

Wiederholung Alignment

Ein *Sequenzalignment* A zweier Sequenzen a und b über Alphabet Σ ist ein Paar von Wörtern a^\diamond und b^\diamond gleicher Länge über dem Alphabet $\Sigma \cup \{-\}$, so dass aus a^\diamond und b^\diamond durch Löschen der Gapsymbole a und b entsteht und a^\diamond und b^\diamond keine Gapsymbole $-$ an gleicher Position enthalten.
Ein Alignment zweier Teilsequenzen von a und b heißt *lokal für a und b* .

Wiederholung Alignmentsscore

Ein *linearer Alignmentsscore* ist eine Funktion auf Alignments $A = (a^\diamond, b^\diamond)$ der Form

$$\sigma(A) = \sum_{1 \leq i \leq |A|} \sigma(a_i^\diamond, b_i^\diamond)$$

Was ist gesucht?

Gegeben sei eine Bewertungsfunktion σ für Alignments über Σ . Das *Alignmentproblem* besteht darin für zwei Sequenzen über Σ das durch σ am besten bewertete Alignment (bzw. dessen Bewertung) zu finden.

Naiver Algorithmus

- Finde bestes (globales) Alignment durch *vollständige Aufzählung* aller Alignments.
- Wie können alle Alignments von a und b aufgezählt werden?
 - rekursiver Aufbau der Alignmentsequenzen a^\diamond und b^\diamond
 - dabei “Fortschreiten” in a und b (Positionszeiger).
- 3 rekursive Aufrufe in der Funktion:
 - Aligniere aktuelle Zeichen von a und b
 - Aligniere aktuelles Zeichen von a mit Gap
 - Aligniere aktuelles Zeichen von b mit Gap

Naiver Algorithmus lokal

- Wie bekommt man das *beste lokale Alignment* bei Aufzählung aller globalen Alignments (weniger naiv)?
- bestes lokales Alignment ist optimales Teilalignment eines globalen Alignments
- Suche bestes Teilalignment in linearer Zeit
 - Maximale Teilsumme
 - Scanning line-Prinzip

Naiv aber iterativ?

- `while`-Schleife statt rekursiver Aufruf
- Counter für Rekursionstiefe
- Listen/Arrays für Variablen, die beim Aufruf übergeben werden
- Liste/Array für Position in der aufrufenden Funktion

Needleman-Wunsch Alignment

DP-Algorithmus zur Berechnung von globalen Alignments.

- Matrix D mit $n \times m$ Einträgen
- Jeder Matrixeintrag $D(i, j)$ entspricht der besten Bewertung eines Teilalignments von a_1, a_2, \dots, a_i und b_1, b_2, \dots, b_j
- Bewertungsfunktion (Scoring) $\sigma(x, y)$ bewertet zwei Symbole x und y .

Needleman-Wunsch Initialisierung

- $D(0, 0) = 0$
- Initialisierung der zusätzlichen (ersten) Spalte und (ersten) Zeile:
 - $D(i, 0) = D(i - 1, 0) + \sigma(a_i, -); i = 1, 2, \dots, n - 1$
 - $D(0, j) = D(0, j - 1) + \sigma(-, b_j); j = 1, 2, \dots, m - 1$

Nun kann zeilenweise oder spaltenweise ab $i = 1$ und $j = 1$ die Rekurrenz angewandt werden.

Needleman-Wunsch Rekurrenz

$$D(i, j) = \max \begin{cases} D(i-1, j-1) & + \sigma(\mathbf{a}_i, \mathbf{b}_j) \\ D(i-1, j) & + \sigma(\mathbf{a}_i, -) \\ D(i, j-1) & + \sigma(-, \mathbf{b}_j) \end{cases}$$

Needleman-Wunsch Traceback

Berechnung des Alignments aus der gefüllten Matrix D

- Start an $D(n, m)$ und berechnung rückwärts bis zum Eintrag $D(0, 0)$. Aus welcher Richtung kommt $D(n, m)$? Je nach Richtung bzw. Fall baut man das Alignment von hinten entsprechend auf, d.h man aligniert zwei Buchstaben oder fügt ein gap ein.

Smith-Waterman Alignment

- Berechnung der optimalen lokalen Alignments.
- Beim Berechnen der Matrix gibt es keine negativen Werte.
- Das Traceback beginnt bei $\max_{i,j}\{D(i,j)\}$, dem größten Wert der Matrix; es endet, sobald man auf eine Zelle mit Inhalt 0 stößt.

Smith-Waterman Rekurrenz

$$D(i, j) = \max \begin{cases} 0 \\ D(i-1, j-1) + \sigma(a_i, b_j) \\ D(i-1, j) + \sigma(a_i, -) \\ D(i, j-1) + \sigma(-, b_j) \end{cases}$$

Linearer Speicherverbrauch

- Bei quadratischem Speicherbedarf:
 - Alignment von zwei Virus-Genomen zu je *2000 Basen*
⇒ ca. $4 \cdot 2000^2 = 16 \text{ Megabyte}$
 - Alignment von zwei Bakterien-Genomen zu je *1 Mio. Basen*
⇒ ca. $4 \cdot (10^6)^2 = 4 \text{ Terabyte}$
- Idee:
 - es genügt, statt kompletter Matrix die letzte Zeile zu speichern
 - Zeile *i in-place* aus Zeile *i – 1* berechenbar
 - Hilfsvariable für schräg oben
 - Folge: Traceback nicht mehr aus DP-Matrix berechenbar.