Brotli Compression Algorithm
motivation

Main differences with deflate

Areas

- entropy code reuse
- window size
- context modeling
- literal count coding
- distance cache
- match lengths
- block end codes

Entropy code reuse

Entropy code requires some bytes to be described.

Deflate: the whole entropy code has to be described before the data.

Brotli: a block may re-use past entropy codes (within the same metablock).

Window size

Deflate: deflate can remember past 32 kB.

Brotli: brotli can refer back to past 16 MB.

TBD: freeze a resource sweet spot for the specification, perhaps just 4 MB window.

Context modeling

Deflate: each literal byte is coded independently.

Brotli: literal bytes can have an entropy code that depends on one, two or three last bytes.

Literal count coding

Deflate: every symbol has a probability for literals, backward copy length, or end code. These are entropy coded individually for every symbol.

Brotli: the number of pure literals is codified, next the literals with a literal only entropy code.

Distance cache

Deflate: each distance codified separately.

Brotli: backward references can be described in the form of past four distances

```
...abcdeX12345...
...abcdeY12345...
```

Match lengths

Deflate: match lengths of 3–258. Match lengths codified in the same entropy code as literals.

Brotli: match lengths of 2–enough. Match lengths codified in joint entropy code with literal lengths.

Block end code

Deflate: each symbol can be the block end code, stealing some efficiency from the entropy when unique symbol count is small.

Brotli: block length is codified in the beginning of the block.