Lempel-Ziv: a "one-bit catastrophe" but not a tragedy

G. Lagarde S. Perifel

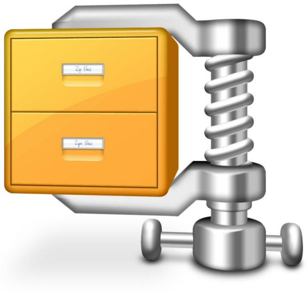
ANR DELTA
27/03/18

A strange scenario...

Original file

1 TB

Your favorite compression algorithm



Compressed file

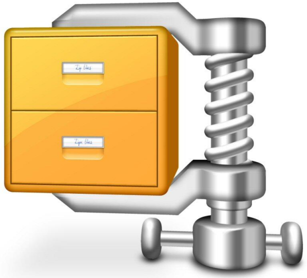
100 MB

A strange scenario...

Original file

1 TB

Your favorite compression algorithm



«Zut alors !», I made a typo... Well, just need to compress the file again...

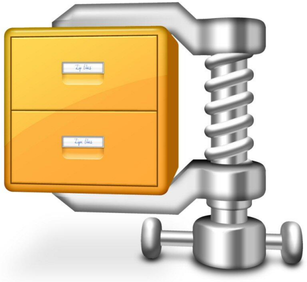
Compressed file

100 MB

A strange scenario...

ORIGINAL FILE

YOUR FAVORITE COMPRESSION ALGORITHM



COMPRESSED FILE

1 TB

? ? ?

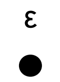
0.9 TB ! !

Lempel-Ziv ?

Lempel-Ziv algorithms?

- Several lossless data compression algorithms (LZ'77, LZ'78, LZW, …)
- Dictionary-based encoding
- Used everywhere
  - Deflate (gzip)
  - GIF
  - `compress` (Unix)
- Theoretical interest
  - Entropy of a random source
  - Lyndon factorisation
  - ...

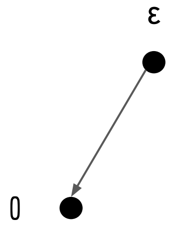
LZ'78



W = 001010110110101...

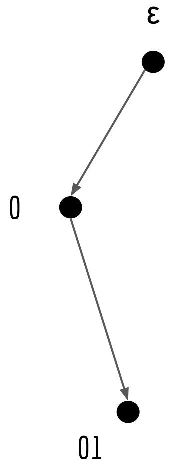
LZ'78

W = 001010110110101...



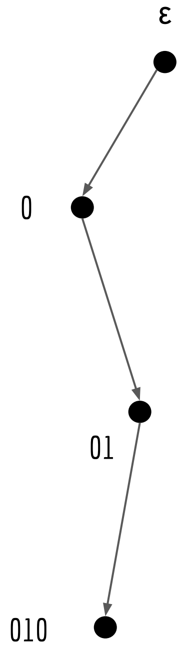
LZ'78

W = 001010110110101...



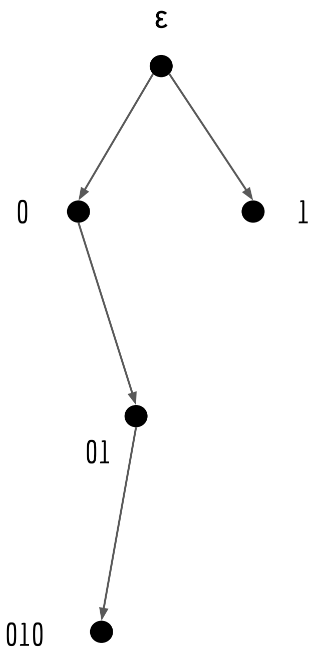
LZ’78

W = 001010110110101…



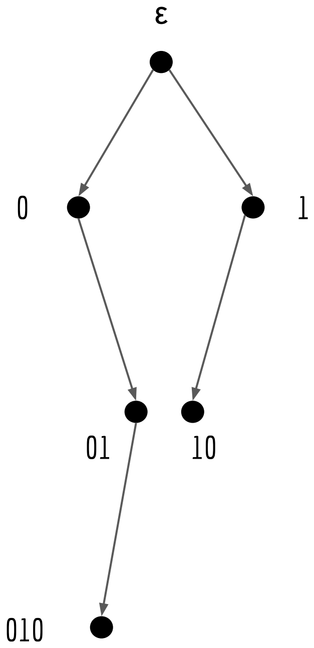
LZ'78

W = 0010101110110101...



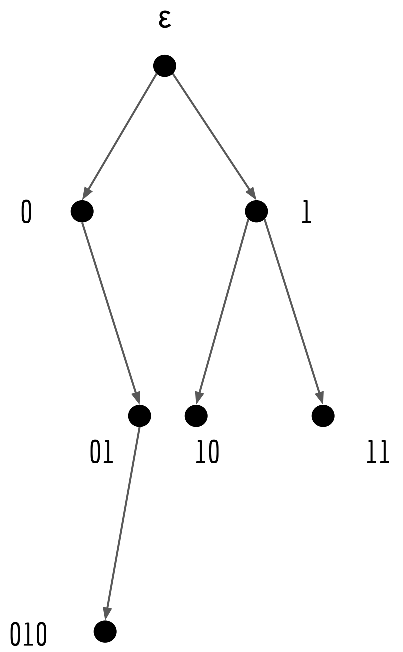
LZ'78

W = 0010101101101101...



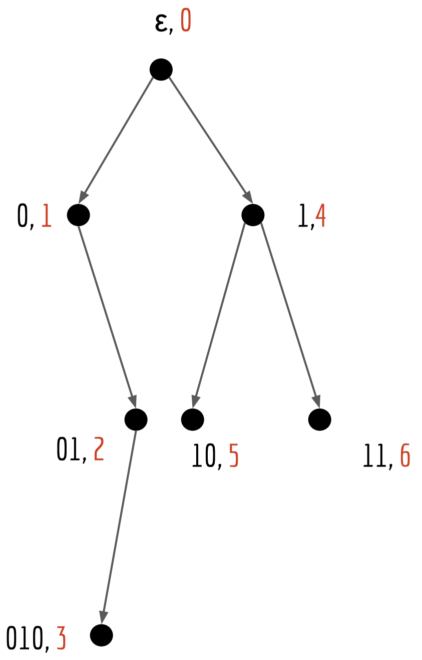
LZ’78

W = 0010101101101101...



LZ'78

W = 0|01|010|1|10|11|0101...



Keep track of times to be able to reconstruct w

LZ'78

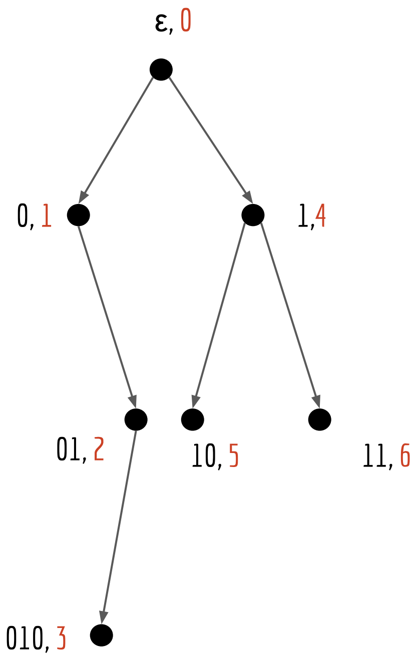
W = 0|01|010|1|10|11|0101...

Dictionary:

$$D(w) = \{\epsilon, 0, 01, 010, 1, 10, 11, \ldots\}$$

Size of the compression:

$$\Theta(|D(w)|.\log|D(w)|)$$



Keep track of times to be able to reconstruct w

LZ'78

W = 0|01|010|1|10|11|0101...

Compressible | Incompressible

Dictionary:

$$D(w) = \{\epsilon, 0, 01, 010, 1, 10, 11, \ldots\}$$

$|D(w)|.\log|D(w)| =$ $o(|w|)$ | $\Theta(|w|)$

Size of the compression:

$$\Theta(|D(w)|.\log|D(w)|)$$

LZ'78

W = $\boxed{0}\boxed{01}\boxed{010}\boxed{1}\boxed{10}\boxed{11}\boxed{0101}$...

Dictionary:

$$D(w) = \{\epsilon, 0, 01, 010, 1, 10, 11, \ldots\}$$

Size of the compression:

$$\Theta(|D(w)|.\log|D(w)|)$$

	Compressible	Incompressible
$\lvert D(w)\rvert.\log\lvert D(w)\rvert =$	$o(\lvert w\rvert)$	$\Theta(\lvert w\rvert)$
$\lvert D(w)\rvert =$	$o(\frac{\lvert w\rvert}{\log\lvert w\rvert})$	$\Theta(\frac{\lvert w\rvert}{\log\lvert w\rvert})$

“One-bit catastrophe” question

Open question (Jack Lutz - Late ‘90s)

Does there exist an infinite word w such that

$$\rho(w) \neq \rho(0w)$$

“One-bit catastrophe” question

Open question (Jack Lutz - Late ‘90s)

Does there exist an infinite word w such that

$$\rho(w) \neq \rho(0w)$$

$$0 \neq \alpha$$

“One-bit catastrophe” question

Theorem 1

There is an infinite word w such that

$$\rho_{sup}(w) = 0$$

$$\rho_{inf}(0w) \geq \frac{1}{6075}$$

“One-bit catastrophe” question

Theorem 1

There is an infinite word w such that

$$\rho_{sup}(w) = 0$$

$$\rho_{inf}(0w) \geq \frac{1}{6075}$$

Theorem 2

For all finite word w and any letter a

$$|D(aw)| \leq 3\sqrt{|w|.\,|D(w)|}$$

(≈Geometric mean)

"One-bit catastrophe" question

Theorem 1

There is an infinite word w such that

$$\rho_{sup}(w) = 0$$

$$\rho_{inf}(0w) \geq \frac{1}{6075}$$

Theorem 2

For all finite word w and any letter a

$$|D(aw)| \leq 3\sqrt{|w|.\,|D(w)|}$$

(≈Geometric mean)

The "not such a tragedy" part

$$|D(w)| = o(\frac{|w|}{\log^2 |w|}) \longrightarrow |D(aw)| = o(\frac{|w|}{\log |w|})$$

“One-bit catastrophe” question

Theorem 1

There is an infinite word w such that

$$\rho_{sup}(w) = 0$$

$$\rho_{inf}(0w) \geq \frac{1}{6075}$$

Theorem 2

For all finite word w and any letter a

$$|D(aw)| \leq 3\sqrt{|w|.\,|D(w)|}$$

(≈Geometric mean)

Theorem 3

Theorem 2 is tight up to a multiplicative constant

For all finite word w and any letter a

$$|D(aw)| \leq 3\sqrt{|w|.\,|D(w)|}$$

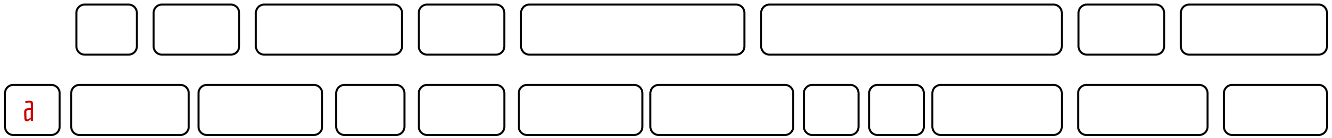
W =

For all finite word w and any letter a

$$|D(aw)| \leq 3\sqrt{|w|.\,|D(w)|}$$

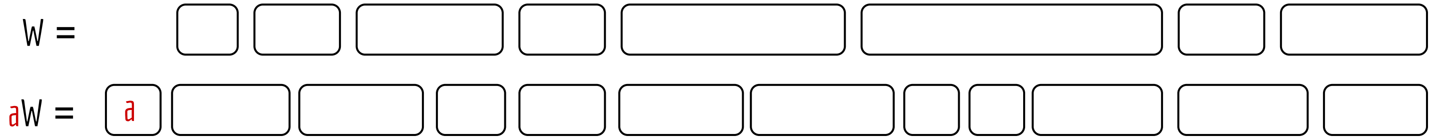
W =

aW =



For all finite word w and any letter a

$$|D(aw)| \leq 3\sqrt{|w|.\,|D(w)|}$$



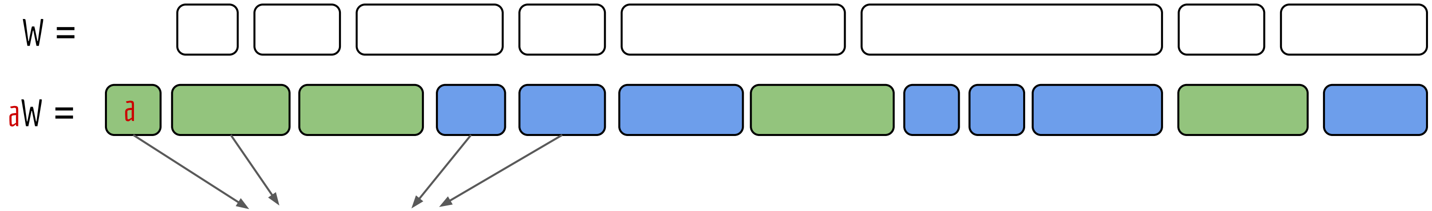
$$D(aw) = D_1 \cup D_2$$

For all finite word w and any letter a

$$|D(aw)| \leq 3\sqrt{|w|.|D(w)|}$$

D_1 : Blocks of 0w that overlap 2 blocks of w. $|D_1| \leq |D(w)| \leq \sqrt{|w|.|D(w)|}$

D_2 : Blocks of 0w included in a block of w.



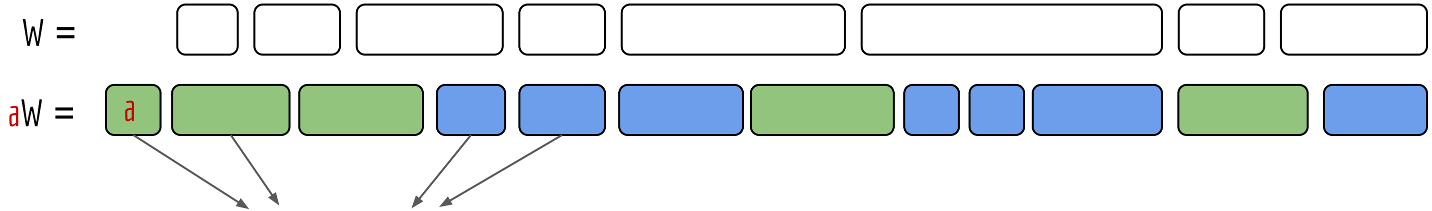
$$D(aw) = D_1 \cup D_2$$

For all finite word w and any letter a

$$|D(aw)| \leq 3\sqrt{|w|.\,|D(w)|}$$

D_1 : Blocks of 0w that overlap 2 blocks of w. $|D_1| \leq |D(w)| \leq \sqrt{|w|.\,|D(w)|}$

D_2 : Blocks of 0w included in a block of w.



$$D(aw) = D_1 \cup D_2$$

$$|D_1| \leq |D(w)|$$

For all finite word w and any letter a

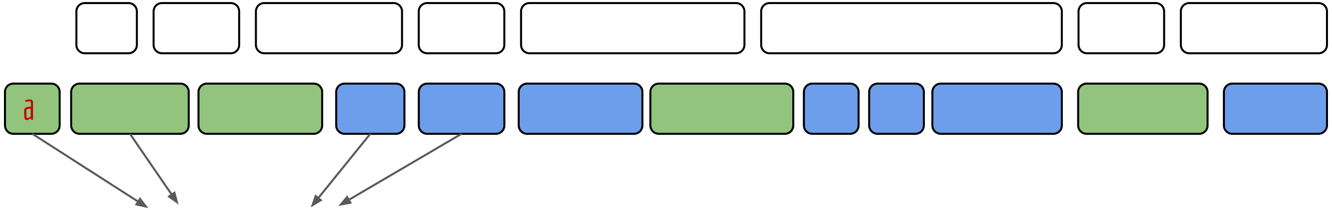
$$|D(aw)| \leq 3\sqrt{|w|.|D(w)|}$$

D_1 : Blocks of 0w that overlap 2 blocks of w. $|D_1| \leq |D(w)| \leq \sqrt{|w|.|D(w)|}$

D_2 : Blocks of 0w included in a block of w.

W =

aW =



$$D(aw) = D_1 \cup D_2$$

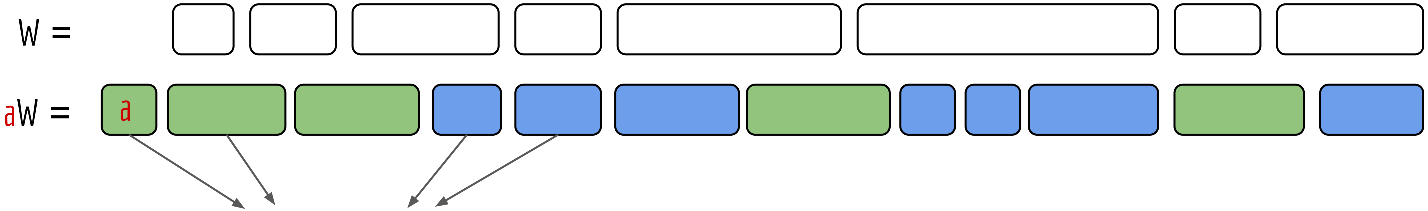
1. Each  is a substring of a 

For all finite word w and any letter a

$$|D(aw)| \leq 3\sqrt{|w|.\,|D(w)|}$$

D_1 : Blocks of 0w that overlap 2 blocks of w. $|D_1| \leq |D(w)| \leq \sqrt{|w|.\,|D(w)|}$

D_2 : Blocks of 0w included in a block of w.



$$D(aw) = D_1 \cup D_2$$

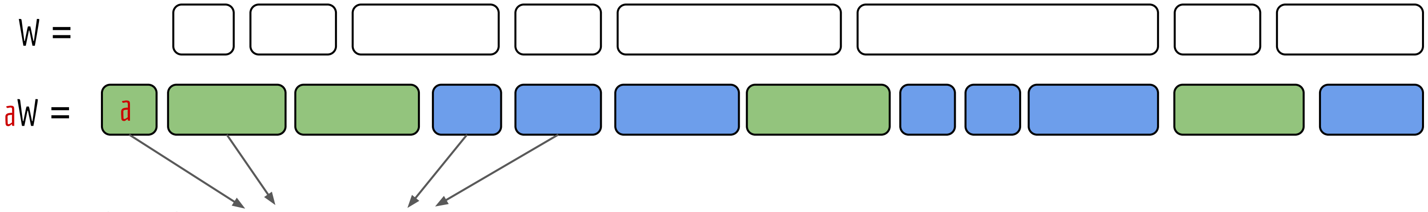
1. Each is a substring of a
2. For any i, the number of distinct blocks of size i is $\leq |D(w)|$

For all finite word w and any letter a

$$|D(aw)| \leq 3\sqrt{|w|.|D(w)|}$$

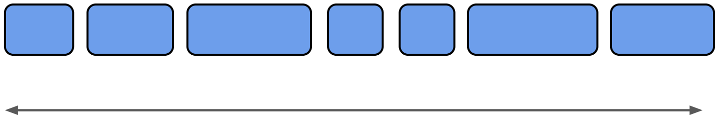
D_1 : Blocks of 0w that overlap 2 blocks of w. $|D_1| \leq |D(w)| \leq \sqrt{|w|.|D(w)|}$

D_2 : Blocks of 0w included in a block of w.



$$D(aw) = D_1 \cup D_2$$

1. Each is a substring of a
2. For any i, the number of distinct blocks of size i is $\leq |D(w)|$
3. W' =

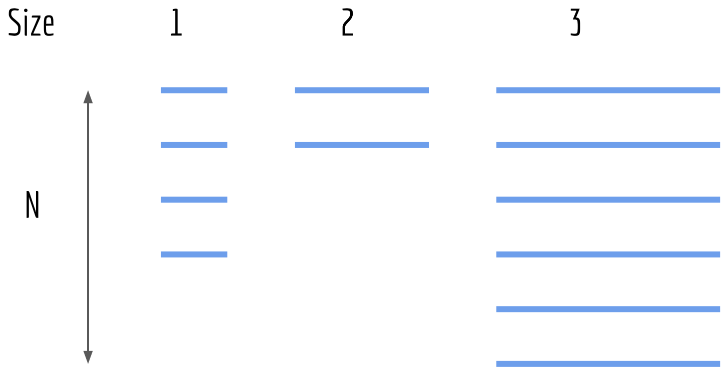


$\leq |w|$

GOAL: Cover L with segments
CONSTRAINTS: At most N segments of each length

Then

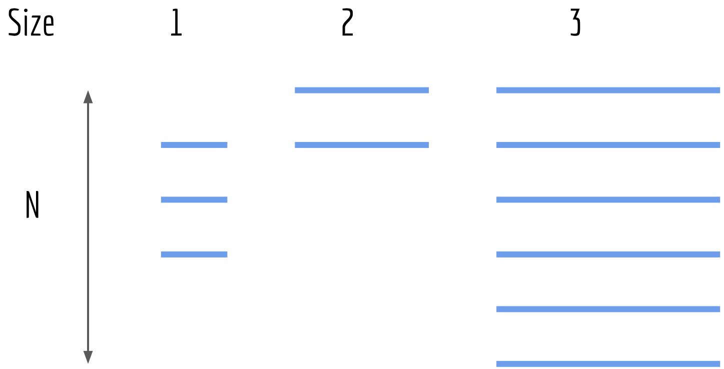
segments used $\leq 2\sqrt{L.N}$

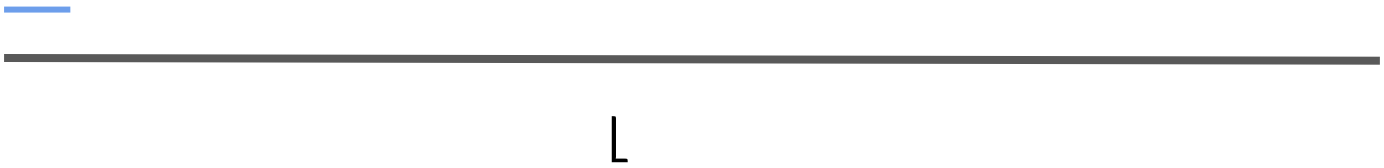


GOAL: Cover L with segments
CONSTRAINTS: At most N segments of each length

Then

\# segments used $\leq 2\sqrt{L.N}$

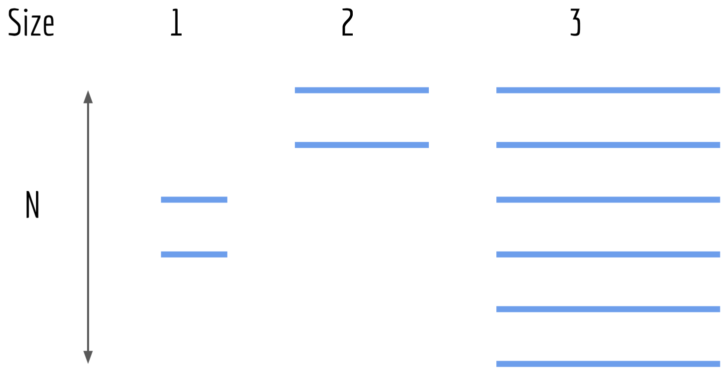


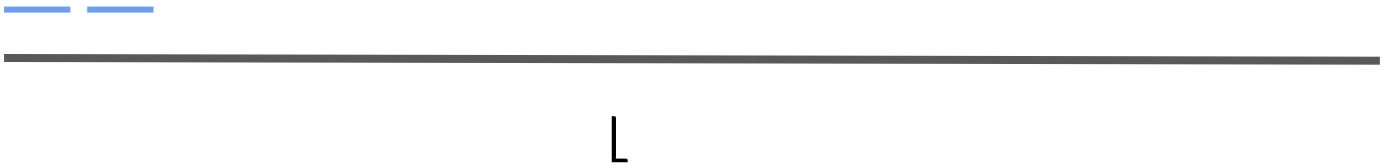


GOAL: Cover L with segments
CONSTRAINTS: At most N segments of each length

Then

segments used $\leq 2\sqrt{L.N}$

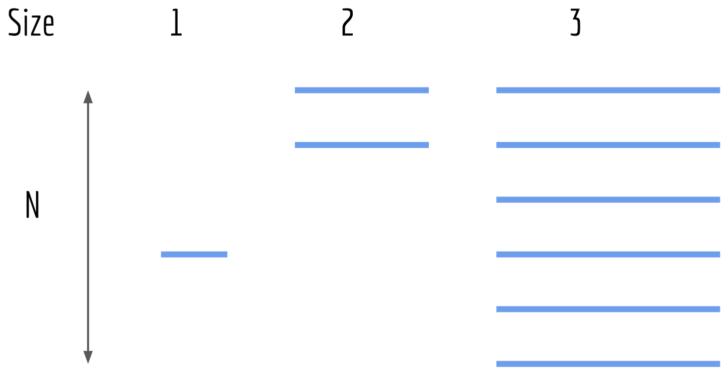


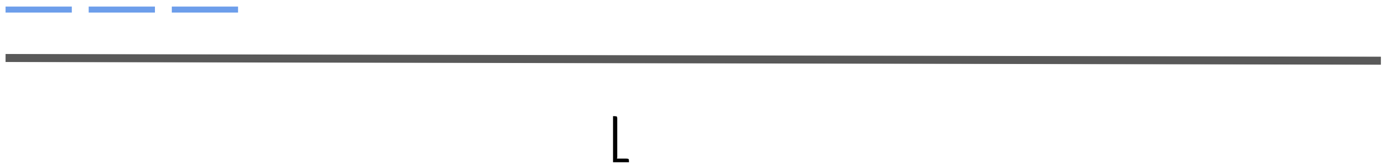


GOAL: Cover L with segments
CONSTRAINTS: At most N segments of each length

Then

segments used $\leq 2\sqrt{L.N}$

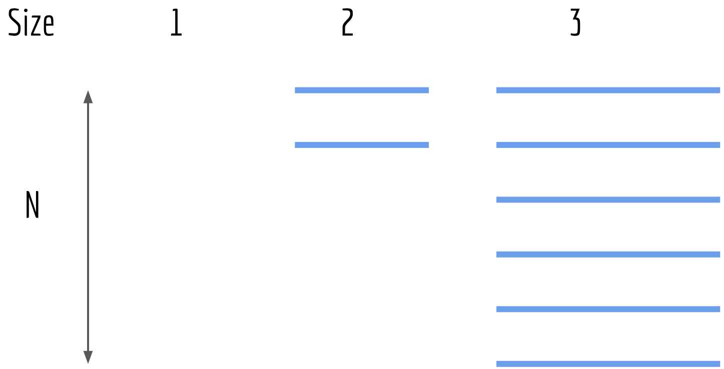


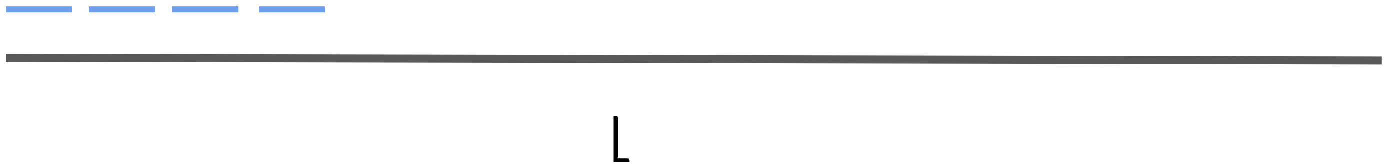


GOAL: Cover L with segments
CONSTRAINTS: At most N segments of each length

Then

segments used $\leq 2\sqrt{L.N}$

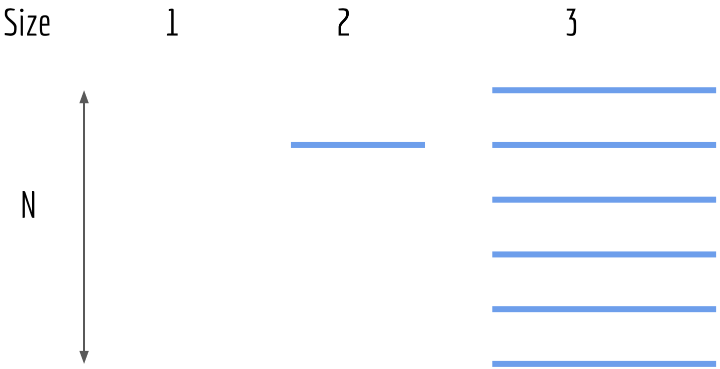


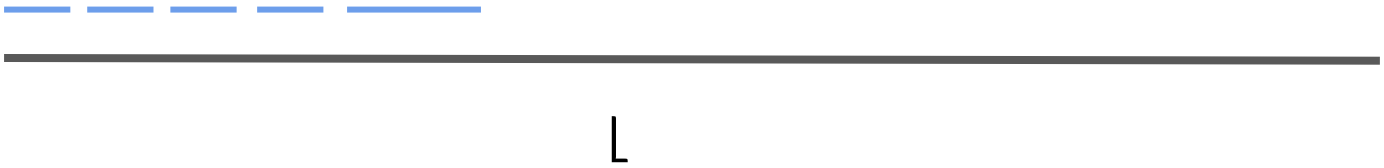


GOAL: Cover L with segments
CONSTRAINTS: At most N segments of each length

Then

segments used $\leq 2\sqrt{L.N}$

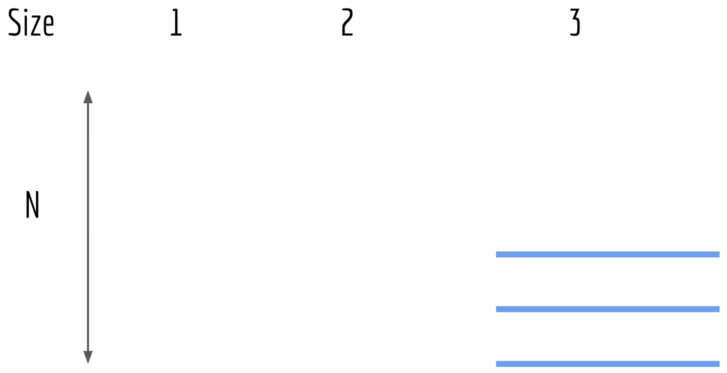


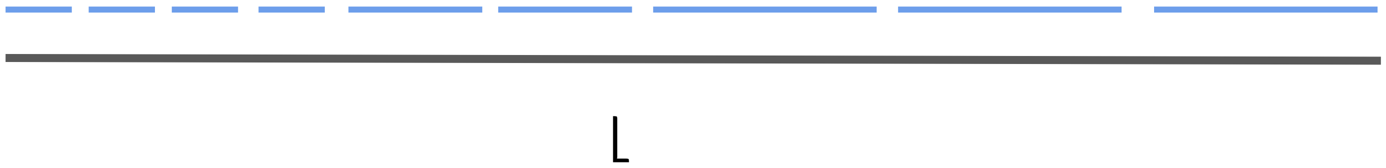


GOAL: Cover L with segments
CONSTRAINTS: At most N segments of each length

Then

segments used $\leq 2\sqrt{L.N}$



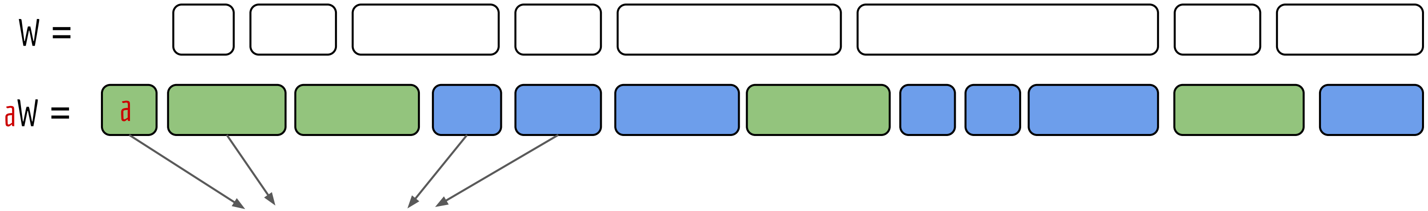


For all finite word w and any letter a

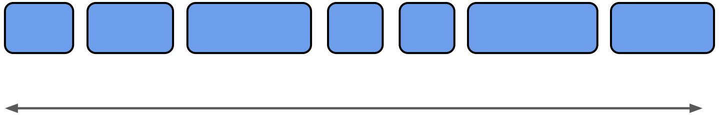
$$|D(aw)| \leq 3\sqrt{|w|.|D(w)|}$$

D_1 : Blocks of 0w that overlap 2 blocks of w. $|D_1| \leq |D(w)| \leq \sqrt{|w|.|D(w)|}$

D_2 : Blocks of 0w included in a block of w.



$$D(aw) = D_1 \cup D_2$$

1. Each  is a substring of a 
2. For any i, the number of distinct  blocks of size i is $\leq |D(w)|$
3. W' = 

$\leq |w|$

\# segments used $\leq 2\sqrt{L.N}$

\#  $\leq 2\sqrt{|w'|.|D(w)|}$

“Get a catastrophe”: starter kit

1. Get a “weak” catastrophe.
   i.e maximal variation for an optimally compressible word.

“Get a catastrophe”: starter kit

1. Get a “weak” catastrophe.
   i.e maximal variation for an optimally compressible word.

2. A “true” catastrophe:
   a. Create a lot of independent “de Bruijn-style” words.
   b. Use them to concatenate independent “weak” catastrophes.

Future work

- Improve the constants (from 0 to 1?)
- Remove the gadgets
- Is the weak catastrophe the typical case for optimally compressible words?
- What happens over real data?

Merci !