Marburg University

Department of Computer Science
Prof. Dr. Sebastian Wild

Date: 2025-11-28
Version: 2025-12-01 23:03

Übungsblatt 7 zur Vorlesung Effiziente Algorithmen (Winter 2025/26)

Abgabe: *Bis 2025-12-05 18:00, on ILIAS.*

1. Aufgabe

30 Punkte

Gegeben Sei ein Alphabet $\Sigma = \{\texttt{A}, \texttt{C}, \texttt{G}, \texttt{T}\}$, ein Muster `TCCGA` und der Text

`CATGCACTCTCCAGTATCCGA`

Wenden Sie den Rabin-Karp Algorithmus mit folgender Hash-Funktion an:

$$h(S) = |S|_{\texttt{A}} + 2 \cdot |S|_{\texttt{C}} + 3 \cdot |S|_{\texttt{T}} + 4 \cdot |S|_{\texttt{G}}.$$

Geben Sie in jedem Schritt den berechenten Hash-Wert an, und markieren Sie, welche Buchstaben miteinander tatsächlich verglichen werden.

2. Aufgabe

60 Punkte

Beweisen Sie die folgenden *No-Free-Lunch* Theoreme für verlustfreie Kompression.

1. *Schwache Variante:* Für jeden Kompressions-Algorithmus A und jedes $n \in \mathbb{N}_{\geq 1}$ gibt es einen Input $w \in \Sigma^n$ für welchen $|A(w)| \geq |w|$, d.h. das Resultat der Kompression ist nicht kleiner als der Input.

   Hinweis: Versuchen Sie einen Widerspruchsbeweis. Es gibt mehrere Möglichkeiten, dieses Theorem zu beweisen.

2. *Starke Variante:* Für jeden Kompressions-Algorithmus A und $n \in \mathbb{N}$ gilt:

   $$\bigl|\{w \in \Sigma^{\leq n} : |A(w)| < |w|\}\bigr| \quad < \quad \tfrac{1}{2} \cdot \bigl|\Sigma^{\leq n}\bigr| \,.$$

   D.h. weniger als die Hälfte aller möglichen Inputlängen (bis n) können so komprimiert werden, dass sie kleiner als die ursprüngliche Größe sind.

   Hinweis: Bestimmen Sie zunächst $|\Sigma^{\leq n}|$.

Die Theoreme gelten für jedes nicht unäre Alphabet, aber Sie können sich auf den binären Fall beschränken, d.h. $\Sigma = \{\texttt{0}, \texttt{1}\}$.

$\Sigma^\star$ ist die Menge aller (endlichen) Strings über dem Alphabet Σ. $\Sigma^{\leq n}$ ist die Menge aller String mit Länge $\leq n$. Als Definitionsbereich von (allen) Kompressions-Algorithmen nehmen wir die Menge von (allen) injektiven Funktionen $\Sigma^\star \to \Sigma^\star$ an, d.h. Funktionen, welche jeden möglichen Input String auf einen Output String (Encoding) abbilden, wobei keine zwei Strings auf den gleichen Output abgebildet werden.

3. Aufgabe

40 Punkte

Komprimieren Sie den Text $T =$ `HANNAHBANSBANANASMAN` mit Hilfe von Huffman-Kodierung. Geben Sie als Rechenweg Folgendes an:

1. die Häufigkeiten der Buchstaben,
2. ein schrittweiser Aufbau des Huffman Baums,
3. den Huffman Code
4. den codierten Text.
5. Geben Sie die Kompressionsrate des Ergebnissen an
   (Ignorieren Sie dabei den Platz, der benötigt wird, um den der Huffman Code zu speichern).

Wichtig: Halten Sie sich präzise an die tie-breaking rules aus der Vorlesung:

1. Gibt es mehrere **Code-Tries** mit **gleichem minimalen** Gewicht,
   wählen Sie zuerst den Trie, der den nach dem Alphabet kleinsten Buchstaben enthält.
2. Beim **Verschmelzen** von zwei Tries mit **unterschiedlichem Gewicht**,
   platzieren Sie den Trie mit geringerem Gewicht nach links (`0`-Kante).
3. Beim **Verschmelzen** von zwei Tries mit **gleichem Gewicht**,
   platzieren Sie den Trie mit dem alphabetisch kleinsten Buchstaben nach links.

4. Aufgabe

20 Punkte

Der gegebene binäre String $C \in \{\texttt{0}, \texttt{1}\}^\star$ ist das Resultat einer Codierung eines binären Strings $S \in \{\texttt{0}, \texttt{1}\}^\star$ mit Hilfe von Run-Length Encoding. Decodieren Sie C, d. h. geben Sie S an und geben Sie als Rechenweg jeweils b, ℓ und k an, wie in der Vorlesung vorgestellt.

$$C = \texttt{000100010011100101}$$