Genome Rearrangement Algorithmen

Martin Bader

Vector Informatik GmbH
Ingersheimer Straße 24
70499 Stuttgart
martin.bader@vector.com

Abstract: Durch die steigende Anzahl von vollständig sequenzierten Genomen gewinnt der Vergleich von Spezies auf Basis der sequenzierten Genome immer mehr an Bedeutung. Im Gegensatz zum klassischen Ansatz, welcher ein Distanzmaß basierend auf Punktmutationen verwendet, lassen *Genome Rearrangement Algorithmen* kleinere Mutationen ausser acht und berücksichtigen nur größere Umstrukturierungen, welche die Reihenfolge der Gene auf den Chromosomen verändern. Dadurch sind diese Algorithmen ein wertvolles Hilfsmittel zum Vergleich von Spezies, welche seit Millionen von Jahren divergieren, sowie von sich schnell verändernden Genomen, wie sie zum Beispiel in Krebszellen vorkommen.
Die Untersuchung dieser Genomumstrukturierungen führt zu einer Vielzahl von algorithmischen Fragestellungen, wie die Berechnung von evolutionären Distanzen, die Rekonstruktion evolutionärer Ereignisse und der Genreihenfolge in hypothetischen Vorfahren, bis hin zur Berechnung ganzer phylogenetischer Bäume. In der hier vorgestellten Dissertation [Bad11] werden einige dieser Problemstellungen sowohl von der theoretischen als auch von der praktischen Seite untersucht. So wird gezeigt, dass das Median-Problem, bei welchem ein möglichst plausibler Vorfahr für drei gegebene Genome gesucht wird, sowohl für die *Transpositionsdistanz* als auch für die *gewichtete Reversal- und Transpositionsdistanz* zu den NP-vollständigen Problemen gehört. Dazu wird ein Algorithmus vorgestellt, welcher in der Praxis vorkommende Instanzen dieser Probleme dennoch lösen kann. Desweiteren werden ein neuer Algorithmus zur phylogenetischen Rekonstruktion basierend auf diesen Distanzen sowie erstmals Algorithmen zum paarweisen Genomvergleich, welche Duplikationen und Deletionen beliebiger Länge zulassen, vorgestellt. Ein weiteres Ergebnis der Dissertation, auf welches in dieser Zusammenfassung jedoch nicht näher eingegangen werden soll, ist eine effiziente Implementierung eines Vor- und Nachverarbeitungsschritt, welcher für viele Genome Rearrangement Algorithmen benötigt wird.

1 Einleitung

Im Laufe der Evolution wird ein Genom immer wieder durch größere Umstrukturierungen, den sogenannten *Genome Rearrangements*, verändert. Diese evolutionären Operationen können sowohl die Reihenfolge der Gene auf den Chromosomen als auch die Orientierung der Gene verändern. Unter der Orientierung eines Genes versteht man hierbei die Information, auf welchem Strang der Doppelhelix sich die codierende Nukleotidsequenz des Genes befindet. Bei einem *Reversal* zum Beispiel wird ein Stück eines Chromosoms

ausgeschnitten und umgedreht an der selben Stelle wieder eingefügt. Hierbei wird der Rückwärtsstrang des ausgeschnittenen Abschnittes in den Vorwärtsstrang des Chromosoms und der Vorwärtsstrang des ausgeschnittenen Abschnittes in den Rückwärtsstrang des Chromosoms eingefügt. Dadurch ändert sich sowohl die Reihenfolge aller Gene im ausgeschnittenen Abschnitt, als auch deren Orientierung. Ein beispielhaftes Reversal ist in Abb. 1 dargestellt.

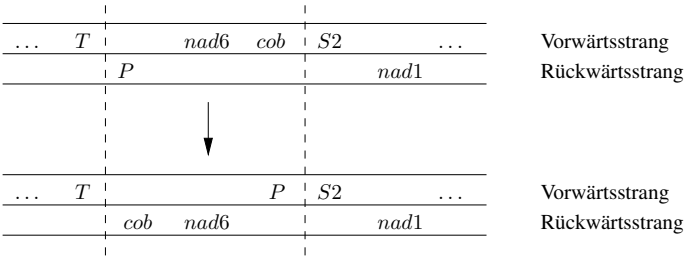

Abbildung 1: Beispiel für ein Reversal. Die drei Gene P, $nad6$ und cob ändern sowohl ihre Reihenfolge als auch den Strang des Chromosoms.

Für die algorithmische Betrachtung von Genome Rearrangements sind lediglich die unterschiedliche Reihenfolge und Orientierung der Gene auf den Chromosomen in verschiedenen Organismen von Interesse, jedoch nicht die genaue Bezeichnung der Gene. Deshalb kann hier eine abstraktere Schreibweise verwendet werden: In einem Genom, dem Referenzgenom, werden die Gene von 1 bis n durchnummeriert, ein Pfeil über der Zahl gibt die Orientierung des Gens an (wobei $\overrightarrow{x}$ der Vorwärtsstrang und $\overleftarrow{x}$ der Rückwärtsstrang ist). In den anderen Genomen werden die Gene durch die entsprechenden Nummern aus dem Referenzgenom ersetzt. In obigen Beispiel könnte also das erste Genom als $(\dots \overrightarrow{1}\overleftarrow{2}\overrightarrow{3}\overrightarrow{4}\overrightarrow{5}\overleftarrow{6} \dots)$ und das Ergebnis des Reversals als $(\dots \overrightarrow{1}\overleftarrow{4}\overleftarrow{3}\overrightarrow{2}\overrightarrow{5}\overleftarrow{6} \dots)$ geschrieben werden. Im Folgenden werden wir zwei weitere vereinfachende Annahmen machen, welche für bakterielle und mitochondriale Genome weitestgehend zutreffen.

1. Jedes Genom besteht aus genau einem zirkulären Chromosom (d.h. das Chromosom hat eine ringförmige Struktur und keinen fest definierten Anfangspunkt).

2. Jedes der betrachteten Gene kommt in jedem Genom exakt einmal vor.

Dadurch ergibt sich folgende mathematische Definition eines Genoms:

Definition 1 *Ein* Genom $\pi = (\pi_1 \dots \pi_n)$ *ist ein Wort über dem Alphabet* $\Sigma_n = \{1, \dots, n\}$*, in welchem die Indizes zyklisch sind (d.h. der Nachfolger von* n *ist* 1*) und jedes Element* x *eine positive oder negative Orientierung besitzt (gekennzeichnet durch* $\overrightarrow{x}$ *oder* $\overleftarrow{x}$*).*

Ein Teilwort $\pi_i \dots \pi_j$ eines Genoms bezeichnen wir als *Segment*. Auf einem Genom lassen sich jetzt die folgenden drei evolutionären Operationen definieren:

Definition 2 *Ein* Reversal $rev(i,j)$ *angewandt auf ein Genom* $\pi = (\pi_1 \dots \pi_n)$ *schneidet das Segment* $\pi_i \dots \pi_{j-1}$ *aus einem Genom, invertiert die Reihenfolge und Orientierung aller Gene des Segments, und fügt das Segment wieder an der selben Stelle in das Genom ein. Das heißt,*

$$rev(i,j) \cdot \pi = (\pi_1 \dots \pi_{i-1} - \pi_{j-1} \dots - \pi_i \pi_j \dots \pi_n),$$

wobei $-\pi_k$ *das Gen* π_k *mit invertierter Orientierung ist.*
Eine Transposition $tp(i,j,k)$ *angewandt auf ein Genom* $\pi = (\pi_1 \dots \pi_n)$ *schneidet das Segment* $\pi_i \dots \pi_{j-1}$ *aus einem Genom und fügt es vor dem Element* π_k *wieder ein. Das heißt,*

$$tp(i,j,k) \cdot \pi = (\pi_1 \dots \pi_{i-1} \pi_j \dots \pi_{k-1} \pi_i \dots \pi_{j-1} \pi_k \dots \pi_n).$$

Eine invertierte Transposition $itp(i,j,k)$ *angewandt auf ein Genom* $\pi = (\pi_1 \dots \pi_n)$ *ist die Konkatenation des Reversals* $rev(i,j)$ *und der Transposition* $tp(i,j,k)$ *angewandt auf* π*. Das heißt,*

$$itp(i,j,k) \cdot \pi = tp(i,j,k) \circ rev(i,j) \cdot \pi = (\pi_1 \dots \pi_{i-1} \pi_j \dots \pi_{k-1} - \pi_{j-1} \dots - \pi_i \pi_k \dots \pi_n).$$

Tatsächlich sind diese drei Operationen die drei in der Evolution am häufigsten vorkommenden Genome Rearrangements bei unichromosomalen Genomen. Anhand dieser Operationen lassen sich nun Genome Rearrangement Distanzen definieren:

Definition 3 *Die* Reversal- und Transpositionsdistanz $d_{rt}(\pi^1, \pi^2)$ *zwischen zwei Genomen* π^1 *und* π^2 *ist die minimale Anzahl an Reversals, Transpositionen und invertierten Transpositionen, welche benötigt wird, um* π^1 *in* π^2 *zu transformieren.*

Die *Reversaldistanz* $d_{rev}(\pi^1, \pi^2)$ und die *Transpositionsdistanz* $d_{tp}(\pi^1, \pi^2)$ sind analog definiert, nur dass hier die Operationen auf Reversals bzw. Transpositionen beschränkt sind. Bei der *gewichteten Reversal- und Transpositionsdistanz* $d_{wrt}(\pi^1, \pi^2)$ sind die verschiedenen Operationen unterschiedlich gewichtet, die Distanz ist hier das minimale Gewicht einer Sequenz von Operationen, welche π^1 in π^2 transformiert. Eine Sequenz von Operationen, welche π^1 in π^2 transformiert, nennt man eine sortierende Sequenz. In Abb. 2 ist eine sortierende Sequenz dargestellt, welche das Genom $\pi^1 =$

$$rev(2,5) \cdot (\overrightarrow{1}\ \overleftarrow{2}\ \overrightarrow{9}\ \overleftarrow{5}\ \overrightarrow{3}\ \overrightarrow{7}\ \overrightarrow{4}\ \overleftarrow{8}\ \overleftarrow{6}\ \overrightarrow{10}) = (\overrightarrow{1}\ \overrightarrow{5}\ \overleftarrow{9}\ \overrightarrow{2}\ \overrightarrow{3}\ \overrightarrow{7}\ \overrightarrow{4}\ \overleftarrow{8}\ \overleftarrow{6}\ \overrightarrow{10})$$

$$itp(6,7,9) \cdot (\overrightarrow{1}\ \overrightarrow{5}\ \overleftarrow{9}\ \overrightarrow{2}\ \overrightarrow{3}\ \overrightarrow{7}\ \overrightarrow{4}\ \overleftarrow{8}\ \overleftarrow{6}\ \overrightarrow{10}) = (\overrightarrow{1}\ \overrightarrow{5}\ \overleftarrow{9}\ \overrightarrow{2}\ \overrightarrow{3}\ \overrightarrow{4}\ \overleftarrow{8}\ \overleftarrow{7}\ \overleftarrow{6}\ \overrightarrow{10})$$

$$tp(2,4,7) \cdot (\overrightarrow{1}\ \overrightarrow{5}\ \overleftarrow{9}\ \overrightarrow{2}\ \overrightarrow{3}\ \overrightarrow{4}\ \overleftarrow{8}\ \overleftarrow{7}\ \overleftarrow{6}\ \overrightarrow{10}) = (\overrightarrow{1}\ \overrightarrow{2}\ \overrightarrow{3}\ \overrightarrow{4}\ \overrightarrow{5}\ \overleftarrow{9}\ \overleftarrow{8}\ \overleftarrow{7}\ \overleftarrow{6}\ \overrightarrow{10})$$

$$rev(6,10) \cdot (\overrightarrow{1}\ \overrightarrow{2}\ \overrightarrow{3}\ \overrightarrow{4}\ \overrightarrow{5}\ \overleftarrow{9}\ \overleftarrow{8}\ \overleftarrow{7}\ \overleftarrow{6}\ \overrightarrow{10}) = (\overrightarrow{1}\ \overrightarrow{2}\ \overrightarrow{3}\ \overrightarrow{4}\ \overrightarrow{5}\ \overrightarrow{6}\ \overrightarrow{7}\ \overrightarrow{8}\ \overrightarrow{9}\ \overrightarrow{10})$$

Abbildung 2: Das Genom $(\overrightarrow{1}\ \overleftarrow{2}\ \overrightarrow{9}\ \overleftarrow{5}\ \overrightarrow{3}\ \overrightarrow{7}\ \overrightarrow{4}\ \overleftarrow{8}\ \overleftarrow{6}\ \overrightarrow{10})$ wird durch 4 Operationen in das Genom $(\overrightarrow{1}\ \overrightarrow{2}\ \overrightarrow{3}\ \overrightarrow{4}\ \overrightarrow{5}\ \overrightarrow{6}\ \overrightarrow{7}\ \overrightarrow{8}\ \overrightarrow{9}\ \overrightarrow{10})$ transformiert.

$(\overrightarrow{1}\overleftarrow{2}\overrightarrow{9}\overleftarrow{5}\overrightarrow{3}\overrightarrow{7}\overrightarrow{4}\overleftarrow{8}\overleftarrow{6}\overrightarrow{10})$ in das Genom $\pi^2 = (\overrightarrow{1}\overrightarrow{2}\overrightarrow{3}\overrightarrow{4}\overrightarrow{5}\overrightarrow{6}\overrightarrow{7}\overrightarrow{8}\overrightarrow{9}\overrightarrow{10})$ mit zwei Reversals, einer Transposition und einer invertierten Transpositionen transformiert. Es lässt sich zeigen, dass diese Sequenz minimal ist, also ist $d_{rt}(\pi^1, \pi^2) = 4$.
Ein weiteres häufig verwendetes Distanzmaß ist die *Breakpointdistanz*. Diese ist jedoch nicht über die Länge einer sortierenden Sequenz definiert, sondern es wird lediglich die Anzahl der unterschiedlichen Genübergänge in den Genomen gezählt.

2 Median-Probleme

Bei einem Medianproblem wird nach einem Genom gesucht, welches „möglichst mittig“ zwischen drei anderen Genomen liegt. Formal besteht die Eingabe aus drei Genomen π^1, π^2 und π^3, und es wird nach einem Genom σ gesucht, welches unter einem gegebenen Distanzmaß $d(\cdot, \cdot)$ die Summe der Distanzen $\Sigma_{i=1}^{3} d(\sigma, \pi^i)$ minimiert. Ein Programm, welches das Medianproblem entweder exakt oder approximativ löst, nennt man Mediansolver. Solche Mediansolver werden in allen aktuellen Programmen zur phylogenetischen Rekonstruktion basierend auf Genome Rearrangements verwendet. Allerdings sind die Medianprobleme ungleich schwerer zu berechnen als die korrespondieren paarweisen Distanzfunktionen. So ist die Reversaldistanz in linearer Zeit berechenbar [BMY01], Caprara konnte jedoch 1999 die NP-Vollständigkeit des *Reversal-Medianproblems* (kurz *RMP*) beweisen [Cap99, Cap03]. Ausgangspunkt dieses Beweises ist eine Variante des *Eulerian Cycle Decomposition Problem* (kurz *ECD*), bei dem gefragt ist, ob sich ein Graph vollständig in kantendisjunkte Dreiecke zerlegen lässt. Die NP-Vollständigkeit von ECD wurde bereits 1981 von Holyer bewiesen [Hol81]. Caprara reduziert in seinem Beweis zunächst ECD auf das *Zyklen-Medianproblem*, wobei für drei Genome π^1, π^2 und π^3 ein Genom σ gesucht wird, welches die Anzahl der Zyklen $\Sigma_{i=1}^{3} c(\sigma, \pi^i)$ im *multiplen Breakpointgraph* maximiert. Diese Zyklenanzahl ist eng mit der Reversaldistanz verknüpft, es gilt folgende Ungleichung:

$$\Sigma_{i=1}^{3} d_{rev}(\sigma, \pi^i) \geq 3n - \Sigma_{i=1}^{3} c(\sigma, \pi^i)$$

Anschließend wird gezeigt, dass der Reversalmedian gleichzeitig auch der Zyklenmedian ist, da im konstruierten Graph für den Zyklenmedian in obiger Formel die Gleichheit gilt. Dieser Schritt vervollständigt also die Reduktion von ECD auf das RMP.
Für die Transpositionsdistanz war der Status für sowohl die paarweise Distanz als auch für das Medianproblem lange Jahre offen. In meiner Dissertation gelang es mir zu beweisen, dass zumindest das *Transpositions-Medianproblem* (kurz *TMP*) ebenfalls NP-vollständig ist[1]. Dazu musste der Beweis von Caprara an verschiedenen Stellen abgeändert werden. So musste sichergestellt werden, dass beim Zyklusmedian alle Gene dieselbe Orientierung haben. Dieser Schritt war für das RMP nicht notwendig, da Reversals die Orientierung von Genen verändern. Bei der Transpositiondistanz wird jedoch nie die Orientierung eines Gens geändert, der Zyklusmedian könnte also nicht durch Transpositionen in die Ausgangsgenome transformiert werden und wäre somit keine gültige Lösung für das TMP.

[1] Mittlerweile konnte sogar die NP-Vollständigkeit der paarweisen Transpositionsdistanz bewiesen werden [BFR11].

Eine weitere Schwierigkeit beim TMP ist, dass für die untere Schranke die Länge der Zyklen im multiplen Breakpointgraph beachtet werden muss. Es gilt:

$$\Sigma_{i=1}^{3} d_{tp}(\sigma, \pi^i) \geq \frac{1}{2} \cdot (3n - \Sigma_{i=1}^{3} c_{odd}(\sigma, \pi^i))$$

Hierbei ist $c_{odd}(\sigma, \pi^i)$ die Anzahl der Zyklen ungerader Länge zwischen σ und π^i im multiplen Breakpointgraph. Schließlich muss noch gezeigt werden, dass für den Zyklusmedian bei obiger Formel die Gleichheit gilt. Dies ist jedoch nicht ohne weiteres möglich, da kein polynomieller Algorithmus für die Transpositionsdistanz zur Verfügung steht. Daher ist ein weiterer Reduktionsschritt nötig, welcher die Genome so modifiziert, dass die Transpositionsdistanzen zwischen diesen Genomen einfach zu berechnen sind.
Der Beweis lässt sich auch einfach auf das *gewichtete Reversal- und Transpositions-Medianproblem* (kurz *wRTMP*) übertragen, dieses ist also ebenfalls NP-vollständig. Für die Berechnung dieser Probleme in der Praxis habe ich einen Branch-and-Bound Algorithmus entwickelt, welcher ebenfalls auf der Arbeit von Caprara beruht. Der Algorithmus kann den TMP oder den wRTMP wahlweise exakt oder approximativ berechnen. Experimentelle Ergebnisse zeigen, dass der Algorithmus sowohl schneller ist als auch eine bessere Approximationsgüte besitzt als die einzige andere verfügbare Implementierung eines Mediansolvers für das TMP [YZT08] (welcher jedoch auf einer anderen Technik beruht). Für das wRTMP gibt es keine andere Implementierung, die praxisrelevante Instanzen des Problems lösen kann.

3 Phylogenetische Rekonstruktion

Bei der phylogenetischen Rekonstruktion geht es darum, zu einer Menge von Organismen einen möglichst plausiblen Abstammungsbaum zu ermitteln. Im Zusammenhang mit Genome Rearrangements werden die Spezies durch ihre Genome repräsentiert und die Topologie des phylogenetischen Baumes sowie Genreihenfolgen für innere Knoten des Baumes berechnet. Die Genreihenfolgen der inneren Knoten entsprechen also den Genomen hypothetischer Vorfahren. Die Güte eines phylogenetischen Baumes wird über die Distanzen benachbarter Knoten berechnet, wobei als Distanzmaß eine festgelegte Genome Rearrangement Distanz verwendet wird (also z.B. die Reversaldistanz oder die Transpositionsdistanz). Formal lässt sich dies wie folgt definieren:

Definition 4 *Ein* phylogenetischer Baum *einer Menge von Genomen $P = \{\pi^1, \ldots, \pi^k\}$ (den Eingabegenomen) ist ein Baum $T = (V, E)$, wobei V die Knoten und E die Blätter des Baumes sind. Jeder Knoten ist mit einem Genom π^i markiert, und es besteht eine Bijektion zwischen den Eingabegenomen und den Blättern des Baumes (d.h. jedes Element aus P ist Markierung von genau einem Blatt). Das Gewicht einer Kante (π^i, π^j) ist die Distanz $d(\pi^i, \pi^j)$. Das Gewicht eines Baumes $w(T)$ ist die Summe der Gewichte seiner Kanten.*

Beim *Phylogenie-Problem* geht es nun darum, zu einer Menge von Eingabegenomen einen Baum mit möglichst geringem Gewicht zu finden. Da dieses Probelm für alle bekannten

Genome Rearrangement Distanzen NP-hart ist, werden hierfür in der Praxis heuristische Algorithmen eingesetzt. Die Approximationsgüte dieser Algorithmen lässt sich rechnerisch nur sehr grob bestimmen, deshalb werden die Algorithmen anhand von Benchmark-Datensätzen getestet und verglichen. Prinzipiell finden sich in der Literatur zwei verschiedene algorithmische Strategien für das Phylogenie-Problem:

Strategie 1: Erstelle zuerst die Topologie des Baumes und berechne dann Genome für die inneren Knoten

Diese Strategie wurde erstmals im Tool `BPAnalysis` [SB98] verwendet, welches über alle möglichen Topologien iteriert und eine schnelle Heuristik zur initialen Berechnung der Genome der inneren Knoten verwendet. Anschließend werden die Genome der inneren Knoten mit Hilfe eines Mediansolvers iterativ verbessert. Dieser erste Ansatz war relativ langsam, hauptsächlich wegen der Iteration über alle möglichen Topologien. Deshalb wurde dieser Schritt durch eine Heuristik ersetzt, ebenfalls wurde die restliche Implementierung des Algorithmus fortlaufend verbessert [CJM$^+$00, MWB$^+$01, MSTL02]. Der Schwachpunkt dieser Strategie ist jedoch, dass die Topologie nicht mit der eigentlich verwendeten Genome Rearrangement Distanz, sondern mit der leichter zu berechnenden *Breakpoint Distanz* berechnet wird.

Strategie 2: Beginne mit einem leeren Baum und füge die Eingabegenome iterativ hinzu. Markiere in jedem Schritt neu hinzugekommene innere Knoten

Diese Strategie wurde zuerst in einem Softwaretool namens `MGR` implementiert [BP02]. Die Heuristik zum Hinzufügen eines Genoms verwendet einen Mediansolver für das RMP (der Einsatz von Mediansolvern für andere Distanzmaße ist auch möglich, siehe z.B. [AS08, XM10]). Eine weitere Verbesserung namens `amGRP` nutzt die Tatsache, dass der Median im Allgemeinen nicht eindeutig ist, und wählt aus allen Medianen mittels einer Heuristik denjenigen aus, welcher für den weiteren Verlauf des Algorithmus möglichst gut ist [BMM07]. Der Vorteil der Strategie ist, dass der phylogenetische Baum direkt bezüglich dem gewünschten Distanzmaß aufgebaut wird. Allerdings müssen in jeder Iteration sehr viele Mediane berechnet werden. Dies ist akzeptabel für die Reversaldistanz, die Mediansolver für das TMP und das wRTMP sind jedoch für diesen Ansatz trotz der im letzten Kapitel angesprochenen Verbesserungen zu langsam.

Um das Phylogenie-Problem bezüglich der gewichteten Reversal- und Transpositionsdistanz zu lösen, sollte ein Algorithmus also zwei Eigenschaften erfüllen. Erstens sollte der phylogenetische Baum direkt bezüglich dieser Distanz aufgebaut werden. Zweitens sollte der Einsatz von Mediansolvern weitestgehend vermieden werden. Der von mir entwickelte Algorithmus verfolgt Strategie 2, er baut also den Baum iterativ auf. Anstatt neue innere Knoten mit einem Mediansolver zu bestimmen, wird für jede Kante (π^i, π^j) eine Menge von Knoten $cloud(\pi^i, \pi^j)$ gespeichert, welche „zwischen“ π^i und π^j liegen. Formal gilt für die Knoten $\pi^s \in cloud(\pi^i, \pi^j) : \; d_{wrt}(\pi^i, \pi^s) + d_{wrt}(\pi^s, \pi^j) \leq d_{wrt}(\pi^i, \pi^j) + \delta$ für eine kleine Konstante δ. Anstatt beim Hinzufügen von einem Knoten $\pi^p \in P$ zum bestehenden Baum den Median von π^i, π^j und π^p zu berechnen, wird für alle Knoten $\pi^s \in cloud(\pi^i, \pi^j)$ die Distanz $d_{wrt}(\pi^s, \pi^p)$ berechnet. Derjenige Knoten, welcher die-

se Distanz minimiert, wird als Median verwendet. Diese Wahl ist in den meisten Fällen nicht viel schlechter als der tatsächliche Median, durch eine Beschränkung der Größe von $cloud(\pi^i, \pi^j)$ ist dieses Verfahren jedoch wesentlich schneller. Der so erstellte phylogenetische Baum durchläuft abschliessend noch zwei Verbesserungsschritte. Im ersten Schritt wird durch systematisches Entfernen und Neueinfügen die Topologie des Baumes verbessert. Dieser Schritt bedient sich abermals der „Clouds“ zwischen den Knoten. Im zweiten Schritt werden die Genome der inneren Knoten wie bei Strategie 1 beschrieben mit einem Mediansolver optimiert. Da der Baum schon vor diesem Schritt relativ gut ist, müssen hierbei nur wenige Iterationen durchlaufen werden, so dass der Einsatz des Mediansolvers nicht zu kostspielig ist.
Mangels eines anderen Tools, welches phylogenetische Bäume bezüglich der gewichteten Reversal- und Transpositionsdistanz berechnet, war nur eine indirekte Evaluierung des Algorithmus möglich. Dazu wurden phylogenetische Bäume bezüglich der Reversaldistanz mit den State-of-the-Art-Tools `GRAPPA` [MWB$^+$01], `MGR` [BP02] und `amGRP` [BMM07] berechnet und für diese Bäume das Gewicht bezüglich der gewichteten Reversal- und Transpositionsdistanz ermittelt. Dabei zeigte sich, dass der von mir gewählte direkte Ansatz zu besseren Ergebnissen führt. Anschließend wurde bei meinem Algorithmus das Distanzmaß auf die Reversaldistanz umgestellt. Hier schnitt mein Algorithmus bei ähnlicher Laufzeit wie `amGRP` nur unwesentlich schlechter als dieser ab, `GRAPPA` und `MGR` wurden sowohl in Laufzeit als auch in der Qualität der Ergebnisse deutlich geschlagen.

4 Genome Rearrangements mit Duplikationen

In den vorigen Kapiteln haben wir angenommen, dass jedes Genom aus nur einem Chromosom besteht, und dass jedes Gen in jedem Genom exakt einmal vorkommt. Diese Annahme ist für bakterielle und mitochondriale DNA praktikabel, trifft jedoch nicht für die Genome höherer Organismen zu. Insbesondere bei pflanzlicher DNA sind lange duplizierte Abschnitte der DNA häufig. Ebenso ist in der Krebsforschung die Betrachtung von Duplikationen wichtig, da hier viele der Mutationen aus solchen bestehen. Aus algorithmischer Sicht sind Duplikationen jedoch problematisch. So ist zum Beispiel die Reversaldistanz in linearer Zeit berechenbar, falls keine duplizierten Gene vorliegen. Mit duplizierten Genen ist dieses Problem jedoch NP-vollständig [CZF$^+$05]. Falls die Anzahl der Vorkommen eines Gens in den zu vergleichenden Genomen unterschiedlich sind, müssen ausserdem zusätzliche Operationen wie *Duplikationen* und *Deletionen* im Algorithmus erlaubt werden. Durch diese algorithmischen Schwierigkeiten wurden Duplikationen nur wenig untersucht. Die meisten Algorithmen, welche Duplikationen erlauben, beschränken die Länge der Duplikation auf ein einzelnes Gen oder ein Segment beschränkter Länge (siehe z.B. [EM02]). Zum Zeitpunkt meiner Dissertation gab es nur einen Ansatz, welcher Duplikationen beliebiger Länge erlaubte, allerdings auf einem stark vereinfachten Genommodell [OFS08]. In meiner Dissertation habe ich einen Algorithmus entwickelt, welcher eine sortierende Sequenz zwischen zwei Genomen π^1 und π^2 berechnet und dabei eine Vielzahl an verschiedenen Operationen berücksichtigt, unter anderem *Tandemduplikationen* und *Deletionen* beliebiger Länge. Obwohl das Startgenom π^1 der Restrik-

tion unterliegt, dass seine Chromosomen entweder identisch oder elementendisjunkt sein müssen (d.h. hier können duplizierte Gene nur bei einer exakten Kopie eines Chromosoms auftreten), kann dieses Modell in der Praxis eingesetzt werden, z.B. zur Untersuchung von Mutationen in Krebszellen. Der Algorithmus basiert auf einer Erweiterung des *Breakpoint-Graphen* (für Details siehe [BP93]). Während dieser im „klassischen Fall" (d.h. ohne duplizierte Gene) in Zyklen zerfällt, erhält man mit Duplikationen Zusammenhangskomponenten. Es lässt sich zeigen, dass die Anzahl der Komponenten maximiert wird, falls $\pi^1 = \pi^2$ gilt und π^1 oben angesprochene Restriktion erfüllt. Desweiteren kann die Anzahl der Komponenten durch jede Operation maximal um 1 erhöht werden. Daraus lässt sich eine Greedy-Strategie entwickeln: Sortiere π^2 rückwärts nach π^1 (π^1 darf nicht verändert werden, damit es die Restriktion immer erfüllt) und wende (falls möglich) eine Operation an, welche die Anzahl der Komponenten erhöht. Leider ist dies nicht in jedem Schritt möglich. Deshalb wurde ein weiteres Maß $\tau(\pi^1, \pi^2)$ eingeführt, welches die „Unordnung" zwischen den Genomen bestimmt. Dieses Maß berücksichtigt einerseits die Anzahl an Breakpoints, andererseits die absolute Anzahl der Elemente, welche in π^2 eingefügt oder gelöscht werden müssen. Falls es keine Operation gibt, welche die Anzahl der Komponenten vergrößert, wird diejenige Operation genommen, welche $\tau(\pi^1, \pi^2)$ minimiert. Ausserdem wird $\tau(\pi^1, \pi^2)$ verwendet, falls mehrere Operationen die Anzahl der Komponenten vergrößert, d.h. von diesen wird ebenfalls immer diejenige gewählt, welche $\tau(\pi^1, \pi^2)$ minimiert. Um die Terminierung des Algorithmus zu garantieren, ist ausserdem ein einfacher Fallback-Algorithmus notwendig, der in der Praxis aber nur sehr selten zum Einsatz kommt.
Zur praktischen Evaluation des Algorithmus habe ich zufällige Sequenzen von Operationen auf ein festes Genom π^1 angewandt, um ein Zielgenom π^2 zu erhalten. Nun wurde mein Algorithmus verwendet, um eine sortierende Sequenz zwischen π^1 und π^2 zu berechnen, und diese mit der zur Generierung verwendeten Sequenz verglichen. Die Ergebnisse zeigen, dass die Längen dieser Sequenzen sehr ähnlich sind, solange diese nicht zu lang sind. Ebenfalls entsprechen sich hier die relativen Häufigkeiten der verschiedenen Operationen in der berechneten und generierten Sequenz. Weitere Experimente wurden auf der *Mitelman Database of Chromosome Abberations and Gene Fusions in Cancer* [MJM10] durchgeführt, welche eine Sammlung von Krebskaryotypen beinhaltet, die von Hand aus der Literatur der letzten 20 Jahre zusammengetragen wurde. Alle beschriebenen Genome konnten von meinem Algorithmus sehr schnell analysiert werden, die resultierenden sortierenden Sequenzen entsprechen sehr gut den in der Literatur beschriebenen Sequenzen.

5 Zusammenfassung

In meiner Dissertation habe ich verschiedene Probleme im Bereich Genome Rearrangements betrachtet. Die Hauptergebnisse meiner Dissertation sind folgende:

- Eine effiziente Implementierung eines für viele Genome Rearrangement Algorithmen benötigten Vor- und Nachverarbeitungsschrittes (nicht in dieser Zusammenfassung beschrieben).

- Der Beweis der NP-Vollständigkeit des *Transpositions-Medianproblems*, sowie ein praktisch einsetzbarer Branch-and-Bound Algorithmus für eben dieses Problem und für das (ebenfalls NP-vollständige) *gewichtete Reversal- und Transpositions-Medianproblem*. Der Algorithmus bringt eine deutliche Verbesserung sowohl in der Laufzeit als auch in der Qualität der Ergebnisse im Vergleich zum bisher einzigen verfügbaren Transpositions-Mediansolver.
- Ein Algorithmus zur phylogenetischen Rekonstruktion basierend auf der *gewichteten Reversal- und Transpositionsdistanz*. Dies ist der bisher einzige Algorithmus, der direkt auf dieser Distanz arbeitet. Die Qualität der vom Algorithmus berechneten Bäume ist wesentlich besser als bei einem indirekten Ansatz über die Reversaldistanz. Selbst bei der phylogenetischen Rekonstruktion basierend auf der Reversaldistanz, für die der Algorithmus eigentlich nicht entwickelt wurde, ist er nur unwesentlich schlechter als der momentan hierfür beste verfügbare Algorithmus.
- Ein Algorithmus zum paarweisen Vergleich von Genomen mit duplizierten Genen. Der Algorithmus betrachtet eine Vielzahl von verschiedenen evolutionären Operationen und ist einer der wenigen Algorithmen, welche Duplikationen und Deletionen beliebiger Länge zulassen.

Literatur

[AS08] Z. Adam und D. Sankoff. The ABCs of MGR with DCJ. *Evolutionary Bioinformatics*, 4:69–74, 2008.

[Bad11] M. Bader. *Genome Rearrangement Algorithms*. Dissertation, Universität Ulm, 2011.

[BFR11] L. Bulteau, G. Fertin und I. Rusu. Sorting by Transpositions is Difficult. In L. Aceto, M. Henzinger und J. Sgall, Hrsg., *Proc. 38th Int. Coll. on Automata, Languages and Programming, Part I*, Seiten 654–665. Springer-Verlag, Heidelberg, 2011.

[BMM07] M. Bernt, D. Merkle und M. Middendorf. Using Median Sets for Inferring Phylogenetic Trees. *Bioinformatics*, 23:e129–e135, 2007.

[BMY01] D.A. Bader, B.M.E. Moret und M. Yan. A Linear-Time Algorithm for Computing Inversion Distance between Signed Permutations with an Experimental Study. *J. of Computational Biology*, 8:483–491, 2001.

[BP93] V. Bafna und P.A. Pevzner. Genome Rearrangements and Sorting by Reversals. In *Proc. 34th IEEE Symposium on Foundations of Computer Science*, Seiten 148–157. IEEE Computer Society Press, 1993.

[BP02] B. Bourque und P.A. Pevzner. Genome-Scale Evolution: Reconstructing Gene Orders in the Ancestral Species. *Genome Research*, 12(1):26–36, 2002.

[Cap99] A. Caprara. Formulations and Hardness of Multiple Sorting by Reversals. In S. Istrail, P. Pevzner und M.S. Waterman, Hrsg., *Proc. 3rd Annual Int. Conf. on Computational Molecular Biology*, Seiten 84–93. ACM Press, New York, 1999.

[Cap03] A. Caprara. The Reversal Median Problem. *INFORMS J. on Computing*, 15(1):93–113, 2003.

[CJM+00] M.E. Cosner, R.K. Jansen, B.M.E. Moret, D.A. Raubeson, L.-S. Wang, T. Warnow und S.K. Wyman. A New Fast Heuristic for Computing the Breakpoint Phylogeny and Experimental Phylogenetic Analyses of Real and Synthetic Data. In P. Bourne, M. Gribskov, R. Altman, N. Jensen, D. Hope, T. Lengauer, J. Mitchell, E. Scheeff, C. Smith, S. Strande und H. Weissig, Hrsg., *Proc. 8th Int. Conf. on Intelligent Systems for Molecular Biology*, Seiten 104–115. AAAI Press, Menlo Park, 2000.

[CZF+05] X. Chen, J. Zheng, Z. Fu, P. Nan, Y. Zhong, S. Lonardi und T. Jiang. The Assignment of Orthologous Genes via Genome Rearrangement. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, 2(4):302–315, 2005.

[EM02] N. El-Mabrouk. Reconstructing an Ancestral Genome Using Minimum Segments Duplications and Reversals. *J. of Computer and System Sciences*, 65:442–464, 2002.

[Hol81] I. Holyer. The NP-Completeness of Some Edge-Partition Problems. *SIAM J. on Computing*, 10(4):713–717, 1981.

[MJM10] F. Mitelman, B. Johansson und F. Mertens, Hrsg. *Mitelman Database of Chromosome Aberrations and Gene Fusions in Cancer*, http://cgap.nci.nih.gov/Chromosomes/Mitelman, 2010.

[MSTL02] B.M.E. Moret, A. Siepel, J. Tang und T. Liu. Inversion Medians Outperform Breakpoint Medians in Phylogeny Reconstruction from Gene-order Data. In R. Guigó und D. Gusfield, Hrsg., *Proc. 2nd Workshop on Algorithms in Bioinformatics*, Seiten 521–536. Springer-Verlag, Heidelberg, 2002.

[MWB+01] B.M.E. Moret, S.K. Wyman, D.A. Bader, T. Warnow und M. Yan. A New Implementation and Detailed Study of Breakpoint Analysis. In R.B. Altman, A.K. Dunkerand L. Hunker, K. Lauderdale und T.E. Klein, Hrsg., *Proc. 6th Pacific Symposium on Biocomputing*, Seiten 583–594, 2001.

[OFS08] M. Ozery-Flato und R. Shamir. Sorting Cancer Karyotypes by Elementary Operations. In C. Nelson und S. Vialette, Hrsg., *Proc. 6th Annual RECOMB Satellite Workshop on Comparative Genomics*, Seiten 211–225. Springer-Verlag, Heidelberg, 2008.

[SB98] D. Sankoff und M. Blanchette. Multiple Genome Rearrangement and Breakpoint Phylogeny. *J. of Computational Biology*, 5(3):555–570, 1998.

[XM10] A.W. Xu und B.M.E. Moret. Genome Rearrangement Analysis on High-Resolution Data. Submitted, 2010.

[YZT08] F. Yue, M. Zhang und J. Tang. Phylogenetic Reconstruction from Transpositions. *BMC Genomics*, 9(Suppl 2):S15, 2008.

Dr. Martin Bader wurde am 4. Februar 1980 In Friedrichshafen geboren. Von 2000 bis 2005 studierte er Informatik an der Universität Ulm. Sein Diplom schloss er mit Gesamtnote 1,0 ab, seine Diplomarbeit zum Thema „Sorting by weighted transpositions and reversals“ wurde mit dem NRW Undergraduate Science Award 2005 in der Kategorie Bioinformatics and Genome Research ausgezeichnet. Nach seinem Abschluss als Diplominformatiker promovierte er im Institut für Theoretische Informatik an der Universität Ulm. Am 6. Mai 2011 schloss er seine Promotion zum Thema „Genome Rearrangement Algorithms“ mit dem Gesamtprädikat „summa cum laude“ ab. Seitdem arbeitet er für die Vector Informatik GmbH in Stuttgart als Software Development Engineer.