Algorithmen auf Sequenzen

Vorlesung von Prof. Dr. Sven Rahmann
im Sommersemester 2008

Kapitel 6
Alignments

Webseite zur Vorlesung
http://ls11-www.cs.tu-dortmund.de/people/rahmann/teaching/ss2008/AlgorithmenAufSequenzen

Sprechstunde
Mo 16-17 in OH14, R214

Globale und Lokale Alignments

Bisher: Problem der approximativen Suche eines Musters in einem Text
- das gesamte Muster wird gesucht („global“ im Muster)
- als ein Teilstring des Textes („lokal“ im Text)

Zu dem Problem gibt es verschiedene Varianten:
- globales Alignment (Sequenzvergleich global/global,
  Text und Muster sollten in etwa gleich lang sein)
- Überlapp-Feststellung (overlap detection), d.h. global bis auf Endstücke,
  Gaps beliebiger Länge am Ende jeder Sequenz sind umsonst (Kosten 0).
- lokales Alignment (lokal/lokal):
  es werden die am besten übereinstimmenden **Teilstrings** in Muster und Text gesucht.
  Achtung: Bei Verwendung von Kosten ist das immer der leere String mit Kosten 0.
  Statt Kosten zu minimieren, maximiert man die Ähnlichkeit (Score),
  wobei sowohl positive wie auch negative Ähnlichkeits-Scores auftreten können.
  Sinnvoll ist das nur, wenn der Vergleich von zufälligen Zeichen im Erwarungswert
  zu einer negativen Ähnlichkeit führt.

Allen Varianten ist gemeinsam, dass der algorithmische Kern,
die Minimierung / Maximierung über drei Vorgängerfälle, derselbe bleibt
(beim lokalen Alignment kommt ein vierter Fall hinzu)
Unterschiede gibt es bei der Initialisierung und bei der Auswertung des Ergebnisses.
Details: Übung

Alignments als Wege maximalen Gewichts in Graphen

Definiere Alignmentgraphen zu zwei Sequenzen s,t:
- Knoten := alle Paare (i,j), sowie „Start“ und „Ende“
- gewichtete Kanten :=
  - (i-1,j-1) -> (i,j) mit Gewicht sim(si,tj) [Ähnlichkeitswert zw. si und tj]
  - (i-1, j) -> (i,j) mit Gewicht -g [Kosten für Gap]
  - (i, j-1) -> (i,j) mit Gewicht -g

Hinzu kommen noch Kanten von „Start“ und in „Ende“, abh. vom Alignment-Typ.
Ihr Gewicht ist immer 0.
- globales Alignment: „Start“ -> (0,0) und (m,n) -> „Ende“
- lokales Alignment:
- Überlapp-Alignment:
- Pattern-Suche in Text (s Pattern, t Text):

Jedem Knoten v ordnet man den maximalen Score S(v) zu,
der sich auf einem Weg von „Start“ nach v erreichen lässt.
Jeder solche Weg entspricht einem Alignment des entsprechenden Typs,
das mit den Symbolen, die v entsprechen, endet.
Da der Graph azyklich ist, funktioniert die Berechnung generisch mit

$$S(v) = \max \{ S(w) + \text{Gewicht}(w \to v) : w \text{ direkter Vorgänger von } v \}.$$

Gapkosten

Bisher:
Summe der Kosten eines Alignments ist die Summe über die Kosten der Spalten.
Häufig sind die Kosten nicht zeichenspezifisch und daher konstant:
Gap der Länge *k* verursacht *k*-fache Kosten eines Gaps der Länge 1 (lineare Gapkosten)

Allgemeineres Modell:
beliebige Gapkosten (nicht zeichenspezifisch):
Gap der Länge *k* verursacht Kosten $g(k)$
Häufig sinnvolle Forderung (aber nicht notwendig): $g(k)$ monoton wachsend.

Das Alignment von und kann nun auf $1+i+j$ verschiedene Arten enden:
Statt über drei Vorgänger muss man die Distanz über $1+i+j$ Fälle minimieren,
bzw. die Ähnlichkeit entsprechend maximieren.

$$D(i,j) = \min \{ \; D(i\text{-}1,j\text{-}1) + d(p_i, T_j),$$
$$\min \{ D(i', j) + g(i - i') \; : 0 <= i' < i \},$$
$$\min \{ D(i, j') + g(j - j') \; : \; 0 <= j' < j \}$$
$$\}$$

Dies führt auf eine Laufzeit von $O(mn\,(m+n))$.
Da dies für die Praxis zu langsam ist, schränkt man das Modell wieder ein.

Affine Gapkosten

Beschränkt man das Modell auf affine Gapkosten $g(k) = o + ek$,
so lässt sich das Problem mit einem Trick wieder in $O(mn)$ Zeit lösen.
Bezeichnungen: o heißt **gap-open** Kosten, e heißt **gap-extend** Kosten.

Beispiel: lokales Alignment von s,t mit Score-Maximierung.

Wir definieren die DP-Matrix
$S(i,j)$:= maximaler Score aller lokalen Alignments, die mit s_i und t_j enden.
Es gilt stets $S(i,j) >= 0$, da das leere Alignment immer zugelassen ist.

Man beweist mit demselben Argument wie beim globalen Alignment, dass gilt:
$S(i,0) = S(0,j) = 0$ für alle i und j;

$$S(i,j) = \max \{ \; S(i-1,j-1) + \text{sim}(s_i, t_j),$$
$$\max \{ S(i', j) - g(i - i') \; : 0 <= i' < i \},$$
$$\max \{ S(i, j') - g(j - j') \; : \; 0 <= j' < j \},$$
$$0$$
$$\}$$

Gegenüber dem globalen Alignment sind nur Initialisierung und der 0-Term anders.
Das Ergebnis (bester lokaler Alignment-Score) berechnet man als $\max_{i,j} S(i,j)$.

Wie kann man sich in der -Rekurrenz die Maximierung im 2. und 3. Fall sparen?

Affine Gapkosten

Wegen $g(k) = o + ek$ muss man nur 2 Fälle unterscheiden:
- Gap muss erst geöffnet werden (Kosten o noch nicht bezahlt)
- Gap ist schon geöffnet (Kosten o bereits bezahlt)

Wir definieren zwei Hilfsmatrizen für die zwei Arten von Gaps:
$X(i,j)$:= maximaler Score aller lokalen Alignments, in denen s_i und t_j die letzten Symbole sind und die mit einem horizontalen Gap $(-,t_j)$ enden;
$Y(i,j)$:= analog für vertikale Gaps.
Aus Eleganzgründen kann man noch definieren:
$Z(i,j)$:= max. Score aller lokalen Alignments, die mit dem alignierten Paar (s_i, t_j) enden.

Man beweist jetzt:
$S(i,j) = \max \{X(i,j),\ Y(i,j),\ Z(i,j)\}$
$Z(i,j) = S(i\text{-}1,j\text{-}1) + \text{sim}(s_i, t_j)$
$Y(i,j) = \max \{\ S(i', j) - g(i - i')\ :\ 0 <= i' < i\ \} = \max \{\ S(i\text{-}1, j) - (o+e),\ Y(i\text{-}1, j) - e\ \}$
$X(i,j) = \max \{\ S(i, j') - g(j - j')\ :\ 0 <= j' < j\ \} = \max \{\ S(i, j\text{-}1) - (o+e),\ X(i, j\text{-}1) - e\ \}$

Dank der Hilfsmatrizen X, Y benötigt man nur noch O(1) Zeit pro Zelle (i,j).

Konvexe / Konkave Gapkosten

Web-basierte Berechnung von Alignments

Am European Bioinformatics Institute (EBI) erhält man per Web-Formular Zugriff auf EMBOSS (European Molecular Biology Open Software Suite).

EMBOSS enthält verschiedene Programme, unter anderem solche zum Berechnen optimaler Alignments. [http://emboss.sourceforge.net/]

Das Alignment-Web-Formular am EBI ist abrufbar unter http://www.ebi.ac.uk/emboss/align/.

Berechnung optimaler Alignments

Erklärungen:

Method: needle(global) oder water(local) - Sollen die gesamten Sequenzen aligniert werden, oder soll das am besten passende Stück gesucht und nur das aligniert werden?

Gap open: Kosten(!) für eine einzelne Gap-Spalte im Alignment (der Score ist das Negative davon)

Gap extend: Kosten für weitere konsekutive Gap-Spalten (z.B. bezahlt man hier für drei aufeinanderfolgende Gap-Spalten nur 10+0.5+0.5 = 11 statt 3*10 = 30)

Man kann sowohl eine Sequenz in das Formular hineinkopieren, also auch eine auf dem Computer gespeicherte Datei hochladen.

Globale vs. Lokale Alignments

Method, needle(global) oder water(local):
Sollen die gesamten Sequenzen aligniert werden (global),
oder soll das am besten passende Stück gesucht und nur das aligniert werden (lokal)?

Woher kommen die Namen needle und water?

Von den Entdeckern der Verfahren (Algorithmen), mit denen optimale
- globale [Needleman SB, Wunsch CD (1970) J. Mol. Biol. 48; 443-453]
- lokale [Smith TF, Waterman MS (1981) J. Mol. Biol 147(1); 195-197]

Alignments effizient berechnet werden können

Wann global, wann lokal?

Ein globales Alignment macht dann und nur dann Sinn, wenn beide Sequenzen einander über ihre gesamte Länge hinweg ähnlich (und damit auch in etwa gleich lang) sind.

Ein lokales Alignment sucht in jeder Sequenz nach dem am besten zur anderen Sequenz passenden Teilstück und aligniert nur diese Teilstücke.
Dies macht vor allem dann Sinn, wenn die Sequenzen nicht über ihre ganze Länge homolog (evolutionär verwandt) sind, sondern nur in Teilen.

Das PWM-Modell

Wir haben bereits verschiedene Pattern-Modelle für die Textsuche kennengelernt:
- einfache Strings,
- generalized Strings,
- Menge von Strings,
- reguläre Ausdrücke,
- alle obigen mit Fehlertoleranz (z.B. Edit-Distanz <= k).

Insbesondere im Zusammenhang mit Signalen in DNA-Sequenzen
(z.B. Modellierung von Transkriptionsfaktorbindestellen)
ist ein weiteres Modell von Bedeutung:

Position Weight Matrices (PWMs), auch:
Position-Specific Score Matrices (PSSMs)
- Gaps nicht erlaubt
- spezifizieren Ähnlichkeit separat für jede Position und jedes Symbol:
  A x L – Matrix mit Ähnlichkeitswerten; L die Länge des Musters.
- Score einer Sequenz der Länge L ist Summe über die entsprechenden Ähnlichkeitswerte an den jeweiligen Positionen