Datenkompression: Übungsblatt 3

Enno Ohlebusch
Timo Beller

Das Übungsblatt wird am 04.06.2013 besprochen.

Aufgabe 1

Sei S ein String der Länge n ohne Abschlusszeichen \$ und L die letzte Spalte der Matrix, die alle n Rotationen von S in lexikografischer Reihenfolge enthält. Sei SA das Suffix Array von S und der String $\mathsf{BWT}[1..n]$ definiert durch:

$$\mathsf{BWT}[i] = \begin{cases} S[\mathsf{SA}[i]-1] & \text{falls } \mathsf{SA}[i] > 1 \\ S[n] & \text{falls } \mathsf{SA}[i] = 1 \end{cases}$$

- Finden Sie einen String S für den gilt $L \neq \mathsf{BWT}$.
- Sei f_L die Funktion $(\Sigma^n \to \Sigma^n)$, welche jedem String S den String L zuordnet. Ist f_L bijektiv?
- Sei f_{BWT} die Funktion $(\Sigma^n \to \Sigma^n)$, welche jedem String S den String BWT zuordnet. Ist f_{BWT} bijektiv?
- Sei D_n die Menge der Strings welche das Abschlusszeichen \$ genau einmal am Ende enthalten und (inkl. Abschlusszeichen) n Zeichen lang sind und W_n die Menge der Strings der Länge n, welche das Abschlusszeichen \$ genau einmal (aber an beliebiger Position) enthalten. Sei g_{BWT} die Funktion $(D_n \to W_n)$, welche jedem String S mit Abschlusszeichen \$ den String BWT zuordnet. Ist g_{BWT} bijektiv?

Aufgabe 2

Führen Sie die inverse BWT für *tttaa\$aac* durch, d.h. dekodieren Sie *tttaa\$aac*. Implementieren Sie das Verfahren und erweitern Sie es, um zusätzlich zum urspünglichen String auch dessen Suffix Array zu erhalten.

Aufgabe 3

Implementieren Sie die Move-To-Front-Transformation. Wenden Sie diese auf die Strings $S = \mathit{diesisteintestdiesisteintest\$}$ und $\mathsf{BWT} = \mathit{tt\$ttiittddeessiieeeeiissssnn}$ an. Vergleichen Sie die mittlere Huffman-Codewortlänge der beiden Move-To-Front-Transformierten.

Aufgabe 4

Sei n die Länge und σ die Alphabetgröße der Eingabe. Ist es möglich, die Move-To-Front-Transformation in besserer Zeitkomplexität als $\mathcal{O}(n \cdot \sigma)$ durchzuführen?

Aufgabe 5

Betrachten Sie den String $S = dcdbdcdadcdbdcd\$$. Führen Sie den Induced Sorting Algorithmus mindestens bis zur 1. Rekursionsstufe durch. Wie viele Rekursionsstufen benötigt der Induced Sorting Algorithmus um das Suffix Array von S zu berechnen? Gibt es einen String S' der nicht länger als S ist und für den der Induced Sorting Algorithmus mehr rekursive Aufrufe benötigt als für S?