

Seminar  
Aktuelle Themen der Bioinformatik SS 2006

# Identifying Conserved Gene Clusters in the Presence of Homology Families

Von Florian Rorsch

Matrikelnr: 2695934  
Studiengang: Bioinformatik

## **Inhalt:**

- Einleitung  
*Einführung in das Themengebiet*
- Vom Wolf zum Gen-Cluster  
*Kleine Geschichte zur Erklärung des biologischen Hintergrunds*
- Wissenschaftlicher Nutzen  
*Interessante Forschungsthemen und –Gebiete*
- Modellbildung  
*Umsetzung der biologischen Realität in ein rechnerkonformes Modell*
- Der Algorithmus  
*Vorstellung der einzelnen Bestandteile und der Grundidee*
- Laufzeit und Speicherplatzbedarf  
*Angabe der O-Notation und Bedingungen unter denen diese Schranken erreicht werden können*
- Ergebnisse mit realen Daten  
*Quantitative Analyse*
- Interpretation der Ergebnisse  
*Bedeutung und Nutzen der ermittelten Cluster*
- Eine Implementierung  
*Umsetzung des Algorithmus durch M. Goldwasser*
- Statistische Signifikanz der Ergebnisse  
*Klärung der Frage, ob ein gefundener Cluster statistisch abgesichert ist*
- Themenverwandte Arbeiten

## **Einleitung:**

### *Einführung in das Themengebiet*

In dieser Ausarbeitung soll es um das weit reichende Gebiet der Clustersuche, Analyse und Nutzen gehen.

Cluster bedeutet übersetzt soviel wie Gruppierung bzw. Haufen. Grob gesagt werden also Anhäufungen von Genen untersucht. Dabei wird des Weiteren auf die Bedeutung von Homologen Familien und die biologische Relevanz von konservierten Bereichen eingegangen.

Zu Beginn sollte man definieren was ein Gen-Cluster, sowie eine Homologe Familie ist, um spätere Verwechslungen mit anderen ähnlich klingenden Definitionen auszuschließen.

#### *Gen-Cluster:*

Unter einem Gen-Cluster versteht man aus biologischer Sicht einen Bereich auf dem Träger der Erbinformationen (meist DNS), in dem eine Anhäufung bestimmter Gene zu finden ist.

#### *Homologe Familie:*

Abstammung einer oder mehrerer Gene von einem Ursprungsgen und daraus resultierend ähnliche Funktion und Struktur  
NOCHMAL NACHSCHAUEN

## **Vom Wolf zum Gen-Cluster:**

### *Kleine Geschichte zur Erklärung des biologischen Hintergrunds*

Zu aller erst möchte ich auf die biologische Relevanz der Clusterbildung eingehen. In der Natur würde sich keine Errungenschaft durchsetzen, wenn Sie für das Individuum bzw. die Art nicht einen Vorteil bringen würde. Bezogen auf genetische Cluster bedeutet das: Irgendeinen Vorteil muss es geben.

Dieser offenbart sich anschaulicherweise durch eine kleine Geschichte:

Ein Mann geht im Wald spazieren. Es wird langsam dunkel und weit und breit ist keine Menschenseele zu sehen. Plötzlich taucht aus dem Unterholz ein Wolf auf. Er fletscht die Zähne während er ein lautes kehliges Knurren von sich gibt. Er kommt immer näher...

Ich denke jeder kann sich jetzt bestimmt vorstellen wie es dem Mann geht:

- Der Puls rast
- Alle Muskeln sind angespannt
- Jeglicher Gedanke, der vorher da war ist weg (zum Beispiel das leckere Abendessen nach dem Spaziergang)
- Die Körpertemperatur steigt u.s.w.

Das ganze ist eine komplexe Zusammensetzung von unterschiedlichen Reaktionsmechanismen im Körper. Sinn der „Aufregung“ soll eine möglichst schnelle Handlungsbereitschaft sein. In diesem Fall wäre es vielleicht sinnvoll die Beine in die Hand zu nehmen und wegzulaufen. Damit diese Bereitschaft möglichst schnell

erfolgt müssen haufenweise Gene abgelesen werden. Liegen die Transkriptionsbereiche für eine Folge von Reaktionen nah beieinander oder sogar unter gleicher Transkriptionskontrolle kann dies schneller geschehen, wie bei einem Genom, welches in einem ungeordneten Zustand vorliegt. Ungeordnet bedeutet hier, dass alle Gene zufällig über das Genom verteilt sind.

Abgesehen von der gesteigerten Effizienz ergibt sich für einen Organismus mit geclustertem Genom ein weiterer Vorteil: Die Qualitätssicherung. Wurde irgendwann einmal eine vorteilhafte Errungenschaft während der Evolution entwickelt, wäre es natürlich nicht sinnvoll diese gleich wieder zu verwerfen. Deshalb haben sich Bereiche im Genom entwickelt, die besonders beständig sind – in der Fachsprache auch „konserviert“ genannt. Gene in konservierten Bereichen sind wesentlich seltener von Veränderungen betroffen – häufig aus dem Grund, weil Veränderungen hier oft letal sind. Aus diesem Grund sammeln sich im Laufe der Evolution oft auch Gene in einem konservierten Cluster, die für das Individuum unerlässlich sind – wie z.B. Gene, deren Produkte im Energiehaushalt verwendet werden.

### **Wissenschaftlicher Nutzen:**

*Interessante Forschungsthemen und –Gebiete*

Aus Sicht der Forschung ergeben sich durch das Vorhandensein von Clustern zwei wichtige Themengebiete. Das erste beschäftigt sich mit der Phylogenie. Da Cluster sehr lange bestehen bleiben und kaum Veränderungen unterworfen sind, sind sie wesentlich besser für Sequenzvergleiche von weit entfernten Arten geeignet, wie beispielsweise komplette Genomvergleiche, da bei letzterem viel zu viele Veränderungen statt gefunden haben und eine exakte Rekonstruktion oft nicht mehr möglich ist.

Das zweite Gebiet basiert auf der Eigenschaft, dass Gene in einem Cluster häufig eine ähnliche Funktion haben und / oder an einem Reaktionsmechanismus beteiligt sind. Dies kann als ein Anhaltspunkt für die Analyse von Wirkung und Funktion von Genen genutzt werden. Zielgruppen sind hierbei Medizin, Pharmazie und Chemie.

### **Modellbildung:**

*Umsetzung der biologischen Realität in ein rechnerkonformes Modell*

Bevor diese Cluster allerdings für irgendeine Analyse verwendet werden können, müssen sie logischerweise erst einmal gefunden werden. Im Folgenden werde ich Schritt für Schritt auf eine Umsetzung der biologischen Realität in ein rechnerkonformes Modell eingehen.

Zu allererst definieren wir die Eingabemenge. Diese besteht in der Realität aus einem oder mehreren kompletten Genomen. Zur Clustersuche verwenden wir allerdings nur eine eingeschränkte Menge – die Menge aller Homologen Gene. Ein homologes Gen ist – im Unterschied zu einer homologen Familie – ein Gen, welches in zwei zu vergleichenden Genom vorliegt. Es werden also nur diejenigen Gene betrachtet, die in beiden Genomen einen oder mehrere Vertreter aufweisen.

Als nächstes soll eine genaue Definition eines solches Gens und des Trägers, dem Chromosom folgen.

Definition eines „Chromosoms“:

$$C = (\Sigma, X)$$

$\Sigma$  = Set (Menge) von homologen Familien

$X$  = Geordnete Menge von Genen

Daraus kann man ableiten, dass zu einer homologen Familie mehrere Gene gehören können, jedes Gen aber eindeutig einer Familie zugeordnet werden kann bzw. aus der Menge der betrachteten Gene entfernt wird.

Definition eines „Gens“:

$$g = (p, f)$$

$p$  = Physikalische Position des Gens auf dem Chromosom

$f (\in \Sigma)$  = Homologie-Familie, zu der das Gen gehört

Beispiel Gen:

$g = (1, c) \leftarrow$  Gen an physikalischer Position „1“ gehört zu der Homologie-Familie „c“

Beispiel Chromosom:

$\Sigma = \{a, b, c, d, e\} \leftarrow$  Menge von Homologen Familien (Familie a, Familie b, ...)

$X = \{ (1, c), (4, e), (5, d), (6, a), (8, b), (10, e), (11, d) \}$

Eine andere kürzere Darstellung, bei der der Abstand zwischen zwei relevanten Genen deutlicher zu erkennen ist:

$C = \langle c - - e d a - b - e d \rangle \leftarrow$  Minuszeichen (oder \*) als Platzhalter für irrelevante Gene

In der unteren Darstellung eines Chromosoms lässt jetzt schon erahnen, dass man ein Chromosom in mehrere Untereinheiten (Substrukturen) aufbrechen kann. Dies wird eine der wesentlichen Grundlagen sein, an Hand derer ein Cluster später begrenzt wird. Darum sollen jetzt Definitionen folgen, die ein Chromosom in Unterstrukturen unterteilen.

Definition einer „Subsequenz“:

Gegeben  $C = (\Sigma, X)$ , nennt man ein Paar  $(\Sigma(X'), X')$ , wobei  $X' \leq X$ , eine Subsequenz.

Definition eines „Subchromosoms“:

$C' = (\Sigma', X')$ , falls  $X'$  eine fortlaufende Teilmenge von  $X$  ist

$\Sigma'$  = Teilmenge der homologen Familien

$X'$  = Teilmenge der Gene

Zu Erläuterung wieder zwei Beispiele, die auf dem gleichen Beispielchromosom, wie in den vorherigen Beispielen basieren.

Beispiel für eine Subsequenz:

$X' = \{ (5, d), (6, a), (10, e), (11, d) \} \leftarrow$  Teilmenge der Gene eines Chromosoms

$\Sigma(X') = \{ a, b, d, e \} \leftarrow$  Teilmenge der homologen Familien

Beispiel Subchromosom:

$C' = ( \{ a, b, d, e \}, \{(5,d), (6,a), (10,e), (11,d)\} )$

Die obige Unterteilung in Subchromosom wurde rein willkürlich gewählt. Dies ist aber weder aus biologischer, noch aus informatischer Sicht sinnvoll. Deshalb muss man ein Kriterium wählen, nachdem ein Chromosom in Unterchromosomen aufgeteilt werden kann. Aus Beobachtungen von geclusterten Genomen kann man ableiten, dass zwei Cluster eigentlich nie in direkter Nachbarschaft liegen. Dies erscheint sinnvoll, da sie sonst vermutlich im Laufe der Zeit zu einem Cluster verschmolzen wären. Ein gutes Kriterium ist also der Abstand zwischen Clustern. Genau diese Idee wird auch in unserem Modell verwendet, wobei der Abstand hier durch die Anzahl Lücken zwischen zwei relevanten Genen gemessen wird. Im Modell von M. Goldwasser wird deshalb ein Parameter  $\delta$  eingeführt, der angibt, ob zwei relevante Gene als benachbart anzusehen sind oder so weit voneinander entfernt sind, dass sie eben nicht mehr als benachbart gelten.

2 Gene  $g_i$  und  $g_j$  heißen benachbart, wenn  $\Delta(g_i, g_j) < \delta$ , wobei  $\delta > 0$ ,  $\delta \in \mathbb{N}$

$\Delta(g_i, g_j) = |p_i - p_j|$

Zur Erinnerung:  $p_i$  ist die physikalische Position von Gen  $i$  auf dem Genom.

Beispiel:

Chromosom  $C = \langle c - - e d a - b - e d \rangle$

$\delta = 2$

Dann sind Gen 1 und Gen 4 nicht benachbart, da  $|1-4| = 3$  und  $3 > \delta$

Gen 6 und Gen 8 sind nach obiger Definition gerade nicht mehr benachbart, da  $|6-8| = 2$  und  $2 = \delta$

Aus den beiden Beispielen kann man schon erkennen, dass die Wahl des  $\delta$ -Parameters eine wichtige Rolle spielen wird. Sind zwei Gene benachbart gehören sie zur gleichen Subsequenz. Anders herum gesagt zerbricht ein Chromosom immer dort in zwei Subchromosomen, wo die Lücke zwischen zwei relevanten Genen  $\geq \delta$  ist. Alle (relevanten) Gene zwischen zwei Endpunkten, wo die Anzahl der Lücken zu groß ist werden jetzt zu einer Struktur zusammenfasst – dem  $\delta$ -run.

Definition eines „ $\delta$ -runs“:

$C'$  ist ein  $\delta$ -run, wenn es in Bezug auf alle relevanten Gene eine maximale  $\delta$ -Subsequenz ist.

Beispiel  $\delta$ -run für  $\delta = 1$ :

Es gibt 2  $\delta$ -runs:  $\{(1,c)\}$  und  $\{(4,e), (5,d), (6,a), (8,b), (10,e), (11,d)\}$

Im Gegensatz zu einer  $\delta$ -Subsequenz muss ein  $\delta$ -run maximal gewählt werden. Umgangssprachlich gesagt also so groß wie möglich, so lange die  $\delta$ -Regel eingehalten wird.

Bislang wurden alle Gruppierungen über Gene definiert. Entweder gehörte ein Gen zu einer Gruppe oder eben nicht. Der Autor nimmt an dieser Stelle Abstand von dieser Form und gruppiert nun nicht mehr Gene, sondern deren Familien. Der Grund hierfür liegt in der Art

der Beziehungen zwischen zwei Genomen. Bei einer Definition über Gene kann in diesem Modell lediglich eine 1-zu-1-

Grafik 1

COG type	No. COGs	Genes assigned to such COGs	
		<i>E. coli</i>	<i>B. subtilis</i>
one-to-one	578	578	578
one-to-many	299	517	568
many-to-many	260	1237	1193

Distribution of orthologous groups in the compared genomes

Beziehung simuliert werden. Das heißt, für jedes relevante Gen in Genom 1 muss es genau ein relevantes Gen in Genom 2 geben, ansonsten ist es irrelevant und fällt aus der Betrachtung heraus. Grafik 1 zeigt allerdings, dass 1-zu-1-Beziehungen nur etwa ein Drittel aller Beziehungen ausmachen. Die beiden anderen Typen von Beziehungen sind während der Evolution durch Genduplikation entstanden, bei viele-zu-viele-Beziehungen bevor sich eine Art aufgespalten hat und bei 1-zu-viele Beziehungen danach.

Deshalb ist es wichtig, dass die beiden anderen Beziehungstypen ebenfalls Beachtung in unserem Modell finden. Wie schon erwähnt folgt daher jetzt eine Zusammengehörigkeitsdefinition über Familien, statt über Gene.

Definition einer „ $\delta$ -chain“:

Formal: Gegeben ein Chromosom  $C = (\Sigma, X)$ , nennt man  $\Sigma' \subseteq \Sigma$  eine  $\delta$ -chain, falls es eine  $\delta$ -Subsequenz  $C'$  gibt, so dass  $\Sigma' = \Sigma(C')$  gilt.  $C'$  bezeugt dann eine solche  $\delta$ -chain.

Umgangssprachlich: Gruppen von homologen Familien, bei denen die darunter liegenden Gene die Delta-Regel erfüllen.

In Chromosom  $C = \langle c - - e d a - b - e d \rangle$  gibt es mehrere für  $\delta = 3$  (siehe Definition von „benachbart“)  $\delta$ -chains:

3 Beispiele (nicht alle!):

$\{ d, e \}$  bezeugt durch  $\{ (4,e), (5,d) \}$

$\{ a, b, d, e \}$  bezeugt durch  $\{ (4,e), (5,d), (6,a), (8,b), (10,e), (11,d) \}$

$\{ d, a, b \}$  bezeugt durch  $\{ (5,d), (6,a), (8,b) \}$

$\delta$ -chains sind im Gegensatz zu  $\delta$ -runs keine maximalen Subsequenzen von Genen sondern Gruppen von Homologen Familien, bei denen die  $\delta$ -Regel (Anzahl Gaps -1 kleiner  $\delta$ ) erfüllt ist!

Bislang waren Gruppierungen immer nur auf ein Genom bezogen. Ein Cluster ergibt sich aber nicht zwangsläufig dadurch, dass irgendwo eine Lücke (die allerdings über 2 Genome definiert ist!) groß genug ist. Wichtig ist vor allem auch, dass ein potentieller Cluster in mehr, wie einem Genom auftritt. Deshalb sucht man sinnvollerweise  $\delta$ -chains, die in beiden Eingabegenomen vorkommen. Diese nennt man dann aber nicht mehr  $\delta$ -chains, sondern  $\delta$ -sets.

Definition eines „ $\delta$ -sets“:

Wenn  $\Sigma$  ein Set (Menge) von Homologen Familien für 2 (!!) Chromosome C und D ist, dann ist  $\Sigma' \subseteq \Sigma$  ein  $\delta$ -set von C und D, wenn  $\Sigma'$  eine  $\delta$ -chain von sowohl C, als auch D ist.

Beispiel:

2 Chromosome C und D:

C = <c - - e d a - b - e d>

D = <a b - b a - - c - d e e - a>

$\delta=1$

-> Nicht-triviale  $\delta$ -sets von C und D: {a,b}, {a,e}, {d,e}, {a,d,e}

{a,b} ist ein  $\delta$ -set, weil zwischen Familie a und Familie b in beiden Chromosomen weniger wie  $\delta$  Gaps liegen.

Ein trivialer  $\delta$ -set wäre z.B. {c}. Dieses hat allerdings aus biologischer Sicht keine Bedeutung, da Cluster der Kardinalität eins nicht wirklich der Definition eines Clusters entsprechen. Ein Cluster ergibt sich schließlich durch den Zusammenschluss mehrerer Gene.

$\delta$ -sets können allerdings eine beliebige Größe haben (auch eins). Wählt man sie maximal nennt man sie  $\delta$ -Teams.

Definition eines „ $\delta$ -Teams“:

Ein  $\delta$ -Team ist ein maximales  $\delta$ -set bezüglich der homologen Familien.

Beispiel  $\delta$ -Team (gleiche Eingaben wie oben für  $\delta$ -sets):

Nicht-triviale  $\delta$ -teams: {a,b}, {d,e}, {a,d,e}

Unterschied zu  $\delta$ -set: Teams müssen MAXIMAL sein, dürfen also nicht durch weitere Familien erweiterbar sein, so dass die  $\delta$ -Regel erfüllt bleibt. Jedes  $\delta$ -Team ist auch gleichzeitig ein  $\delta$ -set, umgekehrt gilt das nicht!

{a,e} ist auch ein  $\delta$ -set. Es könnte aber durch d erweitert werden und ist somit kein  $\delta$ -Team.

Nun ist das Modell endlich vollständig. Ziel des folgenden Algorithmusses ist es  $\delta$ -Teams zu finden.

## Der Algorithmus:

### *Vorstellung der einzelnen Bestandteile und der Grundidee*

Vorgestellt wird der Algorithmus von Michael Goldwasser.

Eingabe dafür sind 2 Genome inklusive der Abfolge der enthaltenen Gene und der Homologen Familie, zu der jedes Gen gehört (falls vorhanden) und der  $\delta$ -Parameter. Aus den beiden Genomen kann man dann  $\Sigma$ , die Menge aller homologen Familien, sowie die Länge beider Genome aufstellen. Die Länge des 1. Genoms (C) sei m, die des 2. Genoms (D) n.

Der Algorithmus gliedert sich in 5 Funktionen. Die übergeordnete Funktion FINDTEAMS ruft alle weiteren Funktionen auf initialisiert verschiedene Variablen.

```
-----  
FINDTEAMS(C,D)  
//Globale Variablen initialisieren  
globaltime = 0  
For each f in  $\Sigma$   
    stamp[f] = 0  
    tempmarkf] = 0  
//Chromosom C in seine  $\delta$ -runs zerlegen  
localtime = MARKCOMMONALPHABET(C,D)  
Crest = C  
repeat  
    Cfirst = FINDFIRSTRUN(Crest,localtime)  
    Crest = Crest - Cfirst  
    FINDTEAMSRECURSE(B, Cfirst)  
until (Crest =  $\emptyset$ )  
-----
```

Nach der Initialisierung muss das gemeinsame Alphabet der beiden Genome markiert werden. Dazu dient die Funktion MARKCOMMONALPHABET. Dazu wird in einem globalen Array für jede homologe Familie (solche, die in beiden Genomen vorkommen) ein Integerwert abgespeichert. Bei jedem Aufruf der Funktion wird zu erst die Integer-Variable globaltime erhöht, um die einzelnen Ergebnisse der Funktionsaufrufe unterscheiden zu können. Wird die Funktion das 1. Mal aufgerufen, erhält jede gemeinsame Familie im Array stamp eine 1, beim 2. Aufruf eine 2 u.s.w. Zurückgegeben wird nur der verwendete Integerwert. Der stamp-Array ist sowieso global.

```

-----
MARKCOMMONALPHABET(A,B)
globaltime = globaltime + 1
for each g in A
    sei f die Homologie-Familie von g
    tempmark[f] = globaltime
For each g in B
    sei f die Homologie-Familie von g
    if tempmark[f] = globaltime then
        stamp[f] = globaltime
Return globaltime
-----

```

Die Funktion FINDFIRSTRUN dient dazu, wie der Name schon sagt, den ersten  $\delta$ -run zu finden. Dazu wird der  $\delta$ -Parameter als Abgrenzung genutzt. Zurückgegeben wird dann das Subchromosom (der  $\delta$ -run und die dazugehörigen Menge der Familie).

```

-----
FINDFIRSTRUN(A,timestamp)
endrun = das 1. Gen in A mit stamp[f] >= timestamp
nextgene = das Gen in A nach endrun
while(nextgene wohldefiniert und (endrun,nextgene)  $\leq$   $\delta$ ) do
    Sei f die Homologie-Familie von nextgene
    //Wenn nextgene in common family ist, erweitere den run
    endrun = nextgene
    nextgene = das Gen in A nach nextgen
Return das Subchromosom bis endrun (inklusive)
-----

```

Dieses Subchromosom wird dann als Eingabe für die Funktion FINDTEAMSRECURSE verwendet, welche rekursiv gemeinsame  $\delta$ -Teams ermittelt. Dazu werden wieder die Markierungsfunktion und die Funktion zum Finden von  $\delta$ -runs benutzt. Wurde ein gemeinsamer  $\delta$ -run gefunden, wird das daraus resultierende  $\delta$ -Team über die nicht näher spezifizierte Funktion REPORTTEAM ausgegeben.

```

FINDTEAMSRECURSE(A,B)
localtime = MARKCOMMONALPHABET(A,B)
Afirst = FINDFIRSTRUN(A,localtime)
Arest = A - Afirst
if Arest =  $\emptyset$  then
  REPORTTEAM(A,B)
else
  repeat
    FINDTEAMSRECURSE(B,Afirst)
    Afirst = FINDFIRSTRUN(Arest,localtime)
    Arest = Arest - Afirst
  until (Arest =  $\emptyset$ )
FINDTEAMSRECURSE(B,Afirst)
-----

```

### **Laufzeit und Speicherplatzbedarf:**

*Angabe der O-Notation und Bedingungen unter denen diese Schranken erreicht werden können*

Der Algorithmus ist trotz Rekursion linear im Ablauf. Bei näherer Betrachtung wird jedes Gen nur genau einmal benutzt und auch nur einmal gespeichert. Es erfolgt also keine Zwischenspeicherung oder mehrmalige Betrachtung. Daraus ergeben sich folgende Laufzeit und Speicherplatzbedarf:

Laufzeit: =  $O(n+m)$

Speicherplatzbedarf  $O(n+m)$

Zur Erinnerung:  $m$  ist die Zahl der Gene im 1. Genom, und  $n$  die Zahl der Gene im 2. Genom

Damit der Speicherplatz linear bleibt muss explizit darauf geachtet werden, dass die Subprobleme während den Rekursionen nicht mehrmals abgearbeitet werden. Des weiteren müssen entsprechende Datenstrukturen gewählt werden, die die linearen Laufzeiten nicht zerstören.

Ebenfalls leicht ersichtlich ist die Tatsache, dass bei diesem Algorithmus keine großen versteckten Faktoren auftreten, die bei einer realen Anwendung die Laufzeit trotz Linearität unpraktikabel machen könnten. Trotz dieser tollen Laufzeit gibt es ein nicht zu verachtendes Manko: Der Algorithmus verarbeitet immer nur 2 Genome. Ein Multigenomvergleichen ist nicht vorgesehen. Dadurch basieren alle gefunden Cluster in erster Linie nur auf 2 Genomen. Ein gleichzeitiger Vergleich mit mehreren Genomen würde sicherlich die Zuverlässigkeit der Ergebnisse erhöhen, hätte allerdings auch zwei Nachteile: 1. Die Laufzeit würde sehr viel schlechter ausfallen, da man jedes Genom mit jedem vergleichen müsste und 2. müsste sichergestellt werden, dass ein gefundener Cluster bei einem Vergleich nicht gleich wieder aus

dem Gesamtergebnis gestrichen wird, nur weil er bei einem weiteren Vergleich mit einem anderen Genom nicht auftritt.

Möchte man aber dennoch mit dem oben angegebenen Algorithmus Cluster in mehr wie zwei Genomen finden, muss man noch eine weitere Routine einbauen, die den obigen Algorithmus mit jeder Möglichen Kombination aller Genome aufruft. Dadurch erhöht sich die Laufzeit auf „Anzahl der Kombinationen“ mal „Anzahl der Gene der jeweiligen Kombination“, was natürlich wesentlich mehr wie  $O(n+m)$  ist.

## Ergebnisse mit realen Daten:

### *Quantitative Analyse*

Michael Goldwasser und Xin He testeten ihren Algorithmus mit zwei realen prokaryotischen Genomen aus den Organismen E. Coli K12 und B. Subtilis. Diese beiden Genome wurden bereits sehr gut untersucht und die Cluster wurden schon mittels anderer Verfahren gefunden. Dadurch ist es möglich das Ergebnis mit schon vorliegenden Ermittlungen zu vergleichen.

In Schritt 1 wurden die orthologen Beziehungen gesucht. In einer international verfügbaren Datenbank des NCBI finden sich Eintragungen zur Homologie verschiedenster Genome. Die Datenbank enthält zur Zeit über 4800 COGs (identifiziert aus 43 kompletten Genomen). COG steht für „Cluster of orthologues Groups“ und ist in diesem Zusammenhang gleichbedeutend mit der Definition einer Homologie-Familie. Die Proteine in den heruntergeladenen Dateien enthalten schon eine so genannten COG-Nummer, also eine Zugehörigkeit zu einer Familie. Nachteil dieser Variante ist, dass hier nur auf die Familien der 43 verwendeten Genome zurückgegriffen werden kann. Will man andere Genome untersuchen, müssen zuerst die Familien der Gene aufgedeckt werden. Das soll hier aber nicht das Thema sein. Dafür gibt es andere Verfahren.

Wie viele Gene zu einer identifizierten Familie gehören zeigt Grafik 2. Werden alle Gene ignoriert, die zu keinem COG gehören bzw. nur in einem der beiden Genome vorkommen, entfallen bei diesem Test etwa die Hälfte

Grafik 2

Genome	Number of genes		
	Overall	From an identified COG	From an identified COG common to both genomes
<i>E. coli</i>	4290	3289 (76.7%)	2332 (54.4%)
<i>B. subtilis</i>	4101	2818 (68.7%)	2339 (57.0%)

aller Gene. Das reduziert die Laufzeit logischerweise auch deutlich. Anders herum werden allerdings auch nur Cluster mit Genen gefunden, deren Familie bekannt ist, was natürlich nicht unbedingt der Realität entspricht.

In Schritt 2 wurde der Algorithmus angewandt um alle  $\delta$ -Teams zu identifizieren.

Neben den beiden Genomen muss noch der  $\delta$ -Parameter an den Algorithmus übergeben werden. Dies klingt trivial – ist es aber nicht. Denn die Wahl des Parameters beeinflusst das Ergebnis ganz maßgeblich, wie Grafik 3 illustriert. Bei zu großer Wahl werden viele Falsch-Positive gefunden. Es werden also Gene zu einem

Cluster als zugehörig gewertet, die in Realität nicht dazugehören. Bei zu kleiner Wahl des Parameters sinkt die Zahl der gefundenen Cluster bzw. die Cluster werden unvollständig da Gene, die dazu gehören sollten nicht als zugehörig erkannt werden, weil die Lücke zwischen einem Gen und dem Rest vom Cluster zu groß ist. Aus biologischer Sicht bedeutet ein Cluster nicht zwangsläufig, dass alle relevanten Gene direkt aneinander gereiht sein müssen. Da das

Grafik 3

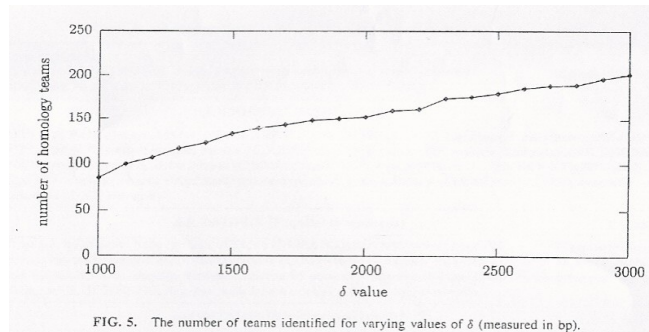


FIG. 5. The number of teams identified for varying values of  $\delta$  (measured in bp).

„Zusammenfinden“ der Gene zu einem Cluster nicht in einem Schritt passiert, sondern über unzählige Generationen hinweg und vor allem zufällig, ist es sogar sehr wahrscheinlich, dass zwischen zwei Genen, die für einen Cluster wichtig sind, andere Gene stehen.

Um den  $\delta$ -Parameter zu bestimmen, wurde ein Benchmark-Verfahren angewendet, der diesen Parameter schätzen soll. Dazu wurden vier bekannte Cluster gewählt:

- Ribosomalen Protein Cluster
- ATP Synthase Operon
- Tryptophan Biosynthese Operon
- ABC Ribose-Transport Operon

Alle vier Cluster sind sehr essentiell für die beiden Organismen. Das ATP Synthase Operon enthält beispielsweise sehr zentrale Gene zur Bildung von ATP, ohne das bei diesen (und vielen anderen) Organismen nichts laufen würde.

Nun wurde der Algorithmus für verschiedene  $\delta$ -Parameter gestartet und am Ende geschaut, bei welchem Wert die obigen vier Cluster bestmöglich wiederhergestellt wurden. Da vier sehr unterschiedliche Cluster (in Bezug auf die Funktion der Genprodukte) gewählt wurde, kann zumindest ansatzweise davon ausgegangen werden, dass eine gewisse Abdeckung der Bandbreite an Cluster gegeben ist. Ansonsten könnte nicht zwangsläufig davon ausgegangen werden, dass der ermittelte  $\delta$ -Parameter auch für alle anderen gefundenen Cluster gelten könnte. Das beste Benchmark-Ergebnis wurde erhalten, als  $\delta = 1900$  Basenpaare gewählt wurde. Das entspricht etwa 1-2 Genlängen.

Neben den vier oben identifizierten Clustern wurden noch 150 andere potentielle Cluster gefunden. Diese wurden mit bereits bekannten Clustern verglichen. Dabei wurden von vorne herein alle ribosomalen Cluster, sowie Cluster der Kardinalität zwei nicht betrachtet. Erstere, weil sie biologisch uninteressant sind und Zweitere, weil Cluster der Länge zwei ebenfalls keine besondere Bedeutung haben und auch nicht signifikant genug sind, wie ich später noch erläutern werde.

Die verbleibenden 43 Cluster wurden den folgenden vier Typen zugeordnet:

1. Exakte Übereinstimmung mit einem bekannten Operon (10)
2. Teilweise Übereinstimmung (20)
3. Solche, die vorhergesagte Operone treffen (11)
4. Keine Übereinstimmung mit Operonen (2)

Die Zahl in Klammer gibt die Anzahl der Cluster an, die dem jeweiligen Typ zugeordnet wurden.

### **Interpretation der Ergebnisse:**

#### *Bedeutung und Nutzen der ermittelten Cluster*

Die 10 exakten Übereinstimmungen enthalten sehr zentrale Cluster (z.B. ATP Synthase). Die Ergebnisse der 2. Gruppe sind allerdings viel interessanter. Hierzu gibt es diverse Interpretationsmöglichkeiten:

1. Möglichkeit: Fehlende oder falsche Zuordnungen von Genen zu COGs
2. Möglichkeit:  $\delta$ -Wert zu klein gewählt
3. Möglichkeit: Einige Operone sind nicht in beiden Organismen konserviert
4. Möglichkeit: Zugeordnete Gene könnten noch nicht als zugehörig (zum Operon) entdeckt worden sein

Punkt 4 birgt einiges an Potential für weitere Untersuchungen. Denn alle anderen Verfahren, die bereits angewendet wurde haben das gleiche Problem: Das Ergebnis kann nicht 100% als sicher gelten, schon allein deshalb, weil Cluster dem Zufall unterliegen. Darum kann es gut sein, dass vorhandene Methoden ein Gen einfach nicht als zugehörig klassifiziert haben. Ähnliches kann man zu Typ 4 sagen (keine Übereinstimmung): Es ist durchaus möglich, dass ein Verfahren ein komplettes Cluster nicht findet.

### **Eine Implementierung:**

#### *Umsetzung des Algorithmus durch M. Goldwasser und Xin He*

Michael Goldwasser und Xin He haben erfreulicherweise nicht nur ein Modell und einen Algorithmus zur Clusterfindung vorgestellt, sondern diesen auch gleich noch implementiert. Das Programm ist in C++ geschrieben, in diverse Klassen unterteilt und gut dokumentiert. Es kann frei <http://euler.slu.edu/~goldwasser/homologyteams/> heruntergeladen und nach belieben verändert werden [Stand Juli 2006].

Die beiden Autoren legen neben den Beispieldaten und Ergebnissen auch noch ein paar Tools bei, die die Rohdaten der NCBI-Datenbank in ein entsprechendes Format

umwandelt, welches als Eingabe dient. Verschiedene Ausgabeformate können ebenfalls gewählt werden, um z.B. alle Cluster der Kardinalität vier herauszufiltern.

### **Statistische Signifikanz der Ergebnisse:**

*Klärung der Frage, ob ein gefundener Cluster statistisch abgesichert ist*

Alle Ergebnisse sind nur so lange gut, wie sichergestellt werden kann, dass die Wahrscheinlichkeit, dass dieses Ergebnis eingetreten ist viel größer ist als die Wahrscheinlichkeit, dass dieses Ergebnis zufällig zu Stande gekommen ist. Im konkreten Fall wäre es ja auch denkbar, dass Cluster eine rein zufällige Sache sind und Gene in Wahrheit gar nicht clustern, sondern randomisiert über das ganze Genom verstreut liegen. Das würde bedeuten, dass das komplette Modell eines Clusters auf keiner Realität basieren würde. Als Vergleich aus dem Alltag kann man einen simplen Gefrierschrank heranziehen. Schauen Sie in einen fremden Gefrierschrank, kann ich nicht sicher sagen, ob die drei Packungen Spinat rein zufällig in der gleichen Schublade liegen (weil z.B. woanders kein Platz war) oder ob der Besitzer ganz gezielt gleiche / ähnliche Nahrungsmittel gruppiert.

Jeder würde aber sagen, dass eine gewisse Ordnung vorliegt, wenn er beispielsweise zehn Spinatpäckchen, fünf mal Erbsen und 3 mal irgendein anderes Gemüse in der gleichen Schublade finden würde und in einer anderen Schublade nur Fleisch. Während diese Betrachtungsweise aber von Person zu Person schwanken kann (wo ist die Grenze?), sollte jedes wissenschaftliche Ergebnis irgendwie statistisch abgesichert sein. Im vorliegenden Fall muss also die Null-Hypothese sehr unwahrscheinlich sein. Komplettausschluss kann man die Null-Hypothese allerdings nicht. Die zehn Spinatpäckchen könnten schließlich trotz aller Unwahrscheinlichkeit rein zufällig in der gleichen Schublade gelandet sein.

Zur Vereinfachung soll angenommen werden, dass die Zahl der Gene beider Genome gleich ist. Laut Aussage der Autoren soll dies kein Problem sein. Sei also:

$n$  = Anzahl der Gene im Genom / Chromosom

$k$  = Anzahl der „relevanten“ Gene, die einen Cluster bilden

$d$  = Anzahl der „irrelevanten“ Gene, die zwischen zwei „relevanten“ Genen stehen dürfen ( $d = \delta - 1$ )

Zu erst soll nur ein Teil des Modells betrachtet werden – alle 1-zu-1-Beziehungen. Später werden dann auch die beiden anderen Beziehungstypen mit eingeschlossen.

Hat ein Genom  $n$  Gene, gibt es  $n!$  Permutationen um diese  $n$  Gene anzuordnen. Will man nun ein Cluster der Größe  $k$  generieren ergeben sich wiederum  $k!$

Permutationen diese  $k$  Gene anzuordnen. Da  $k$  aber nur die Zahl der relevanten Gene in diesem Cluster angibt, müssen noch die Möglichkeiten  $i$  Lücken zu platzieren hinzugerechnet werden:

Anzahl „Lücken“ im Cluster =  $i$

$$0 \leq i \leq (k-1)*d$$

Beispiel:  $d = 2, k = 5$

$i = 0$  ergibt sich, wenn keine Lücken auftreten

<xxxxx> <-- x steht für ein relevantes Gen.

$i = 8$  ergibt sich, wenn zwischen 2 relevanten Genen immer genau  $d$  irrelevante Gene liegen

<x--x--x--x--x>

Dies sind auch gleichzeitig die beiden Extremfälle die eintreten dürfen. Nun kennt man die Anzahl Möglichkeiten einen  $k$ -Cluster zu generieren, genauso wie die Anzahl Möglichkeiten  $i$  Lücken zu platzieren. Hier ergibt sich leider das Problem, dass die Kombination von beidem zu keiner einfachen geschlossenen Formel führt. Deshalb sei an dieser Stelle die Anzahl Möglichkeiten einen  $k$ -Cluster mit  $i$  Lücken zu generieren gleich  $s(k,d,i)$ . Dazu später mehr.

Zum Schluss muss man noch berechnen wie viele Möglichkeiten es gibt den kompletten Cluster im Genom zu platzieren. Der gesamte Cluster hat nun eine Größe von  $k+i$ . Dies ergibt:

$n-(k+i)+1$  Möglichkeiten.

Nun werden aber nicht mehr  $n$  Gen um diesen Cluster herum verteilt, sondern nur noch  $n-k$ . Es sind nicht  $n-(k+i)$ , weil die  $i$  Lücken ebenfalls noch mit unterschiedlichen Kombinationen aus den restlichen Genen gefüllt werden können.  $i$  gibt nur die Zahl der Lücken an. Deshalb bleiben noch  $(n-k)!$  Möglichkeiten übrig um die restlichen Gene zu verteilen.

Jetzt können wir die einzelnen Bausteine zusammensetzen:

$$P(n,k,d) = \frac{(n-k)!k! \sum_{i=0}^{(k-1)d} s(k,d,i)*(n-k-i+1)}{n!}$$

Erklärung der einzelnen Terme von  $P(n,k,d)$ :

$n!$	Anzahl Permutationen bei $n$ Genen
$(n-k)!$	Restliche Gene plazieren
$k!$	Anzahl Permutationen um $k$ Gene zu plazieren
$s(k,d,i)$	Anzahl Möglichkeiten eine $k$ -Cluster mit $i$ Lücken zu generieren
$(n-k-i+1)$	Kompletten Cluster im Genom plazieren

Die Summe beginnt bei  $i = 0$ , da ein Cluster als Minimum keine Lücke enthalten kann und endet bei  $(k-1)*d$ , was die maximale Anzahl an Lücken darstellt.

Über dem Bruchstrich stehen die Anzahl Möglichkeiten ein Genom mit einem k-Cluster und i Lücken zu generieren, unten die Anzahl, wenn man n Gene rein zufällig verteilt.

Durch Umformen erhält man folgende Formel:

$$P(n,k,d) = \frac{\sum_{i=0}^{(k-1)d} s(k, d, i) * (n - k - i + 1)}{\binom{n}{k}}$$

Würden nur 1-zu-1-Beziehungen in Betracht gezogen, wäre die obige Formel jetzt vollständig. Diese machen in unserem Modell aber nur etwa 1/3 aller Beziehungen aus. Darum wird die Berechnung jetzt um einige weitere Möglichkeiten, einen Cluster zu bilden erweitert.

Sei:

m = Anzahl Familien im Genom

M = {f<sub>1</sub>, f<sub>2</sub>, ..., f<sub>m</sub>} = Die Menge aller Familien.

φ<sub>j</sub> = Anzahl Gene, die zu Familie f<sub>j</sub> gehören

i<sub>j</sub> = Index der Familie zu der das Gene j gehört

Der Unterschied zur vorherigen Betrachtung liegt in der Anzahl Möglichkeiten für ein Gen. Für das j-te Gen gibt es nun φ<sub>i<sub>j</sub></sub> Möglichkeiten aus der Familie i<sub>j</sub>.

Daraus ergibt sich jetzt eine neue Anzahl an Möglichkeiten k Gene auszuwählen, die einen Cluster bilden sollen:

$$\phi_{i_1} * \phi_{i_2} * \dots * \phi_{i_k}$$

Die Gesamtformel wird somit um einige weitere kombinatorische Möglichkeiten erweitert. Daraus ergibt sich:

$$Q(M,n,k,d) = P(n,k,d) * \prod_{j=1}^k \phi_{i_j}$$

Die Formel zeigt leider auf den ersten Blick nicht allzu viel. Eine verlässliche Aussage lässt sich nur durch Einschränken des Parameterraums erreichen. He und Goldwasser geben daher eine Approximation von Q(M,n,k,d) an, die auf real zu erwartenden Variablengrößen basiert.

Die erste Einschränkung bezieht sich auf die Clustergröße und somit auf die Variablen  $k$  und  $i$ . Ein natürlicher Cluster überschreitet so gut wie nie die Größe zehn. Das erscheint nicht unlogisch, wenn man sich überlegt, dass eine einzelne Transkriptionskontrolle ansonsten wohl viel zu grob wäre. Des Weiteren wäre eine sehr dichte Packung (Faltungen) nicht mehr möglich.

Als nächstes kann man den Raum für den Eingabeparameter  $\delta$  eingrenzen. Durch das weiter oben angegebene Benchmarkverfahren wurde schon gezeigt, dass  $\delta$  am besten zwischen eins und zwei gewählt werden sollte. Um nicht zu stark einzugrenzen, wählen die Autoren  $\delta$  zwischen eins und drei.

Letztlich bestimmt die Genomgröße noch maßgeblich die Anzahl an Kombinationen, da  $n$  für gewöhnlich mehrere Tausend Gene umfasst und somit wesentlich größer wie  $k$ ,  $d$ ,  $i$  ist. Darum kann man den Parameter  $i$  getrost vernachlässigen. Eine weitere Schwachstelle der Formel für  $Q(M,n,k,d)$  war die ungeschlossene Teilformel, die mit  $s(k,d,i)$  bezeichnet wurde. Dies lässt sich aber auflösen, indem man die Anzahl Kombination für eine Lücke der Größe eins, der Größe zwei, der Größe drei,..., der Größe  $d$ , berechnet. Bei  $d+1$  Möglichkeiten, Lücken zwischen zwei relevanten Genen zu verteilen, und falls der Cluster  $k$  relevante Gene besitzt ergibt sich somit für die 1. Gesamtfomel:

$$P(n,k,d) \approx \frac{(n - k - i + 1) * (d + 1)^{k-1}}{\binom{n}{k}}$$

Nun kann man in die daraus resultierende Formel  $Q(M,n,k,d)$  Beispielwerte aus dem Testorganismus E.Coli einsetzen:

$n = 2000$  (Achtung: nur orthologe Gene)

$\delta = 1$  (entspricht etwa 1900 Basenpaaren im Genom)

Der Produktterm soll aus der Formel  $Q(M,n,k,d)$  beträgt etwa  $2^k$ .

Da diese Werte alle fix sind, hängt das Ergebnis von  $k$  ab. Hier einige Ergebnisse für verschiedene  $k$ -Werte:

$$k=2 \Rightarrow Q(2) = 8 * 10^{-3}$$

$$k=3 \Rightarrow Q(3) = 4,8 * 10^{-5}$$

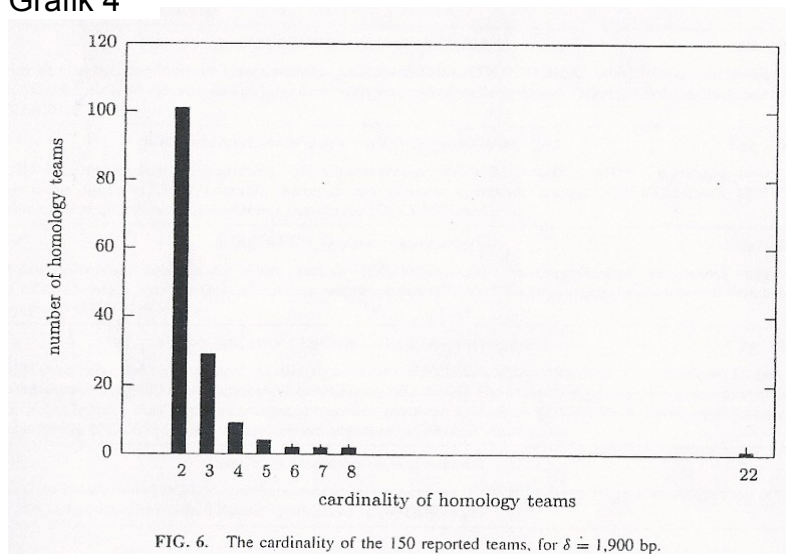
$$k=4 \Rightarrow Q(4) = 3,8 * 10^{-7}$$

Laut Goldwasser und He sind jetzt identifizierte Teams der Größe drei ( $k=3$ ) signifikant genug. Dies ist auch einer der Gründe, weshalb sie Cluster der Größe zwei aus ihrer Analyse weggelassen haben.

Allerdings wird auch gesagt, dass die tatsächlichen Wahrscheinlichkeiten deutlich abweichen können, da die Anzahl der Gene je Familie und die absolut Zahl in der Realität stark variieren.

Grafik 4 zeigt eine Übersicht über die Kardinalität der gefundenen Cluster.

Grafik 4



### **Fazit:**

He und Goldwasser haben ein sehr schnelles Verfahren angegeben, um Cluster über direkten Vergleich von Substrukturen, zu ermitteln. Der Algorithmus lässt sich leider nur bedingt auf Multigenomvergleiche erweitern, da die Laufzeit ansonsten sehr stark ansteigt. Dadurch können gefundene Cluster aber nur über weitere Einzelvergleiche oder schon vorhandene Einträge in Datenbanken besser verifiziert werden.

Die gefundenen Ergebnisse können sich allerdings schon bei der vorhandenen Variante sehen lassen und ermöglichen einige Interpretationsmöglichkeiten und weitere Forschungsansätze, die nicht nur auf den Schwächen des Algorithmus basieren. Dummerweise werden hauptsächlich Cluster der Kardinalität zwei gefunden (siehe Grafik 4), die zum einen statistisch nicht abgesichert sind und zum anderen aus biologischer Sicht kaum eine Bedeutung haben.

Größtes Manko besteht meiner Meinung nach in der mäßigen Analyse der Signifikanz. Es gibt keine vollständige allgemeingültige Formel, sondern nur eine Approximation, die allerdings auch noch durch diverse Einschränkungen in ihrer Aussage geschwächt wird. Positiv zu vermerken ist das Vorhandensein einer Implementierung des Algorithmus inklusive diverser Tools. Das erspart einem Anwender viel Zeit und ermöglicht statt dessen weitgehende Praxistests.

### **Themenverwandte Arbeiten:**

Beal et al. [2] verwendeten in ihrem Ansatz ein ähnliches Modell, welches allerdings nur 1-zu-1-Beziehungen einbezog.

Sankhoff und Trinh [3] richten ihre Arbeiten letztendlich gar nicht auf die Analyse und Ermittlung von Clustern aus, sondern erarbeiteten auf Basis vorangegangener Arbeiten von Pevzner, Tesler und Hannenhalli Erweiterungen zum Thema „Genome Sequenz Rearrangement“.

Wie beim Clustering wird davon ausgegangen, dass einzelne Gene / Segmente auf dem Genom „verschoben“ werden können. Im Allgemeinen spricht man hier von Translokationen. Sie treten wesentlich häufiger auf, wie Insertionen bzw. Deletionen. Mittels Rearrangement-Abfolgen sollte es also möglich sein ein Genom in ein anderes zu überführen. Die Anzahl der Rearrangements kann dabei als Maß für die phylogenetische Distanz gewählt werden, um beispielsweise Stammbäume aufzustellen.

Ein wichtiger Begriff ist hierbei der des „Breakpoints“. Ein Breakpoint tritt immer dann auf, wenn eine in einer Zahlenreihe auf das Element  $i$  nicht das Element  $i+1$  folgt (bzw.  $i-1$  bei negativer Reihe).

Beispiel Breakpoint:

123|654|789

Hier finden sich zwei Breakpoints, die durch eine „|“-Symbol gekennzeichnet werden.

Sankhoff und Trinh erarbeiteten nun, wie oft im Durchschnitt ein solcher Breakpoint bei der Eliminierung durch Rearrangements involviert sei. Kehrt man im obigen Beispiel den Bereich zwischen den beiden Breakpoints um, fallen beiden Breakpoints auf einmal weg. Aus theoretischen Überlegungen kann man ableiten, dass ein Rearrangement maximal zwei Breakpoints eliminieren kann.

Durch Berechnungen haben die Autoren herausgefunden, dass die Zahl der Reuses:

1. Abhängig von der Größe des Genoms ist
2. Abhängig von der Anzahl an Deletionen ist.

Je mehr Stellen im Genom deletiert werden, desto länger bleibt ein Breakpoint (im Durchschnitt) bestehen.

Warum ist das so? Weil die relativ geordnete Genomstruktur (des Ausgangsgenoms) „zusammenbricht“ und hauptsächlich 1-Inversionen angewendet werden können. 1-Inversionen sind solche, die nur genau einen Breakpoint entfernen. Dadurch entsteht automatisch eine Mehrfachbenutzung des nicht eliminierten Breakpoints.

Sankhoff und Trinh bildeten bei ihrer Analyse ein ganz ähnliches Modell, wie He und Goldwasser, allerdings mit einem ganz anderen Ziel.

Zu ihren Ergebnissen gelangten sie durch die Berechnung der kombinatorischen Möglichkeiten von Breakpointeliminierungen und deren Wahrscheinlichkeit.

In einem weiteren Paper [4] stellen Chunfang Zheng und David Sankhoff einen eigenen Algorithmus vor, der ein Genom in ein anderes überführt. Dabei modellieren sie zuerst jedes Genom als einen gerichteten azyklischen Graphen. Mittels Rearrangements und Reversals wird dann ein aus dem gerichteten azyklischen Graphen ein gerichteter Graph, der auch Zyklen beinhalten darf. Beide Graphen (für

zwei Genome) werden dann zu einem gemeinsamen Graphen vereint und anschließend wieder in zwei einzelne Graphen aufgespalten, die dann eine optimale Abfolge von Rearrangements enthalten. Die Autoren gehen dabei explizit in ihrem Modell davon aus, dass unterschiedliche, widersprüchliche und unvollständige Datensätze aus unterschiedlichen Datenbanken vorliegen können und bieten eine Methode an, um solche Konflikte zu lösen. Leider wird keine O-Notation angegeben, dafür aber Laufzeiten auf Referenzsystemen. Laut meiner Meinung könnte der Algorithmus mindestens teilweise auch parallelisiert werden, was die Transformation von einem kompletten großen Genom in ein anderes ermöglichen könnte. Als Ergebnis wird auch angegeben, welche Transformationen nötig sind. Diese können dann für weitere phylogenetische Betrachtungen (z.B. Stammbäume) genutzt werden.

### **Literaturhinweise:**

[1] Xin He, Michael H. Goldwasser, 2005, Identifying Conserved Gene Clusters in the Presence of Homology Families, Journal of computational biology, Volume 12 Number 6

[2] Beal , M.-P., Bergeron, A., Corteel, S., Raffinot, M., 2004, An algorithmic view of gene teams, Theoret. Comput. Sci., 320(2-3), 395-418

[3] David Sankhoff, Phil Trinh, 2005, Chromosomal Breakpoint Reuse in Genome Sequence Rearrangement, Journal of computational biology, Volume Number 6

[4] Chunfang Zheng, David Sankhoff, Genome Rearrangement with Partially Ordered Chromosomes

[5] <http://prion.bchs.uh.edu/~ashwin/evolution/bacteria.html>

[6] <http://gslc.genetics.utah.edu/units/basics/bodypatterns/discover.cfm>