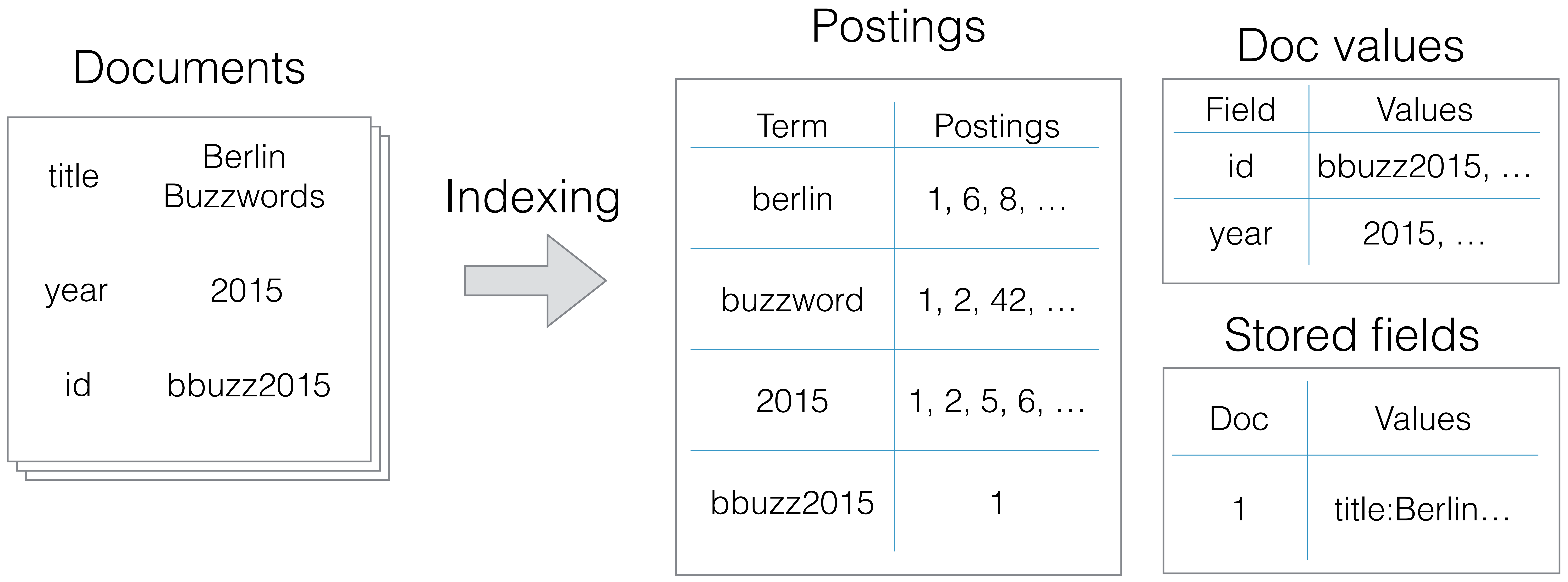
Compression in Lucene

Ryan Ernst

Compression in Lucene



Documents

title	Berlin Buzzwords
year	2015
id	bbuzz2015

Postings

Term	Postings
berlin	1, 6, 8, …
buzzword	1, 2, 42, …
2015	1, 2, 5, 6, …
bbuzz2015	1

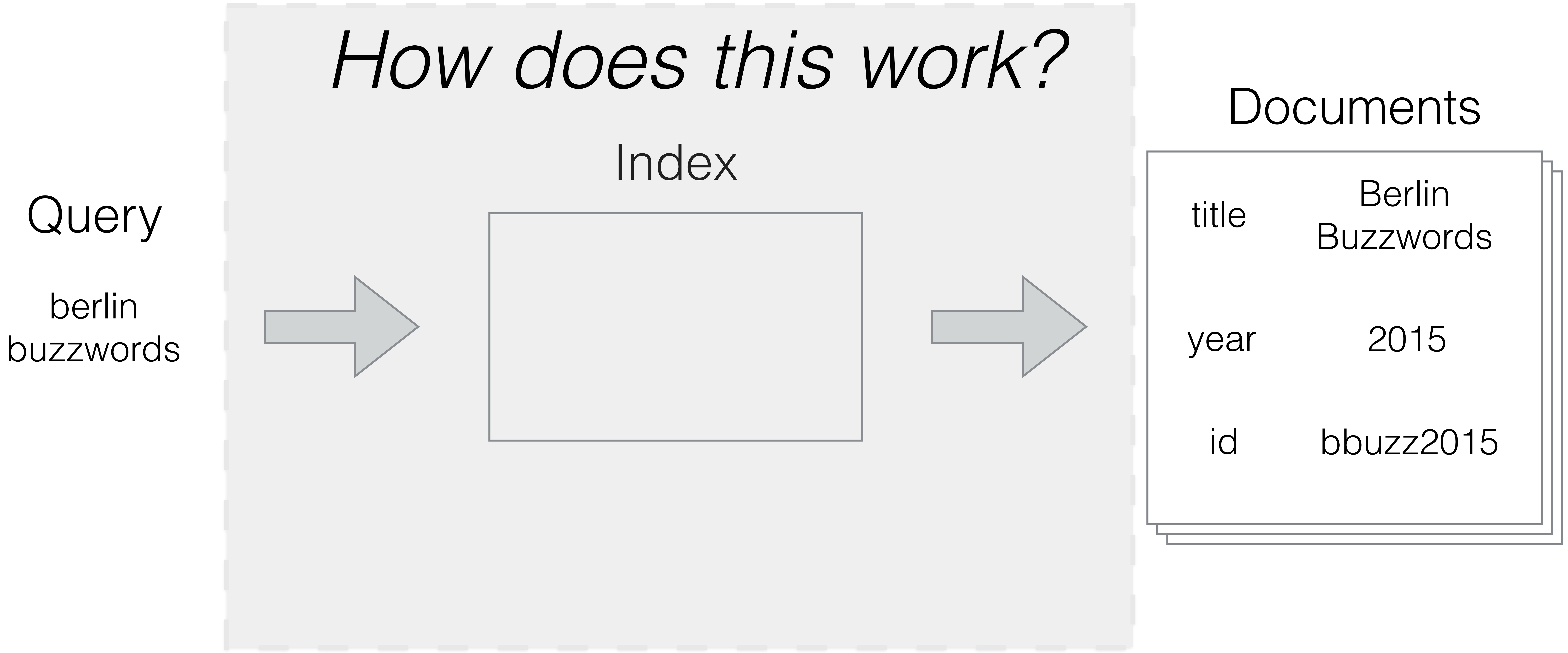
Doc values

Field	Values
id	bbuzz2015, …
year	2015, …

Stored fields

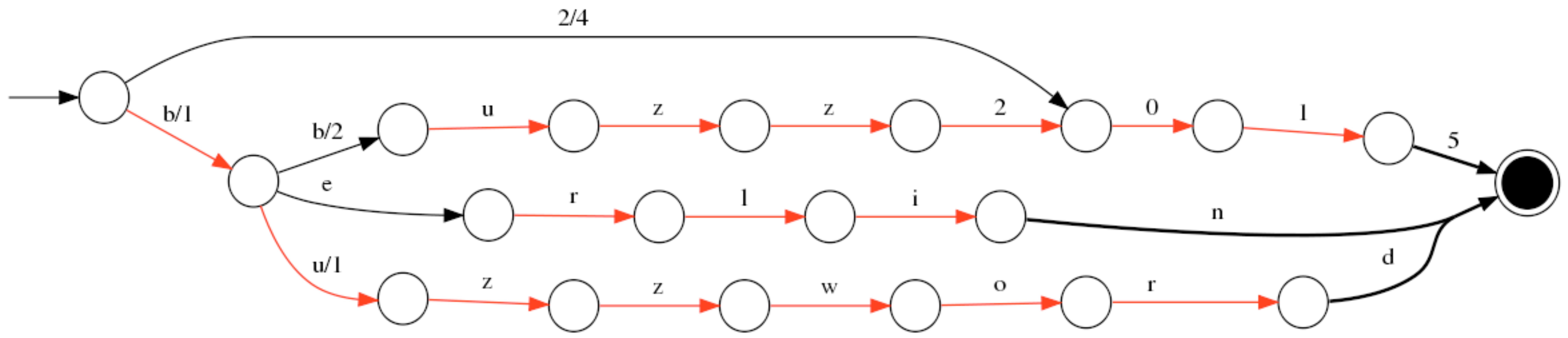
Doc	Values
1	title:Berlin…

Query Example

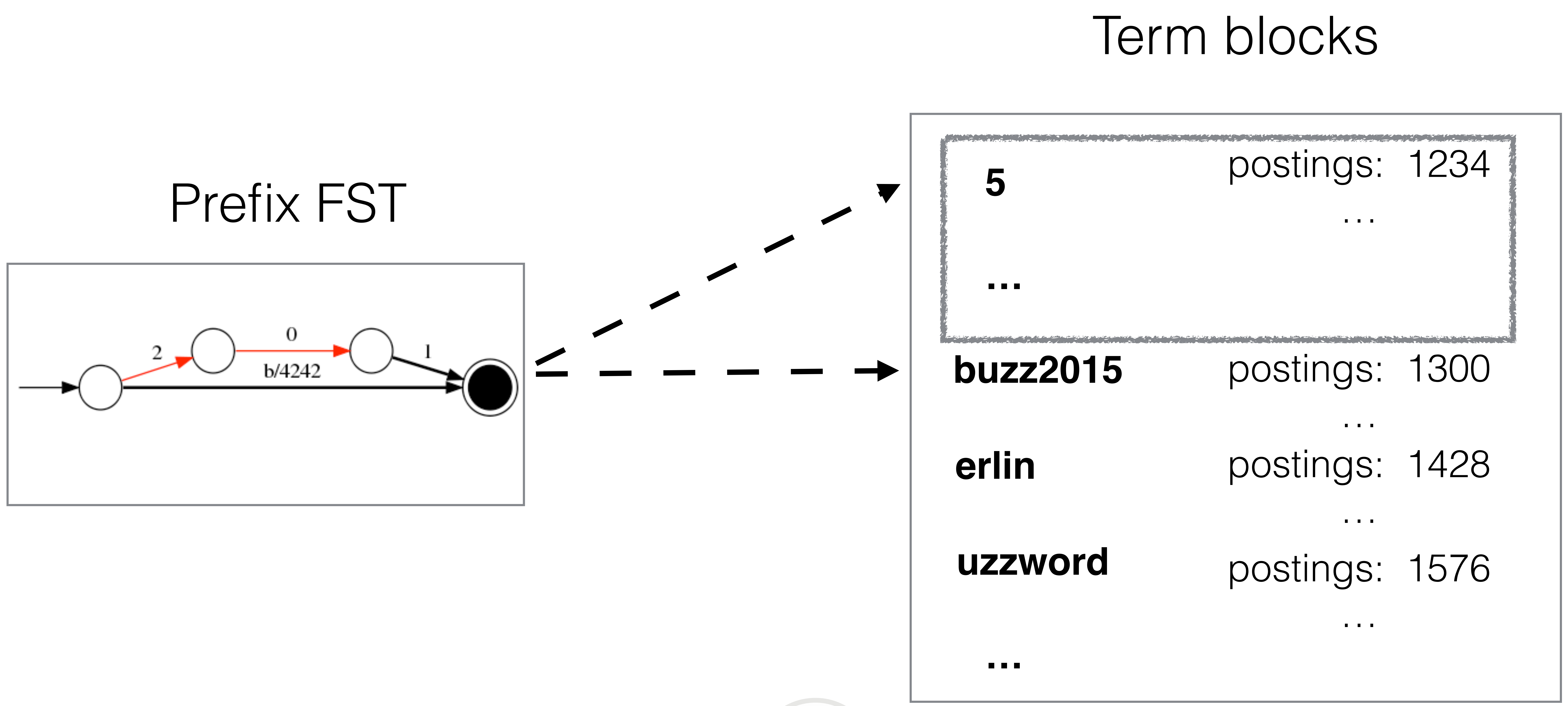


Step 1 - Find the terms

Refresher - FST



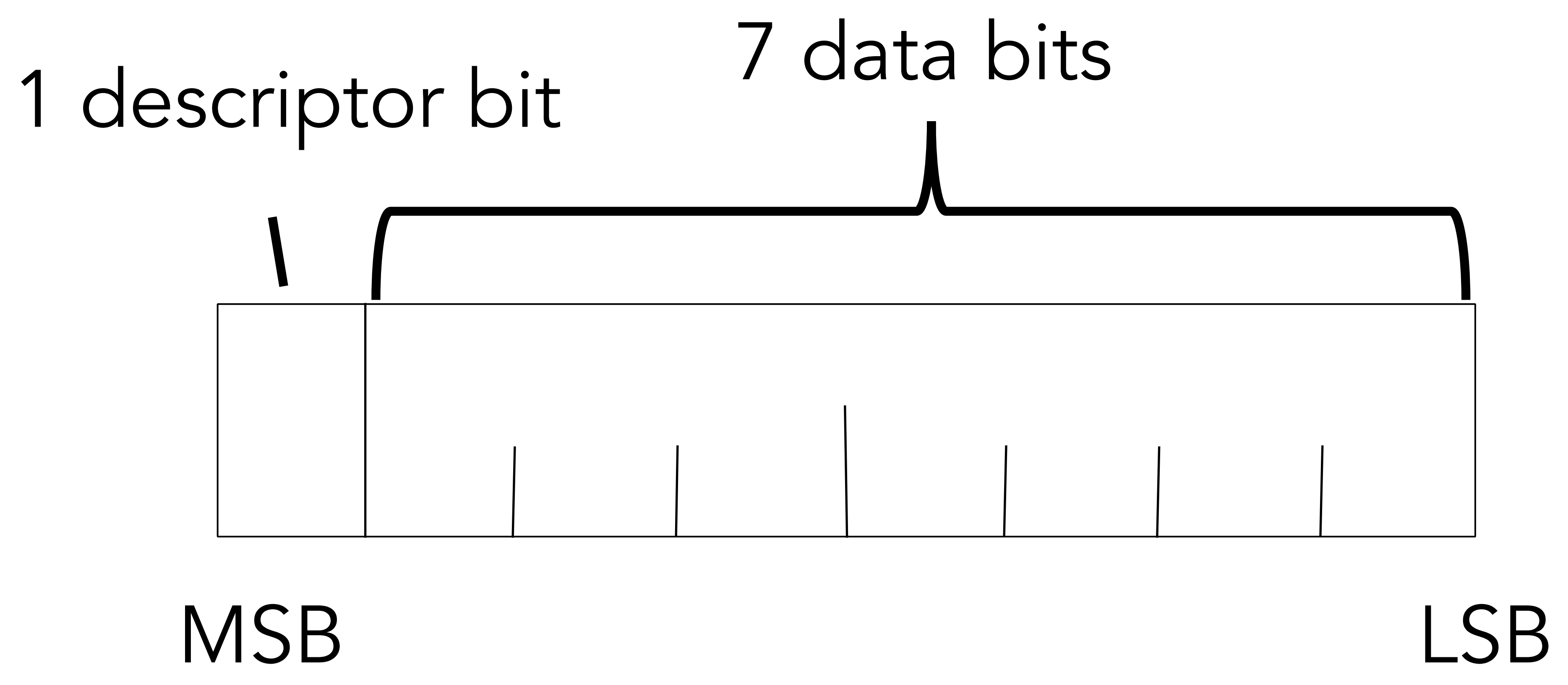
Terms dictionary



5	postings: 1234
	…
…	
buzz2015	postings: 1300
	…
erlin	postings: 1428
	…
uzzword	postings: 1576
	…
…	

Variable length integer (vint)

- 1-5 bytes for a 32 bit unsigned integer
- Use one bit out of every byte to describe continuation – are there more bytes left?



vint examples

Value	Byte 1	Byte 2	Byte 3	Byte 4
1	00000001			
253	10000001	01111101		
10324	10001001	10000000	00000001	
268435455	11111111	11111111	11111111	01111111

Step 2 - Intersect the postings

Postings list for **berlin**

Original docids

1	6	8	...

Deltas

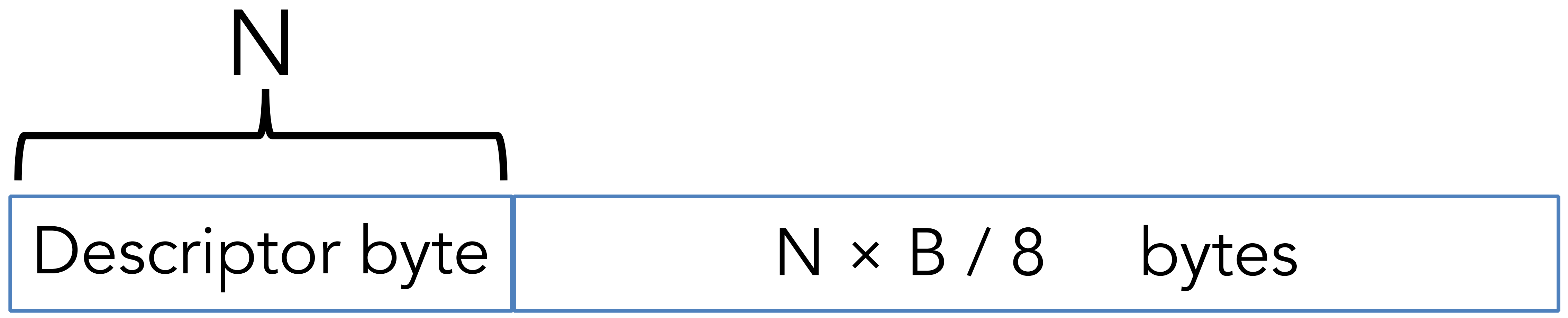
1	5	2	...

vints

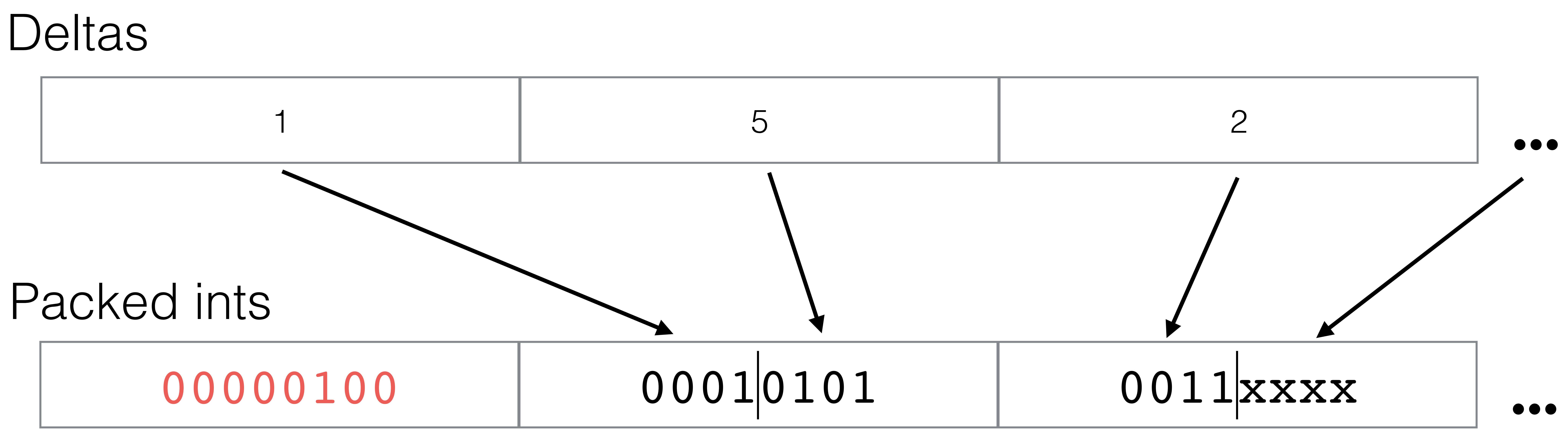
00000001	00000101	00000010	...

Large postings lists - packed ints

- Find minimum number of bits needed for integers within group of B integers
- Pack each integer into a bit field of N bits
  - Fixed width, so we know which byte(s) we need to decode
- Shift/mask to encode and decode



Packed ints postings list



Step 3 - Score the documents

Lucene scoring function

$$score_{q,d} = norm(q) \times \sum_{t \text{ in } q} \sqrt{tf_{t,d}} \times idf_t^2 \times norm(d,field) \times boost(t)$$

Index time normalization

Norms

- Summarizes all index time scoring factors
  - match in shorter field matters more
- 8 bit float by default
  - but API allows for long
- 7 different compression techniques!

Norms - Uncompressed

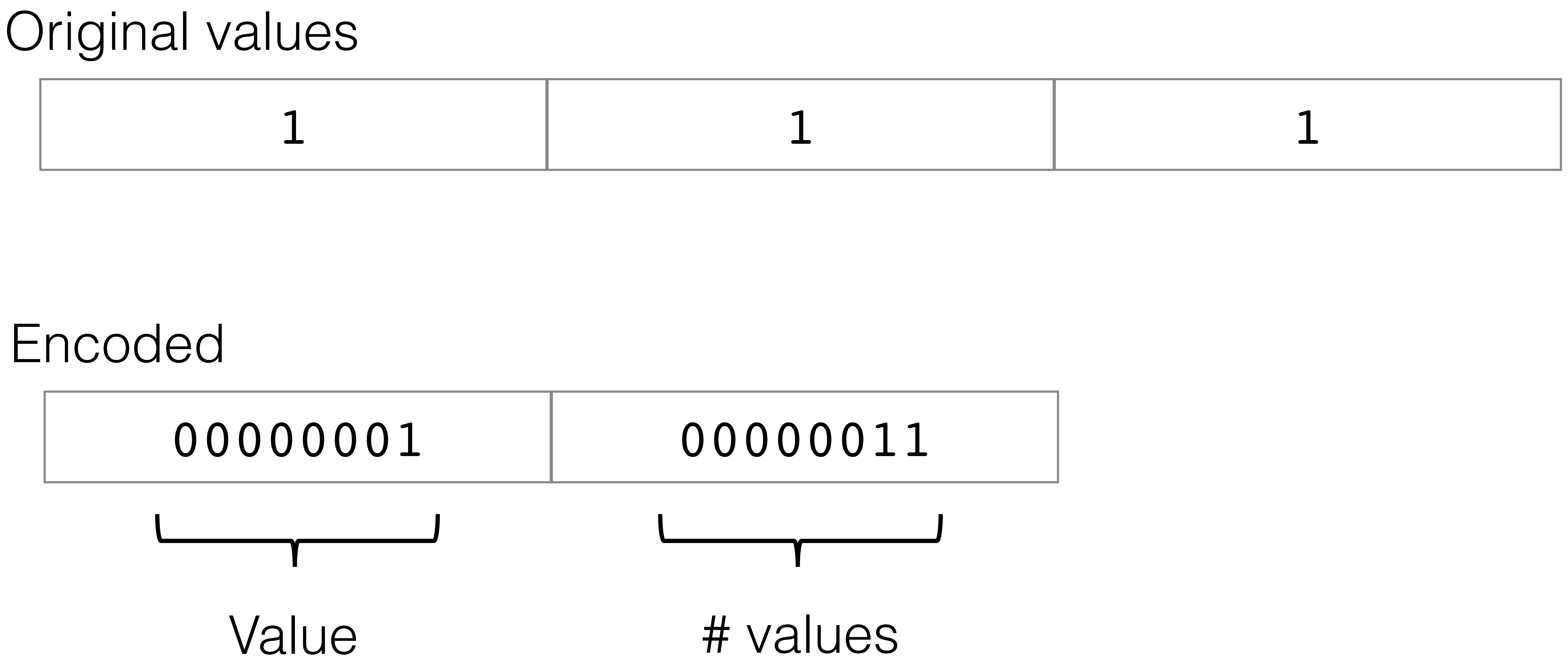
Original values

10	5	20

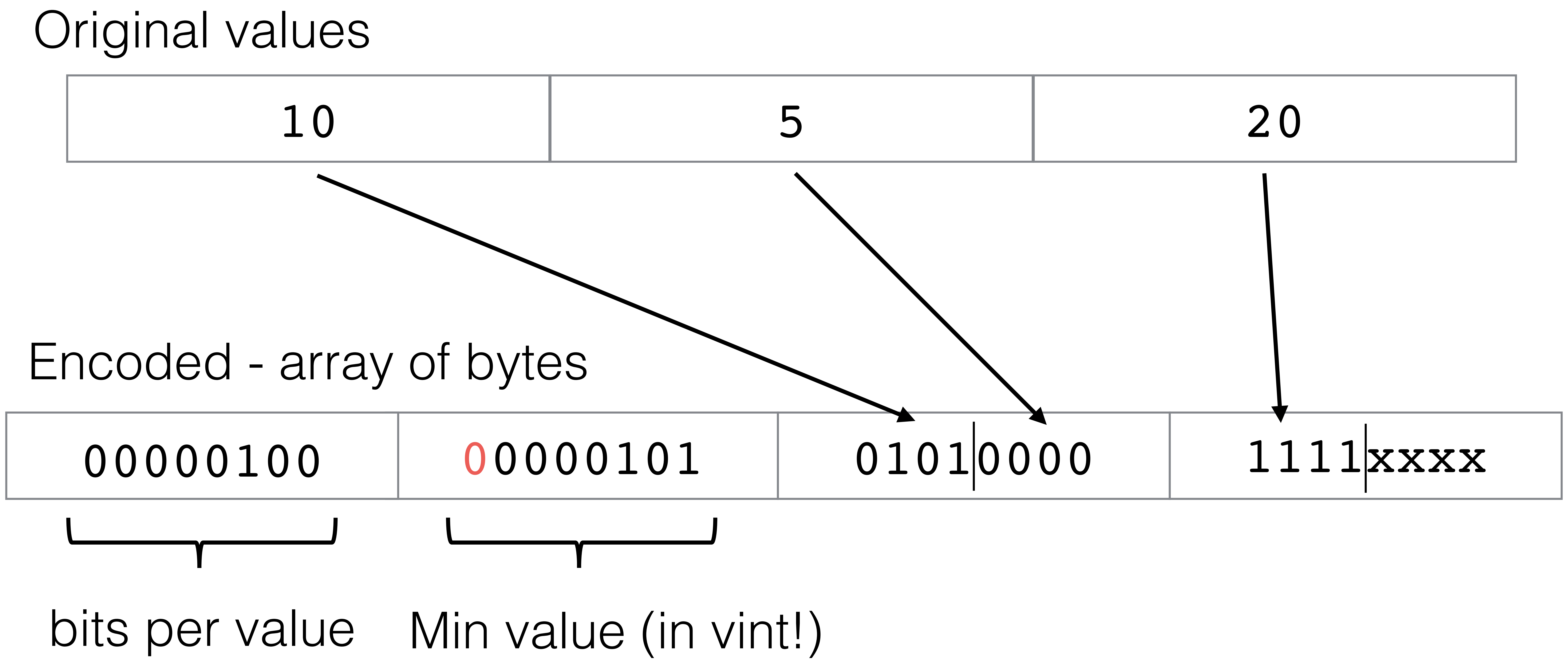
Encoded - array of bytes

00001010	00000101	00010100

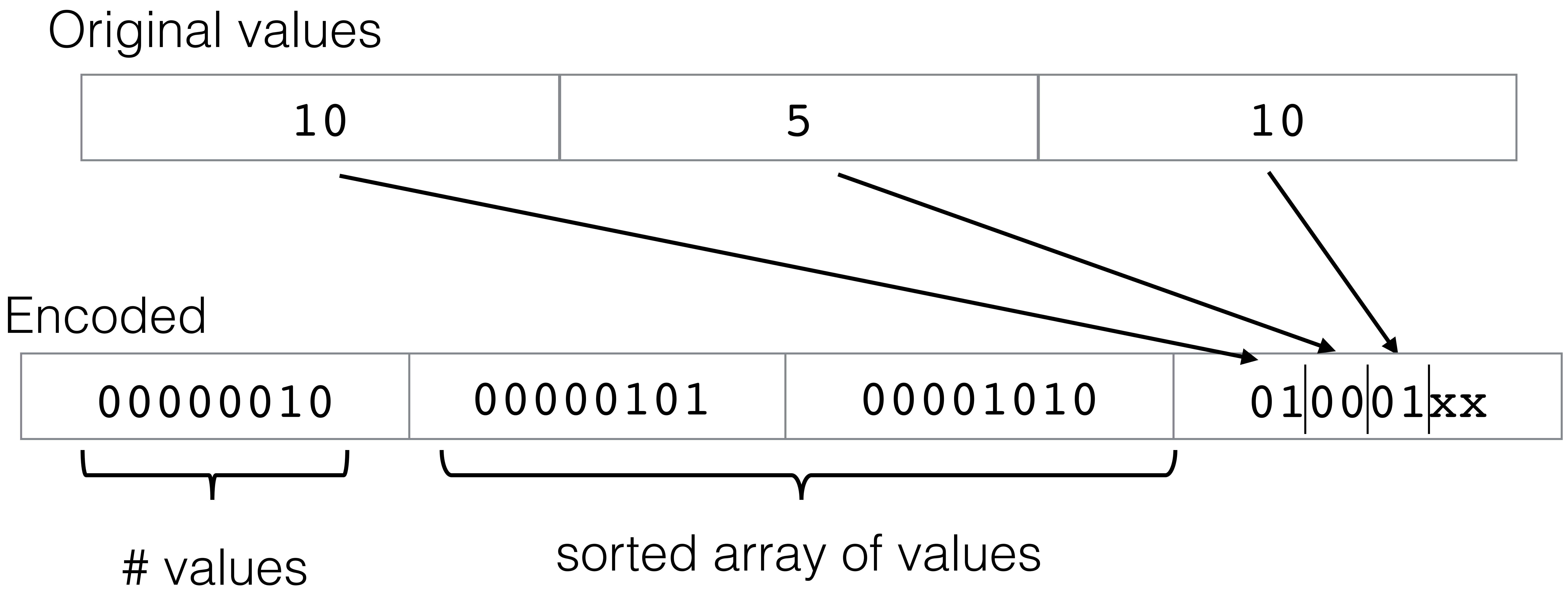
Norms - Constant



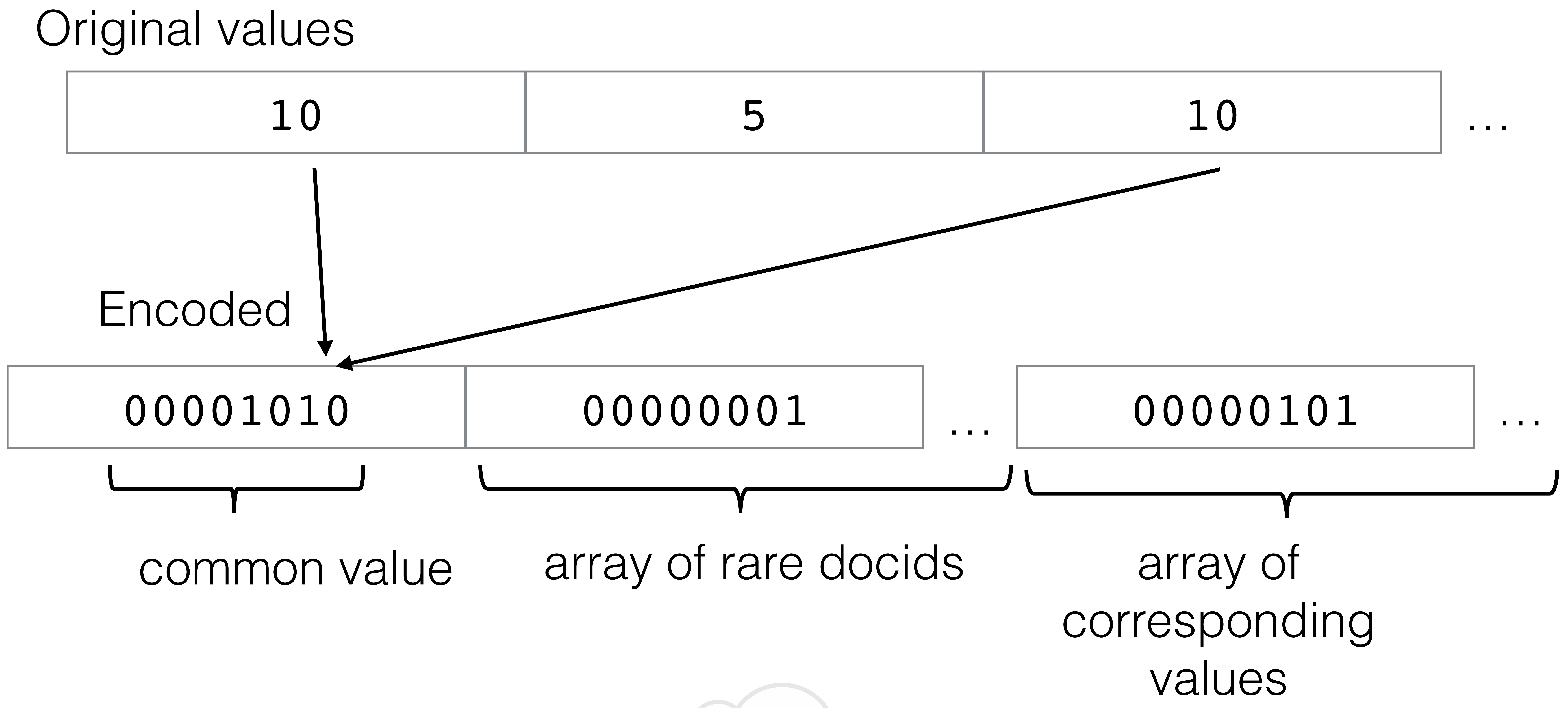
Norms - Delta


Original values
10
5
20
Encoded - array of bytes
00000100
00000101
0101|0000
1111|xxxx
bits per value
Min value (in vint!)

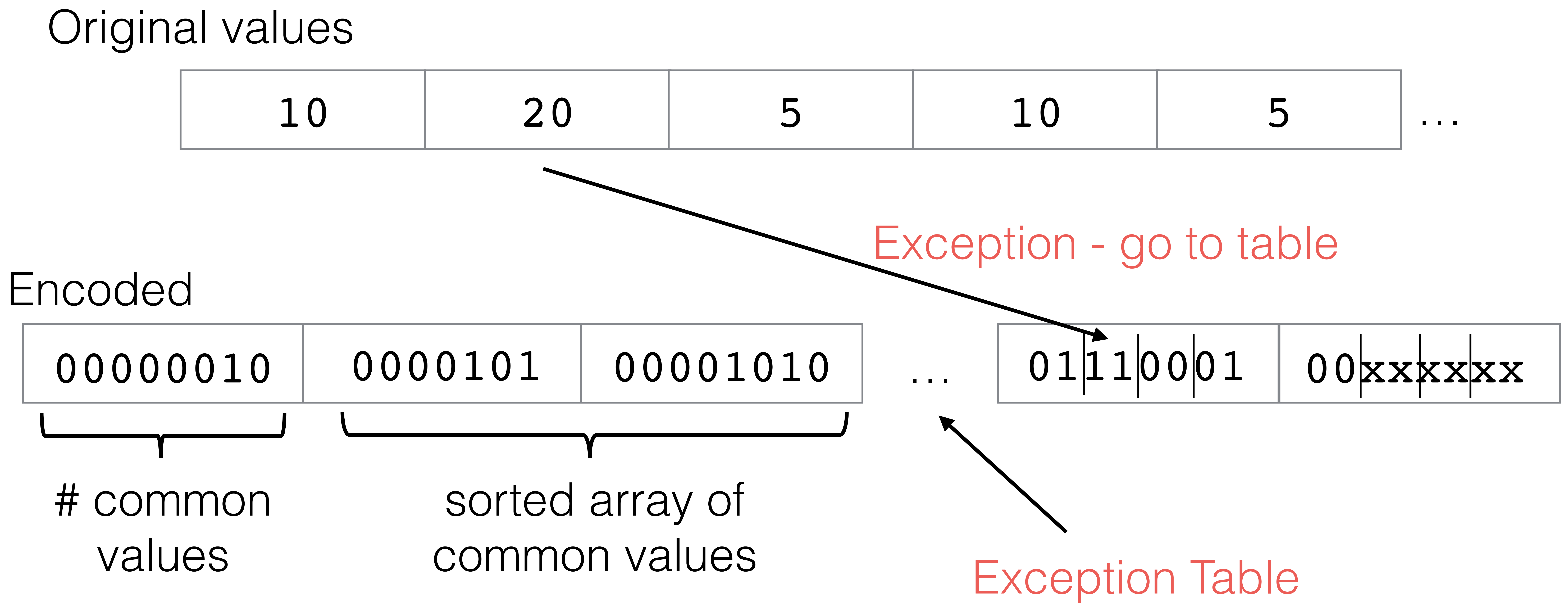
Norms - Table


Original values
10
5
10
Encoded
00000010
00000101
00001010
01|00|01|xx
# values
sorted array of values

Norms - Indirect



Norms - Patched



Additional scoring factors

```
sort = score * popularity
```

Need **doc**id to **values** lookup

Doc values

- 5 types of doc values
  - numeric (single and multi valued)
  - binary
  - binary enumeration (single and multi valued)

Doc values - techniques

- Delta compressed (already seen)
- Table compressed (already seen)
- Const compressed (already seen)
- GCD compressed
- Monotonic compressed
- Prefix compression (same idea as terms dict)

Doc values - GCD

Original values

10	30	20

All divisible by 10!

1	3	2

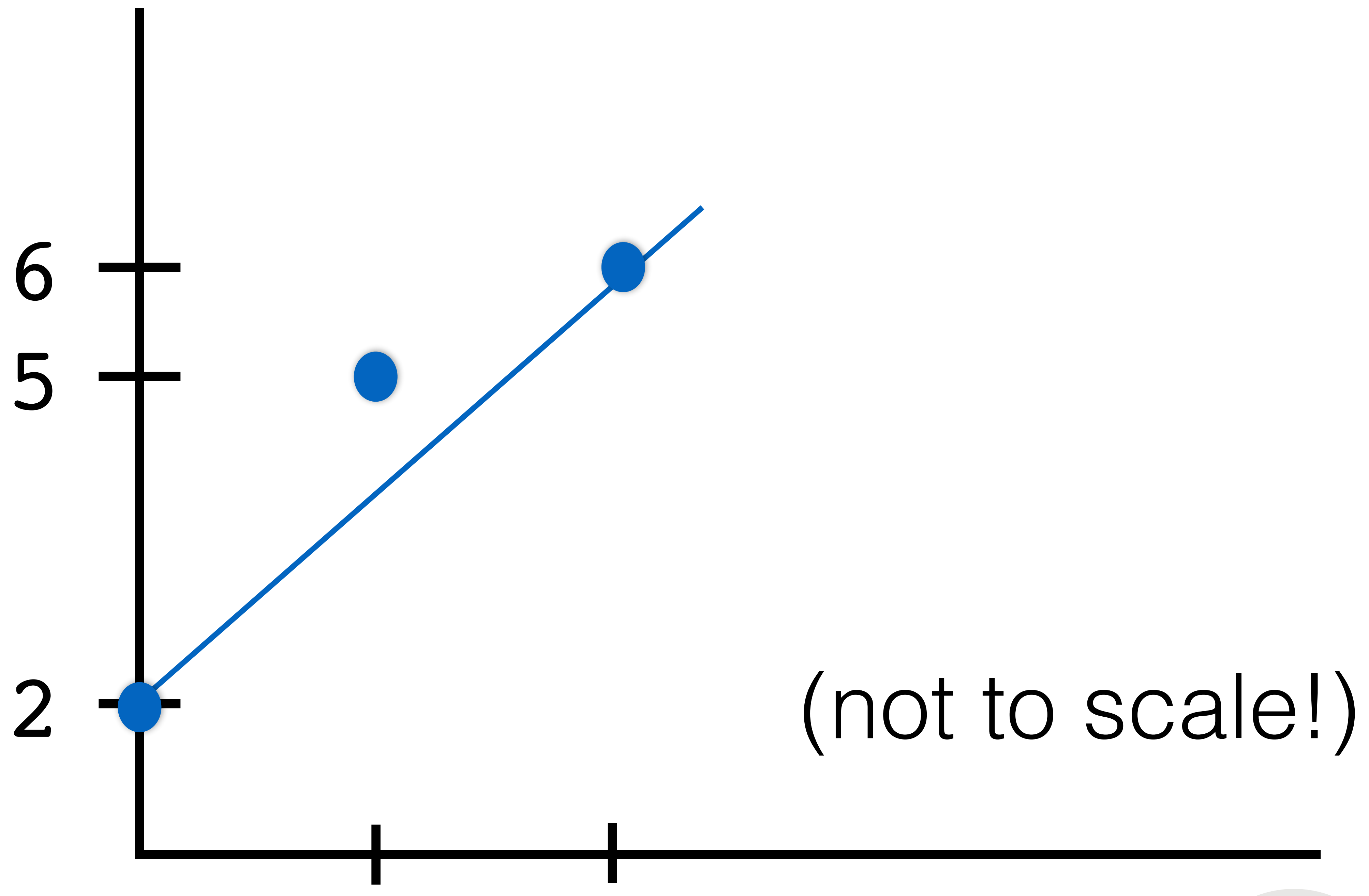
Encoded

$\underbrace{00001010}_{\text{divisor}}$	$\underbrace{00000010}_{\text{bits per value}}$	01|11|10|xx

Doc values - Monotonic

Original values

2	5	6



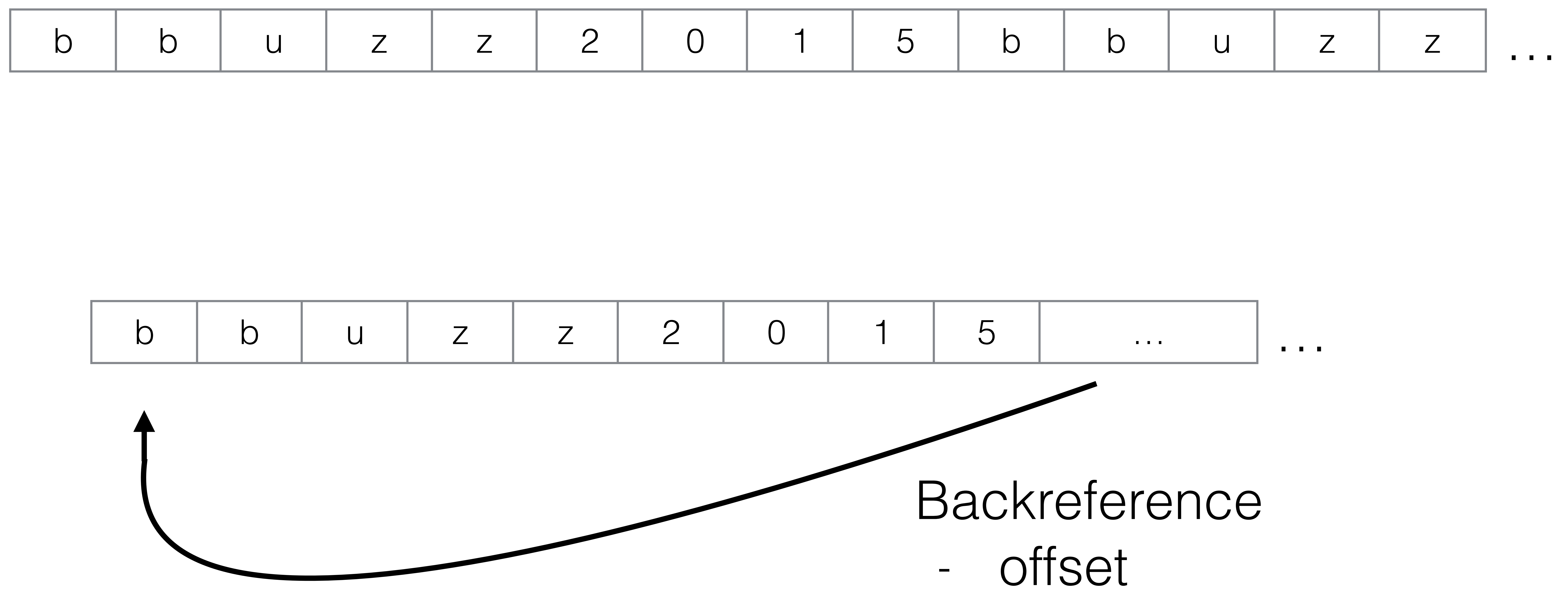
Encoded

00000010	00000010	00|01|00|xx

$$y = 2x + 2$$

Step 4 - Fetch the top documents

Fast compression - LZ4



Backreference
- offset
- length

Best compressed - DEFLATE

- Same basic principle as LZ4 (based on LZ77)
- Second pass compress using huffman codes

Thank you!

@ryanjernst