Algorithmen zur Sequenzanalyse

Wintersemester 2019/2020 — **Übungsblatt 7** — Prof. Dr. E. Ohlebusch,

Besprechung am 31.01.2020 — Institut für Theoretische Informatik

---

Aufgabe 7.1.

Die Burrows-Wheeler-Transformationen der Strings S und S^{rev} sind jeweils in einem Wavelet-Baum gespeichert, siehe Abbildungen 1 und 2. Ermitteln Sie mittels Algorithmus 29 im Skript, wie oft der String *GTA* in S vorkommt.

Aufgabe 7.2.

Erweitern Sie die Vorkommen (beginnend mit den gefundenen Vorkommen aus Aufgabe 2) jeweils mit einem Rückwärtssuchschritt um den Buchstaben c und versuchen Sie mit einem Vorwärtssuchschritt das Muster auf der anderen Seite mit dem Watson-Crick-Komplement von c zu erweitern. Wie lautet das längste Muster, das Sie auf diese Weise finden können?
Die Watson-Crick-Paarungen sind A-T und C-G.

Aufgabe 7.3.

Die Zeitkomplexität eines Suchschrittes mittels Algorithmus 29 im Skript wird durch die Zeitkomplexität des Aufrufes der Prozedur *getBounds* bestimmt. Die Ausführung von *getBounds* erfordert $O(\log \sigma)$ Zeit, wenn man den Wavelet-Baum benutzt (Algorithmus 28 im Skript). Geben Sie eine Implementierung von *getBounds* an, die mehr Platz als der Wavelet-Baum (nämlich $O(n\sigma)$ Bits) benötigt, sodass *getBounds* in $O(1)$ Zeit ausgeführt werden kann.

Aufgabe 7.4.

Beweisen Sie, dass folgendes gilt:

$$\sum_{i=0}^{k} \binom{m}{i} (|\Sigma| - 1)^i \in O(m^k |\Sigma|^k)$$

Aufgabe 7.5.

Geben Sie einen Algorithmus an, der mit Hilfe der BWT von S^{rev} das M_{lr}-Array berechnet. Wie ist die Laufzeit Ihres Algorithmus? Wie kann das M_{lr}-Array effizient gespeichert werden?

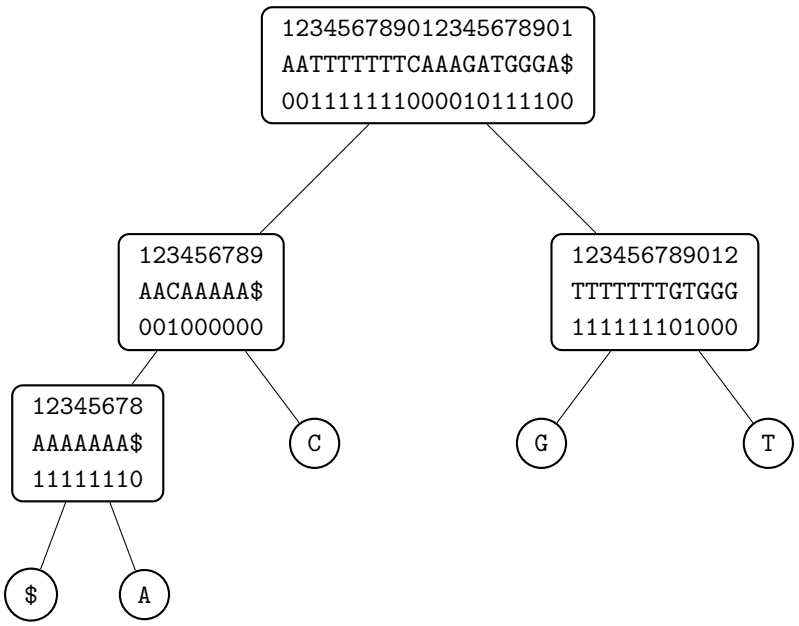

Abbildung 1: Wavelet-Baum von AATTTTTTTCAAAGATGGGA$

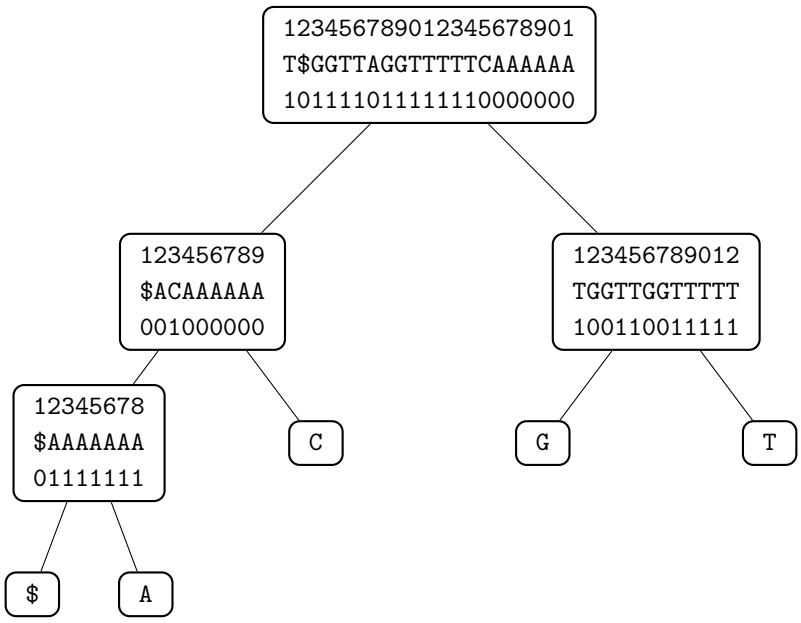

Abbildung 2: Wavelet-Baum von T$GGTTAGGTTTTTCAAAAAA