Run Length Encoding (RLE)

Run Length Encoding is a very simple method of encoding that can either drastically reduce your file size or double its size depending on the redundancy of your data.

RLE works best when your data has a lot of repetition of single values that occur right after each other.

An example of such is: AAAAA

The string can instead be represented as “A 5 times”.

Let’s use the string ABAABACBCCBABC in a real example to show how you would compress this. We will use 1 byte for the length of repetition

|  |  |  |
| --- | --- | --- |
| Data | Amount of times it appears | Final encoding |
| A | 1 | 0x01 0x65 |
| B | 1 | 0x01 0x66 |
| A | 2 | 0x02 0x65 |
| B | 1 | 0x01 0x66 |
| C | 1 | 0x01 0x67 |
| B | 1 | 0x01 0x66 |
| C | 2 | 0x02 0x67 |
| B | 1 | 0x01 0x66 |
| A | 1 | 0x01 0x65 |
| B | 1 | 0x01 0x66 |
| C | 1 | 0x01 0x67 |

Final Output:

0x01 0x65 0x01 0x66 0x02 0x65 0x01 0x66 0x01 0x67 0x01 0x66 0x02 0x67 0x01 0x66 0x01 0x65 0x01 0x66 0x01 0x67

The original output:

0x65 0x66 0x65 0x65 0x66 0x65 0x67 0x66 0x67 0x67 0x66 0x65 0x66 0x67

As you can see, RLE is not very efficient in this case as it adds 8 bytes onto the original output.

In a different example such as AAAABBBBCCCC however, the final output of the encoded string would be this:

0x04 0x65 0x04 0x66 0x04 0x67

Where our original output would be this:

0x65 0x65 0x65 0x65 0x66 0x66 0x66 0x66 0x67 0x67 0x67 0x67

In this example, we save 6 bytes. We cut our data in half.

The general pattern for good compression would be that sorted data with repeated values will yield high compression ratios.