# Bzip2

## Seminar: Datenkompression

Toprak Saricerci
7445073

Florian Brohm
7443251

February 10, 2023

# Bzip2: Überblick

- Verlustfreies Datenkompressionsverfahren

# Bzip2: Überblick

- Verlustfreies Datenkompressionsverfahren
- Ähnlich zu ZIP

# Bzip2: Überblick

- Verlustfreies Datenkompressionsverfahren
- Ähnlich zu ZIP
- Basiert auf Burrows-Wheeler transform

# Bzip2: Überblick

- Verlustfreies Datenkompressionsverfahren
- Ähnlich zu ZIP
- Basiert auf Burrows-Wheeler transform
- Autor: Julian Seward

# Bzip2: Überblick

- Verlustfreies Datenkompressionsverfahren
- Ähnlich zu ZIP
- Basiert auf Burrows-Wheeler transform
- Autor: Julian Seward
- Veröffentlichung: 18. Juli 1996

# Bzip2: Überblick

- Verlustfreies Datenkompressionsverfahren
- Ähnlich zu ZIP
- Basiert auf Burrows-Wheeler transform
- Autor: Julian Seward
- Veröffentlichung: 18. Juli 1996
- Verwendete Algorithmen
  - Run Length Encoding
  - Huffman Encoding
  - Move-to-Front Transform
  - Burrows Wheeler Transform

# Transformation

## Was ist eine Transformation?

- Permutationen (Positionsbezogene Abbildung, spez. Rotation)
- Allgemeine Abbildungen
- Länge Output $\approx$ Länge Input
- Invertierbar (verlustfrei)
- **Idee: Gesteigerte Effizienz einer folgenden Enkodierung**

# Run Length Kodierung

## Überblick

- "Runs" von aufeinander folgenden Zeichen werden zusammengefasst
- **Idee: Große Runs sparen viel Platz**

# Run Length Kodierung

```
function run_length(input):
    out ← ϵ
    counter ← 1
    run_char ← input[1]
    for i in 2..n:
        if input[i−1] = input[i]:
            counter++
        else:
            out.append(counter, run_char)
            counter ← 1
            run_char ← input[i]
    out.append(counter)
    out.append(run_char)
    return out
```

# Huffman Kodierung

## Überblick

- Relative häufigkeit der Zeichen wird betrachtet um eine Entropie-optimale Kodierungstabelle zu erstellen
- Häufig auftretende Zeichen bekommen kleine Kodierungen
- **Idee: Input wird so kodiert, dass die Länge tatsächlich von dem Informationsgehalt der Quelle abhängig ist**

# Huffman Kodierung: Tabelle

```
function huffman_table(input):
    table: string ⇒ string
    heap ← MinHeap()
    for char in input:
        heap.insert(char, occurance(char))
    root ← {}
    while heap.hasItem():
        (char1, char1_occ) ← heap.pop()
        if heap.empty(): break
        (char2, char2_occ) ← heap.pop()
        root ← (char1_occ + char2_occ)
        heap.insert(root)
    for leaf in root:
        table.map(leaf.path → leaf.char)
    return table
```

# Huffman Kodierung

```
function huffman_encode(input):
    table ← huffman_table(input)
    out ← ϵ
    for char in input:
        out.append(table(char))
    return table, out
```

# Huffman Kodierung

```
function huffman_decode(table, input):
    buffer ← ϵ
    out ← ϵ
    for char in input:
        buffer.append(char)
        if buffer in table:
            out.append(table(buffer))
            buffer ← ϵ
    return out
```

# Move-to-Front Transform

## Überblick

- Transformation des Input
- Input wird sequentiell auf Position in einem Buffer abgebildet
- Zeichendistribution wird auf niedrige Codepoints konzentriert
- **Idee: Codepoints werden öfter wiederverwendet**

# Move-to-Front Transform

Sei $\Sigma$ das Eingabealphabet (typisch: $\Sigma$=ASCII, $|\Sigma| = 256$).

```
function mtf(input):
    A[|Σ|] ← Σ
    out ← ϵ
    for char in input:
        char_index ← A.indexOf(char)
        out.append(char_index)
        swap A[1] with A[char_index]
    return A, out
```

# Move-to-Front Transform

```
function inverse_mtf(A, input):
    out ← ϵ
    for char_index in input:
        out.append(A[0])
        swap A[char_index] with A[0]
    return out
```

# Move-to-Front Transform

**Warum ist MTF sinnvoll?**

# Warum MTF sinnvoll?

MTF hat zwei Vorteile:

# Warum MTF sinnvoll?

MTF hat zwei Vorteile:

**Vorteil 1:**

Wiederholende Zeichenfolgen sind nach MTF gut zu enkodieren.

# Warum MTF sinnvoll?

MTF hat zwei Vorteile:

## Vorteil 1:

Wiederholende Zeichenfolgen sind nach MTF gut zu enkodieren.

$$(a_1, \ldots, a_k)^n \overset{\text{MTF}}{\Rightarrow} (c_1, \ldots, c_k) \cdot (k-1)^{k(n-1)}$$

für $(a_1, \ldots, a_k) \in \Sigma^k$ mit $n > 2$ und den Codepoints $c_i$ von $a_i$.

# Warum MTF sinnvoll?

MTF hat zwei Vorteile:

## Vorteil 1:

Wiederholende Zeichenfolgen sind nach MTF gut zu enkodieren.

$$(a_1, \ldots, a_k)^n \overset{\text{MTF}}{\Rightarrow} (c_1, \ldots, c_k) \cdot (k-1)^{k(n-1)}$$

für $(a_1, \ldots, a_k) \in \Sigma^k$ mit $n > 2$ und den Codepoints $c_i$ von $a_i$.

Input: abcdabcdabcdabcdabcdabcd

# Warum MTF sinnvoll?

MTF hat zwei Vorteile:

## Vorteil 1:

Wiederholende Zeichenfolgen sind nach MTF gut zu enkodieren.

$$(a_1, \ldots, a_k)^n \overset{\text{MTF}}{\Rightarrow} (c_1, \ldots, c_k) \cdot (k-1)^{k(n-1)}$$

für $(a_1, \ldots, a_k) \in \Sigma^k$ mit $n > 2$ und den Codepoints $c_i$ von $a_i$.

| | |
|---|---|
| Input: | abcdabcdabcdabcdabcdabcd |
| Output: | 65 66 67 68 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 |

# Warum MTF sinnvoll?

MTF hat zwei Vorteile:

## Vorteil 2:

# Warum MTF sinnvoll?

MTF hat zwei Vorteile:

**Vorteil 2:**

- Häufige Zeichen sind links im Alphabet.
- Transformation erzeugt viele kleine Zahlen.
- Großteil des Outputs $W$ besteht aus Zeichen $Z = \{1, \ldots, 9\}$.
- $|W| \gg |Z| \Rightarrow$ Huffman und Run-Length sind effektiver.

# Burrows-Wheeler Transform

## Überblick

- Transformation des Input
- Input wird basierend auf lexikographischen Vergleichen permutiert
- **Idee: Durchschnittliche Run-Length wird vergrößert**

# Burrows-Wheeler Transform

```
function burrows_wheeler(input):
    rotations[n, n] ← input.rotations()
    rotations.sort()
    return rotations[1..n, n]
```

# Burrows-Wheeler Transform

```
function inverse_burrows_wheeler(input):
    rotations[n, n]
    for k in n..1:
        rotations[1..n, k] ← input
        rotations.sort()
    return rotation ending with $
```

**Warum ist BWT sinnvoll?**

# Warum ist BWT sinnvoll?

Folgender Satz demonstriert die Stärke des BWT:

eulen heulen wegen beulen

# Warum ist BWT sinnvoll?

Folgender Satz demonstriert die Stärke des BWT:

eulen heulen wegen beulen

**Hier fällt auf:**

# Warum ist BWT sinnvoll?

Folgender Satz demonstriert die Stärke des BWT:

eulen heulen wegen beulen

**Hier fällt auf:**

- Nach "e" ist immer ein "n".

# Warum ist BWT sinnvoll?

Folgender Satz demonstriert die Stärke des BWT:

eulen heulen wegen beulen

**Hier fällt auf:**

- Nach "e" ist immer ein "n".
- Nach "n" ist immer ein " ".

# Warum ist BWT sinnvoll?

Folgender Satz demonstriert die Stärke des BWT:

eulen heulen wegen beulen

**Hier fällt auf:**

- Nach "e" ist immer ein "n".
- Nach "n" ist immer ein " ".
- Nach "e" ist immer ein "u".

# Warum ist BWT sinnvoll?

Folgender Satz demonstriert die Stärke des BWT:

eulen heulen wegen beulen

**Hier fällt auf:**

- Nach "e" ist immer ein "n".
- Nach "n" ist immer ein " ".
- Nach "e" ist immer ein "u".
- Nach "u" ist immer ein "l".

# Warum ist BWT sinnvoll?

Folgender Satz demonstriert die Stärke des BWT:

eulen heulen wegen beulen

**Hier fällt auf:**

- Nach "e" ist immer ein "n".
- Nach "n" ist immer ein " ".
- Nach "e" ist immer ein "u".
- Nach "u" ist immer ein "l".

**Diese Regelmäßigkeiten wollen wir nutzen!**

# Warum ist BWT sinnvoll?

eulen heulen wegen beulen

# Warum ist BWT sinnvoll?

eulen heulen wegen beulen

```
eulen heulen wegen beulen§
§eulen heulen wegen beulen
n§eulen heulen wegen beule
en§eulen heulen wegen beul
len§eulen heulen wegen beu
ulen§eulen heulen wegen be
eulen§eulen heulen wegen b
beulen§eulen heulen wegen
 beulen§eulen heulen wegen
n beulen§eulen heulen wege
en beulen§eulen heulen weg
gen beulen§eulen heulen we
egen beulen§eulen heulen w
wegen beulen§eulen heulen
 wegen beulen§eulen heulen
n wegen beulen§eulen heule
en wegen beulen§eulen heul
len wegen beulen§eulen heu
ulen wegen beulen§eulen he
eulen wegen beulen§eulen h
heulen wegen beulen§eulen
 heulen wegen beulen§eulen
n heulen wegen beulen§eule
en heulen wegen beulen§eul
len heulen wegen beulen§eu
ulen heulen wegen beulen§e
```

# Warum ist BWT sinnvoll?

eulen heulen wegen beulen

```
eulen heulen wegen beulen§
§eulen heulen wegen beulen
n§eulen heulen wegen beule
en§eulen heulen wegen beul
len§eulen heulen wegen beu
ulen§eulen heulen wegen be
eulen§eulen heulen wegen b
beulen§eulen heulen wegen
 beulen§eulen heulen wegen
n beulen§eulen heulen wege
en beulen§eulen heulen weg
gen beulen§eulen heulen we
egen beulen§eulen heulen w
wegen beulen§eulen heulen
 wegen beulen§eulen heulen
n wegen beulen§eulen heule
en wegen beulen§eulen heul
len wegen beulen§eulen heu
ulen wegen beulen§eulen he
eulen wegen beulen§eulen h
heulen wegen beulen§eulen
 heulen wegen beulen§eulen
n heulen wegen beulen§eule
en heulen wegen beulen§eul
len heulen wegen beulen§eu
ulen heulen wegen beulen§e
```

```
 beulen§eulen heulen wegen
 heulen wegen beulen§eulen
 wegen beulen§eulen heulen
beulen§eulen heulen wegen_
egen beulen§eulen heulen w
en beulen§eulen heulen weg
en heulen wegen beulen§eul
en wegen beulen§eulen heul
en§eulen heulen wegen beul
eulen heulen wegen beulen§
eulen wegen beulen§eulen h
eulen§eulen heulen wegen b
gen beulen§eulen heulen we
heulen wegen beulen§eulen_
len heulen wegen beulen§eu
len wegen beulen§eulen heu
len§eulen heulen wegen beu
n beulen§eulen heulen wege
n heulen wegen beulen§eule
n wegen beulen§eulen heule
n§eulen heulen wegen beule
ulen heulen wegen beulen§e
ulen wegen beulen§eulen he
ulen§eulen heulen wegen be
wegen beulen§eulen heulen_
§eulen heulen wegen beulen
```

# Warum ist BWT sinnvoll?

```
 beulen§eulen heulen wegen
 heulen wegen beulen§eulen
 wegen beulen§eulen heulen
beulen§eulen heulen wegen_
egen beulen§eulen heulen w
en beulen§eulen heulen weg
en heulen wegen beulen§eul
en wegen beulen§eulen heul
en§eulen heulen wegen beul
eulen heulen wegen beulen§
eulen wegen beulen§eulen h
eulen§eulen heulen wegen b
gen beulen§eulen heulen we
heulen wegen beulen§eulen_
len heulen wegen beulen§eu
len wegen beulen§eulen heu
len§eulen heulen wegen beu
n beulen§eulen heulen wege
n heulen wegen beulen§eule
n wegen beulen§eulen heule
n§eulen heulen wegen beule
ulen heulen wegen beulen§e
ulen wegen beulen§eulen he
ulen§eulen heulen wegen be
wegen beulen§eulen heulen_
§eulen heulen wegen beulen
```

# Warum ist BWT sinnvoll?

```
 beulen§eulen heulen wegen
 heulen wegen beulen§eulen
 wegen beulen§eulen heulen
beulen§eulen heulen wegen_
egen beulen§eulen heulen w
en beulen§eulen heulen weg
en heulen wegen beulen§eul
en wegen beulen§eulen heul
en§eulen heulen wegen beul
eulen heulen wegen beulen§
eulen wegen beulen§eulen h
eulen§eulen heulen wegen b
gen beulen§eulen heulen we
heulen wegen beulen§eulen_
len heulen wegen beulen§eu
len wegen beulen§eulen heu
len§eulen heulen wegen beu
n beulen§eulen heulen wege
n heulen wegen beulen§eule
n wegen beulen§eulen heule
n§eulen heulen wegen beule
ulen heulen wegen beulen§e
ulen wegen beulen§eulen he
ulen§eulen heulen wegen be
wegen beulen§eulen heulen_
§eulen heulen wegen beulen
```

| | |
|---|---|
| Input: | eulen heulen wegen beulen |
| Output: | nnn wgll§hbe uuueeeeeee n |

# Warum ist BWT sinnvoll?

```
 beulen§eulen heulen wegen
 heulen wegen beulen§eulen
 wegen beulen§eulen heulen
beulen§eulen heulen wegen_
egen beulen§eulen heulen w
en beulen§eulen heulen weg
en heulen wegen beulen§eul
en wegen beulen§eulen heul
en§eulen heulen wegen beul
eulen heulen wegen beulen§
eulen wegen beulen§eulen h
eulen§eulen heulen wegen b
gen beulen§eulen heulen we
heulen wegen beulen§eulen_
len heulen wegen beulen§eu
len wegen beulen§eulen heu
len§eulen heulen wegen beu
n beulen§eulen heulen wege
n heulen wegen beulen§eule
n wegen beulen§eulen heule
n§eulen heulen wegen beule
ulen heulen wegen beulen§e
ulen wegen beulen§eulen he
ulen§eulen heulen wegen be
wegen beulen§eulen heulen_
§eulen heulen wegen beulen
```

| | |
|---|---|
| Input: | eulen heulen wegen beulen |
| Output: | nnn wgll§hbe uuueeeeeee n |

Betrachte die roten Zeilen:

# Warum ist BWT sinnvoll?

```
 beulen§eulen heulen wegen
 heulen wegen beulen§eulen
 wegen beulen§eulen heulen
beulen§eulen heulen wegen_
egen beulen§eulen heulen w
en beulen§eulen heulen weg
en heulen wegen beulen§eul
en wegen beulen§eulen heul
en§eulen heulen wegen beul
eulen heulen wegen beulen§
eulen wegen beulen§eulen h
eulen§eulen heulen wegen b
gen beulen§eulen heulen we
heulen wegen beulen§eulen_
len heulen wegen beulen§eu
len wegen beulen§eulen heu
len§eulen heulen wegen beu
n beulen§eulen heulen wege
n heulen wegen beulen§eule
n wegen beulen§eulen heule
n§eulen heulen wegen beule
ulen heulen wegen beulen§e
ulen wegen beulen§eulen he
ulen§eulen heulen wegen be
wegen beulen§eulen heulen_
§eulen heulen wegen beulen
```

| | |
|---|---|
| Input: | eulen heulen wegen beulen |
| Output: | nnn wgll§hbe uuueeeeeee n |

Betrachte die roten Zeilen:

- n ist der erste Buchstabe der Zeilen

# Warum ist BWT sinnvoll?

beulen§eulen heulen wegen
heulen wegen beulen§eulen
wegen beulen§eulen heulen
beulen§eulen heulen wegen_
egen beulen§eulen heulen w
en beulen§eulen heulen weg
en heulen wegen beulen§eul
en wegen beulen§eulen heul
en§eulen heulen wegen beul
eulen heulen wegen beulen§
eulen wegen beulen§eulen h
eulen§eulen heulen wegen b
gen beulen§eulen heulen we
heulen wegen beulen§eulen_
len heulen wegen beulen§eu
len wegen beulen§eulen heu
len§eulen heulen wegen beu
n beulen§eulen heulen wege
n heulen wegen beulen§eule
n wegen beulen§eulen heule
n§eulen heulen wegen beule
ulen heulen wegen beulen§e
ulen wegen beulen§eulen he
ulen§eulen heulen wegen be
wegen beulen§eulen heulen_
§eulen heulen wegen beulen

| | |
|---|---|
| Input: | eulen heulen wegen beulen |
| Output: | nnn wgll§hbe uuueeeeeee n |

Betrachte die roten Zeilen:

- n ist der erste Buchstabe der Zeilen
  - Im Satz ist vor n immer ein e

# Warum ist BWT sinnvoll?

beulen§eulen heulen wegen
heulen wegen beulen§eulen
wegen beulen§eulen heulen
beulen§eulen heulen wegen_
egen beulen§eulen heulen w
en beulen§eulen heulen weg
en heulen wegen beulen§eu1
en wegen beulen§eulen heu1
en§eulen heulen wegen beu1
eulen heulen wegen beulen§
eulen wegen beulen§eulen h
eulen§eulen heulen wegen b
gen beulen§eulen heulen we
heulen wegen beulen§eulen_
len heulen wegen beulen§eu
len wegen beulen§eulen heu
len§eulen heulen wegen beu
n beulen§eulen heulen wege
n heulen wegen beulen§eule
n wegen beulen§eulen heule
n§eulen heulen wegen beule
ulen heulen wegen beulen§e
ulen wegen beulen§eulen he
ulen§eulen heulen wegen be
wegen beulen§eulen heulen_
§eulen heulen wegen beulen

| | |
|---|---|
| Input: | eulen heulen wegen beulen |
| Output: | nnn wgll§hbe uuueeeeeee n |

Betrachte die roten Zeilen:

- n ist der erste Buchstabe der Zeilen
  - Im Satz ist vor n immer ein e
  - Alle e sind in aufeinanderfolgenden Zeilen

# Warum ist BWT sinnvoll?

beulen§eulen heulen wegen
heulen wegen beulen§eulen
wegen beulen§eulen heulen
beulen§eulen heulen wegen_
egen beulen§eulen heulen w
en beulen§eulen heulen weg
en heulen wegen beulen§eul
en wegen beulen§eulen heul
en§eulen heulen wegen beul
eulen heulen wegen beulen§
eulen wegen beulen§eulen h
eulen§eulen heulen wegen b
gen beulen§eulen heulen we
heulen wegen beulen§eulen_
len heulen wegen beulen§eu
len wegen beulen§eulen heu
len§eulen heulen wegen beu
n beulen§eulen heulen wege
n heulen wegen beulen§eule
n wegen beulen§eulen heule
n§eulen heulen wegen beule
ulen heulen wegen beulen§e
ulen wegen beulen§eulen he
ulen§eulen heulen wegen be
wegen beulen§eulen heulen_
§eulen heulen wegen beulen

| | |
|---|---|
| Input: | eulen heulen wegen beulen |
| Output: | nnn wgll§hbe uuueeeeeee n |

Betrachte die roten Zeilen:

- n ist der erste Buchstabe der Zeilen
  - Im Satz ist vor n immer ein e
  - Alle e sind in aufeinanderfolgenden Zeilen
- Ähnlich für

# Warum ist BWT sinnvoll?

```
 beulen§eulen heulen wegen
 heulen wegen beulen§eulen
 wegen beulen§eulen heulen
beulen§eulen heulen wegen_
egen beulen§eulen heulen w
en beulen§eulen heulen weg
en heulen wegen beulen§eul
en wegen beulen§eulen heul
en§eulen heulen wegen beul
eulen heulen wegen beulen§
eulen wegen beulen§eulen h
eulen§eulen heulen wegen b
gen beulen§eulen heulen we
heulen wegen beulen§eulen_
len heulen wegen beulen§eu
len wegen beulen§eulen heu
len§eulen heulen wegen beu
n beulen§eulen heulen wege
n heulen wegen beulen§eule
n wegen beulen§eulen heule
n§eulen heulen wegen beule
ulen heulen wegen beulen§e
ulen wegen beulen§eulen he
ulen§eulen heulen wegen be
wegen beulen§eulen heulen_
§eulen heulen wegen beulen
```

| | |
|---|---|
| Input: | eulen heulen wegen beulen |
| Output: | nnn wgll§hbe uuueeeeeee n |

Betrachte die roten Zeilen:

- n ist der erste Buchstabe der Zeilen
  - Im Satz ist vor n immer ein e
  - Alle e sind in aufeinanderfolgenden Zeilen
- Ähnlich für u

# Warum ist BWT sinnvoll?

```
 beulen§eulen heulen wegen
 heulen wegen beulen§eulen
 wegen beulen§eulen heulen
beulen§eulen heulen wegen_
egen beulen§eulen heulen w
en beulen§eulen heulen weg
en heulen wegen beulen§eul
en wegen beulen§eulen heul
en§eulen heulen wegen beul
eulen heulen wegen beulen§
eulen wegen beulen§eulen h
eulen§eulen heulen wegen b
gen beulen§eulen heulen we
heulen wegen beulen§eulen_
len heulen wegen beulen§eu
len wegen beulen§eulen heu
len§eulen heulen wegen beu
n beulen§eulen heulen wege
n heulen wegen beulen§eule
n wegen beulen§eulen heule
n§eulen heulen wegen beule
ulen heulen wegen beulen§e
ulen wegen beulen§eulen he
ulen§eulen heulen wegen be
wegen beulen§eulen heulen_
§eulen heulen wegen beulen
```

| | |
|---|---|
| Input: | eulen heulen wegen beulen |
| Output: | nnn wgll§hbe uuueeeeeee n |

Betrachte die roten Zeilen:

- n ist der erste Buchstabe der Zeilen
  - Im Satz ist vor n immer ein e
  - Alle e sind in aufeinanderfolgenden Zeilen
- Ähnlich für u, e

# Warum ist BWT sinnvoll?

beulen§eulen heulen wegen
heulen wegen beulen§eulen
wegen beulen§eulen heulen
beulen§eulen heulen wegen_
egen beulen§eulen heulen w
en beulen§eulen heulen weg
en heulen wegen beulen§eul
en wegen beulen§eulen heul
en§eulen heulen wegen beul
eulen heulen wegen beulen§
eulen wegen beulen§eulen h
eulen§eulen heulen wegen b
gen beulen§eulen heulen we
heulen wegen beulen§eulen_
len heulen wegen beulen§eu
len wegen beulen§eulen heu
len§eulen heulen wegen beu
n beulen§eulen heulen wege
n heulen wegen beulen§eule
n wegen beulen§eulen heule
n§eulen heulen wegen beule
ulen heulen wegen beulen§e
ulen wegen beulen§eulen he
ulen§eulen heulen wegen be
wegen beulen§eulen heulen_
§eulen heulen wegen beulen

| | |
|---|---|
| Input: | eulen heulen wegen beulen |
| Output: | nnn wgll§hbe uuueeeeeee n |

Betrachte die roten Zeilen:

- n ist der erste Buchstabe der Zeilen
  - Im Satz ist vor n immer ein e
  - Alle e sind in aufeinanderfolgenden Zeilen
- Ähnlich für u, e und n.

# Warum ist der BWT sinnvoll?

| | |
|---|---|
| Input: | eulen heulen wegen beulen |
| Output: | nnn wglll§hbe uuueeeeeee n |

# Warum ist der BWT sinnvoll?

| | |
|---|---|
| Input: | eulen heulen wegen beulen |
| Output: | nnn wglll§hbe uuueeeeeee n |

Die Run-Length Enkodierung ergibt folgendes:

# Warum ist der BWT sinnvoll?

| | |
|---|---|
| Input: | eulen heulen wegen beulen |
| Output: | nnn wglll§hbe uuueeeeeee n |

Die Run-Length Enkodierung ergibt folgendes:

$$\text{Input} \overset{\text{RL}}{\Rightarrow} \text{1e1u1l1e1n1 1h1e1u1l1e1n1 1w1e1g1e1n1 1b1e1u1l1e1n}$$

# Warum ist der BWT sinnvoll?

| | |
|---|---|
| Input: | eulen heulen wegen beulen |
| Output: | nnn wglll§hbe uuueeeeeee n |

Die Run-Length Enkodierung ergibt folgendes:

$$\text{Input} \overset{\text{RL}}{\Rightarrow} \text{1e1u1l1e1n1 1h1e1u1l1e1n1 1w1e1g1e1n1 1b1e1u1l1e1n}$$
$$\text{Output} \overset{\text{RL}}{\Rightarrow} \text{3n1 1w1g3l1§1h1b1e1 3u7e1 1n}$$

# Warum ist der BWT sinnvoll?

| | |
|---|---|
| Input: | eulen heulen wegen beulen |
| Output: | nnn wglll§hbe uuueeeeeee n |

Die Run-Length Enkodierung ergibt folgendes:

$$\text{Input} \overset{\text{RL}}{\Rightarrow} \text{1e1u1l1e1n1 1h1e1u1l1e1n1 1w1e1g1e1n1 1b1e1u1l1e1n}$$
$$\text{Output} \overset{\text{RL}}{\Rightarrow} \text{3n1 1w1g3l1§1h1b1e1 3u7e1 1n}$$

**Wann ist BWT also sinnvoll:**

# Warum ist der BWT sinnvoll?

| | |
|---|---|
| Input: | eulen heulen wegen beulen |
| Output: | nnn wglll§hbe uuueeeeeee n |

Die Run-Length Enkodierung ergibt folgendes:

$$\text{Input} \overset{\text{RL}}{\Rightarrow} \text{1e1u1l1e1n1 1h1e1u1l1e1n1 1w1e1g1e1n1 1b1e1u1l1e1n}$$
$$\text{Output} \overset{\text{RL}}{\Rightarrow} \text{3n1 1w1g3l1§1h1b1e1 3u7e1 1n}$$

**Wann ist BWT also sinnvoll:**

- Bei Texten mit vielen gleichen Buchstabenfolgen.

# Warum ist der BWT sinnvoll?

| | |
|---|---|
| Input: | eulen heulen wegen beulen |
| Output: | nnn wglll§hbe uuueeeeeee n |

Die Run-Length Enkodierung ergibt folgendes:

$$\text{Input} \overset{\text{RL}}{\Rightarrow} \text{1e1u1l1e1n1 1h1e1u1l1e1n1 1w1e1g1e1n1 1b1e1u1l1e1n}$$
$$\text{Output} \overset{\text{RL}}{\Rightarrow} \text{3n1 1w1g3l1§1h1b1e1 3u7e1 1n}$$

**Wann ist BWT also sinnvoll:**

- Bei Texten mit vielen gleichen Buchstabenfolgen.
- Gibt es in Sprachen sehr oft!

# Vielen Dank fürs Zuhören!