
# Ein graphtheoretischer Ansatz für das *multiple sequence Alignment*-Problem

Bachelorarbeit

vorgelegt von:
**Joschka Strüber**

Matrikelnummer: 418702

Studiengang: B.Sc. Informatik

Thema gestellt von:
**Prof. Dr. Jan Vahrenhold**

Arbeit betreut durch:
**Prof. Dr. Jan Vahrenhold**
**Prof. Dr. Xiaoyi Jiang**

Münster, 29. März 2018

# Inhaltsverzeichnis

# 1 Einleitung und Motivation

## 1.1 Multiple Sequence Alignments

## 1.2 Einsatzgebiete

## 1.3 Komplexität

# 2 Dynamische Programmierung

## 2.1 Trivia

## 2.2 Das Paradigma

## 2.3 Der Algorithmus von Needleman und Wunsch

# 3 DIALIGN

In diesem Kapitel stelle ich zunächst das DIALIGN-Verfahren für multiples Sequenzalignment nach Morgenstern et all. vor Morgenstern *et al.* (1996). Dabei werde ich alle Anpassungen und Verbesserungen des Verfahrens vorstellen, die bis zur Version 2.2 umgesetzt wurden. Anders als der im letzten Kapitel vorgestellte Algorithmus von Needleman-Wunsch aligniert DIALIGN keine einzelnen Symbole, sondern gleich ganze Segmente der Eingabesequenzen. Das hat die Vorteile, dass man zum einen auf die Kosten zum Einfügen von Lücken verzichten kann und dadurch weitgehend von nutzerdefinierten Eingaben unabhängig wird, und weiterhin ist man so in der Lage sowohl global, als auch lokal verwandte Sequenzen einander auszurichten: Wenn man feststellt, dass in einem Bereich keine Segmente vorliegen, die einander ähnlich sind, dann verzichtet man darauf diese sich gegenseitig zuzuweisen und sie werden nicht Teil des Alignments.

DIALIGN kann genau wie Needleman-Wunsch im Sinne der jeweiligen Zielfunktion mathematisch optimale paarweise Alignments berechnen. Anders als bei letzterem, kann man aber auch mit Hilfe einer Heuristik effizient multiple Alignments berechnen, die aus drei oder mehr Sequenzen bestehen. Das grobe Vorgehen sieht dabei wie folgt aus:

**Algorithmus 1** DIALIGN

**Require:** Menge $S$ von Sequenzen mit $|S| = n$
1: Weise allen möglichen Fragmenten $D$ ein Gewicht $w^*(D)$ zu
2: Berechne mit dynamischer Programmierung alle möglichen $\binom{n}{2}$ paarweisen Alignments aus $S$
3: Sortiere alle Fragmente der paarweisen Alignments nach ihrem Gewicht als $D_{1,\ldots,n}$
4: Initialisisiere Ausgabe für Alignment $A := \emptyset$
5: **for** i=1, . . . , n **do**
6: **if** $D_i$ ist zu allen bisher gewählten Fragmenten *konsistent* **then**
7: $A \cup \{D_i\}$
8: **end if**
9: **end for**
10: **return** $A$

## 3.1 Theoretische Grundlagen

Um multiple Sequenzalignments genauer zu verstehen und die dazu nötigen Algorithmen analysieren zu können, brauchen wir zunächst einige Definitionen. Diese sind Morgenstern *et al.* (1996), Abdeddaïm & Morgenstern (2000) und Corel *et al.* (2010) entnommen. Dazu betrachten wir im Folgenden eine $n$-stellige Menge von Sequenzen $S$ über einem endlichen Alphabet. Dabei gibt $l(i)$ die Länge der $i$-ten Sequenz an.

**3.1.1 Definition** (Stelle und Stellenraum)
Eine *Stelle* ist ein Tupel $(i, p)$, bei dem $i$ die Sequenz und $p$ die Position eines Zeichens innerhalb dieser Sequenz angibt. Als *Stellenraum* bezeichnen wir die Menge aller Stellen über unseren Sequenzen $S$: $\mathcal{S} := \{(i, p) | 1 \leq i \leq n, 1 \leq p \leq l(i)\}$
Der Einfachheit identifizieren wir die *Stellen* der $i$-ten Sequenz als $\mathcal{S}_i$. Auf dem *Stellenraum* existiert eine Halbordnung '$\preceq$', wobei $(i, p) \preceq (i', p')$ genau dann gilt, falls $i = i'$ und $p \leq p'$.

Nachdem wir bis jetzt nur umgangssprachlich mit *Alignments* und *Konsistenz* zu tun hatten, möchte ich diese Begriffe nun formalisieren.

**3.1.2 Definition** (Alignment und Konsitenz)
Ein *Alignment* $\mathcal{A}$ ist eine Äquivalenzrelation auf der Menge $\mathcal{S}$, die ein bestimmtes *Konsistenzkriterium* erfüllt. Sei zunächst $\mathcal{R}$ eine beliebige binäre Relation auf $\mathcal{S}$. Wir können diese mit '$\preceq$' zu einer

## 3.2 Gewichtsfunktionen und Substitutionsmatrizen

## 3.3 Paarweise Alignments mit dynamischer Programmierung

### 3.3.1 Laufzeit

$O(n^2 * l^2)$ für die paarweisen Alignments, $O(n^3 * l * logl)$ für die Überlappgewichte

## 3.4 Konsistenz

### 3.4.1 Laufzeit

$O(n^3 * l + n^2 * l^2)$

## 3.5 Gieriges multiples Alignment

### 3.5.1 Laufzeit

$O(n^2 * l * log(n^2 * l))$

## 3.6 Gesamtkomplexität

$O(n^3 * l * logl + n^2 * l^2)$

## 3.7 Probleme

# 4 Ein Min-Cut-Ansatz für das Konsistenzproblem

-skizzierter Ablauf des Algorithmus

## 4.1 Flussnetzwerke

### 4.1.1 Einführung

### 4.1.2 Wichtige Algorithmen

### 4.1.3 Der *Max-Flow-Min-Cut-Satz*

## 4.2 Inzidenzgraphen und das Auflösen von Inkosistenzen mit Hilfe von Flussnetzwerken

### 4.2.1 Komplexität

$O(n^4 * l^{7/2})$

## 4.3 Sukzessorgraphen und der Algorithmus von Pitschi

$O(n^3 * l)$ für n-faches Topological Sort auf einem Knoten mit $O(n^2 * l)$-vielen Kanten im schlimmsten Fall.

## 4.4 Ankerpunkte

## 4.5 Gesamtkomplexität

$O(n^4 * l^{7/2})$

# 5 Programmierung

## 5.1 Speichereffiziente Umsetzung der dynamischen Programmierung

# 6 Validierung der Ergebnisse

## 6.1 Vorstellung BAliBase und (D)IRMBASE

## 6.2 Test auf BAliBase

## 6.3 Test auf DIRMBASE und IRMBASE

# 7 Fazit

## 7.1 Zusammenfassung

## 7.2 Future Works

- Statt paarweisen Alignments, Alignments von je drei Sequenzen berechnen
- Conditional Random Fields statt Gewichtsfunktionen
- Bessere Heuristik zum Löschen der Kanten aus dem Sukzessorgraph benutzen (gerade wenn Tests zeigen, dass oft sehr viele Kanten gelöscht werden) - das könnte vor allem dann vorkommen, wenn die Sequenzen große Überkreuzungen enthalten, weil man dann große Zyklen mit sehr hohen Kantengewichten hat. -> mögliches Paper?
- Statt DIALIGN 2 DIALIGN TX zwischen den Ankerpunkten nutzen
- Wir wollen aus dem Sukzessorgraph möglichst wenige Knoten löschen. Dafür sind Sites, die aber als einzige in ihren Knoten vorkommen, irrelevant.
    1. Jede Kante hinter Knoten hat Gewicht von Anzahl an Sites - 1 => bevorzugt Zuweisungen über möglichst viele Sequenzen hinweg. Es kann aber sein, dass dadurch viele kleine Zuweisungen aufgelöst werden
    2. Jede Kante hinter Knoten hat Gewicht 1, falls mehr als eine Sequenz beteiligt und Gewicht 0, falls nicht => bevorzugt viele kleine Alignments und minimiert Anzahl der Löschungen
- Teile des Algorithmus lassen sich gut parallelisieren:
    1. Die $(\frac{1}{2} * (n^2 - n))$-vielen paarweisen Alignments lassen sich parallelisiert berechnen.
    2. Die Überlappgewichte lassen sich gut parallelisiert berechnen, weil keins der Ergebnisse von denen der anderen abhängt. Paralleles Lesen der paarweisen Alignments ist kein Problem.
    3. Min-Cuts können innerhalb jeder Zusammenhangskomponente parallelisiert berechnet werden.
    4. Bei der alten Methode können auch die kürzesten Pfade durch den Sukzessionsgraphen parallelisiert werden, bei den beiden neuen Ansätzen nicht. Das ist aber nicht so schlimm, weil dieser Abschnitt die Laufzeit nicht dominiert.

# Literaturverzeichnis

Abdeddaïm, S. und Morgenstern, B. Speeding up the DIALIGN multiple alignment program by using the 'Greedy Alignment of BIOlogical Sequences LIBrary' (GABIOS-LIB. In *Computational Biology: First International Conference on Biology, Informatics, and Mathematics*, Seiten 1–11 (2000).

Corel, E., Pitschi, F. und Morgenstern, B. A *min-cut* algorithm for the consistency problem in multiple sequence alignment. *Bioinformatics*, Band 26:8:1015–1021 (2010).

Morgenstern, B., Dress, A. und Werner, T. Multiple DNA and protein sequence alignment based on segment-to-segment comparison. In *Proceedings of the National Academy of Sciences, USA*, Band 93, Seiten 12098–12103. Natl. Acad. Sci. USA (1996).

# Eidesstattliche Erklärung

Hiermit versichere ich, dass die vorliegende Arbeit über *„Titel“* selbstständig verfasst worden ist, dass keine anderen Quellen und Hilfsmittel als die angegebenen benutzt worden sind und dass die Stellen der Arbeit, die anderen Werken – auch elektronischen Medien – dem Wortlaut oder Sinn nach entnommen wurden, auf jeden Fall unter Angabe der Quelle als Entlehnung kenntlich gemacht worden sind.

---

Vorname Nachname, Münster, 29. März 2018

Ich erkläre mich mit einem Abgleich der Arbeit mit anderen Texten zwecks Auffindung von Übereinstimmungen sowie mit einer zu diesem Zweck vorzunehmenden Speicherung der Arbeit in eine Datenbank einverstanden.

---

Vorname Nachname, Münster, 29. März 2018