

# Übungen zur Vorlesung Algorithmische Bioinformatik

Freie Universität Berlin, WS 2004/05

Utz J. Pape · Ben Rich · Dr. Stefan Röpcke · Prof. Dr. Martin Vingron

**Blatt 2 · Ausgabe am 25.10.2004**

**Abgabe am 1.11.2004 vor Beginn der Vorlesung**

**Aufgabe 3 (Varianten zum globalen Alignment).** Der Needleman-Wunsch-Algorithmus berechnet das optimale Alignment zweier Sequenzen über ihre gesamte Länge. Durch relativ einfache Abänderungen des Algorithmus lassen sich Varianten auf bestimmte praktische Anwendungen zuschneiden. Schlagen Sie solche Abänderungen vor um die folgenden Teilprobleme zu lösen:

1. Für eine der Sequenzen verursacht das Einfügen von Gaps keine Kosten. Eine Anwendung wäre zum Beispiel das Alignment von cDNA und genomischer Sequenz, die Introns enthält.
2. Führende Gaps im Alignment sollen nicht bestraft werden.
3. Gaps am Ende sollen nicht bestraft werden.

**Aufgabe 4 (Suboptimale globale Alignments).** Wir betrachten den Needleman-Wunsch-Algorithmus für globales Sequenzalignment mit *linearen Gapkosten*. In diesem Fall kann man auf die "langen" horizontalen und vertikalen Kanten im Alignment-Graphen verzichten; in jeden Punkt  $(i, j)$  mit  $i > 0$  und  $j > 0$  münden also genau drei Kanten. Die Sequenzen haben die Längen  $m$  und  $n$  (Index  $i$  läuft von 0 bis  $m$ , Index  $j$  von 0 bis  $n$ ).

1. Oft man ist man nicht am optimalen Score, sondern am optimalen Score unter zusätzlichen Bedingungen interessiert. In dieser Aufgabe soll ein Ankerpunkt  $(k, l)$  des Alignments fest vorgegeben werden. Gesucht ist also der Score des besten Pfades, der von  $(0, 0)$  durch  $(k, l)$  nach  $(m, n)$  führt. Geben Sie einen Algorithmus zur Berechnung dieses Scores an, und beschreiben Sie seine Laufzeit in Abhängigkeit von  $k, l, m, n$ .
2. Jetzt sind wir am Score aus Teil 1 nacheinander für *jeden* Punkt  $(k, l)$  interessiert. Eine wiederholte Anwendung des Algorithmus aus Teil 1 für jeden Punkt  $(k, l)$  würde zu einem Gesamtaufwand von  $O(m^2n^2)$  führen. Finden Sie ein schnelleres Verfahren, das alle Werte in der Gesamtzeit  $O(mn)$  berechnet.

**Aufgabe 5 (Anzahl von Alignments).** Im Needleman-Wunsch Algorithmus verwendet man 'Dynamic Programming', um ein optimales Alignment zu finden, da man den Score nicht für alle möglichen Alignments berechnen will/kann. Wieviele mögliche Alignments gibt es denn, wenn wir zwei Sequenzen der Länge  $n$  betrachten?

1. Berechnen Sie die Anzahl für  $n \in \{1, 2, 3\}$ .
2. Stellen Sie eine rekursive Gleichung zur Berechnung der Anzahl aller möglichen Alignments auf und motivieren Sie diese. *Tip: Überlegen Sie sich zu jedem Punkt der Needleman-Wunsch Matrix, auf wievielen Wegen man diesen erreichen kann.*

3. *Zusatzaufgabe (etwas schwerer)*: Finden Sie eine geschlossene Formel anstelle der rekursiven im letzten Aufgabenteil. Die Äquivalenz der beiden Formeln können Sie auch beweisen.

**Aufgabe 6 (Programmieraufgabe: Waterman-Eggert).** Implementieren Sie den Waterman-Eggert-Algorithmus mit linearen Gapkosten. Lesen Sie dazu das Paper von Waterman und Eggert, welches auf der Vorlesungsseite liegt. Bitte beachten Sie die Programmierhinweise und die Beispiel Eingabe-Datei auf der Vorlesungsseite.

1. Programmieren Sie zunächst den Smith-Waterman Algorithmus. Dabei können Sie natürlich auch Ihren Code von 'Algorithmen und Datenstrukturen' benutzen.
2. Erweitern Sie das Programm so, dass die  $k$  besten Alignments ausgegeben werde ohne jedes Mal die gesamte Matrix neu zu berechnen. Dabei dürfen - wie in dem Paper beschrieben - die  $k$  besten Alignments jeweils keinen Teil des Pfades gemeinsam haben.
3. Testen Sie ihr Programm.