Fachhochschule Wedel

Seminararbeit

in der Fachrichtung
Informatik
WS 2019/20

Suchverfahren für Datenbanken im Bereich der Bioinformatik

J. Neumann

Erarbeitet im 5. Fachsemester
Abgegeben am 20. Februar 2020

Inhaltsverzeichnis

1 Einleitung und Motivation **3**

2 Hintergrundinformationen **4**
2.1 *Desoxyribonukleinsäure* (DNA) und *Ribonukleinsäure* (RNA) Struktur 5
2.2 Protein Struktur 6

3 Sequenz *Alignment* **7**

4 Suchverfahren **9**
4.1 Der Smith-Waterman Algorithmus 10
4.2 BLAST - Ein Überblick 10
4.2.1 BLAST - Das Verfahren im Detail 11
4.3 Weitere Suchverfahren 14
4.4 Das FastA Dateiformat 15
4.5 BLAST praktisch anwenden 15

5 Datenmanagement - Gegenwart und Zukunft **16**

Abkürzungsverzeichnis

A *Adenin*

ASN.1 *Abstract Syntax Notation 1*

BLAST *Basic Local Alignment Search Tool*

BLAT *BLAST-like alignment tool*

C *Cytosin*

DNA *Desoxyribonukleinsäure*

G *Guanin*

HSP *high-scoring segment pair*

HSPs *high-scoring segment pairs*

MSP *maximal-scoring segment pair*

NCBI *National Center for Biotechnology Information*

NCBI RefSeq *Reference Sequence database des National Center for Biotechnology Information*

NoSQL *Not only Structured Query Language (SQL)*

PSI-BLAST *Position-Specific Iterated BLAST*

PSSM *profile/position-specific scoring matrix*

RNA *Ribonukleinsäure*

SQL *Structured Query Language*

T *Thymin*

U *Uracil*

XML *Extensible Markup Language*

1 Einleitung und Motivation

A, C, G und T. Vier einfache Buchstaben als Stellvertreter für die Vielseitigkeit des Lebens. Diese Vielseitigkeit reicht von winzigen Bakterien bis hin zu den riesigsten Säugetieren.[1] Sie definiert von zunächst simpel erscheinenden Strukturen, wie zum Beispiel einem Knochen, bis hin zum zentralen Nervensystems des Menschen, dessen gesamte Komplexität noch nicht komplett erfasst wurde, alle Komponenten, die ein Organismus zum Leben benötigt. Obwohl sich das Leben auf manigfaltige Art und Weise manifestiert und dadurch weitreichende Unterschiede bei der Betrachtung verschiedenster Lebensformen zu beobachten sind, lassen sich die Zusammenhänge doch auf vier Buchstaben zurückführen. Denn auf der untersten Ebene besteht jede Lebensform doch aus den selben Bausteinen: A, C, G und T.[1] Diese vier Buchstaben stehen für die vier verschiedenen Basen der *Desoxyribonukleinsäure* (DNA), also für *Adenin* (A), *Cytosin* (C), *Guanin* (G) und *Thymin* (T).

Aber welche Rolle spielt die Informatik in diesem biologisch geprägten Bereich? Und wo bestehen Schnittstellen zwischen den beiden Fachgebieten? Eine Anwendung der Informatik in der Biologie wird dem interessierten Leser hier präsentiert: *Suchverfahren für Datenbanken im Bereich der Bioinformatik.*

Das Ziel dieser Arbeit liegt darin, ein Überblick über Suchverfahren für Datenbanken im Bereich der Bioinformatik zu geben. Dafür wird in diesem Kapitel zunächst aufgezeigt, welche Arten von Daten in der Bioinformatik gespeichert werden und mit welcher Motivation diese analysiert werden. In den folgenden Kapiteln werden Informationen zu den biologischen und teilweise auch chemischen Grundlagen gegeben, bevor die Hintergründe von Suchvorgängen und deren Ausprägung mit Hilfe eines spezifischen Suchverfahrens aufgezeigt werden. Im letzten Kapitel folgt dann ein Ausblick auf zukünftige Herausforderungen, die dem Fachbereich der Bioinformatik, im Hinblick auf den umfassenden Umgang mit Daten, bevorstehen.

Von den Unmengen an Daten, die in der Biologie potenziell in Datenbanken gespeichert werden könnten, spielen in der Bioinformatik die Speicherung von DNA Sequenzen mit deren Basenreihenfolge und Protein Sequenzen, beziehungsweise deren Aufbau aus Aminosäuren, eine bedeutende Rolle. Suchverfahren für Datenbanken dieser Art verfolgen das Ziel, Übereinstimmungen, sogenannte *Alignments*, zwischen Sequenzen zu identifizieren. Diese Suche nach Ähnlichkeiten gehört zu den wichtigsten Aufgaben in der Bioinformatik und anderen Lebenswissenschaften.[2] Aber woher rührt die Relevanz, Datenbanken nach solchen Treffern zu durchsuchen? Was motiviert Wissenschaftler, lineare Folgen verschiedener Säuren und Basen zu analysieren?

Vor allem Forschungsarbeiten aus dem 20. Jahrhundert haben den Zusammenhang zwischen der DNA, den Proteinen und der Vielseitigkeit des Lebens offengelegt.[1] Der Vergleich von Sequenzen erlaubt es Gene, den Abschnitt unserer DNA, der die Informationen zur Entwicklung und zu den Eigenschaften eines Individuums speichert,[3] zu identifizieren und andere Sequenzmuster zu erkennen. Dieser Vergleich stellt dann eine reliable Methode dar, um die biologische Funktion neu entdeckter Gene zu erschließen. Man kann zum Beispiel analysieren, ob gewisse Lebewesen einen bestimmten Anteil ihrer Gene mit anderen Lebewesen teilen um so die Evolution und Funktion biomolekularer Sequenzen zu untersuchen.[4]

Da sich die meisten physiologischen und pathologischen Prozesse auf der Ebene der Proteine manifestieren, haben Wissenschaftler aus der Biologie und der Biomedizin zunehmendes Interesse daran, durchsatzfähige Techniken zu entwickeln, die darauf abzielen die Struktur und Funktion von Proteinen zu untersuchen, um ein besseres Verständnis für die grundlegende Molekularbiologie und den Entwicklungsprozess von Krankheiten zu erhalten.[5] Im Laufe der Zeit haben sich große Mengen an Informationen über die Sequenzen von DNA, RNA und Proteinen

[1] Zvelebil, *Understanding Bioinformatics*.
[2] Robert Vaser, "SWORD — a highly efficient protein database search".
[3] Wikipedia, *Gen*.
[4] Chao, *Sequence Comparison: Theory and Methods*.
[5] Chuming Chen, "Protein Bioinformatics Databases and Resources".

angesammelt, sodass die Nutzung elektronischer Datenbanken zum Speichern und Analysieren dieser Daten unabdingbar ist.[6] Die Menge aller biologischer Daten verdoppelt sich alle fünfzehn bis sechzehn Monate und täglich erfolgen ungefähr 40.000 Anfragen an Datenbanken aus dem Bereich der Bioinformatik.[7] Dieser Zuwachs an Daten lässt sich zum Beispiel an der Entwicklung der gespeicherten Proteine in der *Reference Sequence database des National Center for Biotechnology Information* (NCBI RefSeq), einer häufig genutzten Datenbank für Sequenzen, wie in Abbildung 1 dargestellt, beobachten.

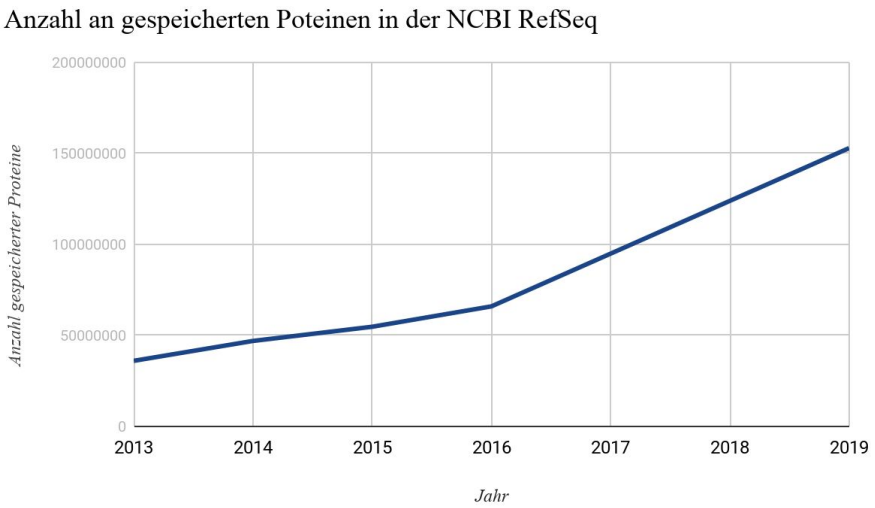

Abbildung 1: Anzahl der in der NCBI RefSeq gespeicherten Proteine. Daten der verschiedenen Jahre entnommen von NCBI, *RefSeq: NCBI Reference Sequence Database*, NCBI, *RefSeq Announcements for 2016*, NCBI, *RefSeq Announcements for 2015*, NCBI, *RefSeq Announcements for 2014* und NCBI, *RefSeq Announcements for 2013*.

2 Hintergrundinformationen

In diesem Kapitel werden vor allem die relevanten biologischen, aber auch chemischen Hintergründe erläutert, damit der Leser einen Zusammenhang zwischen den in der Datenbank gespeicherten Informationen und deren Ursprung knüpfen kann.

Trotz der Vielzahl an Lebewesen auf der Erde, weisen alle Organismen doch eine erstaunliche Ähnlichkeit auf. Jeder Organismus besteht aus Zellen und ist in erster Linie von zwei Arten von Molekülen abhängig. Zum Einen von der *Desoxyribonukleinsäure* (DNA) und zum Anderen von Proteinen. Die DNA ist eine Nukleinsäure, die sich aus einer Kette von verschiedenen Nukleotiden zusammensetzt.[8] Im Laufe der Zeit gibt sich die DNA eines Lebewesens selbstständig von Generation zu Generation weiter und speichert dabei die Informationen, die zur Synthese anderer Proteine und Moleküle genutzt wird.

Proteine bestehen im Gegensatz zur DNA aus einer Folge verschiedener Aminosäuren und haben wiederum die Aufgabe biochemische Reaktionen auszulösen, Signale zwischen Zellen weiterzuleiten und die Bestandteile eines Organismus zu bilden.[9] Die DNA speichert also Informationen, die benötigt werden um Proteine zu generieren. Proteine übernehmen im Organismus dann die Aufgaben, von denen das Leben als solches abhängig ist. Die Vielseitigkeit des Lebens,

[6] Zvelebil, *Understanding Bioinformatics*.
[7] Sung, *Algorithms in Bioinformatics: A Practical Introduction - Database Search*.
[8] Wikipedia, *Desoxyribonukleinsäure*.
[9] Chao, *Sequence Comparison: Theory and Methods*.

vom einfachsten Bakterium bis hin zum größten Säugetier, ist somit in einer linearen Sequenz in jeder lebenden Zelle gespeichert. Informationen über Nukleinsäuren und Proteine stellen daher das grundlegende Arbeitsmaterial für Forscher im Bereich der Bioinformatik dar.[10]

2.1 *Desoxyribonukleinsäure* (DNA) und *Ribonukleinsäure* (RNA) Struktur

Die DNA besteht aus einer linearen Folge von vier verschiedenen Nukleotiden, die sich lediglich in einer untergeordneten Struktur, der Base, unterscheiden. Auch wenn der chemische Aufbau dadurch relativ überschaubar ist, liegt die Komplexität der Struktur in seiner beeindruckenden Länge, die mehrere Millionen Basen pro DNA Molekül umfassen kann, wobei diese Basensequenz die relevante Information zum Erzeugen von Proteinen darstellt. Alle Informationen, die benötigt werden um den menschlichen Organismus in seiner gesamten Komplexität darzustellen, sind in gerade mal 23 Paaren von DNA Molekülen enthalten. Diese Moleküle gehören zu den größten und längsten bisher bekannten Molekülen, wobei die kleinsten 47 Millionen und die Größten 247 Millionen Basen umfassen. Die vom Aufbau verwandte *Ribonukleinsäure* (RNA) ist im Gegensatz zur DNA ein deutlich kleineres Molekül. In den meisten Fällen umfasst auch sie lediglich vier verschiedene Basen, wobei es Ausnahmefälle gibt, in der sie auch Anteile anderer Basen halten kann.[10]

Ein Nukleotid aus einer DNA, beziehungsweise RNA Kette besteht aus der bereits angesprochenen Base, einem Zucker Anteil und einer Phosphat Gruppe. Im Falle der DNA stellt Desoxyribose den Zucker Abschnitt dar, wohingegen es sich bei dem Zucker Anteil der RNA um Ribose handelt. Innerhalb der DNA treten die vier Basen *Guanin* (G), *Adenin* (A), *Cytosin* (C) und *Thymin* (T) auf. Wie bereits angesprochen, bestehen auch die meisten RNA Moleküle aus vier verschiedenen Basen. Im Gegensatz zur DNA tritt Anstelle des *Thymin* jedoch *Uracil* (U) als vierte Base auf. Die lineare Verknüpfung einer Nukleotidkette findet jeweils zwischen der Phosphat Gruppe des einen Nukleotids und dem Zucker Anteil des darauf folgenden Nukleotids statt (Abbildung 2). Entsprechend der Reihenfolge der Basen kann dann die entsprechende DNA oder RNA Sequenz ausgelesen werden.[10]

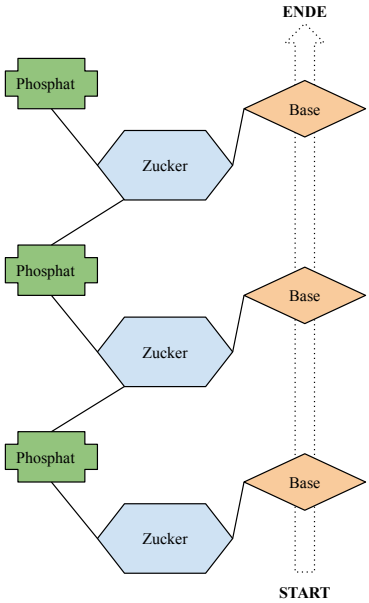

Abbildung 2: Schematischer Aufbau einer Nukleotidkette.

[10]Zvelebil, *Understanding Bioinformatics*.

Die bekannte räumliche Struktur eines DNA Doppelstrangs entsteht, da sich zwei DNA Stränge mit jeweils komplementäre Basenpaare über Wasserstoffbrückenbindungen miteinander verbinden, wobei die Basen jeweils auf der Innenseite beider Stränge liegen und die Zucker-Phosphat Verbindungen auf der Außenseite. Die gebildeten Basenpaare entstehen dabei nach einem festen Muster. Es wurde herausgefunden, dass sich jeweils *Adenin* mit *Thymin* und *Cytosin* mit *Guanin* verbindet. Durch diese komplementäre Abhängigkeit ist man also dazu in der Lage aus einem DNA Strang die dazugehörige Nukleotidkette abzuleiten. Dabei muss jedoch berücksichtigt werden, dass beide Stränge eines DNA Moleküls antiparallel sind, da sie in unterschiedlich Richtungen laufen. Die komplementäre Basensequenz zu `AAG` lautet also nicht `TTC`, sondern `CTT` (Abbildung 3).[11]

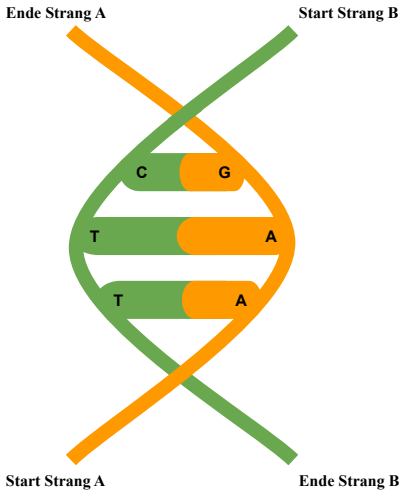

Abbildung 3: Schematischer Aufbau einer DNA Doppelhelix.

DNA Moleküle haben das Potenzial hunderttausende Proteine zu kodieren, wobei ein einzigartiges Segment der DNA ein Protein spezifiziert. Einfach ausgedrückt könnte man sagen, dass ein jedes solcher Segmente das Gen für ein Protein repräsentiert. In der Realität ist es jedoch so, dass ein Gen von Abschnitten umgeben ist, die nicht kodiert sind, die wiederum bestimmen, ob das Gen überhaupt aktiv ist und in welchen Fällen letztendlich ein Protein produziert wird. Der entsprechende Abschnitt, der die Produktion eines Proteins hervorrufen kann, wird auch der genetische Code genannt. Der genetische Code wird in zwei Phasen, durch die Transkription und die Translation entschlüsselt. Für diesen Prozess werden die erwähnten RNA Moleküle benötigt.[11]

2.2 Protein Struktur

Die Primärstruktur, die Sekundärstruktur, die Tertiärstruktur und die Quartärstruktur stellen die verschiedenen Ebenen der Proteinstruktur dar. Für diese Arbeit spielt die Primärstruktur die größte Rolle, da auf diesem Level die Proteinsequenz gespeichert ist. Dabei handelt es sich um eine lineare Folge von Aminosäuren, die ähnlich wie die Basensequenz der DNA, in Datenbanken abgelegt werden kann, um dann auf Basis von Fragestellungen aus dem Bereich der Bioinformatik analysiert zu werden. Abhängig von dieser Sequenz bildet ein Protein eine dreidimensionale Struktur aus, die durch das Falten, beziehungsweise Zusammenlegen sich wiederholender Strukturen, den so genannten sekundär Strukturen, entsteht. Diese Faltung bestimmt die Funktionsweise des Proteins weshalb das Verstehen und das Vorhersagen der Faltung aus der Sequenz des Proteins, in der Bioinformatik eine wichtige Rolle spielt und zu den schwierigsten Aufgaben dieser Disziplin gehört. Die Tertiärstruktur spezifiziert diese Faltung und mit der Quartärstruktur

[11] Zvelebil, *Understanding Bioinformatics*.

wird das Zusammenschließen mehrerer Protein-Untergruppen beschrieben. Proteine bestehen aus zwanzig unterschiedlichen Typen von Aminosäuren. Diese wiederum bestehen ausschließlich aus Kohlenstoff, Stickstoff, Sauerstoff und Wasserstoff und in Ausnahmefällen auch aus Schwefel. Die Primärstruktur eines Proteins entsteht, indem sich mehrere Aminosäuren über Peptidverbindungen miteinander verbinden.[12]

3 Sequenz *Alignment*

In Kapitel 1 und 2 wurden Gründe aufgezeigt, warum Wissenschaftler ein Interesse daran haben Sequenzen nach Übereinstimmungen, sogenannten *Alignments*, zu durchsuchen. In diesem Kapitel wird dargestellt, worum es sich bei diesen *Alignments* eigentlich handelt.

Gegeben sei ein Alphabet Z und die beiden Sequenzen X und Y, die durch Z definiert sind. Um ein *Alignment* zwischen beiden Sequenzen zu bilden wird ein zusätzliches Symbol zu dem Alphabet hinzugefügt, zum Beispiel ein '-'. Dieses Symbol wird verwendet, um Leerräume in einer Sequenz abzubilden. Ein mögliches *Alignment* beider Sequenzen wird dann dargestellt, indem eine zweizeilige Matrix gebildet wird, die die folgenden Eigenschaften erfüllt:

1. Die Buchstaben aus Sequenz X und Y werden in geordneter Reihenfolge in der ersten, beziehungsweise zweiten Zeile der Matrix ausgegeben.

2. In jeder Zeile dürfen eine beliebige Anzahl von Leerräumen zwischen zwei Buchstaben aus Z vorkommen.

3. Jede Spalte der Matrix enthält mindestens einen Buchstaben aus Z.[13]

Betrachtet man beispielhaft zwei kurze DNA Sequenzen, die durch eine, wie in Kapitel 2.1 beschrieben, gültige Basensequenz repräsentiert werden, kann ein gültiges *Alignment* folgendermaßen dargestellt werden:

$X =$ `T A T G A C C`
$Y =$ `C A G G C A C`

Mögliche *Alignment* Matrix für die Sequenzen X und Y:

```
T - A T G - A C C
- C A G G C A - C
```

Bei der Gegenüberstellung zweier Sequenzen kann jede Spalte der entstandenen Matrix durch ein bestimmtes Ereignis beschrieben werden. Befinden sich zwei Buchstaben in einer Spalte, handelt es sich bei identischen Buchstaben um ein *Match*, also eine Übereinstimmung und bei unterschiedlichen Buchstaben um ein *Mismatch*. Letztere können, von einem biologischen Standpunkt aus gesehen, auch als *Substitution* bezeichnet werden, also der Fall, dass zum Beispiel eine Base einer DNA Sequenz durch einen Anderen ersetzt wurde (Tabelle 1).

T	-	A	T	G	-	A	C	C
-	C	A	G	G	C	A	-	C

Tabelle 1: Darstellung der Ereignisse *Match* und *Mismatch* bei dem *Alignment* zwischen Sequenz X und Y.

[12] Zvelebil, *Understanding Bioinformatics*.
[13] Chao, *Sequence Comparison: Theory and Methods*.

Spalten die einen Leerraum beinhalten werden *Indels* genannt, wobei eine weitere Spezialisierung in *Insertions* und *Deletions* erfolgen kann. Bei *Insertions* befindet sich bei der Buchstabenpaarung ein Leerraum in der ersten Zeile und bei *Deletions* ein Leerraum in der zweiten Zeile. In der Literatur und somit auch in dieser Arbeit wird häufig zwischen *gapped* und *ungapped Alignments* unterschieden. Bei *gapped Alignments* sind *Indels* im Sequenzvergleich erlaubt, bei *ungapped Alignments* entsprechend nicht (Tabelle 2).[14]

T	-	A	T	G	-	A	C	C
-	C	A	G	G	C	A	-	C

Tabelle 2: Darstellung der Ereignisse *Insertion* und *Deletion* bei dem *Alignment* zwischen Sequenz X und Y.

Wie bereits beschrieben, stellt das soeben dargestellte *Alignment* nur eine Möglichkeit dar, die beiden Sequenzen X und Y miteinander zu vergleichen. Eine Problemstellung in der Bioinformatik besteht häufig darin, das beste *Alignment* oder je nach Verfahren, die besten *Alignments* zu selektieren. Wie in Tabelle 3 dargestellt, entsteht bei dem Vergleich von zwei Sequenzen mit jeweils gerade einmal acht Elementen bereits eine umfangreiche Menge an möglichen *Alignments*. Je länger also beide Sequenzen sind, desto mehr Kandidaten gibt es für das Beste beziehungsweise die besten *Alignments*. Voraussetzung dafür, dass effiziente Algorithmen überhaupt gewisse Sequenzen selektieren können, ist die Anwendung verschiedener Bewertungsregeln, um eine Aussage darüber zu treffen, welche *Alignments* die größte Bedeutung haben.[14]

	0	1	2	3	4	5	6	7	8
0	1	1	1	1	1	1	1	1	1
1		3	5	7	9	11	13	15	17
2			13	25	41	61	85	113	145
3				63	129	231	377	575	833
4					321	681	1289	2241	3649
5						1683	3653	7183	13073
6							8989	19825	40081
7								48639	108545
8									265729

Tabelle 3: Anzahl möglicher *Alignments* in Abhängigkeit von der Sequenzlänge. In der ersten Zeile und der ersten Spalte werden die Wortlängen beider Sequenzen abgebildet. Inhalte entnommen aus Chao, *Sequence Comparison: Theory and Methods*.

Dafür werden Punktewerte eingeführt, die ein bestimmtes Ereignis bewerten, das bei dem Vergleich von zwei Buchstaben auftreten kann. Normalerweise wird ein *Match*, mit einem positiven Wert und ein *Mismatch* mit einem negativen Wert beurteilt. Wendet man diese Regel auf ein Alphabet an, kann aus den Ergebnissen eine Matrix, eine sogenannte *Scoring Matrix* generiert werden. In Abbildung 4 ist eine mögliche *Scoring Matrix* zu sehen, die bei der Bewertung von *Alignments* zwischen DNA Sequenzen Anwendung finden könnte. Aus Abbildung 4 geht hervor, dass ein *Match* mit zwei Punkten bewertet wird und das für ein *Mismatch* ein Punkt abgezogen wird.[14]

[14] Chao, *Sequence Comparison: Theory and Methods*.

A	2	-1	-1	-1
G	-1	2	-1	-1
C	-1	-1	2	-1
T	-1	-1	-1	2
	A	G	C	T

Abbildung 4: Beispielhafte *Scoring Matrix* für einen Vergleich von DNA Sequenzen. Inhalt entnommen aus Chao, *Sequence Comparison: Theory and Methods*.

Des Weiteren muss noch der Umgang mit *Indels* definiert werden. Anders als bei einem *Mismatch* wird ein *Indel* nicht mit einem festen Wert bestraft, da Nukleotide im Zusammenhang mit biochemischen Prozessen, wie zum Beispiel der DNA Replikation, häufig aus Sequenzen entfernt oder hinzugefügt werden. Ein mögliches Verfahren um Punkte für *Indels* abzuziehen, stellt die *affine gap penalty* Methode dar. Dabei wird der Punkteabzug für eine Lücke der Länge k folgendermaßen berechnet:

```
-(o+k*e)
```

Dabei steht o für den Punktewert, der abgezogen wird, wenn eine *Gap* eröffnet wird und e für den Wert der berechnet wird, wenn eine Lücke um ein Element verlängert wird. Der Wert für die Eröffnung wird meistens größer gewählt, als der Wert für das Erweitern.

Während die Erstellung einer *Scoring Matrix* zur Bewertung von *Alignments* zwischen DNA Sequenzen relativ leicht ist, da alle verschiedenen Treffer gleich bewertet werden, ist der Aufbau einer *Scoring Matrix* für Proteinsequenzen deutlich komplizierter. Bewertungsregeln für Proteinsequenzen basieren auf statistischen Auswertungen realer Sequenzen. Dabei wird die Wahrscheinlichkeit berücksichtigt, wie Aminosäuren, im Zuge biologischer Prozesse, durch andere Aminosäuren ersetzt werden. Daher wird eine *Scoring Matrix*, die für die Berechnung des *Alignments* von Proteinsequenzen genutzt wird, für gewöhnlich als *Substitution Matrix* bezeichnet.[15]

4 Suchverfahren

Die Informationen aus Kapitel 3 beschreiben, wie zwei Sequenzen verglichen werden können. Wie bereits angedeutet, gibt es spezielle Algorithmen, die sehr genau dazu in der Lage sind das optimale *Alignment* zwischen zwei Sequenzen zu berechnen. Wegen der riesigen Masse an biologisch relevanten Daten, sind diese Verfahren jedoch nicht dazu geeignet, um sie als Suchverfahren für große biologische Datenbanken zu verwenden. Für die in Kapitel 1 beschriebenen Datenmengen werden schnellere Verfahren benötigt.[15]

Die Voraussetzung für eine Suchanfrage an Datenbanken im Bereich der Bioinformatik ist mindestens eine Datenbank, die DNA oder Proteinsequenzen enthält und ein Suchstring, der die Anfrage an die Datenbank definiert. Grundlegend besteht das Ziel der Suchanfrage darin, den String in der Datenbank zu finden, der die beste Übereinstimmung mit dem Suchstring aufweist.

Die Güte der Algorithmen, die in verschiedenen Suchverfahren genutzt werden, kann über die Sensitivität und die Spezifität gemessen werden. Die Sensitivität beschreibt die Fähigkeit des Algorithmus, *richtig-positive* Ergebnisse zu erkennen. Sie kann als die Wahrscheinlichkeit ausgedrückt werden einen Treffer zu finden, wenn gegeben ist, dass der Suchstring und die Sequenz in der Datenbank nur einen gewissen Prozentsatz an Gleichheit aufweisen. Die Spezifität hingegen beschreibt die Fähigkeit des Algorithmus, *falsch-positive* Werte zu identifizieren und abzulehnen. Die Spezifität steht somit im Zusammenhang mit der Effizienz des Algorithmus.[16]

[15] Chao, *Sequence Comparison: Theory and Methods*.
[16] Sung, *Algorithms in Bioinformatics: A Practical Introduction - Database Search*.

Eine grundlegende Methode, um die Laufzeit von Suchverfahren zu verbessern, besteht darin, die Datenbanksequenz nach exakten oder ungefähren Übereinstimmungen mit Wörtern aus dem Suchstring zu durchsuchen. Die gefundenen Treffer dienen dann als Grundlage, um zu beurteilen inwiefern zwei Sequenzen eine bedeutende Übereinstimmung aufweisen, indem evaluiert wird, ob die gefundenen Ergebnisse einen bestimmten Schwellenwert überschreiten. Dabei wird über die positive Zahl k die Wortlänge für die Prüfung festgelegt.[17]

4.1 Der Smith-Waterman Algorithmus

Der Smith-Waterman Algorithmus ist ein Algorithmus, der zu der Gruppe der *Brute-Force* Algorithmen gezählt werden kann. Als Input dient dem Algorithmus eine Datenbank d mit einer totalen Länge n und ein Suchstring q mit der Länge m. Der Algorithmus sucht für jede Sequenz s aus der Datenbank d die beste lokale Übereinstimmung zwischen s und q und gibt dann alle besten Ergebnisse zurück. Aufgrund dieser Vorgehensweise ergibt sich für die Laufzeit des Algorithmus die Komplexitätsklasse $\mathbf{O}(nm)$.

Der nun folgende Code-Ausschnitt beschreibt in der Programmiersprache Java die grundlegende Arbeitsweise des Smith-Waterman Algorithmus:

```
// ...
Container c = new Container();
for (Sequence s : d) {
        s.useSmithWaterman(q, c);
}
return c;
// ...
```

Dadurch, dass der Algorithmus in die Klasse der *Brute-Force* Algorithmen eingeteilt werden kann und er somit jede Sequenz berücksichtigt, handelt es sich dabei um den sensitivsten der Algorithmen, die in dieser Arbeit vorgestellt werden. Die hohe Sensitivität bezahlt der Algorithmus jedoch mit Einbußen in der Performance.[18]

4.2 BLAST - Ein Überblick

Das *Basic Local Alignment Search Tool* (BLAST) ist das mitunter populärste Tool für eine (proteinbasierte) Datenbanksuchen.[19,20,17] Es gibt viele verschiedene BLAST Ausprägungen, die jeweils unterschiedliche Einsatzgebiete haben. Im Detail werden diese in dieser Arbeit nicht genauer erläutert, können aber als grobe Übersicht aus der Tabelle 4 entnommen werden.

Eine BLAST Suche kann zum Beispiel über die *National Center for Biotechnology Information* (NCBI) BLAST Webseite oder über eine eigenständige Applikation durchgeführt werden. Zunächst erzeugt die Suche ein strukturiertes Ergebnis im *Extensible Markup Language* (XML) oder *Abstract Syntax Notation 1* (ASN.1) Format. Dies ermöglicht sowohl eine automatische und genaue Syntaxprüfung, als auch ein schnelles Erkennen von Veränderungen. Die einzelnen Resultate innerhalb des Ergebnisses werden als C++ Objekte repräsentiert. Ein Großteil dieser Objekte können im ASN.1 Format dargestellt werden, was sich wiederum gut dazu eignet, um die Ergebnisse zu speichern, zu verteilen und bei Bedarf an der richtigen Stelle wieder zu Objekten zu deserialisieren. Die Übereinstimmungen, die zwischen der Suchsequenz und den Sequenzen aus der Datenbank gefunden wurden, werden mit dem *ASN.1 SeqAlign module* ausgegeben. Diese Ausgabe enthält Informationen über Bewertungen, weitere Parameter über den Beginn von Übereinstimmungen und deren Ende, gibt die Stellen an, an denen *insertions* oder

[17] Chao, *Sequence Comparison: Theory and Methods*.
[18] Sung, *Algorithms in Bioinformatics: A Practical Introduction - Database Search*.
[19] Robert Vaser, "SWORD — a highly efficient protein database search".
[20] Madden, "The BLAST Sequence Analysis Tool".

Name der Ausprägung	Kurze Beschreibung
megaBlast	Eine Nukleotid zu Nukleotid Suche, optimiert für ähnliche Sequenzen, zum Beispiel in der gleichen oder verwandten Spezies
BLASTN	Vergleich einer Nukleotid Sequenz gegen die *National Center for Biotechnology Information* (NCBI) Nukleotid Datenbanken
BLASTP	Ein Protein zu Protein Vergleich und Grundlage anderer BLAST Suchverfahren, wie zum Beispiel BLASTX und TBLASTN
BLASTX	Vergleicht die von einer Nukleotid Suchanfrage verschlüsselte Proteinsequenz gegen eine Protein Datenbank

Tabelle 4: Eine Auswahl von BLAST Ausprägungen und deren Bedeutung. Inhalte entnommen aus Madden, "The BLAST Sequence Analysis Tool".

deletions durchgeführt wurden und informiert den Nutzer bei einer DNA Suche darüber, auf welchem Strang die Treffer gefunden wurden.[21]

Ausgangspunkt für die Nutzung von BLAST ist eine Datenbank D, die Sequenzen beinhaltet und eine Sequenz S, nach der gesucht werden soll. Das Ziel von BLAST liegt darin, S gegen alle Sequenzen in D in angemessener Zeit zu vergleichen, Gebiete mit lokalen Übereinstimmungen zu finden und die statistische Signifikanz der gefundenen Treffer zu berechnen.[22] BLAST verfolgt dabei einen heuristischen Ansatz und bezahlt eine schnellere Rechenzeit mit verringerter Genauigkeit, wodurch die Sensitivität entsprechend sinkt.[23]

Das Suchverfahren, das hinter BLAST steht hat sich im Laufe der Zeit weiterentwickelt. Die erste Version stammt aus dem Jahr 1990 und ist dazu in der Lage *ungapped Alignments* zwischen Sequenzen zu generieren, weshalb es auch unter dem Namen *ungapped BLAST* bekannt ist. BLAST 2.0 wurde im Jahr 1997 veröffentlicht und brachte zwei neue Änderungen mit sich: eine *gapped BLAST* Methode, also ein Verfahren, das *gapped Alignments* entwickelt und trotzdem schneller arbeitet als die vorherige Variante und *Position-Specific Iterated BLAST* (PSI-BLAST), was bei der Suche nach entfernten Verwandschaftsbezügen bei der Analyse von Proteinen eingesetzt werden kann.[22]

4.2.1 BLAST - Das Verfahren im Detail

Bei dem *ungapped BLAST* Verfahren werden beide Sequenzen zunächst miteinander verglichen, indem eine zuvor festgelegte Substitutionsmatrix genutzt wird, um festzulegen, wie mit Übereinstimmungen in den Sequenzen umgegangen wird. Bei einem DNA Sequenz Vergleich kann zum Beispiel festgelegt werden, dass eine Übereinstimmung der Basen mit dem positiven Wert 5 und ein Unterschied mit dem negativen Wert -4 notiert wird. In Abbildung 5 ist der Vergleich von zwei DNA Sequenzen nach dem beschriebenen Muster zu sehen.[22]

Ausgehend von dieser Gegenüberstellung der Sequenzen wird die übereinstimmende Folge mit der besten Punktzahl, das *maximal-scoring segment pair* (MSP) berechnet, das sich aus der Summe der dazugehörigen Werte ergibt. Bei dieser Folge handelt es sich um eine diagonale und zusammenhängende Erweiterung der Partner aus beiden Sequenzen. Um das bestbewerteste MSP zu finden, könnte man die höchste Summe für jede Diagonale der Matrix berechnen. Bezogen auf eine Diagonale gibt es einen Algorithmus mit linearer Laufzeit, der diese Aufgabe bewältigen kann. Wendet man diesen Algorithmus jedoch auf alle Diagonalen der Matrix an,

[21] Madden, "The BLAST Sequence Analysis Tool".
[22] Chao, *Sequence Comparison: Theory and Methods*.
[23] Sung, *Algorithms in Bioinformatics: A Practical Introduction - Database Search*.

	C	T	A	T	C	A	T	T	C	T	G
G	-4	-4	-4	-4	-4	-4	-4	-4	-4	-4	5
A	-4	-4	5	-4	-4	5	-4	-4	-4	-4	-4
T	-4	5	-4	5	-4	-4	5	5	-4	5	-4
C	5	-4	-4	-4	5	-4	-4	-4	5	-4	-4
C	5	-4	-4	-4	5	-4	-4	-4	5	-4	-4
A	-4	-4	5	-4	-4	5	-4	-4	-4	-4	-4
T	-4	5	-4	5	-4	-4	5	5	-4	5	-4
C	5	-4	-4	-4	5	-4	-4	-4	5	-4	-4
T	-4	5	-4	5	-4	-4	5	5	-4	5	-4
T	-4	5	-4	5	-4	-4	5	5	-4	5	-4

Abbildung 5: Zwei Sequenzen in einer Matrix dargestellt und mit der vorher definierten Bewertung versehen. Inhalte übertragen aus Chao, *Sequence Comparison: Theory and Methods*.

benötigt dieser eine Zeit, proportional zu dem Produkt der Längen der beiden Sequenzen. Um die Laufzeit zu verbessern, berechnet der *ungapped BLAST* Ansatz ein näherungsweises MSP, das häufig als *high-scoring segment pair* (HSP) bezeichnet wird. Im Ersten von zwei Schritten wird die Datenbank nach Treffern mit einer festen Wortlänge w durchsucht. Bei Vergleichen von DNA Sequenzen werden alle Treffer berücksichtigt. Bei dem Vergleich von Proteinsequenzen nur die Wörter, die mindestens eine Punkteschwelle T erreicht haben. Legt man für das Beispiel aus Abbildung 5 eine feste Wortlänge von $w = 3$ voraus, so ergeben sich im ersten Schritt des Verfahrens vier Treffer:

	C	T	A	T	C	A	T	T	C	T	G
G	-4	-4	-4	-4	-4	-4	-4	-4	-4	-4	5
A	-4	-4	5	-4	-4	5	-4	-4	-4	-4	-4
T	-4	5	-4	5	-4	-4	5	5	-4	5	-4
C	5	-4	-4	-4	5	-4	-4	-4	5	-4	-4
C	5	-4	-4	-4	5	-4	-4	-4	5	-4	-4
A	-4	-4	5	-4	-4	5	-4	-4	-4	-4	-4
T	-4	5	-4	5	-4	-4	5	5	-4	5	-4
C	5	-4	-4	-4	5	-4	-4	-4	5	-4	-4
T	-4	5	-4	5	-4	-4	5	5	-4	5	-4
T	-4	5	-4	5	-4	-4	5	5	-4	5	-4

Abbildung 6: Exakte Treffer bei einer festen Wortlänge von $w = 3$. Inhalte übertragen aus Chao, *Sequence Comparison: Theory and Methods*.

Im zweiten Schritt werden die gefundenen Treffer diagonal in beide Richtungen erweitert um ein MSP zu finden, das den ursprünglichen Treffer beinhaltet. Die Erweiterung wird so lange in eine Richtung durchgeführt, bis die Bewertung der betrachteten Teilsequenz mehr als den vorher

definierten Wert X unter die bisher höchst bewerteste Folge fällt. Wenn der aus diesem Prozess resultierende Abschnitt eine Punkteschwelle S erreicht, wird er als Ergebnis zurückgegeben. Wendet man den zweiten Schritt auf die gefundenen Treffer aus Abbildung 6 an, ergibt sich das MSP wie in Abbildung 7 dargestellt.[24]

	C	**T**	**A**	**T**	**C**	**A**	**T**	**T**	**C**	**T**	**G**
G	-4	-4	-4	-4	-4	-4	-4	-4	-4	-4	5
A	-4	-4	5	-4	-4	5	-4	-4	-4	-4	-4
T	-4	5	-4	5	-4	-4	5	5	-4	5	-4
C	5	-4	-4	-4	5	-4	-4	-4	5	-4	-4
C	5	-4	-4	-4	5	-4	-4	-4	5	-4	-4
A	-4	-4	5	-4	-4	5	-4	-4	-4	-4	-4
T	-4	5	-4	5	-4	-4	5	5	-4	5	-4
C	5	-4	-4	-4	5	-4	-4	-4	5	-4	-4
T	-4	5	-4	5	-4	-4	5	5	-4	5	-4
T	-4	5	-4	5	-4	-4	5	5	-4	5	-4

Abbildung 7: Das durch diagonale Erweiterung gefundene MSP. Inhalte übertragen aus Chao, *Sequence Comparison: Theory and Methods*.

Beobachtungen haben gezeigt, dass die Erweiterungsphase in der *ungapped BLAST* Methode mit ungefähr 90 % den größten Rechenaufwand darstellt. Zusätzlich dazu hat man festgestellt, dass die *high-scoring segment pairs* (HSPs), die von Interesse sind, häufig viel länger sind als die zuvor definierte Wortlänge w und es sehr wahrscheinlich ist, dass es, auch auf kurzen diagonalen Abschnitten, mehrere Treffer geben wird. Diese Erkenntnisse wurden in die *gapped BLAST* Variante integriert. Der Algorithmus wurde so geändert, dass eine diagonale Erweiterung nur noch eingeleitet wird, wenn zwei, sich nicht überlappende Treffer, eine Lücke zueinander aufweisen, die kleiner oder gleich einer vorher definierten Länge D ist. Der Algorithmus kann in dieser veränderten Form angewendet werden, indem man sich für jede Diagonale die Position des letzten Treffers speichert. Abbildung 8 zeigt beispielhaft, bei welchem Treffer die diagonale Erweiterung ausgelöst wird und bei welchem nicht.

Wie der Name bereits vermuten lässt, besteht eine zusätzliche Weiterentwicklung der *gapped BLAST* Methode darin, dass sie dazu in der Lage ist, *gapped Alignments* zu generieren.[24] Nehmen wir zum Beispiel an, dass `C T T A G C` unserer Suchsequenz entspricht und `C T T C A G C` einer Sequenz aus der Datenbank. Werden die beiden Sequenzen mit einer *ungapped* Variante verglichen, würde keine Übereinstimmung erkannt werden. Ist das Verfahren jedoch in der Lage, *gapped Alignments* zu identifizieren, könnte die *deletion* aus der Suchsequenz erkannt werden und eine Übereinstimmung würde gefunden werden.

```
C T T - A G C
C T T C A G C
```

Eine Erweiterung die Lücken zulässt ist in der Regel deutlich langsamer, als eine Erweiterung, in der keine Lücken erlaubt sind. Im *gapped BLAST* Ansatz werden zwei Methoden angewendet, die dazu führen, dass die Berechnungen trotzdem in adäquater Zeit durchgeführt

[24] Chao, *Sequence Comparison: Theory and Methods*.

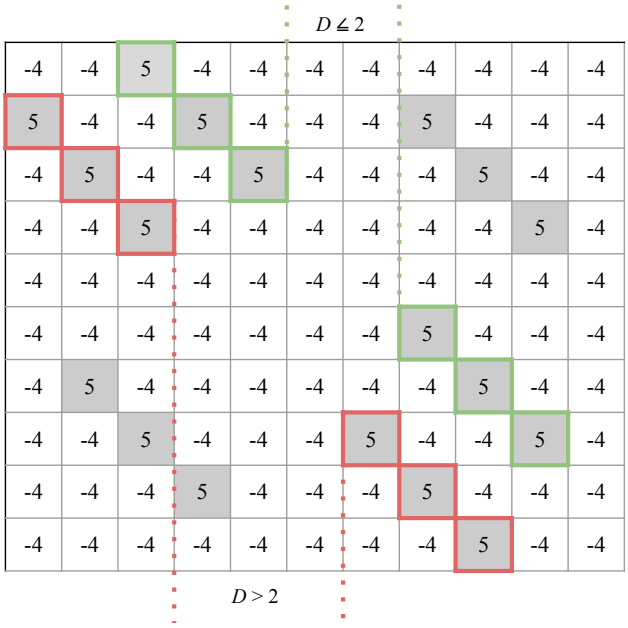

Abbildung 8: Darstellung, für welche Treffer eine Erweiterung durchgeführt wird. Chao, *Sequence Comparison: Theory and Methods*

werden können. Zum Einen werden Erweiterungen, in denen Lücken auftreten dürfen, nur dann durchgeführt, wenn die involvierten HSPs eine Punktebewertung haben, die eine zuvor angelegte Schwelle überschreiten, so dass eine solche Erweiterung nur maximal alle fünfzig Datenbanksequenzen vorkommt. Zum Anderen wird der Aufwand für den Aspekt der angewendeten dynamischen Programmierung verringert, indem nur die Bereiche für die Berechnung ausgewählt werden, die nur maximal einen bestimmten Wert schlechter sind als die bisher beste gefundene Übereinstimmung.[25]

Wie im Überblick bereits beschrieben, besteht die Aufgabe von PSI-BLAST darin, eine Datenbank mit Proteinsequenzen zu durchsuchen, um Sequenzen zu identifizieren, die mit einer bestimmten Protein Familie im Zusammenhang stehen. Die Anforderung an diese Aufgabe kann nicht von dem normalen BLAST Verfahren gelöst werden, da die in Beziehung stehenden Sequenzen in diesem Fall zu unterschiedlich sind. Die nach einer initialen BLAST Suche berechneten Ergebnisse mit der höchsten Bewertung werden genutzt, um eine *profile/position-specific scoring matrix* (PSSM) aufzustellen. In anschließenden BLAST Durchläufen wird die ursprüngliche Substitutionsmatrix durch die berechnete PSSM ersetzt. Diese iterative Abfolge kann dann viele Male wiederholt werden, um in jedem Schritt das Profil der Matrix zu verfeinern und die Sensitivität der Suche zu erhöhen.[26]

4.3 Weitere Suchverfahren

Neben BLAST gibt es noch weitere Suchverfahren, die für Anfragen an Datenbanken aus dem Bereich der Bioinformatik genutzt werden. In dieser Arbeit wird ein kurzer Überblick über den FastA Algorithmus, das *BLAST-like alignment tool* (BLAT), die PatternHunter Methode, das SWORD Verfahren und die DIAMOND Methode gegeben.

Der FastA Algorithmus gehört zu der Klasse der heuristischen Algorithmen und betrachtet man seine Vorgehensweise, kann man erkennen, warum er in diese Gruppe eingeteilt werden kann. Voraussetzung für den Algorithmus sind eine Datenbank und ein Suchstring. FastA sucht dann zwischen allen Sequenzen der Datenbank und dem Suchstring nach *guten* Übereinstimmungen

[25] Chao, *Sequence Comparison: Theory and Methods*.
[26] Zvelebil, *Understanding Bioinformatics*.

und gibt diese aus. Der Algorithmus nimmt an, dass *gute* Treffer solche sind, die einige exakt übereinstimmende Subsequenzen aufweisen. Durch seine Vorgehensweise gewinnt der Algorithmus zwar Geschwindigkeit, muss dafür aber Sensitivität einbüßen.[27]

Die Nutzung des *BLAST-like alignment tool* (BLAT) bietet sich vor allem an, wenn man weiß, dass es eine hohe Ähnlichkeit zwischen der Suchsequenz und den Sequenzen der Datenbank gibt.[28] Bei der PatternHunter Methode hat der Suchende die Möglichkeit, eine genaue Definition eines Treffers anzugeben, um die Suche auf diese Art und Weise zu beeinflussen.[28]

In verschiedenen Publikationen aus dem Jahre 2015[29] und 2016[30] wurden die beiden Suchverfahren DIAMOND und SWORD beschrieben, die für den Abgleich von Proteinsequenzen genutzt werden. Bei beiden Suchverfahren handelt es sich um Open Source Projekte und in den erwähnten Publikationen wird für beide Verfahren eine deutlich höhere Geschwindigkeit im Vergleich zu BLAST angegeben. Für DIAMOND wurde dabei besonders die Verarbeitung kurzer Sequenzen hervorgehoben.[29] SWORD unterstützt zwei verschiedene Modi für die Verarbeitung einer Suche. Zum Einen den sensitiven Modus, bei dem die Sensitivität des Verfahrens gewahrt werden soll und zum Anderen den schnellen Modus, der eine schnellere Rechenzeit bei verringerter Sensitivität aufweist.[30]

4.4 Das FastA Dateiformat

Neben dem FastA Algorithmus ist FastA auch Namensgeber für ein Dateiformat, das bei der Suche in Datenbanken im Bereich der Bioinformatik eine Rolle spielt. Das FastA Format ist eine textbasierte Darstellung einer DNA, beziehungsweise Proteinsequenz. In der ersten Zeile einer FastA Datei ist eine kurze Beschreibung der vorliegenden Sequenz hinterlegt. Der Rest der Datei besteht aus dem Inhalt der Sequenz, wobei es auch die Möglichkeit gibt, zwischen der ersten Zeile und dem Inhalt Kommentare einzufügen, indem die entsprechende Zeile mit einem *';'* eröffnet wird. Die erste Zeile beginnt mit einem *'>'* gefolgt von der Kennung der Sequenz. Nach der Kennung kann eine optionale Beschreibung hinzugefügt werden. Das folgende Beispiel zeigt eine frei erfundene Proteinsequenz mit der Kennung ALLP1234:

```
>ALLP1234 Eine optionale Beschreibung
;Ein optionaler Kommentar
EWIWHTSKHRRSVPNKPIAGXVPQSRPLSQGSNNWGATVI
```

Grundlegend besteht der Inhalt der Datei bei einer Proteinsequenz aus den Buchstaben der entsprechenden Aminosäuren und bei einer DNA Sequenz aus den Buchstaben der Basen. Dabei gibt es zusätzlich Zeichen, die zum Beispiel für eine Lücke, eine Wildcard oder eine Auswahl mehrerer Basen stehen können.[31]

4.5 BLAST praktisch anwenden

Das *National Center for Biotechnology Information* (NCBI) ermöglicht es, BLAST über den Browser zu nutzen und eigene Suchanfragen an eine Datenbank zu stellen. In diesem Kapitel soll eine einfache Anleitung gegeben werden, um sich in dem Tool zurechtzufinden. Die Oberfläche kann über den folgenden Link erreicht werden:

`https://blast.ncbi.nlm.nih.gov/Blast.cgi`

Zunächst muss man auswählen, ob man eine DNA oder eine Protein basierte Suche durchführen möchte. Dafür stehen die Schaltflächen mit der Beschriftung *'Nucleotide BLAST'* und *'Protein*

[27] Sung, *Algorithms in Bioinformatics: A Practical Introduction - Database Search*.
[28] Chao, *Sequence Comparison: Theory and Methods*.
[29] Buchfink, "Fast and sensitive protein alignment using DIAMOND".
[30] Robert Vaser, "SWORD — a highly efficient protein database search".
[31] Wikipedia, *FASTA-Format*.

BLAST' zur Verfügung. Im nächsten Fenster kann die gewünschte Sequenz zum Beispiel im FastA Dateiformat (vergleich Kapitel 4.4) in der Box im oberen Bereich eingegeben werden. Im zweiten Abschnitt der Einstellungen ist bereits eine Datenbank ausgewählt, diese kann bei Bedarf gewechselt werden. Die weiteren optionalen Optionen dienen dazu, die Suche weiter zu spezifizieren. Ein Eintrag in der Zeile *'Organism'* kann zum Beispiel dafür genutzt werden, die Suchsequenz nur mit Sequenzen abzugleichen, die zu dem ausgewählten Organismus gehören. Die letztendliche Suche wird über den Knopf, mit dem Titel *'BLAST'*, im unteren linken Bereich der Maske ausgeführt.

Auf der Detailseite werden im oberen Bereich nochmals die Detailinformationen zu der gestarteten Suche angezeigt. Unter dem Reiter *'Descriptions'* sind alle Sequenzen aufgelistet, die auf Grund der Suchanfrage als signifikante Treffer erkannt wurden. Die Spalte *'E value'* der Tabelle zeigt dabei die Fehlerrate des Alignments an. Je kleiner der Wert ist, desto signifikanter ist die Übereinstimmung zwischen dem Treffer und der Suchanfrage. Der Reiter *'Alignments'* bietet eine detaillierte Übersicht der gefundenen Alignments an, aus der zum Beispiel entnommen werden kann, wie viele *Matches* (*'Identities'*) die beiden Sequenzen aufweisen.

5 Datenmanagement - Gegenwart und Zukunft

Obwohl eine Vielzahl an Datenbanken und Ressourcen für den (proteinbezogenen) Bereich der Bioinformatik entwickelt wurden, gibt es für die Zukunft Herausforderungen, aber auch Möglichkeiten, das Datenmanagement, also den Umgang mit Daten aus dem Bereich der Bioinformatik effizient und zeitgemäß zu gestalten.

Indem der Fokus auf die Bereitstellung von Technologien der nächsten Generation gelegt wird, kann die Datenintegration, das Aufstellen von Hypothesen mit Bezug zu bekannten Daten und die Entdeckung biologischen Wissens gefördert werden. Wie in vielen anderen Bereichen der Informationsverarbeitung spielt die *Big Data* Thematik auch in der Bioinformatik eine zunehmende Rolle.[32] *Big Data* beschäftigt sich mit der Analyse, dem Extrahieren von Informationen oder einer beliebigen anderen Verarbeitung von Datenmengen, die zu groß sind, um sie mit traditionellen Verarbeitungsmethoden zu bearbeiten. Verschiedene Kriterien können den *Big Data* Bereich noch genauer charakterisieren, dazu zählen zum Beispiel: 1) *Volume*: Die Menge der generierten und gespeicherten Daten; 2) *Variety*: Die Art und der Ursprung der Daten; 3) *Velocity*: Die Geschwindigkeit, mit der Daten generiert und verarbeitet werden; 4) *Veracity*: Die Datenqualität.[33] Vor dem Hintergrund der in Kapitel 1 gegebenen Informationen, erscheint es plausibel, dass sich die Bioinformatik in Richtung des *Big Data* Themenbereichs entwickelt. Dadurch können zum Einen neue Optionen geschaffen werden um die Molekularbiologie zu erforschen, aber zum Anderen die Wissenschaftler auch vor die Aufgabe gestellt werden, Wege zu finden, Daten adäquat zu verwalten und analysieren zu können. Daher besteht die erste Herausforderungen der Bioinformatik darin, die große Menge an Daten effizient zu speichern und zu verwalten.[32]

Zusätzlich zur besseren Hardware, müssen dann auch, bereits vorhandene, parallele Speichersysteme zum Einsatz kommen. Im Zusammenhang mit *Big Data* können Probleme bei der Verwendung der traditionellen relationalen Datenbanken entstehen, da diese nicht gut horizontal skalierbar sind, starke Konsistenz benötigen und dazu neigen sehr Komplex zu werden, wenn sie große Mengen an heterogenen Daten verarbeiten müssen. Eine mögliche Alternative stellt die Verwendung von nicht relationalen *Not only Structured Query Language (SQL)* (NoSQL) Datenbanken dar, da diese ihren Fokus mehr auf die Skalierbarkeit und Flexibilität legen.

Neben dem Speichern und Verwalten von Daten spielt die Analyse dieser Daten die nächste große Rolle. Das letztendliche Ziel der Analyse besteht darin, Wissen aus biomedizinischen Daten zu ziehen. Auch dafür werden skalierbare technische Einrichtungen und *Frameworks* für

[32] Chuming Chen, "Protein Bioinformatics Databases and Resources".
[33] Wikipedia, *Big data*.

die effiziente Datenanalyse benötigt. *Cloud* basierte Ansätze verändern hier bereits die Art und Weise wie Analysen durchgeführt werden und fungieren dabei als ökonomische Lösung. Weitere Verfahren, wie z.B. *machine learning* und *data mining* Algorithmen sind notwendig, um Daten erfolgreich in Wissen zu transformieren.[34]

Die wahrscheinlich größte Herausforderung bei der Arbeit mit Big Data liegt darin, mit der Heterogenität und Komplexität der Daten umzugehen und eine bessere Möglichkeit zu finden diese Daten zu integrieren. Zu einem gewissen Anteil wird dieser Punkt durch die Flexibilität von NoSQL Datenbanken abgedeckt. Desweiteren stellen die *Ontology* und *Semantic Web* Technologien vielversprechende Kandidaten bei der Handhabung dieser Problematik dar. So erlaubt der standardisierte Einsatz von *Ontologies* der Gemeinschaft an Forschern es, biologische und biomedizinische Daten zu annotieren und zu integrieren. Und der Einsatz von *Semantic Web* Ansätzen erlaubt es Daten im Internet zu veröffentlichen und untereinander zu verbinden, um so ein Netzwerk aus verbundenen Daten für den gesamten Bereich der Lebenswissenschaften zu bilden, das dann von Applikationen genutzt werden kann, um die relevanten Daten zu extrahieren, aufzubereiten und dem Benutzer zu präsentieren.[34]

Eine weitere mögliche Anforderung an die zukünftigen Datenbanken im Bereich der Bioinformatik ist die Bereitstellung responsiver Webseiten und Applikationen, da die immer größere Relevanz mobiler Endgeräte auch in diesem Bereich an Priorität gewinnt. Neben dem optisch ansprechenden Design spielt in diesem Kontext auch die Performance zum schnellen Erhalt der Informationen eine bedeutende Rolle.[34]

Durch die hier beschriebenen Anforderungen und Herausforderungen im Bereich der Bioinformatik und die Neuentwicklung oder Weiterentwicklung innovativer oder bereits bestehender Suchverfahren lassen sich zusätzliche Themen ableiten, die im Fokus weiterer Seminararbeiten stehen könnten.

[34] Chuming Chen, "Protein Bioinformatics Databases and Resources".

Abbildungsverzeichnis

1 Anzahl der in der NCBI RefSeq gespeicherten Proteine. Daten der verschiedenen Jahre entnommen von NCBI, *RefSeq: NCBI Reference Sequence Database*, NCBI, *RefSeq Announcements for 2016*, NCBI, *RefSeq Announcements for 2015*, NCBI, *RefSeq Announcements for 2014* und NCBI, *RefSeq Announcements for 2013* . . . 4
2 Schematischer Aufbau einer Nukleotidkette. 5
3 Schematischer Aufbau einer DNA Doppelhelix. 6
4 Beispielhafte *Scoring Matrix* für einen Vergleich von DNA Sequenzen. Inhalt entnommen aus Chao, *Sequence Comparison: Theory and Methods*. 9
5 Zwei Sequenzen in einer Matrix dargestellt und mit der vorher definierten Bewertung versehen. Inhalte übertragen aus Chao, *Sequence Comparison: Theory and Methods*. 12
6 Exakte Treffer bei einer festen Wortlänge von $w = 3$. Inhalte übertragen aus Chao, *Sequence Comparison: Theory and Methods*. 12
7 Das durch diagonale Erweiterung gefundene MSP. Inhalte übertragen aus Chao, *Sequence Comparison: Theory and Methods*. 13
8 Darstellung, für welche Treffer eine Erweiterung durchgeführt wird. Chao, *Sequence Comparison: Theory and Methods* 14

Tabellenverzeichnis

1 Darstellung der Ereignisse *Match* und *Mismatch* bei dem *Alignment* zwischen Sequenz X und Y. 7
2 Darstellung der Ereignisse *Insertion* und *Deletion* bei dem *Alignment* zwischen Sequenz X und Y. 8
3 Anzahl möglicher *Alignments* in Abhängigkeit von der Sequenzlänge. In der ersten Zeile und der ersten Spalte werden die Wortlängen beider Sequenzen abgebildet. Inhalte entnommen aus Chao, *Sequence Comparison: Theory and Methods*. . . . 8
4 Eine Auswahl von BLAST Ausprägungen und deren Bedeutung. Inhalte entnommen aus Madden, “The BLAST Sequence Analysis Tool”. 11

Literatur

Buchfink Xie, Huson. "Fast and sensitive protein alignment using DIAMOND". In: *Nature Methods* 12.1 (2015), S. 59–60. URL: https://www.ncbi.nlm.nih.gov/pubmed/25402007# (besucht am 03. 12. 2019).

Chao, Zang. *Sequence Comparison: Theory and Methods*. Springer, 2008. ISBN: 978-1848003194.

Chuming Chen Hongzhan Huang, Cathy H. Wu. "Protein Bioinformatics Databases and Resources". In: *Methods in Molecular Biology* 1558 (2017), S. 3–39. URL: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC5506686/ (besucht am 06. 11. 2019).

Madden, Thomas. "The BLAST Sequence Analysis Tool". In: *The NCBI Handbook [Internet]. 2nd edition* (2013). URL: https://www.ncbi.nlm.nih.gov/books/NBK153387/ (besucht am 06. 11. 2019).

NCBI. *RefSeq Announcements for 2013*. URL: https://www.ncbi.nlm.nih.gov/refseq/announcements/2013/ (besucht am 13. 11. 2019).

— *RefSeq Announcements for 2014*. URL: https://www.ncbi.nlm.nih.gov/refseq/announcements/2014/ (besucht am 13. 11. 2019).

— *RefSeq Announcements for 2015*. URL: https://www.ncbi.nlm.nih.gov/refseq/announcements/2015/ (besucht am 13. 11. 2019).

— *RefSeq Announcements for 2016*. URL: https://www.ncbi.nlm.nih.gov/refseq/announcements/2016/ (besucht am 13. 11. 2019).

— *RefSeq: NCBI Reference Sequence Database*. URL: https://www.ncbi.nlm.nih.gov/refseq/ (besucht am 13. 11. 2019).

Robert Vaser Dario Pavlović, Mile Šikić. "SWORD — a highly efficient protein database search". In: *Bioinformatics* 32.17 (2016), S. i680–i684. URL: https://doi.org/10.1093/bioinformatics/btw445 (besucht am 06. 11. 2019).

Sung, Wing-Kin. *Algorithms in Bioinformatics: A Practical Introduction - Database Search*. URL: https://www.comp.nus.edu.sg/~ksung/algo_in_bioinfo/slides/Ch5_database.pdf (besucht am 06. 11. 2019).

Wikipedia. *Big data*. URL: https://en.wikipedia.org/wiki/Big_data (besucht am 15. 11. 2019).

— *Desoxyribonukleinsäure*. URL: https://de.wikipedia.org/wiki/Desoxyribonukleins%C3%A4ure (besucht am 13. 11. 2019).

— *FASTA-Format*. URL: https://de.wikipedia.org/wiki/FASTA-Format (besucht am 16. 02. 2020).

— *Gen*. URL: https://de.wikipedia.org/wiki/Gen (besucht am 13. 11. 2019).

Zvelebil, Baum. *Understanding Bioinformatics*. Taylor & Francis, 2007. ISBN: 978-0815340249.