Algorithmen und Werkzeuge zur Analyse von Hochdurchsatz-DNA-Sequenzierungsdaten*

Marcel Martin

Science for Life Laboratory (SciLifeLab), Stockholm
marcel.martin@scilifelab.se

Abstract: Hochdurchsatz-DNA-Sequenzierung hat in den letzten Jahren die biologische und medizinische Forschung revolutioniert. Gestiegene Datenmengen und völlig neue Arten der Sequenzierung haben auch zu neuen Herausforderungen in der Informatik geführt. Wir beschreiben hier Algorithmen und Methoden aus vier verschiedenen Gebieten der Hochdurchsatzsequenzierung. Es geht hierbei um das Entfernen von Verunreinigungen durch Adaptersequenzen, eine effiziente Methode um Methylierungsmuster in DNA zu entdecken, um eine neuartige Methode zum interaktiven Aufspüren von krankheitsverursachenden Mutationen und schließlich um eine Reduktion von Sequenzierfehlern durch einen neuen Alignment-Algorithmus.

1 Einführung

Eine auch heute noch gebräuchliche Methode zur DNA-Sequenzierung wurde bereits 1977 vorgestellt [SNC77] und über die Jahre so verbessert, dass 2001 die Sequenz des menschlichen Genoms veröffentlicht werden konnte [LLB+01]. Die *Sanger-Seqenzierung* liefert bis zu ca. 1000 Basen pro DNA-Fragment, allerdings zu vergleichsweise hohen Kosten: Das Humangenom mit seinen drei Milliarden Basen konnte nur durch eine internationale Anstrengung, die Milliarden von Dollarn kostete, sequenziert werden. Sequenzierungsgeräte der zweiten Generation, die *Hochdurchsatzsequenzierungsmethoden* nutzen, kamen wenige Jahre später auf den Markt [MEA+05]. In ihnen laufen die Reaktionen hochparallel und miniaturisiert ab, sodass die Kosten pro Base drastisch reduziert werden. Sie veränderten und verändern die biomedizinische Forschung daher grundlegend. Die rasante Entwicklung hält immer noch an und das 1000-Dollar-Genom ist in Reichweite.

Damit einhergehend kamen aber auch völlig neue informatische Probleme. Zum einen macht das schiere Datenvolumen immer effizientere Algorithmen erforderlich: Ein typischer Lauf des im Jahr 2013 aktuellsten Sequenziergeräts der Firma Illumina produziert 600 GB pro Woche. Sequenzierzentren, in denen dutzende von Geräten parallel betrieben werden, müssen proportional mehr bewältigen. Zum anderen erfordert die spezielle Art der Daten die Entwicklung neuer Verfahren, denn die neuen Geräte liefern zwar Milliar-

*Englischer Titel der Dissertation: "Algorithms and Tools for the Analysis of High-Throughput DNA Sequencing Data"

den von Sequenzen gleichzeitig, aber jeder dieser sogenannten *Reads* ist sehr kurz, typischerweise nur 100 Basen, und mit einer im Vergleich zur Sanger-Sequenzierung hohen Fehlerrate von – je nach Technologie – bis zu 1% behaftet.

Die hier zusammengefasste Dissertation [Mar13] liefert einen entscheidenden Beitrag dabei, vier der durch die neuen Technologien auftretenden Probleme zu lösen. Zuerst widmen wir uns dem Problem, eine durch den Sequenzierungsprozess bedingte häufig vorkommende Verunreinigung der Reads durch sogenannte *Adaptoren* zu entfernen. Erst hierdurch werden die Daten in nachfolgenden Schritten nutzbar. Als zweites geht es um das *Read Mapping*, d. h. darum, die Reads durch fehlertolerante Textsuchalgorithmen in einem Referenzgenom wiederzufinden, wobei wir uns jedoch auf einen speziellen und biologisch wichtigen Anwendungsfall konzentrieren, in dem die DNA vor der Sequenzierung einer chemischen Veränderung unterzogen wurde, um zusätzliche Informationen über ihren Aufbau zu erhalten. Die chemischen Veränderungen führen zu bestimmten Basenersetzungen, die das Read Mapping verkomplizieren, aber mit einer neuen von uns vorgeschlagenen Datenstruktur berücksichtigt werden können. Im dritten Abschnitt zeigen wir, wie ein System aussehen kann, mit dem die Forschung in der Humangenetik stark vereinfacht werden kann, wenn es unter anderem darum geht, die Ursachen für seltene Krankheiten zu finden oder auch die Mutationen zu identifizieren, die in einem menschlichen Tumor auftreten. Zum Schluss geht es wieder zurück zu den Grundlagen des Sequenzierens und wir beschreiben ein völlig neues Verfahren, um Fehler in Sequenzierungsdaten zu beheben, die von Geräten stammen, welche nach dem Prinzip der *Flowgramm-Sequenzierung* arbeiten.

Molekularbiologische Grundlagen Das Erbgut in jeder biologischen Zelle ist in Form von DNA gespeichert, die als Doppelstrang vorliegt. Die beiden Stränge sind jeweils Kettenmoleküle, welche aus den vier Bausteinen Adenin, Cytosin, Guanin und Thymin (A, C, G, T) aufgebaut sind, wobei sich immer A mit T und C mit G des jeweils anderen Strangs verbinden (Basenpaare). Die Informationen liegen somit redundant vor, da sich die Basenabfolge des einen Strang aus dem anderen ableiten lässt. Einzelne Abschnitte der DNA (Gene) werden von der Zelle ausgelesen und in Proteine übersetzt, die alle möglichen Funktionen im Organismus übernehmen.

2 Entfernen von Adaptersequenzen

Zur Sequenzierung wird die zunächst einsträngig vorliegende DNA im Laufe des Sequenzierungsprozess schrittweise zu einem Doppelstrang ergänzt (siehe Abb. 1). Die dabei jeweils eingebauten Basen werden detektiert und resultieren im Read, der somit eine Zeichenkette über dem Alphabet $\Sigma = \{\mathrm{A}, \mathrm{C}, \mathrm{G}, \mathrm{T}\}$ ist. Dieser Prozess ist mit Fehlern behaftet und typische Fehlerraten von 1% oder mehr müssen in den Algorithmen berücksichtigt werden. Beim Sequenzieren von derart kurzen Molekülen, dass sie die fest vorgegebene Readlänge nicht ausnutzen, erscheinen die Sequenzen der für den Sequenzierprozess notwendigen Adaptersequenzen als Teil des Reads und müssen entfernt werden. Solche

Abbildung 1: Vor und nach dem Sequenzierungsprozess. Die schwarzen Buchstaben gehören zum Adapter, die konturierten gehören zum uns interessierenden DNA-Fragment. Oben: Das zum Sequenzieren vorbereitete DNA-Molekül wurde mit Hilfe des Adapters AGGTT innerhalb des Geräts auf einer Oberfläche befestigt. Unten: Nach Abschluss des Sequenzierprozesses ist der DNA-Doppelstrang vollständig aufgebaut. Der Read lautet AAT...CAAGGA und schließt somit die Adaptersequenz mit ein.

kurzen Moleküle können z. B. microRNA-Moleküle sein, welche in der Zelle genutzt werden um Genexpression zu regulieren.

Das sich ergebende und von uns definierte *Adapter Trimming Problem* ist eine Variante des bereits von Gusfield [Gus97] beschriebenen Problems, fehlertolerant Überlappungen zweier Zeichenketten zu finden und kann prinzipiell mit semiglobalen Alignment gelöst werden. Herausforderungen bestehen in den nötigen Anpassungen: Zum einen gibt es nicht nur den beschriebenen Typ von Adaptoren, sondern mindestens drei weitere. Zum anderen gilt es eine Höchstfehlerrate als Parameter zuzulassen.

2.1 Höchstfehlerrate

Ein semiglobales Alignment zwischen dem Read `FRAGMENTADOPT` und der Adaptersequenz `ADAPTER` kann so veranschaulicht werden:

```
      ADAPTER
      ||X||
FRAGMENTADOPT
```

Da eine Substitution vorliegt (Insertionen und Deletionen sind ebenso möglich) und fünf Basen des Adapters betroffen sind, ist die Fehlerrate $r = \frac{1}{5}$. Problematisch ist nun, dass eine Minimierung der Fehlerrate immer zu Alignments mit null Fehlern führen würde, in denen sich die Sequenzen aber nicht überlappen. Bisher wurde in semiglobalen Alignments daher die Summe der Punkte, die für Substitutionen, Übereinstimmungen, Insertionen und Deletionen vergeben wurde, maximiert. Problematisch hierbei ist, dass diese

Punkte ohne tiefere Bedeutung sind und die Ergebnisse daher nicht intuitiv verständlich sind. Wir verändern daher das Optimierungskriterium folgendermaßen: Betrachte zu allen erlaubten Überlappungen alle optimale Alignments und nimm dann unter denjenigen, deren Fehlerrate unter der vorgegebenen Höchstfehlerrate liegt, dasjenige mit der höchsten Anzahl Übereinstimmungen. Der zugehörige Algorithmus kann mit einigen Verbesserungen Adaptoren der Länge m in Reads der Länge n in Zeit $O(n\varepsilon m)$ finden, wobei ε die Höchstfehlerrate ist. Dieser Ansatz, so können wir zeigen, funktioniert in der Praxis ausgesprochen gut und wurde in der Form des sehr erfolgreichen Programms *cutadapt*[1] der Bioinformatik-Gemeinschaft zur Verfügung gestellt.

Weitere in der Dissertation betrachtete Aspekte sind u. a. theoretische Überlegungen zu Sensitivität und Spezifizität und eine Erweiterung des Algorithmus auf sogenannte *Colorspace*-Reads, in denen die Zeichen eines Reads als "Farben" $0, \ldots, 3$ kodiert sind, die aber jeweils von zwei aufeinanderfolgenden Basen im DNA-Molekül abhängen.

3 Bisulfit-Read-Mapping

Manche Basen der DNA sind chemisch so modifiziert, dass an ihnen eine CH_3-Gruppe heftet; sie sind *methyliert*. In Wirbeltier-DNA, also auch menschlicher, können nur Cytosine vor Guaninen (auch CpG genannt) methyliert sein. Die Methylierung hat für die Zelle eine wichtige Funktion, denn es gibt Bereiche in der Nähe von Genen, in denen CpGs besonders häufig auftreten, genannt CpG-Inseln, die als Schalter fungieren: Ist eine Insel methyliert, so wird das zugehörige Gen inaktiviert. Auf diese Weise können die Zellen eines mehrzelligen Organismus, die ja alle die gleiche DNA enthalten, dennoch ganz unterschiedlich ausgeprägt sein. Zum Beispiel können so in Leberzellen andere Gene als in Nervenzellen aktiv sein.

Um Methylierungsmuster herauszufinden, wird die DNA vor dem Sequenzieren chemisch mit (Natrium-)Bisulfit behandelt. Dies führt zu einer vom Methylierungszustand abhängigen Veränderung im Read: Unmethylierte C werden zu T umgewandelt und methylierte C bleiben unverändert. Durch den Vergleich zur Referenz kann dann festgestellt werden, welche Basen methyliert waren.

Wenn ein Referenzgenom bekannt ist, werden Reads (evtl. nach Entfernen der Adaptoren) im allerersten Verarbeitungsschritt fehlertolerant ihrer Position in der Referenz zugeordnet. Dies wird auch *Read Mapping* genannt und es existieren verschiedene Algorithmen, die das Problem lösen [LD09]. Als wir DNA-Methylierung in menschlichen X-Chromosomen untersuchten [ZMB+09], gab es keine zuverlässigen Algorithmen, um bisulfitbehandelte Reads aus Hochdurchsatzexperimenten zu mappen. Wir haben daher eine neue Datenstruktur entwickelt, die beim Mappen alle derartigen Bisulfit-Ersatzungsregeln berücksichtigen kann. Zum Mappen nutzt unser Algorithmus den *Seed-and-extend*-Ansatz: Zuerst werden kurze, exakte Übereinstimmungen zwischen Referenz und Read gefunden (*seeds*), die dann in einem zweiten Schritt fehlertolerant auf den vollen Read erweitert werden (*extend*). Beide Phasen werden von uns an die Bisulfit-Regeln angepasst.

[1] https://github.com/marcelm/cutadapt/

Um kurze Übereinstimmungen zu finden, erweitern wir den *q-Gramm-Index*. Ein solcher Index zu einem Text s ist eine Funktion, die alle Zeichenketten der Länge q auf die Liste ihrer Vorkommen in s abbildet. Implementiert wird der Index als 4^q Arrays, was speicheraufwändig ist, aber $O(1)$ Zugriff auf jedes Vorkommen garantiert. Unsere *Bisulfit-q-Gramm-Index* ist eine Erweiterung, die die Frage beantwortet: Gegeben einen String der Länge q, wo kommt dieser in der Referenz s vor unter der Annahme, dass Bisulfitveränderungen durchgeführt worden? Wird beispielsweise `TTG` (mit $q = 3$) gesucht, so werden auch Positionen, die `TCG` lauten, zurückgeliefert. Aber auch: Wird nach `CAG` gesucht, so werden nur Positionen zurückgeliefert, an denen `TAG` steht, denn ein C, welches nicht vor G steht, ist nie methyliert und wird daher immer in T umgewandelt. Erschwerend kommt hinzu, dass die DNA nach der Bisulfitbehandlung kopiert wird und dadurch aufgrund der Doppelsträngigkeit der DNA letztendlich auch Ersetzungen der Form G→A auftreten können, wenn das G nach einem C steht. Auch diese Form von Ersetzungen sowohl den Fall, dass die Bisulfitbehandlung unvollständig war und somit teilweise gar keine Ersetzungen vorkommen, wird von dem neuen Index behandelt, indem die Bisulfitbehandlung bei der Indexerstellung simuliert wird. Auch für die Erweiterung der kurzen Treffer müssen Anpassungen vorgenommen werden und wir zeigen, dass dies besonders effizient mit einem bitparallelen Algorithmus möglich ist, wenn die DNA als Binärstring dargestellt ist, in dem je zwei Bits eine Base kodieren.

In der Dissertation wird weiterhin untersucht, wie viele unterschiedliche Strings der Länge n es gibt, die aus Bisulfitbehandlung herrühren. Es lässt sich zeigen, dass die Anzahl dieser Bisulfitstrings näherungsweise $1.19 \cdot 3.3^n$ ist. Eine weiterer Aspekt ist die Frage, ob sich der Index komprimieren lässt, da insbesondere auch die Bisulfitvariante den Speicher erhöht. Wir können zeigen, dass eine Platzreduktion von 25% ohne Verlust von Laufzeiteffizienz möglich ist.

4 Exomsequenzierung

Das *Exom* ist der Teil des Genoms, der aus allen *exprimierten Sequenzen* besteht, also vereinfacht gesagt aus allen Genen. Das Exom macht mit ca. 50 Mio. Basen nur ca. 2% des menschlichen Genoms aus, ist aber der Teil, in dem sich Mutationen am stärksten auswirken. Ein großer Teil der Erbkrankheiten und Tumorarten entstehen durch Veränderungen im Exom. Exomsequenzierung besteht darin, vor dem Sequenzieren das Exom aus der vorliegenden Gewebeprobe zu extrahieren. Da es durch den so reduzierten Sequenzieraufwand möglich ist, zu gleichen Kosten mehr Exome und somit Patienten zu untersuchen, hat dieses Vorgehen in letzter Zeit an besonderer Bedeutung gewonnen [NTR+09].

Wir stellen hier eine neue Methode zur effizienten Analyse von Exomen vor, die in der Software *Exomate*[2] implementiert wurde. Das Ziel solch einer Analyse ist es üblicherweise, die kranhkeitsverursachenden Mutationen zu finden. Im einfachen Fall ist nur ein Gen betroffen, bei Tumoren sind es mehrere, deren Zusammenspiel zur Erkrankung führt. Die Schwierigkeit besteht darin, dass es immer tausende bis zehntausende Varianten zwischen

[2] https://bitbucket.org/marcelm/exomate

der untersuchten Probe und dem Referenzgenom gibt, die aber nicht schädlich sind, sondern durch die natürlichen, normalen Unterschiede zwischen Individuuen begründet sind (Haarfarbe etc.). Es ist daher elementar, geschickte Filterungsstrategien einzusetzen. Diese Strategien müssen jedoch für jede Studie angepasst werden, wobei es von Vorteil ist, wenn die Mediziner oder Genetiker, die die Studie durchführen, dies direkt und interaktiv im Webbrowser tun können. Exomate besteht daher aus drei Teilen: Einem Pipeline-Modul, einer Datenbank und einer Weboberfläche.

Pipeline. Das Pipeline-Modul von Exomate übernimmt alle nötigen Berechnungen, um von den Rohdaten, also den Reads, zu zuverlässigen Mengen von Varianten zu gelangen. Hier sind mehrere Zwischenschritte erforderlich, u. a. Read Mapping, Entfernen von doppelten Sequenzen (PCR-Duplikate) und Rekalibrierung der Qualitätswerte, die mit jeder Base verknüpft sind. Die Pipeline geht hierbei nach den *Best practices* des GATK-Framework [DBP+11] vor und ruft Standardprogramme auf. Die Schritte sind jedoch vollständig automatisiert, sodass alle gefundenen Varianten einer Probe am Ende automatisch in die Datenbank eingetragen werden. Hier findet nur sehr schwache Filterung statt, um keine Sensitivität zu verlieren. Jede Variante ist mit mehreren Qualitätswerten versehen, die dabei helfen zu beurteilen, ob die Variante reell ist oder evtl. von systematischen Sequenzierfehlern verursacht wurde.

Datenbank. Die Postgres-Datenbank enthält zum einen Metadaten über Familien, Patienten und sequenzierte Proben. Erfasst werden z. B. Datum des Sequenzierlaufs, Name des Sequenziergeräts, Statistiken zur Qualität, welche Krankheit mit einer Probe verknüpft ist und das Geschlecht des Patienten. Patienten werden anhand von anonymen Bezeichnern identifiziert, wobei die Zuordnung zwischen Patient und Bezeichner nicht im System gespeichert ist.

Den weitaus größeren Teil der Datenbank machen aber die Varianten selbst aus, die aus zwei Quellen kommen: Sequenzierte Proben und öffenliche Datenbanken, wie z. B. dbSNP. Die sequenzierten Proben stammen auch von gesunden Kontrollpatienten und dienen zusammen mit den dbSNP-Varianten der Filterung. Alle Varianten haben eine eindeutige Identifikationsnummer y und in der Liste aller Varianten wird nur gespeichert, dass Patient x Variante y hat. Dies ist eine Form der Normalisierung, die hier eine besonders schnelle Filterung ermöglicht, da der Datenbank-Server im Grunde nur Mengenoperationen auf Mengen von Identifikationsnummern durchführen muss. Die Varianten selbst sind in einer separaten Tabelle gespeichert und beschreiben jeweils Chromosom, Koordinate auf dem Chromosom und die genaue Veränderung, also z. B. „Chr. 6, 157 406 006, C→T“.

Weboberfläche. Die Weboberfläche dient im Gegensatz zu anderen Systemen nicht nur der Anzeige von Tabellen, sondern enthält einen entscheidenden Teil der Intelligenz des Systems, die wie erwähnt hauptsächlich im geschickten Filtern der Varianten besteht. Über die Python-Bibliothek SQLalchemy[3] ist es möglich, auf vergleichsweise einfache und lesbare Art aufwändige SQL-Abfragen automatisch erstellen zu lassen, die darüberhinaus

[3] http://www.sqlalchemy.org/

	Gene	Sample	Transcripts	Nucleotide	Codon	Amino acid	Type	Region	Quality	Location
1	ARID1A	M024	001, 201, 002, more	C → T	CGA→ TGA	R → *	Nonsense	Coding	75.19	1:27106354
2	ARID1B	M026	201, 203, 009, more	C → T	CGA→ TGA	R → *	Nonsense	Coding	37.98	6:157406006
3	SMARCB1	M021	005, 001, 002, 003	G → A	CGG→ CAG	R → Q	Missense	Coding	224.79	22:24176330

Abbildung 2: Diese von Exomate gelieferte Ergebnistabelle zeigt alle Mutationen in von uns untersuchten Coffin-Siris-Patienten (eine seltene Erbkrankheit), die nach dem Filtern von nicht relevanten Mutationen übrig bleiben. Für jede Mutation wird u. a. der Genname und Chromosomenposition angegeben, welche Nukleotidveränderung im Detail aufgetreten ist und welche Aminosäureveranderung herbeigeführt wurde.

noch dynamisch sind, das bedeutet zum Beispiel, dass in einer Version der Abfrage ein JOIN durchgeführt wird und in der anderen nicht. Auf diese Art werden Filter, die von uns mit einem LEFT OUTER JOIN realisiert wurden, an- und abgeschaltet. Neben der wichtigsten Einstellung, welche Proben untersucht werden sollen, beeinflussen weitere Parameter die Suche: Schwellwerte für verschiedenste Qualitätswerte, welche dbSNP-Varianten gefiltert werden sollen, ob auch synonyme Varianten angezeigt werden sollen (diese verändern normalerweise das entstehende Protein nicht), welche Gene von Interesse sind, welche Proben ausgefiltert werden sollen usw. Eine beispielhaftes Ergebnis ist in Abb. 2 zu sehen.

5 Flowgramm-Alignment

Die Sequenziergeräte der Firmen 454 Life Sciences und Ion Torrent liefern ihre Rohdaten als *Flowgramme*, aus denen dann die Reads in der gewohnten Darstellung als Zeichenkette über dem DNA-Alphabet gewonnen werden. Um die bei der Konversion enstehenden Fehler zu verringern, ist es sinnvoll, direkt mit den Flowgrammen zu arbeiten. 454 und Ion Torrent detektieren nicht einzelne Basen, sondern messen die Länge von *Homopolymeren*, also Läufe identischer Basen. Diese Längenmessung ist fehlerbehaftet. Ein einzelner *Flow* ist ein Paar $(c, f) \in \Sigma \times \mathbb{R}_0^+$, wobei wir dies als c^f schreiben und f als *Intensität* bezeichnen. Ein Flowgramm ist eine Sequenz von Flows. Wird beispielsweise das DNA-Molekül `TTCGG` sequenziert, so ist ein mögliches gemessenes Flowgramm $\mathtt{T}^{2.3}\mathtt{A}^{0.1}\mathtt{C}^{0.9}\mathtt{G}^{1.9}$.

Üblicherweise werden die reellwertigen Intensitäten gerundet, um eine normale Zeichenkette zu erhalten. Dies führt zu Informationsverlust: Zum Beispiel werden sowohl $\mathtt{C}^{3.50}$ als auch $\mathtt{C}^{4.49}$ zu `CCCC`, obwohl sie beim Vergleich mit einer Referenz, an der `CCC` steht, anders bewertet werden sollten. Ein ähnliches Problem gibt es, wenn $\mathtt{C}^{4.49}$ und $\mathtt{C}^{4.50}$ einmal zu `CCCC` und einmal zu `CCCCC` werden. Hier geht die Information verloren, dass beide Intensitäten sehr ähnlich sind. Statt also Flowgramme erst zu Strings zu konvertieren, um dann Standard-Alignment-Algorithmen zu nutzen, um sie gegen eine Referenz zu alignieren, ist es vorteilhaft, die Flowgramme ohne Informationsverlust direkt an die Referenz zu alignieren. Hierfür wurden bereits Verfahren vorgeschlagen [VJZL08], die aber entweder die Referenz in ein künstliches Flowgramm oder den Read in ein Flowgramm/String-

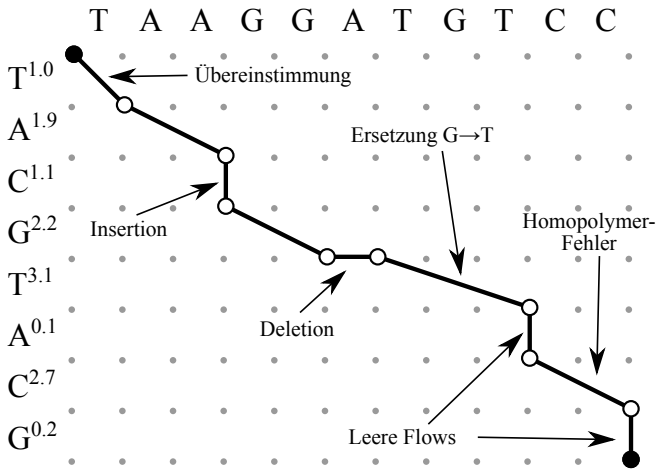

Abbildung 3: In diesem Flowgramm-Alignment-Graph ist ein Pfad eingezeichnet, der für ein mögliches Flowgramm-Alignment steht. Der Übersichtlichkeit halber sind nur die tatsächlich gewählten Kanten eingezeichnet, nicht alle möglichen.

Hybrid konvertieren, wodurch sie nicht in der Lage sind, alle Arten von Veränderungen korrekt zu modellieren. Unsere Methode ist hingegen ein *direktes* Alignment zwischen Flowgramm und Referenz und reduziert so die durch das Runden verursachten Sequenzierfehler.

Die Idee ist, jeden Flow mit einem Teilstring der Referenz zu paaren, wobei die Konkatenation aller Teilstrings die Referenz ergibt und einzelne Teilstrings leer sein dürfen (ε). Außerdem dürfen Teilstrings der Referenz mit einem „Gap“($-$) gepaart werden. Ein Flowgramm-Alignment sieht beispielsweise so aus (auch in Abb. 3 zu sehen):

$\mathtt{T}^{1.0}$	$\mathtt{A}^{2.1}$	$\mathtt{C}^{1.1}$	$\mathtt{G}^{2.2}$	$-$	$\mathtt{T}^{3.1}$	$\mathtt{A}^{0.1}$	$\mathtt{C}^{2.7}$	$\mathtt{G}^{0.2}$
T	AA	ε	GG	A	TGT	ε	CC	ε

Es sind also alle Arten von Veränderungen darstellbar: $\mathtt{T}^{3.1}$ aligniert an TGT enthält einen Substitution; $\mathtt{C}^{1.1}$ aligniert an ε ist eine Insertion und der Gap aligniert an A ist eine Deletion. Wie andere Alignments auch lässt sich ein Flowgramm-Alignment als ein Pfad in einem Alignment-Graphen darstellen (siehe Abb. 3). Jeder Knoten in Spalte i (Randknoten ignorierend) hat eine Kante zu seinem linken Nachbarn und zu allen Knoten $1, \ldots, i$ in der Zeile darüber. Es stellen sich nun zwei Fragen: Wie kann ein optimales Alignment, also der optimale Pfad gefunden werden und welche Scorefunktion sollte für die Kanten gewählt werden?

Optimaler Pfad. Der optimale Pfad lässt sich, wie in regulären Alignments auch, durch dynamische Programmierung finden, indem an jedem Knoten über den Score des Vorgängers plus den Score entlang der Kante zum Vorgänger maximiert wird.

Kantenscore. Ein Kantenscore muss bewerten, wie gut ein Flow zu einem Teilstring der Referenz passt, also z. B. $\texttt{T}^{3.1}$ zu `TGT`. Genau genommen sind hier zwei unterschiedliche Prozesse involviert: Zum einen gibt es Unterschiede zwischen der Referenz und der zu untersuchenden Probe, die dazu führen, dass ein Homopolymer entsteht. In diesem Beispiel wurde (augenscheinlich) aus `TGT` so `TTT`. Zum anderen gibt es Messfehler bei der Intensitätsmessung. Hier ist also die Messung von 3.1 statt 3.0 zu bewerten. Wir setzen daher den Score aus zwei Termen s_{edit} und s_{measure} zusammen.

s_{edit} ist der Score für ein Alignment des Teilstrings der Referenz an ein Homopolymer einer vorgegebenen Länge ℓ. Ein Alignment erfordert normalerweise quadratische Laufzeit, aber aufgrund der speziellen Struktur des Problems können wir eine geschlossene Formel für diesen Teilscore angeben, sodass er sich effektiv in $O(1)$ berechnen lässt. s_{measure} ist der Score, der die Messung bewertet und wir lernen die Scorefunktion, wie auch schon Vacic et al. [VJZL08], aus vorliegenden, zuverlässigen Alignments (log-odds-Scores).

Entscheidend ist noch ein weiteres Detail: Die tatsächliche Länge ℓ des Homopolymers ist nicht bekannt, so muss also auch z. B. die Möglichkeit berücksichtigt werden, dass `TGT` mit einer Deletion zu `TT` wurde ($\ell = 2$) und dann mit Intensität 3.1 gemessen wurde. Dies erfordert für eine einzelne Scoreberechnung theoretisch eine Maximierung über alle möglichen Längen ℓ. In der Dissertation können wir zeigen, dass sich dies mit wenig Speicherbedarf einfach vorberechnen lässt, sodass ein einzelner Kantenscore dennoch in $O(1)$ berechnet werden kann.

Wir können schließlich zeigen, dass mit unserer neuen Methode des direkten Flowgramm-Alignment die Sequenzierfehlerrate im Vergleich zu der naiven Rundungsmethode deutlich sinkt.

6 Abschlussbemerkungen

Bioinformatik-Forschung ist ein Feld der Informatik, in dem die Nützlichkeit für die Praxis besonders im Vordergrund steht, also letztlich ob mit einer neuen Methode Fragen der Biologie besser oder eventuell überhaupt erst beantwortet werden können. Diese Arbeit hält sich an diesen Grundsatz und neben einer soliden theoretischen Grundlage der Algorithmen wurde auf gute Benutzbarkeit der Implementierung geachtet. Insbesondere *Exomate* und *cutadapt* haben sich im praktischen Einsatz bewährt, letzteres ist weltweit in vielen Bioinformatik-Gruppen im praktischen Einsatz.

Literatur

[DBP+11] Mark A DePristo, Eric Banks, Ryan Poplin, Kiran V Garimella, Jared R Maguire, Christopher Hartl, Anthony A Philippakis, Guillermo Del Angel, Manuel A Rivas, Matt Hanna et al. A framework for variation discovery and genotyping using next-generation DNA sequencing data. *Nature Genetics*, 43(5):491–498, Mai 2011.

[Gus97] Dan Gusfield. *Algorithms on Strings, Trees and Sequences*. Cambridge University

Press, Mai 1997.

[LD09] Heng Li und Richard Durbin. Fast and accurate short read alignment with Burrows-Wheeler transform. *Bioinformatics*, 25(14):1754–1760, Juli 2009.

[LLB+01] E. S. Lander, L. M. Linton, B. Birren, C. Nusbaum, M. C. Zody, J. Baldwin, K. Devon, K. Dewar, M. Doyle, W. FitzHugh et al. Initial sequencing and analysis of the human genome. *Nature*, 409(6822):860–921, Feb 2001.

[Mar13] Marcel Martin. *Algorithms and Tools for the Analysis of High-Throughput DNA Sequencing Data*. Dissertation, TU Dortmund, 2013.

[MEA+05] Marcel Margulies, Michael Egholm, William E Altman, Said Attiya, Joel S Bader, Lisa A Bemben, Jan Berka, Michael S Braverman, Yi-Ju Chen, Zhoutao Chen et al. Genome sequencing in microfabricated high-density picolitre reactors. *Nature*, 437(7057):376–380, September 2005.

[NTR+09] Sarah B. Ng, Emily H. Turner, Peggy D. Robertson, Steven D. Flygare, Abigail W. Bigham, Choli Lee, Tristan Shaffer, Michelle Wong, Arindam Bhattacharjee, Evan E. Eichler et al. Targeted capture and massively parallel sequencing of 12 human exomes. *Nature*, 461(7261):272–276, September 2009.

[SNC77] F. Sanger, S. Nicklen und A. R. Coulson. DNA sequencing with chain-terminating inhibitors. *Proceedings of the National Academy of Sciences of the United States of America*, 74(12):5463–5467, Dezember 1977.

[VJZL08] Vladimir Vacic, Hailing Jin, Jian-Kang Zhu und Stefano Lonardi. A probabilistic method for small RNA flowgram matching. *Pacific Symposium on Biocomputing*, Seiten 75–86, 2008.

[ZMB+09] Michael Zeschnigk, Marcel Martin, Gisela Betzl, Andreas Kalbe, Caroline Sirsch, Karin Buiting, Stephanie Gross, Epameinondas Fritzilas, Bruno Frey, Sven Rahmann et al. Massive parallel bisulfite sequencing of CG-rich DNA fragments reveals that methylation of many X-chromosomal CpG islands in female blood DNA is incomplete. *Human Molecular Genetics*, 18(8):1439–1448, April 2009.

Marcel Martin studierte Naturwissenschaftliche Informatik an der Universität Bielefeld. Mit der Bioinformatik kam er in Kontakt, als er dort in der AG Genominformatik als studentische Hilfskraft arbeitete. Seine Diplomarbeit schrieb er über Clusteringalgorithmen für Graphen, wie sie beim Vergleich von bakteriellen Proteinsequenzen entstehen. Für die Promotion wechselte er zur TU Dortmund in die dort gerade entstandene Arbeitsgruppe *Bioinformatik für Hochdurchsatztechnologien*. Seit Januar 2014 ist er als Postdoc am SciLifeLab in Stockholm.