DETAILED SOFTWARE SPECIFICATIONS

Delta Compression

Document revision history

DATE	REVISION	OBJECT	Author
08/01/16	0	Creation	S.D.

Controlled paper distribution:

	CREATED / WRITTEN BY	APPROVED BY
NAME	DUTERTRE Stéphane	
POSITION	R.B.E.	

TABLE OF CONTENTS

1. OBJECT OF THE DOCUMENT 3

2. APPLICABLE DOCUMENTS - REFERENCE DOCUMENTS 3

3. DEFINITIONS - TERMINOLOGY 3

4. SPECIFICATIONS 3

4.1 BASIC PRINCIPLE 3
4.2 THE PROBABILITY TREE 3
4.2.1 Detail of the probability tree 4
4.2.2 Bit size of encoding 5
4.2.3 Bit storage 6
4.2.4 Absolute compression limits 7
4.3 COMPRESSION EXAMPLE 7

1. OBJECT OF THE DOCUMENT

Description of the operating principle of Delta compression. This algorithm allows a buffer of raw data to be converted into a reduced size storage buffer.
This algorithm is mainly of interest for the compression of data from physical measurements in low, even very low, quantity. It is also relevant for computer systems with low computing power and very limited storage resources.

2. APPLICABLE DOCUMENTS - REFERENCE DOCUMENTS

General specifications ref.:

3. DEFINITIONS - TERMINOLOGY

Lossless compression algorithm: This is a compression / decompression algorithm which, after running through the compression procedure and then through the decompression algorithm, recovers the entirety and order of the initial data.
Compression ratio: measure of the performance of a computer data compression algorithm. Here, the computing principle used corresponds to the initial size of the buffer divided by the size of the compressed buffer.
Differential coding: takes into account the mathematical difference between two contiguous data stored in the memory. For it to be effective, it needs to know the format of the data to be compressed (8, 12, 16, ... bits).
Simplified Huffman coding: A probability tree is predefined without taking into account the actual recurrence rate of the data.
Variable length data: The datum is stored as a sequence of bits of variable length which is not modulo 2^8. It is important that this point is taken into account because it means that a datum can be recorded on less than 8 bits and/or that it can be stored unaligned on two bytes or more.

4. SPECIFICATIONS

4.1 Basic principle

The basic principle is quite simple to understand:

- If a sequence of data is derived from physical measurements subject to little change, the difference between two measurements tends to be small. This difference can be stored in the memory using less coding information.

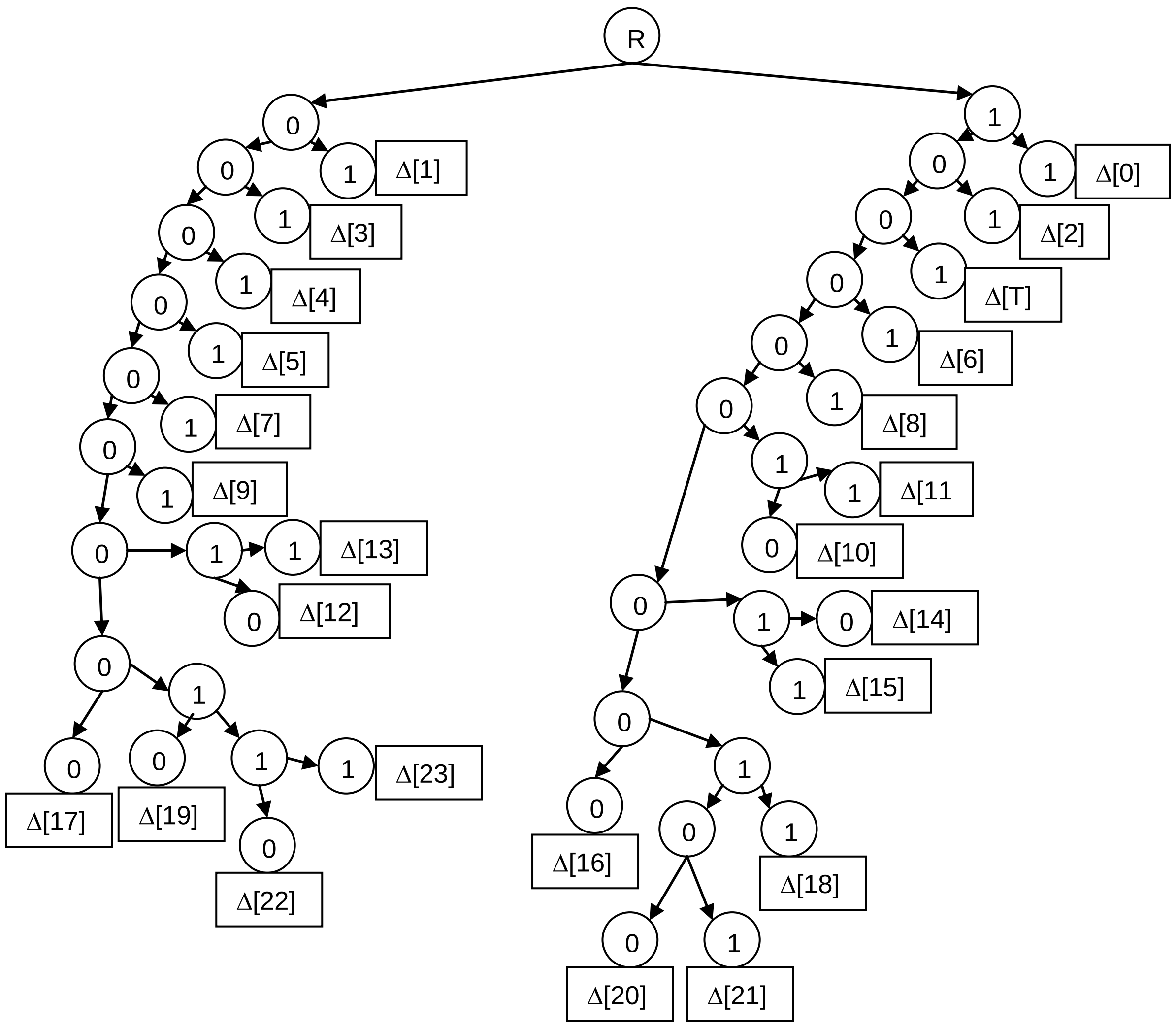
However, in order to best optimise the coding, a symbol must be used that indicates the number of bits (information) that follow.
The algorithm is based on a simplified Huffman coding with the starting hypothesis that the data are subject to little or no change between two measurements. In other words, this means that the probability that the next datum in the buffer has the same value is greater than the next datum varying by one point. Similarly, the probability of finding a datum that varies only by one point is greater than the datum varying by two points, etc...
It is also anticipated that if the datum varies too greatly, the new datum is stored as a raw datum with a specific coding symbol. In this case, a one-off compression ratio of less than 1 is obtained.
This also means that if the stored data which follows in a buffer varies excessively, the compression is ineffective.

4.2 The probability tree

As mentioned above, the probability tree is pre-established in order to simplify (and even optimise) the coding and the compressed code.
Note that this is true only if there is little data to be compressed. If the volume of data is significant, a different, more effective solution would have to be preferred to this type of compression.

Another important point is that the probability tree is the same regardless of the format of the data to be compressed.
Moreover, the tree defined hereafter is designed using a pre-established tree and is therefore certainly not fully optimised for a particular data buffer.
IMPORTANT: this probability tree is unique regardless of the format of the source data. This point is important because as it is known, it is not transmitted in order to further reduce the size.

4.2.1 Detail of the probability tree



4.2.1.1 Probability code

This probability tree gives us the following for a code:

- "0b11" ➔ No difference
- "0b01" ➔ Difference of +/- 1.
- "0b101" ➔ Difference between +/- 2 and 3
- "0b001" ➔ Difference between +/- 4 and 7
- "0b1001" ➔ A new raw value
- "0b0001" ➔ Difference between +/- 8 and 15
- "0b00001" ➔ Difference between +/- 16 and 31

- "0b10001" ➔ Difference between +/- 32 and 63
- "0b000001" ➔ Difference between +/- 64 and 127
- "0b100001" ➔ Difference between +/- 128 and 255
- "0b0000001" ➔ Difference between +/- 256 and 511
- "0b10000010" ➔ Difference between +/- 512 and 1023
- "0b10000011" ➔ Difference between +/- 1024 and 2047
- "0b000000010" ➔ Difference between +/- 2048 and 4095
- "0b000000011" ➔ Difference between +/- 4096 and 8191
- "0b100000010" ➔ Difference between +/- 8192 and 16383
- "0b100000011" ➔ Difference between +/- 16384 and 32767
- "0b100000000" ➔ Difference between +/- 32768 and 65535
- "0b000000000" ➔ Difference between +/- 65536 and 131071
- "0b1000000011" ➔ Difference between +/- 131072 and 262143
- "0b0000000010" ➔ Difference between +/- 262144 and 524287
- "0b10000000100" ➔ Difference between +/- 524288 and 1048575
- "0b10000000101" ➔ Difference between +/- 1048576 and 2097151
- "0b00000000110" ➔ Difference between +/- 2097152 and 4194303
- "0b00000000111" ➔ Difference between +/- 4194304 and 8388607

Thus probability codes represent the following fields in the memory:
0b11
0b01S
0b101SX
0b001SXX
0b1001XX….XX (depending on the format of the base datum)
0b0001SXXX
0b00001SXXXX
0b10001SXXXXX
0b000001SXXXXXX
0b100001SXXXXXXX
0b0000001SXXXXXXXX
0b10000010SXXXXXXXXX
0b10000011SXXXXXXXXXX
0b000000010SXXXXXXXXXXX
0b000000011SXXXXXXXXXXXX
0b100000010SXXXXXXXXXXXXX
0b100000011SXXXXXXXXXXXXXX
0b100000000SXXXXXXXXXXXXXXX
0b000000000SXXXXXXXXXXXXXXXX
0b1000000011SXXXXXXXXXXXXXXXXX
0b0000000010SXXXXXXXXXXXXXXXXXX
0b10000000100SXXXXXXXXXXXXXXXXXXX
0b10000000101SXXXXXXXXXXXXXXXXXXXX
0b00000000110SXXXXXXXXXXXXXXXXXXXXX
0b00000000111SXXXXXXXXXXXXXXXXXXXXXX

With S as the sign and XX..XX a binary value.
If S = 1, then the delta is negative, otherwise it is positive.

4.2.2 Bit size of encoding

Bit encoding allows for variable sizes, such as:

1) 0b11 has a length of 2 bits
2) 0b01 ... has a length of 3 bits
3) 0b101 … has a length of 5 bits
4) 0b001 … has a length of 6 bits

5) 0b1001 … has a length of 4bits + format of the datum
6) 0b0001 … has a length of 8 bits
7) 0b00001 … has a length of 10 bits
8) 0b10001 … has a length of 11 bits
9) 0b000001 … has a length of 13 bits
10) 0b100001 … has a length of 14 bits
11) 0b0000001 … has a length of 16 bits
12) 0b10000010 … has a length of 18 bits
13) 0b10000011 … has a length of 19 bits
14) 0b000000010 … has a length of 21 bits
15) 0b000000011 … has a length of 22 bits
16) 0b100000010 … has a length of 23 bits
17) 0b100000011 … has a length of 24 bits
18) 0b100000000 … has a length of 25 bits
19) 0b000000000 … has a length of 26 bits
20) 0b1000000011 … has a length of 28 bits
21) 0b0000000010 … has a length of 29 bits
22) 0b10000000100 … has a length of 31 bits
23) 0b10000000101 … has a length of 32 bits
24) 0b00000000110 … has a length of 33 bits
25) 0b00000000111 … has a length of 34 bits

For example, one can see that the best compression ratio possible for an initial datum stored over 16 bits is a factor of 8 (if the datum does not change).
The effective compression limit (greater than 1) is when the data vary by less than +/- 255.

Note that since it is necessary to re-record a new raw value, 4 + 16 = 20 bits are required. We use the probability codes up to "0b10000011" because it only stores 19 bits in the memory, and is therefore more effective.
The same applies to the other compression formats:
- 0b10001 (i.e. 11 bits) for an 8-bit data format
- 0b10000011 (i.e. 19 bits) for a 16-bit data format
- 0b00000000111 (i.e. 34 bits) for a 32-bit data format.

4.2.3 Bit storage

Bits are stored in the memory in big endian order, i.e. with the most significant bit first.
Upon decompression, data bits are stored in little endian order, i.e. with the least significant first.
If we consider the following sequence: 202, 197, 198, 197, 196, 204, i.e. the following table in 16-bit little endian format:
0xCA, 0x00; 0xC5, 0x00; 0xC6, 0x00; 0xC5, 0x00; 0xC4, 0x00; 0xCC, 0x00.

After running it through the compression algorithm, we have: 0x09, 0x30, 0xC5, 0xCA, 0x46, 0x00.

Further analysis provides the following sequence:
0b00001001, 0b00110000, 0x11000101, 0b11001010, 0x01000110, 0x00000000
In violet the code: 0b1001 i.e. an absolute value.
In green the datum: 0b00000000,11001010
In orange the next code: 0b001
In blue the associated datum 0b101, here the delta is the negative of 1 + 4, i.e. -5
In red the next code 0b01
In grey the associated datum 0b0, i.e. +1
In purple the next code, i.e. 0b01
In orange the next associated datum 0b1, i.e. -1
In dark blue the next code: 0b01
In dark green the next associated datum: 0b1, i.e. -1

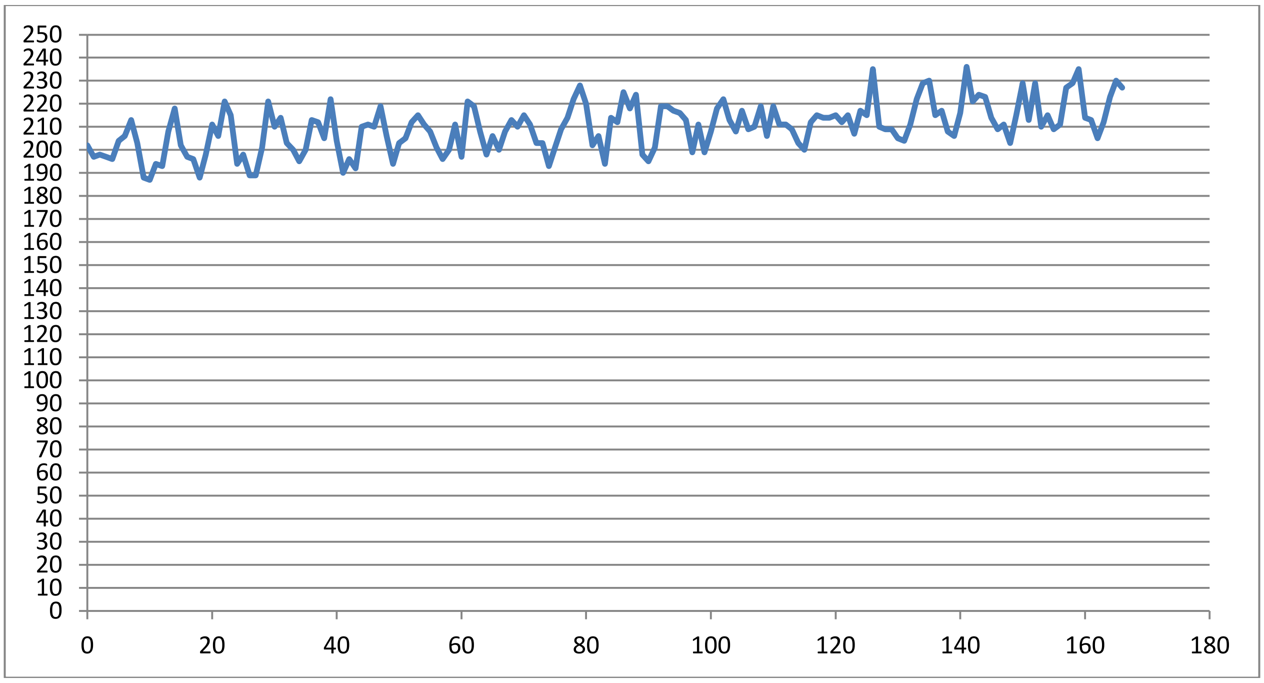
In blue the next code, i.e.: 0b0001
In red the associated datum, i.e.: 0b0000 i.e. +8

Hence, the original suite is reconstituted:
Namely 202 (=0x00CA); 197 (=202-5); 198 (=197+1); 197 (=198-1); 196 (=197-1); 204 (196+8)
Our initial data are indeed recovered.

4.2.4 Absolute compression limits

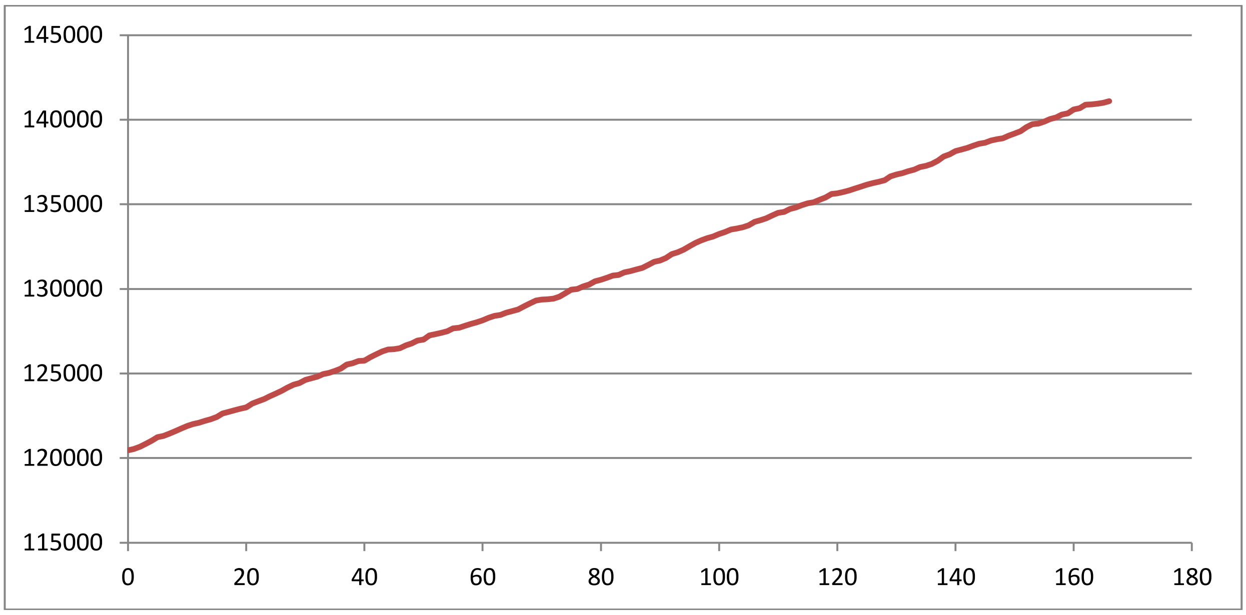
For data:
- In 8-bit format, the maximum compression ratio is < 4.
- In 16-bit format, the maximum compression ratio is < 8.
- In 32-bit format, the maximum compression ratio is < 16.

4.3 Compression example



A pseudo-temperature curve, as below, which comprises 167 measurements in 16-bit format.
Namely, a buffer of 167 * 2 = 334 bytes gives a compression result of 142 bytes, i.e. a factor of 2.3 approximately.

If we now consider the example of an S0 type counter over 32 bits, with a curve of this type



There are also 167 measurements but over 32 bits, i.e. 668 bytes. The compression gives a result of 278 bytes, i.e. a gain in size by a factor of approximately 2.4.