Serializing Thing Descriptions

Objectives

→ **Represent** TD information

What do we want **optimize**?
— compactness?
— usefulness for processing?

What are our assumptions about the devices…
— holding TDs?
— processing TDs?

Maximizing compactness

- Use traditional data compression
  - e.g., deflate (algorithm behind gzip)
  - LZ77: exploit bytewise sharing over window
- no relation to semantics
- compression requires view of window
- decompression is a copying process

Choosing a compressor

- LZ77: Sharing via <distance, length>
- Need entropy encoder, too
  - Huffman, Arithmetic
  - Recent advances

Classical algorithms:

- Deflate (RFC 1951 from 1996): LZ77 + Huffman
- LZ4 (low complexity)

Maximizing processability

- Keep data in a processable form
- Minimize the need for copying
  - avoid escaping
  - avoid piecing together
- Maybe avoid a classical compressor
  - "Semantic Sharing"?

Experiment

`wot-thing-description/test-bed/data/plugfest/2017-05-osaka/MyLED_f.jsonld`

- JSON file: 3116
- **JSON no whitespace: 1447**
  - deflate: 323, lz4: 415, lz4hc: 411
- **CBOR: 1210**
  - deflate: 325, lz4: 416, lz4hc: 404
- CBOR packed (semantic sharing only): 793
- CBOR packed (prefix compression, too): 564

Conclusion

- Packing (exploiting structural sharing)
  - maintains processability
  - saves ~ 1/3 (implementation not yet complete)
- Prefix sharing helps with URLs, another 20 %
  - but reduces processability
- Could further improve by adding static dictionary
  - In the example: 119 bytes of mostly static data:

```
["name","@type","links","application/json","outputData","mediaType","href",
 {"valueType":{"type":"number"}},["Property"],"writable","valueType","type"]
```