Übungen zur Vorlesung Sequenzanalyse

Universität Bielefeld, WS 2016/2017
Prof. Dr. Jens Stoye · M.Sc. Tizian Schulz

https://gi.cebitec.uni-bielefeld.de/teaching/2016winter/sa

Übungsblatt 6 vom 6.12.2016
Abgabe in einer Woche vor Beginn der Vorlesung.

Aufgabe 1 (Σ-Baum) (4 Punkte)
Gegeben sei die Menge der Worte $W = \{star, tars, sad, salad, art, card, at, scar, cars, cat\}$.

1. Zeichne den kleinsten Σ-Baum und den kleinsten kompakten Σ^+-Baum, welche alle Worte aus W darstellen.
2. Gib jeweils die Menge der Worte $x \in \Sigma^*$ an, für die $node(x)$ definiert ist.
3. Welche Menge words(T) von Worten wird durch die Bäume dargestellt?

Aufgabe 2 (Suffixbäume) (4 Punkte)

1. Analysiere die *worst-case*- und *average-case*-Laufzeit des WOTD-Algorithmus in eigenen Worten.
2. Zeige, dass der Speicherverbrauch eines Suffixbaums linear bezüglich der Eingabe ist.

Aufgabe 3 (Maximale Repeats) (3+3* Punkte)
Lies den Abschnitt 7.7.3 im Skript über das effiziente Auffinden von maximalen Repeats in einem Text s mit Hilfe des Suffixbaums von s.

1. Stelle den Suffixbaum von $s =$ `CGTATACTATGTAC` auf.
2. **Satz:** In jedem String der Länge n gibt es höchstens n maximale Repeats.

   Argumentiere unter Berücksichtigung des Suffixbaums, warum diese Aussage korrekt ist. Bedenke: Es stimmt nicht, dass an jeder Position nur ein maximales Repeat beginnen oder enden kann.
3. Finde alle maximalen Repeats in s unter Verwendung des im Skript geschilderten Algorithmus. Beschreibe dein Vorgehen beim Annotieren des Suffixbaums aus Aufgabenteil 1.

Aufgabe 4 (Suffixarray) (4 Punkte)
Im Skript ist in Abschnitt 8.3.3 angegeben, wie sich die Arrays *rank* und *lcp* aus dem Suffixarray *pos* effizient berechnen lassen.

1. Implementiere zwei Funktionen, die das *rank*- und *lcp*-Array berechnen, wenn das Array *pos* gegeben ist. Die Funktionen sollen so in ein Programm eingebettet sein, dass der Benutzer nur das *pos*-Array übergeben muss.

   Verwende eine Programmiersprache, die mit deinem Tutor abgesprochen ist und sende ihm deinen Quellcode per Email zu. Beachte, dass die Indizierung mit 0 beginnen soll.
2. Berechne mit deinem Programm das *rank*- und das *lcp*-Array für $s =$ `ATACAATCTCTAT` und gib diese an.