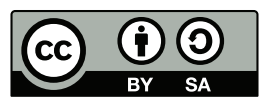
Variable Length Encoding

Roland Hieber

January 14, 2016



Problem

- “I want to send a number.”
- “Well, just use 8/16/32/64 bits.”
- But what if. . .
  - the 64 bit range is simply not enough?
  - the required range might change in the future?
  - the required range is unknown?
  - bandwidth is low/expensive, don’t want to send too much unneeded data

Solutions

- UTF-8
  - ISO/IEC 10646:2014
- SDNV
  - Self-Delimiting Numeric Values
  - RFC 5050, Section 4.1

UTF-8

- encodes Unicode characters up to 0x7FFFFFFF
- values 0x00–0x7F are equal to 7-bit ASCII

Encoding Rules

- Bit 8 is 0: only one byte in total.
- Bit 8 is 1 $\Rightarrow$ more bytes to follow
  - Bit 8–7 are 10: This is a following byte.
  - Bit 8–6 are 110: This is a leading byte, one byte follows
  - Bit 8–5 are 1110: ...two bytes follow
  - Bit 8–4 are 11110: ...three bytes follow
  - ...
- "Free" bits are used to encode the value.

UTF-8: Examples

Hex:	0x13
Binary:	00010011
UTF-8:	00010011

Hex:	0x1337
Binary:	0001 001100 110111
UTF-8:	11100001 10001100 10110111

Hex:	0x31337
Binary:	000 110001 001100 110111
UTF-8:	11110000 10110001 10001100 10110111

SDNV

- specified for up to 64 bit (0xFFFFFFFF)

Encoding Rules

- MSB is `1`: This is a leading byte.
- MSB is `0`: This is a following byte.
- “Free” bits are used to encode the value.

SDNV: Examples

Hex: 0x80
Binary: 1 0000000
SDNV: 10000001 00000000

Hex: 0x1337
Binary: 0100110 0110111
SDNV: 10100110 00110111

Hex: 0x31337
Binary: 1100 0100110 0110111
SDNV: 10001100 00100110 00110111

Comparison

UTF-8

- (+) first byte contains number of total bytes
  - (–) what about missing follow bytes?
- sync pattern: `0xxxxxxx` or `110xxxxx`

SDNV

- sync pattern: `1xxxxxxx`

both

- (–) susceptible against bit flips/missing bytes