B. Morgenstern / M. Stanke

Übungen zur Vorlesung "Algorithmen der Bioinformatik II"
Wintersemester 2005/2006

Blatt 8

1. Ein neues Sequenzierverfahren, das etwa 100 mal schneller ist, als die Sanger-Sequenziermethode wurde vorgestellt ("Genome sequencing in microfabricated high-density picolitre reactors", Nature, Juli 2005). Die mittlere Readlänge ist dabei allerdings nur $\ell_2 = 100$bp anstatt $\ell_1 = 700$bp wie bei der Sanger-Methode. Die Autoren behaupten: "A completely random genome covered with 100-bases reads requires approximately 50% more reads to yield the same number of contiguous regions (contigs) as achieved with 700-bases reads, assuming the need for a 30-bases overlap between reads". Argumentiere intuitiv, ohne die Lander-Waterman-Gleichungen zu verwenden, warum die Aussage falsch sein muss. Benutze dann die Lander-Waterman-Gleichungen (mit $t = 30$) um folgende Aussage zu zeigen: Wenn die Coverage mit der neuen Methode $c_2 = 40$ ist und c_2 um 50% höher ist als die Coverage c_1 beim Sequenzieren mit der Sanger-Methode (d.h. $c_2 = 1.5c_1$), dann ist unabhängig von der Genomgröße g die Anzahl der Contigs, die sich mit den beiden Methoden ergeben ungefähr gleich groß.

   5 Punkte

2. Wir wollen eine Sequenz T mit der Shotgun-Methode sequenzieren, die einen Repeat X enthält, der sich nicht selbst überlappt:

   

   Zeige, daß der kürzeste gemeinsamen Oberstring S aller Reads ungleich T sein muß, wenn X mehr als zweimal so lang ist wie die längsten Reads.

   5 Punkte

3. Sei $\mathcal{F}$ eine Teilstring-freie Menge von Strings, d.h. kein Element von $\mathcal{F}$ ist Teilstring eines anderen. Der *Überlapp* eines Paares von Strings (f, g) mit $f, g \in \mathcal{F}$ ist die größte Zahl k, so daß das Suffix von f der Länge k ein Präfix von g ist. Die *Vereinigung* von (f, g) sei der String $f[1..|f| - k]g$. Betrachte folgende, gierige Strategie, einen möglichst kurzen gemeinsamen Oberstring von $\mathcal{F}$ zu finden.

   Greedy KGO:
   1) Wenn $\mathcal{F}$ nur aus einem String f besteht, gebe f als gemeinsamen Oberstring aus.
   2) Suche ein Paar von Strings (f, g) in $\mathcal{F}$ mit größtem Überlapp.
   3) Entferne f und g aus $\mathcal{F}$ und füge stattdessen die Vereinigung von (f, g) hinzu.
   4) Gehe zu Schritt 1.

   Gebe für jedes $m > 0$ ein Beispiel einer Teilstring-freien Menge $\mathcal{F}_m$, so daß Greedy KGO einen Oberstring liefert, der mindestens um m länger ist, als der kürzeste gemeinsame Oberstring von $\mathcal{F}$.

   5 Punkte

4. Angenommen, wir wollen die folgenden Reads assemblieren:

   $f_1 =$ ATCCGTTGAAGCCGCGGGC
   $f_2 =$ TTAACTCGAGG
   $f_3 =$ TTAAGTACTGCCCG
   $f_4 =$ ATCTGTGTCGGG
   $f_5 =$ CGACTCCCGACACA
   $f_6 =$ CACAGATCCGTTGAAGCCGCGGG
   $f_7 =$ CTCGAGTTAAGTA
   $f_8 =$ CGCGGGCAGTACTT

   Nimm an, daß die Reads fehlerfrei sind und suche eine möglichst kurze Sequenz S, die für jeden Read r Oberstring von r oder vom reversen Komplement von r ist. Hinweis: Es gibt einen solche Sequenz S der Länge 50. Schreibe hierzu ein Programm, das die gierige Strategie implementiert.

   5 Punkte

5. In der Vorlesung wurde die Burrows-Wheeler Transformation (BWT) T von einem String S definiert. Z.B. ist für S = ABRAKADABRA\$ die BWT T = ARD\$KRAAAABB. S kann aus T auf folgende Weise *zurückermittelt* werden:

   Sortiere die Buchstaben von T lexikographisch und schreibe sie in eine Spalte L neben T. Gehe in die Zeile, in der in T das \$ steht; hier die vierte Zeile. Gebe das Zeichen von L in dieser Zeile aus: A. Es ist das erste Zeichen von S. Dieses A ist das dritte A von oben in L. Gehe in die Zeile mit dem dritten A von oben bei T; die achte Zeile. Gebe das Zeichen von L in dieser Zeile aus: B. Das ist das zweite Zeichen von S. Zähle nun bei jedem ausgegebenen Zeichen x, das wievielte Zeichen von oben dieses x unter allen x in Spalte L ist. Wenn es das k-te x ist, gehe als nächstes in die Zeile, in der das k-te x in der Spalte T steht. Gebe dann das Zeichen in dieser Zeile in der L-Spalte aus und wiederhole bis das letzte Zeichen von S, das \$, ausgegeben wurde. Die Pfeile in nebenstehender Tabelle geben die Sprünge zwischen den Zeilen an. Wir erhalten so den rücktransformierten String ABRAKADABRA\$.

T	L
A	\$
R	A
D	A
\$	A
K	A
R	A
A	B
A	B
A	D
A	K
B	R
B	R

   a) Rücktransformiere den B.-W.-transformierten String T = SBNN\$AAA.

   b) Erkläre, warum oben beschriebenes Verfahren für T immer den String S liefert, dessen BWT T ist.

   5 Punkte

Abgabe bis und Besprechung am Donnerstag, den 26. Januar. 20 Punkte = 100%.