Übungen zur Vorlesung
Grundlagen der Sequenzanalyse

Universität Bielefeld, WS 2006/07
Dr. Sven Rahmann · Dipl.-Bioinf. Katharina Jahn

Blatt 13 vom 25.01.2007
Abgabe am 01.02.2007 vor der Vorlesung um 8:30 in H3

Aufgabe 1 Gegeben sei ein Alphabet der Größe σ. Wir betrachten einen Graphen, dessen Knoten sämtliche σ^n Strings der Länge n über diesem Alphabet sind. Zwei Knoten seinen genau dann durch eine Kante verbunden, wenn sie Hamming-Distanz 1 haben. Wie viele Kanten hat der Graph?

Antwort: Es gibt σ^n Knoten. Jeder hat $(\sigma-1)\cdot n$ Nachbarn mit Hamming-Distanz 1, denn an jeder der n Stellen kann man den dort stehenden Buchstaben durch einen der $(\sigma-1)$ anderen ersetzen. Der Graph hat also $\sigma^n\cdot(\sigma-1)\cdot n/2$ Kanten. Der Faktor $1/2$ ergibt sich, weil man bei der Zählung aller Nachbarn jede Kante zweimal betrachtet.

Aufgabe 2 Ein String s der Länge n hat $n-q+1$ Substrings der Länge q, aber diese sind nicht alle notwendigerweise verschieden: Zum Beispiel hat A^n (der String aus n As) nur einen Substring der Länge q, nämlich A^q. Wie kann man in Linearzeit herausfinden, wie viele verschiedene Teilstrings der Länge q es gibt, wenn man das `lcp`-Array von s hat?

Antwort: Wenn alle Substrings der Länge q verschieden sind, hat man $n-q+1$. Jeder `lcp`-Wert, der $\geq q$ ist, zeigt einen wiederholten Substring der Länge $\geq q$ an. Also müssen wir nur das `lcp`-Array von links nach rechts durchgehen und die Anzahl der Werte $\geq q$ bestimmen. Die gesuchte Zahl ist $n-q+1-|\{i: 0\leq i<n,\ \texttt{lcp}[i]\geq q\}|$.

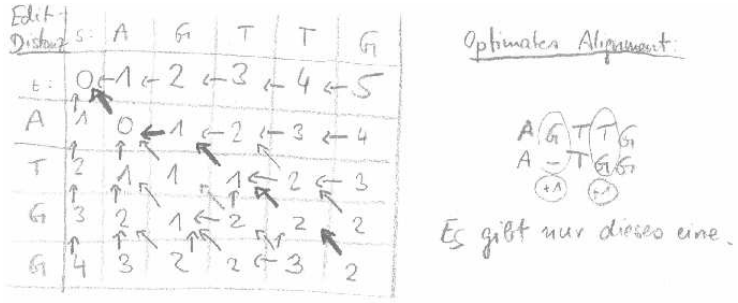
Aufgabe 3 Wir betrachten zwei unabhängige zufällige Sequenzen der Länge n über einem Alphabet der Größe σ. Jeder Buchstabe, darunter A, ist mit gleicher Wahrscheinlichkeit an jeder Position anzutreffen, unabhängig von den anderen Positionen (iid Modell). Wie groß ist die Wahrscheinlichkeit (es genügt eine Formel), dass die erste Sequenz genauso viele As enthält wie die zweite?

Antwort: Die Wahrscheinlichkeit, dass die erste Sequenz genau k As enthält, ist $\binom{n}{k}\cdot\frac{1}{\sigma^k}\cdot\left(1-\frac{1}{\sigma}\right)^{n-k}$. Für die zweite Sequenz ergibt sich die gleiche Wahrscheinlichkeit. Das Produkt muss nun noch über alle Möglichkeiten für k summiert werden. Es ergibt sich:

$$\sum_{k=0}^{n}\left(\binom{n}{k}\cdot\frac{1}{\sigma^k}\cdot\left(1-\frac{1}{\sigma}\right)^{n-k}\right)^2$$

Aufgabe 4 Berechne die Standard-Edit-Distanz zwischen den DNA-Sequenzen $s=$ AGTTG und $t=$ ATGG. Gibt alle(!) optimalen Alignments an.

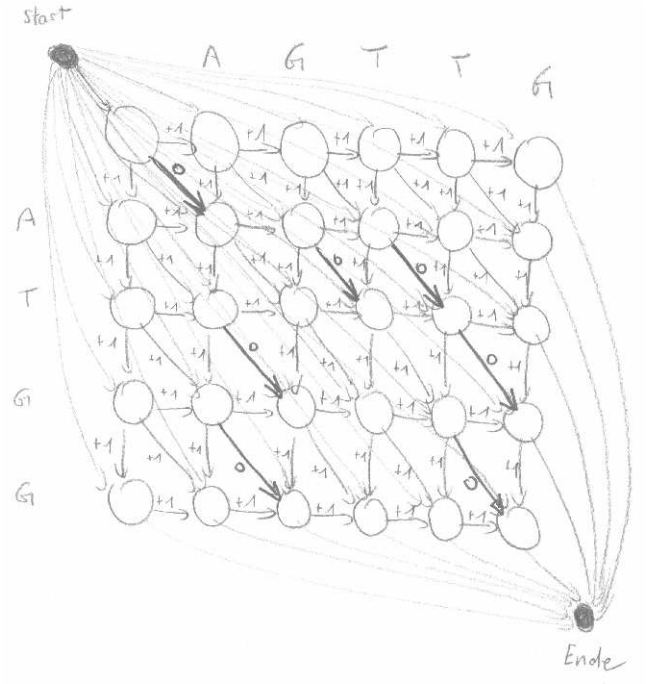
Antwort: Die Edit-Distanz beträgt 2:



Aufgabe 5 Zeichne den lokalen(!) Alignment-Graphen $G(s,t)$ für s und t aus der vorigen Aufgabe. Beschrifte alle Kanten mit ihren Kosten und markiere die minimierenden Kanten. Ermittle alle optimalen lokalen Alignments durch Rückverfolgen aller minimierenden Pfade.

Antwort: Die Aufgabe ist relativ sinnlos, aber das sollte man erkennen! Da alle Kosten nichtnegativ sind, ist das "leere" lokale Alignment immer unter den optimalen! Deswegen sind die Kanten vom Startknoten in jeden Knoten und aus jedem Knoten in den Endknoten immer minimierende Kanten! Diese sind unten nicht explizit eingezeichnet. Außerdem sind nicht alle Kanten, die in den Endknoten münden, eingezeichnet. Die Zeichnung ist schon unübersichtlich genug.

Jeder exakte gemeinsame Teilstring ist ein optimales Alignment mit Kosten Null (leer oder A oder G oder T oder TG, jeweils mit sich selbst).



Aufgabe 6 Gib die reversen Komplemente der DNA-Sequenzen aus der vorigen Aufgabe an. Wie groß ist ihre Edit-Distanz? Warum muss man diese nicht neu berechnen?

Antwort: Das reverse Komplement von s ist CCAT, das von t ist CAACT. Ihre Edit-Distanz ist ebenfalls 2, also gleich der von s und t, weil die Edit-Distanz invariant ist unter (a) Umkehrung der Sequenz, (b) beliebigen Permutationen des Alphabets, also insbesondere A $\leftrightarrow$ T und C $\leftrightarrow$ G.

Aufgabe 7 Wie sieht der Alignment-Graph für Alignments mit kostenfreien Gaps an den Enden der Sequenzen aus (free end gap alignment)? Nimm an, dass die Sequenzen die Längen m und n haben. Gib genau an, welche Knoten und Kanten es in dem Graphen gibt.

Antwort: Es gibt den Start- und den Endknoten und die $(m+1)\cdot(n+1)$ Knoten, die durch ihre Koordinaten (i, j) mit $0 \leq i \leq m$, $0 \leq j \leq n$ bezeichnet werden.

Kanten gehen vom Startknoten zu allen $(i, 0)$ mit $0 \leq i \leq m$ und allen $(0, j)$ mit $0 \leq j \leq n$. Weiterhin von allen (i, n) mit $0 \leq i \leq m$ und von allen (m, j) mit $0 \leq j \leq n$ zum Endknoten. Hinzu kommen die üblichen Kanten

- $(i-1, j) \to (i, j)$ für $1 \leq i \leq m$, $0 \leq j \leq n$,
- $(i, j-1) \to (i, j)$ für $0 \leq i \leq m$, $1 \leq j \leq n$,
- $(i-1, j-1) \to (i, j)$ für $1 \leq i \leq m$, $1 \leq j \leq n$.

Aufgabe 8 Gib eine Rekursionsformel für die Anzahl der Alignments $N(m, n)$ zweier Sequenzen der Längen m und n an und begründe ihre Richtigkeit.

Antwort: Es ist $N(m, 0) = 1$ und $N(0, n) = 1$ für alle natürlichen Zahlen m und n, da es genau eine Möglichkeit gibt, eine Sequenz mit der leeren zu alignieren (nämlich mit m bzw. n gaps).

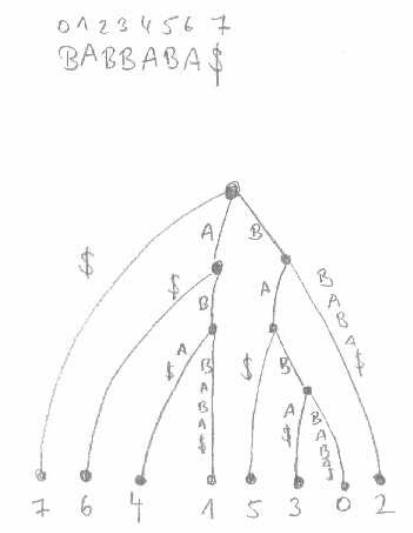
Weiter ist für $m \geq$ und $n \geq 1$:

$$N(m, n) = N(m-1, n-1) + N(m, n-1) + N(m-1, n)$$

Der Grund dafür ist, dass ein Alignment auf genau eine von drei Arten enden muss: Entweder mit einem alignierten Paar (match oder mismatch): Dann gibt es $N(m-1, n-1)$ Möglichkeiten, die Präfixe zu alignieren. Oder mit einem gap in der ersten Sequenz; dann gibt es $N(m, n-1)$ Möglichkeiten, die erste Sequenz mit dem Präfix der zweiten zu alignieren. Oder entsprechend mit einem gap in der zweiten Sequenz. Die genannten Möglichkeiten decken alle Alignments vollständig ab und zählen keins doppelt. Daher gilt Gleichheit in der Formel oben.

Aufgabe 9 Zeichne den Suffixbaum von t = BABBABA\$, wobei \$ < A < B.

Antwort:



Aufgabe 10 Berechne die Links-Rechts-Partition von s = BABAABABB bezüglich t aus der vorigen Aufgabe. Inwiefern ist der Suffixbaum von t dazu nützlich? Welche maximal matches Distanz $\delta(s\|t)$ ergibt sich und warum?

Antwort: Die Links-Rechts-Partition von s ist (BABA, **A**, BABB), da am Anfang BABA das maximale Präfix von s ist, das als Substring in t vorkommt (am Ende). Damit ist das nächste A das Trennzeichen. Der Rest kommt wiederum als Substring in t vor. Da ein Trennzeichen benötigt wurde und die Links-Rechts-Partition minimal ist, gilt $\delta(s\|t) = 1$.

Der Suffixbaum von t wird benötigt, um die maximalen Teilstrings, die an der aktuellen Position von s beginnen und irgendwo in t vorkommen können, in Linearzeit zu finden.

Aufgabe 11 Gib das q-gram Profil von ABCBBCABAA für $q = 2$ an. Verwende die Kodierung A$\mapsto$ 0, B$\mapsto$ 1, C$\mapsto$ 2.

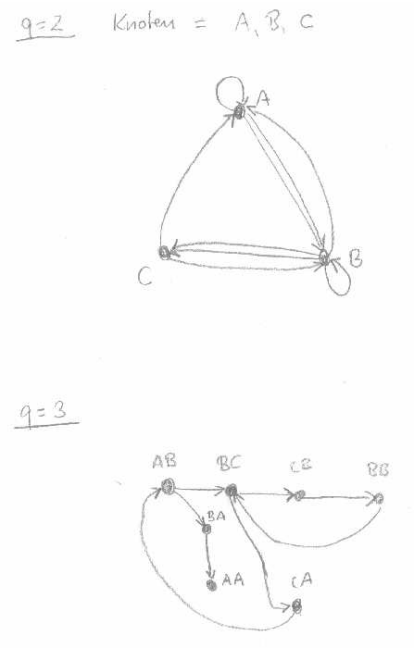
Antwort: Das q-gram Profil ist ein Vektor von Zahlen mit den Häufigkeiten von (in dieser Reihenfolge aufgrund der Kodierung!) AA, AB, AC, BA, BB, BC, CA, CB, CC im String, also:

$$(1, 2, 0, 1, 1, 2, 1, 1, 0)$$

Aufgabe 12 Zeichne den De Bruijn Graphen von ABCBBCABAA für $q = 2$ und auch für $q = 3$. Gibt es jeweils weitere Wörter mit dem gleichen q-gram Profil? Warum (nicht)? Wenn ja, gibt ein Wort an.

Antwort: Graphen siehe Abbildung. Bei $q = 2$ sind alle Knoten balanciert; es gibt viele Eulerkreise (Kreise, die jede Kante einmal benutzen), ein sich daraus ergebendes Wort wäre z.B. AABBABCBCA. Bei $q = 3$ gibt es nur einen Eulerpfad. Beachte, dass die Knoten AB und AA nicht balanciert sind. AB hat eine ausgehende Kante mehr als eingehende Kanten und muss daher der Startknoten

sein. AA hat nur eine eingehende Kante und muss daher der Endknoten sein. Man sieht, dass es nur eine Möglichkeit gibt, von AB nach AA unter Benutzung aller Kanten zu kommen.

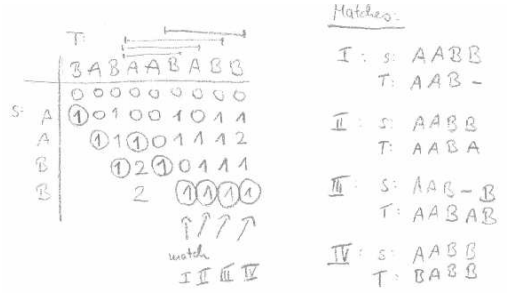


Aufgabe 13 Wie viele Blätter und wie viele innere Knoten haben jeweils die Suffixbäume von AAAAAA\$ und ABCDEF\$?

Antwort: ABCDEF\$: Keinen außer der Wurzel. AAAAAA\$: 5 außer der Wurzel, also mit der Wurzel 6 (aber nicht 7!)

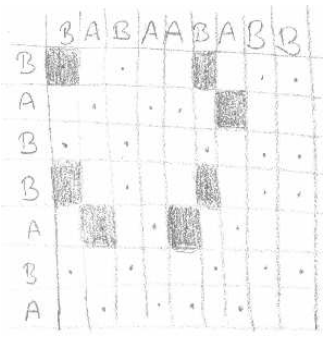
Aufgabe 14 Finde mit Hilfe der cutoff-Verbesserung von Sellers' Algorithmus alle Vorkommnisse von AABB in BABAABABB mit maximal einem Edit-Fehler. Berechne nur die Einträge der Edit-Matrix, die benötigt werden. Gib auch in jeder Spalte den letzten essentiellen Index mit an.

Antwort: Der letzte essentielle Index ist jeweils eingekreist. Es wurden nur die tatsächlich notwendigen Einträge berechnet. Zusätzlich wurden vier 1-Fehler Matches angegeben, die an verschiedenen Positionen im Text enden (es gibt aber für manche Endpositionen mehrere gleich gute Alignments).



Aufgabe 15 Zeichne den gefilterten Dotplot für BABBABA und BABAABABB für $q = 3$ und berechne den FASTA-Score ebenfalls für $q = 3$.

Antwort: siehe Zeichnung: Matches einzelner Zeichen sind kleine Punkte. Ausgemalte Kästen entsprechen einem 3-gram. Der FASTA-Score ist 2, weil es Diagonalen mit 2 ausgemalten Kästchen (q-gram matches) gibt, aber keine mit 3.

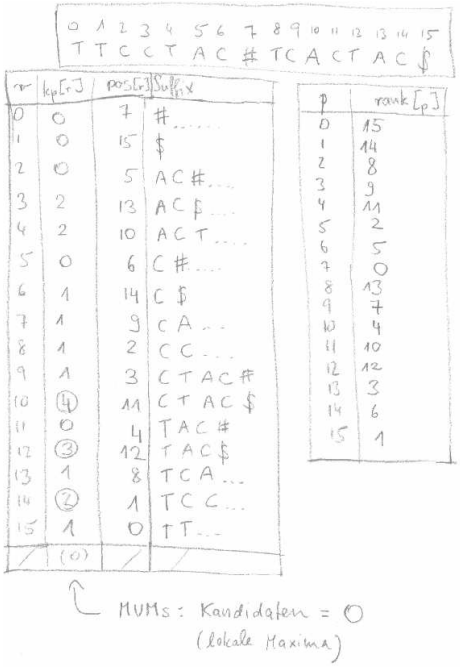


Aufgabe 16 Gib einen Linearzeit-Algorithmus an, mit dem man aus dem Suffixarray `pos` sein Inverses `rank` berechnen kann.

Antwort: Annahme: Beide Arrays sind von 0 bis $n-1$ indiziert.
`for(i=0; i<n; i++) rank[pos[i]]=i;`
Die Korrektheit folgt daraus, dass `pos` eine Permutation ist, also jede Zahl von 0 bis $n-1$ genau einmal enthält.

Aufgabe 17 Gib die Suffixarray-Tabellen `pos`, `rank` und `lcp` für $s\#t\$$ an, wobei s = TTCCTAC und t = TCACTAC und $\# < \$ < \mathrm{A} < \mathrm{C} < \mathrm{T}$.

Antwort:



Aufgabe 18 Finde alle MUMs (maximal unique matches) zwischen s und t aus der vorigen Aufgabe unter Benutzung der dort erstellten Tabellen.

Antwort: Zunächst suchen wir die lokalen Maxima der `lcp`-Tabelle, d.h. die Einträge, die größer sind als ihre beiden Nachbarn. Diese sind eingekringelt ($r \in \{10, 12, 14\}$). Als nächstes stellen wir sicher, dass die zugehörigen Positionen jeweils links und rechts vom Trennzeichen # liegen. Dies ist stets der Fall: (3,11), (4,12), (8,1). Jetzt müssen wir noch die Linksmaximalität überprüfen: CTAC ist linksmaximal, TAC nicht (beide male steht ein C links davon), TC ist linksmaximal. Also sind die MUMs CTAC und TC.

Aufgabe 19 Gib die Definition einer Metrik an.

Antwort: Das kann man direkt aus dem Skript abschreiben.

Aufgabe 20 Für zwei Strings s und t gelte $d_q(s,t) = 4$ für $q = 3$ (q-gram-Distanz), sowie $\delta(s\|t) = 4$ und $\delta(t\|s) = 1$. Was kann man über die Edit-Distanz von s und t sagen?

Antwort: Wenn $d(s,t)$ die Edit-Distanz von s und t ist, dann gilt laut Skript $d_q(s,t)/(2q) = 4/6 \leq d(s,t)$ und $\delta(s\|t) = 4 \leq d(s,t)$ und $\delta(t\|s) = 1 \leq d(s,t)$. Insgesamt folgt also $d(s,t) \geq 4$.

Aufgabe 21 Welcher Zusammenhang besteht im Suffixbaum einer Sequenz s zwischen den Blättern und den Suffixen von s? Welcher Zusammenhang besteht zwischen den verzweigenden Knoten und den rechtsverzweigenden Teilworten von s?

Antwort: Wenn die Sequenz mit einem Zeichen aufhört, das sonst nicht vorkommt, dann gilt: Es gibt eine eins-zu-eins-Beziehung zwischen Blättern und Suffixen und eine eins-zu-eins-Beziehung zwischen den inneren verzweigenden Knoten und rechtsverzweigenden Teilworten von s.

Wenn es ein nichtleeres Suffix von s gibt, das noch einmal als Teilstring von s vorkommt, gelten diese Eigenschaften nicht.

Aufgabe 22 Zeige, dass (und wie!) sich der Suffixbaum einer Sequenz in linearem Platz bezüglich der Sequenzlänge abspeichern lasst.

Antwort: Der Suffixbaum einer Sequenz der Länge n hat (höchstens) so viele Blätter wie Suffixe, also n. Da jeder innere Knoten verzweigend ist, also mindestens zwei Kinder hat, gibt es maximal $n - 1$ innere Knoten, inkl. Wurzel. Die Gesamtheit der Information zu jedem Knoten (z.B. Tiefe, Kinder, etc.) lässt sich in linearem Platz abspeichern, da jeder Knoten (außer der Wurzel) das Kind von genau einem anderen Knoten ist. Ein Problem ist die Annotation der Kanten mit Substrings von s: Statt explizit den Substring mit jeder Kante zu speichern, wird lediglich ein Indexpaar gespeichert, dass z.B. Start- und Endposition eines passenden Substrings von s angibt.