— As an example, engineers unrelated to the monitoring team have created a new data visualization which will show large sets of horizon charts, which themselves are re-ductions across many time series.

```
Code | Value
------------------
0    | 'kompr'
10   | 'imert om '
11   | 'esjon'

010011
```

Dagsorden:

1. Skaff data

Dagsorden:

1. Skaff data
2. Elektronisk databehandling (Gorilla)

DATA

Bits som trengs for et tall er $\log_2(n)$:

Eksempler:

```
13. Bits: ⌈log₂(13)⌉ = 4. Binært: 1101
17. Bits: ⌈log₂(17)⌉ = 5. Binært: 10001
```

TVETYDIGHET

```
100011101 = [17, 13]
100011101 = [285]
100011101 = [142, 1]
...
```

De mest vanlige typene vi ser i dag er:

```
long   - 64 bits - 8 bytes
double - 64 bits - 8 bytes
int    - 32 bits - 4 bytes
float  - 32 bits - 4 bytes
short  - 16 bits - 2 bytes
byte   -  8 bits - 1 byte*
```

* Den minste adresserbare typen er en 8-bit byte

Eksempel:

```
t1: 3 , bits needed: ceil(log2(13)) = 2, binary: 0011
t2: 15, bits needed: ceil(log2(17)) = 4, binary: 1111
```

Lagre:

```
byte b = (t1 << 4) | t2  # 00111111
```

Hente:

```
byte t1 = b >> 4
byte t2 = b & 0x0F
```

VARIABEL LENGDE KODE

```
i  b  b  b  b  b  b  b
^
indikator
```

VARIABEL LENGDE KODE

```
1 i b b b b b b    b b b b b b b b b
^
indikator
```

VARIABEL LENGDE KODE

```
1 1 i b b b b b    b b b b b b b b b   b b b b b b b b b
^
indicator bit
```

VARIABLE LENGDE KODE

Eksempel:

```
[1, 2, 125, 1576627287446471948, 0, 1]
```

Kode:

```
1 = _0_0000001
2 = _0_0000010
125 = _0_1111101
1576627287446471948 = _11111111   0_0010101 11100001 01001110 00101010 00100011 01101111 01000101 00001100
0 = _0_0000000
1 = _0_0000001
1 = _0_0000001
```

GORILLA

1. The block header stores the starting time stamp, t−1, which is aligned to a two hour window; the first timestamp, t0, in the block is stored as a delta from t−1 in 14 bits.

2. For subsequent time stamps, tn:

- Calculate the delta of delta: D= (tn−tn−1)−(tn−1−tn−2)
- If D is zero, then store a single ‘0’ bit
- If D is between [-63, 64], store ‘10’ followed byt he value (7 bits)
- If D is between [-255, 256], store ‘110’ followed by the value (9 bits)
- if D is between [-2047, 2048], store ‘1110’ followed by the value (12 bits)
- Otherwise store ‘1111’ followed by D using 32 bits

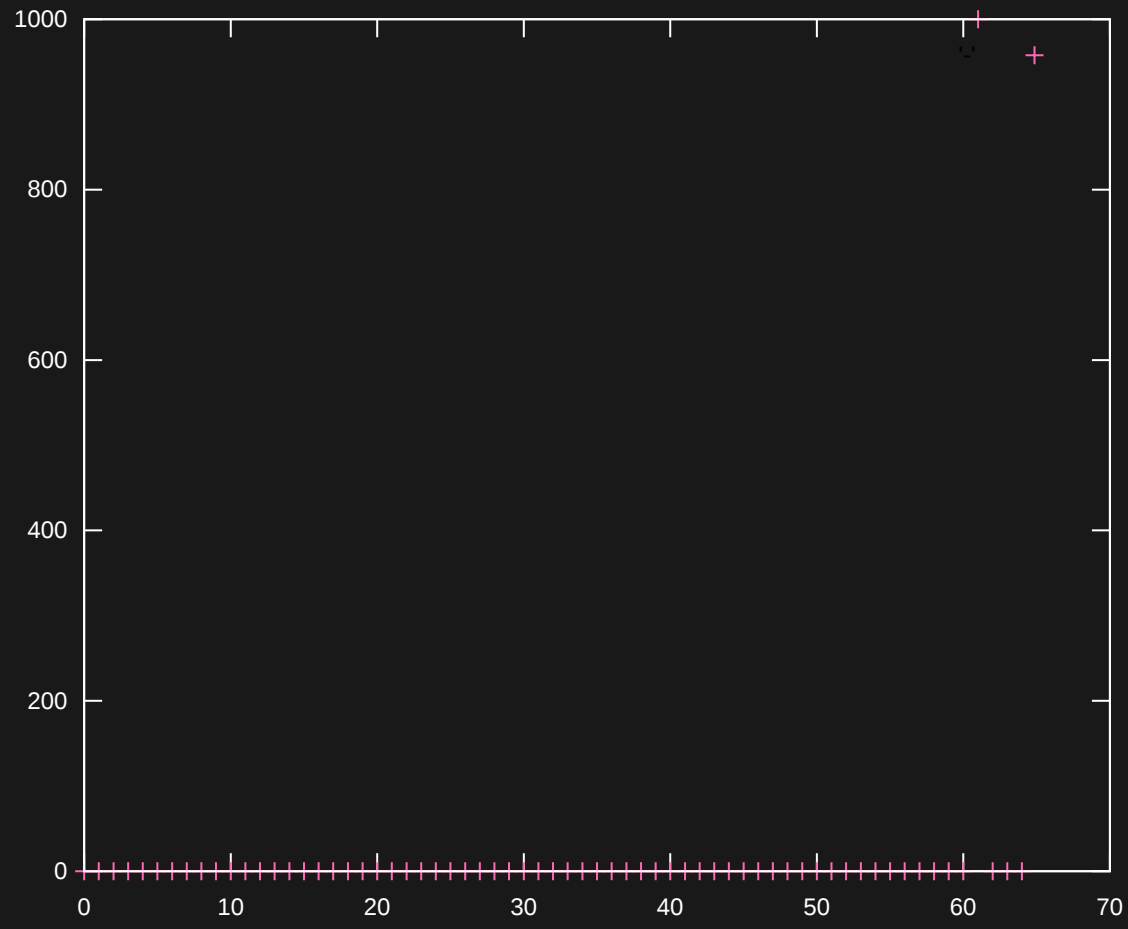
GORILLA

1. The block header stores the starting time stamp, t−1, which is aligned to a two hour window; the first timestamp, t0, in the block is stored as **a delta from t−1 in 14 bits.**
2. For subsequent time stamps, tn:

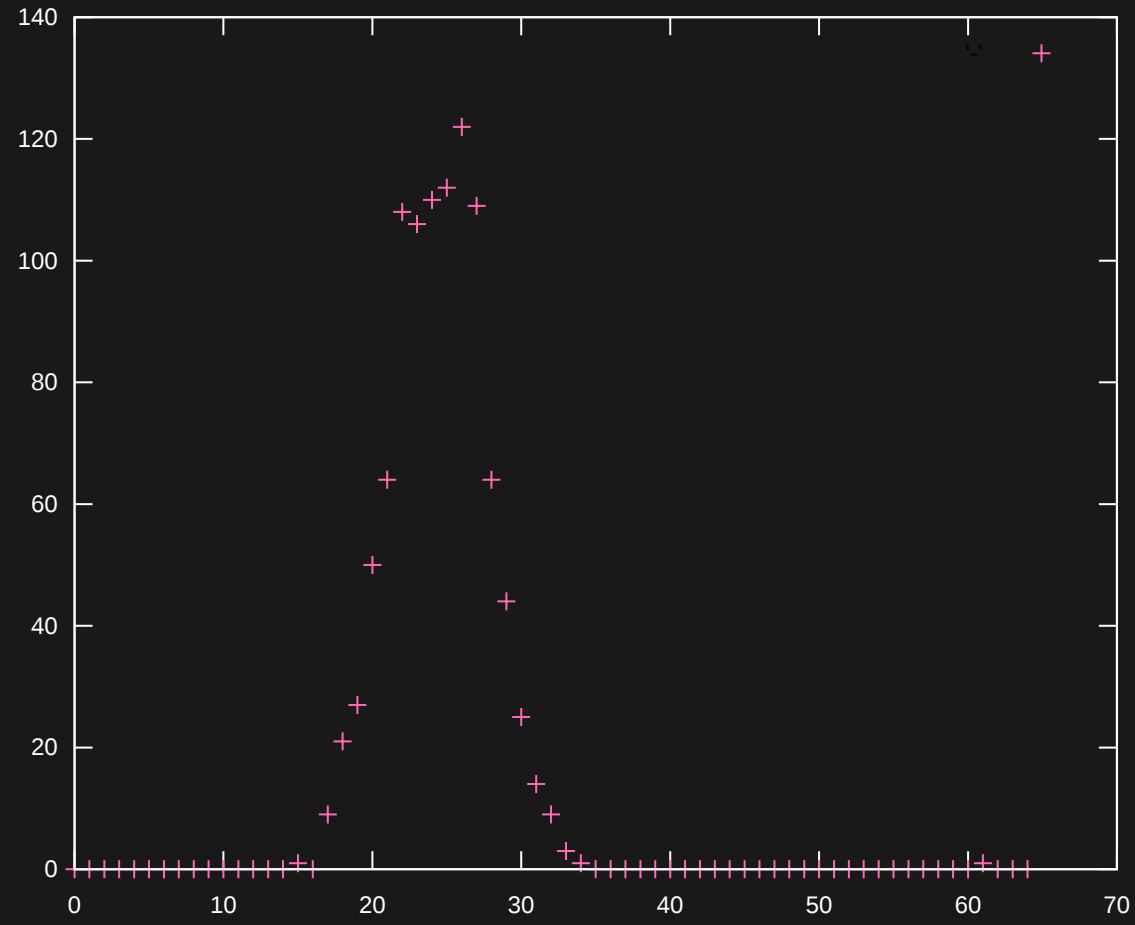
- Calculate the delta of delta: D= (tn−tn−1)−(tn−1−tn−2)
- If D is zero, then store a single ‘0’ bit
- If D is between [-63, 64], store ‘10’ followed byt he value (7 bits)
- If D is between [-255, 256], store ‘110’ followed by the value (9 bits)
- if D is between [-2047, 2048], store ‘1110’ followed by the value (12 bits)

- Otherwise store ‘1111’ followed by D using 32 bits

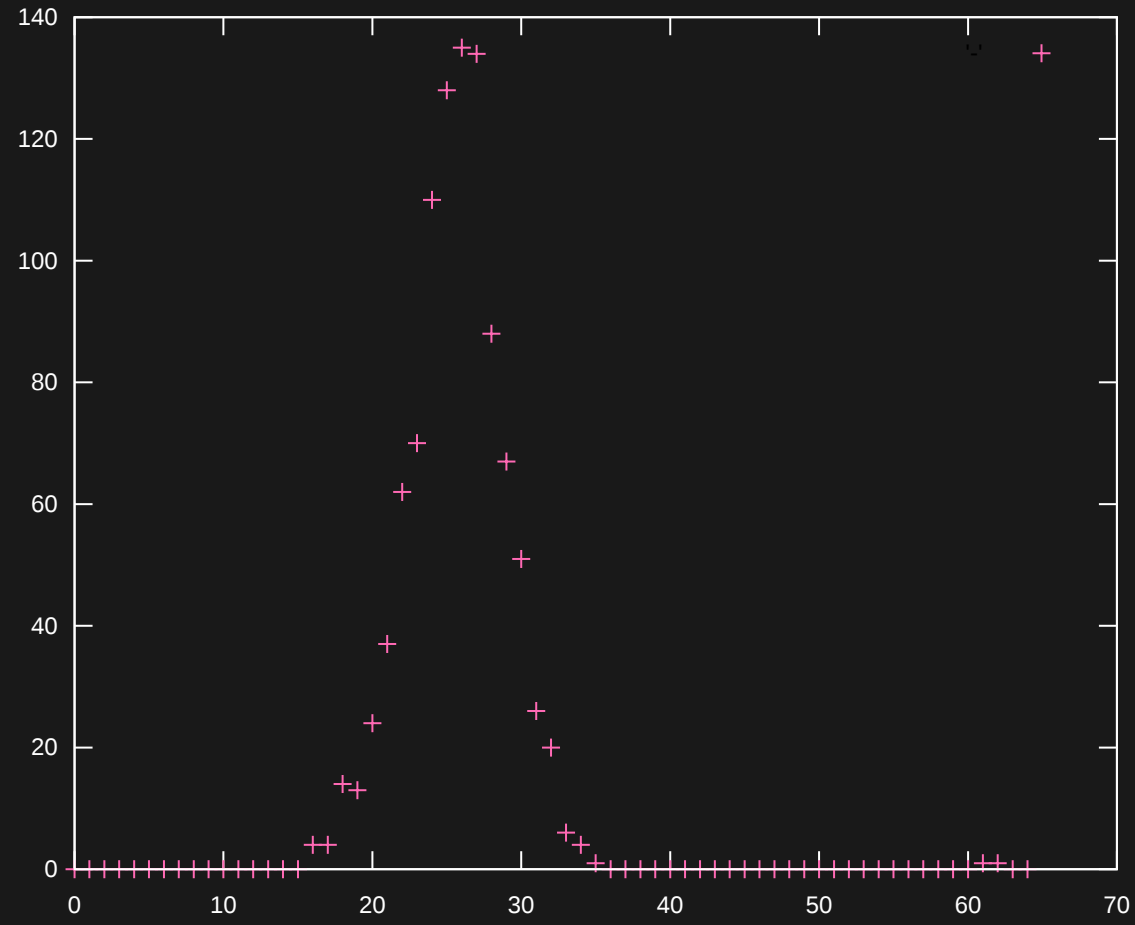
RÅDATA



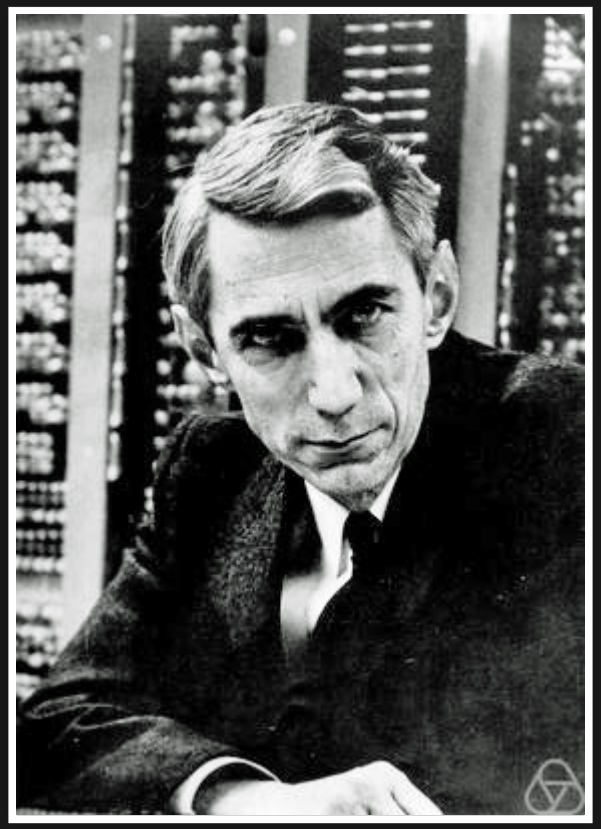
DELTA



DELTA AV DELTA



INFORMASJONSTEORI



Credits

ENTROPI

- Entropi av et tall n er $\log_2(n)$
- Entropi av et sett av symboler s er $H(s) = -\sum p_i \log_2(p_i)$

ENTROPI EKSEMPEL

s = [a, b, a, a, b, c, b, a, b, b, b, a]

Symbol	Frekvens	Sannsynlighet
a	5	5/12 = 0.42
b	6	6/12 = 0.50
c	1	1/12 = 0.08

H(s) = 1.317

ENTROPI EKSEMPEL

TIDSSERIE

$s = [10, 20, 30, 40]$

$H(s) = 2$

DELTA SERIE

$s = [10, 10, 10, 10]$

$H(s) = 0$

TRIKS I LUDO

Algo	Bytes	Ratio
-	8000	1
Gzip	5166	0.646
Zstd	4431	0.554
VLE	9000	1.125
DE-VLE	3933	0.492

APE

Delta-enkoding + custom VLE

Indikator	Bits
0	23
10	26
110	32
111	64

TRIKS I LUDO

Algo	Bytes	Ratio
-	8000	1
Gzip	5166	0.646
Zstd	4431	0.554
VLE	9000	1.125
DE-VLE	3933	0.492
Ape	3567	0.446

STATISTIKK

- Test datasett var 1000 punkter `[1572505277659571746 ... 1572505418711115011]`

STATISTIKK

- Test datasett var 1000 punkter `[1572505277659571746 ... 1572505418711115011]`
- Det er 3 sekunder med data, fra *ett sett*

STATISTIKK

- Test datasett var 1000 punkter `[15725052776595717 46 ... 15725054187111150 11]`
- Det er 3 sekunder med data, fra *ett sett*
- Grovt ekstrapolert til 24 timer blir det:
- `24 x 60 x 60 x 333 = 28771200` elementer i et sett
- ~2000 forskjellige sett
- 128 bits per entry
- ~1TB dagen ukomprimert

FLÅSETE BENCHMARK

Hastighet på ekte data:

Algo	Tid (ms)	Size
Gzip	5363	23720929
Zstd	460	22559413
Xz	12474	15210320
Ape*	263	15872528

* JVM oppstartstid er fjernet, koden er ikke optimalisert

MATNYTTIG

Rask Histogram

```
$ sort -n file | uniq -c | gnuplot -p -e "set style histogram; plot '-' using 1:xtic(2)"
```

Binær/oktal/hex osv

```
$ xxd -b file
```

Utregning på tvers av baser med bc

```
$ bc -l
< ibase=16
< obase=2
< AF
> 10101111
```

FOR - FRAME OF REFERENCE

FOR - FRAME OF REFERENCE

Eksempel:

```
[513100, 513102, 513110, 513109]
```

FOR - FRAME OF REFERENCE

Eksempel:

```
[513100, 513102, 513110, 513109] ⇒ [513100] [0, 2, 10, 9]
```

FOR - FRAME OF REFERENCE

Eksempel:

```
[94143178827, 177147, 129140163, 1594323] ⇒ [23, 11, 17, 13]*
```

* 3 opphøyd i n

FOR - FRAME OF REFERENCE

Eksempel:

```
[1, 513102, 513110, 513109] ⇒ [1, 513102, 513110, 513109]
```

PFOR - PATCHED FRAME OF REFERENCE

PFOR - PATCHED FRAME OF REFERENCE

- Velg en bit-størrelse som dekker 90% av blokken, røffly:

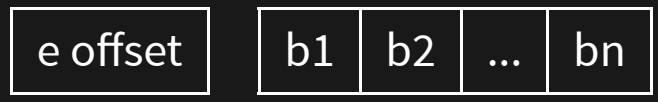
```
for (i = 1; i<=64 && members < Math.floor(len*0.9); ++i) {
  members = 0;
  for (int j = 0; j<len; ++j) {
    if (bits(in[j]) <= i) members += 1;
  }
}
```

PFOR - PATCHED FRAME OF REFERENCE

- Velg en bit-størrelse som dekker 90% av blokken
- Alloker plass til hele blokken

PFOR - PATCHED FRAME OF REFERENCE

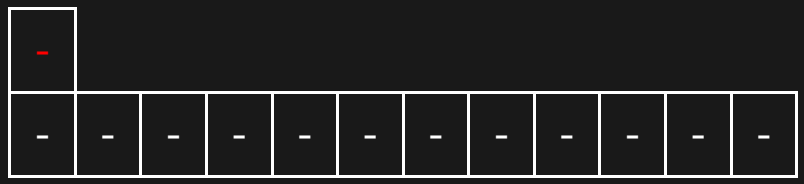
- Velg en bit-størrelse som dekker 90% av blokken
- Alloker plass til hele blokken
- Sett av plass til offset til første feil i headeren



P(FOR EKSEMPEL)

3 9 4 10 5 4 5 9 1000 3 1337 3

Lagring:



P(FOR EKSEMPEL)

3 9 4 10 5 4 5 9 1000 3 1337 3

Lagring:

-											
3	9	4	10	5	4	5	9	-	-	-	-

P(FOR EKSEMPEL)

3 9 4 10 5 4 5 9 1000 3 1337 3

Lagring:

9											
3	9	4	10	5	4	5	9	-	3	-	-

1000

P(FOR EKSEMPEL)

3 9 4 10 5 4 5 9 1000 3 1337 3

Lagring:

9											
3	9	4	10	5	4	5	9	2	3	-	3

1000	1337

P(FOR EKSEMPEL)

3 9 4 10 5 4 5 9 1000 3 1337 3

Lagring:

9											
3	9	4	10	5	4	5	9	2	3	0	3

1000	1337

PFOR - EKLE GRENSETILFELLER

- Om b er mindre enn $bits(offset)$

RLE - RUN-LENGTH ENCODING

RLE - RUN-LENGTH ENCODING

BBCCCDDDDDDDDDDEEEEFFFFF ⇒ B2C3D10E4F5

RLE - RUN-LENGTH ENCODING

THECATINTHEHAT ⇒ T1H1E1C1A1T1I1N1T1H1E1H1A1T1

THE BURROWS WHEELER TRANSFORM

THE BURROWS WHEELER TRANSFORM

Inndata: THECATINTHEHAT

Data = THECATINTHEHAT$

THECATINTHEHAT$

HECATINTHEHAT$T

ECATINTHEHAT$TH

CATINTHEHAT$THE

...

HAT$THECATINTHE

AT$THECATINTHEH

T$THECATINTHEHA

$THECATINTHEHAT

THE BURROWS WHEELER TRANSFORM

Inndata: THECATINTHEHAT

Data = THECATINTHEHAT$

THECATINTHEHAT$
HECATINTHEHAT$T
ECATINTHEHAT$TH
CATINTHEHAT$THE
...
HAT$THECATINTHE
AT$THECATINTHEH
T$THECATINTHEHA
$THECATINTHEHAT

⇒

$THECATINTHEHAT
AT$THECATINTHEH
ATINTHEHAT$THEC
CATINTHEHAT$THE
...
T$THECATINTHEHA
THECATINTHEHAT$
THEHAT$THECATIN
TINTHEHAT$THECA

THE BURROWS WHEELER TRANSFORM

Inndata: THECATINTHEHAT

Data = THECATINTHEHAT$

THECATINTHEHAT$
HECATINTHEHAT$T
ECATINTHEHAT$TH
CATINTHEHAT$THE
...
HAT$THECATINTHE
AT$THECATINTHEH
T$THECATINTHEHA
$THECATINTHEHAT

⇒

$THECATINTHEHA	T
AT$THECATINTHE	H
ATINTHEHAT$THE	C
CATINTHEHAT$TH	E
...	
T$THECATINTHEH	A
THECATINTHEHAT	$
THEHAT$THECATI	N
TINTHEHAT$THEC	A

THCEHHETTTIA$NA

ACGTTGCCAA CGGTACGTTA CGGATCACTG CATACACGTT GCCAACGGTA CGTTACGGAT
CACTGCATAC ACGTTGCCAA CGGTACGTTA CGGATCACTG CATACACGGT GCTCAAGGTA

ATCCCCTTTT TTAAACTTTC C$CCCACCCG GGCCCTAAAT TTGGGGGGAA AAAAAAAAAA
AGAAAGGGTT TTTTTCCCAC CCCGGGGGCC CCCCGAAATT TGGGCAAACC CTTTGGGGGG
G

Algo	Size
-	120
RLE	198
BW-RLE	71

THE INVERSE BURROWS WHEELER TRANSFORM

Inndata: THCEHHETTTIA$NA

T H C E H H E T T T I A $ N A

THE INVERSE BURROWS WHEELER TRANSFORM

$	A	A	C	E	E	H	H	H	I	N	T	T	T	T
T	H	C	E	H	H	E	T	T	T	I	A	$	N	A

THE INVERSE BURROWS WHEELER TRANSFORM

$\$_1$	A_1	A_2	C_1	E_1	E_2	H_1	H_2	H_3	I_1	N_1	T_1	T_2	T_3	T_4
T_1	H_1	C_1	E_1	H_2	H_3	E_2	T_2	T_3	T_4	I_1	A_1	$\$_1$	N_1	A_2

THE INVERSE BURROWS WHEELER TRANSFORM

$\$_1$	A_1	A_2	C_1	E_1	E_2	H_1	H_2	H_3	I_1	N_1	T_1	T_2	T_3	T_4
T_1	H_1	C_1	E_1	H_2	H_3	E_2	T_2	T_3	T_4	I_1	A_1	$\$_1$	N_1	A_2

Output: T

THE INVERSE BURROWS WHEELER TRANSFORM

$\$_1$	A_1	A_2	C_1	E_1	E_2	H_1	H_2	H_3	I_1	N_1	T_1	T_2	T_3	T_4
T_1	H_1	C_1	E_1	H_2	H_3	E_2	T_2	T_3	T_4	I_1	A_1	$\$_1$	N_1	A_2

Output: TH

THE INVERSE BURROWS WHEELER TRANSFORM

$\$_1$	A_1	A_2	C_1	E_1	E_2	H_1	H_2	H_3	I_1	N_1	T_1	T_2	T_3	T_4
T_1	H_1	C_1	E_1	H_2	H_3	E_2	T_2	T_3	T_4	I_1	A_1	$\$_1$	N_1	A_2

Output: THE

THE INVERSE BURROWS WHEELER TRANSFORM

$\$_1$	A_1	A_2	C_1	E_1	E_2	H_1	H_2	H_3	I_1	N_1	T_1	T_2	T_3	T_4
T_1	H_1	C_1	E_1	H_2	H_3	E_2	T_2	T_3	T_4	I_1	A_1	$\$_1$	N_1	A_2

Output: THEC

THE INVERSE BURROWS WHEELER TRANSFORM

$\$_1$	A_1	A_2	C_1	E_1	E_2	H_1	H_2	H_3	I_1	N_1	T_1	T_2	T_3	T_4
T_1	H_1	C_1	E_1	H_2	H_3	E_2	T_2	T_3	T_4	I_1	A_1	$\$_1$	N_1	A_2

Output: THECA

THE INVERSE BURROWS WHEELER TRANSFORM

$\$_1$	A_1	A_2	C_1	E_1	E_2	H_1	H_2	H_3	I_1	N_1	T_1	T_2	T_3	T_4
T_1	H_1	C_1	E_1	H_2	H_3	E_2	T_2	T_3	T_4	I_1	A_1	$\$_1$	N_1	A_2

Output: THECAT

THE INVERSE BURROWS WHEELER TRANSFORM

$\$_1$	A_1	A_2	C_1	E_1	E_2	H_1	H_2	H_3	I_1	N_1	T_1	T_2	T_3	T_4
T_1	H_1	C_1	E_1	H_2	H_3	E_2	T_2	T_3	T_4	I_1	A_1	$\$_1$	N_1	A_2

Output: THECATI

THE INVERSE BURROWS WHEELER TRANSFORM

$\$_1$	A_1	A_2	C_1	E_1	E_2	H_1	H_2	H_3	I_1	N_1	T_1	T_2	T_3	T_4
T_1	H_1	C_1	E_1	H_2	H_3	E_2	T_2	T_3	T_4	I_1	A_1	$\$_1$	N_1	A_2

Output: THECATIN

THE INVERSE BURROWS WHEELER TRANSFORM

$\$_1$	A_1	A_2	C_1	E_1	E_2	H_1	H_2	H_3	I_1	N_1	T_1	T_2	T_3	T_4
T_1	H_1	C_1	E_1	H_2	H_3	E_2	T_2	T_3	T_4	I_1	A_1	$\$_1$	N_1	A_2

Output: THECATINT

THE INVERSE BURROWS WHEELER TRANSFORM

$\$_1$	A_1	A_2	C_1	E_1	E_2	H_1	H_2	H_3	I_1	N_1	T_1	T_2	T_3	T_4
T_1	H_1	C_1	E_1	H_2	H_3	E_2	T_2	T_3	T_4	I_1	A_1	$\$_1$	N_1	A_2

Output: THECATINTH

THE INVERSE BURROWS WHEELER TRANSFORM

$\$_1$	A_1	A_2	C_1	E_1	E_2	H_1	H_2	H_3	I_1	N_1	T_1	T_2	T_3	T_4
T_1	H_1	C_1	E_1	H_2	H_3	E_2	T_2	T_3	T_4	I_1	A_1	$\$_1$	N_1	A_2

Output: THECATINTHE

THE INVERSE BURROWS WHEELER TRANSFORM

$\$_1$	A_1	A_2	C_1	E_1	E_2	H_1	H_2	H_3	I_1	N_1	T_1	T_2	T_3	T_4
T_1	H_1	C_1	E_1	H_2	H_3	E_2	T_2	T_3	T_4	I_1	A_1	$\$_1$	N_1	A_2

Output: THECATINTHEH

THE INVERSE BURROWS WHEELER TRANSFORM

$\$_1$	A_1	A_2	C_1	E_1	E_2	H_1	H_2	H_3	I_1	N_1	T_1	T_2	T_3	T_4
T_1	H_1	C_1	E_1	H_2	H_3	E_2	T_2	T_3	T_4	I_1	A_1	$\$_1$	N_1	A_2

Output: THECATINTHEHA

THE INVERSE BURROWS WHEELER TRANSFORM

$\$_1$	A_1	A_2	C_1	E_1	E_2	H_1	H_2	H_3	I_1	N_1	T_1	T_2	T_3	T_4
T_1	H_1	C_1	E_1	H_2	H_3	E_2	T_2	T_3	T_4	I_1	A_1	$\$_1$	N_1	A_2

Output: THECATINTHEHAT

THE INVERSE BURROWS WHEELER TRANSFORM

$\$_1$	A_1	A_2	C_1	E_1	E_2	H_1	H_2	H_3	I_1	N_1	T_1	T_2	T_3	T_4
T_1	H_1	C_1	E_1	H_2	H_3	E_2	T_2	T_3	T_4	I_1	A_1	$\$_1$	N_1	A_2

Output: THECATINTHEHAT$

PUSTEPAUSE

badger badger badger badger badger badger badger badger badger badger badger badger
mushroom mushroom a badger badger badger badger badger badger badger badger badger
badger badger mushroom mushroom a badger badger badger badger badger badger badger
badger badger badger badger mushroom mushroom a badger badger badger badger badger
badger badger badger badger badger badger a snake a snake

badger badger badger badger badger badger badger badger badger badger badger badger
mushroom mushroom a badger badger badger badger badger badger badger badger badger
badger badger mushroom mushroom a badger badger badger badger badger badger badger
badger badger badger badger mushroom mushroom a badger badger badger badger badger
badger badger badger badger badger badger a snake a snake

Symbol	Frequency
a	5 / 58 = 0.086
badger	45 / 58 = 0.776
mushroom	6 / 58 = 0.103
snake	2 / 58 = 0.035

H(a ) = ~1.095

OPPSLAGSTABELL

Symbol	Kode
a	00
badger	01
mushroom	10
snake	11

OPPSLAGSTABELL

Symbol	Kode
a	0
badger	10
mushroom	110
snake	111

OPPSLAGSTABELL

Symbol	Kode
badger	0
mushroom	10
a	110
snake	111

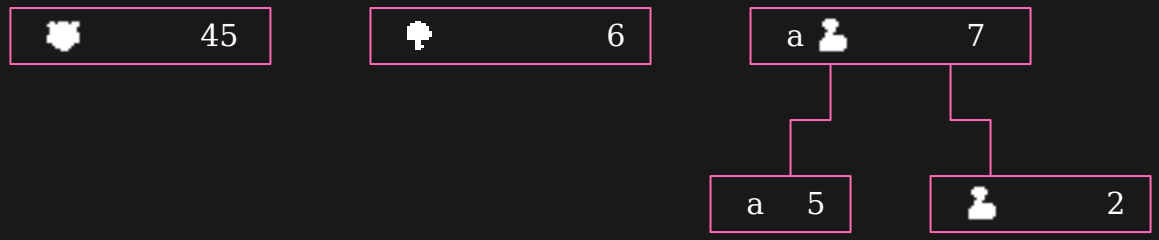
HUFFMAN-KODING

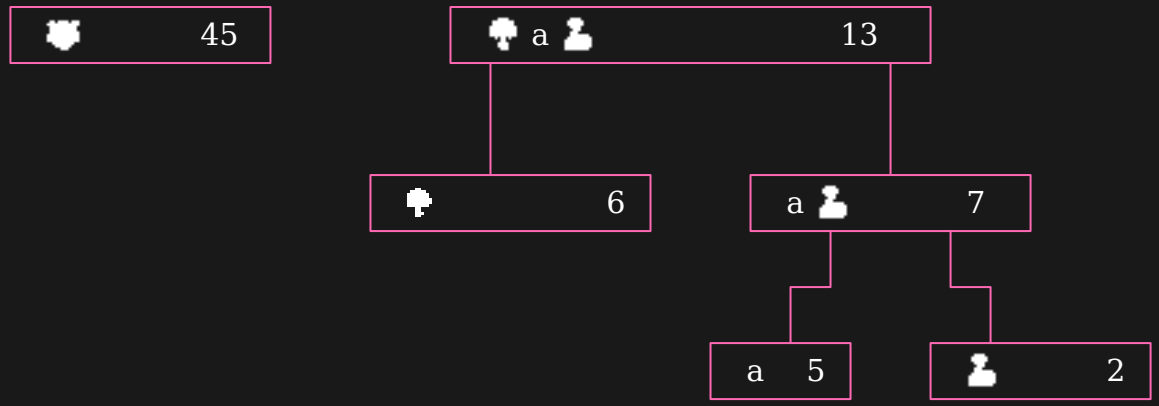

45

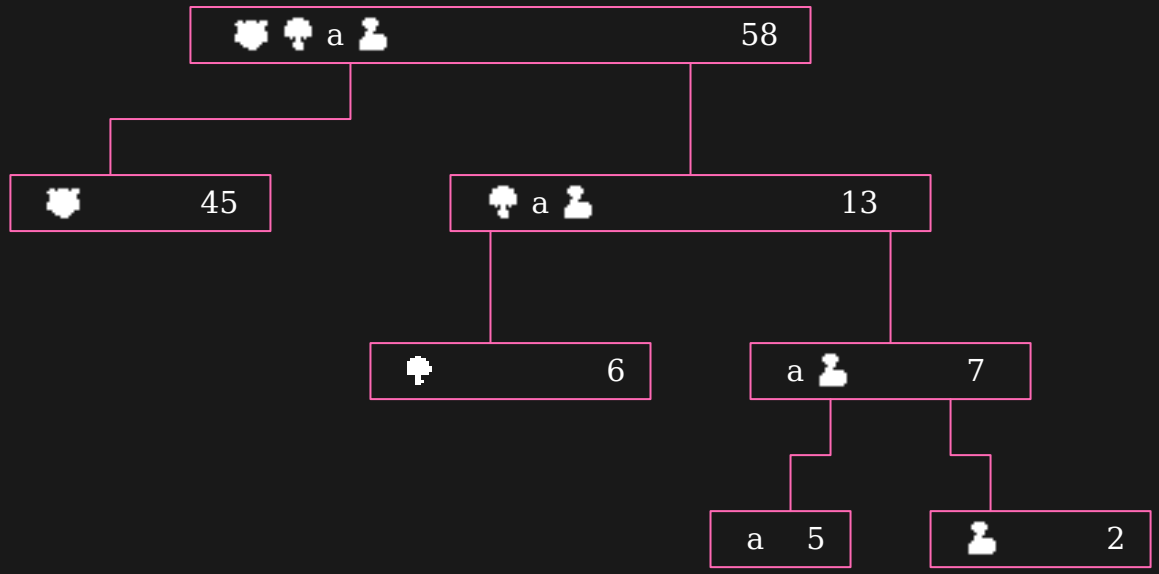
6

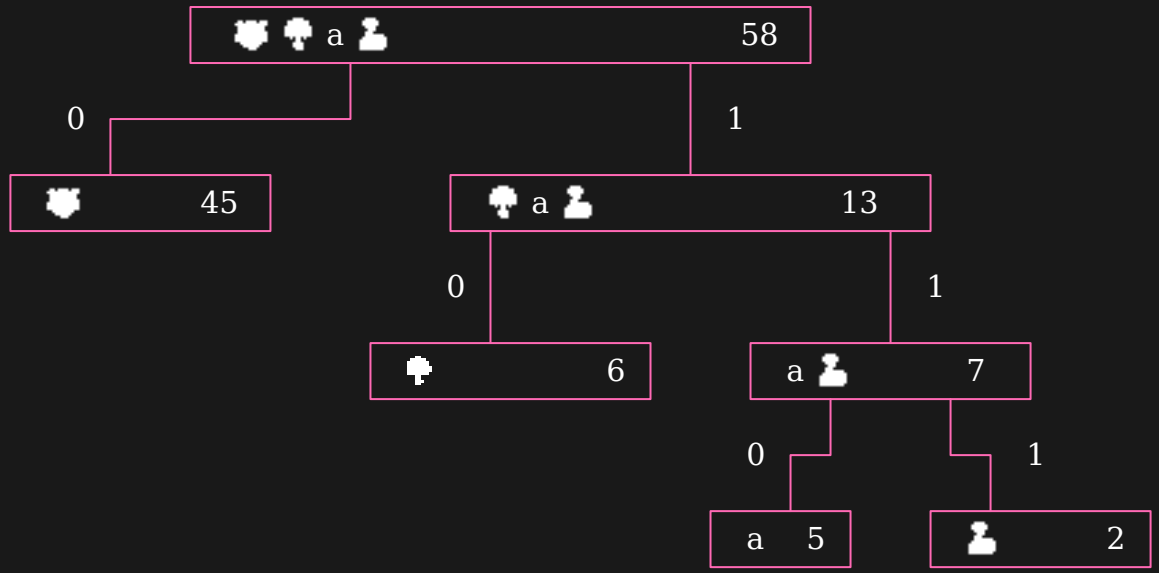
a 5

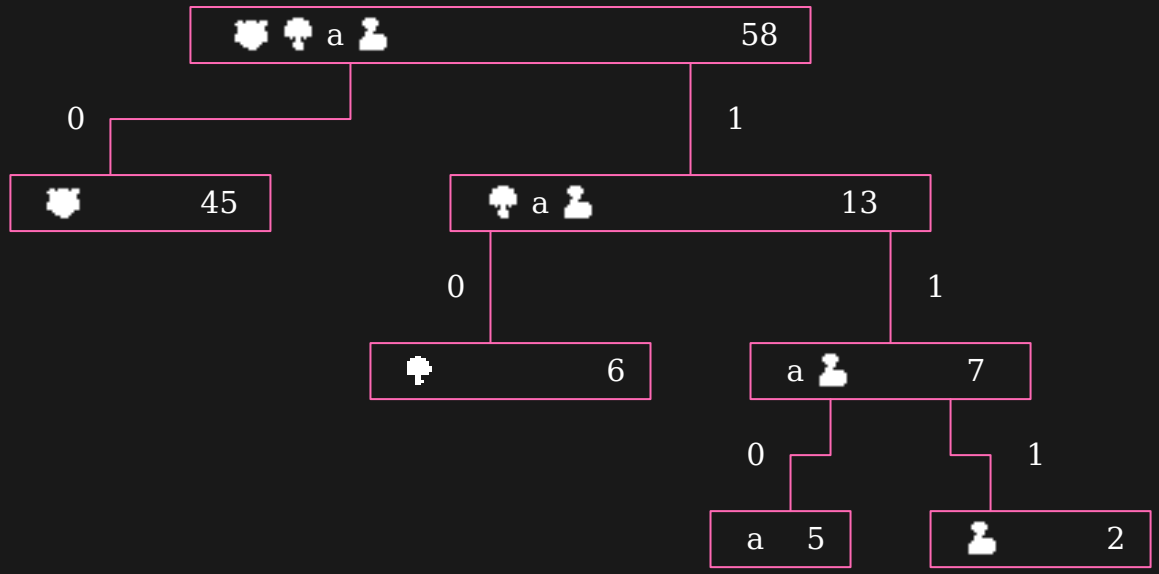
2


45
6
a
7
a
5
2


45
a
13
6
a
7
a
5
2


a
58
45
a
13
6
a
7
a
5
2


a
58
0
1
45
a
13
0
1
6
a
7
0
1
a
5
2



Symbol	Kode
badger	0
mushroom	10
a	110
snake	111

LEMPEL ZIV

LZ77 ENCODING

Inndata: abracadabra


Search
Look ahead
Stream
Output
a
b r a c a d a b r a

Search	Look ahead	Stream	Output
	a	b r a c a d a b r a	
a	b	r a c a d a b r a	[0, 0, a]

Search	Look ahead	Stream	Output
	a	b r a c a d a b r a	
a	b	r a c a d a b r a	[0, 0, a]
a b	r	a c a d a b r a	[0, 0, b]

Search	Look ahead	Stream	Output
	a	b r a c a d a b r a	
a	b	r a c a d a b r a	[0, 0, a]
a b	r	a c a d a b r a	[0, 0, b]
a b r	a	c a d a b r a	[0, 0, r]

Search	Look ahead	Stream	Output
	a	b r a c a d a b r a	
a	b	r a c a d a b r a	[0, 0, a]
a b	r	a c a d a b r a	[0, 0, b]
a b r	a	c a d a b r a	[0, 0, r]
a b r	a c	a d a b r a	

Search	Look ahead	Stream	Output
	a	b r a c a d a b r a	
a	b	r a c a d a b r a	[0, 0, a]
a b	r	a c a d a b r a	[0, 0, b]
a b r	a	c a d a b r a	[0, 0, r]
a b r	a c	a d a b r a	
a b r a	c	a d a b r a	[3, 1]

Search	Look ahead	Stream	Output
	a	b r a c a d a b r a	
a	b	r a c a d a b r a	[0, 0, a]
a b	r	a c a d a b r a	[0, 0, b]
a b r	a	c a d a b r a	[0, 0, r]
a b r	a c	a d a b r a	
a b r a	c	a d a b r a	[3, 1]
a b r a c	a	d a b r a	[0, 0, c]

Search	Look ahead	Stream	Output
	a	b r a c a d a b r a	
a	b	r a c a d a b r a	[0, 0, a]
a b	r	a c a d a b r a	[0, 0, b]
a b r	a	c a d a b r a	[0, 0, r]
a b r	a c	a d a b r a	
a b r a	c	a d a b r a	[3, 1]
a b r a c	a	d a b r a	[0, 0, c]
a b r a c	a d	a b r a	

Search	Look ahead	Stream	Output
	a	b r a c a d a b r a	
a	b	r a c a d a b r a	[0, 0, a]
a b	r	a c a d a b r a	[0, 0, b]
a b r	a	c a d a b r a	[0, 0, r]
a b r	a c	a d a b r a	
a b r a	c	a d a b r a	[3, 1]
a b r a c	a	d a b r a	[0, 0, c]
a b r a c	a d	a b r a	
a b r a c a	d	a b r a	[2, 1]

Search	Look ahead	Stream	Output
	a	b r a c a d a b r a	
a	b	r a c a d a b r a	[0, 0, a]
a b	r	a c a d a b r a	[0, 0, b]
a b r	a	c a d a b r a	[0, 0, r]
a b r	a c	a d a b r a	
a b r a	c	a d a b r a	[3, 1]
a b r a c	a	d a b r a	[0, 0, c]
a b r a c	a d	a b r a	
a b r a c a	d	a b r a	[2, 1]
a b r a c a d	a	b r a	[0, 0, d]

Search	Look ahead	Stream	Output
	a	b r a c a d a b r a	
a	b	r a c a d a b r a	[0, 0, a]
a b	r	a c a d a b r a	[0, 0, b]
a b r	a	c a d a b r a	[0, 0, r]
a b r	a c	a d a b r a	
a b r a	c	a d a b r a	[3, 1]
a b r a c	a	d a b r a	[0, 0, c]
a b r a c	a d	a b r a	
a b r a c a	d	a b r a	[2, 1]
a b r a c a d	a	b r a	[0, 0, d]
a b r a c a d	a b	r a	

Search	Look ahead	Stream	Output
	a	b r a c a d a b r a	
a	b	r a c a d a b r a	[0, 0, a]
a b	r	a c a d a b r a	[0, 0, b]
a b r	a	c a d a b r a	[0, 0, r]
a b r	a c	a d a b r a	
a b r a	c	a d a b r a	[3, 1]
a b r a c	a	d a b r a	[0, 0, c]
a b r a c	a d	a b r a	
a b r a c a	d	a b r a	[2, 1]
a b r a c a d	a	b r a	[0, 0, d]
a b r a c a d	a b	r a	
a b r a c a d	a b r	a	

Search	Look ahead	Stream	Output
	a	b r a c a d a b r a	
a	b	r a c a d a b r a	[0, 0, a]
a b	r	a c a d a b r a	[0, 0, b]
a b r	a	c a d a b r a	[0, 0, r]
a b r	a c	a d a b r a	
a b r a	c	a d a b r a	[3, 1]
a b r a c	a	d a b r a	[0, 0, c]
a b r a c	a d	a b r a	
a b r a c a	d	a b r a	[2, 1]
a b r a c a d	a	b r a	[0, 0, d]
a b r a c a d	a b	r a	
a b r a c a d	a b r	a	
a b r a c a d	a b r a		

Search	Look ahead	Stream	Output
	a	b r a c a d a b r a	
a	b	r a c a d a b r a	[0, 0, a]
a b	r	a c a d a b r a	[0, 0, b]
a b r	a	c a d a b r a	[0, 0, r]
a b r	a c	a d a b r a	
a b r a	c	a d a b r a	[3, 1]
a b r a c	a	d a b r a	[0, 0, c]
a b r a c	a d	a b r a	
a b r a c a	d	a b r a	[2, 1]
a b r a c a d	a	b r a	[0, 0, d]
a b r a c a d	a b	r a	
a b r a c a d	a b r	a	
a b r a c a d	a b r a		
a b r a c a d a b r a			[7, 4]

DEKODING

Kø: [[0, 0, a], [0, 0, b], [0, 0, r], [3, 1], [0, 0, c], [2, 1], [0, 0, d], [7, 4]]

Ut: []

DEKODING

Kø: [[0, 0, a], [0, 0, b], [0, 0, r], [3, 1], [0, 0, c], [2, 1], [0, 0, d], [7, 4]]

Ut: [a]

DEKODING

Kø: [[0, 0, b], [0, 0, r], [3, 1], [0, 0, c], [2, 1], [0, 0, d], [7, 4]]

Ut: [ab]

DEKODING

Kø: [[0, 0, r], [3, 1], [0, 0, c], [2, 1], [0, 0, d], [7, 4]]

Ut: [abr]

DEKODING

Kø: [[3, 1], [0, 0, c], [2, 1], [0, 0, d], [7, 4]]

Ut: [abra]

DEKODING

Kø: [[0, 0, c], [2, 1], [0, 0, d], [7, 4]]

Ut: [abrac]

DEKODING

Kø: [[2, 1], [0, 0, d], [7, 4]]

Ut: [abracа]

DEKODING

Kø: [[0, 0, d], [7, 4]]

Ut: [abracad]

DEKODING

Kø: [[7, 4]]

Ut: [abracadabra]