

Technische Universität München
Institut für Informatik
Lehrstuhl für Rechnerarchitektur und Rechnerorganisation

Diplomarbeit

Entwicklung von Heuristiken zum computerunterstützten Alignment ribosomaler RNS-Sequenzen unter Berücksichtigung ihrer Sekundärstruktur

Ralf Jost

Inhaltsverzeichnis

Überblick	1
1 Einleitung	3
1.1 Biologische Hintergrundinformationen	4
1.2 Das Alignment	7
1.3 Das ARB-Projekt	7
1.4 Definitionen und Begriffe	8
2 Primärstrukturbasiertes Alignment	11
2.1 Alignment mittels Distanzberechnung	13
2.1.1 Distanz zweier Sequenzen	13
2.1.2 Distanzberechnung mittels dynamischer Programmierung	15
2.2 Zweistufiges Alignment im ARB-Projekt	18
2.2.1 Das Prealignment	20
2.2.2 Das Mainalignment	24
3 Sekundärstrukturbasiertes Alignment	29
3.1 Problematik der Sekundärstruktursuche bei veränderter Primärstruktur	31
3.2 Suche nach möglichen Strukturelementen	33
3.2.1 Analyse der vorgegebenen Struktur	36
3.2.2 Beschränkung des Suchraums	38
3.2.3 Suche nach Strukturelementen	44

3.2.4	Zuordnung der gefundenen Strukturelemente	46
3.3	Suche nach dem optimalen Set der Strukturelemente	51
3.3.1	Konflikte und Qualität von Sets	52
3.3.2	Auflösung der Konflikte	54
4	Algorithmen	59
4.1	Suche nach möglichen Strukturelementen	60
4.1.1	Anforderungen an den Suchalgorithmus	61
4.1.2	Sammlung und Diskussion verschiedener Alternativen	63
4.1.3	Der angepaßte Positionsbaum als geeignetster Algorithmus	64
4.2	Suche nach dem optimalen Set der Strukturelemente	73
4.2.1	Anforderungen an den Optimierungsalgorithmus	74
4.2.2	Sammlung und Diskussion verschiedener Alternativen	74
4.2.3	Threshold Accepting als geeignetster Algorithmus	76
5	Kombiniertes Alignment	81
5.1	Gründe für eine Kombination	82
5.2	Mögliche Kombinationen	82
6	Zusammenfassung und Ausblick	89
	Literaturverzeichnis	91

Abbildungsverzeichnis

1.1	Aufbau eines rRNS-Moleküls	5
1.2	Sekundärstruktur der 16S rRNA von E. coli (Garrett und Grisham, 1995)	6
1.3	Ausschnitt aus einem Alignment	7
2.1	Wie kann die neue Sequenz durch Einfügen von Füllzeichen an der Konsensussequenz ausgerichtet werden?	12
2.2	Mögliche Ausrichtung einer Sequenz an einer Konsensussequenz	12
2.3	Eine Pfadmatrix mit drei vollständigen Verweisketten	15
2.4	Berechnung der Kosten eines Feldes der Pfadmatrix aus seinen Vorgängerfeldern	16
2.5	Die kostengünstigsten Verweisketten in einer vollständig berechneten Pfadmatrix	18
2.6	Das Alignment im ARB-Projekt	19
2.7	Bestimmung einer Sequenzfamilie anhand der Anzahl übereinstimmender Teilsequenzen einer Sequenz mit den Sequenzen eines Alignments (TS = Teilsequenz).	21
2.8	Bildung eines diskreten Konsensus	23
2.9	Bildung eines statistischen Konsensus	23
2.10	Pfadmatrix eines Prealignments	25
2.11	Berechnung der dynamischen Kosten eines Feldes der Pfadmatrix aus seinen Vorgängerfeldern	26

3.1	Wie kann die neue Sequenz durch Einfügen von Füllzeichen anhand der Sekundärstruktur an der Konsensussequenz ausgerichtet werden?	29
3.2	Mögliche Ausrichtung einer Sequenz an einer Konsensussequenz anhand der Sekundärstruktur	30
3.3	Mögliche Ausrichtung einer Sequenz an einer Konsensussequenz anhand der Sekundärstruktur ohne Übereinstimmungen in der Primärstruktur	30
3.4	Mögliche Bindungen derselben Teilsequenz, die zu einer Helix führen	32
3.5	Struktureller Ablauf des sekundärstrukturbasierten Alignments	34
3.6	Tatsächlicher Aufbau einer Helix (oben) und die aufgefaltete Ansicht (unten) mit Klammerstruktur zu ihrer Verdeutlichung	35
3.7	Ermittlung der realisierten Helices einer Sequenzfamilie anhand einer diskreten Konsensussequenz der Familie und der Helixstruktur des Alignments	37
3.8	Empirisch ermittelte Bindungsstärken der Basen relativ zur Bindungsstärke zwischen Adenin und Uracil	37
3.9	Problematik einer nicht vollständigen Helixstruktur eines Alignments und einer Konsensussequenz mit Füllzeichen	38
3.10	Beispiel für mögliche 'mismatches' in den Helices	40
3.11	Begrenzung des Suchraums durch primärstrukturbasiertes Prealignment der Sequenz und der damit verbundenen Unterteilung in aus- und unausgerichtete Bereiche	41
3.12	Anpassung der Bereichsgrenzen bei Überlappung mit Helices	42
3.13	Eine symmetrische Bereichsmatrix, die die Anzahl der Helices und deren Bereichsabhängigkeiten angibt	42
3.14	Obere rechte Dreiecksmatrix einer Bereichsmatrix mit den für die Suche nach den möglichen Helices relevanten Informationen	45
3.15	Bildung aller Teilsequenzen der Längen 7 - 9 eines Bereiches der Länge 10	46
3.16	Bereichsmatrix für die möglichen Helices ohne 'mismatches'. In Klammern ist jeweils die Anzahl der realisierten Helices angegeben	47
3.17	Zuordnung der realisierten Helices jedes Bereiches zu einer Teilmenge der möglichen Helices dieses Bereiches	48

3.18	Auswahl möglicher Helices aufgrund ihrer Ähnlichkeit bezüglich ihrer Längen und durchschnittlichen Bindungsstärken zu einer realisierten Helix	49
3.19	Zuordnung der ähnlichsten möglichen Helices zu den realisierten Helices nach Aufhebung der Bereiche	51
3.20	Ein Helix-Set ist die geordnete Menge von Helices aus den Mengen möglicher Helices	52
3.21	Konfliktfreie Zuordnung realisierter und möglicher Helices	53
3.22	Zuordnung realisierter und möglicher Helices mit Konflikt durch Überlappung	54
3.23	Zuordnung realisierter und möglicher Helices mit Konflikt in der Reihenfolge der Helixhälften (Überschneidung)	55
3.24	Konfliktfreie Pfade durch die Matrix der möglichen Helices	56
3.25	Beispiel für einen nicht auflösbaren Konflikt zwischen zwei Mengen möglicher Helices	57
3.26	Anzahl der Vergleichsoperationen für alle Pfade	58
4.1	Nach der Suche nach allen möglichen Helices wird die optimale Teilmenge, die Helixstruktur ausgewählt	60
4.2	Erzeugung der Suchmuster aus der Sequenz, wobei rückwärts vorgegangen wird, da sich Helices aus zwei antiparallelen, komplementären Teilen zusammensetzen	62
4.3	Zu jedem Muster kann es bis zu 2^m paarender Gegenstücke geben	62
4.4	Mögliche Formen eines 'mismatch' an einer Position im Muster	63
4.5	Ein Positionsbaum mit zusätzlicher Endmarkierung	65
4.6	Rekursiver Aufbau eines Positionsbaums durch Suche nach eindeutigen Teilsequenzen	66
4.7	Suche nach allen komplementären, gleichlangen Teilsequenzen	68
4.8	Suche nach allen komplementären Teilsequenzen, die durch das Präfix der Länge 2 des Musters bestimmt sind	69
4.9	Direkter Vergleich des Restmusters mit der Sequenz bei Mustern, die länger sind als ihre komplementären Pfade im Baum	70
4.10	Alle impliziten Veränderungen einer Mustersequenz bei einer Suche mit einem 'mismatch', wobei * eine beliebige Base repräsentiert	71

4.11 Suche nach Teilsequenzen mit einem 'mismatch', die zusätzlich in der Länge exakt mit den impliziten Mustern übereinstimmen und sich komplett im Baum befinden	72
4.12 Das erste Helix-Set aus den ähnlichsten Helices	77
4.13 Zufällige Auswahl eines Konflikts aus der Konfliktmatrix	78
4.14 Auswahl einer neuen Helix durch zufällige Erhöhung oder Erniedrigung des Index um den Wert 1	78
4.15 Nach Abbruch der Optimierung vorliegendes, beinahe optimales Helix-Set	79
5.1 Direkte Kombination des primär- mit dem sekundärstrukturbasierten Alignment	83
5.2 Direkte Kombination des sekundär- mit dem primärstrukturbasierten Alignment	84
5.3 Überprüfung und positionelle Anpassung des sekundärstrukturbasierten an das primärstrukturbasierte Alignment	85
5.4 Vergleich von primärstruktur- und sekundärstrukturbasiertem Alignment und Übernahme der signifikanteren Positionen	85
5.5 Weitere Unterteilung der unausgerichteten Bereiche in Regionen hoher Wahrscheinlichkeit für die einzelnen Helixhälften anhand des primärstrukturbasierten Alignment	86
5.6 Anpassung der Kosten für das primärstrukturbasierte Alignment in den Helixbereichen anhand des sekundärstrukturbasierten Alignment unter Berücksichtigung der Basenpaarungen	88

Überblick

In der Mikrobiologie können die Verwandtschaftsbeziehungen von Mikroorganismen anhand ihrer ribosomalen rNS bestimmt werden. Nach der Sequenzierung einer rNS wird die Abfolge ihrer Basen durch deren Anfangsbuchstaben repräsentiert. Die Unterschiede in solchen rNS-Sequenzen verschiedener Mikroorganismen geben Auskunft über deren Verwandtschaftsgrad. Um einen sinnvollen Vergleich der Sequenzen zu ermöglichen, werden sie untereinander in einem sogenannten Alignment angeordnet und so ausgerichtet, daß homologe, d.h. aus derselben 'Ur'-Base hervorgegangene Basen untereinander zu liegen kommen. Bei diesem Vorgang sind zwei Strukturmerkmale der Sequenzen ausschlaggebend:

- Zum einen ihre Primärstruktur, die sich ausschließlich aus der Abfolge ihrer Basen ergibt.
- Zum anderen ihre Sekundärstruktur, die den zweidimensional darstellbaren Teil der räumlichen Struktur eines rNS-Moleküls erfaßt.

Wegen der großen Länge und der hohen Anzahl der Sequenzen ist es wünschenswert, ein solches Alignment maschinell erzeugen zu lassen. Die vorliegende Arbeit beschäftigt sich daher mit Verfahren zum Einfügen und Ausrichten einer Sequenz in ein bestehendes Alignment.

Zunächst wird ein deduktives, ebenfalls an diesem Lehrstuhl entwickeltes Verfahren vorgestellt, das sich vollständig auf die Primärstruktur stützt. Der hier zugrundeliegende Algorithmus basiert auf der Distanzberechnung zweier Sequenzen nach [SNee].

Anschließend wird ein heuristisches Verfahren entwickelt, das nur die Informationen der Sekundärstruktur berücksichtigt. Es basiert auf einer einschränkenden Suche nach möglichen Strukturelementen und der anschließenden Auswahl der möglichen Sekundärstrukturelemente, die die bestÜbereinstimmung mit dem Sekundärstrukturmodell des Alignments liefern. Die grundlegenden Algorithmen - ein speziell angepaßter Positionsbaum für die Suche nach den Strukturelementen und ein Mutations-Selektions-Algorithmus für die Auswahl der geeignetsten Strukturelemente - werden gesondert beschrieben.

Abschließend werden einige Methoden zur Kombination von primär- und sekundärstrukturbasierten Verfahren vorgeschlagen und diskutiert, die es ermöglichen sollen ein umfassenderes und beide Strukturmerkmale vollständig berücksichtigendes Verfahren zu entwickeln.

Kapitel 1

Einleitung

In der Biologie sind die hierarchischen Verwandtschaftsbeziehungen zwischen Organismen entscheidend für deren Klassifizierung. Bei Spezies höherer Ordnung können diese Beziehungen anhand von phänotypischen Merkmalen wie Anatomie oder Morphologie und der Embryonalentwicklung ermittelt werden.

In der Mikrobiologie dagegen werden überwiegend einzellige Lebewesen wie Bakterien untersucht. Auch hier können phänotypische Merkmale zur Klassifizierung herangezogen werden, aufgrund ihrer geringen Zahl erlauben sie aber keine systematische Beschreibung der Verwandtschaftsbeziehungen. Aus diesem Grund wurde nach anderen konservierten Merkmalen gesucht, die also in allen Mikroorganismen vorkommen, für deren Überleben notwendig sind und stets dieselbe Aufgabe erfüllen. Die ribosomale rRNA erfüllt diese Kriterien. Dabei handelt es sich um ein Makromolekül, das entscheidend bei der Proteinbiosynthese, der Transkription der rDNA in rRNA-Sequenzen mitwirkt. Es liegt in der Zelle als Nukleotidkette (auch -sequenz) vor. Die Unterschiede in diesen Sequenzen können als Maß für die Verwandtschaft von Mikroorganismen herangezogen werden.

Da die Methoden zur Sequenzierung von Nukleinsäuren immer weiter verbessert wurden, sind heute mehrere tausend rRNA Sequenzen bekannt. Die Sequenzen selbst setzen sich wiederum aus bis zu mehreren tausend Basen zusammen. Um solche Datenmengen zu bewältigen, wird immer stärker auf maschinelle Methoden zurückgegriffen. Auch die Analyse der Verwandtschaftsbeziehungen wird rechnergestützt durchgeführt. Um einen Vergleich der Sequenzen zu ermöglichen, werden sie in einem sogenannten Alignment so untereinander angeordnet und ausgerichtet, daß jeweils die Bereiche, die sich aus einer gemeinsamen Ur-Bereich entwickelt haben, untereinander zu liegen kommen. Dabei müssen gewisse Strukturmerkmale, die Primär- und Sekundärstruktur genannt werden, berücksichtigt werden. Es wird vermutet, daß das Einfügen und Ausrichten einer Sequenz in ein Alignment unter Berücksichtigung beider Strukturmerkmale ein NP-vollständiges Problem darstellt, da starken Abhängigkeiten zwischen diesen bestehen. Also ein Problem, das

selbst mit nichtdeterministischen Algorithmen nicht in polynomialer Zeit gelöst werden kann. Um die Komplexität etwas zu reduzieren, liegt es nahe, jeweils nur eines der Merkmale zu berücksichtigen. Nachdem in [BR01] eine Methode für ein primärstrukturbasiertes Alignment vorgestellt wurde, soll nun in der vorliegenden Arbeit eine Methode für ein sekundärstrukturbasiertes Alignment entwickelt und ein Ausblick auf Möglichkeiten zur Kombination beider Verfahren gegeben werden.

1.1 Biologische Hintergrundinformationen

In diesem und den folgenden Abschnitten sollen die Informationen geliefert werden, die zum Verständnis dieser Arbeit notwendig sind. Abschließend werden die wichtigsten Begriffe und Definitionen noch einmal mit kurzen Erklärungen versehen aufgeführt.

Wie in der Einleitung bereits erwähnt, basiert das Klassifizierungssystem der Biologie auf den hierarchischen Verwandtschaftsbeziehungen, der Phylogenie von Organismen. Anhand von phänotypischen Merkmalen, wie Anatomie oder Morphologie und auch Embryonalentwicklung höher entwickelter Spezies und deren Vergleich untereinander oder mit Fossilienfunden können phylogenetische Bäume oder Stammbäume erstellt werden, die die Evolution der Organismen widerspiegeln. Für die phylogenetischen Beziehungen von Einzellern in der Mikrobiologie sind diese Merkmale aber unzulänglich. Hier sind mikrobiologische Merkmale gefragt, die ubiquitär, essentiell und funktional konserviert sind, d.h. in allen Mikroorganismen vorkommen, für deren Überleben unbedingt notwendig sind und in allen Mikroorganismen stets dieselbe Funktion erfüllen. Es hat sich gezeigt, daß zwei Nucleinsäuren diese Merkmale erfüllen: die DNS und die RNS, insbesondere die ribosomale RNS (rRNS). Beide Makromoleküle liegen in der Zelle als Nucleotidketten vor. Ihr Aufbau kann mit biochemischen Methoden der Sequenzierung erfaßt werden. Allerdings eignet sich die DNS aufgrund ihrer enormen Länge von über einer Million Nucleotiden nicht für eine vergleichende Analyse. Die rRNS besteht dagegen nur aus bis zu einigen tausend Nucleotiden, weswegen sie für die phylogenetische Analyse von Mikroorganismen herangezogen wird. Zusammen mit zwei anderen Nucleinsäuren, der mRNA und der tRNA ist die rRNS entscheidend an der Proteinbiosynthese, der Translation der DNS in Proteinsequenzen, beteiligt. Sie kommt in allen Lebewesen vor und hat immer dieselbe Aufgabe. Aufgrund dieser funktionalen Konserviertheit hat sich ihre Struktur kaum geändert, wodurch sie sich für vergleichende Analysen zur Aufdeckung von Verwandtschaftsbeziehungen besonders eignet.

Der Aufbau von rRNS-Molekülen folgt einem sehr einfachen Prinzip. Es handelt sich um eine alternierende Abfolge von Ribose (einem fünffach Zuckerein-Molekül) und Phosphorsäure, die die Ribose-(Mononucleotid-)Einheiten zu einer Kette ver-

bindet. An der Ribose bindet jeweils eine der vier Basen Adenin, Cytosin, Guanin und Uracil, die im folgenden mit ihren Anfangsbuchstaben abgekürzt werden sollen. Die Phosphorsäure verknüpft die Ribose unterschiedlich, so daß zwei verschiedene Enden der Nucleotidkette entstehen, die mit 5' und 3' bezeichnet werden und als Anfang und Ende identifiziert werden können, wie das in Abbildung 1.1 dargestellt ist.

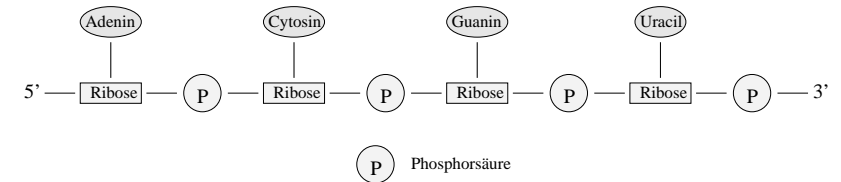


Abbildung 1.1: Aufbau eines rRNS-Moleküls

Entscheidend für die Funktion eines rRNS-Moleküls sind aber nur die Basen, bzw. deren Abfolge, die als Primärstruktur bezeichnet wird. Somit läßt sich ein rRNS-Molekül als Sequenz seiner Basen z.B. folgendermaßen darstellen:

5' ... ACGUCAGCAGUCGACGUCAGUGGCCAAUCG ... 3'

Die Primärstruktur kann sich durch Fehler im Reproduktionsprozeß verändern. Diese Mutationen treten zufällig an beliebigen Positionen in der Sequenz auf. Dadurch ändern sich die chemischen und physikalischen Eigenschaften des rRNS-Moleküls und seine Funktionsweise kann so stark beeinträchtigt werden, daß der Mikroorganismus zugrunde geht.

Ein weiteres Strukturmerkmal der rRNS, die in der Zelle einsträngig vorliegt, ist ihre Sekundärstruktur. Da die chemische Struktur der Basen paarweise komplementär ist, können diese (Wasserstoffbrücken-)Bindungen eingehen. Dabei bindet Adenin mit Uracil und Cytosin mit Guanin. Abgesehen von diesen typischen Bindungen existieren noch Bindungen zwischen Adenin und Guanin und zwischen Guanin und Uracil, die allerdings deutlich seltener auftreten. Sind nun zwei Teilbereiche der rRNS-Sequenz antiparallel komplementär, so können sie sich miteinander verbinden, wobei eine Schleife entsteht, der sog. 'Loop'-Bereich, der durch die bindenden Teilbereiche, der sog. Helix, abgeschlossen ist. In einer rRNS gibt es viele solcher Bereiche, so daß sich das Molekül zu seiner charakteristischen Form zusammenlegt. Ein Beispiel einer solchen Sekundärstruktur wird in Abbildung 1.2 gegeben.

Es hat sich gezeigt, daß die Sekundärstruktur stark konservativ ist. Veränderungen treten überwiegend in den Helices auf, wobei sie sich in ihrer Länge und/oder

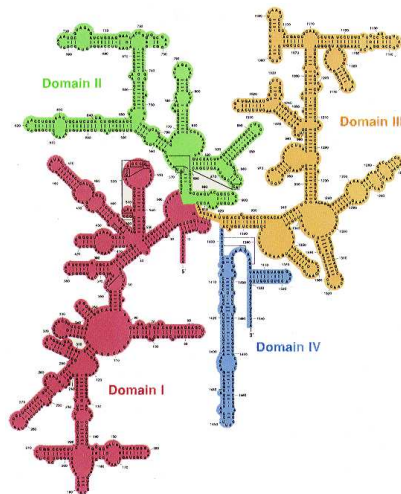


Abbildung 1.2: Sekundärstruktur der 16S rRNA von *E. coli* (Garrett und Grisham, 1995)

Primärstruktur ändern, die Gesamtstruktur aber weitgehend unberührt bleibt.

Die Basen, die nicht an der Bildung der Sekundärstruktur beteiligt sind, können noch weitere Bindungen eingehen, die das Molekül noch weiter zusammenfalten und ihm seine endgültige, dreidimensionale Form geben, der Tertiärstruktur. Für die Funktion des Moleküls in der Zelle ist meistens seine dreidimensionale Form ausschlaggebend.

1.2 Das Alignment

Durch die Art der Mutationen, die in einer rRNS auftreten können, kommt es zu Veränderungen einzelner Positionen oder auch ganzer Abschnitte, aber nicht zu Veränderung in der Reihenfolge der einzelnen Abschnitte. Diesen Sachverhalt kann man ausnutzen, um ein Alignment zu erstellen. Es handelt sich dabei um eine vertikale Anordnung der rRNS-Sequenzen, wobei sie horizontal so ausgerichtet werden, daß die Basen, die aus einer gemeinsamen 'Ur'-Base hervorgegangen sind, untereinander zu liegen kommen. Man bezeichnet solche Basen als homolog oder im Falle ganzer Abschnitte als homologe Bereiche. Abbildung 1.3 soll dies verdeutlichen.

```

G A | G C U G U | - - - - - G C A A - - - - - G C G G U | G A
G A | G G G U | - - - - - G A A C - - - - - G C U C C | G A
G A | U C U G U | C A - U C - G C A A - G A - G A A | G C A G U | G A
G A | G A U G U | U A G C C - G U A A - G G | C G A A | G C A U U | G A

```

Paarende Bereiche

Abbildung 1.3: Ausschnitt aus einem Alignment

Im Laufe der Evolution kann es vorgekommen sein, daß manche Basen ganz weggefallen sind oder andere eingefügt wurden, so daß es notwendig wird, an den Stellen, an denen keine 'Ur'-Base mehr existiert, Füllzeichen ('-') in das Alignment einzufügen. Für Basen, die nicht zweifelsfrei bestimmt werden konnten, fügt man ein '.' ein. Die Füllzeichen können auf zwei Arten interpretiert werden. Zum einen kann es in der Sequenz an dieser Position zur Auslöschung einer Base gekommen sein, zum anderen in einer anderen Sequenz zu einer Einfügung.

1.3 Das ARB-Projekt

Durch die fortschreitende Entwicklung neuer Sequenzierungsmethoden steigt die Anzahl der bekannten Nukleinsäuren, also auch der rRNS, sehr schnell an. Für

die Verwaltung und Auswertung dieser Datenmengen werden damit maschinelle Verfahren unumgänglich. In einer interdisziplinären Arbeitsgruppe zwischen dem Lehrstuhl für Mikrobiologie Prof. Dr. K. H. Schleifer und dem Lehrstuhl für Rechnertechnik und Rechnerorganisation Prof. Dr. A. Bode entstand deshalb im Rahmen von Diplomarbeiten und Fortgeschrittenen Praktika ein Programmpaket zur Verwaltung und Analyse von rRNS-Sequenzen. Ursprünglich wurde es zur Erstellung phylogenetischer Bäume entwickelt, weswegen es in Anlehnung an das lateinische Wort arbor, -oris für Baum mit ARB betitelt wurde. Inzwischen umfaßt es mehrere Applikationen, die sich für

- die Berechnung und Darstellung phylogenetischer Bäume,
- die Bearbeitung von Sequenzen und deren Sekundärstrukturdarstellung,
- die Berechnung von Distanzmatrizen,
- den Entwurf von Hybridisierungs sonden und
- die Sekundärstrukturanalyse

eignen. Mittels einer allen Applikationen zugrundeliegenden, für den Mehrbenutzerbetrieb geeigneten Datenbank für die Sequenzdaten, wird eine Kommunikation zwischen den einzelnen Applikationen ermöglicht und die Datenkonsistenz gewährleistet. Darüber hinaus besitzen alle Applikationen aufgrund einer eigens dafür entwickelten Bibliothek eine einheitliche Benutzeroberfläche. Die vorliegende Arbeit entstand ebenfalls im Rahmen dieses Projekts.

1.4 Definitionen und Begriffe

In diesem Abschnitt werden zunächst die mathematischen Definitionen, auf die später zurückgegriffen wird, dargelegt und anschließend noch einmal die wichtigsten Begriffe kurz erläutert.

Definitionen:

- $\mathcal{B} = \{A, C, G, U\}$
Die Menge der Zeichen, die die Basen Adenin, Cytosin, Guanin und Uracil repräsentieren.
- $\hat{\mathcal{B}} = \mathcal{B} \cup \{-, n, \cdot\}$
Die Menge der relevanten Zeichen, die in einem Alignment vorkommen können, wobei '-' für eine Insertion, 'n' für eine beliebige Base aus \mathcal{B} und '.' für eine unsequenzierte Base (oder Bereich) steht.

- $s = \langle b_1, b_2, \dots, b_n \rangle$ mit $b_i \in \mathcal{B}$ und $i = 1, 2, \dots, n$, wobei $|s| = n$
Eine Basensequenz mit Elementen aus \mathcal{B} und der Länge n .
- $S = \{b_1, b_2, \dots, b_n\}$ mit $b_i \in \mathcal{B}$ und $i = 1, 2, \dots, n$
Eine geordnete Menge der in einer Sequenz s auftretenden Basen.
- Die Ordnungsrelation $<$ auf S mit $b_1 < b_2 < \dots < b_n$.
- $S \supseteq T_{ij} = \{b_i, b_{i+1}, \dots, b_{i+(j-1)}\}$ mit $1 \leq i \leq n - j + 1$
Eine geordnete Teilmenge der Länge j aus S , beginnend an der Position i in S .
- $H = \{(T_{pj}, T_{qj}) \mid p \leq q - j \wedge b_{p+l} \text{ komplementär zu } b_{(q+j-1)-l}\}$ mit $0 \leq l < j$
Eine Helix der Länge j bestehend aus zwei antiparallelen, komplementären Teilsequenzen einer Sequenz S .

Begriffe:

- Alignment
Horizontale Anordnung und vertikale Ausrichtung von (hier) rRNS-Sequenzen mittels Fullzeichen, so daß homologe Basen untereinander zuliegen kommen.
- Helix
Struktur einer (hier) rRNS-Sequenz, die sich durch das Verbinden zweier antiparalleler, komplementärer Sequenzabschnitte bildet.
- Heuristik
Bezeichnung für ein Lösungsverfahren, das nicht auf wissenschaftlichen Erkenntnissen, sondern auf Hypothesen, Analogien oder Erfahrungen aufbaut.
- Homologe Basen
Basen, die aus derselben 'Ur'-Base hervorgegangen sind.
- Komplementäre Basen
Basen, die miteinander Wasserstoffbrückenbindungen eingehen können.
- mismatch
Nicht paarende oder fehlende Basen in Helices.
- Primärstruktur
Die Abfolge der Basen (hier) in einer rRNS-Sequenz.

- Phylogenie
Die Stammesgeschichte der Lebewesen.
- Sekundärstruktur
Struktur eines (hier) rRNS-Moleküls, die sich durch das Zusammenfallen der Sequenz aufgrund der Verbindung antiparalleler, komplementärer Sequenzabschnitte ergibt.

Kapitel 2

Primärstrukturbasiertes Alignment

Beim Einfügen einer neuen Sequenz in ein bestehendes Alignment sind sowohl die Primär- als auch die Sekundärstruktur zu beachten. Da die Sekundärstruktur eines rRNS Moleküls stark konservativ ist, stimmen die Sekundärstrukturen aller im Alignment enthaltenen Sequenzen weitgehend überein. Die Primärstrukturen dagegen können stark voneinander abweichen. Allerdings besitzt auch die Primärstruktur stark konservative Bereiche. Diese Eigenschaft wird in einem Alignment dazu genutzt homologe Bereiche zu identifizieren und somit einfach untereinander anzuordnen. Nah verwandte Sequenzen folgen dabei dicht aufeinander, da sie sich in den konservativen Bereichen ihrer Primärstruktur kaum unterscheiden. Man spricht dann von einer Sequenzfamilie. Es erscheint sinnvoll, die neu einzufügende Sequenz gegen die Sequenzen ihrer Familie auszurichten, da hier die größten Übereinstimmungen in der Primär- und Sekundärstruktur zu finden sind. Da eine Ausrichtung gegen mehrere Sequenzen algorithmisch aufwendig ist, werden die Sequenzen einer Familie durch einen Repräsentanten ersetzt. Man spricht dann von der Konsensussequenz einer Familie. Am einfachsten ist es sicherlich, eine diskrete Konsensussequenz aufzubauen, in der an jeder Position nur die häufigste Base der Familiensequenzen vertreten ist. Da dabei allerdings Informationen vernachlässigt werden, weicht man auf andere Repräsentationen aus, z.B. auf einen statistischen Konsensus. Die Erstellung eines solchen wird in Kapitel 2.2.1 beschrieben.

Nachdem die Konsensussequenz gefunden ist, stellt sich das eigentliche Problem: Wie kann man durch Einfügen und Entfernen von Füllzeichen die einzufügende Sequenz so verändern, daß sie sowohl in ihrer Primär- als auch ihrer Sekundärstruktur mit der Konsensussequenz möglichst gut übereinstimmt? Am folgenden, in Abbildung 2.1 dargestellten Beispiel soll die Problematik etwas verdeutlicht werden. Gleiche Buchstaben sollen Basenbindungen symbolisieren sollen.

Da sich eine gute Ausrichtung fast ohne Beachtung der Sekundärstruktur erreichen

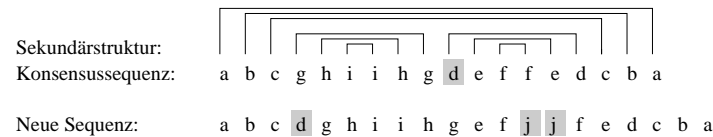


Abbildung 2.1: Wie kann die neue Sequenz durch Einfügen von Füllzeichen an der Konsensussequenz ausgerichtet werden?

läßt, ist dieses Beispiel sehr einfach. Man kann allerdings bereits erahnen, welche Abhängigkeiten zwischen der Primär- und der Sekundärstruktur bestehen und zu welchen Problemen diese führen können, vor allem in den variablen Bereichen der Sequenzen, in denen die Primärstrukturen unter Umständen stark differieren. Abbildung 2.2 zeigt eine mögliche Ausrichtung des vorangegangenen Beispiels.

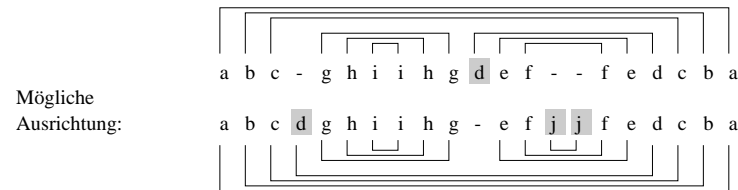


Abbildung 2.2: Mögliche Ausrichtung einer Sequenz an einer Konsensussequenz

Wie man sieht ist das Ausrichten einer Sequenz eine Abbildung $align$ von einer Sequenz s ohne Füllzeichen auf eine Sequenz s' mit Füllzeichen, bei der die Abweichungen ($cost$) in der Primär- und Sekundärstruktur von einer Konsensussequenz möglichst gering gehalten werden sollen. Formal handelt es sich also um eine Abbildung

$$align : \mathcal{B}^* \rightarrow \hat{\mathcal{B}}^*, s \mapsto align(s) = s', \quad (2.1)$$

wobei eine gewichtete Kostenfunktion

$$cost(align) = w_{prim} cost_{prim}(align) + w_{sec} cost_{sec}(align),$$

mit $w_{prim}, w_{sec} \in [0, 1]$ und $w_{prim} + w_{sec} = 1$ (2.2)

minimiert werden soll.

Berücksichtigt man nur die Primärstruktur, vereinfacht sich das Problem, da man die Abhängigkeiten zwischen der Primär- und der Sekundärstruktur außer Acht lassen kann. Formal gewichtet man dabei $cost_{sec}$ einfach mit 0. Allerdings verzichtet

man dann auch auf die Informationen, die in der Sekundärstruktur enthalten sind. Besonders geeignet scheint ein rein primärstrukturbasiertes Alignment zum Auffinden der stark konservativen Bereiche einer Sequenz. Diese Bereiche machen einen Großteil einer Sequenz aus, was ein solches Verfahren rechtfertigt. In [BRei] werden Verfahren vorgestellt, die ein primärstrukturbasiertes Alignment ermöglichen. Im folgenden werde ich diese Verfahren in ihren Grundzügen darlegen, da sie die Grundlage für die vorliegende Arbeit bilden. Es hat sich gezeigt, daß die relativ wenigen variablen Bereiche diesen Verfahren starke Probleme bereiten. Der Versuch, die Informationen, die die Sekundärstruktur liefert, in diesen Ansatz einfließen zu lassen, hat zu der Einsicht geführt, daß dies nur sehr eingeschränkt möglich ist.

2.1 Alignment mittels Distanzberechnung

Das Problem beim primärstrukturbasierten Alignment reduziert sich darauf, eine neue Sequenz an einer anderen, in diesem Fall der Konsensussequenz, so auszurichten, daß die Abweichungen bezüglich der Primärstruktur möglichst gering werden. Man benötigt also ein Maß für die Ähnlichkeit zweier Sequenzen, um die Güte des Alignments bewerten zu können. In [BRei] wird eine Algorithmus vorgeschlagen, der dies ermöglicht. Er wurde [DSan] entnommen, stammt aber ursprünglich von Needleman und Wunsch [SNee], und so adaptiert, daß er den Anforderungen, die dieses Anwendungsgebiet aus der Mikrobiologie stellt, Rechnung trägt. Mit Hilfe dieses Algorithmus kann unter Verwendung gewisser Operationen eine Sequenz in eine andere überführt werden. Jede dieser Operationen erzeugt spezifische Kosten, so daß die Summe aller Kosten einer Folge von Operationen, die eine Sequenz in eine andere überführen, als Maß der Distanz zwischen diesen beiden Sequenzen verwendet werden kann. Da verschiedene Operationen denkbar sind, sind auch unterschiedliche Gesamtkosten möglich. Sinnvollerweise wird deshalb die Reihenfolge mit den niedrigsten Gesamtkosten als Maß für die Distanz zweier Sequenzen verwendet.

2.1.1 Distanz zweier Sequenzen

Was unter der Distanz zweier Sequenzen zu verstehen ist, läßt sich nicht ohne weiteres in einer prägnanten Definition fassen, da sie ist mit dem Algorithmus zu ihrer Berechnung verwoben ist. Um die Distanz zwischen zwei Sequenzen a und b über einem Alphabet A mit $|a| = m$ und $|b| = n$ zu berechnen, benötigt man bewertbare Operationen, mit denen die eine Sequenz in die andere überführt werden kann. Diese Operationen sollten dem tatsächlichen Mutationsverhalten von rRNS-Sequenzen nachempfunden sein. Es hat sich gezeigt, daß es drei relevante Arten von Mutationen gibt, die sich sehr gut formal fassen lassen:

- DEL(x) Löschung einer Base,
- INS(x) Einfügung einer Base,
- SUB(x,y) mit $x, y \in A$ Ersetzung einer Base durch eine andere.

Jede dieser Operationen verursacht spezifische Kosten und kann somit bewertet werden. Also können auch Folgen solcher Operationen, vor allem die, die a in b überführen, mit der Summe ihrer Einzelkosten bewertet werden. Diese Kosten können z.B. durch statistische Analysen der Sequenzen einer Familie ermittelt werden. Für eine genaue Beschreibung solcher Verfahren sei auf [BRei] verwiesen. Anhand eines Beispiels soll die Überführung einer Sequenz in eine andere noch einmal verdeutlicht werden: Sei die Sequenz $\langle huhu \rangle$ in die Sequenz $\langle hallo \rangle$ zu überführen, kann dies folgendermaßen geschehen:

- SUB(h,h), SUB(u,a), SUB(h,l), SUB(u,l), INS(o)
also h u h u
 h a l l o
oder
- SUB(h,h), INS(a), INS(l), SUB(u,l), SUB(h,o), DEL(u)
also h u h u
 h a l l o
oder
- SUB(h,h), DEL(u), DEL(h), INS(a), INS(l), DEL(u), INS(l), INS(o)
also h u h u
 h a l l o

Offensichtlich gibt es schon bei diesem einfachen Beispiel sehr viele Operationenfolgen, die die Überführung bewerkstelligen. Da die Kosten der Einzeloperationen unterschiedlich gewichtet sind, ergeben sich auch unterschiedliche Gesamtkosten. Als Maß für die Distanz zwischen zwei Sequenzen nimmt man deshalb die Gesamtkosten der Operationenfolge, die die Überführung mit den geringsten Gesamtkosten realisiert. Um diese zu finden, liegt es nahe, alle diese Operationenfolgen zu erzeugen und zu bewerten und sich schließlich diejenige mit den niedrigsten Gesamtkosten herauszusuchen. Dafür weicht man auf eine günstigere Repräsentation der Operationenfolgen aus: eine Pfadmatrix. In ihr werden die einzelnen Operationen durch Pfeile mit einer bestimmten Orientierung repräsentiert. Die Pfadmatrix wird von den beiden Sequenzen aufgespannt. Ein Beispiel einer Pfadmatrix ist in Abbildung 2.3 dargestellt, wobei hier noch nicht alle vollständigen Verweisketten vorliegen.

Eine vollständige Verweiskette ist eine, die a vollständig in b überführt, das heißt, im Element $M(1, 1)$ der Pfadmatrix M beginnt und im Feld $M(m, n)$ endet. Alle

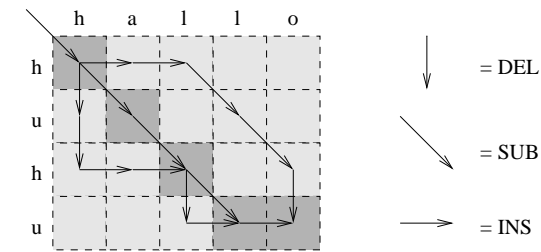


Abbildung 2.3: Eine Pfadmatrix mit drei vollständigen Verweisketten

möglichen, vollständigen Verweisketten liegen dann vor, wenn in jedes Matrixelement drei Pfeilspitzen zeigen. Eine Abschätzung der Anzahl aller vollständigen Verweisketten ergibt eine Komplexität von mindestens $O(3^{\min(m,n)})$, weshalb eine Optimierung des Algorithmus dringend geboten ist. Da das Verfahren vorwärts gerichtet ist, und die Datenabhängigkeiten gering sind, bietet sich eine Optimierung mittels dynamischer Programmierung an.

2.1.2 Distanzberechnung mittels dynamischer Programmierung

Es stellt sich das Problem, einen effizienten Algorithmus zu finden, der die Distanzberechnung zweier Sequenzen, wie in 2.1.1 beschrieben, realisiert. Mit einer Pfadmatrix über den beiden Sequenzen können alle vollständigen Verweisketten, also Lösungen für die Überführung der einen Sequenz in die andere, erzeugt werden. Jeder Verweiskette wird dabei die Summe ihrer Einzeloperationen zugeordnet, so daß anschließend die Verweiskette mit den niedrigsten Gesamtkosten gefunden werden kann. Jedes Feld der Pfadmatrix stellt dabei den Überführungszustand der Sequenz bis zu dem zugehörigen Zeichen dar, also des Präfixes der zu überführenden Sequenz, das durch dieses Zeichen bestimmt ist. Zudem hängt jedes Matrixelement nur von maximal drei anderen Elementen, dem links neben ihm, dem über ihm und dem schräg links über ihm, ab. Wenn die Kosten der Überführung bis zu diesen drei Elementen bekannt sind, können die minimalen Kosten für jedes Matrixelement direkt aus ihnen berechnet werden. Graphisch läßt sich dieser Sachverhalt wie in Abbildung 2.4 darstellen.

Es bietet sich also an, die Matrix zeilen- oder spaltenweise von links oben nach rechts unten abzarbeiten und dabei für jedes Feld die minimalen Kosten, berechnet aus diesen drei Nachbarfeldern zuzüglich der Kosten für die entsprechende Operation, die von dort in das aktuelle Feld führt, zu notieren. Da aber nicht nur die minimalen Kosten für die Überführung gefragt sind, sondern auch die Folge

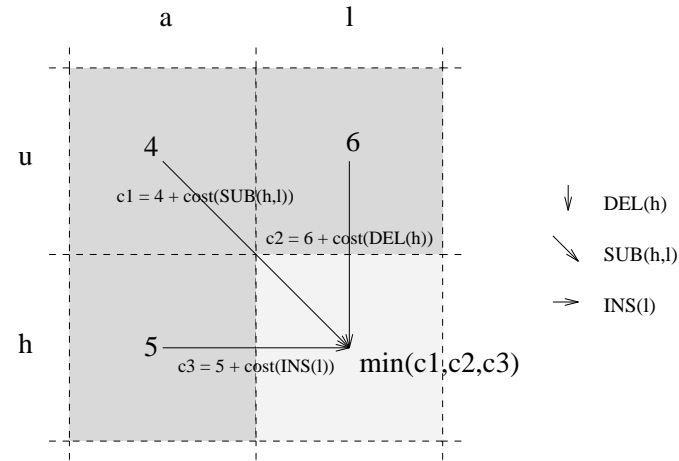


Abbildung 2.4: Berechnung der Kosten eines Feldes der Pfadmatrix aus seinen Vorgängerfeldern

der Operationen, die diese erzeugt haben, ist es sinnvoll, zusätzlich in jedem Feld zu vermerken, aus welchem Vorgängerfeld seine Kosten entstanden sind. Auf diese Weise gelingt es, das Problem aus der Komplexität $O(3^{\min(m,n)})$ 2.1.1 auf ein Problem der Komplexität $O(mn)$ zu reduzieren.

Sind also zwei Sequenzen $a, b \in \mathcal{B}^*$ gegeben mit $a = \langle a_1, \dots, a_m \rangle$, $|a| = m$ und $b = \langle b_1, \dots, b_n \rangle$, $|b| = n$, dann lassen sich die inneren Elemente $M_{i,j}$ der Pfadmatrix $M^{m,n}$ mit $i, j > 1 \wedge i \leq m \wedge j \leq n$ wie folgt berechnen:

$$M_{i,j} = \min \begin{cases} M_{i-1,j} + \text{cost}(DEL(a_i)) \\ M_{i,j-1} + \text{cost}(INS(b_j)) \\ M_{i-1,j-1} + \text{cost}(SUB(a_i, b_j)) \end{cases} \quad (2.3)$$

Um mit der Berechnung beginnen zu können, müssen noch die Felder der ersten Zeile und der ersten Spalte besetzt werden. Eine gesonderte Stellung nimmt vor allem das Feld $M_{1,1}$ ein. Es gibt an, wie a_1 in b_1 übergeführt werden kann. Möglich sind dabei die Substitution von a_1 durch b_1 , die Deletion von a_1 mit anschließender Insertion von b_1 oder die Insertion von b_1 mit der anschließenden Deletion von a_1 .
Formal also:

$$M_{1,1} = \min \begin{cases} \text{cost}(DEL(a_1)) + \text{cost}(INS(b_1)) \\ \text{cost}(SUB(a_1, b_1)) \\ \text{cost}(INS(b_1)) + \text{cost}(DEL(a_1)) \end{cases} \quad (2.4)$$

Zur Berechnung der Felder $M_{i,1}$ der ersten Spalte mit $i > 1 \wedge i \leq m$, die angeben, wie b_1 als erstes Zeichen an jeder Stelle in a positioniert werden kann, stehen ebenfalls drei Möglichkeiten zur Verfügung:

$$M_{i,1} = \min \begin{cases} M_{i-1,1} + \text{cost}(DEL(a_i)) \\ \sum_{k=1}^{i-1} \text{cost}(DEL(a_k)) + \text{cost}(SUB(a_i, b_1)) \\ \sum_{k=1}^i \text{cost}(DEL(a_k)) + \text{cost}(INS(b_1)) \end{cases} \quad (2.5)$$

Ähnlich lassen sich die Felder $M_{1,j}$ der ersten Zeile mit $j > 1 \wedge j \leq n$ berechnen, die angeben, wie jedes Präfix von b an der ersten Stelle von a positioniert werden kann:

$$M_{1,j} = \min \begin{cases} M_{1,j-1} + \text{cost}(INS(b_j)) \\ \sum_{k=1}^{j-1} \text{cost}(INS(b_k)) + \text{cost}(SUB(a_1, b_j)) \\ \sum_{k=1}^j \text{cost}(INS(b_k)) + \text{cost}(DEL(a_1)) \end{cases} \quad (2.6)$$

In Abbildung 2.5 ist eine komplette Pfadmatrix für das Beispiel aus 2.1.1 dargestellt. Sie wurde nach den gerade vorgestellten Berechnungsvorschriften erstellt und soll das Verfahren noch einmal verdeutlichen. Die verwendeten Kosten wurden willkürlich festgelegt.

a_i	h	u
$\text{cost}(DEL(a_i))$	1	2

b_j	h	a	l	o
$\text{cost}(INS(b_j))$	2	3	2	1

$\text{cost}(SUB(a_i, b_j))$	h	a	l	o
h	0	5	4	3
u	3	4	3	2

Auffallend an diesem Beispiel ist, daß es mehrere vollständige Verweisketten enthält. Offensichtlich kann die Überführung von a nach b durch fünf verschiedene Operationenfolgen bewerkstelligt werden, die alle dieselben Gesamtkosten erzeugen. Bei rRNS Sequenzen hat sich gezeigt, daß sich die Verweisketten in großen Teilen

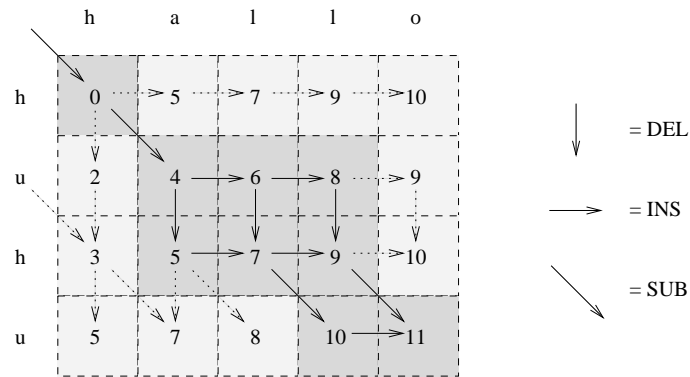


Abbildung 2.5: Die kostengünstigsten Verweisketten in einer vollständig berechneten Pfadmatrix

überdecken; dies sind die konservativen Bereiche der Sequenzen, die mit diesem Verfahren sehr gut ausgerichtet werden können. Wo sich die Verweisketten nicht mehr überdecken, sind die variablen Bereiche der Sequenzen. Um auch in diesen Bereichen eine möglichst gute Ausrichtung zu erreichen, muß das Verfahren noch verfeinert werden. Man kann dazu auf weitere Erkenntnisse aus der Mikrobiologie zurückgreifen: z.B. entspricht die stillschweigende Annahme, die aufeinanderfolgende Deletion mehrerer Basen erzeuge dieselben Kosten wie die Summe der Einzel-Deletionen dieser Basen nicht unbedingt der Realität. Durch Einführung dynamischer Kosten kann das Verfahren noch verbessert werden. Auch ein Kostenfaktor für Abweichungen in der Sekundärstruktur wäre denkbar. In [BRei] werden einige Ansätze aufgezeigt, dieses Verfahren zu optimieren. Wie dieses Verfahren konkret eingesetzt werden kann, wird im nächsten Abschnitt am Beispiel des ARB-Projekts beschrieben.

2.2 Zweistufiges Alignment im ARB-Projekt

Im ARB-Projekt des Lehrstuhls für Mikrobiologie der Technischen Universität München, in dessen Rahmen auch die vorliegende Arbeit entstanden ist, wird eine umfangreiche, rechnergestützte Umgebung zur phylogenetischen Analyse von Mikroorganismen entwickelt. Grundlegend für diese Analysen ist die korrekte Erfassung der rRNA-Sequenzen der Mikroorganismen und die Einordnung und Ausrichtung dieser Sequenzen zu und an denen ihrer nächsten Verwandten: also ein

Alignment, das auch als Datenbank Verwendung findet. Innerhalb des Projekts kommt zur Zeit noch ein rein primärstrukturbasiertes Alignment zum Einsatz, was sich aber mit der vorliegenden Arbeit und dem dazugehörigen Implementierungsteil ändern wird. Das zur Zeit noch verwendete Verfahren stützt sich auf dem in Kapitel 2.1.2 beschriebenen Algorithmus zur Distanzberechnung zweier Sequenzen mittels dynamischer Programmierung ab. Abbildung 2.6 zeigt schematisch, wie das Alignment im ARB-Projekt realisiert ist.

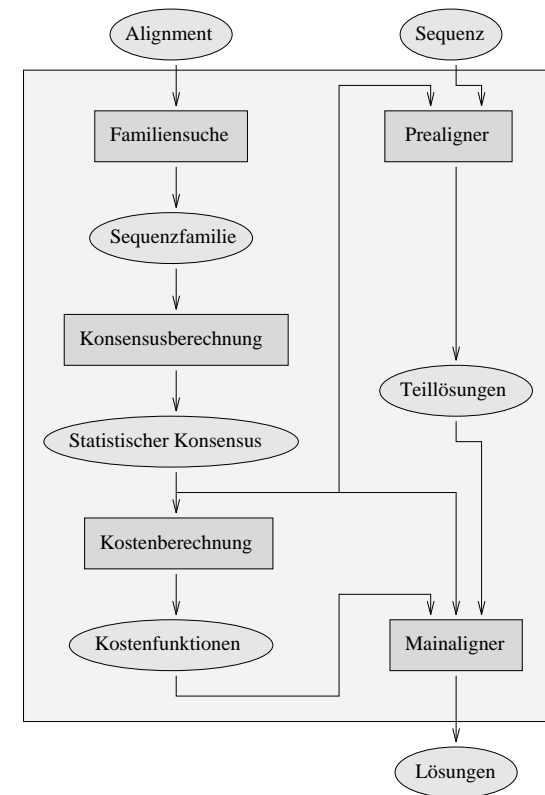


Abbildung 2.6: Das Alignment im ARB-Projekt

Bevor eine Sequenz ausgerichtet werden kann, werden ihre nächsten Verwand-

ten bestimmt. Das dabei angewandte Verfahren basiert auf der Tatsache, daß nah verwandte Sequenzen nur relativ geringe Abweichungen innerhalb der konservativen Bereiche und der Sekundärstruktur haben. Die Häufigkeit der Vorkommen und die Qualität der Übereinstimmung solcher Teilsequenzen in und mit den Sequenzen des Alignments wird als Maß für die Zugehörigkeit zur Familie der neuen Sequenz verwendet. Aus den Sequenzen der Familie wird anschließend ein statistischer Konsensus errechnet. Es entsteht eine Sequenz, die an jeder Position die Häufigkeit aller in den Sequenzen der Familie an dieser Position vorkommenden Basen beinhaltet. Mit dieser Konsensussequenz wird nun ein sogenanntes Prealignment durchgeführt, das nur dem Zweck dient, die hoch konservativen Bereiche der Sequenz zu finden und auszurichten. Es verwendet den in 2.1.2 vorgestellten Algorithmus. Es werden noch keine dynamischen Kosten eingesetzt, da das Prealignment nur dazu dient, die konservativen Bereiche auszurichten und die variablen für eine Weiterverarbeitung aufzufinden. Auf diese variablen Bereiche wird schließlich das sogenannte Mainalignment angewandt. Dieses stützt sich ebenfalls auf den bereits erwähnten Algorithmus, verwendet aber, da die variablen Bereiche wesentlich seltener und kürzer sind als die konservativen, aufwendigere Kostenmodelle. Die benötigten Kostenfunktionen für Deletion, Insertion und Substitution von Basen sind von der jeweiligen Sequenzfamilie abhängig. In [BRei] werden die mathematischen Verfahren, mit denen die einzelnen Kostenfunktionen ermittelt werden können, detailliert vorgestellt.

2.2.1 Das Prealignment

Das im ARB-Projekt verwendete Verfahren zum primärstrukturbasierten Alignment arbeitet zweistufig. Zuerst werden die hoch konservativen Bereiche der Sequenz ausgerichtet. Die verbleibenden, variablen Bereiche werden dann in der zweiten Stufe mit aufwendigeren Kostenmodellen ausgerichtet. Um mit der ersten Stufe, dem Ausrichten der konservativen Bereiche beginnen zu können, sind zuerst einige Vorarbeiten notwendig:

- Die Familiensuche

Aus vergleichenden Analysen sehr vieler rRNS Sequenzen ist bekannt, daß nah verwandte Mikroorganismen sich in den konservativen Bereichen der Primärstruktur ihrer rRNS Sequenzen stark ähneln. Auch sind die Abweichungen in der Sekundärstruktur sehr gering. Diese Informationen kann man zum einen ausnutzen, um eine neue Sequenz an der richtigen Stelle im Alignment einzufügen, zum anderen, um sie möglichst gut auszurichten. Man benötigt also ein Verfahren, das es ermöglicht, aus den Sequenzen eines Alignments, oftmals mehrere tausend, die nächsten Verwandten einer neu einzufügenden Sequenz zu finden. Im ARB-Projekt wird ein Verfahren verwendet, das es darüberhinaus erlaubt, die Sequenzen nach dem Grad ihrer Familienzugehörigkeit zu bewerten. Es basiert auf einer effizienten Suche von

Teilsequenzen der neuen Sequenz in den Sequenzen des Alignments, wobei auch eine gewisse Anzahl von Nicht-Übereinstimmungen, sogenannten 'mismatches' erlaubt sind. Je mehr solcher Teilsequenzen in einer Sequenz gefunden werden, desto ähnlicher ist sie der einzufügenden Sequenz und desto näher verwandt sind die zugehörigen Mikroorganismen. Somit läßt sich der Grad der Verwandtschaft über die Anzahl der gefundenen Teilsequenzen bewerten, wie in Abbildung 2.7 dargestellt.

Einzufügende Sequenz:



Sequenzfamilie dieser Sequenz:

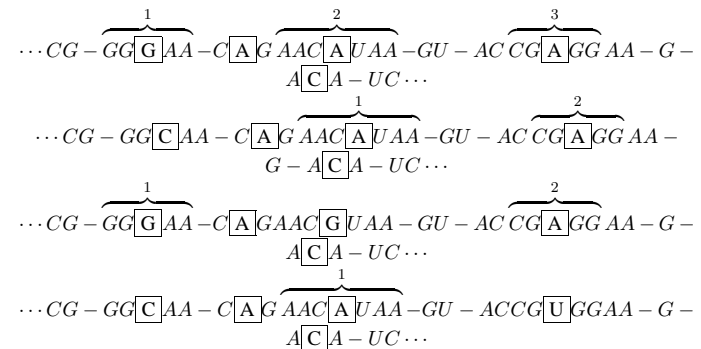


Abbildung 2.7: Bestimmung einer Sequenzfamilie anhand der Anzahl übereinstimmender Teilsequenzen einer Sequenz mit den Sequenzen eines Alignments (TS = Teilsequenz).

Problematisch bei diesem Verfahren zur näherungsweise Bestimmung der Familienbeziehungen sind die möglichen, zufälligen Übereinstimmungen von Teilsequenzen. Es stellt sich die Frage, mit welcher Wahrscheinlichkeit eine Teilsequenz s der Länge n mit m zugelassenen 'mismatches' zufällig in den Sequenzen des Alignments gefunden werden kann. Mit Hilfe von empirischen Auswertungen und statistischen Verfahren, die in [BRei] ausführlich behandelt werden, konnte für diese Wahrscheinlichkeit p folgende Formel ermittelt werden:

$$p \leq p_{max}^{n-m} (1 - p_{min})^m \binom{n}{m} \quad (2.7)$$

Die beiden Werte p_{min} und p_{max} wurden von B. Reichel in [BRe] wie folgt abgeschätzt:

$$\begin{aligned} p_{min} &= 0,16 \\ p_{max} &= 0,36 \end{aligned} \quad (2.8)$$

Es wurde gezeigt, daß die Abschätzung dieser Werte nur mit einer Wahrscheinlichkeit von 0,01 fehlerhaft ist. Aufgrund dieser Erkenntnisse ist es nun möglich, die Qualität der Familiensuche nach eigenem Gutdünken zu bestimmen. Je unwahrscheinlicher eine zufällige Übereinstimmung mit den ausgewählten Teilsequenzen ist und je weniger 'mismatches' erlaubt sind, desto ähnlicher werden die gefundenen Sequenzen der neuen Sequenz sein, aber auch um so seltener.

• Der statistische Konsensus

Um zu einer brauchbaren Repräsentation der Sequenzen einer Familie zu gelangen, wird eine Konsensussequenz erzeugt. Eine Sequenz, die genauso lang ist, wie die Sequenzen des Alignments, aber an jeder Position spezifische Informationen über die Basen aller Familiensequenz an dieser Position trägt. Die einfachste Möglichkeit einer Konsensusbildung besteht sicherlich darin, für jede Position nur die häufigste Base der Familiensequenzen zu notieren, also einen diskreten Konsensus aufzubauen, wie in Abbildung 2.8 exemplarisch dargestellt.

Dabei bleiben aber die Informationen sowohl über den Grad der Verwandtschaft der Familiensequenzen als auch über die restlichen Basen unberücksichtigt. Durch Einführung zusätzlicher Zeichen, z.B. für gleichhäufige Basen, kann dieser Ansatz noch verfeinert werden. Will man aber nach Möglichkeit keine Information ungenutzt lassen, ist ein statistischer Konsensus unabweichlich. Der im ARB-Projekt verwendete, statistische Konsensus enthält an jeder Position die durchschnittliche Gewichtung jeder der vier möglichen Basen an dieser Position der Familiensequenzen. Dabei ist unter durchschnittlicher Gewichtung folgendes zu verstehen: Wie im vorangegangenen Abschnitt dargelegt, kann jeder Sequenz $s_i = \langle s_{i1}, \dots, s_{in} \rangle$ mit $i \in [1, m]$ einer Sequenzfamilie ein Gewicht $g_i \in [0, 1]$ zugeordnet werden, das den Grad ihrer Zugehörigkeit zu dieser Familie angibt. An jeder Position wird für jede Base über alle Familiensequenzen die Summe der Gewichte g der Sequenzen gebildet, in denen sie an dieser Position auftritt. Diese Summen werden daraufhin durch die Anzahl des Auftretens der zugehörigen Basen

Sequenzfamilie:

$\dots CG - GG \boxed{G} AA - CAGAAC \boxed{A} UAA - GU - ACCG \boxed{A} GGAA - G - ACA - UC \dots$
 $\dots CG - GG \boxed{C} AA - CAGAAC \boxed{A} UAA - GU - ACCG \boxed{A} GGAA - G - ACA - UC \dots$
 $\dots CG - GG \boxed{G} AA - CAGAAC \boxed{G} UAA - GU - ACCG \boxed{A} GGAA - G - ACA - UC \dots$
 $\dots CG - GG \boxed{C} AA - CAGAAC \boxed{A} UAA - GU - ACCG \boxed{U} GGAA - G - ACA - UC \dots$

Diskrete Konsensussequenz:

$\dots CG - GG \boxed{?} AA - CAGAAC \boxed{A} UAA - GU - ACCG \boxed{A} GGAA - G - ACA - UC \dots$

Abbildung 2.8: Bildung eines diskreten Konsensus

an dieser Position geteilt. In Abbildung 2.9 soll dieses Vorgehen verdeutlicht werden, wobei Gewichte g_i mit $i = 1, \dots, 4$ willkürlich so gewählt wurden, daß $\sum_i g_i = 1$ gilt.

$g_1 = \frac{3}{8}, \dots GG \boxed{G} AA - CAGAAC \boxed{A} UAA - GU - ACCG \boxed{A} GGAA \dots$
 $g_2 = \frac{1}{4}, \dots GG \boxed{C} AA - CAGAAC \boxed{A} UAA - GU - ACCG \boxed{A} GGAA \dots$
 $g_3 = \frac{1}{4}, \dots GG \boxed{G} AA - CAGAAC \boxed{G} UAA - GU - ACCG \boxed{A} GGAA \dots$
 $g_4 = \frac{1}{8}, \dots GG \boxed{C} AA - CAGAAC \boxed{A} UAA - GU - ACCG \boxed{U} GGAA \dots$

	1	2	3
A	0	$\frac{\frac{3}{8} + \frac{1}{4} + \frac{1}{8}}{3} = \frac{1}{4}$	$\frac{\frac{3}{8} + \frac{1}{4} + \frac{1}{4}}{3} = \frac{7}{24}$
C	$\frac{\frac{1}{4} + \frac{1}{8}}{2} = \frac{3}{16}$	0	0
G	$\frac{\frac{3}{8} + \frac{1}{4}}{2} = \frac{5}{16}$	$\frac{1}{4}$	0
U	0	0	$\frac{1}{8}$

Abbildung 2.9: Bildung eines statistischen Konsensus

Tritt eine Base an einer Position nicht auf, wird sie mit 0 gewichtet; tritt nur eine Base auf errechnet sich ihre durchschnittliche Gewichtung zu 1. Formal läßt sich der statistische Konsensus K als Sequenz von Vektoren oder als Matrix darstellen:

$$K = \begin{pmatrix} w_{A_1} & \cdots & w_{A_n} \\ w_{C_1} & \cdots & w_{C_n} \\ w_{G_1} & \cdots & w_{G_n} \\ w_{U_1} & \cdots & w_{U_n} \end{pmatrix}, \quad \text{wobei}$$

$$w_{b_j} = \frac{\sum_{p=1 \text{ mit } s_{p_j}=b}^m g_p}{\sum_{q=1 \text{ mit } s_{q_j}=b}^m 1} \quad \text{mit } b \in \mathcal{B} \quad \text{und } j \in [1, n] \quad (2.9)$$

Der statistische Konsensus soll zur Ausrichtung einer neuen Sequenz mittels des in 2.1.2 beschriebenen Verfahrens herangezogen werden. Eine noch ausstehende Voraussetzung sind die benötigten Kostenfunktionen. Aus dem Konsensus können zum Teil die dynamischen Kosten für das Mainalignment berechnet werden. Im hier besprochenen Prealignment werden aber feste Kosten für das Löschen, Einfügen und Ersetzen von Basen verwendet, weswegen für die Ermittlung der Kosten auf [BRei] verwiesen sei.

Nachdem die Voraussetzungen geschaffen wurden, kann jetzt mit dem Ausrichten der neuen Sequenz an der Konsensussequenz begonnen werden. Das zugrundeliegende Verfahren wurde bereits in 2.1.2 beschrieben. Die verwendeten Kostenfunktionen arbeiten noch kontextfrei, d.h. die Kosten für Deletion, Insertion oder Substitution hängen weder von den Basen vor oder nach der geänderten Position ab, noch von der vorangegangenen Operation. Deshalb eignen sie sich auch besonders für die Ausrichtung der konservativen Bereiche. Die Pfadmatrix eines Prealignments könnte sich, wie in Abbildung 2.10 dargestellt, ergeben.

Die Bereiche, in denen der Pfad sich verzweigt, sind die variablen Bereiche. Sie konnten mit diesem Verfahren, das aus Effizienzgründen noch nicht alle möglichen Informationen ausnutzt, nicht ausgerichtet werden. Dagegen ist die Ausrichtung der konservativen Bereiche, der Bereiche, in denen der Pfad eindeutig verläuft, gelungen. Zudem liegt jetzt eine Unterteilung in ausgerichtete konservative und nicht ausgerichtete variable Bereiche vor. Das Problem der Ausrichtung einer neuen Sequenz konnte also partitioniert und zugleich, wie sich herausgestellt hat, für große Teile gelöst werden. Die verbleibenden, relativ wenigen variablen Bereiche können nun mit rechenintensiveren Algorithmen ausgerichtet werden.

2.2.2 Das Mainalignment

In der zweiten Stufe des Alignments sollen nun die noch unpositionierten variablen Bereiche der neuen Sequenz ausgerichtet werden. Da der im Prealignment

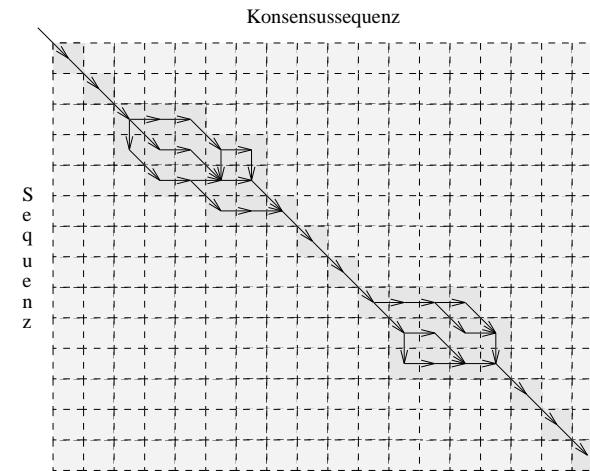


Abbildung 2.10: Pfadmatrix eines Prealignments

verwendete Algorithmus aus 2.1.2 und die verwendeten einfachen Kostenfunktionen nicht in der Lage waren, diese Bereiche zu positionieren, müssen sie verbessert werden. Vor allem das Kostenmodell muß so abgeändert werden, daß es zum einen der Realität mehr entspricht und zum anderen die Informationen des Alignments besser nutzt. In der Mikrobiologie wird davon ausgegangen, daß die Kosten für aufeinanderfolgende Löschnungen oder Einfügungen mehrerer Basen nicht notwendigerweise gleich der Summe der Kosten der Einzeloperationen ist. Die Bewertung einer Operation hängt also von der vorangegangenen ab. Substitutionen werden allerdings davon nicht betroffen. Bei Deletionen und Insertionen werden die Kosten dann mit einem zusätzlichen Faktor w_d bzw. w_i gewichtet, wenn dieselbe Operation unmittelbar vorangegangen ist. Für den Algorithmus aus 2.1.2 bedeutet das, daß in jedem Feld der Pfadmatrix nicht die minimalen Kosten aller drei Operationen, sondern die minimalen Kosten jeder der drei Operationen abgelegt werden müssen. Dasselbe gilt für die Verweiskette. Für jeden der drei Werte muß vermerkt werden, aus welcher Vorgängeroperation er entstanden ist, damit abschließend die Verweiskette mit den geringsten Gesamtkosten rekonstruiert werden kann. Abbildung 2.11 zeigt die neuen Abhängigkeiten der Felder der Pfadmatrix.

Formal lassen sich die einzelnen Kosten M^{DEL} , M^{INS} und M^{SUB} eines Feldes $M_{i,j}$ einer Pfadmatrix M wie folgt berechnen:

Kapitel 3

Sekundärstrukturbasiertes Alignment

Die beiden entscheidenden Faktoren für ein erfolgreiches Einfügen und Ausrichten einer Sequenz in ein bestehendes Alignment sind die Primär- und die Sekundärstruktur dieser Sequenz. In einem Alignment werden die Sequenzen so angeordnet, daß homologe Bereiche untereinander zu liegen kommen. Nah verwandte Sequenzen folgen dabei aufeinander, da sie sich in den homologen, konservativen Bereichen kaum unterscheiden, und bilden so eine Sequenzfamilie. In den variablen Bereichen dieser Sequenzen können dennoch starke Unterschiede auftreten. Es hat sich gezeigt, daß die Sekundärstrukturen stärker konservativ sind als die Primärstrukturen, sich also kaum unterscheiden. Deshalb soll in dieser Arbeit ein Algorithmus entwickelt werden, der diese Information für das Ausrichten einer Sequenz nutzt, also ein sekundärstrukturbasiertes Alignment¹ von Sequenzen ermöglicht. In Abbildung 3.1 soll die Sequenz aus Kapitel 2 anhand der Sekundärstruktur ausgerichtet werden.

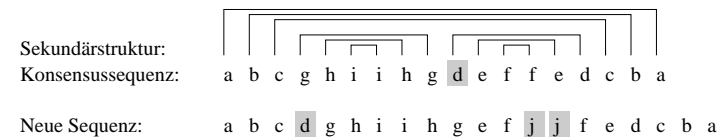


Abbildung 3.1: Wie kann die neue Sequenz durch Einfügen von Füllzeichen anhand der Sekundärstruktur an der Konsensussequenz ausgerichtet werden?

Obwohl die paarenden Basen in diesem Beispiel durch gleiche Buchstaben darge-

¹Auf ein primärstrukturbasiertes Alignment und eine formale Darstellung der Alignment-Problematik wird in Kapitel 2 eingegangen.

stellt sind, ist es für einen Menschen schwierig, die Sekundärstruktur der neuen Sequenz sofort zu erfassen. Eine andere Repräsentation erscheint notwendig. Dies könnte z.B. durch eine hierarchische Klammerstruktur, wie im ARB-Projekt, oder, wie hier im Beispiel in Abbildung 3.2, durch eine grafische Darstellung geschehen.

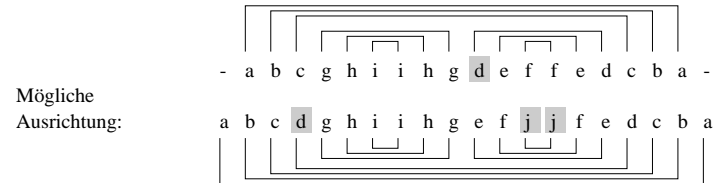


Abbildung 3.2: Mögliche Ausrichtung einer Sequenz an einer Konsensussequenz anhand der Sekundärstruktur

In diesem Fall kann die Ausrichtung anhand der Sekundärstruktur ohne große Verschiebungen realisiert werden. Allerdings fällt sofort der starke Widerspruch zum Alignment anhand der Primärstruktur aus Kapitel 2 auf. Dies liegt an den geringen Primärstrukturunterschieden zwischen der Sequenz und der Konsensussequenz. Es handelt sich demnach um einen bzgl. der Primärstruktur recht konservativen Bereich der Sequenz, in dem ein sekundärstrukturbasiertes Alignment nicht sehr sinnvoll ist. Würde es sich dagegen um einen variablen Bereich handeln, wie in Abbildung 3.3, in dem keine Übereinstimmungen in der Primärstruktur zu finden sind, wäre ein sekundärstrukturbasiertes Alignment die einzige Möglichkeit, eine sinnvolle Ausrichtung der Sequenz zu erreichen.

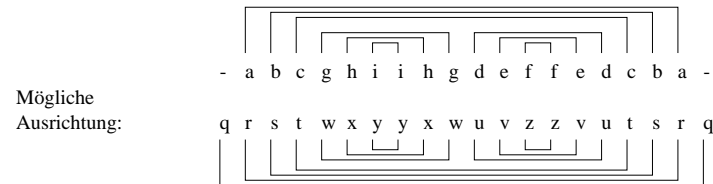


Abbildung 3.3: Mögliche Ausrichtung einer Sequenz an einer Konsensussequenz anhand der Sekundärstruktur ohne Übereinstimmungen in der Primärstruktur

Die Sekundärstruktur setzt sich aus Strukturelementen, paaren Sequenzbereichen, die Helices genannt werden, zusammen. Wie in Abbildung 3.3 deutlich wird, erreicht man ein sekundärstrukturbasiertes Alignment, indem man die Strukturelemente der neuen Sequenz den Strukturelementen der Konsensussequenz zuordnet. Es handelt sich also um eine Abbildung $align_{sec}$ von den Strukturelementen H_a

einer Sequenz a auf die Strukturelemente H_b einer anderen Sequenz b , in diesem Fall einer Konsensussequenz. Sind also zwei Sequenzen $a = \langle a_1, a_2, \dots, a_n \rangle$ und $b = \langle b_1, b_2, \dots, b_n \rangle$ mit $|a| = |b| = n$ und $a_i, b_i \in \mathcal{B}$ mit $i = 1, 2, \dots, n$ und die Mengen ihrer Helices H_a und H_b (s. 1.4) gegeben, dann läßt sich das sekundärstrukturbasierte Alignment der Sequenz a an der Sequenz b formal wie folgt darstellen:

$$align_{sec} : H_a \rightarrow H_b, h_a \mapsto align_{sec}(h_a) = h_b \quad (3.1)$$

Ein vollständiges Ausrichten einer Sequenz nur anhand der Sekundärstruktur ist allerdings nicht möglich, da die Sequenzen nur zum Teil aus Helices bestehen. Diese sind häufig die variablen Bereiche der Sequenzen und scheinen sich auf diese Weise gut ausrichten zu lassen. Da hier nur sehr wenig Primärstrukturinformation erhalten sein kann, sind primärstrukturbasierte Verfahren oftmals überfordert. In Abbildung 3.2 wurde ein Beispiel für die Ausrichtung einer bzgl. der Primärstruktur konservativen Sequenz anhand der Sekundärstruktur gegeben, das dem Ergebnis aus Abbildung 2.2 widerspricht. Offensichtlich sind eine Unterteilung der Sequenzen in konservative und variable Bereiche und eine entsprechende Verwendung von primärstruktur- und sekundärstrukturbasierten Algorithmen zur Ausrichtung unvermeidbar. In der vorliegenden Arbeit wurde ein Verfahren entwickelt, das es ermöglicht, eine Ausrichtung von Sequenzen anhand der Sekundärstruktur vorzunehmen. Wie dieses Verfahren arbeitet und wie es in das bestehende, primärstrukturbasierte Alignment im ARB-Projekt eingebunden werden kann, wird in den folgenden Kapiteln dargelegt.

3.1 Problematik der Sekundärstruktursuche bei veränderter Primärstruktur

Ist eine neue Sequenz sequenziert worden und soll sie nun in ein Alignment eingefügt und ausgerichtet werden, liegt sie zunächst als Abfolge von Basen ohne jede weitere Information vor. Um ein sekundärstrukturbasiertes Alignment dieser Sequenz durchführen zu können, muß man aber ihre Sekundärstruktur kennen, um dann die einzelnen Helices den Helices des Sekundärstrukturmodells des Alignments zuordnen zu können. Wie die Helixstruktur einer neuen Sequenz gefunden werden kann, ist die erste Frage, mit der sich dieses Kapitel beschäftigt. Es existieren Algorithmen, die die Sekundärstruktur einer rRNS Sequenz ohne zusätzliche Informationen, allein aus der Primärstruktur berechnen können. Sie arbeiten ganz ähnlich zu dem in Kapitel 2.1.1 beschriebenen Algorithmus zur Distanzberechnung zweier Sequenzen, nur daß sie die Sequenz gegen ihr eigenes, gespiegeltes Komplement ausrichten. Aufgrund der hierarchischen Eigenschaften der Sekundärstruktur muß die Matrix allerdings sehr oft abgearbeitet werden, weswegen

diese Algorithmen eine Komplexität von $O(n^3)$ haben. Sie benötigen für die Berechnung, aufgrund der großen Länge der Sequenzen (je nach Typ bis zu mehreren tausend Basen) auf gängigen Arbeitsplatzrechnern oftmals mehrere Stunden, was für eine Einbindung in ein interaktives Programmpaket wie das ARB-Projekt nicht tragbar ist. Andere Verfahren sind also gefragt, Verfahren, die die vorhandenen Informationen, wie z.B. die Helixstruktur des Alignments, in das die Sequenz eingefügt werden soll, besser ausnutzen. Damit sind z.B. die maximalen und minimalen Längen, innerhalb gewisser Toleranzen, bekannt, die in der Sequenz auftreten können. Sucht man sich alle komplementären, antiparallelen Bereiche der Sequenz zusammen, erhält man eine Menge von **möglichen** Helices, aus denen dann die der Helixstruktur des Alignments am besten entsprechenden ausgewählt werden können. Diese Suche wird allerdings durch zwei Faktoren erschwert:

- Es existiert nicht zu jeder Base genau eine komplementäre Base, mit der sie eine Bindung eingehen kann, sondern meistens mehrere (s. 1.1).
- Die Primärstruktur der Sequenz kann sich durch Mutationen verändert haben, so daß sich in den Helices vielleicht Basen wiederfinden, die die Helixstruktur aufbrechen ('mismatches').

Dieser Sachverhalt wird am Beispiel in Abbildung 3.4 verdeutlicht.

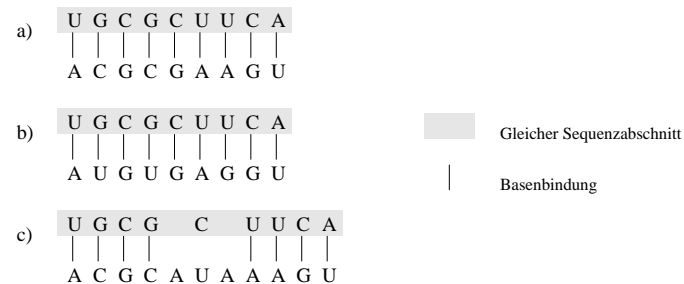


Abbildung 3.4: Mögliche Bindungen derselben Teilsequenz, die zu einer Helix führen

Die Anzahl der möglichen Helices hängt von der Länge dieser Sequenz und den Längen der Helices ab. Sei p die Wahrscheinlichkeit an einer bestimmten Position eine zu einer Base komplementäre Base anzutreffen, n die Länge der Sequenz und h die Länge einer möglichen Helix, dann lassen sich wahrscheinlich

$$p^h \frac{(n-h)(n-2h)}{2} = p^h \frac{n^2 - 3hn + 2h^2}{2} \quad (3.2)$$

solcher möglichen Helices der Länge h in der Sequenz finden. Die Wahrscheinlichkeit p kann in etwa mit $\frac{1}{2}$ abgeschätzt werden, da eine Base durchschnittlich mit zwei anderen Bindungen eingehen kann. Sucht man z.B. alle Helices der Länge 10 in einer Sequenz der Länge 1000, so wird man etwa 474 finden können. Ist man an Helices verschiedener Länge interessiert, vervielfacht sich dieser Wert noch. Im Gegensatz zu der hohen Anzahl möglicher Helices existieren z.B. bei 16S rRNS Sequenzen etwa 50 tatsächlich realisierter Helices in verschiedenen Längen. Erschwerend kommt noch hinzu, daß sich viele der möglichen Helices überschneiden oder gar kürzere Helices komplett enthalten. Bei der Zuordnung der Helices zu denen der Helixstruktur des Alignments, dem eigentlichen Alignment-Vorgang, muß darauf geachtet werden, daß es

- zu keinen Überschneidungen kommt und
- die Reihenfolge der Helices der Helixstruktur des Alignments eingehalten wird.

Bei 16S rRNS Sequenzen z.B., muß eine Auswahl von etwa fünfzig möglichen Helices getroffen werden, die je einer der Helices der Helixstruktur des Alignments zugeordnet werden, ohne das es dabei zu den gerade beschriebenen Konflikten kommt. In Abbildung 3.5 wird dargestellt, wie der zu dieser Arbeit gehörende Implementierungsteil strukturell aufgebaut ist und schrittweise die beschriebenen Probleme löst, bis er schließlich zu einem sekundärstrukturbasierten Alignment einer Sequenz gelangt.

3.2 Suche nach möglichen Strukturelementen

Im Gegensatz zu den deduktiven Algorithmen, die aus der Primärstruktur und den möglichen Basenbindungen die Sekundärstruktur einer rRNS-Sequenz ableiten, wird in dieser Arbeit ein konstruktiver Weg beschritten. Die Helixstruktur einer neuen, in ein gegebenes Alignment einzufügenden Sequenz wird aus allen möglichen Helixstrukturen durch Vergleich mit der Helixstruktur des Alignments, welche als Modell für die Helixstrukturen aller Sequenzen des Alignments fungiert, ermittelt. Dazu ist es notwendig, alle möglichen antiparallelen, komplementären Teilsequenzen, die Helices bilden könnten, zu erfassen. Wie in Abbildung 3.6 setzt sich eine Helix aus zwei, nicht unbedingt gleichlangen Teilsequenzen zusammen, die sich nicht überschneiden und deren Basen untereinander Bindungen eingehen können.

Jede Base geht zwar bevorzugt mit einer bestimmten, komplementären Base eine Bindung ein, kann aber auch mit anderen Basen schwächere Bindungen eingehen. Will man alle möglichen Helices beliebiger Länge einer mehrere tausend Basen langen Sequenz suchen, so wird deren Anzahl in Anbetracht der Formel 3.2 sehr

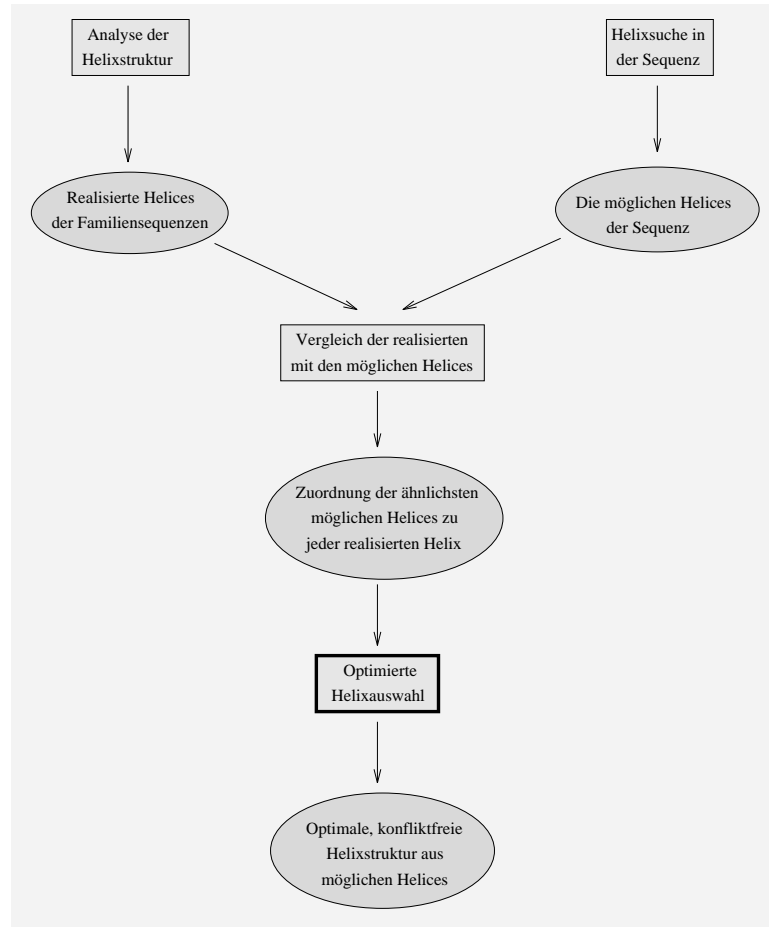


Abbildung 3.5: Struktureller Ablauf des sekundärstrukturbasierten Alignments

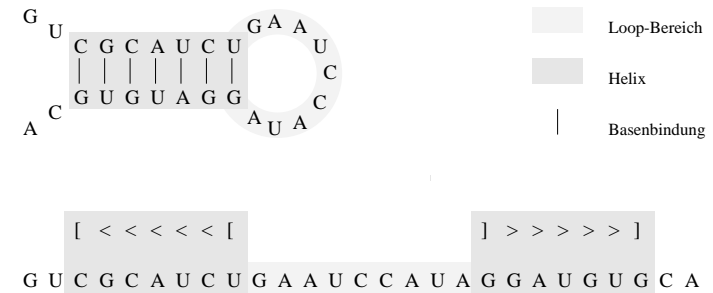


Abbildung 3.6: Tatsächlicher Aufbau einer Helix (oben) und die aufgefaltete Ansicht (unten) mit Klammerstruktur zu ihrer Verdeutlichung

hoch sein. Bei der Suche nach allen möglichen Helices der Längen 7 bis 12 in einer Sequenz aus 1000 Basen z.B. kommt man auf 7509 Möglichkeiten. Läßt man auch noch eine gewisse Anzahl k von 'mismatches' zu, also Basen, die (z.B. aufgrund von Mutationen) im anderen Teil der Helix kein paarendes Gegenstück besitzen, vervielfältigen sich die Möglichkeiten noch einmal. In diesem Fall muß die Formel 3.2 wie folgt abgeändert werden:

$$\left(p^h + p^{h-k} \binom{h}{k} \right) \frac{n^2 - 3hn + 2h^2}{2} \quad (3.3)$$

Läßt man in obigem Beispiel genau einen 'mismatch' zu, so erhält man 126159 Möglichkeiten, also um Faktor 17 mehr. Man muß daher Maßnahmen ergreifen, um den Suchraum so einzugrenzen, daß die Anzahl der möglichen Helices handhabbar wird. Tatsächlich sind die unternommenen Schritte zum Auffinden der Helices eine sukzessive Einschränkung des Suchraums.

- Zunächst wird ein primärstrukturbasiertes Prealignment (s. 2.2.1) der Sequenz durchgeführt, wodurch die Sequenz in ausgerichtete, konservative und nicht ausgerichtete, variable Bereiche unterteilt wird.
- Zusätzlich wird eine diskrete Konsensussequenz der Sequenzfamilie dieser Sequenz erzeugt, mit deren Hilfe die Helixstruktur des Alignments analysiert wird, d.h. aus dem Sekundärstrukturmodell des Alignments die in dieser Familie tatsächlich realisierten Helices herausgesucht werden.
- Daraufhin werden, für die noch nicht ausgerichteten Bereiche der Sequenz, alle möglichen Helices gesucht, die eine gewisse Ähnlichkeit mit den realisierten Helices dieses Bereiches haben. In diesem ersten Auswahlschritt

bezieht sich die Ähnlichkeit nur auf maximale und minimale Längen und minimale, durchschnittliche Bindungsstärken der Helices.

- Im zweiten Auswahlschritt werden jeder realisierten Helix eine Menge der ähnlichsten möglichen Helices aus deren Bereich zugeordnet. Hierbei wird dann nicht nur die Länge und die durchschnittliche Bindungsstärke, sondern auch die Positionen der Helixhälften, die Anzahl der Füllzeichen und noch einige Kriterien mehr zum Vergleich herangezogen.
- Zum Schluß wird zu jeder realisierten Helix je eine der zugehörigen möglichen Helices so ausgewählt, daß zwischen diesen keinerlei Konflikte auftreten. Was darunter zu verstehen ist, wird in Kapitel 3.3 geklärt.

Die einzelnen gerade beschriebenen Schritte werden in den folgenden Abschnitten eingehend beschrieben. Bezüglich des primärstrukturbasierten Prealignments und der erwähnten Konsensussequenz sei noch einmal auf Kapitel 2.2.1 verwiesen.

3.2.1 Analyse der vorgegebenen Struktur

Die aus einem Alignment gewonnene Helixstruktur, die eine Art Modell der Sekundärstrukturen aller Sequenzen eines Alignments darstellt, kann dazu verwendet werden, die am wahrscheinlichsten in einer neu einzufügenden Sequenz vorkommenden Helices zu ermitteln. Führt man zuerst ein primärstrukturbasiertes Alignment, z.B. nach dem in 2.1.2 vorgestellten Algorithmus, einer neu einzufügenden Sequenz durch, so werden dabei die nächsten Verwandten der Sequenz ermittelt. Aus dieser Sequenzfamilie kann dann eine diskrete Konsensussequenz (s. Abbildung 2.8) gebildet werden, die an jeder Position die häufigste in den Sequenzen der Familie an dieser Position vorkommende Base enthält. Vergleicht man nun diese Konsensussequenz mit der Helixstruktur des Alignments, kann man die in dieser Familie realisierten Helices, wie in Abbildung 3.7 dargestellt, bestimmen.

Dabei ist zu überprüfen, ob alle Basenpaare, die die Helixstruktur vorgibt, in der Konsensussequenz tatsächlich vorhanden sind und ob sie auch Bindungen eingehen können. Aus empirischen Analysen ist bekannt, daß die Basen, wie in Abbildung 3.8 dargestellt, miteinander unterschiedlich starke Bindungen eingehen können.

Die minimalen und maximalen Längen und die durchschnittlichen Bindungsstärken der realisierten Helices der Sequenzfamilie sollen später verwendet werden, um die Suche nach möglichen Helices der neuen Sequenz zu begrenzen. Bei dieser Suche wird nach allen komplementären, antiparallelen Bereichen einer Teilsequenz t der Sequenz s gesucht, mit denen diese eine Helix bilden könnte. Dabei stellt sich die Frage, ab welcher Länge n der Teilsequenz t eine Suche überhaupt sinnvoll wird. Mithilfe der Formeln 3.2 und 3.3 kann die Anzahl der paarenden Teilsequenzen einer bestimmten Länge berechnet werden. Je kürzer die Teilsequenz und je länger

Helixstruktur:

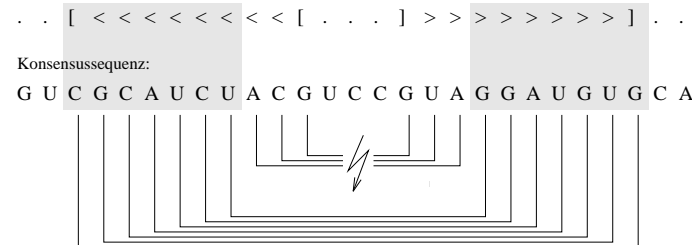


Abbildung 3.7: Ermittlung der realisierten Helices einer Sequenzfamilie anhand einer diskreten Konsensussequenz der Familie und der Helixstruktur des Alignments

	A(denin)	C(ytosin)	G(uanin)	U(racil)
A(denin)	0	0	0,5	1,0
C(ytosin)	0	0	1,5	0
G(uanin)	0,5	1,5	0	0,8
U(racil)	1,0	0	0,8	0

Abbildung 3.8: Empirisch ermittelte Bindungsstärken der Basen relativ zur Bindungsstärke zwischen Adenin und Uracil

die Sequenz, desto mehr Möglichkeiten gibt es, weswegen eine Suche nach Teilsequenzen der Längen eins oder zwei unterbleiben sollte. Es muß auch berücksichtigt werden, daß die Helices selten eine Länge von zwanzig Basen überschreiten, weshalb die Untergrenze für die zu suchenden Teilsequenzen auf fünf bis sieben festgelegt wurde.

Die in einem Alignment enthaltene Helixstruktur wird manuell erfaßt und ständig erweitert, bzw. anhand neuer Informationen durch neu hinzugekommene Sequenzen berichtigt. Es können also durchaus Fehler in dieser Helixstruktur enthalten sein. Vor allem an den Rändern von Helices ist es schwierig, die Helixstruktur an alle Sequenzen richtig anzupassen, weshalb ein Algorithmus, der die realisierten Helices einer Sequenzfamilie herausucht, diese möglichen Fehlerquellen abfangen sollte. In Abbildung 3.9 ist ein solcher Fall exemplarisch dargestellt, in dem die Helix um zwei weitere Bindungen erweitert werden mußte.

Der für die beschriebene Aufgabe der Helixstrukturanalyse implementierte Algorithmus untersucht deshalb die Basen vor und hinter den Helixhälften auf eventuelle Bindungsmöglichkeiten. Sollten sich solche ergeben, obwohl die Helixstruktur keine vorsieht, wird die Helix, sofern es dabei nicht zu Konflikten mit anderen Helices kommt, erweitert. Schwierig zu behandeln sind dabei die besonders schwachen

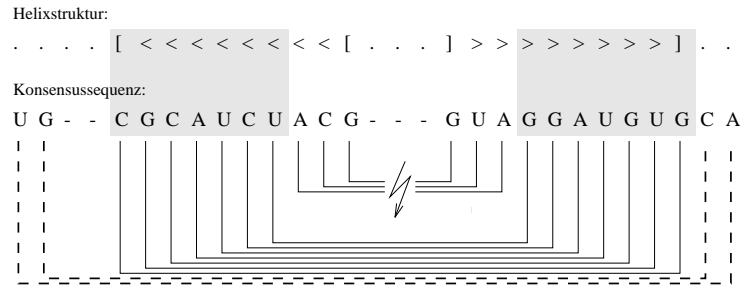


Abbildung 3.9: Problematik einer nicht vollständigen Helixstruktur eines Alignments und einer Konsensussequenz mit Füllzeichen

Bindungen. Es könnte sich dabei um eine zufällige Paarung handeln. Deshalb wurde ein Schwellenwert eingeführt, der angibt, ab welcher Bindungsstärke eine Helix aufgrund solcher zusätzlichen Bindungen erweitert werden soll.

Aus den realisierten Helices der Familie einer neu einzufügenden Sequenz können nun die ersten Informationen gezogen werden, mit denen die spätere Suche nach den möglichen Helices eingegrenzt werden kann. Da sind zum einen die Längeninformationen. Die Mindestlänge für Helices ergibt sich, wie oben beschrieben, unabhängig von der Sequenzfamilie aus den statistischen Betrachtungen über die Zufälligkeit von komplementären, antiparallelen Bereichen zu fünf bis sieben Basen. Existieren aber beispielsweise gar keine so kurzen Helices unter den realisierten Helices, ist es auch nicht sinnvoll nach solchen zu suchen. Sowohl die Unter- als auch die Obergrenze für die Länge von den gesuchten Helices sollte sich innerhalb gewisser Toleranzen etwas unter der kürzesten und etwas über der längsten realisierten Helix bewegen, wobei die Obergrenze stärker erweitert werden kann, um auch stärker veränderte Helices finden zu können. Des Weiteren ist die minimale durchschnittliche Bindungsstärke der realisierten Helices interessant. Mit ihrer Hilfe können Helices, die deutlich unter diese Grenze fallen, aussortiert werden, was eine weitere Eingrenzung des Suchraums ermöglicht. Im folgenden werden noch weitere Verfahren zur Eingrenzung des Suchraums und der Suche selbst vorgestellt.

3.2.2 Beschränkung des Suchraums

Das Ziel eines sekundärstrukturbasierten Alignments ist eine möglichst eindeutige Zuordnung der tatsächlich in einer Sequenzfamilie realisierten Helices zu denen in einer Sequenz möglichen Helices. Da in der vorliegenden Arbeit ein konstruk-

tiver Weg beschritten wird, das heißt, die Sekundärstruktur so aus den möglichen Helices zusammengesetzt werden soll, daß sie der tatsächlichen möglichst gut entspricht, muß die Anzahl einer realisierten Helix zuordbaren möglichen Helices sukzessive verringert werden. Jeder realisierten Helix wird zunächst eine Teilmenge der möglichen Helices zugeordnet, die dann Schritt für Schritt verkleinert wird, bis schließlich nur je eine mögliche Helix übrig bleibt. Dabei sind allerdings gewisse Randbedingungen, wie Überschneidungen und die Helixreihenfolge, zu beachten. Auf diese Problematik wird noch ausführlich in Kapitel 3.3.1 eingegangen werden. Für eine Sequenz s der Länge $|s| = n$, die durch die geordnete Menge $S = \{b_1, \dots, b_n\}$ ihrer Basen bestimmt sei, läßt sich eine Teilsequenz T_j wie folgt definieren:

$$S \supseteq T_j := \{b_i, b_{i+1}, \dots, b_{i+(j-1)}\} \quad \text{mit } 1 \leq j \leq n \quad \text{und} \quad 1 \leq i \leq n - j + 1 \quad (3.4)$$

Somit kann die Menge der möglichen Helices H_m der Sequenz s formal als die Menge aller Paare (T_{pj}, T_{qj}) von komplementären, antiparallelen Teilsequenzen von s dargestellt werden:

$$H_m = \{(T_{pj}, T_{qj}) \mid b_{p+l} \text{ ist komplementär zu } b_{q+j-1-l}\} \\ \text{mit } p \leq q - j \quad \text{und} \quad 0 \leq l < j \quad (3.5)$$

Diese Menge bestimmt den Suchraum, wenn man nur Helices zuläßt, deren Helixlängen gleich lang sind. Will man aber auch eventuell mutierte Helices finden, die, wie in Abbildung 3.10 dargestellt, einen oder mehrere 'mismatches' enthalten können, vergrößert sich der Suchraum enorm.

Durch ein primärstrukturbasiertes Prealignment der Sequenz können die hoch konservativen Bereiche bereits ausgerichtet werden. Dadurch wird implizit erreicht, daß die Sequenz in ausgerichtete und unausgerichtete Bereiche unterteilt wird. Das sekundärstrukturbasierte Alignment wird also nur für die noch unausgerichteten Bereiche benötigt: Nur die realisierten Helices der noch unausgerichteten Bereiche müssen noch je einer der möglichen Helices zugeordnet werden. Auf diese Weise gelingt es, den Suchraum zu partitionieren und die Suche nur auf die relevanten Bereiche zu konzentrieren. Es wird, wie in Abbildung 3.11 veranschaulicht, zum einen die Menge der zuzuordnenden, realisierten Helices verringert und zum anderen die Sequenz in kürzere Abschnitte unterteilt, in denen schneller gesucht werden kann.

Bei der Unterteilung der Sequenz in ausgerichtete und unausgerichtete Bereiche kann es zu Überlappungen von realisierten Helices der Sequenzfamilie und Bereichsgrenzen kommen. Es ist aber wünschenswert, daß die Helices, d.h. eigentlich

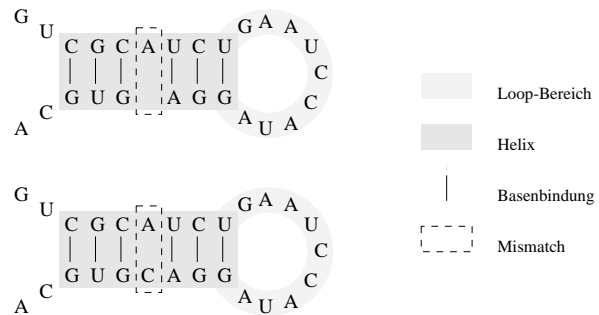


Abbildung 3.10: Beispiel für mögliche 'mismatches' in den Helices

nur die Helixhälften vollständig in einem, nicht notwendigerweise demselben Bereich liegen, da dann die einzelnen Bereiche unabhängig voneinander durchsucht werden können. In den Fällen, in denen es zu solchen Überlappungen kommt, d.h. eine realisierte Helixhälfte sowohl in einem ausgerichteten als auch einem unausgerichteten Bereich liegt, sollten die Bereichsgrenzen, wenn möglich, entsprechend angepaßt werden. In Abbildung 3.12 ist dieser Vorgang schematisch dargestellt. Es ist einleuchtend, daß die Bereichsgrenzen nur so verschoben werden können, daß die bereits ausgerichteten Bereiche verkleinert und die unausgerichteten vergrößert werden, da in letzteren ja noch keine Information darüber vorliegt, ob die Helix an diesen Stellen mit bindenden Basen besetzt ist.

Wie gerade angesprochen, ist es nicht notwendig, daß die Helixhälften in ein und demselben Bereich liegen, solange sie nach Möglichkeit vollständig in ihrem eigenen Bereich liegen. Tatsächlich verhält es sich so, daß bei der Unterteilung der Sequenz in Bereiche fünf Klassen von Helices entstehen:

1. Helices, deren Hälften in demselben, ausgerichteten Bereich liegen,
2. Helices, deren Hälften in unterschiedlichen, ausgerichteten Bereichen liegen,
3. Helices, deren Hälften in demselben, unausgerichteten Bereich liegen,
4. Helices, deren Hälften in unterschiedlichen, unausgerichteten Bereichen liegen und
5. Helices, deren Hälften je in einem ausgerichteten und einem unausgerichteten Bereich liegen.

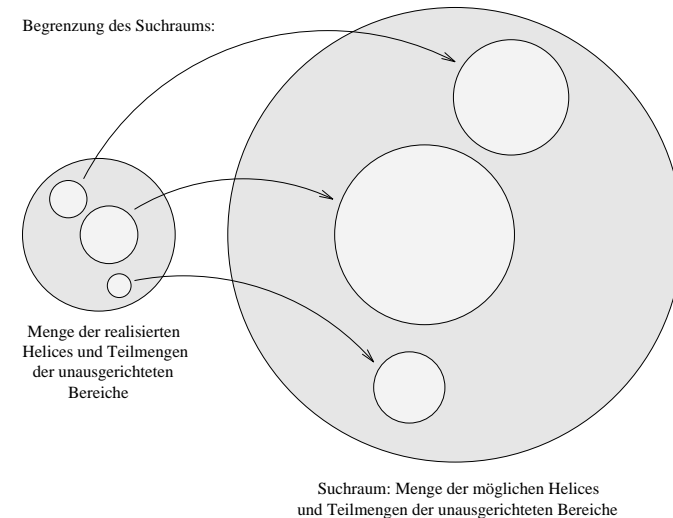
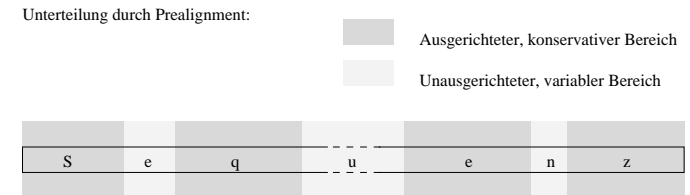


Abbildung 3.11: Begrenzung des Suchraums durch primärstrukturbasiertes Prealignment der Sequenz und der damit verbundenen Unterteilung in aus- und unausgerichtete Bereiche

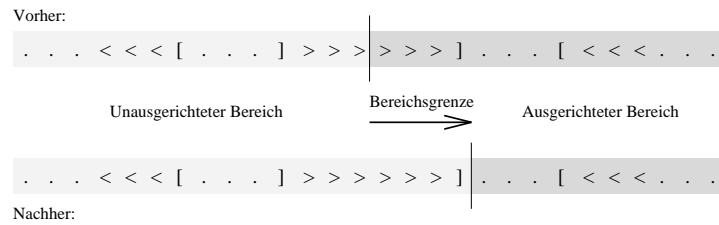


Abbildung 3.12: Anpassung der Bereichsgrenzen bei Überlappung mit Helices

Die Helices der ersten beiden Klassen sind allerdings für das sekundärstrukturbasierte Alignment uninteressant, da sie ja schon vollständig ausgerichtet sind. Die Helices der dritten Klasse sind relativ einfach zu handhaben, da auf sie eine bereichsunabhängige Suche angewandt werden kann. Auf die Helices der beiden letzten Klassen dagegen muß eine bereichsabhängige Suche angewendet werden. Die Helices der letzten Klasse sind besonders interessant, da sie zur Hälfte bereits ausgerichtet sind, man also die Primärstruktur einer Helixhälfte bereits kennt. Die bereichsabhängigkeiten der Helices lassen sich in einer Matrix über den Bereichen gut darstellen, wie es Abbildung 3.13 zeigt. In diesem Beispiel wurde eine 23S rRNS Sequenz verwendet, die in sieben Bereiche unterteilt ist.

		Bereich						
		1	2	3	4	5	6	7
Bereich	1	8	2					1
	2	2	1					
	3			23		1		
	4				1	3		
	5			1	3	27	1	
	6					1	2	
	7	1						4

	Unausgerichtet
	Aus/Unausgerichtet
	Ausgerichtet

Abbildung 3.13: Eine symmetrische Bereichsmatrix, die die Anzahl der Helices und deren bereichsabhängigkeiten angibt

Eine solche Bereichsmatrix ist offensichtlich symmetrisch zur Hauptdiagonale, weswegen die obere rechte Dreiecksmatrix zur Darstellung der Abhängigkeiten ausreicht. Bei der Implementierung hat sich allerdings gezeigt, daß die komplette Matrix leichter und effizienter zu handhaben ist. Am vorangegangenen Beispiel fällt auf, daß die Matrix weitgehend unbesetzt ist. Die meisten besetzten Felder liegen auf der Hauptdiagonale oder dicht daneben. Die Helixhälfen liegen also nie allzuweit auseinander, abgesehen von der Helix, die sich an den Enden der Sequenz bildet. Es lassen sich also schon gewisse Rückschlüsse auf die Sekundärstruktur der Sequenzen dieser Familie ziehen. Aus dieser Matrix können auch Abhängigkeitsspuren für die Bereiche gebildet werden, d.h. die von einander abhängigen Bereiche zusammengefaßt werden. In obigen Beispiel sind es genau zwei:

1. die Bereiche 1,2 und 7 und
2. die Bereiche 3,4,5 und 6.

Diese Abhängigkeiten zwischen den Bereichen werden später bei der Zuordnung der Helices eine Rolle spielen.

Durch das primärstrukturbasierte Prealignment wird die Sequenz in ausgerichtete und unausgerichtete Bereiche unterteilt. Nach der Anpassung der Bereichsgrenzen an die realisierten Helices können die bereichsabhängigkeiten der Helices in einer Bereichsmatrix dargestellt werden. Durch dieses Vorgehen ist der Suchraum für die möglichen Helices deutlich geschrumpft. Allerdings sind noch weitere Maßnahmen notwendig, den Suchraum einzuschränken. Eine wird direkt von der vorangegangenen Unterteilung geliefert. Nachdem jetzt bekannt ist, welche der Helices in welchen Bereichen liegen und ob nach ihnen bereichs- oder bereichsunabhängig gesucht werden muß, kann man für jeden Bereich die minimale und die maximale Länge der realisierten Helices ermitteln. Bei der Suche nach den möglichen Helices kann man sich dann auf die beschränken, deren Längen sich mit gewissen Toleranzen innerhalb dieser Grenzen befinden, da eine drastische Verkürzung oder Verlängerung der Helices nicht zu erwarten ist. Die Untergrenze kann durch die Längenbeschränkung bei der Suche nach den realisierten Helices nicht unter fünf bis sieben sinken. Für die Obergrenze wurde recht großzügig das Doppelte der Maximallänge verwendet. Dennoch beschränkt man den Suchraum dadurch schon sehr stark. Alle sehr kurzen Helices mit Längen unter fünf bis sieben fallen somit aus dem Suchraum, was eine enorme Einschränkung bedeutet, da bei solchen Helices das Auffinden zufälliger komplementärer, antiparalleler Bereiche sehr wahrscheinlich ist. Die Begrenzung nach oben ist auch sinnvoll, da besonders lange Helices nicht sehr häufig sind und man somit eine sinnlose Suche nach überlangen Helices unterbindet.

Ein weiterer Ansatz zur Beschränkung des Suchraums ist nicht ganz so unproblematisch. Man kann aus den realisierten Helices eines Bereiches eine Untergrenze für die durchschnittliche Bindungsstärke der Helices berechnen. Die da-

zu notwendigen Stärken der Basenbindungen sind in Abbildung 3.8 dargestellt. Die durchschnittliche Bindungsstärke einer Helix berechnet sich aus der Summe der Bindungsstärken der sich bindenden Basen, geteilt durch die Länge der Helix. Problematisch an der Verwendung dieses einschränkenden Faktors ist, daß es sich dabei um einen aus der Primärstruktur ermittelten Faktor handelt. Allerdings wird im weiteren Verlauf, vor allem bei der Zuordnung der Helices, die Primärstruktur eine immer stärkere Rolle spielen. Darüberhinaus wird die Vermutung zugrunde gelegt, daß sich bei eventuellen Mutationen die durchschnittliche Bindungsstärke nicht gravierend ändert. Diese Vermutung ist zwar plausibel, entbehrt aber bis jetzt noch sowohl einer empirischen Bestätigung, als auch einer Widerlegung. Es hat sich gezeigt, daß diese Vermutung bei einer Suche nach Helices mit 'mismatches' eine deutliche Verringerung der Möglichkeiten mit sich bringt, bei einer Suche ohne 'mismatches' aber keine besondere Auswirkung auf die Anzahl der gefundenen möglichen Helices hat. Zusammenfassend läßt sich feststellen, daß sich durch ein vorangegangenes primärstrukturbasiertes Alignment doch einige Möglichkeiten bieten, den Suchraum für die möglichen Helices einzuschränken. Im folgenden Kapitel wird darauf eingegangen werden, wie diese Möglichkeiten bei der Suche eingesetzt werden können.

3.2.3 Suche nach Strukturelementen

Die Unterteilung der Sequenz in ausgerichtete und unausgerichtete Bereiche ermöglicht es, die Suche nach den möglichen Helices auf die unausgerichteten Sequenzbereiche zu beschränken, statt die Suche über der gesamten Sequenz durchzuführen. Die realisierten Helices der ausgerichteten Bereiche sind nur dann relevant, wenn eine ihrer Helixhälften in einem unausgerichteten Bereich liegt. Abbildung 3.14 zeigt noch einmal das Beispiel einer Bereichsmatrix aus Abbildung 3.13 auf das Wesentliche reduziert.

Aus der Bereichsmatrix kann man ablesen, in welchen Bereichen die möglichen Helices zu suchen sind, wobei die in Kapitel 3.2.2 beschriebenen Ansätze die Suche etwas vereinfachen. Ziel ist es, für die möglichen Helices ebenfalls eine solche Bereichsmatrix aufzustellen, so daß eine Zuordnung der Helices möglich wird. In Abbildung 3.16 ist eine solche abgebildet. Es handelt sich dabei um die Bereichsmatrix, die sich aus der Suche nach den möglichen Helices der Sequenz aus Beispiel 3.14 ergibt, wenn keine 'mismatches' zugelassen werden. Gesucht werden Helices, also Paare von komplementären, antiparallelen Teilsequenzen, eventuell mit einem oder mehreren 'mismatches' (s. Kapitel 3.2.2), deren Längen sich innerhalb gewisser Grenzen bewegen und deren mittlere Bindungsstärken nicht unter bestimmten Werten liegen dürfen. Es handelt sich also um eine Mustersuche auf der Primärstruktur der einzelnen Sequenzbereiche. Der verwendete Suchalgorithmus, ein Positionsbau, wird in Kapitel 4.1.3 eingehend beschrieben werden. Es sei hier nur vorweggegriffen, daß der Algorithmus recht strengen Anforderungen gerecht

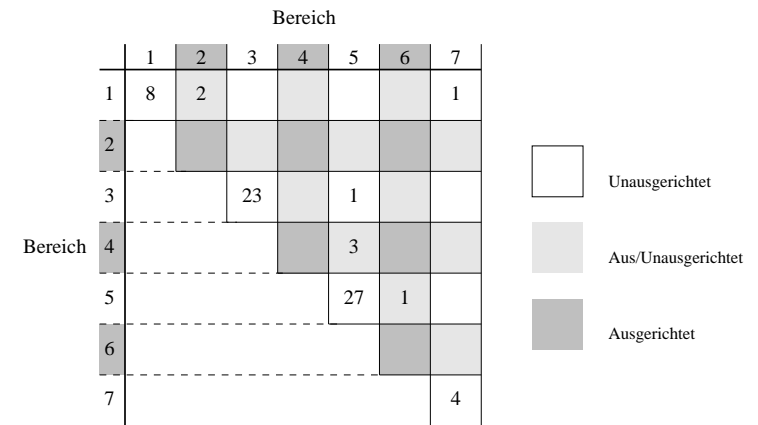


Abbildung 3.14: Obere rechte Dreiecksmatrix einer Bereichsmatrix mit den für die Suche nach den möglichen Helices relevanten Informationen

werden muß. Zum einen muß er aufgrund der großen Anzahl der möglichen Helices eine effiziente Suche erlauben, zum anderen muß er die Längenbeschränkungen und mittleren Basenbindungen berücksichtigen können und eine Suche mit einer gewissen Anzahl von 'mismatches' zulassen.

Steht ein geeigneter Algorithmus zur Verfügung, stellt sich die Frage nach den zu suchenden Mustern. Bei den realisierten Helices, die sich zur einen Hälfte in einem ausgerichteten und zur anderen in einem unausgerichteten Bereich befinden, ist die Musterwahl kein Problem. Die Helixhälften, die im ausgerichteten Bereich liegen, stellen die Suchmuster dar. Die Primärstruktur und die Länge sind bekannt, also kann direkt mit der Suche nach allen komplementären, antiparallelen Teilsequenzen derselben Länge in dem zugehörigen unausgerichteten Bereich begonnen werden. Bei den Helices, bei denen sich die Hälften in demselben oder in unterschiedlichen unausgerichteten Bereichen liegen, müssen die Muster zuerst erzeugt werden. Durch die realisierten Helices dieser Bereiche lassen sich die Längenbeschränkungen (s. 3.2.2) ermitteln. Es müssen alle Teilsequenzen des Bereiches gebildet werden, deren Längen innerhalb dieser Grenzen liegen, wie in Abbildung 3.15 dargestellt, um sie als Muster für die Suche nach ihren möglichen Gegenstücken in demselben oder anderen unausgerichteten Bereichen verwenden zu können. Die Anzahl n der Muster errechnet sich bei einem Bereich der Länge l und den beiden

L'angengrenzen l_{min} und l_{max} wie folgt:

$$n = \sum_{m=l_{min}}^{l_{max}} l - m + 1 = \sum_{m=l+1-l_{max}}^{l+1-l_{min}} m \quad \text{mit} \quad 0 < l_{min} \leq l_{max} \leq l \quad (3.6)$$

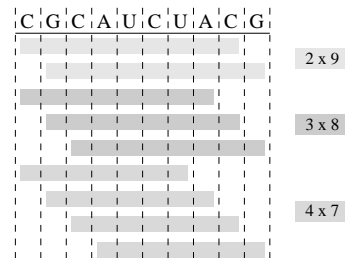


Abbildung 3.15: Bildung aller Teilsequenzen der L'angen 7 - 9 eines Bereiches der L'ange 10

Der verwendete Suchalgorithmus vereinfacht dieses Vorgehen etwas. Es gen'ugt als Muster alle Teilsequenzen der maximalen L'ange bereitzustellen. Unter Verwendung der minimalen L'ange liefert der Suchalgorithmus alle komplement'aren, antiparallelen Teilsequenzen, deren L'angen zwischen der Ober- und der Untergrenze liegen.

Bereits bei einer Suche ohne 'mismatches', wie in Abbildung 3.16 dargestellt, ist die Anzahl der m'oglichen Helices sehr hoch. L'asst man einen oder gar mehrere 'mismatches' zu, so vervielfacht sich die Anzahl noch. In den folgenden Abschnitten wird darauf eingegangen, wie trotz so grober Auswahlm'oglichkeiten eine sinnvolle Zuordnung der realisierten Helices erreicht werden kann.

3.2.4 Zuordnung der gefundenen Strukturelemente

Zu jedem unausgerichteten Bereich und zu den ausgerichteten, deren realisierte Helices sich zur H'alfte in unausgerichteten Bereichen befinden, liegen jetzt eine groBe Anzahl m'oglicher Helices vor. Diese Anzahl h'angt vor allem von der L'ange der Bereiche ab. Die in Abbildung 3.16 angegebenen Zahlen f'ur die m'oglichen Helices der Bereiche sind aus einer sehr unrealistischen Unterteilung der Sequenz entstanden. In diesem Fall sind die ausgerichteten Bereiche nur einige hundert Basen lang, wohingegen die unausgerichteten Bereiche aus bis zu mehreren tausend Basen bestehen. Normalerweise ist ein umgekehrtes Verh'altnis anzustreben. Dennoch besteht diese M'oglichkeit, weswegen der n'achste Schritt nicht die direkte

		Bereich						
		1	2	3	4	5	6	7
Bereich	1	(8) 1432	(2) 28					(1) 1
	2							
	3			(23) 13142		(1) 129		
	4				(3) 1071			
	5					(27) 18934	(1) 11	
	6							
	7							(4) 45

	Unausgerichtet
	Aus/Unausgerichtet
	Ausgerichtet

Abbildung 3.16: Bereichsmatrix f'ur die m'oglichen Helices ohne 'mismatches'. In Klammern ist jeweils die Anzahl der realisierten Helices angegeben

Zuordnung jeder realisierten Helix eines Bereiches zu genau einer m'oglichen Helix dieses Bereiches sein wird, sondern die Zuordnung zu je einer wie in Abbildung 3.17 dargestellten Teilmenge der m'oglichen Helices dieses Bereiches.

Die m'oglichen Helices dieser Teilmengen werden nach ihrer 'Ähnlichkeit zu der ihnen zugeordneten realisierten Helix ausgew'ahlt und auch bewertet, da dieses Kriterium sp'ater zu ihrer Sortierung herangezogen wird. Dabei stellt sich unmittelbar die Frage, wie die 'Ähnlichkeit zweier Helices zu bestimmen ist, wenn man auf m'oglichst wenig Prim'arstrukturinformation zur'uckgreifen will, da es sich hierbei um ein sekund'arstrukturbasiertes Alignment handelt. Bei der Suche nach den m'oglichen Helices wurden im Prinzip auch 'Ähnlichkeiten ausgenutzt, allerdings 'Ähnlichkeiten von Mengen:

- Die 'Ähnlichkeit bez'uglich des Bereiches: Es wurden nur Helices ausgew'ahlt, die in dem- oder denselben Bereichen lagen wie die zugeh'origen realisierten Helices.
- Die 'Ähnlichkeit bez'uglich der L'ange: Es wurden nur Helices ausgew'ahlt, deren L'angen in gewissen, durch die realisierten Helices festgelegten Grenzen lagen.
- Die 'Ähnlichkeit bez'uglich der durchschnittlichen Bindungsst'arke: Es wurden nur Helices ausgew'ahlt, die eine minimale, durch die realisierten Helices

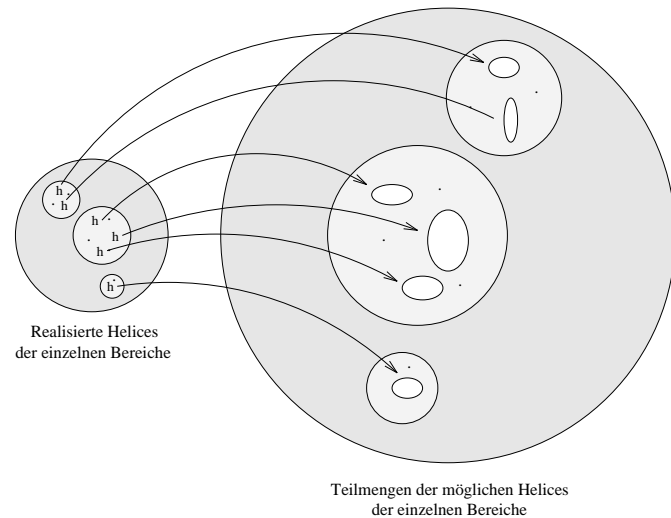


Abbildung 3.17: Zuordnung der realisierten Helices jedes Bereiches zu einer Teilmenge der möglichen Helices dieses Bereiches

bestimmte durchschnittliche Bindungsstärke aufwiesen.

Zwei dieser Kriterien bieten sich unmittelbar für den Vergleich von Helices an: die Länge und die mittlere Bindungsstärke. Jeder realisierten Helix werden nur solche möglichen Helices zugeordnet, deren Längen und mittleren Bindungsstärken nur um gewisse Prozentsätze von ihrer eigenen abweichen. Man erzeugt für jede realisierte Helix eine Art Fenster, wie in Abbildung 3.18 dargestellt, und ordnet ihr alle die möglichen Helices zu, die in diesem Fenster auftauchen.

Die so für die realisierten Helices entstandenen Teilmengen von möglichen Helices des Bereiches können disjunkt sein, wie es in Abbildung 3.17 der Fall ist, sind es aber höchst wahrscheinlich nicht. Wenn sich die realisierten Helices nicht gravierend in ihren Längen und mittleren Bindungsstärken unterscheiden, werden sicherlich einige der möglichen Helices mehreren realisierten Helices zugeordnet, was unter anderem für die Konfliktsituationen, die in Kapitel 3.3.1 besprochen werden, verantwortlich ist. Wie bereits erwähnt, sollen die Elemente der Teilmengen bezüglich ihrer Ähnlichkeit zur zugehörigen realisierten Helix sortiert werden können. Man benötigt also nicht nur ein Verfahren, das aufgrund hinreichender Ähnlichkeit, wie gerade dargelegt, entscheiden kann, sondern auch ein Maß für die

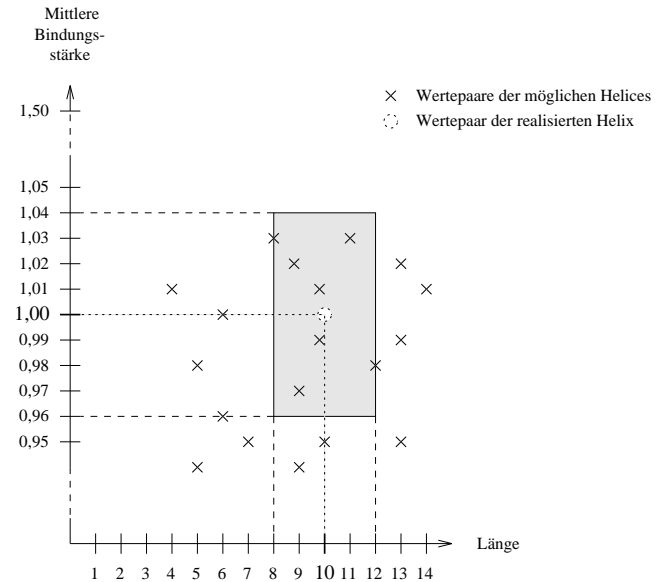


Abbildung 3.18: Auswahl möglicher Helices aufgrund ihrer Ähnlichkeit bezüglich ihrer Längen und durchschnittlichen Bindungsstärken zu einer realisierten Helix

Ähnlichkeit. Da die Auswahl der möglichen Helices durch ein geometrisches Verfahren realisiert wurde, bietet es sich an, aus den beiden beteiligten Parametern, den Längen und den mittleren Bindungsstärken der Helices, eine Ähnlichkeitsdistanz nach dem Satz von Pythagoras zu errechnen. Sei Δl der Längenunterschied und Δw der Unterschied der mittleren Bindungsstärken einer realisierten Helix r und einer möglichen Helix m , dann ließe sich die Ähnlichkeitsdistanz A wie folgt berechnen:

$$A = \sqrt{\Delta^2 l + \Delta^2 w} \quad \text{mit } l \in N \quad \text{und } w \in R \quad (3.7)$$

Am Beispiel aus Abbildung 3.18 wird deutlich, daß diese Art der Berechnung zwar unmittelbar einleuchtend ist, aber aufgrund der stark unterschiedlichen Wertebereiche der beiden Parameter die Unterschiede in der Länge wesentlich stärker ins Gewicht fallen als die Unterschiede der mittleren Bindungsstärken. Die im Beispiel gewählten Werte sind durchaus realistisch, da sich die Unterschiede in den Längen der Helices in der Größenordnung 10^0 und die Unterschiede der mittleren Bindungsstärken sich in der Größenordnung 10^2 bewegen. Eine Normierung dieser Werte wird somit notwendig. Die maximale mittlere Bindungsstärke einer

Helix ergäbe sich dann, wenn sie sich nur aus G-C Bindungen zusammensetzen würde, da dies die stärkste Basenbindung ist. Folglich sollte die Differenz der mittleren Basenbindungen zweier Helices an diesem Wert normiert werden. Für die Differenz der Längen muß ein Wert gefunden werden, der sich auf alle Helices eines Bereiches anwenden läßt. Sinnvollerweise wählt man hier die Differenz zwischen der längsten und der kürzesten realisierten Helix. Sei B_{GC} die Stärke der Bindung zwischen den Basen G und C und l_{max} und l_{min} die Längen der längsten und kürzesten Helices des Bereiches, so stellt sich die Ähnlichkeitsdistanz aus 3.7 nun wie folgt dar:

$$A = \sqrt{\left(\frac{\Delta l}{B_{GC}}\right)^2 + \left(\frac{\Delta w}{l_{max} - l_{min}}\right)^2} \quad \text{mit } l \in \mathbb{N} \quad \text{und } w \in \mathbb{R} \quad (3.8)$$

Auf diese Weise erhält man eine Maßzahl für die Ähnlichkeit zweier Helices. Sollte sich aus empirischen Analysen ergeben, daß der eine oder andere Parameter eine stärkere Rolle spielt, kann noch eine zusätzliche Gewichtung der Parameter durchgeführt werden. Auch für den Fall, daß noch andere Faktoren bei der Bewertung der Ähnlichkeit berücksichtigt werden sollen, können diese, geeignet normiert und gegebenenfalls auch gewichtet, als weiterer Summand in die Formel 3.8 eingefügt werden. Solche weiteren Faktoren könnten z.B. die Differenzen der Positionen der öffnenden und schließenden Helixhälfte sowie die Differenz der Anzahl der Fullzeichen in den Helixhälfen sein. Ob die zusätzliche Verwendung solcher Faktoren sinnvoll ist, hängt von der Sequenz ab, die ausgerichtet werden soll. Handelt es sich dabei um eine gerade neu erfaßte Sequenz, ist dies sicherlich nicht sinnvoll, da sie in ihren unausgerichteten Bereichen noch keine Fullzeichen enthält und somit auch die Positionen der Helices noch völlig ohne Aussagekraft sind. Handelt es sich dagegen um eine vielleicht mit einem anderen Verfahren ausgerichtete Sequenz des Alignments, deren Ausrichtung überprüft werden soll, ist die Hinzunahme dieser Faktoren durchaus sinnvoll, da so Unstimmigkeiten aufgezeigt werden können.

Nachdem die Auswahl und Bewertung der ähnlichsten möglichen Helices jeder realisierten Helix eines Bereiches abgeschlossen ist, sollte jetzt der letzte Schritt, die Auswahl je einer möglichen Helix, so erfolgen, daß die neue Helixstruktur widerspruchsfrei und der alten möglichst ähnlich ist. Problematisch daran ist, daß zum einen die Mengen der zugeordneten möglichen Helices nicht disjunkt sind, es also zur Auswahl ein und derselben möglichen Helix bei verschiedenen realisierten Helices kommen kann, zum anderen, daß die Bereichsabhängigkeiten, wie in Kapitel 3.2.2 beschrieben, berücksichtigt werden müssen. Darüber hinaus können noch andere Konfliktsituationen entstehen, die aber in Kapitel 3.3.1 eingehend besprochen werden. Es erscheint einfacher, diese Probleme für alle realisierten Helices gleichzeitig zu lösen, als sie bereichsbezogen zu betrachten, weswegen auf die Unterteilung in Bereiche ab jetzt verzichtet wird. Die realisierten Helices werden, wie in Abbildung 3.19 dargestellt, mit den ihnen zugeordneten möglichen Helices als Folge betrachtet. Die Notwendigkeit einer Unterteilung ist jetzt auch nicht mehr gegeben, da die Suche, deren Beschleunigung der Hauptgrund für die Untertei-

lung war, abgeschlossen ist und sie sich für die weitere Bearbeitung als hinderlich erwiesen hat.

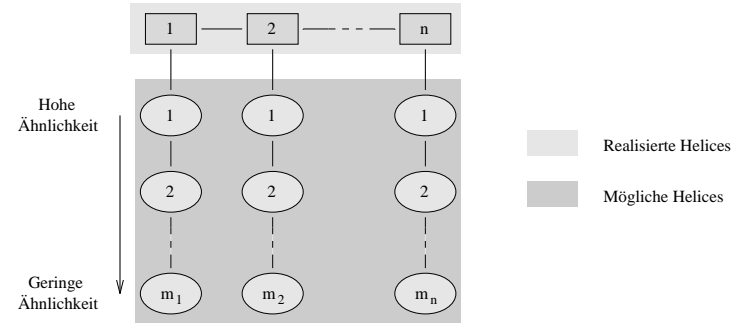


Abbildung 3.19: Zuordnung der ähnlichsten möglichen Helices zu den realisierten Helices nach Aufhebung der Bereiche

3.3 Suche nach dem optimalen Set der Strukturelemente

Das Ergebnis der vorangegangenen Analyse- und Auswahl Schritte sind geordnete Mengen möglicher Helices einer Sequenz, die je einer realisierten Helix ihrer Sequenzfamilie zugeordnet wurden. Die Mächtigkeiten dieser Mengen sind nicht konstant, sondern von der zugeordneten realisierten Helix abhängig, weshalb eine Matrixdarstellung ungeeignet ist. Das Ziel dieser Arbeit ist es, jeder der realisierten Helices eine mögliche Helix aus den zu ihnen gehörenden Helixmengen so zuzuordnen, daß sie vor allem zwei Bedingungen genügen:

- Die Helices dürfen sich nicht überschneiden.
- Die Reihenfolge der öffnenden und schließenden Helixhälfen, die durch die Abfolge der realisierten Helices bestimmt ist, muß erhalten bleiben.

Die Abfolge der realisierten Helices ist genau die, in der sie in den Sequenzen der Familie angeordnet sind und die durch die Helixstruktur des Alignments repräsentiert wird. Unter einem Set von Strukturelementen oder einem Helix-Set versteht man eine geordnete Menge von möglichen Helices. Besteht die Helixstruktur aus n realisierten Helices, so hat auch diese Menge n Elemente. Jedes Element des Sets rekrutiert sich dabei aus einer Menge der möglichen Helices genau so, daß die

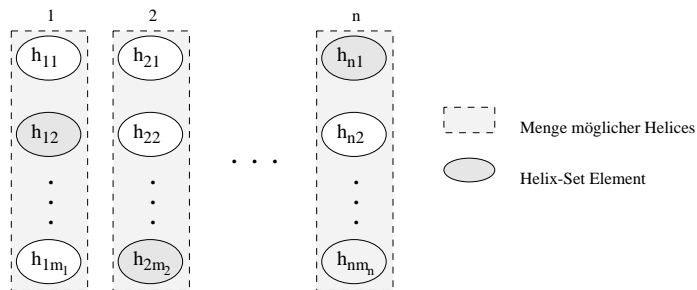


Abbildung 3.20: Ein Helix-Set ist die geordnete Menge von Helices aus den Mengen möglicher Helices

Ordnung der realisierten Helices beibehalten wird. In Abbildung 3.20 wird dieser Sachverhalt noch einmal verdeutlicht.

Enthält jede Menge M_i mit $i = 1, \dots, n$ der realisierten Helices m_i Elemente, so beläuft sich die Anzahl A der möglichen Helix-Sets auf

$$A = \prod_{i=1}^n m_i. \quad (3.9)$$

Es gilt nun das Helix-Set herauszufinden, das die oben genannten Bedingungen erfüllt und sich dabei aus den, den realisierten Helices ähnlichsten möglichen Helices zusammensetzt. Um dies zu erreichen und um ein Maß für die Ähnlichkeit eines Helix-Sets zu der Helixstruktur des Alignments bereitzustellen, wurden die Mengen der möglichen Helices nach ihrer Ähnlichkeit zu der zu ihnen gehörenden realisierten Helices sortiert. Die Summe über den Indizes der Helices eines Helix-Sets kann jetzt zum Vergleich herangezogen werden. Je niedriger diese Summe ist, desto besser ist das Set. In den folgenden beiden Kapiteln soll dargelegt werden, welche Arten von Konflikten zwischen den Helices eines Sets als Folge der beiden, oben erwähnten Bedingungen entstehen können und mit welchen Verfahren man diese Konflikte auflösen kann.

3.3.1 Konflikte und Qualität von Sets

Bei der Suche und Auswahl der möglichen Helices wurden Verfahren eingesetzt, die zu einer maximalen Ähnlichkeit mit den realisierten Helices führen sollten, wobei aber keine Rücksicht auf die Abhängigkeiten der Helices untereinander genommen wurde. Das muß jetzt nachgeholt werden. Bei der Auswahl der Elemente eines Helix-Sets soll erreicht werden, daß sich keine der Helices überschneiden und

3.3. SUCHE NACH DEM OPTIMALEN SET DER STRUKTURELEMENTE 53

es zur genauen Einhaltung der Abfolge der Helixhälften, wie sie die Helixstruktur des Alignments festlegt, kommt. Durch die Auswahl der Elemente ist die Zuordnung zu den realisierten Helices implizit festgelegt (s. 3.3), wobei eine optimale Zuordnung, wie in Abbildung 3.21 dargestellt, ablaufen sollte.

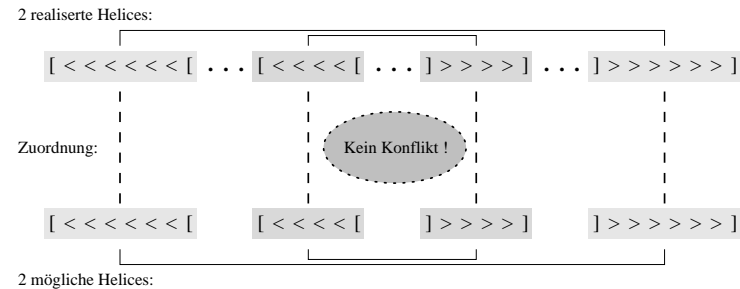


Abbildung 3.21: Konfliktfreie Zuordnung realisierter und möglicher Helices

Durch die Vorgehensweise bei der Auswahl der möglichen Helices und deren Zusammenfassung zu Mengen, die wiederum je einer realisierten Helix zugeordnet wurden, ist die Wahrscheinlichkeit einer Doppelzuordnung relativ hoch. Die Mengen der möglichen Helices sind also nicht notwendigerweise disjunkt. Das ist einer der Gründe, weshalb es zu Überlappungen der Helices, wie in Abbildung 3.22, kommen kann. Ein anderer Grund liegt in den unterschiedlichen Längen der Helices. In einer langen Helix sind zugleich mehrere kürzere Helices enthalten, die sich aus paarenden Teilbereichen der Helixhälften zusammensetzen.

Obwohl die Suche nach den möglichen Helices in getrennten Bereichen der Sequenz vorgenommen wurde, kann es, aufgrund der Auswahl und Zuordnung der Helices zu den realisierten Helices zu Unstimmigkeiten in der Reihenfolge der Helices kommen, wie Abbildung 3.23 verdeutlicht. Auch wenn diese Auswahl nur die ähnlichsten der möglichen Helices berücksichtigte, können derartige Konflikte nicht ausgeschlossen werden.

Das Ziel bei der Auswahl der Elemente eines Helix-Sets ist es, die Zahl von Konflikten zu minimieren und zugleich die Ähnlichkeit zu den realisierten Helices zu maximieren. Damit sind die beiden Qualitätsmerkmale eines Helix-Sets bestimmt:

- Anzahl der Konflikte zwischen den Helices und
- Ähnlichkeit zu den realisierten Helices.

Arbeitet man die Helices des Sets der Reihe nach ab und überprüft dabei jede Helix auf die Konflikte, die gerade beschrieben wurden, mit jeder ihrer Vorgängerinnen,

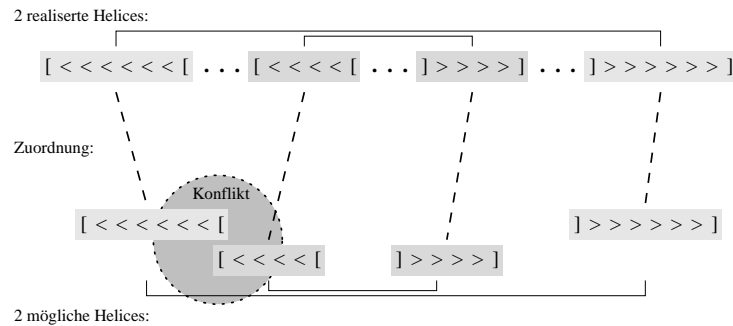


Abbildung 3.22: Zuordnung realisierter und möglicher Helices mit Konflikt durch Überlappung

so erhält man die Anzahl der Konflikte eines Sets. Je niedriger diese Zahl, desto besser ist das Set. Die Minimierung der Zahl der Konflikte ist das vorrangige Ziel. Allerdings kann es zu Situationen kommen, in denen, unabhängig von der Wahl der Set-Elemente, ein Konflikt nicht aufgelöst werden kann. Wie dann zu verfahren ist und wodurch diese Situationen entstehen können wird in Kapitel 3.3.2 genauer erörtert. Die Bewertung der Ähnlichkeit der Set-Elemente mit den realisierten Helices wird durch die Ordnung der Mengen möglicher Helices gewährleistet. Sie sind, wie in Kapitel 3.3 bereits erwähnt, nach der Ähnlichkeit zu den zugeordneten realisierten Helices sortiert, weshalb der Index eines Mengenelements dem Grad seiner Ähnlichkeit entspricht. Somit kann die Summe der Indizes aller Set-Elemente als Maß für die Ähnlichkeit des Sets zu der geordneten Menge der realisierten Helices dienen. Genau wie bei der Zahl der Konflikte ist hier ein niedriger Wert besser als ein hoher. Wie schon kurz angesprochen, hat die Zahl der Konflikte einen höheren Stellenwert als die Ähnlichkeit des Sets. Erst bei gleicher Zahl von Konflikten wird das Set mit der größeren Ähnlichkeit, also der niedrigeren Summe über die Indizes der Elemente, bevorzugt. Die Auflösung der Konflikte, wie sie im folgenden Kapitel besprochen wird, ist also vorrangig.

3.3.2 Auflösung der Konflikte

Aus allen Helix-Sets, die sich aus den Mengen der möglichen Helices bilden lassen, soll dasjenige herausgesucht werden, das, sofern überhaupt möglich, konfliktfrei ist und dabei die höchste Ähnlichkeit zur Helixstruktur des Alignments aufweist. Wie die Zahl der Konflikte und der Grad der Ähnlichkeit ermittelt werden kann, wurde im vorangegangenen Kapitel erläutert. Um das Problem zu lösen,

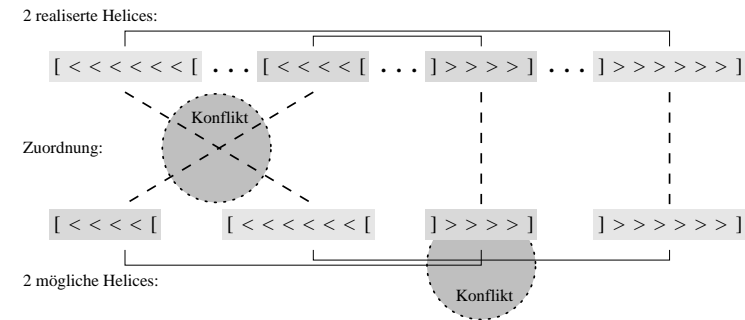


Abbildung 3.23: Zuordnung realisierter und möglicher Helices mit Konflikt in der Reihenfolge der Helixh"alten Überschneidung)

könnte man zuerst alle konfliktfreien Sets, oder zumindest die mit der geringsten Anzahl an Konflikten, ermitteln und dann aus diesen das Set mit der höchsten Ähnlichkeit herausuchen. Auf diese Weise wäre sichergestellt, daß die optimale Lösung auf jeden Fall gefunden wird. Ein Algorithmus, der dies leistet, wird im folgenden vorgestellt. Zur Darstellung des Suchraums wird jetzt doch auf eine Matrix zurückgegriffen, allerdings auf eine nicht vollständig besetzte, da die Mengen der möglichen Helices unterschiedlich viele Elemente enthalten können, so daß in jeder Spalte die letzten Elemente unbesetzt bleiben können. Die Matrixspalten repräsentieren dabei die Mengen der möglichen Helices in der Reihenfolge der realisierten Helices. Hat die geordnete Menge der realisierten Helices eine Mächtigkeit von n und die jeder realisierten Helix zugeordnete geordnete Menge möglicher Helices die Mächtigkeit m_i , dann besteht die Matrix M aus $m = \max(m_i)$ Zeilen und n Spalten. Gesucht werden nun alle Pfade durch diese Matrix, wie in Abbildung 3.24 dargestellt, beginnend in der ersten und endend in der n -ten Spalte, deren Kanten alle von einer Spalte in die darauffolgende führen und deren so bestimmte Helices untereinander konfliktfrei sind.

Dies kann durch einen rekursiven Algorithmus erreicht werden, der in jeder Rekursionsstufe der Reihe nach alle Elemente einer Spalte daraufhin überprüft, ob sie den bisherigen Pfad fortsetzen können. Dazu müssen die zugehörigen Helices auf Konflikte mit allen durch den bisherigen Pfad bereits festgelegten Helices überprüft werden. Tritt kein Konflikt auf, kann der Pfad fortgesetzt werden und dem Feld der Index seines Vorgängerfeldes in der vorangegangenen Spalte hinzugefügt werden. Gleichzeitig kann durch Summieren der Indizes der Grad der Ähnlichkeit des durch den Pfad bestimmten Sets ermittelt werden. Auf diese Weise können alle konfliktfreien Pfade durch die Matrix gefunden und zugleich bewertet werden,

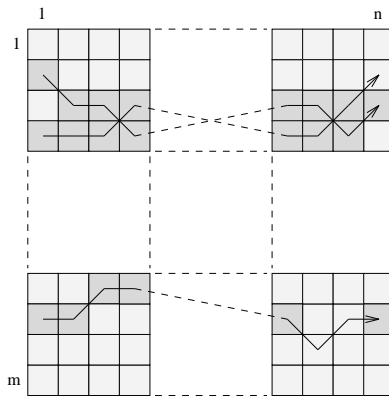


Abbildung 3.24: Konfliktfreie Pfade durch die Matrix der möglichen Helices

so daß am Schluß nur noch der Pfad mit der höchsten Ähnlichkeit herausgesucht werden muß.

Allerdings können auch Situationen auftreten, in denen ein Konflikt nicht aufgelöst werden kann. In diesem Falle gäbe es keinen vollständigen Pfad durch die Matrix und das eben beschriebene Verfahren wäre gescheitert. Solche Situationen entstehen meistens bei benachbarten realisierten Helices, deren zugeordneten Mengen möglicher Helices nur sehr wenige Elemente enthalten. Sie stammen meistens aus demselben Bereich, und bei der Auswahl der möglichen Helices wurden, aufgrund der Ähnlichkeiten zwischen diesen beiden Helices, die möglichen so ausgewählt, daß sich jetzt keine konfliktfreie Kombination finden läßt, wie das Beispiel in Abbildung 3.25 verdeutlichen soll.

Für solche Fälle muß der Algorithmus so abgeändert werden, daß er die Spalte, die den Pfad unterbrechen würde, überspringt. Die Anzahl, wie oft dies innerhalb eines Pfades geschehen darf, sollte genügend klein gewählt werden, da solche Auslassungen für die Helixstruktur bedeuten, daß nicht alle Helices zugeordnet werden können.

Der beschriebene Algorithmus ermittelt zwar die optimale Lösung, eine Abschätzung seiner Komplexität ist aber doch ratsam. Zu diesem Zweck wird der Durchschnitt m der Elemente der Mengen der möglichen Helices ermittelt, um eine einheitliche Matrix, wie in Abbildung 3.26 dargestellt, bilden zu können. In jeder Spalte gibt es dann m Möglichkeiten, den Pfad fortzusetzen, insgesamt m^n mögliche Pfade. Für jeden dieser Pfade müssen zusätzlich noch $\frac{(n-1)n}{2}$ Vergleiche bezüglich der

3.3. SUCHE NACH DEM OPTIMALEN SET DER STRUKTURELEMENTE 57

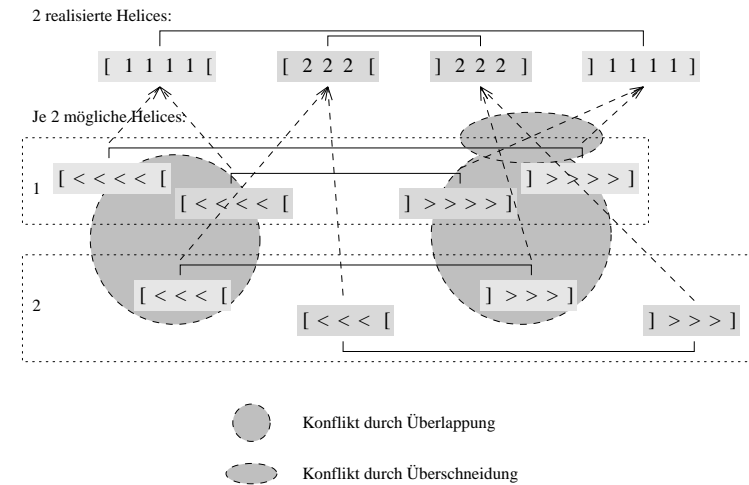


Abbildung 3.25: Beispiel für einen nicht auflösbaren Konflikt zwischen zwei Mengen möglicher Helices

Konflikte durchgeführt werden, also insgesamt $\frac{(n-1)n}{2} m^n$ Vergleichsoperationen. Damit hat allein die Ermittlung der Konflikte eines einzelnen Pfades eine Komplexität von $O(n^2)$, was aber nicht mehr weiter ins Gewicht fällt, da die Anzahl der möglichen Pfade der Problematik eine exponentielle Komplexität von $O(m^n)$ verleiht.

Das ist eine denkbar schlechte Ausgangssituation. Bei einer 16S rRNS Sequenz sind etwa 50 Helices zuzuordnen. Selbst wenn jede Menge der möglichen Helices im Schnitt nur zwei Elemente enthalten würde und jeder Vergleich in einer Nano-Sekunde durchgeführt werden könnte, würde die Berechnung aller Pfade immer noch etwa 36 Jahre benötigen. Deshalb muß man auf andere Verfahren ausweichen, die es ermöglichen, den Zeitaufwand solcher Probleme drastisch zu senken. Im allgemeinen ist dafür der Preis der optimalen Lösung zu zahlen. Es hat sich herausgestellt, das nicht deterministische Verfahren wie Mutations-Selektions-Verfahren oder genetische Algorithmen solche Probleme in annehmbarer Zeit lösen können, aber nicht unbedingt das optimale Ergebnis liefern. Solange die Abweichungen von der optimalen Lösung innerhalb akzeptabler Grenzen liegt, ist diese Einschränkung, angesichts des Zeitaufwands bei der Verwendung deterministischer Verfahren, vertretbar. Im Implementierungsteil wurde ein Threshold Accepting genannter Mutations-Selektions-Algorithmus verwendet, wie er in [WKin] be-

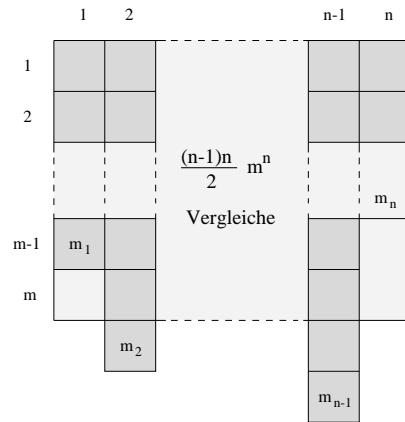


Abbildung 3.26: Anzahl der Vergleichsoperationen für alle Pfade

beschrieben wird. Eine genau Beschreibung des Algorithmus wird in Kapitel 4 vorgenommen. Auf den in diesem Kapitel beschriebenen deterministischen Algorithmus wird erst dann zurückgegriffen, wenn Konflikte auftreten, die durch Mutation und Selektion nicht aufzulösen sind, und die Anzahl der Konflikte in einem vertretbaren Rahmen liegt.

Kapitel 4

Algorithmen zum sekundärstrukturbasierten Alignment

Das in der vorliegenden Arbeit entwickelte Verfahren zum sekundärstrukturbasierten Alignment von rRNS-Sequenzen stützt sich auf zwei grundlegenden Algorithmen ab. Der eine soll ein effizientes Auffinden von komplementären Teilsequenzen ermöglichen, der andere eine möglichst gute Auswahl solcher Teilsequenzen treffen. Warum diese Teilsequenzen gefunden werden müssen, wozu eine Auswahl notwendig ist und wie deren Qualität definiert ist, kann in Kapitel 3 nachgelesen werden. In diesem Kapitel soll darauf eingegangen werden, welche Merkmale diese Algorithmen haben müssen, welche Alternativen zur Wahl standen, warum die verwendeten ausgewählt wurden und schließlich, wie sie funktionieren.

Durch den konstruktiven Charakter des vorgestellten Verfahrens werden ein Such- und ein Auswahlalgorithmus zu seinen grundlegenden Komponenten. Die gesuchte Lösung, die Sekundärstruktur einer rRNS-Sequenz, setzt sich aus vielen Strukturelementen, den Helices, zusammen. Diese wiederum müssen selbst gewissen Strukturanforderungen, den Basenpaarungen, genügen. Das Verfahren, sehr vereinfacht in Abbildung 4.1 dargestellt, stellt deshalb erst einmal eine große Menge solcher Strukturelemente zur Verfügung. Dazu wird ein Algorithmus benötigt, der alle sinnvollen Strukturelemente in der Sequenz finden kann: der Suchalgorithmus. Aus dieser Menge wird dann eine Teilmenge herausgesucht, deren Elemente zusammen den Anforderung durch die Sekundärstruktur am besten entsprechen. Dazu wiederum wird ein optimierender Auswahlalgorithmus benötigt. Obwohl das eine sehr einfache Beschreibung des Verfahrens ist und natürlich noch eine ganze Reihe anderer Algorithmen verwendet wurden, kann so doch gezeigt werden, warum genau diese beiden Algorithmen eine so exponierte Stellung einnehmen.

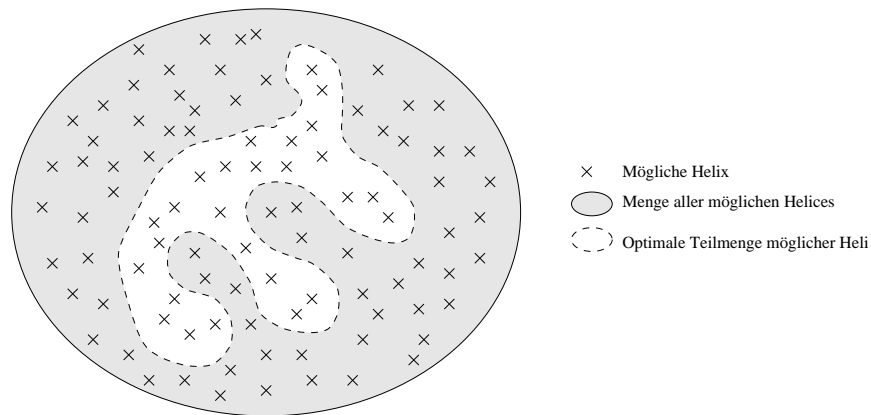


Abbildung 4.1: Nach der Suche nach allen möglichen Helices wird die optimale Teilmenge, die Helixstruktur ausgewählt

4.1 Suche nach möglichen Strukturelementen

Die Suche nach allen Helices, die in einer rRNS-Sequenz denkbar sind, ist der erste Schritt bei der Suche nach der Sekundärstruktur dieser Sequenz. Denkbar sind allerdings sehr viele. Sucht man zum Beispiel nur sehr kurze Helices, etwa der Länge 1, so wird man wahrscheinlich in einer Sequenz der Länge n $\frac{(n-1)n}{2}$ solcher Helices finden, da jede Base im Mittel mit zwei anderen paart und in einer rRNS-Sequenz nur vier unterschiedliche Basen auftreten. Gesucht werden aber nicht nur Helices einer bestimmten Länge, sondern aller möglichen Längen. Deshalb müssen Maßnahmen ergriffen werden, die Zahl der Möglichkeiten einzuschränken. Zum einen geschieht dies eben genau über die Länge. Mithilfe der in den Familiensequenzen realisierten Helices kann die Länge der gesuchten Helices eingeschränkt werden. Allerdings kann man in einer Sequenz der Länge 1000, wie in Kapitel 3.2 berechnet, bis zu 7509 Helices der Längen 7 – 12 finden. Läßt man dabei noch einen 'mismatch' zu, steigt diese Zahl um das 17-fache an. Weitere Schritte sind also notwendig. Durch ein primärstrukturbasiertes Prealignment wird die Sequenz in ausgerichtete und unausgerichtete Bereiche unterteilt. Die Suche muß dann nur noch in den unausgerichteten Bereichen durchgeführt werden, die meistens deutlich kürzer sind als die ausgerichteten. Eine weitere Einschränkung kann über die mittlere Bindungsstärke der in den Familiensequenzen realisierten Helices geschehen. Über sie kann man ein Minimum für die mittlere Bindungsstärke der zu suchenden Helices festlegen. Allerdings ist das ein Kriterium, das unter Umständen

schon bei der Implementierung des Suchalgorithmus berücksichtigt werden muß, weswegen im folgenden auf die möglichen Algorithmen eingegangen werden soll.

4.1.1 Anforderungen an den Suchalgorithmus

Der Suchalgorithmus ist ein zentraler Punkt des beschriebenen Verfahrens zum sekundärstrukturbasierten Alignment von rRNS-Sequenzen. Aufgrund der großen Vielfalt der möglichen Helices (s. 4.1 und 3.2) ist anzunehmen, dass die Suche nach ihnen sehr oft durchgeführt werden muß, weswegen ein hoch effizienter Suchalgorithmus eingesetzt werden sollte. Um eine Auswahl an Suchalgorithmen zu erhalten, müssen zunächst einmal die Voraussetzungen für die Suche geklärt werden. Anhand der Anforderungen, die diese Problematik an den Algorithmus stellt, kann dann der geeignetste ausgesucht werden.

Der Gegenstand der Suche sind Helices. Da sie aus zwei Teilen, der öffnenden und der schließenden Helixhälfte bestehen, die durch die Bindungen ihrer Basen miteinander verbunden sind, kann die eine als Muster zur Suche nach der anderen verwendet werden. Dabei können Füllzeichen, nicht exakt spezifizierte Basen und nicht sequenzierte Bereiche vernachlässigt werden. Die Muster sind also Zeichenfolgen der Länge m über dem Alphabet $\mathcal{B} = (\mathcal{B} \setminus \{-, \cdot\})^*$. Der Suchraum dagegen ist ein Teilbereich einer rRNS-Sequenz, die entweder neu sequenziert, oder bereits (zumindest teilweise) ausgerichtet im Alignment vorliegt. Es handelt sich also um einen Textabschnitt der Länge n mit $m < n$ über dem Alphabet \mathcal{B}^* . Somit würden sich prinzipiell alle Algorithmen, die eine Suche nach Zeichenfolgen in Texten effizient realisieren, für dieses Problem eignen. Allerdings stellt es einige Anforderungen, die nicht jeder Algorithmus erfüllen kann.

Die einfachste Anforderung an den Suchalgorithmus ist, eine Suche nach Zeichenfolgen unterschiedlicher Länge zu ermöglichen. Dies scheint selbstverständlich zu sein, kann aber z.B. bei Hash-Tabellen zu Problemen führen. Da die Muster der Reihe nach aus der Sequenz erzeugt werden, wie in Abbildung 4.2 und somit die kürzeren Muster ab einer bestimmten Position in dem längsten Muster ab dieser Position enthalten sind, wäre es angenehm, wenn dem Suchalgorithmus nur jeweils das längste Muster übergeben werden müßte, d.h. wenn er so funktionieren würde, daß er bei der Suche nach langen Zeichenfolgen die darin enthaltenen kürzeren Zeichenfolgen ebenfalls findet.

Ebenfalls mit der Mustergenerierung hängt der nächste Punkt zusammen. Jede Base hat die Eigenschaft, mit mehreren anderen Basen Bindungen eingehen zu können. Für ein Muster heißt das, daß es dazu nicht genau eine komplementäre Zeichenfolge gibt, sondern viele (s. auch Abbildung 4.3). Da jede Base im Mittel mit zwei anderen Basen paart, ergibt sich somit für ein Muster der Länge m ein theoretische Anzahl von 2^m paarender Gegenstücke.

Eine weitere Anforderung an den Suchalgorithmus ist, daß er eine gewisse Anzahl

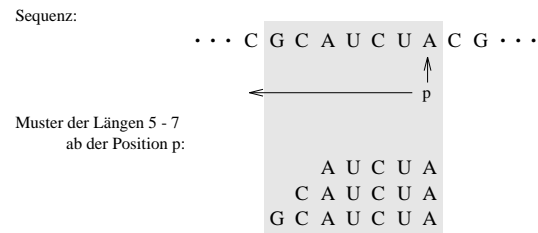


Abbildung 4.2: Erzeugung der Suchmuster aus der Sequenz, wobei rückwärts vorgangen wird, da sich Helices aus zwei antiparallelen, komplementären Teilen zusammensetzen



Abbildung 4.3: Zu jedem Muster kann es bis zu 2^m paarender Gegenstücke geben

k von 'mismatches' mit $k < m$ zulassen soll. Da eine Abbildung der realisierten Helices der Sequenzfamilie auf eine Teilmenge der möglichen Helices der Sequenz angestrebt wird, muß bei der Suche berücksichtigt werden, daß die Helices sich durch Mutation sehr wahrscheinlich verändert haben. Einige mögliche Paarungen mit einem 'mismatch' an genau einer Position sind in Abbildung 4.4 dargestellt.

Die Anforderungen an den Suchalgorithmus sind somit klargestellt. Im folgenden Abschnitt werden einige Algorithmen untersucht und auf ihre Eignung hin bewertet, weswegen die Anforderung hier noch einmal kurz zusammengefaßt werden:

- Der Algorithmus soll sich für die Suche nach Zeichenfolgen eignen.
- Er soll Zeichenfolgen unterschiedlicher Längen finden können.
- Er soll die in einem Muster implizit beinhalteten kürzeren Muster ebenfalls suchen können.
- Er soll die Suche mit 'mismatches' erlauben.
- Er soll die Tatsache, daß immer im gleichen Text gesucht wird, für eine effiziente Suche ausnutzen.

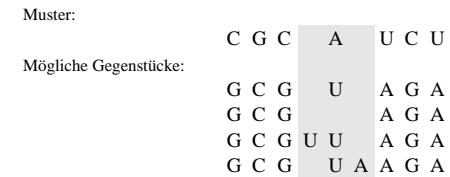


Abbildung 4.4: Mögliche Formen eines 'mismatch' an einer Position im Muster

Der letzte Punkt wurde bis jetzt noch nicht weiter besprochen. Gedacht wurde dabei aber an Algorithmen, die den Text erst geeignet bearbeiten, z.B. in einer bestimmten Form codieren, wie es der Huffman-Algorithmus [RSed] tut, um später eine effiziente Suche zu ermöglichen.

4.1.2 Sammlung und Diskussion verschiedener Alternativen

Nachdem die Voraussetzungen und die Anforderungen an den Suchalgorithmus geklärt wurden, soll jetzt anhand von drei Beispielen deren Eignung für das gestellte Suchproblem überprüft werden. Es handelt sich dabei um Pattern Matching mittels regulärer Ausdrücke, Hashing und um einen Positionsbaum. Allerdings wird dabei nicht deren Arbeitsweisen beschrieben, d.h. nur insoweit, als es zur ihrer Bewertung notwendig ist, aber es wird zu jedem ein Literaturverweis angegeben, in dem die Funktionsweise des Algorithmus nachgeschlagen werden kann.

- Pattern Matching mittels regulärer Ausdrücke ([JHop] und [RSed]):
Reguläre Ausdrücke wurden speziell für die Suche nach ähnlichen Zeichenfolgen entwickelt, weswegen sie sich natürlich für eine Suche nach Zeichenfolgen eignen. Durch ihre Beschreibungssprache lassen sich sowohl unterschiedlich lange Suchmuster definieren, als auch Suchmuster, bei denen sich die durch sie gefundenen Zeichenfolgen in der Länge unterscheiden können. Ebenfalls leicht zu realisieren sind Suchmuster, die gewisse 'mismatches' zulassen. Ist allerdings z.B. nur ein 'mismatch' gewünscht, so müssen mehrere Suchmuster erzeugt werden, die die Beschreibung des 'mismatches' jeweils an einer anderen Position tragen. Die implizite Suche nach Präfixen eines Musters ist zwar prinzipiell möglich, in diesem speziellen Fall aber schwierig, da solche Muster auch als 'mismatches' aufgefaßt werden können. Da reguläre Ausdrücke den Text zeichenweise abarbeiten, können sie keinen Vorteil aus wiederholten Suchvorgängen über demselben Text ziehen.
- Hashing ([RSed] und [AAho]):

Hash-Tabellen eignen sich für alle Arten von Daten, solange sich ihre Schlüssel als Zahl darstellen lassen und eine geeignete Hash-Funktion gefunden werden kann, also auch für Zeichenketten. Hash-Tabellen wurden speziell für wiederkehrende Suchoperationen über demselben Datensatz entwickelt und sollten sich deshalb sehr gut für das vorliegende Aufgabenstellung eignen. Allerdings müssen die Schlüssel einer Hash-Tabelle alle gleich lang sein. Da hier die Zeichenketten selber als Schlüssel dienen, ließe sich eine Suche nach unterschiedlich langen Zeichenketten nur über mehrere Hash-Tabellen realisieren. Das Suchen nach impliziten Mustern müsste dann aber in mehreren Hash-Tabellen geschehen und die Suche mit 'mismatches' wird ganz unmöglich.

- Positionsbaum ([MKem] und [RSed]):

Ein Positionsbaum stellt alle unterschiedlichen Teilsequenzen einer Sequenz, also auch einer Zeichenkette dar. Dadurch eignet er sich besonders für wiederkehrende Suchoperationen. Die Möglichkeit der Suche nach Mustern unterschiedlicher Länge und nach impliziten Mustern ergibt sich direkt aus der Baumstruktur. Die Suche mit 'mismatches' erfordert zwar etwas Raffinement bei der Implementierung der Suchfunktion, ist aber durchaus möglich.

Eine vergleichende Bewertung der drei Algorithmen ist in der nachfolgenden Tabelle abgebildet.

	Reguläre Ausdrücke	Hashing	Positionsbaum
Suche in Zeichenfolgen	+	+	+
Unterschiedliche Längen	+	+	+
Implizite Muster	+	-	+
'mismatches'	+	-	+
Vorverarbeitung	-	+	+
Komplexität	$O(mn^2)$	$O(\lg n + 1)$	$O(mn)$

Dabei steht '+' für *ist möglich*, '-' für *ist nicht möglich* und '+-' für *ist möglich, aber aufwendig*. Die Komplexität bezieht sich dabei auf die Suche nach allen Mustern der Länge m in einer Sequenz der Länge n . Alles spricht damit für den Positionsbaum. Er ist der effizienteste der drei Algorithmen und wird auch als einziger allen Anforderungen gerecht. Im folgenden Kapitel wird deshalb der Positionsbaum als optimaler Suchalgorithmus ausführlich besprochen.

4.1.3 Der angepaßte Positionsbaum als geeignetster Algorithmus

In der vorliegenden Arbeit werden Positionsbaume über Sequenzabschnitten einer rRNS-Sequenz erzeugt. Dabei repräsentiert jeder Pfad von der Wurzel zu ei-

nem Blatt eine eindeutige Teilsequenz dieses Abschnitts. Das zugehörige Blatt repräsentiert die Position, an der diese Teilsequenz im Sequenzabschnitt beginnt. Alle Pfade zusammen ergeben die Menge aller eindeutigen Teilsequenzen dieses Abschnitts. Damit ist jedem Blatt genau eine Position aus dem Sequenzabschnitt zugeordnet. Jede Kante im Baum kann eindeutig der Menge $\mathcal{B} \cup \{\#\}$ zugeordnet werden. Die Einführung des Zeichens # ist notwendig, um das Ende des Sequenzabschnitts zu markieren, spielt aber bei der Suche nach Teilsequenzen keine Rolle. In Abbildung 4.5 ist ein solcher Baum dargestellt.

Position: $\lfloor 0 \rfloor \lfloor 1 \rfloor \lfloor 2 \rfloor \lfloor 3 \rfloor \lfloor 4 \rfloor \lfloor 5 \rfloor \lfloor 6 \rfloor \lfloor 7 \rfloor \lfloor 8 \rfloor \lfloor 9 \rfloor \lfloor 10 \rfloor \lfloor 11 \rfloor \dots$
 Sequenz: $\lfloor C \rfloor \lfloor G \rfloor \lfloor C \rfloor \lfloor A \rfloor \lfloor U \rfloor \lfloor C \rfloor \lfloor U \rfloor \lfloor G \rfloor \lfloor G \rfloor \lfloor U \rfloor \lfloor C \rfloor \lfloor A \rfloor \lfloor \# \rfloor$

Positionsbaum:

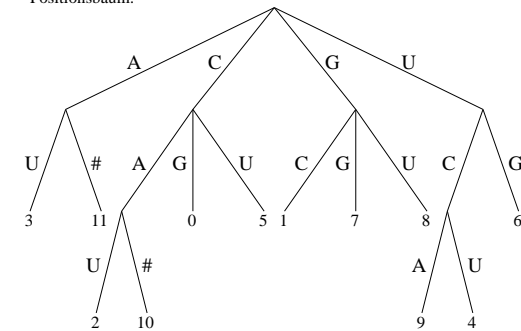


Abbildung 4.5: Ein Positionsbaum mit zusätzlicher Endmarkierung

In einem Sequenzabschnitt können auch Zeichen aus der Menge $\{-, n, .\}$ vorkommen. Das Füllzeichen '-' wird sowohl beim Baumaufbau als auch bei der Suche ignoriert. Die anderen beiden Zeichen für beliebige Basen und nicht sequenzierte Bereiche dienen gegebenenfalls als Endmarkierung einer Teilsequenz, kommen im Baum also nur als letzte Kante vor einem Blatt vor.

4.1.3.1 Aufbau des Positionsbaums

Ein Positionsbaum repräsentiert alle eindeutigen Teilsequenzen eines Sequenzabschnitts. Also muß man beim Aufbauen des Baums alle eindeutigen Teilsequenzen suchen. Dies kann man bewerkstelligen, indem man noch uneindeutige Teilsequenzen jeweils um das darauffolgende Zeichen erweitert, solange bis sie eindeutig sind. Ein rekursiver Algorithmus, der dies leistet, könnte wie folgt arbeiten:

1. Suche alle eindeutigen Basen des Sequenzabschnitts und merke alle ihre Positionen. Die gefundenen Basen seien ab jetzt Teilsequenzen der Länge 1 an den gemerkten Positionen.
2. Für alle Teilsequenzen, die an mehreren Positionen vorkommen, tue das folgende:
 - (a) Füge eine Kante mit dem letzten Zeichen der Teilsequenz ein.
 - (b) Erweitere die Teilsequenz an all ihren Positionen um 1 Zeichen.
 - (c) Merke alle Positionen dieser neuen Teilsequenzen und gehe zu 2.
3. Füge ein Blatt mit der Position der Teilsequenz ein.

In der folgenden Abbildung 4.6 werden die einzelnen Rekursionsschritte des Beispiels aus Abbildung 4.5 dargestellt.

Position: $_0\ _1\ _2\ _3\ _4\ _5\ _6\ _7\ _8\ _9\ _{10}\ _{11}\ _ _ _$
 Sequenz: $_C\ _G\ _C\ _A\ _U\ _C\ _U\ _G\ _G\ _U\ _C\ _A\ _\#$

Rekursiver Baumaufbau:

Rekursionsstufe r	Teilsequenz t mit t =r		Teilsequenz t mit t =r		Teilsequenz t mit t =r		Teilsequenz t mit t =r	
		Positionen		Positionen		Positionen		Positionen
1.	A	3 11	C	0 2 5 10	G	1 7 8	U	4 6 9
2.	A U	3	C A	2 10	G C	1	U C	4 9
	A #	11	C G	0	G G	7	U G	6
3.			C A U	2			U C A	9
			C A #	10			U C U	4

Abbildung 4.6: Rekursiver Aufbau eines Positionsbaums durch Suche nach eindeutigen Teilsequenzen

Der tatsächlich implementierte Positionsbaum wird prinzipiell genau so erzeugt, allerdings wurden aus Rücksicht auf die Speicherkapazität einige Anpassungen vorgenommen. So werden z.B. die Kanten komprimiert verwaltet und müssen über bestimmte Masken angesprochen werden. Diese Maßnahmen haben den Speicherbedarf auf etwa $\frac{1}{6}$ reduziert.

4.1.3.2 Suche im Positionsbaum

Nachdem der Positionsbaum eines Sequenzabschnitts erzeugt worden ist, kann mit der Suche nach den Teilsequenzen begonnen werden. Zur Erzeugung der Suchmuster sei auf Kapitel 4.1 verwiesen. Die Suchmuster liegen rückwärts vor, da die Helices sich aus zwei antiparallelen Hälften zusammensetzen, und enthalten nur Zeichen aus B . Zunächst einmal soll der Suchvorgang etwas vereinfacht, d.h. ohne Spezialfälle, sondern nur mit Mustern, die einen entsprechenden, gleichlangen Pfad im Baum haben, und vor allen Dingen ohne 'mismatches' dargestellt werden. Problematisch an der Suche ist die Tatsache, daß die Basen mit verschiedenen anderen Basen Bindungen eingehen können. Deshalb muß für jede Base des Musters geprüft werden, wie viele Kanten im aktuellen Knoten existieren, die Basen repräsentieren mit denen sie eine Bindung eingehen könnte. Der Baum muß dann entlang all dieser Kanten weiterverfolgt werden, was zu einer starken Auffächerung des rekursiven Suchalgorithmus führen kann. Beginnend mit dem ersten Zeichen des Musters in der Wurzel des Baums arbeitet der Algorithmus folgendermaßen:

1. Falls noch ein Zeichen im Muster enthalten ist, mache dieses zur aktuellen Base, anderenfalls fahre fort bei 4.
2. Suche alle Kanten dieses Knotens, die eine Base repräsentieren, mit der die aktuelle eine Bindung eingehen kann.
3. Folge allen diesen Kanten der Reihe nach zu ihrem nächsten Knoten und beginne bei 1. mit der nächsten Base des Musters.
4. Ist der aktuelle Knoten ein Blatt, so füge dessen Position in die Liste der Lösungen ein.

Mit diesem Vorgehen lassen sich alle komplementären Teilsequenzen finden, die genauso lang sind wie die Mustersequenz. In Abbildung 4.7 soll dieser Sachverhalt noch einmal verdeutlicht werden.

Eine der Anforderungen an den Suchalgorithmus war, im Muster implizit enthaltene Teilmuster (nur die Präfixe) ebenfalls zu suchen, sofern sie eine gewisse Mindestlänge überschreiten. Dies kann mit diesem Suchalgorithmus sehr leicht realisiert werden. Ist nämlich die Rekursionstiefe größer als diese Mindestlänge, kann von diesen Knoten aus nach allen Blättern gesucht und deren Positionen in die Liste der Lösungen eingetragen werden, denn alle Teilsequenzen, die durch diese Blätter bestimmt sind, beginnen mit dem durch die Mindestlänge definierten Komplement des ersten Musterpräfix. Dazu gibt Abbildung 4.8 ein Beispiel.

Man könnte nun vermuten, daß man nur noch bis zu diesen Knoten suchen und von dort aus die Positionen aller darunterliegenden Blätter in die Lösungsliste eintragen muß, da ja alle längeren Teilsequenzen auch in diesen Unterbäumen liegen. Dies

Mustersequenz: G G A Komplemente: C A U an Position 2
 Positionsbaum: U C U an Position 4

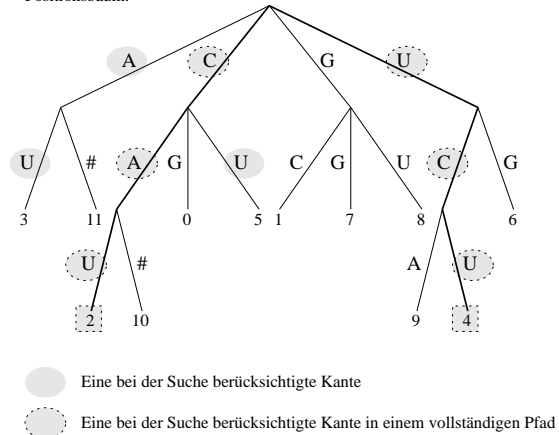


Abbildung 4.7: Suche nach allen komplementären, gleichlangen Teilsequenzen

ist leider nicht möglich, da durch diese Art der sparsamen Mustergenerierung dem übergeordneten suchenden Algorithmus die Längen der gefundenen Teilsequenzen nicht bekannt sind und sie deshalb vom Suchalgorithmus mit den Teilsequenzen mitgeliefert werden müssen. Deswegen muß für jedes Musterpräfix, dessen Länge besagte Mindestgrenze überschreitet, die Suche nach den Blättern neu angestoßen werden, bis das Muster komplett abgearbeitet ist. Dieser Nachteil ist der Preis für eine sehr einfache Mustergenerierung, arbeitet aber trotzdem immer noch etwa doppelt so schnell¹ wie die Variante, in der die Muster aller Längen explizit erzeugt werden.

In Kapitel 3.2 wurde angekündigt, daß der Suchraum dadurch eingeschränkt werden kann, daß nur Helices zugelassen werden, deren mittlere Bindungsstärken über einem vorher aus den realisierten Helices des Bereiches ermittelten Wert liegen. Die mittleren Bindungsstärken werden während der Suche in jeder Rekursionsstufe errechnet bzw. erweitert. Dadurch kann, sobald eine Teilsequenz als mögliche Lösung identifiziert ist, überprüft werden, ob sie dieser Bedingung genügt. Allerdings wird so, zumindest bei diesem Suchalgorithmus, nicht der Suchraum eingeschränkt, sondern nur der Lösungsraum.

¹empirisch ermittelt

Mustersequenz: G G A Komplemente: A U an Position 3 C A U an Position 2
 Positionsbaum: C A an Position 10 U C A an Position 9
 C U an Position 5 U C U an Position 4

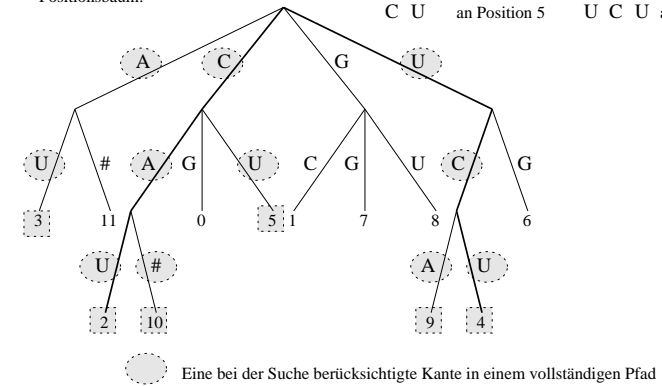


Abbildung 4.8: Suche nach allen komplementären Teilsequenzen, die durch das Präfix der Länge 2 des Musters bestimmt sind

Der beschriebene Suchalgorithmus war bis jetzt nur darauf ausgelegt, alle komplementären Teilsequenzen zu finden, deren Längen mit der des Musters übereinstimmen, und alle Teilsequenzen, die mit zu den Präfixen des Musters komplementären Bereichen beginnen, sofern sie eine gewisse Mindestlänge überschritten haben. Bei der Suche können aber noch andere Fälle auftreten, die auch zu Lösungen führen und deswegen nicht ignoriert werden können:

- Nachdem das letzte Zeichen des Musters gesucht worden ist, befindet sich der Algorithmus in einem Blatt.
 Dies ist der optimale Fall, indem eine oder mehrere gleichlange, zum Muster komplementäre Teilsequenzen gefunden werden konnten. Diese Situation ist in Abbildung 4.7 dargestellt und ist im vorangegangenen Text bereits besprochen worden.
- Nachdem das letzte Zeichen des Musters gesucht worden ist, befindet sich der Algorithmus in einem inneren Knoten.

Dieser Fall ist äquivalent zu der bereits beschriebenen Suche nach allen Präfixen eines Musters. Da das Muster zu Ende ist, bedeutet das, daß alle Teilsequenzen, die durch die Blätter unter diesem Knoten bestimmt sind mit einem zum Muster komplementären Präfix derselben Länge beginnen und somit alle in die Lösungsmenge aufgenommen werden müssen. In Abbil-

dung 4.8 wurde dieser Sachverhalt bereits illustriert.

- Bevor das letzte Zeichen des Musters gesucht werden konnte, befindet sich der Algorithmus in einem Blatt.

In diesem Fall stellt die durch das Blatt definierte Teilsequenz ein komplementäres Präfix des Suchmusters dar. Da der Baum keine weitere Auskunft darüber geben kann, ob in dem Sequenzabschnitt nach der gefundenen Teilsequenz noch weitere, zum Rest des Musters komplementäre Basen folgen, muß dies jetzt direkt überprüft werden. Abbildung 4.9 soll dies verdeutlichen. Die Überprüfung geschieht zeichenweise, wobei wieder jedes Präfix wie auch die mittlere Bindungsstärke berücksichtigt werden müssen.

Position: 0 1 2 3 4 5 6 7 8 9 10 11 _ _
 Sequenz: C G C A U C U G G U C A #

Mustersequenz: C C A G Komplemente: G G an Position 7
 G G U an Position 7
 G G U C an Position 7

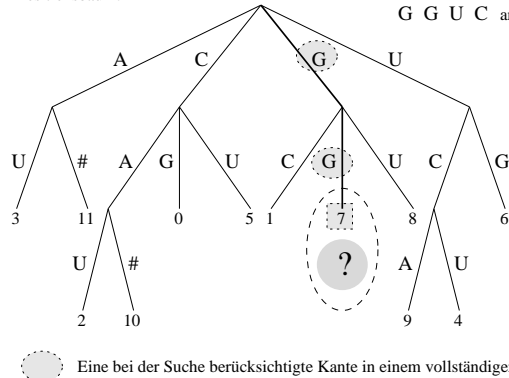


Abbildung 4.9: Direkter Vergleich des Restmusters mit der Sequenz bei Mustern, die länger sind als ihre komplementären Pfade im Baum

4.1.3.3 Suche mit 'mismatches'

Aufgrund von Mutationen ist es möglich, daß Helices sich nicht aus zwei vollständig paarenden Helixhälften zusammensetzen, sondern einige der Basenpaare aus nicht paarenden Basen bestehen oder eine dieser Basen komplett verschwunden ist. Man

unterscheidet dabei drei Arten von Mutationen: Deletion, Insertion und Substitution, wie sie in Kapitel 2 beschrieben wurden. Jede dieser Mutationen kann zu 'mismatches' in den Helixhälften führen. Für den Suchalgorithmus heißt das, daß er in jeder Rekursionsstufe etwaige 'mismatches' in Betracht ziehen muß. Bei der Suche ist eine gewisse Höchstzahl an 'mismatches' zwischen einem Muster und einer zu ihr komplementären Teilsequenz zugelassen. Im folgenden soll diese Zahl auf 1 festgelegt werden, so daß bei der Suche die weitere 'mismatches' nur dann berücksichtigt werden, wenn bis zur aktuellen Rekursionsstufe noch kein 'mismatch' aufgetreten ist. Der Suchalgorithmus behandelt 'mismatches' zur Zeit ihres Auftretens in den Teilsequenzen, so daß das Muster selbst nicht modifiziert werden muß, obwohl man eigentlich vom Übergang zu einer Mustermenge sprechen kann. Anhand des Beispiels in Abbildung 4.10 soll dies verdeutlicht werden. Da die 'mismatches', in diesem Fall nur einer, an beliebiger Position im Muster auftreten können, vervielfältigen sich die möglichen Muster zusätzlich.

Deletion	Insertion	Substitution
G G A	G G A	G G A
G A	* G G A	* G A
(G A) doppelt !	G * G A	G * A
G G	G G * A	G G *
	G G A *	

Abbildung 4.10: Alle impliziten Veränderungen einer Mustersequenz bei einer Suche mit einem 'mismatch', wobei * eine beliebige Base repräsentiert

Aus diesem Beispiel wird auch ersichtlich, wie der Algorithmus mit den drei möglichen Arten von 'mismatches' umgehen muß:

- Bei einer Deletion wird die aktuelle Base des Musters ignoriert und die Suche mit der nächsten Base im selben Knoten fortgesetzt.
- Bei einer Insertion wird die aktuelle Base des Musters in die nächste Rekursionsstufe gerettet, d.h. es wird der Reihe nach entlang aller Kanten in die nachfolgenden Knoten gesprungen und von dort aus mit der aktuellen Base weiter gesucht.
- Bei einer Substitution wird entlang aller Kanten weitergesucht, anstatt nur entlang der Kanten, die Basen repräsentieren, mit denen die aktuelle Base des Musters eine Bindung eingehen könnte.

Wird eine Suche mit 'mismatches' angefordert, so muß der Algorithmus, solange die Höchstgrenze für die Anzahl von 'mismatches' noch nicht überschritten ist, in jeder Rekursionsstufe alle drei Arten der Behandlung von 'mismatches' durchführen. Es ist leicht einzusehen, daß sich der Rechenaufwand dadurch stark

schen Verfahren oftmals nicht beizukommen. Im folgenden soll deshalb ein Threshold Accepting genannter Mutations-Selektions-Algorithmus vorgestellt werden, der dieses Problem in akzeptabler Zeit löst, dabei aber nicht unbedingt die optimale, aber eine ausreichend gute Lösung findet. Vorweg werden einige Überlegungen gestellt, die zur Wahl dieses Algorithmus führten.

4.2.1 Anforderungen an den Optimierungsalgorithmus

Die Suche nach dem optimalen Set möglicher Helices ist sicherlich einer der wichtigsten Punkte im hier dargelegten Verfahren zum sekundärstrukturbasierten Alignment. An dieser Stelle wird jeder realisierten Helix genau eine der möglichen Helices zugeordnet, also das eigentliche Mapping durchgeführt, durch das die Sequenz dann ausgerichtet werden kann. Leider hat sich in Kapitel 3.3.2 herausgestellt, daß das Problem eine Komplexität von $O(m^n)$ aufweist, wobei m für die durchschnittliche Anzahl der Elemente der n Mengen möglicher Helices steht. Da dem Problem mit dem in Kapitel 3.3.2 beschriebenen, deterministischen Verfahren in akzeptabler Zeit nicht beizukommen ist, wird auf nicht deterministische Verfahren ausgewichen. Diese Verfahren liefern im allgemeinen nur eine näherungsweise Lösung des Problems, weswegen die Anforderungen an die Algorithmen etwas freier definiert werden müssen:

- Ein Set soll möglichst wenige positionelle oder strukturelle Konflikte aufweisen und dabei
- die höchst mögliche Ähnlichkeit zu den realisierten Helices aufweisen.
- Darüberhinaus soll der Algorithmus sich möglichst einfach implementieren lassen, um eine gute Einbindung in die gegebene Programmstruktur zu ermöglichen.

Die Lockerung der Bedingung für die Zahl der Konflikte rührt zudem von der Tatsache her, daß es zu Konfliktsituationen kommen kann, die nicht aufgelöst werden können. In diesem Fall muß dann nachträglich auf die Zuordnung einer Helix verzichtet werden. Hinter dem letzten Punkt steht die begründete Hoffnung, einen Algorithmus zu finden, der direkt auf den bereits bestehenden Datenstrukturen implementiert werden kann. Ein paar solcher Algorithmen werden im folgenden besprochen.

4.2.2 Sammlung und Diskussion verschiedener Alternativen

Die Auswahl eines geeigneten Algorithmus stellt sich in diesem Fall deutlich schwieriger als in Kapitel 4.1.2 bei der Wahl des Suchalgorithmus dar, da die Kriterien

4.2. SUCHE NACH DEM OPTIMALEN SET DER STRUKTURELEMENTE 75

in diesem Fall wesentlich uneindeutiger sind. Die ersten beiden sind Voraussetzungen, die der Algorithmus auf jeden Fall erfüllen muß. Er muß das Problem in akzeptabler Zeit ausreichend gut lösen können, also die ersten beiden Kriterien erfüllen. Der dritte Punkt ist zwar ein brauchbares Kriterium, allerdings ein sehr subjektives, weswegen die Qualität der Verfahren anhand geeigneter Literatur, in diesem Fall [WKin], vorgenommen wurde. In Frage kommen:

- Neuronale Netze:

Obwohl sie sicherlich die flexibelsten Algorithmen zur Lösung hoch komplexer Probleme sind und schon oft erfolgreich auf Optimierungsprobleme angewendet wurden, kommen sie doch wegen des hohen Implementierungs- und Trainingsaufwandes für das hier gestellte Problem nicht in Frage.

- Genetische Algorithmen:

Der Implementierungsaufwand für einen, an das Problem angepaßten genetischen Algorithmus ist dagegen sicherlich vertretbar. Sie basieren auf der Mutation und Rekombination einer Lösungspopulation unter dem Selektionsdruck der optimalen Lösung. Sie sollten sich also sehr gut für dieses Problem eignen. Allerdings ist die Brauchbarkeit der Möglichkeit zur Rekombination von Lösungen bei dieser Anwendung zumindest fraglich, da die Helices eines Sets stark voneinander abhängen. Ob bei einer Rekombination zweier Lösungen überhaupt eine Verbesserung zu erwarten ist, müßte noch erprobt werden. Auf jeden Fall bieten genetische Algorithmen eine Alternative, die unter Umständen bessere Ergebnisse erzielen kann, als der relativ einfach implementierte Mutations-Selektions-Algorithmus.

- Mutations-Selektions-Verfahren:

Diese Verfahren basieren auf der Mutation zufällig ausgewählter Chromosome, in diesem Fall Helices eines Sets, ebenfalls unter dem Selektionsdruck der optimalen Lösung. Der primäre Unterschied bei diesen Verfahren liegt in der Methode der Selektion. Beim

- Simulated Annealing

wird die Selektion durch eine sich langsam verändernde Zufallsfunktion realisiert, die die Akzeptanz einer besseren bzw. schlechteren Lösung immer wahrscheinlicher bzw. unwahrscheinlicher werden läßt.

- Threshold Accepting

wird die Selektion nicht über eine Zufallsfunktion, sondern über einen sich langsam verkleinernden Schwellenwert realisiert, der die Akzeptanz von schlechteren Lösungen immer stärker einschränkt, bis nur noch bessere Lösungen angenommen werden.

Nach [WKin] wurde in großangelegten Versuchen am Traveling Salesman und anderen Problemen festgestellt, daß das Verfahren des Threshold Accepting in mehr als der Hälfte der Fälle bessere Ergebnisse lieferte als Simulated Annealing. Zudem ist es das weniger rechenintensive Verfahren, da keine Zufallsfunktion implementiert werden muß, weshalb es zur Lösung des vorliegenden Problems gewählt wurde.

4.2.3 Threshold Accepting als geeignetster Algorithmus

Die Auswahl des besten Helix-Sets wird in der vorliegenden Arbeit über einen Mutations-Selektions-Algorithmus, der nach seiner Eigenschaft, Verschlechterungen bis zu einem gewissen Schwellenwert zuzulassen, Threshold Accepting genannt wird, realisiert. Sei M eine Menge n -dimensionaler Vektoren, $T \in \mathcal{R}_0^+$ ein Schwellenwert, und sei ein Vektor $v \in M$ gesucht, für den eine 'fitness'-Funktion $f : M \rightarrow \mathcal{R}, v \mapsto f(v)$ maximal wird, so arbeitet dieser Algorithmus (nach [WKin]) prinzipiell wie folgt:

1. Wähle einen beliebigen Startvektor $v \in M$.
2. Verändere diesen Vektor v zu v' .
3. Berechne $r = F(v') - T$.
4. Falls $f(v') \geq r$ wähle v' , ansonsten behalte v als neuen Vektor v .
5. Verkleinere T .
6. Falls $f(v)$ noch nicht groß genug oder ein anderes Abbruchkriterium noch nicht erfüllt ist, fahre fort bei 2.

Dabei wird noch keine Aussage darüber getroffen, wie die Veränderung von v nach v' vorzunehmen ist. Man könnte z.B. alle Koordinaten des Vektors gleichzeitig ändern. In dieser Arbeit wird dagegen immer nur eine Koordinate verändert. In jedem Schritt wird überprüft, ob sich die 'fitness' $F(v')$ des neuen Vektors v' gegenüber der des alten verbessert hat. Hat sie dies, wird das gleiche Verfahren mit diesem neuen Vektor durchgeführt. Sollte sie sich verschlechtern, wird nur dann der neue Vektor v' weiterverwendet, wenn seine 'fitness' höchstens um T unter der des alten Vektors v liegt. Zusätzlich wird diese Schranke jedesmal etwas kleiner. Auf diese Weise erzeugt man einen langsam ansteigenden Selektionsdruck zu Gunsten der höheren 'fitness'.

Die Suche nach dem besten Helix-Set läßt sich leicht auf diese Struktur abbilden. Der gesuchte Vektor $v \in M$ entspricht dem optimalen Helix-Set. Die Menge M ist die Menge aller möglichen Helix-Sets (s. Kapitel 3.3.2). Die 'fitness'-Funktion

4.2. SUCHE NACH DEM OPTIMALEN SET DER STRUKTURELEMENTE77

f setzt sich aus der Anzahl der Konflikte und der Summe der Ähnlichkeiten der Helices des Sets zusammen, muß aber in diesem Fall minimiert werden. Für den Schwellenwert T kann ein geeigneter Startwert empirisch ermittelt werden. Dennoch sind einige Anpassungen notwendig. Im folgenden soll nun die Anwendung des Threshold Accepting Algorithmus auf die Suche nach dem optimalen Helix-Set beschrieben werden. Als Startvektor wird das Set gewählt, daß die höchste Ähnlichkeit aufweist. Wie in Abbildung 4.12 dargestellt, handelt es sich dabei um das Set in dem alle Helices den Index 1 tragen, da die Elemente der n Mengen möglicher Helices nach ihrer Ähnlichkeit zu der ihnen zugeordneten realisierten Helix sortiert sind.

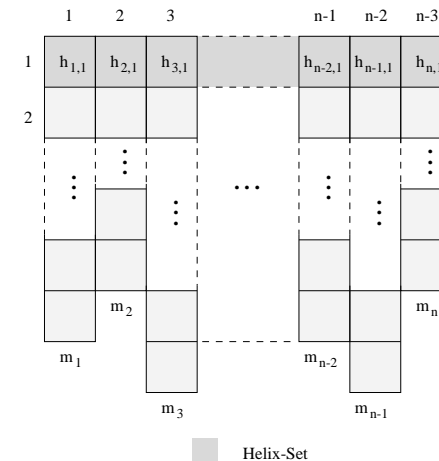


Abbildung 4.12: Das erste Helix-Set aus den ähnlichsten Helices

Über den Helices dieses ersten Sets wird eine symmetrische Konfliktmatrix aufgebaut. Abbildung 4.13 stellt eine solche dar. Diese Matrix wird durch paarweises Überprüfen der Helices des Sets auf positionelle und strukturelle Konflikte erzeugt. Für die Überprüfung auf strukturelle Konflikte müssen jeweils noch die beiden zugehörigen realisierten Helices herangezogen werden. Da Änderungen des Sets immer nur an einer Position vorgenommen werden, muß diese Matrix nicht immer wieder neu aufgebaut werden, sondern kann im Verlauf des Optimierungsprozesses immer auf dem aktuellen Stand gehalten werden.

Aus der Menge der Konflikte wird nun zufällig ein Konflikt ausgewählt (s. Abbildung 4.13). Da an einem Konflikt immer zwei Helices beteiligt sind, muß eine von diesen ausgewählt werden. Dies soll wiederum zufällig geschehen. Allerdings ist

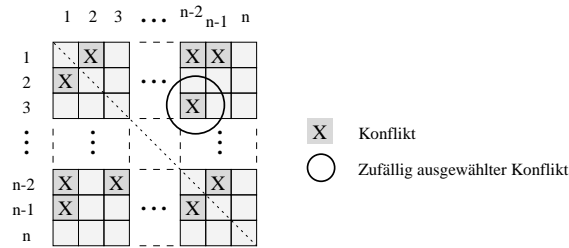


Abbildung 4.13: Zufällige Auswahl eines Konflikts aus der Konfliktmatrix

zu beachten, daß die zugehörigen Mengen möglicher Helices unterschiedlich viele Elemente enthalten können und da es sinnvoller ist, eine neue Helix aus der Menge zu wählen, in der die größere Auswahl herrscht, sollte dies bei der Wahl der Helix berücksichtigt werden. Dies kann durch Verwendung einer Zufallsfunktion geschehen, die eine der beiden Helices h_1 und h_2 mit den Wahrscheinlichkeiten

$$p(h_1) = \frac{m_1}{m_1 + m_2} \quad \text{und} \quad p(h_2) = \frac{m_2}{m_1 + m_2} \quad \text{mit} \quad m_1 = |M_1|, m_2 = |M_2| \quad (4.1)$$

auswählt. Dadurch wird die größere der beiden Mengen bevorzugt ohne die kleinere völlig auszuschließen. Jetzt steht fest, an welcher Stelle im Helix-Set eine Veränderung vorgenommen werden soll, also eine andere Helix die alte ersetzen soll. Es hat sich herausgestellt, daß eine starke Veränderung, z.B. durch eine zufällige Auswahl, bei der alle Helices der Menge gleich wahrscheinlich sind, keine guten Ergebnisse liefert. Deshalb wird nur eine zufällige Verschiebung um eine Position nach oben (niedrigerer Index) oder nach unten (höherer Index) vorgenommen, wie in Abbildung 4.14 zu sehen ist.

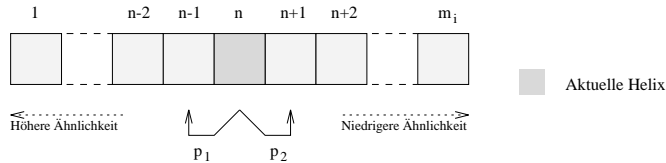


Abbildung 4.14: Auswahl einer neuen Helix durch zufällige Erhöhung oder Erniedrigung des Index um den Wert 1

Die Wahrscheinlichkeit, mit der die Erhöhung bzw. Erniedrigung des Index vorgenommen wird, sollte am Anfang des Optimierungsprozesses eine Erhöhung bevorzugen, da zu diesem Zeitpunkt die Indizes noch relativ niedrig sind. Zwar wird da-

durch erst einmal eine Verschlechterung bzgl. der Ähnlichkeit in Kauf genommen: es stellt aber zunächst den einzigen Weg dar, eine Verringerung der Konflikte zu erreichen. Mit dem Fortschreiten des Optimierungsprozesses sollte sich aber diese Wahrscheinlichkeit langsam wieder zugunsten niedriger Indizes verschieben, um so einen Selektionsdruck in Richtung höherer Ähnlichkeit zu bewirken.

Für die Akzeptanz eines neuen Helix-Sets ist dann allerdings nur die Zahl der Konflikte ausschlaggebend. Zuerst wird die Anzahl der Konflikte des neuen Sets ermittelt und dann mit der des alten Sets verglichen. Ist sie niedriger oder nur um den Wert der Toleranzschwelle höher als die des alten Sets, wird das neue Set akzeptiert, ansonsten wird die Optimierung mit dem alten Set fortgesetzt. Im Falle der Akzeptanz muß die Konfliktmatrix aufbereitet werden, d.h. die Konflikte der alten Helix werden aus der Matrix entfernt und die der neuen eingetragen. Jetzt kann, sofern noch Konflikte vorliegen, ein weiterer Optimierungslauf mit dem veränderten Set durchgeführt werden. Der Wert der Toleranzschwelle sollte sich ebenfalls bei fortlaufender Optimierung verringern. Da es sich hierbei um Werte aus \mathcal{Z} handelt, sollte die Verringerung jeweils um den Wert 1 geschehen und ihr Zeitpunkt von der Zahl der durchlaufenen Optimierungsschleifen abhängen. Als Abbruchkriterium dient also zum einen die Anzahl der Konflikte, zum anderen die Anzahl der durchlaufenen Optimierungsschleifen. Ein Beispiel für ein nach einem Abbruch vorliegendes Helix-Set ist in Abbildung 4.15 zu sehen.

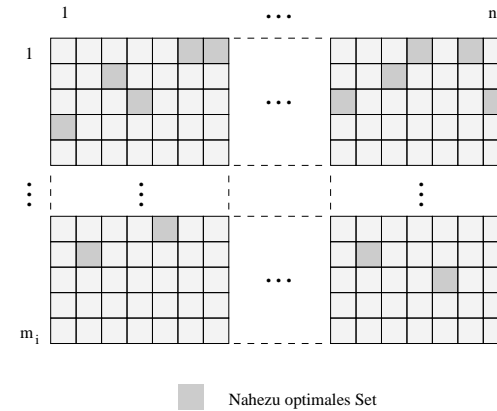


Abbildung 4.15: Nach Abbruch der Optimierung vorliegendes, beinahe optimales Helix-Set

Es hat sich gezeigt, daß doch einige Änderungen des Grundalgorithmus des Threshold Accepting notwendig waren, um ihn an die durch diese Problematik gegebene

nen Bedingungen anzupassen. Der augenfälligste Unterschied liegt in der Veränderung des aktuellen Vektors. Ein Teil der 'fitness'-Funktion wurde an diese Stelle verlagert. Die Einflußnahme auf die Ähnlichkeit wurde ausschließlich durch die Vorgabe des ersten Sets und die anschließenden Vektorveränderungen erzielt. Bei der eigentlichen 'fitness'-Bewertung war nur noch die Anzahl der Konflikte ausschlaggebend.

Zum Abschluß dieses Kapitels soll noch geklärt werden, welche Maßnahmen ergriffen werden, wenn es dem Threshold Accepting Algorithmus nicht gelingt, alle Konflikte aufzulösen. Dies kann zum einen geschehen, weil bei solchen nicht deterministischen Algorithmen keine Erfolgsgarantie gegeben werden kann, zum anderen, weil es, wie in Kapitel 3.3.1 beschrieben, zu Konstellationen kommen kann, in denen kein Algorithmus diese Konflikte auflösen könnte. Sind also nach Abbruch des Algorithmus noch Konflikte vorhanden, so werden diese, sofern das ihre Zahl zuläßt, kombinatorisch aufgelöst. Dazu werden zuerst alle beteiligten Helices ermittelt, um die Mengen der möglichen Helices, in die sie gehören, zu erfahren. Anschließend werden alle Helixkombinationen aus diesen Mengen gebildet und auf Konflikte sowohl untereinander als auch gegenüber den restlichen Helices des Sets überprüft, bis alle Konflikte aufgelöst sind. Sollte dabei ein Konflikt gefunden werden, der nicht aufgelöst werden kann, wird eine der beiden beteiligten Helices zufällig aus dem Set gestrichen. Leider verringert sich dadurch die Anzahl der Helices, die später zum Ausrichten der Sequenz zur Verfügung stehen.

Kapitel 5

Kombiniertes Alignment

Bevor in diesem Kapitel auf die Notwendigkeit einer Kombination von primärstruktur- und sekundärstrukturbasiertem Alignment eingegangen wird und verschiedene Möglichkeiten aufgezeigt werden diese Kombination zu realisieren, soll noch einmal dargestellt werden, worauf die Alignment-Problematik eigentlich beruht.

Die beiden entscheidenden Faktoren beim Ausrichten einer rRNS Sequenz gegen eine andere, etwa eine Konsensussequenz, sind die Primär- und die Sekundärstruktur dieser beiden Sequenzen. Da sich diese beiden Strukturen in den beiden Sequenzen unterscheiden, ist beim Ausrichten der einen Sequenz an der anderen für jede Struktur ein gewisser Aufwand zu betreiben. Man könnte auch sagen, das Ausrichten erzeugt strukturabhängige Kosten. Betrachtet man den Alignment-Vorgang als Abbildung

$$\text{align} : \mathcal{B}^* \rightarrow \hat{\mathcal{B}}^*, s \mapsto \text{align}(s) = s', \quad (5.1)$$

so kann daraus ein Kostenmodell für diese Funktion abgeleitet werden:

$$\begin{aligned} \text{cost}(\text{align}) &= w_{\text{prim}} \text{cost}_{\text{prim}}(\text{align}) + w_{\text{sec}} \text{cost}_{\text{sec}}(\text{align}), \\ \text{mit } w_{\text{prim}}, w_{\text{sec}} &\in [0, 1] \quad \text{und} \quad w_{\text{prim}} + w_{\text{sec}} = 1. \end{aligned} \quad (5.2)$$

Das Ziel muß es sein, den Alignment-Vorgang so zu gestalten, daß diese Kosten minimiert werden. Durch die Abhängigkeiten, die zwischen der Primär- und der Sekundärstruktur bestehen, wird das Auffinden einer solchen Abbildung wesentlich erschwert. Es wird vermutet, daß ein Alignment unter Berücksichtigung beider Strukturen ein \mathcal{NP} -vollständiges Problem darstellt. Deswegen wird nach Möglichkeiten gesucht, sinnvolle Vereinfachungen zu finden. Die naheliegendste Vereinfachung ist die Berücksichtigung nur einer der beiden Strukturvarianten. Formal ist

dies mit einer Gewichtung der Kosten der anderen Strukturvariante mit 0 gleichzusetzen. In der vorliegenden Arbeit und in [BRei] wurden Verfahren entwickelt, die diesen Teilspekten des Alignment-Problems gerecht werden.

5.1 Gründe für eine Kombination

Um das Alignment-Problem zu vereinfachen, wurde also eine Unterteilung in primär- und sekundärstrukturbasiertes Alignment vorgenommen. Nachdem nun Verfahren zur Lösung dieser Teilprobleme zur Verfügung stehen, ist eine Kombination dieser Verfahren im Sinne der Kostenfunktion $cost(align)$ notwendig, will man die Alignment-Problematik vollständig erfassen. Die Entsprechung zum Kostenmodell ist sicherlich der zwingendste Grund, sich mit möglichen Methoden zur Kombination zu befassen. Darüberhinaus hat sich auch gezeigt, daß keines der beiden Verfahren für sich genommen ein optimales Alignment einer Sequenz gewährleisten kann. Das sekundärstrukturbasierte Alignment kann dies allein deswegen schon nicht leisten, weil es gar nicht die gesamte Sequenz erfaßt. Deutlich bessere Ergebnisse lassen sich über ein primärstrukturbasiertes Alignment erzielen. Allerdings weisen rRNS Sequenzen in ihrer Primärstruktur Bereiche auf, die so variabel sind, daß das Verfahren hier gar keinen Ansatzpunkt für eine Ausrichtung finden kann. Da es sich oftmals so verhält, daß die variablen Bereiche der Primärstruktur von der Sekundärstruktur und die fehlenden Teilbereiche innerhalb der Sekundärstruktur von den konservativen Bereichen der Primärstruktur abgedeckt werden, können sich aber beide Verfahren durch eine geeignete Kombination ergänzen.

5.2 Mögliche Kombinationen

Methoden, die Kombination von primär- und sekundärstrukturbasiertem Alignment zu realisieren, existieren sicherlich genügend. Entscheidend dabei ist immer, in wie weit sie dem Kostenmodell für das vollständige Alignment gerecht werden. Im folgenden werden deshalb einige, zum Teil heuristische Verfahren vorgeschlagen und auf ihre Eignung hin untersucht. Dieses Kapitel soll die Grundlage zu weiteren Überlegungen bieten, mit dem in dieser Arbeit entwickelten Verfahren zum sekundärstrukturbasierten Alignment und dem in [BRei] entwickelten Verfahren zum primärstrukturbasierten Alignment zu einem vollständigen Alignment zu gelangen. Bei mehreren dieser Methoden ist es notwendig Signifikanzvergleiche der Alignments an bestimmten Positionen bzw. Helices vorzunehmen. Die notwendigen Informationen können von den beiden Alignment-Verfahren selbst geliefert werden, d.h. während des primärstrukturbasierten und des sekundärstrukturbasierten Alignments kann anhand der Möglichkeiten für eine Zuordnung bestimmt werden, wie hoch die Wahrscheinlichkeit einer Fehlzusammenordnung ist. Diese Abschätzungen müssen im jeweiligen Alignment zur Verfügung stehen, damit sie bei einer

Kombination berücksichtigt werden können.

1. Direkte Kombination:

Das erste Verfahren zur Kombination von primär- und sekundärstrukturbasiertem Alignment, das hier vorgeschlagen werden soll, ergibt sich unmittelbar aus der in 5.1 erwähnten Eigenschaft der Verfahren, sich gegenseitig zu ergänzen. Da gerade die Helixbereiche in ihrer Primärstruktur variabler sind als die meisten anderen Bereiche, bietet es sich an, die konservativen Bereiche mittels primärstrukturbasiertem und die variablen mittels sekundärstrukturbasiertem Alignment auszurichten. Dabei kann man auf zwei Arten vorgehen, wie in Abbildung 5.1 und Abbildung 5.2 dargestellt.

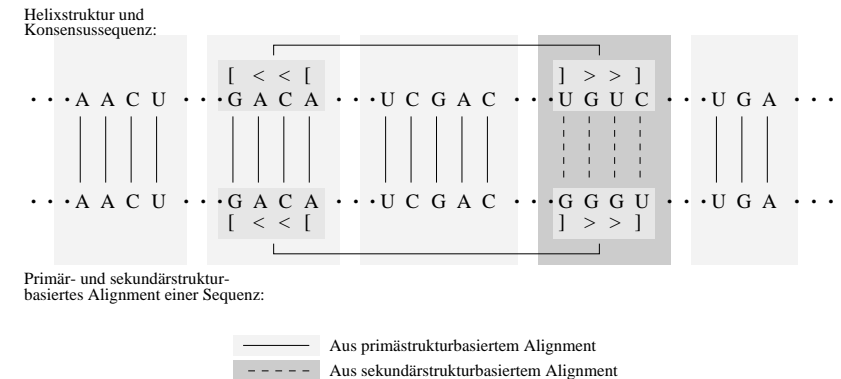


Abbildung 5.1: Direkte Kombination des primär- mit dem sekundärstrukturbasierten Alignment

In der ersten Variante, die sich stärker auf die Primärstruktur stützt, werden nur in den Bereichen, die durch das primärstrukturbasierte Alignment nur schlecht oder sehr unsicher ausgerichtet werden konnten, versucht das Alignment mit Hilfe des sekundärstrukturbasierten Alignments zu kompletieren.

Bei der zweiten Variante werden im Gegensatz dazu nur die Bereiche durch das primärstrukturbasierte Alignment erfaßt, die von dem sekundärstrukturbasierten Alignment nicht berücksichtigt oder unsicher ausgerichtet wurden.

Beide Möglichkeiten haben allerdings zwei gravierende Nachteile. Zum einen wird durch den exklusiven Gebrauch eines Alignment-Typs auf jeden Bereich der Sequenz auf die Möglichkeit verzichtet, die beiden Alignments mit-

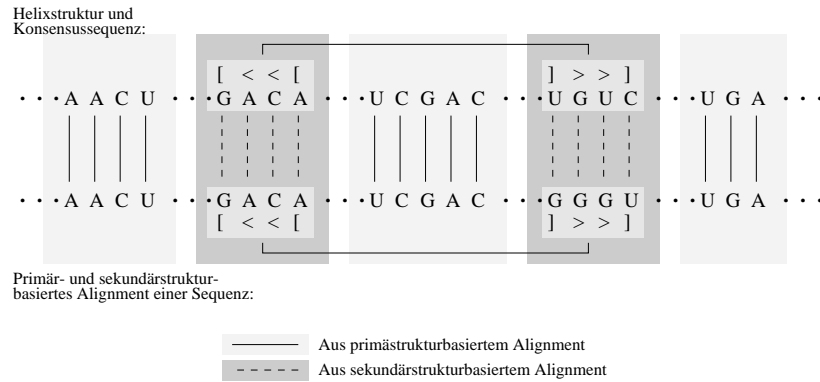


Abbildung 5.2: Direkte Kombination des sekundär- mit dem primärstruktur-basierten Alignment

einander zu vergleichen und zu verifizieren. Vor allem beim sekundärstruktur-basierten Alignment ist dies ratsam, da es aufgrund der vielen Möglichkeiten für die Sekundärstruktur ein eher unsicheres Verfahren ist. Zum anderen entspricht es nicht dem Kostenmodell für ein komplettes Alignment. In beiden Fällen geht nicht einmal einer der Alignment-Typen vollständig in das Alignment ein, geschweige denn beide. Damit stellt sich die augenfälligste Kombinationsmethode als die wohl am wenigsten geeignetste heraus.

2. Kombination durch Verifikation:

Eine andere Methode zur Kombination bietet sich durch die Verifikation des sekundärstruktur-basierten Alignments anhand des primärstruktur-basierten. Dabei wird nach Abweichungen der Sekundärstruktur in der Primärstruktur gesucht, was sich durch nicht übereinstimmende Basen in den Helixhälften mit denen an den ihnen zugeordneten Positionen in der Primärstruktur äußert. Ist eine solche gefunden kann diese z.B. durch Insertionen oder Deletionen von Basen in den von der Sekundärstruktur nicht erfaßten Bereichen zustande gekommen sein. Die Helices können dadurch an andere Positionen verschoben worden sein. Abhilfe kann dann, wie in Abbildung 5.3 dargestellt, durch eine Suche in der näheren Umgebung und eine entsprechende Verschiebung der Helixhälften bringen.

Auf diese Weise können sowohl Fehlzuordnungen in der Sekundärstruktur, als auch in der Primärstruktur aufgedeckt werden. Sollte die Signifikanz der Primärstruktur in einem solchen Bereich unzulänglich sein, weil es sich leicht um einen variablen Bereich handelt, kann dann auf die Sekundärstruktur

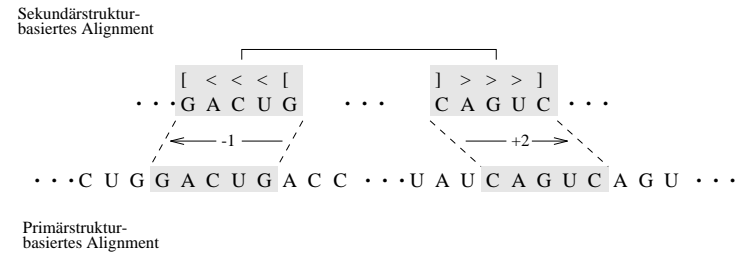


Abbildung 5.3: Überprüfung und positionelle Anpassung des sekundärstruktur-basierten an das primärstruktur-basierte Alignment

zurückgegriffen werden. Dem Kostenmodell für ein vollständiges Alignment wird bei dieser Methode vollkommen entsprochen, da beide Strukturen vollständig in das Alignment einfließen. Allerdings müssen in diesem Fall die Kosten für das primärstruktur-basierte Alignment stärker gewichtet werden, da es die Grundlage für den Vergleich bildet.

3. Vergleichende Kombination:

Eine weitere Methode zur Kombination könnte darin bestehen, das primärstruktur- und das sekundärstruktur-basierte Alignment positionsweise miteinander zu vergleichen und jeweils die signifikantere Ausrichtung zu übernehmen. Ein solches Vorgehen wird in Abbildung 5.4 dargestellt.

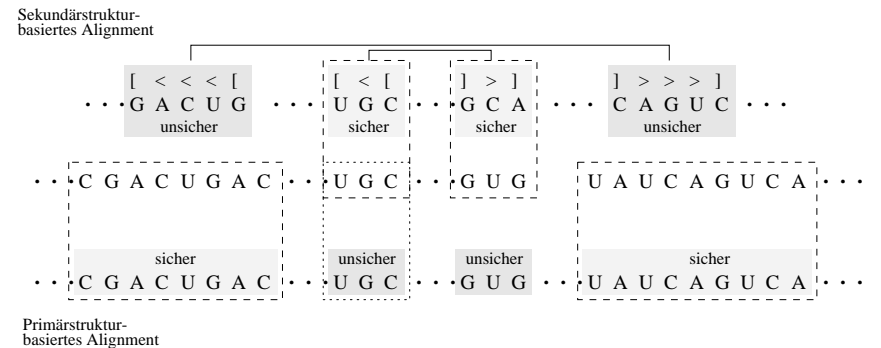


Abbildung 5.4: Vergleich von primärstruktur- und sekundärstruktur-basiertem Alignment und Übernahme der signifikanteren Positionen

Signifikante, bzw. sichere Positionen sind solche, die von den Alignment-Verfahren mit einer gewissen Wahrscheinlichkeit richtig zugeordnet wurden. Problematisch wird diese Methode dann, wenn es z.B. zu Überschneidungen von sicheren Bereichen der Sekundärstruktur sowohl mit sicheren als auch mit unsicheren Bereichen der Primärstruktur kommt. Beim Übergang von dem sicheren in den unsicheren Bereich der Primärstruktur, für den dann eigentlich die Ausrichtung durch das an dieser Stelle sichere sekundärstrukturbasierte Alignment herangezogen werden sollte, kommt es zum Konflikt mit dem vorangegangenen sicheren Bereich der Primärstruktur. Da beide Strukturen vollständig in das Alignment einfließen, entspricht diese Methode dem Kostenmodell für ein vollständiges Alignment. Die Gewichtung der Einzelkosten hängt dann vom Grad des Einflusses der beiden Alignment-Typen auf das gesamte Alignment ab.

4. Integration der Primärstruktur in die Sekundärstruktursuche:

Einen etwas anders gearteten Ansatz stellen die nächsten beiden Methoden dar. Hier werden nicht beide Alignment-Typen unabhängig von einander erzeugt und anschließend verknüpft, sondern jeweils nur eins wird erzeugt und anschließend bei der Durchführung des anderen verwendet. In diesem Fall wird das primärstrukturbasierte Alignment in das sekundärstrukturbasierten integriert. Bei dem in Kapitel 3 beschriebenen Verfahren zum sekundärstrukturbasierten Alignment wird die Sequenz durch ein primärstrukturbasiertes Prealignment in ausgerichtete und unausgerichtete Bereiche aufgeteilt. Dadurch kann die Suche nach der Sekundärstruktur auf die unausgerichteten, variablen Bereiche beschränkt werden. Wird dieses Prealignment zu einem vollständigen primärstrukturbasierten Alignment erweitert, könnten nun diese Bereiche noch weiter in Regionen unterteilt werden, in denen die einzelnen Helixhälften am wahrscheinlichsten zu finden sind. Abbildung 5.5 soll dies verdeutlichen.

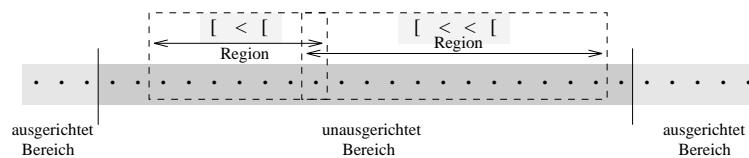


Abbildung 5.5: Weitere Unterteilung der unausgerichteten Bereiche in Regionen hoher Wahrscheinlichkeit für die einzelnen Helixhälften anhand des primärstrukturbasierten Alignment

Auf diese Weise könnte die Anzahl der Möglichkeiten für die gesuchten Helices stark eingeschränkt werden, was den Algorithmus wesentlich beschleunigen würde. Die Ermittlung der Regionen könnte z.B. so erfolgen,

daß die Positionen der realisierten Helices der Familiensequenzen in dem primärstrukturbasierten Alignment ermittelt und nach links und rechts erweitert wird, wobei die Länge der Erweiterung durch eine gewisse Anzahl von Fullzeichen beschränkt wird, also von den Insertionen in der Umgebung der Helices abhängt. Dem Kostenmodell für ein vollständiges Alignment wird bei dieser Methode vollkommen entsprochen, da beide Alignment-Typen komplett berücksichtigt werden.

5. Integration der Sekundärstruktur in die Primärstruktursuche

Die letzte Methode, die hier vorgeschlagen werden soll ist zugleich die vielversprechendste. Sie funktioniert genau entgegengesetzt zu der vorherigen. Das sekundärstrukturbasierte Alignment wird dabei bei der Ermittlung des primärstrukturbasierten verwendet. Ist die Sekundärstruktur einer Sequenz bekannt, kann sie in dem in Kapitel 2 vorgestellten Verfahren zur Anpassung der Kosten in den Helixbereichen, wie in Abbildung 5.6 dargestellt, herangezogen werden.

Dabei sollten die Basenpaarungen berücksichtigt werden. Erreichen läßt sich dies, indem man die jeweils zugehörige andere Helixhälfte gespiegelt über den aktuellen Bereich legt. Jetzt können die Kosten so verändert werden, daß innerhalb der Helixbereiche die Sequenz bevorzugt entlang der Basenpaarungen ausgerichtet wird. Dadurch wird die zweite Helixhälfte automatisch mit ausgerichtet. Um allerdings zu vermeiden, daß der zweiten Helixhälfte dadurch eine Ausrichtung aufgezwungen wird, die vielleicht gar nicht in ihre Umgebung paßt, sollte der Vorgang für jede Helixhälfte einzeln in ihrer eigenen Umgebung durchgeführt werden, wodurch jede Helixhälfte zweimal ausgerichtet wird. Diese Methode entspricht ebenso wie die vorangegangenen dem Kostenmodell für ein vollständiges Alignment. Darüberhinaus läßt sie sich gut in den mathematisch sehr sauberen Algorithmus von Needleman und Wunsch [SNee] integrieren, weswegen sie hier favorisiert wird.

Abschließend soll noch einmal bemerkt werden, daß die vorgestellten Methoden zur Kombination von primär- und sekundärstrukturbasiertem Alignment zu einem vollständigen Alignment als Vorschläge zu verstehen sind. Es wird kein Anspruch auf Vollständigkeit erhoben noch eine Garantie der Eignung dieser Methoden abgegeben. Sie sollen eine Synthese aus dem in dieser Arbeit und dem in [BRei] entwickelten Verfahren zu einem vollständigen Alignment ermöglichen.

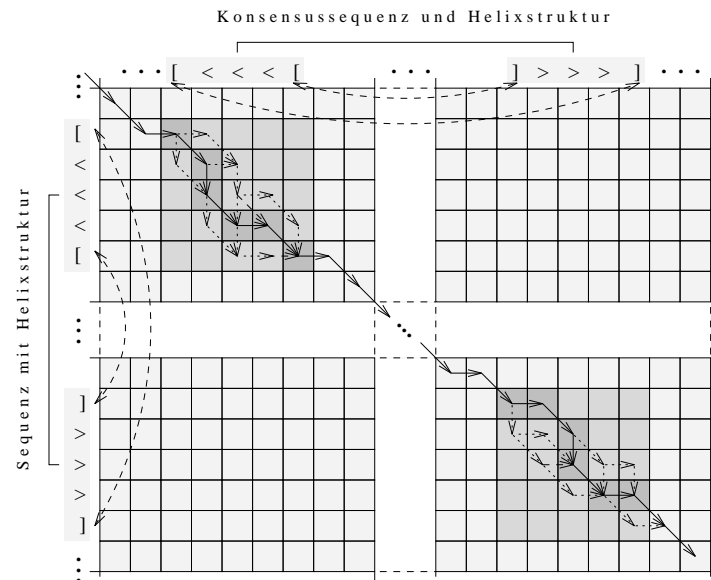


Abbildung 5.6: Anpassung der Kosten für das primärstrukturbasierte Alignment in den Helixbereichen anhand des sekundärstrukturbasierten Alignment unter Berücksichtigung der Basenpaarungen

Kapitel 6

Zusammenfassung und Ausblick

Das in dieser Arbeit entwickelte und implementierte Verfahren ermittelt auf teilweise heuristischem Weg die Sekundärstruktur einer rRNS-Sequenz unter Zuhilfenahme der Informationen des Alignment, in das sie eingefügt werden soll. Darüber hinaus liefert es eine Zuordnung der Strukturelemente der Sequenz zu denen der ihr am nächsten verwandten Sequenzen, wodurch ein Ausrichtung der Sequenz in dem Alignment ermöglicht wird. Voraussetzungen sind dabei eine Methode zur Ermittlung ihrer nächsten Verwandten im Alignment - ihrer Sequenzfamilie - und eine Methode für ein primärstrukturbasiertes Prealignment ihrer hoch konservativen Bereiche. Beide Verfahren entstammen der Arbeit [BRei]. Aus der Sequenzfamilie wird eine diskrete Konsensussequenz berechnet, mit der die realisierten Helices der Familie durch Vergleich mit der Helixstruktur des Alignments ermittelt werden. Durch ein primärstrukturbasiertes Prealignment der Sequenz können ihre hoch konservativen und somit auch ihre variablen Bereiche gefunden werden. Über die variablen Bereiche werden nun, um die anschließende Suche möglichst effizient zu gestalten, Positionsbaume aufgebaut. Ein speziell angepasster Suchalgorithmus ermittelt dann aus den Suchbäumen alle möglichen Helices der variablen Bereiche. Dabei werden nur solche berücksichtigt, deren Längen sich innerhalb gewisser Grenzen befinden und deren mittleren Bindungsstärken nicht unter einem gewissen Wert liegen. Diese Einschränkungen hängen von den realisierten Helices dieser Bereiche in der Konsensussequenz ab. Aus den für jeden Bereich nun vorliegende Menge möglicher Helices werden zu jeder realisierten Helix die ihr ähnlichsten ausgewählt. Dazu war die Entwicklung eines Maßes für die Ähnlichkeit zweier Helices notwendig. Das Ziel besteht darin, jeder realisierten Helix genau eine ihrer möglichen Helices zuzuordnen, wobei es weder zu positionellen noch zu strukturellen Konflikten unter den möglichen Helices kommen darf. Dennoch sollen die ausgewählten möglichen Helices die größtmögliche Ähnlichkeit zu denen ihnen zugeordneten realisierten Helices aufweisen. Es hat sich gezeigt, daß die Komplexität dieses Problems exponentiell mit der Zahl der realisierten Helices steigt. Deswegen wurde zu seiner Lösung auf einen Mutations-Selektions-

Algorithmus zurückgegriffen. Liegt schließlich die konfliktfreie Auswahl vor, so repräsentiert sie zum einen die wahrscheinlichste Sekundärstruktur der Sequenz und kann zum anderen zu ihrer Ausrichtung im Alignment verwendet werden.

Mit dem primärstrukturbasierten Alignment, das in [BRei] entwickelt wurde, stehen damit zwei Verfahren zur Verfügung, die die beiden für ein Alignment entscheidenden Strukturmerkmale berücksichtigen. Bei dem vorliegenden wäre allerdings noch zu untersuchen, inwieweit die optimierende Auswahl der möglichen Helices an Stelle eines Mutations-Selektions-Algorithmus mit z.B. genetischen Algorithmen sinnvoll ist und welche Verbesserungen sie einbringt. Da beide Verfahren nur einen Teilaspekt der Alignment-Problematik beleuchten, wäre der nächst logische Schritt eine Kombination beider Verfahren bzw. eine Einbindung des einen in das andere. Das Ziel ist ein vollständiges Alignment, in dem sowohl die Primär- als auch die Sekundärstrukturinformation vollständig ausgenutzt wird.

Literaturverzeichnis

- [AAho] Alfred V. Aho, Ravi Sethi, Jeffrey D. Ullman: *Compilerbau (Teil 1)*, Addison-Wesley, 1988.
- [BAIb] Bruce Alberts, Dennis Bray, Julian Lewis, Martin Raff, Keith Roberts and James D. Watson: *Molecular biology of the cell*, Garland Publishing Inc., 1983.
- [MBre] Manfred Bretz: *Algorithmen und Berechenbarkeit* Vieweg, 1992.
- [SGru] Silke Grumann: *Untersuchungen zum automatischen Alignment von ribosomalen RNS-Sequenzen*, Diplomarbeit, Institut für Informatik, Technische Universität München, 1993.
- [JHop] John E. Hopcroft, Jeffrey D. Ullman: *Einführung in die Automaten-theorie, formale Sprachen und Komplexitätstheorie*, Addison-Wesley, 1994.
- [MKem] Michael Kempf: *Positionsbäume, Index-Darstellung von Zeichenreihen, Baufbau und Korrektur*, Dissertation, Institut für Informatik, Technische Universität München, 1987.
- [WKin] Werner Kinnebrock: *Optimierung mit genetischen und selektiven Algorithmen*, Oldenbourg, 1994.
- [EKre] Erwin Kreyszig: *Statistische Methoden und ihre Anwendungen*, Vandenhoeck & Ruprecht, 1975.
- [AKol] Albert Kollman: *Einführung in die Genetik*, Diesterweg Salle Sauerländer, 1984.
- [SNee] Saul B. Needleman, Christian D. Wunsch: *A General Method Applicable to the Search for Similarities in the Amino Acid Sequence of Two Proteins*, J. Mol. Biol. 48, 1970.
- [BRei] Boris Reichel: *Automatisches Alignment von ribosomalen RNS-Sequenzen zur Unterstützung phylogenetischer Analysen von Organismen*, Diplomarbeit, Institut für Informatik, Technische Universität München, 1994.

- [KRei] Karl Rüdiger Reischuk: *Einführung in die Komplexitätstheorie*, B. G. Teubner Stuttgart, 1990.
- [DSan] David Sankoff, Joseph B. Kruskal: *Time warps, string edits, and macromolecules: The theory and practice of sequence comparison*, Addison-Wesley, 1983.
- [KSch] K. H. Schleifer, W. Ludwig: *Phylogenetic relationships among bacteria; The Hierarchy of Life*, Elsevier Science Publishers B.V., 1989.
- [RSed] Robert Sedgewick: *Algorithmen*, Addison-Wesley, 1992.
- [CWoe1] C. R. Woese: *Bacterial evolution*, Microbiol.Rev. 51, S.221–271, 1987.
- [CWoe2] C. R. Woese, Robin R. Gutell: *Evidence for several higher order structural elements in ribosomal RNA*, Proc. Natl. Acad. Sci. USA 86, 1989.