Sommersemester 2003
Dipl.-Inf. A. Thum, Dr. A. Gerisch

8. Übung „Algorithmen der Bioinformatik I“

Abgabe am 26. Juni 2003 in der Übung

Institut für Informatik
Martin-Luther-Universität Halle-Wittenberg

Aufgabe 1. Ermitteln Sie alle optimalen globalen Alignments für die Strings „MUENSTER“ und „MINISTER“, wobei die Kostenfunktion $\delta(a, b)$ für zwei Buchstaben a und b des Alphabets oder Leerstrings λ wie folgt definiert ist:

$$\delta(a,b) = \begin{cases} 0, & a = b \\ 2, & a \neq b, \quad a, b \neq \lambda \\ 1, & a = \lambda \text{ oder } b = \lambda \end{cases}$$

Geben Sie die Edit-Transkripte und die Edit-Distanzen an.

Aufgabe 2. Programmieren Sie den Algorithmus für das optimale globale Alignment von zwei Strings. Die Parameter (Gewichte) der Kostenfunktion $\delta(a, b)$ für zwei Buchstaben a und b bzw. Leerstrings sollen dabei frei wählbar sein und die erhaltene Kostenmatrix $D(i, j)$ soll ausgegeben werden.
Testen Sie Ihren Algorithmus am Beispiel aus Aufgabe 1 und an einem weiteren Beispiel. Dabei soll sowohl die minimale Edit-Distanz (welche Parameter hat die Kostenfunktion hier?) als auch ein zugehöriges optimales Alignment der beiden Strings ausgegeben werden (Leerstellen mit „-“ markieren). Zur Ausgabe eines optimales Alignments muss also eine weitere Funktion programmiert werden, die ein solches aus der Kostenmatrix $D(i, j)$ mit Hilfe von zusätzlich gespeicherten Links von Einträgen von $D(i, j)$ zu den entsprechenden minimierenden Vorgängern extrahiert. Geben Sie die Testergebnisse an!
Hinweis: Wenn Sie eine Möglichkeit finden, ein optimales Alignment ohne die Verwendung zusätzlicher Links aus $D(i, j)$ zu bestimmen, dann müssen Sie im Programm diese Links auch nicht berechnen und speichern.

Aufgabe 3. Die Kostenmatrix $D(i, j)$ aus dem Algorithmus zum Finden optimaler globaler Alignments von zwei Strings enthält auch ohne zusätzlich abzuspeichernde Links zu den minimierenden Vorgängern alle Informationen über Alignments der beiden Strings. Geben Sie einen effizienten Algorithmus an, der **alle** optimalen globalen Alignments der beiden Strings ausgibt, dabei nicht auf eventuell zusätzlich gespeicherte Links zurückgreift und **pro** Alignment Zeit $\mathcal{O}(n + m)$ benötigt.

Aufgabe 4. Entwickeln Sie einen „Dynamisches Programmieren“ Algorithmus, der die **Anzahl** aller optimalen globalen Alignments zweier Strings der Längen n und m in Zeit $\mathcal{O}(nm)$ berechnet und beweisen Sie dessen Korrektheit.