 0000`	1/4096
17	`11110 0 00 0001`	1/4096

To decode a Levenstein-coded integer:

1. Count the number of "1" bits until a "0" is encountered.
2. If the count is zero, the value is zero, otherwise
3. Start with a variable *N*, set it to a value of 1 and repeat *count minus 1* times:

4. Read *N* bits, prepend "1", assign the resulting value to *N*

The Levenstein code of a positive integer is always one bit longer than the Elias omega code of that integer. However, there is a Levenstein code for zero, whereas Elias omega coding would require the numbers to be shifted so that a zero is represented by the code for one instead.

Example code [edit]

Encoding [edit]

```
void levenshteinEncode(char* source, char* dest)
{
    IntReader intreader(source);
    BitWriter bitwriter(dest);
    while (intreader.hasLeft())
    {
        int num = intreader.getInt();
        if (num == 0)
            bitwriter.outputBit(0);
        else
        {
            int c = 0;
            BitStack bits;
            do {
                int m = 0;
                for (int temp = num; temp > 1; temp>>=1)  // calculate 
floor(log2(num))
                    ++m;
                for (int i=0; i < m; ++i)
                    bits.pushBit((num >> i) & 1);
                num = m;
                ++c;
            } while (num > 0);
            for (int i=0; i < c; ++i)
                bitwriter.outputBit(1);
            bitwriter.outputBit(0);
            while (bits.length() > 0)
                bitwriter.outputBit(bits.popBit());
        }
    }
}
```

Decoding [edit]

```
void levenshteinDecode(char* source, char* dest)
{
    BitReader bitreader(source);
    IntWriter intwriter(dest);
    while (bitreader.hasLeft())
    {
        int n = 0;
        while (bitreader.inputBit())     // potentially dangerous with malformed 
files.
            ++n;
        int num;
        if (n == 0)
            num = 0;
        else
        {
            num = 1;
            for (int i = 0; i < n-1; ++i)
            {
                int val = 1;
                for (int j = 0; j < num; ++j)
                    val = (val << 1) | bitreader.inputBit();
                num = val;
            }
        }
```

```
        intwriter.putInt(num);            // write out the value
    }
    bitreader.close();
    intwriter.close();
}
```

See also [edit]

- Elias omega coding
- Iterated logarithm

References [edit]

1. ^ "1968 paper by V. I. Levenshtein (in Russian)" (PDF).
2. ^ David Salomon (2007). *Variable-length codes for data compression*. Springer. p. 80. ISBN 978-1-84628-958-3.

v · t · e		Data compression methods [hide]
Lossless	Entropy type	Unary · Arithmetic · Golomb · Huffman (Adaptive · Canonical · Modified) · Range · Shannon · Shannon–Fano · Shannon–Fano–Elias · Tunstall · Universal (Exp-Golomb · Fibonacci · Gamma · **Levenshtein**)
	Dictionary type	Byte pair encoding · DEFLATE · Lempel–Ziv (LZ77 / LZ78 (LZ1 / LZ2) · LZJB · LZMA · LZO · LZRW · LZS · LZSS · LZW · LZWL · LZX · LZ4 · Statistical)
	Other types	BWT · CTW · Delta · DMC · MTF · PAQ · PPM · RLE
Audio	Concepts	Bit rate (average (ABR) · constant (CBR) · variable (VBR)) · Companding · Convolution · Dynamic range · Latency · Nyquist–Shannon theorem · Sampling · Sound quality · Speech coding · Sub-band coding
	Codec parts	A-law · μ-law · ACELP · ADPCM · CELP · DPCM · Fourier transform · LPC (LAR · LSP) · MDCT · Psychoacoustic model · WLPC
Image	Concepts	Chroma subsampling · Coding tree unit · Color space · Compression artifact · Image resolution · Macroblock · Pixel · PSNR · Quantization · Standard test image
	Methods	Chain code · DCT · EZW · Fractal · KLT · LP · RLE · SPIHT · Wavelet
Video	Concepts	Bit rate (average (ABR) · constant (CBR) · variable (VBR)) · Display resolution · Frame · Frame rate · Frame types · Interlace · Video characteristics · Video quality
	Codec parts	Lapped transform · DCT · Deblocking filter · Motion compensation
Theory	Entropy · Kolmogorov complexity · Lossy · Quantization · Rate–distortion · Redundancy · Timeline of information theory	
	Compression formats · Compression software (codecs)	

Categories: Numeral systems | Lossless compression algorithms