Algorithmen 2

Fortsetzung Übung Stringology

Florian Kurpicz

Burrows-Wheeler-Transformation

Karlsruher Institut für Technologie

Definition: Zyklische Rotation

Sei T ein Text der Länge n, dann ist die i-te **zyklische Rotation**

$$T^{(i)} = T[i..n]T[1..i)$$

- i-te zyklische Rotation ist die Konkatenation des i-ten Suffixes und des $(i-1)$-ten prefixes

$T =$ ababcabcabba\$

$T^{(1)}$	$T^{(2)}$	$T^{(3)}$	$T^{(4)}$	$T^{(5)}$	$T^{(6)}$	$T^{(7)}$	$T^{(8)}$	$T^{(9)}$	$T^{(10)}$	$T^{(11)}$	$T^{(12)}$	$T^{(13)}$
a	b	a	b	c	a	b	c	a	b	b	a	\$
b	a	b	c	a	b	c	a	b	b	a	\$	a
a	b	c	a	b	c	a	b	b	a	\$	a	b
b	c	a	b	c	a	b	b	a	\$	a	b	a
c	a	b	c	a	b	b	a	\$	a	b	a	b
a	b	c	a	b	b	a	\$	a	b	a	b	c
b	c	a	b	b	a	\$	a	b	a	b	c	a
c	a	b	b	a	\$	a	b	a	b	c	a	b
a	b	b	a	\$	a	b	a	b	c	a	b	c
b	b	a	\$	a	b	a	b	c	a	b	c	a
b	a	\$	a	b	a	b	c	a	b	c	a	b
a	\$	a	b	a	b	c	a	b	c	a	b	b
\$	a	b	a	b	c	a	b	c	a	b	b	a

Burrows-Wheeler-Transformation

Karlsruher Institut für Technologie

Definition: Zyklische Rotation

Sei T ein Text der Länge n, dann ist die i-te **zyklische Rotation**

$$T^{(i)} = T[i..n]T[1..i)$$

- i-te zyklische Rotation ist die Konkatenation des i-ten Suffixes und des $(i-1)$-ten prefixes

Definition: Burrows-Wheeler Transform (Alt.)

Sei T ein Text und M die Matrix, welche alle zyklischen Rotationen von T in lexikographischer Reihenfolge als Spalten enthält, dann ist die **BWT** die letzte Zeile von M

$T = $ ababcabcabba$

$T^{(1)}$	$T^{(2)}$	$T^{(3)}$	$T^{(4)}$	$T^{(5)}$	$T^{(6)}$	$T^{(7)}$	$T^{(8)}$	$T^{(9)}$	$T^{(10)}$	$T^{(11)}$	$T^{(12)}$	$T^{(13)}$
a	b	a	b	c	a	b	c	a	b	b	a	$
b	a	b	c	a	b	c	a	b	b	a	$	a
a	b	c	a	b	c	a	b	b	a	$	a	b
b	c	a	b	c	a	b	b	a	$	a	b	a
c	a	b	c	a	b	b	a	$	a	b	a	b
a	b	c	a	b	b	a	$	a	b	a	b	c
b	c	a	b	b	a	$	a	b	a	b	c	a
c	a	b	b	a	$	a	b	a	b	c	a	b
a	b	b	a	$	a	b	a	b	c	a	b	c
b	b	a	$	a	b	a	b	c	a	b	c	a
b	a	$	a	b	a	b	c	a	b	c	a	b
a	$	a	b	a	b	c	a	b	c	a	b	b
$	a	b	a	b	c	a	b	c	a	b	b	a

Burrows-Wheeler-Transformation

Karlsruher Institut für Technologie

Definition: Zyklische Rotation

Sei T ein Text der Länge n, dann ist die i-te **zyklische Rotation**

$$T^{(i)} = T[i..n]T[1..i)$$

- i-te zyklische Rotation ist die Konkatenation des i-ten Suffixes und des $(i-1)$-ten prefixes

Definition: Burrows-Wheeler Transform (Alt.)

Sei T ein Text und M die Matrix, welche alle zyklischen Rotationen von T in lexikographischer Reihenfolge als Spalten enthält, dann ist die **BWT** die letzte Zeile von M

T = ababcabcabba$

$T^{(13)}$	$T^{(12)}$	$T^{(1)}$	$T^{(9)}$	$T^{(6)}$	$T^{(3)}$	$T^{(11)}$	$T^{(2)}$	$T^{(10)}$	$T^{(7)}$	$T^{(4)}$	$T^{(8)}$	$T^{(5)}$
$	a	a	a	a	a	b	b	b	b	b	c	c
a	$	b	b	b	b	a	a	b	c	c	a	a
b	a	a	b	c	c	$	b	a	a	a	b	b
a	b	b	a	a	a	a	c	$	b	b	b	c
b	a	c	$	b	b	b	a	a	b	c	a	a
c	b	a	a	b	c	a	b	b	a	a	$	b
a	c	b	b	a	a	b	c	a	$	b	a	b
b	a	c	a	$	b	c	a	b	a	b	b	a
c	b	a	b	a	b	a	b	c	b	a	a	$
a	c	b	c	b	a	b	b	a	a	$	b	a
b	a	b	a	a	$	c	a	b	b	a	c	b
b	b	a	b	b	a	a	$	c	c	b	a	a
a	b	$	c	c	b	b	a	a	a	a	b	b

Burrows-Wheeler-Transformation

Definition: Zyklische Rotation

Sei T ein Text der Länge n, dann ist die i-te **zyklische Rotation**

$$T^{(i)} = T[i..n]T[1..i)$$

- i-te zyklische Rotation ist die Konkatenation des i-ten Suffixes und des $(i-1)$-ten prefixes

Definition: Burrows-Wheeler Transform (Alt.)

Sei T ein Text und M die Matrix, welche alle zyklischen Rotationen von T in lexikographischer Reihenfolge als Spalten enthält, dann ist die **BWT** die letzte Zeile von M

T = ababcabcabba$

$T^{(13)}$	$T^{(12)}$	$T^{(1)}$	$T^{(9)}$	$T^{(6)}$	$T^{(3)}$	$T^{(11)}$	$T^{(2)}$	$T^{(10)}$	$T^{(7)}$	$T^{(4)}$	$T^{(8)}$	$T^{(5)}$
$	a	a	a	a	a	b	b	b	b	b	c	c
a	$	b	b	b	b	a	a	b	c	c	a	a
b	a	a	b	c	c	$	b	a	a	a	b	b
a	b	b	a	a	a	a	c	$	b	b	b	c
b	a	c	$	b	b	b	a	a	b	c	a	a
c	b	a	a	b	c	a	b	b	a	a	$	b
a	c	b	b	a	a	b	c	a	$	b	a	b
b	a	c	a	$	b	c	a	b	a	b	b	a
c	b	a	b	a	b	a	b	c	b	a	a	$
a	c	b	c	b	a	b	b	a	a	$	b	a
b	a	b	a	a	$	c	a	b	b	a	c	b
b	b	a	b	b	a	a	$	c	c	b	a	a
a	b	$	c	c	b	b	a	a	a	a	b	b

Erste und Letzte Zeile

- zwei wichtige Zeilen in der Matrix
- es gibt eine spezielle Beziehung zwischen den beiden Zeilen
- alle anderen Zeilen werden nicht benötigt

Erste Zeile **F**

- enthält alle Zeichen von T in sortierter Reihenfolge

Letzte Zeile **L**

- ist die *BWT*

$T =$ ababcabcabba$

	$T^{(13)}$	$T^{(12)}$	$T^{(1)}$	$T^{(9)}$	$T^{(6)}$	$T^{(3)}$	$T^{(11)}$	$T^{(2)}$	$T^{(10)}$	$T^{(7)}$	$T^{(4)}$	$T^{(8)}$	$T^{(5)}$
F	$	a	a	a	a	a	b	b	b	b	b	c	c
	a	$	b	b	b	b	a	a	b	c	c	a	a
	b	a	a	b	c	c	$	b	a	a	a	b	b
	a	b	b	a	a	a	a	c	$	b	b	b	c
	b	a	c	$	b	b	b	a	a	b	c	a	a
	c	b	a	a	b	c	a	b	b	a	a	$	b
	a	c	b	b	a	a	b	c	a	$	b	a	b
	b	a	c	a	$	b	c	a	b	a	b	b	a
	c	b	a	b	a	b	a	b	c	b	a	a	$
	a	c	b	c	b	a	b	b	a	a	$	b	a
	b	a	b	a	a	$	c	a	b	b	a	c	b
	b	b	a	b	b	a	a	$	c	c	b	a	a
L	a	b	$	c	c	b	b	a	a	a	a	b	b

Ränge der Zeichen

Definition: Rang (im Text)

Sei T ein Text über dem Alphabet Σ, der **Rang** eines Zeichens an der Position $i \in [1, n]$ ist

$$rang(i) = |\{j \in [1, i] \colon T[i] = T[j]\}|$$

- der Rang entspricht der Anzahl gleicher Zeichen, die vorher im Text vorkommen

$T = \texttt{ababcabcabba\$}$

	$T^{(13)}$	$T^{(12)}$	$T^{(1)}$	$T^{(9)}$	$T^{(6)}$	$T^{(3)}$	$T^{(11)}$	$T^{(2)}$	$T^{(10)}$	$T^{(7)}$	$T^{(4)}$	$T^{(8)}$	$T^{(5)}$
F	\$	a	a	a	a	a	b	b	b	b	b	c	c
	a	\$	b	b	b	b	a	a	b	c	c	a	a
	b	a	a	b	c	c	\$	b	a	a	a	b	b
	a	b	b	a	a	a	a	c	\$	b	b	b	c
	b	a	c	\$	b	b	b	a	a	b	c	a	a
	c	b	a	a	b	c	a	b	b	a	a	\$	b
	a	c	b	b	a	a	b	c	a	\$	b	a	b
	b	a	c	a	\$	b	c	a	b	a	b	b	a
	c	b	a	b	a	b	a	b	c	b	a	a	\$
	a	c	b	c	b	a	b	b	a	a	\$	b	a
	b	a	b	a	a	\$	c	a	b	b	a	c	b
	b	b	a	b	b	a	a	\$	c	c	b	a	a
L	a	b	\$	c	c	b	b	a	a	a	a	b	b

Ränge der Zeichen

Karlsruher Institut für Technologie

Definition: Rang (im Text)

Sei T ein Text über dem Alphabet Σ, der **Rang** eines Zeichens an der Position $i \in [1, n]$ ist

$$rang(i) = |\{j \in [1, i] \colon T[i] = T[j]\}|$$

- der Rang entspricht der Anzahl gleicher Zeichen, die vorher im Text vorkommen

T	a	b	a	b	c	a	b	c	a	b	b	a	$
rang	1	1	2	2	1	3	3	2	4	4	5	5	1

$T =$ ababcabcabba$

	$T^{(13)}$	$T^{(12)}$	$T^{(1)}$	$T^{(9)}$	$T^{(6)}$	$T^{(3)}$	$T^{(11)}$	$T^{(2)}$	$T^{(10)}$	$T^{(7)}$	$T^{(4)}$	$T^{(8)}$	$T^{(5)}$
F	$	a	a	a	a	a	b	b	b	b	b	c	c
	a	$	b	b	b	b	a	a	b	c	c	a	a
	b	a	a	b	c	c	$	b	a	a	a	b	b
	a	b	b	a	a	a	a	c	$	b	b	b	c
	b	a	c	$	b	b	b	a	a	b	c	a	a
	c	b	a	a	b	c	a	b	b	a	a	$	b
	a	c	b	b	a	a	b	c	a	$	b	a	b
	b	a	c	a	$	b	c	a	b	a	b	b	a
	c	b	a	b	a	b	a	b	c	b	a	a	$
	a	c	b	c	b	a	b	b	a	a	$	b	a
	b	a	b	a	a	$	c	a	b	b	a	c	b
	b	b	a	b	b	a	a	$	c	c	b	a	a
L	a	b	$	c	c	b	b	a	a	a	a	b	b

Ränge der Zeichen

Definition: Rang (im Text)

Sei T ein Text über dem Alphabet Σ, der **Rang** eines Zeichens an der Position $i \in [1, n]$ ist

$$rang(i) = |\{j \in [1, i] \colon T[i] = T[j]\}|$$

- der Rang entspricht der Anzahl gleicher Zeichen, die vorher im Text vorkommen

T	a	b	a	b	c	a	b	c	a	b	b	a	\$
rang	1	1	2	2	1	3	3	2	4	4	5	5	1

$T =$ ababcabcabba\$

	$T^{(13)}$	$T^{(12)}$	$T^{(1)}$	$T^{(9)}$	$T^{(6)}$	$T^{(3)}$	$T^{(11)}$	$T^{(2)}$	$T^{(10)}$	$T^{(7)}$	$T^{(4)}$	$T^{(8)}$	$T^{(5)}$
F	\$	a	a	a	a	a	b	b	b	b	b	c	c
	1	5	1	4	3	2	5	1	4	3	2	1	2
	a	\$	b	b	b	b	a	a	b	c	c	a	a
	b	a	a	b	c	c	\$	b	a	a	a	b	b
	a	b	b	a	a	a	a	c	\$	b	b	b	c
	b	a	c	\$	b	b	b	a	a	b	c	a	a
	c	b	a	a	b	c	a	b	b	a	a	\$	b
	a	c	b	b	a	a	b	c	a	\$	b	a	b
	b	a	c	a	\$	b	c	a	b	a	b	b	a
	c	b	a	b	a	b	a	b	c	b	a	a	\$
	a	c	b	c	b	a	b	b	a	a	\$	b	a
	b	a	b	a	a	\$	c	a	b	b	a	c	b
	b	b	a	b	b	a	a	\$	c	c	b	a	a
L	a	b	\$	c	c	b	b	a	a	a	a	b	b

Ränge der Zeichen

Definition: Rang (im Text)

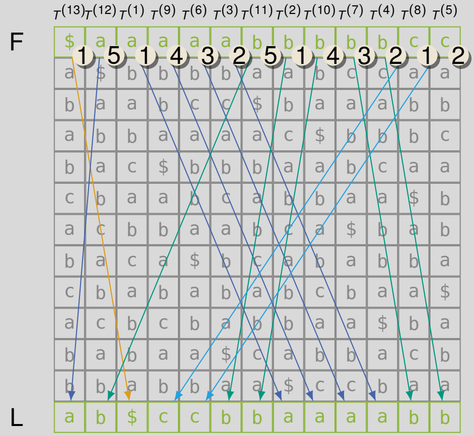
Sei T ein Text über dem Alphabet Σ, der **Rang** eines Zeichens an der Position $i \in [1, n]$ ist

$$rang(i) = |\{j \in [1, i] \colon T[i] = T[j]\}|$$

- der Rang entspricht der Anzahl gleicher Zeichen, die vorher im Text vorkommen

T	a	b	a	b	c	a	b	c	a	b	b	a	\$
rang	1	1	2	2	1	3	3	2	4	4	5	5	1

$T = $ ababcabcabba\$



Ränge der Zeichen

Definition: Rang (im Text)

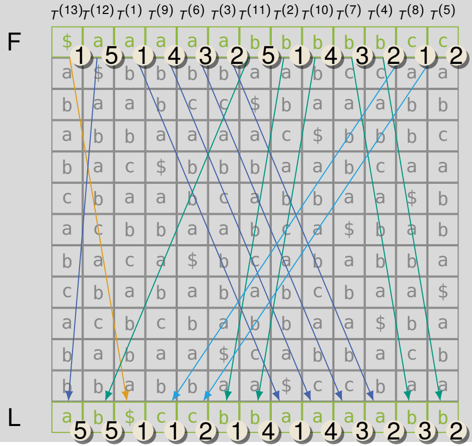
Sei T ein Text über dem Alphabet Σ, der **Rang** eines Zeichens an der Position $i \in [1, n]$ ist

$$rang(i) = |\{j \in [1, i] \colon T[i] = T[j]\}|$$

- der Rang entspricht der Anzahl gleicher Zeichen, die vorher im Text vorkommen

T	a	b	a	b	c	a	b	c	a	b	b	a	$
rang	1	1	2	2	1	3	3	2	4	4	5	5	1

$T =$ ababcabcabba$



Ränge der Zeichen

Karlsruher Institut für Technologie

Definition: Rang (im Text)

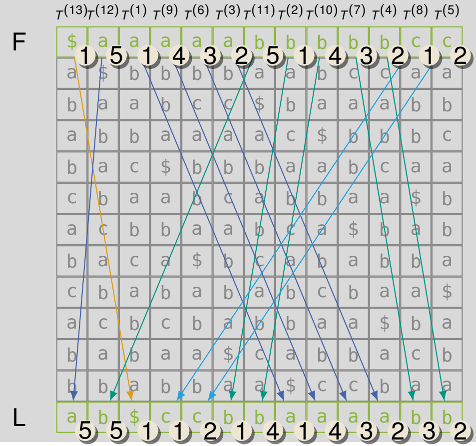
Sei T ein Text über dem Alphabet Σ, der **Rang** eines Zeichens an der Position $i \in [1, n]$ ist

$$rang(i) = |\{j \in [1, i] \colon T[i] = T[j]\}|$$

- der Rang entspricht der Anzahl gleicher Zeichen, die vorher im Text vorkommen
- für jedes Zeichen $\alpha \in \Sigma$ ist die Reihenfolge der Ränge in der ersten und letzten Zeile gleich

T	a	b	a	b	c	a	b	c	a	b	b	a	\$
rang	1	1	2	2	1	3	3	2	4	4	5	5	1

$T = \texttt{ababcabcabba\$}$



LF-Mapping (1/2)

- wir möchten die Zeichen aus der letzten Zeile auf die erste Zeile abbilden
- warum?
  - ermöglicht Mustersuche
  - Transformation *BWT* zurück zu T

Definition: *LF*-Mapping

Sei T ein Text der Länge n und SA das Suffix-Array von T, dann ist das LF-Mapping eine Permutation von $[1, n]$, so dass

$$LF(i) = j \iff SA[j] = SA[i] - 1$$

T = ababcabcabba$

	$T^{(13)}$	$T^{(12)}$	$T^{(1)}$	$T^{(9)}$	$T^{(6)}$	$T^{(3)}$	$T^{(11)}$	$T^{(2)}$	$T^{(10)}$	$T^{(7)}$	$T^{(4)}$	$T^{(8)}$	$T^{(5)}$
F	$	a	a	a	a	a	b	b	b	b	b	c	c
	a	$	b	b	b	b	a	a	b	c	c	a	a
	b	a	a	b	c	c	$	b	a	a	a	b	b
	a	b	b	a	a	a	a	c	$	b	b	b	c
	b	a	c	$	b	b	b	a	a	b	c	a	a
	c	b	a	a	b	c	a	b	b	a	a	$	b
	a	c	b	b	a	a	b	c	a	$	b	a	b
	b	a	c	a	$	b	c	a	b	a	b	b	a
	c	b	a	b	a	b	a	b	c	b	a	a	$
	a	c	b	c	b	a	b	b	a	a	$	b	a
	b	a	b	a	a	$	c	a	b	b	a	c	b
	b	b	a	b	b	a	a	$	c	c	b	a	a
L	a	b	$	c	c	b	b	a	a	a	a	b	b

LF-Mapping (1/2)

- wir möchten die Zeichen aus der letzten Zeile auf die erste Zeile abbilden
- warum?
  - ermöglicht Mustersuche
  - Transformation *BWT* zurück zu *T*

Definition: *LF*-Mapping

Sei *T* ein Text der Länge *n* und *SA* das Suffix-Array von *T*, dann ist das *LF*-Mapping eine Permutation von $[1, n]$, so dass

$$LF(i) = j \iff SA[j] = SA[i] - 1$$

$T = \texttt{ababcabcabba\$}$

	$T^{(13)}$	$T^{(12)}$	$T^{(1)}$	$T^{(9)}$	$T^{(6)}$	$T^{(3)}$	$T^{(11)}$	$T^{(2)}$	$T^{(10)}$	$T^{(7)}$	$T^{(4)}$	$T^{(8)}$	$T^{(5)}$
F	$	a	a	a	a	a	b	b	b	b	b	c	c
	a	$	b	b	b	b	a	a	b	c	c	a	a
	b	a	a	b	c	c	$	b	a	a	a	b	b
	a	b	b	a	a	a	a	c	$	b	b	b	c
	b	a	c	$	b	b	b	a	a	b	c	a	a
	c	b	a	a	b	c	a	b	b	a	a	$	b
	a	c	b	b	a	a	b	c	a	$	b	a	b
	b	a	c	a	$	b	c	a	b	a	b	b	a
	c	b	a	b	a	b	a	b	c	b	a	a	$
	a	c	b	c	b	a	b	b	a	a	$	b	a
	b	a	b	a	a	$	c	a	b	b	a	c	b
	b	b	a	b	b	a	a	$	c	c	b	a	a
L	a	b	$	c	c	b	b	a	a	a	a	b	b

LF-Mapping (1/2)

Karlsruher Institut für Technologie

- wir möchten die Zeichen aus der letzten Zeile auf die erste Zeile abbilden
- warum?
  - ermöglicht Mustersuche
  - Transformation *BWT* zurück zu *T*

Definition: *LF*-Mapping

Sei T ein Text der Länge n und SA das Suffix-Array von T, dann ist das LF-Mapping eine Permutation von $[1, n]$, so dass

$$LF(i) = j \iff SA[j] = SA[i] - 1$$

$T = \texttt{ababcabcabba\$}$

	$T^{(13)}$	$T^{(12)}$	$T^{(1)}$	$T^{(9)}$	$T^{(6)}$	$T^{(3)}$	$T^{(11)}$	$T^{(2)}$	$T^{(10)}$	$T^{(7)}$	$T^{(4)}$	$T^{(8)}$	$T^{(5)}$
F	$	a	a	a	a	a	b	b	b	b	b	c	c
	a	$	b	b	b	b	a	a	b	c	c	a	a
	b	a	a	b	c	c	$	b	a	a	a	b	b
	a	b	b	a	a	a	a	c	$	b	b	b	c
	b	a	c	$	b	b	b	a	a	b	c	a	a
	c	b	a	a	b	c	a	b	b	a	a	$	b
	a	c	b	b	a	a	b	c	a	$	b	a	b
	b	a	c	a	$	b	c	a	b	a	b	b	a
	c	b	a	b	a	b	a	b	c	b	a	a	$
	a	c	b	c	b	a	b	b	a	a	$	b	a
	b	a	b	a	a	$	c	a	b	b	a	c	b
	b	b	a	b	b	a	a	$	c	c	b	a	a
L	a	b	$	c	c	b	b	a	a	a	a	b	b

LF-Mapping (1/2)

- wir möchten die Zeichen aus der letzten Zeile auf die erste Zeile abbilden
- warum?
  - ermöglicht Mustersuche
  - Transformation *BWT* zurück zu T

Definition: *LF*-Mapping

Sei T ein Text der Länge n und SA das Suffix-Array von T, dann ist das *LF*-Mapping eine Permutation von $[1, n]$, so dass

$$LF(i) = j \iff SA[j] = SA[i] - 1$$

$T = \mathtt{ababcabcabba\$}$

	$T^{(13)}$	$T^{(12)}$	$T^{(1)}$	$T^{(9)}$	$T^{(6)}$	$T^{(3)}$	$T^{(11)}$	$T^{(2)}$	$T^{(10)}$	$T^{(7)}$	$T^{(4)}$	$T^{(8)}$	$T^{(5)}$
F	$\$$	a	a	a	a	a	b	b	b	b	b	c	c
	a	$\$$	b	b	b	b	a	a	b	c	c	a	a
	b	a	a	b	c	c	$\$$	b	a	a	a	b	b
	a	b	b	a	a	a	a	c	$\$$	b	b	b	c
	b	a	c	$\$$	b	b	b	a	a	b	c	a	a
	c	b	a	a	b	c	a	b	b	a	a	$\$$	b
	a	c	b	b	a	a	b	c	a	$\$$	b	a	b
	b	a	c	a	$\$$	b	c	a	b	a	b	b	a
	c	b	a	b	a	b	a	b	c	b	a	a	$\$$
	a	c	b	c	b	a	b	b	a	a	$\$$	b	a
	b	a	b	a	a	$\$$	c	a	b	b	a	c	b
	b	b	a	b	b	a	a	$\$$	c	c	b	a	a
L	a	b	$\$$	c	c	b	b	a	a	a	a	b	b

LF-Mapping (1/2)

- wir möchten die Zeichen aus der letzten Zeile auf die erste Zeile abbilden
- warum?
  - ermöglicht Mustersuche
  - Transformation *BWT* zurück zu *T*

Definition: *LF*-Mapping

Sei *T* ein Text der Länge *n* und *SA* das Suffix-Array von *T*, dann ist das *LF*-Mapping eine Permutation von $[1, n]$, so dass

$$LF(i) = j \iff SA[j] = SA[i] - 1$$

$T = \texttt{ababcabcabba\$}$

	$T^{(13)}$	$T^{(12)}$	$T^{(1)}$	$T^{(9)}$	$T^{(6)}$	$T^{(3)}$	$T^{(11)}$	$T^{(2)}$	$T^{(10)}$	$T^{(7)}$	$T^{(4)}$	$T^{(8)}$	$T^{(5)}$
F	$	a	a	a	a	a	b	b	b	b	b	c	c
	a	$	b	b	b	b	a	a	b	c	c	a	a
	b	a	a	b	c	c	$	b	a	a	a	b	b
	a	b	b	a	a	a	a	c	$	b	b	b	c
	b	a	c	$	b	b	b	a	a	b	c	a	a
	c	b	a	a	b	c	a	b	b	a	a	$	b
	a	c	b	b	a	a	b	c	a	$	b	a	b
	b	a	c	a	$	b	c	a	b	a	b	b	a
	c	b	a	b	a	b	a	b	c	b	a	a	$
	a	c	b	c	b	a	b	b	a	a	$	b	a
	b	a	b	a	a	$	c	a	b	b	a	c	b
	b	b	a	b	b	a	a	$	c	c	b	a	a
L	a	b	$	c	c	b	b	a	a	a	a	b	b

LF-Mapping (1/2)

Karlsruher Institut für Technologie

- wir möchten die Zeichen aus der letzten Zeile auf die erste Zeile abbilden
- warum?
  - ermöglicht Mustersuche
  - Transformation *BWT* zurück zu *T*

Definition: *LF*-Mapping

Sei T ein Text der Länge n und SA das Suffix-Array von T, dann ist das LF-Mapping eine Permutation von $[1, n]$, so dass

$$LF(i) = j \iff SA[j] = SA[i] - 1$$

T = ababcabcabba$

LF-Mapping (2/2)

Definition: *C*-Array und *Rang*-Function

Sei T ein Text der Länge n über dem Alphabet Σ, $\alpha \in \Sigma$ und $i \in [1, n]$, dann gilt

$$C[\alpha] = |i \in [1, n] \colon T[i] < \alpha|$$

und

$$rang_\alpha(i) = |\{j \in [1, i] \colon T[j] = \alpha\}|$$

- C enthält die absolute Anzahl kleinerer Zeichen
- $rang_\alpha$ enthält die Anzahl αs in Präfix $T[1..i]$

LF-Mapping (2/2)

Definition: *C*-Array und *Rang*-Function

Sei T ein Text der Länge n über dem Alphabet Σ, $\alpha \in \Sigma$ und $i \in [1, n]$, dann gilt

$$C[\alpha] = |i \in [1, n] \colon T[i] < \alpha|$$

und

$$rang_\alpha(i) = |\{j \in [1, i] \colon T[j] = \alpha\}|$$

- C enthält die absolute Anzahl kleinerer Zeichen
- $rang_\alpha$ enthält die Anzahl αs in Präfix $T[1..i]$

T	a	b	a	b	c	a	b	c	a	b	b	a	$
rang	1	1	2	2	1	3	3	2	4	4	5	5	1

- Ränge jetzt auf der *BWT*
- C ist exklusive Präfixsumme über Histogramm

LF-Mapping (2/2)

Definition: *C*-Array und *Rang*-Function

Sei T ein Text der Länge n über dem Alphabet Σ, $\alpha \in \Sigma$ und $i \in [1, n]$, dann gilt

$$C[\alpha] = |i \in [1, n] \colon T[i] < \alpha|$$

und

$$rang_\alpha(i) = |\{j \in [1, i] \colon T[j] = \alpha\}|$$

- C enthält die absolute Anzahl kleinerer Zeichen
- $rang_\alpha$ enthält die Anzahl αs in Präfix $T[1..i]$

T	a	b	a	b	c	a	b	c	a	b	b	a	$
rang	1	1	2	2	1	3	3	2	4	4	5	5	1

- Ränge jetzt auf der *BWT*
- C ist exklusive Präfixsumme über Histogramm

Definition: *LF*-Mapping (Alt.)

Gegeben die *BWT*, das C-array, und die *rang*-Function, dann gilt

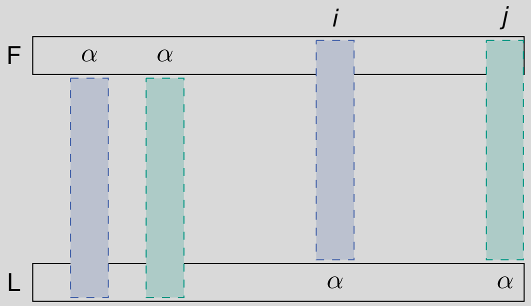
$$LF(i) = C[BWT[i]] + rang_{BWT[i]}(i)$$

Reversing the BWT (1/2)

- Zeichen (bzgl. Text) behalten Reihenfolge in L und F
- LF-Mapping gibt vorheriges Zeichen zurück

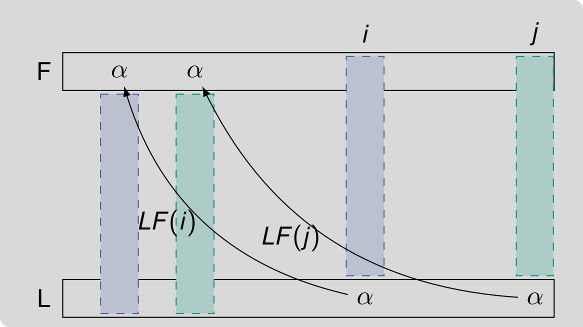
Reversing the BWT (1/2)

- Zeichen (bzgl. Text) behalten Reihenfolge in L und F
- LF-Mapping gibt vorheriges Zeichen zurück



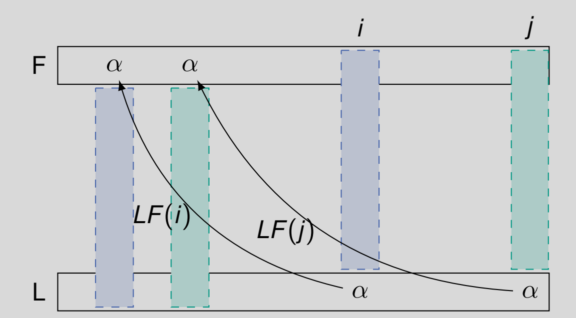
Reversing the BWT (1/2)

- Zeichen (bzgl. Text) behalten Reihenfolge in L und F
- LF-Mapping gibt vorheriges Zeichen zurück



Reversing the BWT (1/2)

- Zeichen (bzgl. Text) behalten Reihenfolge in *L* und *F*
- *LF*-Mapping gibt vorheriges Zeichen zurück

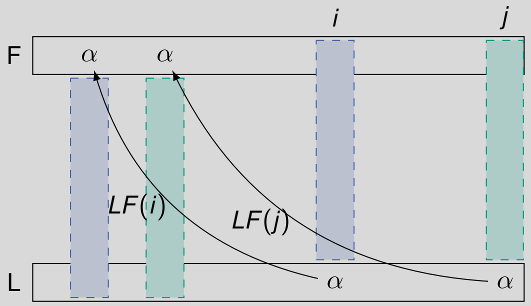


T = ababcabcabba$

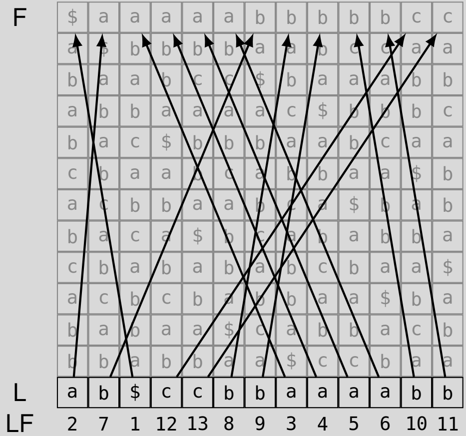
F	$	a	a	a	a	a	b	b	b	b	b	c	c
	a	$	b	b	b	b	a	a	b	c	c	a	a
	b	a	a	b	c	c	$	b	a	a	a	b	b
	a	b	b	a	a	a	a	c	$	b	b	b	c
	b	a	c	$	b	b	b	a	a	b	c	a	a
	c	b	a	a	b	c	a	b	b	a	a	$	b
	a	c	b	b	a	a	b	c	a	$	b	a	b
	b	a	c	a	$	b	c	a	b	a	b	b	a
	c	b	a	b	a	b	a	b	c	b	a	a	$
	a	c	b	c	b	a	b	b	a	a	$	b	a
	b	a	b	a	a	$	c	a	b	b	a	c	b
	b	b	a	b	b	a	a	$	c	c	b	a	a
L	a	b	$	c	c	b	b	a	a	a	a	b	b
LF	2	7	1	12	13	8	9	3	4	5	6	10	11

Reversing the BWT (1/2)

- Zeichen (bzgl. Text) behalten Reihenfolge in L und F
- LF-Mapping gibt vorheriges Zeichen zurück



$T = \texttt{ababcabcabba\$}$



Umkehrung der BWT (2/2)

- Zeichen (bzgl. Text) behalten Reihenfolge in *L* und *F*
- *LF*-Mapping gibt vorheriges Zeichen zurück

	1	2	3	4	5	6	7	8	9	0	11	12	13
L	a	b	$	c	c	b	b	a	a	a	a	b	b
LF	2	7	1	12	13	8	9	3	4	5	6	10	11

Umkehrung der BWT (2/2)

- Zeichen (bzgl. Text) behalten Reihenfolge in L und F
- LF-Mapping gibt vorheriges Zeichen zurück

- $T[n] = \$$ und $T^{(n)}$ in erster Reihe
- wende LF-Mapping auf Ergebnis an und erhalte Zeichen

$$T[n-i] = L[\underbrace{LF(LF(\ldots(}_{i-1 \text{ times}} LF(1))\ldots))]$$

	1	2	3	4	5	6	7	8	9	0	11	12	13
L	a	b	$	c	c	b	b	a	a	a	a	b	b
LF	2	7	1	12	13	8	9	3	4	5	6	10	11

Umkehrung der BWT (2/2)

- Zeichen (bzgl. Text) behalten Reihenfolge in L und F
- LF-Mapping gibt vorheriges Zeichen zurück

- $T[n] = \$$ und $T^{(n)}$ in erster Reihe
- wende LF-Mapping auf Ergebnis an und erhalte Zeichen

$$T[n-i] = L[\underbrace{LF(LF(\ldots(}_{i-1 \text{ times}}LF(1))\ldots))]$$

	1	2	3	4	5	6	7	8	9	0	11	12	13
L	a	b	$	c	c	b	b	a	a	a	a	b	b
LF	2	7	1	12	13	8	9	3	4	5	6	10	11

- $T[13] = \$$ und $k = 1$

Umkehrung der BWT (2/2)

- Zeichen (bzgl. Text) behalten Reihenfolge in L und F
- LF-Mapping gibt vorheriges Zeichen zurück

- $T[n] = \$$ und $T^{(n)}$ in erster Reihe
- wende LF-Mapping auf Ergebnis an und erhalte Zeichen

$$T[n-i] = L[\underbrace{LF(LF(\ldots(}_{i-1 \text{ times}}LF(1))\ldots))]$$

	1	2	3	4	5	6	7	8	9	0	11	12	13
L	a	b	$	c	c	b	b	a	a	a	a	b	b
LF	2	7	1	12	13	8	9	3	4	5	6	10	11

- $T[13] = \$$ und $k = 1$
- $T[12] = L[1] =$ a und $k = LF(1) = 2$

Umkehrung der BWT (2/2)

- Zeichen (bzgl. Text) behalten Reihenfolge in L und F
- LF-Mapping gibt vorheriges Zeichen zurück

- $T[n] = \$$ und $T^{(n)}$ in erster Reihe
- wende LF-Mapping auf Ergebnis an und erhalte Zeichen

$$T[n-i] = L[\underbrace{LF(LF(\ldots(}_{i-1 \text{ times}}LF(1))\ldots))]$$

	1	2	3	4	5	6	7	8	9	0	11	12	13
L	a	b	$	c	c	b	b	a	a	a	a	b	b
LF	2	7	1	12	13	8	9	3	4	5	6	10	11

- $T[13] = \$$ und $k = 1$
- $T[12] = L[1] = \mathtt{a}$ und $k = LF(1) = 2$
- $T[11] = L[2] = \mathtt{b}$ und $k = LF(2) = 7$

Umkehrung der BWT (2/2)

- Zeichen (bzgl. Text) behalten Reihenfolge in L und F
- LF-Mapping gibt vorheriges Zeichen zurück

- $T[n] = \$$ und $T^{(n)}$ in erster Reihe
- wende LF-Mapping auf Ergebnis an und erhalte Zeichen

$$T[n-i] = L[\underbrace{LF(LF(\ldots(}_{i-1 \text{ times}}LF(1))\ldots))]$$

	1	2	3	4	5	6	7	8	9	0	11	12	13
L	a	b	\$	c	c	b	b	a	a	a	a	b	b
LF	2	7	1	12	13	8	9	3	4	5	6	10	11

- $T[13] = \$$ und $k = 1$
- $T[12] = L[1] = \mathrm{a}$ und $k = LF(1) = 2$
- $T[11] = L[2] = \mathrm{b}$ und $k = LF(2) = 7$
- $T[10] = L[7] = \mathrm{b}$ und $k = LF(7) = 9$

Umkehrung der BWT (2/2)

- Zeichen (bzgl. Text) behalten Reihenfolge in L und F
- LF-Mapping gibt vorheriges Zeichen zurück

- $T[n] = \$$ und $T^{(n)}$ in erster Reihe
- wende LF-Mapping auf Ergebnis an und erhalte Zeichen

$$T[n-i] = L[\underbrace{LF(LF(\ldots(}_{i-1 \text{ times}}LF(1))\ldots))]$$

	1	2	3	4	5	6	7	8	9	0	11	12	13
L	a	b	$	c	c	b	b	a	a	a	a	b	b
LF	2	7	1	12	13	8	9	3	4	5	6	10	11

- $T[13] = \$$ und $k = 1$
- $T[12] = L[1] = \mathtt{a}$ und $k = LF(1) = 2$
- $T[11] = L[2] = \mathtt{b}$ und $k = LF(2) = 7$
- $T[10] = L[7] = \mathtt{b}$ und $k = LF(7) = 9$
- $T[9] = L[9] = \mathtt{a}$ und $k = LF(9) = 4$

Umkehrung der BWT (2/2)

- Zeichen (bzgl. Text) behalten Reihenfolge in L und F
- LF-Mapping gibt vorheriges Zeichen zurück

- $T[n] = \$$ und $T^{(n)}$ in erster Reihe
- wende LF-Mapping auf Ergebnis an und erhalte Zeichen

$$T[n-i] = L[\underbrace{LF(LF(\dots(}_{i-1 \text{ times}}LF(1))\dots))]$$

	1	2	3	4	5	6	7	8	9	0	11	12	13
L	a	b	$	c	c	b	b	a	a	a	a	b	b
LF	2	7	1	12	13	8	9	3	4	5	6	10	11

- $T[13] = \$$ und $k = 1$
- $T[12] = L[1] = \mathrm{a}$ und $k = LF(1) = 2$
- $T[11] = L[2] = \mathrm{b}$ und $k = LF(2) = 7$
- $T[10] = L[7] = \mathrm{b}$ und $k = LF(7) = 9$
- $T[9] = L[9] = \mathrm{a}$ und $k = LF(9) = 4$
- $T[9] = L[4] = \mathrm{c}$ und $k = LF(4) = 12$

Umkehrung der BWT (2/2)

- Zeichen (bzgl. Text) behalten Reihenfolge in L und F
- LF-Mapping gibt vorheriges Zeichen zurück

- $T[n] = \$$ und $T^{(n)}$ in erster Reihe
- wende LF-Mapping auf Ergebnis an und erhalte Zeichen

$$T[n-i] = L[\underbrace{LF(LF(\ldots(}_{i-1 \text{ times}}LF(1))\ldots))]$$

	1	2	3	4	5	6	7	8	9	0	11	12	13
L	a	b	$	c	c	b	b	a	a	a	a	b	b
LF	2	7	1	12	13	8	9	3	4	5	6	10	11

- $T[13] = \$$ und $k = 1$
- $T[12] = L[1] = \mathrm{a}$ und $k = LF(1) = 2$
- $T[11] = L[2] = \mathrm{b}$ und $k = LF(2) = 7$
- $T[10] = L[7] = \mathrm{b}$ und $k = LF(7) = 9$
- $T[9] = L[9] = \mathrm{a}$ und $k = LF(9) = 4$
- $T[9] = L[4] = \mathrm{c}$ und $k = LF(4) = 12$
- ...

Eigenschaften von LZ78

Alte Klausuraufgabe

Zeigen Sie, dass die LZ78-Faktorisierung für einen Text der Länge n über einem konstanten Alphabet bis zu $\Theta(\sqrt{n})$ Faktoren erzeugt.

Eigenschaften von LZ78

Karlsruher Institut für Technologie

Alte Klausuraufgabe

Zeigen Sie, dass die LZ78-Faktorisierung für einen Text der Länge n über einem konstanten Alphabet bis zu $\Theta(\sqrt{n})$ Faktoren erzeugt.

- Bei welchen Eingaben erzeugen wir möglichst viele Faktoren?
- Alphabet darf keine Rolle spielen

Eigenschaften von LZ78

Alte Klausuraufgabe

Zeigen Sie, dass die LZ78-Faktorisierung für einen Text der Länge *n* über einem konstanten Alphabet bis zu $\Theta(\sqrt{n})$ Faktoren erzeugt.

- Bei welchen Eingaben erzeugen wir möglichst viele Faktoren?
- Alphabet darf keine Rolle spielen
- $T = a^{n-1}\$$

Eigenschaften von LZ78

Alte Klausuraufgabe

Zeigen Sie, dass die LZ78-Faktorisierung für einen Text der Länge n über einem konstanten Alphabet bis zu $\Theta(\sqrt{n})$ Faktoren erzeugt.

- Bei welchen Eingaben erzeugen wir möglichst viele Faktoren?
- Alphabet darf keine Rolle spielen
- $T = a^{n-1}\$$

- Wie sieht die LZ78-Faktorisierung davon aus?

Eigenschaften von LZ78

Karlsruher Institut für Technologie

Alte Klausuraufgabe

Zeigen Sie, dass die LZ78-Faktorisierung für einen Text der Länge n über einem konstanten Alphabet bis zu $\Theta(\sqrt{n})$ Faktoren erzeugt.

- Bei welchen Eingaben erzeugen wir möglichst viele Faktoren?
- Alphabet darf keine Rolle spielen
- $T = a^{n-1}\$$

- Wie sieht die LZ78-Faktorisierung davon aus?
- $T = a|aa|aaa|\dots$

Eigenschaften von LZ78

Karlsruher Institut für Technologie

Alte Klausuraufgabe

Zeigen Sie, dass die LZ78-Faktorisierung für einen Text der Länge n über einem konstanten Alphabet bis zu $\Theta(\sqrt{n})$ Faktoren erzeugt.

- Bei welchen Eingaben erzeugen wir möglichst viele Faktoren?
- Alphabet darf keine Rolle spielen
- $T = a^{n-1}\$$

- Wie sieht die LZ78-Faktorisierung davon aus?
- $T = a|aa|aaa|\ldots$

- Können wir das genauer sagen?

Eigenschaften von LZ78

Karlsruher Institut für Technologie

Alte Klausuraufgabe

Zeigen Sie, dass die LZ78-Faktorisierung für einen Text der Länge n über einem konstanten Alphabet bis zu $\Theta(\sqrt{n})$ Faktoren erzeugt.

- Bei welchen Eingaben erzeugen wir möglichst viele Faktoren?
- Alphabet darf keine Rolle spielen
- $T = a^{n-1}\$$

- Wie sieht die LZ78-Faktorisierung davon aus?
- $T = a|aa|aaa|\dots$

- Können wir das genauer sagen?
- $T = a|aa|aaa|\dots|a^{k-1}|a^k|a^{n-k(k+1)/2}$

Eigenschaften von LZ78

Alte Klausuraufgabe

Zeigen Sie, dass die LZ78-Faktorisierung für einen Text der Länge n über einem konstanten Alphabet bis zu $\Theta(\sqrt{n})$ Faktoren erzeugt.

- Bei welchen Eingaben erzeugen wir möglichst viele Faktoren?
- Alphabet darf keine Rolle spielen
- $T = a^{n-1}\$$

- Wie sieht die LZ78-Faktorisierung davon aus?
- $T = a|aa|aaa|\dots$

- Können wir das genauer sagen?
- $T = a|aa|aaa|\dots|a^{k-1}|a^k|a^{n-k(k+1)/2}$
- für $k(k+1)/2 < n$

Eigenschaften von LZ78

Karlsruher Institut für Technologie

Alte Klausuraufgabe

Zeigen Sie, dass die LZ78-Faktorisierung für einen Text der Länge n über einem konstanten Alphabet bis zu $\Theta(\sqrt{n})$ Faktoren erzeugt.

- Bei welchen Eingaben erzeugen wir möglichst viele Faktoren?
- Alphabet darf keine Rolle spielen
- $T = a^{n-1}\$$

- Wie sieht die LZ78-Faktorisierung davon aus?
- $T = a|aa|aaa|\ldots$

- Können wir das genauer sagen?
- $T = a|aa|aaa|\ldots|a^{k-1}|a^k|a^{n-k(k+1)/2}$
- für $k(k+1)/2 < n$
- also für $k \in \Theta(\sqrt(n))$