Algorithmen zur Sequenzanalyse

Wintersemester 2015/2016 — **Übungsblatt 2** — Prof. Dr. E. Ohlebusch, Doktoranden

Besprechung am 13.11.2015 — Institut für Theoretische Informatik

---

Aufgabe 2.1.

In Tabelle 1 ist das Suffix Array von $S = annaassannasananas\$$ gegeben. Berechnen Sie mit Hilfe des Kasai-Algorithmus das LCP-Array von S.

i	SA	$S_{\mathsf{SA}[i]}$
1	19	$
2	4	aassannasananas$
3	13	ananas$
4	15	anas$
5	1	annaassannasananas$
6	8	annasananas$
7	17	as$
8	11	asananas$
9	5	assannasananas$
10	3	naassannasananas$
11	14	nanas$
12	16	nas$
13	10	nasananas$
14	2	nnaassannasananas$
15	9	nnasananas$
16	18	s$
17	12	sananas$
18	7	sannasananas$
19	6	ssannasananas$

Tabelle 1: Das Suffix Array des Strings $S = annaassannasananas\$$.

Aufgabe 2.2.

Der Φ-Algorithmus berechnet aus einem String und dessen Suffix Array das LCP-Array. Im Gegensatz zum Kasai-Algorithmus wird jedoch das so genannte Φ-Array statt dem inversen Suffix Array benutzt. In Algorithmus 1 ist der Beginn des Φ-Algorithmus dargestellt. Vervollständigen Sie den Algorithmus, sodass das LCP-Array berechnet wird. Wenden Sie diesen Algorithmus auf das Suffixarray aus der vorherigen Aufgabe an.

Algorithmus 1 Der Φ-Algorithmus bekommt einen String S der Länge n und dessen Suffix Array SA übergeben. Achtung: Im Algorithmus wird von 0 an gezählt.

```
1:  function Φ-Algorithmus(S, SA)
2:      Φ[0] ← SA[n − 1]
3:      for i ← 1 to n − 1 do
4:          Φ[SA[i]] ← SA[i − 1]
5:      ℓ ← 0
6:      for i ← 0 to n − 1 do
7:          j ← Φ[i]
8:          while S[j + ℓ] = S[i + ℓ] do
9:              ℓ ← ℓ + 1
10:         Φ[i] ← ℓ
11:         ℓ ← max(0, ℓ − 1)
12:     // ToDo
```

Aufgabe 2.3.

Geben Sie einen Algorithmus in Pseudocode an, der bei Eingabe zweier Strings S^1 und S^2 einen längsten gemeinsamen Teilstring von S^1 und S^2 berechnet und ausgibt. Der Algorithmus soll die worst-case Laufzeit von $O(n_1 + n_2)$ haben, wobei n_1 und n_2 die Längen von S^1 und S^2 sind.

Aufgabe 2.4.

Ein Eintrag im LCP-Array gibt die Länge des längsten gemeinsamen Präfixes zweier benachbarter Suffixe im Suffix Array an. Beschreiben Sie eine Methode, die es erlaubt, die Länge des längsten gemeinsamen Präfixes zweier beliebiger Suffixe im Suffix Array in konstanter Zeit zu berechnen (dabei ist eine Vorverarbeitung mit linearer Laufzeit erlaubt).

Hinweis: Benutzen Sie die Tatsache, dass *range minimum queries* auf dem LCP-Array (nach einer Vorverarbeitung mit linearer Laufzeit) in konstanter Zeit beantwortet werden können.