Abteilung für Genominformatik
S. Kurtz, F. Meyer und S. Steinbiß

Universität Hamburg
Zentrum für Bioinformatik

Grundlagen der Sequenzanalyse
Winter
Übungen zur Vorlesung: Ausgabe am 15.10.2013

Punkteverteilung: Aufgabe 1.1: 4 Punkte, Aufgabe 1.2: 4 Punkte, Aufgabe 1.3: 2 Punkte

Abgabe bis zum 21.10.2013. Hinweis zu Aufgaben 1.2 und 1.3: Anstatt Pseudocode kann auch lauffähiger Programmcode abgegeben werden.

Aufgabe 1.1

1. Hamming Distanz: Berechnen Sie, falls möglich, die Hamming-Distanz für alle Paare der folgenden Sequenzen: $s_1 = acgatactag$, $s_2 = aggtcattga$, $s_3 = aggcattga$, $s_4 = cgatactaga$.
2. Euklidische Distanz und Block-Distanz: Berechnen Sie jeweils die Euklidische Distanz und die Block-Distanz für alle Paare der folgenden drei Vektoren der Länge 7:

   $v_1 = (1, 4, 3, 9, 1, 2, 5)$,
   $v_2 = (6, 4, 11, -9, -4, 8, 7)$,
   $v_3 = (5, 1, 4, 3, 9, 1, 2)$.

Aufgabe 1.2 Bei der Analyse von Sequenzen ist es häufig nötig, alle Substrings w der Länge k (k-Worte) einer Sequenz s aufzuzählen. Man kann sich dies als Bewegen eines „Fensters" der Länge k über s vorstellen, so dass der jeweils in Schritt i aktuell betrachtete Inhalt sich in einem Puffer w befindet.

Beispiel für $k = 5$ und $s = gagtccgatggcgttggat$:

$$\overbrace{gagtc}^{i=1}\,cgatggcgttggat \rightarrow g\,\overbrace{agtcc}^{i=2}\,gatggcgttggat \rightarrow ga\,\overbrace{gtccg}^{i=3}\,atggcgttggat \rightarrow \ldots$$
$$\underbrace{}_{w_1} \qquad \underbrace{}_{w_2} \qquad \underbrace{}_{w_3}$$

Bei großen Sequenzen ist es manchmal nicht möglich, die gesamte Sequenz in den Arbeitsspeicher zu laden. Dann ist im Allgemeinen nur ein sequenzieller Zugriff — etwa über einen Massenspeicher — möglich, bei dem sich das jeweils nächste Zeichen der Sequenz über eine gegebene Funktion $getnext(s)$ lesen lässt. Ein Zugriff auf vorher gelesene Zeichen ist also nur möglich, wenn man diese selbst speichert. Beschreiben Sie einen Algorithmus, der die k-Worte w_i von s ausgibt, ohne jedoch s dabei komplett in den Speicher zu laden. Der verwendete Speicherplatz soll auf k Zeichen beschränkt sein. Falls Pseudocode abgegeben wird, muss die Funktion $getnext(s)$ nicht beschrieben werden und kann als gegeben hingenommen werden.

Aufgabe 1.3 Man unterscheidet bei doppelsträngiger DNA zwischen der Basensequenz auf dem *plus*- und dem *minus*-Strang. Durch Basenpaarung bedingt sind diese *revers komplementär* zueinander. Sei $\mathcal{A}_{dna} = \{a, c, g, t\}$ das DNA-Alphabet. Das *Komplement* $\overline{x}$ eines Zeichens $x \in \mathcal{A}_{dna}$ ist dann definiert als $\overline{a} = t$, $\overline{t} = a$, $\overline{c} = g$ und $\overline{g} = c$.

Für eine DNA Sequenz $s \in \mathcal{A}_{dna}^*$ der Länge n ist das *reverse Komplement* $\overleftarrow{s}$ von s definiert als $\overleftarrow{s} := \overline{s[n]}\ \overline{s[n-1]}\ \ldots\ \overline{s[2]}\ \overline{s[1]}$. So ist z.B. $\overleftarrow{gagctgaa} = ttcagctc$.

Beschreiben Sie einen Algorithmus, der zu einer gegebenen Sequenz das reverse Komplement bestimmt, ohne mehr Speicherplatz als die ursprüngliche Sequenz zu verbrauchen, d.h. es soll keine Kopie angelegt werden. (Konstanter Speicherplatz, etwa zum temporären Zwischenspeichern einzelner Zeichen, ist zulässig.)

Die Lösungen zu diesen Aufgaben werden am 22.10.2013 besprochen.