Philipps-Universität Marburg
Fachbereich Mathematik und Informatik
Arbeitsgruppe Algorithmik

Wintersemester 2024/2025

Version 2024-11-29 14:45

Prof. Dr. Sebastian Wild
Dr. Nikolaus Glombiewski

Übungen zur Vorlesung
Effiziente Algorithmen

Abgabe: 06.12.2024,
bis **spätestens** 19:00 Uhr
über die ILIAS Plattform

Übungsblatt 6

Aufgabe 6.1: Rabin-Karp (3 Punkte)

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

`CATGCACTCTCCAGTATCCGA`

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

$$h(S) = \#\text{A's in } S + 2 \cdot \#\text{C's in } S + 3 \cdot \#\text{T's in } S + 4 \cdot \#\text{G's in } S.$$

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

Aufgabe 6.2: No Free Lunch (6 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:

   $$\left|\{w \in \Sigma^{\leq n} : |A(w)| < |w|\}\right| \;<\; \tfrac{1}{2} \cdot \left|\Sigma^{\leq n}\right| .$$

   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 $\left|\Sigma^{\leq n}\right|$.

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

$\Sigma^\star$ 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.

Aufgabe 6.3: Huffman-Kodierung (4 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).

Aufgabe 6.4: Run-Length Encoding (2 Punkte)

Der gegebene 01-String C ist das Resultat einer Codierung eines 01-Strings S mit Hilfe von Run-Length Encoding. Decodieren Sie C, d.h. geben Sie S an und geben Sie als Rechenweg jeweils b, l, k an, wie in der Vorlesung vorgestellt.

$$C = 000100010011100101$$

Aufgabe 6.5: Lempel-Ziv (2+3) (5 Punkte)

Für $n \in \mathbb{N}_{\geq 1}$ definieren wir als A^n einen String der aus n Kopien von A besteht. So entspricht beispielsweise A^5 dem String $AAAAA$.

Gegeben sei ein Alphabet aus 128 ASCII Zeichen und einem Wörterbuch mit maximal 4096 Einträgen, welches die einzelnen Zeichen bereits in den ersten 128 Einträgen (d.h. die Einträge $0, \ldots, 127$) gespeichert hat. Entsprechend hat der Buchstabe A den Wörterbuch Eintrag 65. Somit ist die erste freie Position im Wörterbuch 128.

a) Geben Sie die Encodierung für den String A^{16} an. Geben Sie als Rechenweg analog zur Vorlesung die Tabelle (nach dem Eintrag 127) an.

b) Welches ist das kleinste n, sodass das Wörterbuch für die Encodierung von A^n zu klein ist, d.h. ab welchem n benötigt man die Position 4096 im Wörterbuch? Begründen Sie Ihre Lösung.