Technische Fakultät
Center for Biotechnology (CeBiTec)
Computational Genomics

Comparative genomics on gene and single nucleotide level

Zur Erlangung des akademischen Grades eines Doktors der Naturwissenschaften an der Technischen Fakultät der Universität Bielefeld vorgelegte Dissertation

von

Jochen Blom

21. Januar 2013

Jochen Blom
Ravensberger Str. 36
33602 Bielefeld
jblom@cebitec.uni-bielefeld.de

Supervisors:	Prof. Dr. Jens Stoye
	Dr. Alexander Goesmann

Gedruckt auf alterungsbeständigem Papier (ISO 9706).

Widmung

Für meinen Vater

Paul Blom

†02.10.2012

Contents

CHAPTER 1

Introduction

DNA sequencing, the determination of the order of the nucleotide bases in a DNA molecule, is one of the most important and most widespread technologies in modern microbiology. It has various applications like full genome sequencing (*de novo* sequencing and re-sequencing), amplicon sequencing, transcriptome sequencing, and metagenomics. Despite the relevance of DNA sequencing for modern microbiology, with the chain-termination method firstly proposed by Sanger and Coulson (1975) the state-of-the-art sequencing method remained unchanged for nearly three decades. This changed in 2005 when *454 Life Sciences* released the Genome Sequencer 20 system, the first marketable "next generation" sequencing machine. Since the release of the GS20 a rapid development in sequencing techniques was set in motion that lowered the costs and raised the output of modern sequencing systems by several orders of magnitude over the last 7 years (see Figure 1.1).
The sequencing costs for 1 million base pairs (1 Megabase - Mb) was ∼2000$ for Sanger sequencing, with the newest Illumina system the costs decreased to ∼0.07$ per Mb. Thus, while it was a major scientific project to sequence a complete bacterial genome ten years ago, requiring a lot of time, money, and personnel (Tauch et al., 2002), nowadays whole genome sequencing is a routine task and became the standard starting point in genome analysis. This is reflected by the development of the Genomes On Line Database (GOLD) (Pagani et al., 2012). GOLD is a resource that collects data about ongoing and finished genome sequencing projects. It collects information about the sequenced organism, about technical details, links to other resources, and meta data like the source of an isolate or details about the host in case of a clinical isolate. Due to the massive drop in sequencing costs the number of (in particular bacterial) genome sequencing projects increased dramatically from below 1,000 in 2005 to nearly 10,000 at the end of 2011 (Figure 1.2).

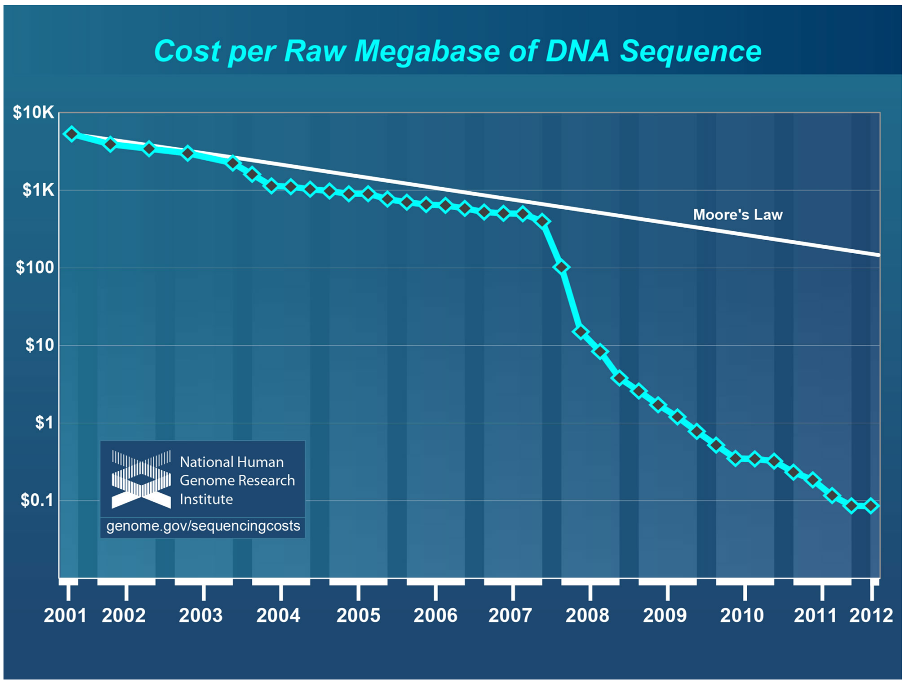


Figure 1.1.: Development of sequencing costs for one million base pairs since 2001 (Wetterstrand, 2012) compared to Moore's law. Moore's law describes the observation that the number of transistors on integrated circuits has a constant doubling time of approximately two years.

The described technical quantum leap in sequencing in recent years allows completely new questions to be asked. The massive decrease in sequencing costs made it feasible to sequence not just a single bacterial species, but several related strains. For example one could sequence a set of species from the same genus in phylogenetic studies, or different clinical isolates of a pathogenic bacterial strain in medical studies. Hence, the rapid development in sequencing technologies opened up a completely new branch of bioscience: comparative genomics.

1.1. Comparative genomics

The term "Comparative genomics" describes the analysis of the similarities and differences between the genome sequences and resultant features of related biological strains or species (Bachhawat, 2006). Comparative genomics explores genomes on different levels, ranging from the analysis of large scale genomic rearrangements to the comparison of genomic features like genes, coding sequences, RNAs, or regulatory regions and finally to small scale differences like single nucleotide polymorphisms. The huge amounts of data that come with the analysis of several complete genomes at once necessitate an automation of the comparative methods. Large scale rearrangements can be analyzed quite easily using genome alignment tools like Mauve (Darling et al., 2004), MUMmer (Kurtz et al., 2004), or MGA

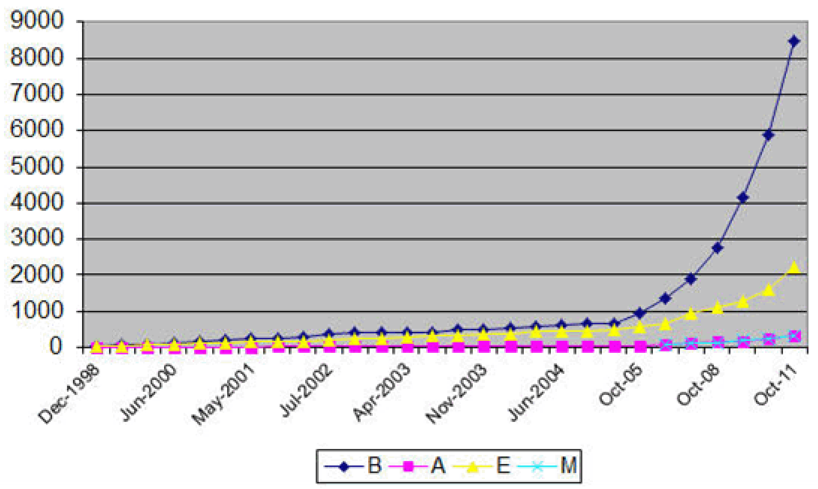


Figure 1.2.: Development of the number of sequencing projects in the Genomes On Line Database (GOLD). The figure shows that the number of registered sequencing projects increased linearly from 1998 until 2004. At the end of 2005 the number of projects starts to grow exponentially due to the introduction of next-generation sequencing. Labels: **B**acteria, **A**rchaea, **E**ukaryotes, **M**etagenome projects.

(Höhl et al., 2002). Multi-genome comparisons on feature level are realized mainly by the use of pairwise sequence alignment tools, the most prominent being the Basic Local Alignment Search Tool BLAST (Altschul et al., 1990). The main goal of comparative genomics on gene level is the linkage of the gene content of analyzed strains or groups with their phenotypical features. This may be particularly interesting in the comparison of pathogenic with non-pathogenic bacteria. For this type of comparisons several genomic subsets may be of interest, e.g, the core genome, the set of genes shared by all strains, or the singleton genes, genes that are found only in one of the strains of a comparison set. To support gene content analyses, the first software presented in this work was developed: EDGAR ("**E**fficient **D**atabase framework for comparative **G**enome **A**nalyses using BLAST score **R**atios").

For very closely related strains of the same species the comparison on gene level can be still to coarse grained and a higher resolution approach is needed. A comparison of single nucleotide polymorphisms (SNPs) is the most detailed analysis level that can be achieved in comparative genomics. This level of detail may be needed in, e.g., population studies, the analysis of clinical isolates or outbreak samples. One important step in the analysis of single nucleotide differences between several strains is the mapping of sequence information from the analyzed strain to a template strain. Naturally, the accuracy of the sequence mapping is crucial for all following analysis steps. The mapping approaches for short DNA fragments coming from current sequencing systems prior to this work were either fast enough to handle huge amounts of data, but heuristic and thus potentially inaccurate, or they were accurate but very time consuming. Therefore the second software presented in this work is a novel read mapping software: SARUMAN (**S**emiglobal **A**lignment of short **R**eads **U**sing CUDA and Needle**MAN**-Wunsch).

1.2. Overview of the thesis

Following this introduction the basic aspects of comparative genomics in times of next generation sequencing will be presented in Chapter 2. It starts with a historical overview and technical details of the most important past and all commercially relevant current sequencing systems. Furthermore, the most fundamental sequence processing steps in the progress from raw sequencing output to a complete genome will be explained. In the third part of Chapter 2 a historical background of the field of comparative genomics will be given and specific terminology will be introduced.
In Chapter 3 the development of EDGAR and SARUMAN will be motivated. The general requirements for software in the field of comparative genomics will be analyzed and based on this requirements the goals of this work will be defined.
Chapter 4 is the first of two main chapters and covers comparative genomics on gene level. First, existing software approaches for comparative gene content analyses will be presented and compared. Subsequently the software design and the resulting implementation of the software EDGAR will be introduced. The usefulness of EDGAR is demonstrated by successful applications of the software on several different biological data sets. The chapter closes with a discussion of the results of this first part of the work.
Chapter 5 as the second main chapter is about comparative genomics on single nucleotide level. Like Chapter 4 it starts with an introduction and comparison of existing approaches in the field which are all either based on spaced seeds or use the Burrows-Wheeler transformation. Thus, these two techniques will be explained in detail, furthermore a historical overview of parallelization in sequence analyses will be given. Subsequently, the design and implementation of the software SARUMAN will be presented, including a formal definition of the short read matching problem and the algorithmic techniques used in SARUMAN to solve this problem. The performance of SARUMAN will be elaborately evaluated in comparison to existing approaches. Finally, examples of successful applications of the SARUMAN short read aligner will be presented.
In the Chapter 6 the overall results of this work will be summarized and discussed, furthermore an outlook on the future of the field of comparative genomics will be provided.

CHAPTER 2

Comparative genome analyses in times of ultrafast sequencing

As introduced in the previous chapter, DNA sequencing is one of the core techniques in modern microbiology. Especially whole genome sequencing, the sequencing of complete genomes of target organisms, became more and more popular in the last years due to the decreasing sequencing costs. In this chapter an overview of the development of sequencing technologies will be given, from the Sanger technique that dominated the field for the last decades to recent ultra-high throughput systems. The technical background of this so-called "next-generation" sequencing (NGS) will be presented, followed by a basic introduction into genome assembly. Furthermore, the emerging trend towards draft genomes, unfinished genomes where gaps in the sequence are tolerated, will be discussed.

2.1. First generation sequencing: Sanger sequencing

The first attempts towards DNA sequencing have been made in the mid seventies with the methods of Maxam and Gilbert (1977) and Sanger and Coulson (1975). While the Maxam-Gilbert method uses chemical treatment to generate breaks between single or di-nucleotides, the Sanger method uses dideoxynucleotides to terminate the replication of a single stranded DNA (ssDNA). Although published two years later, the Maxam-Gilbert method was more popular than the Sanger method in the first few years as it did not require any cloning steps. But due to its complexity and use of hazardous chemicals, soon the Sanger method became the method of choice and remained to be the standard approach for DNA sequencing for nearly three decades. Frederick Sanger and Walter Gilbert were awarded the 1980 Nobel Prize in Chemistry for their groundbreaking work on DNA sequencing.

2.1.1. Sanger sequencing: Chain termination by dideoxyribonucleotides

The chain termination sequencing method was developed mainly by Frederick Sanger in 1975 (Sanger and Coulson, 1975) and is therefore often called Sanger sequencing. In its classical form, the method is based on the replication of a ssDNA template and relies on two basic principles:

- Using a DNA polymerase, DNA primers, and the basic components of DNA, deoxynucleotidetriphosphates (dNTPs), a DNA strand reverse complementary to the a ssDNA template can be synthesized.
- Dideoxynucleotidetriphospates (ddNTPs) are modified dNTPs that lack the 3'-hydroxyl(-OH) group on their deoxyribose sugar that is required for the phosphodiester bond that connects the sugars and phosphates of the DNA backbone. When these ddNTPs are added to a DNA replication assay, the replication is terminated whenever a ddNTP is incorporated instead of a dNTP.

These two principles are used in Sanger sequencing by creating four identical DNA replication assays for one template DNA. To each reaction one of the four possible ddNTPs is added, which results in chain terminations and DNA fragments of varying size. The fragments of the four assays are separated by size in a denaturing polyacrylamide-urea gel with one particular lane for each reaction. The DNA fragments can be observed as bands visualized by UV light or autoradiography showing the relative positions of chain termination events in the four assays. From the progression of these bands the DNA sequence can be read (see Figure 2.1). Using this technique, Frederick Sanger and coworkers were able to estimate the 5,386 bp sequence of bacteriophage φX174, generating the first completely sequenced genome (Sanger et al., 1977).

2.1.2. Capillary electrophoresis

An enhancement to the classical Sanger sequencing is the capillary electrophoresis, in which only one reaction is needed instead of four. This is achieved by labeling the four ddNTPs with different dyes, each of which emit light of a distinct wavelength. Instead of separating the fragments in a gel, capillary electrophoresis is used to get the sequence of the DNA. Initiated by an electric field between a source and a destination vial, the negatively charged DNA migrates through a thin capillary. During this process the fragments are separated by size due to their electrophoretic mobility. At the outlet of the capillary the fluorescence of the labeled ddNTPs is detected. The result is usually displayed as an electropherogram, a plot of the

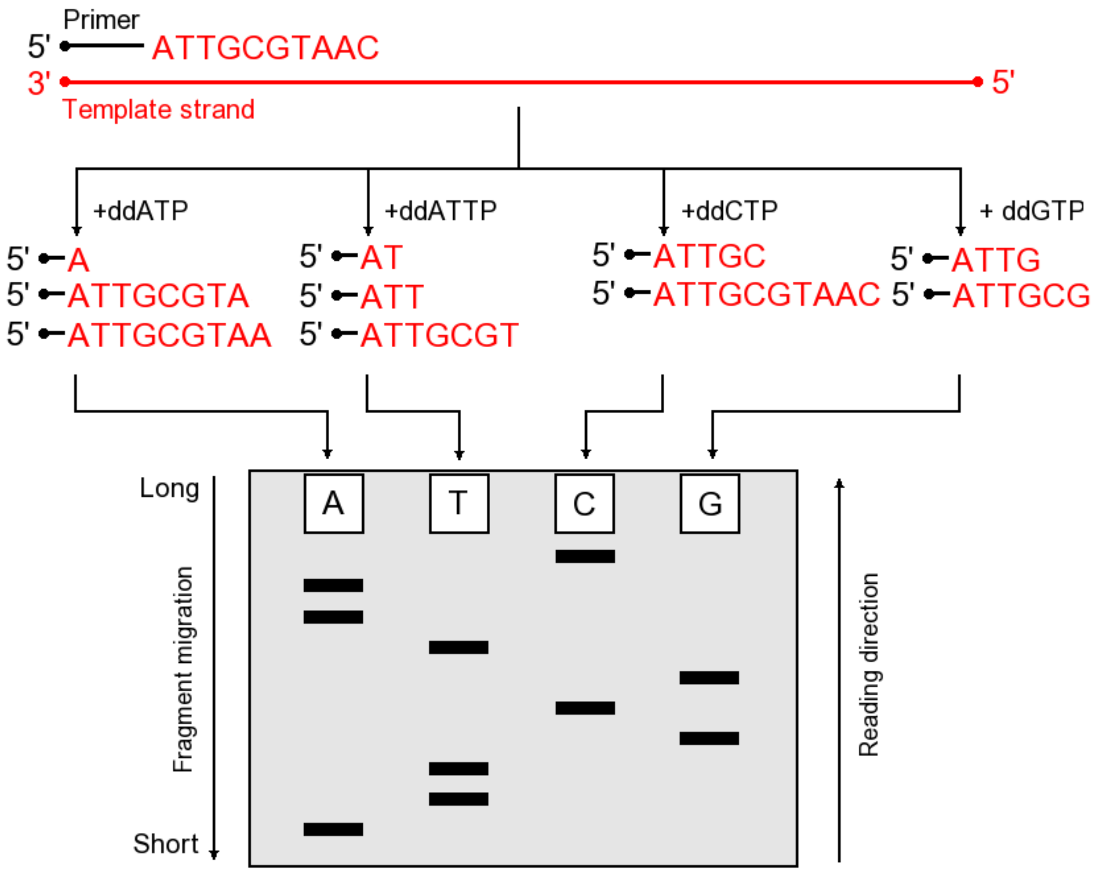


Figure 2.1.: Schematic overview of the classical gel-based Sanger sequencing. Four reactions with four different ddNTPs are prepared. The ddNTPs cause chain terminations after the corresponding bases, resulting in fragments of different size. These fragments are separated by size in a polyacrylamide gel, where smaller fragments migrate further than larger ones. By visualizing the bands in the gel and reading them against the electrophoresis direction the DNA sequence can be obtained.

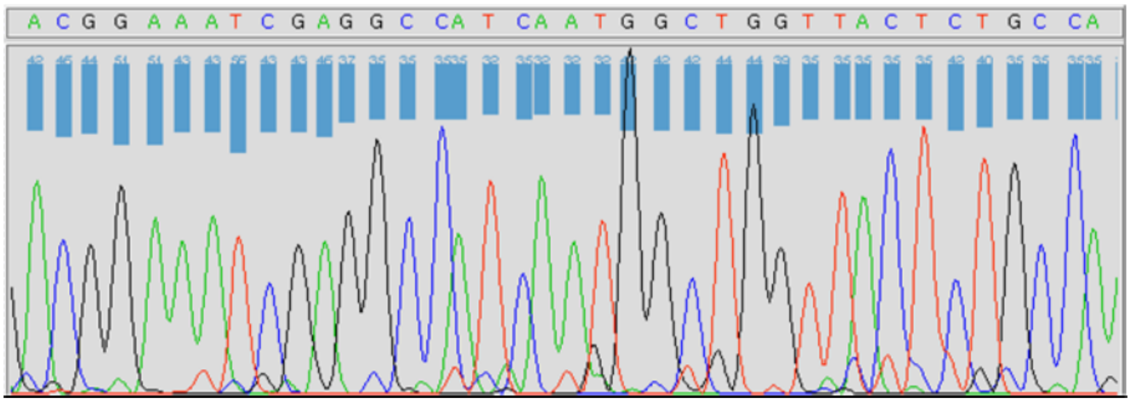


Figure 2.2.: Electropherogram from a capillary electrophoresis. Fluorescence of four different wavelengths is detected, each wavelength is associated with one of the bases. A fluorescence plot is displayed in the lower part of the image, the colors are assigned as follows: Green = A, Blue = C, Black = G, Red = T. In the upper part the resulting detected bases are listed.

detected fluorescence of different wavelength as a function of time (see Figure 2.2), from which the sequence can be deduced. As the capillary electrophoresis is faster and requires less manual work, it replaced the gel electrophoresis as far as possible.

Recent capillary sequencing systems like the ABI3730xl are able to achieve a total of 600,000 bp within one 2 hour run with reads of up to 1200 bp. The capillary electrophoresis was used for all genome sequencing projects up to 2005, even the human genome with its approximately three billion base pairs was sequenced using mainly this technique (Lander et al., 2001).

2.2. Second generation sequencing

Various attempts have been made to improve the output and to reduce the costs of DNA sequencing. In 1987 Pål Nyrén described how the enzymes ATP-sulfurylase and firefly luciferase could be used to monitor the activity of DNA polymerase (Nyrén, 1987). Ronaghi et al. (1996) took up this principle in 1996 and proposed a sequencing method based on the sulfurylase-luciferase reaction. As the polymerase reaction is constantly monitored in this approach, it was called real-time sequencing. This approach was the technical basis for pyrosequencing as marketed by *454 Life Science*[1]. The first marketable machine from 454 Life Sciences, the Genome Sequencer 20 (GS20) released in 2005, was able to produce up to 20 Mb of sequencing output in a one-day run, with read lengths of up to 100bp (Margulies et al., 2005). This system was the first one referred to as "Second generation sequencer" and lowered the costs for large sequencing projects so drastically that whole genome sequencing became a standard approach. Several other systems based on different

[1] 454 Life Sciences, a Roche company, Branford, CT, USA

new sequencing technologies have been released since then: The follow-up systems by *454 Life Sciences*, "GS FLX" and "GS FLX Titanium", the *Illumina*[2] systems "GAIIx" and "HighSeq2000", the systems "SOLiD" and "Ion Torrent" by *Life Technologies*[3], the "HeliScope" by *Helicos*[4], and the "PacBio RS" by *Pacific Biosciences*[5]. A further trend are so-called "Benchtop" sequencers, systems using established techniques on smaller scale. Most noticable are the "GS Junior", based in *454*'s pyrosequencing and the "MiSeq" as smaller version of the *Illumina* systems. All these techniques will be described in the following sections.
454's pyrosequencing and *Illumina*'s "sequencing by synthesis" are established for years now at the Bielefeld University's Center for Biotechnology (CeBiTec), recently also an "Ion Torrent" system was installed. As the majority of results presented in this work was obtained using data from the 454 or the Illumina system, the focus of the following chapter will be on this two techniques.

2.2.1. Roche 454: Pyrosequencing

2.2.1.1. Technique

The pyrosequencing technique by 454 Life Sciences is based on a massive parallelization of the real-time sequencing by Ronaghi et al. (1996). In a first step template DNA is broken down into fragments of a desired size (100bp for the GS20 system, 200bp, 400bp, or 800bp for different chemistry versions of the GS FLX system). This fragmentation is done physically by nebulization. Specific adapters are ligated to the DNA fragments which facilitate the binding to primer coated DNA capture beads (see Figure 2.3 A + B). Single beads bearing (ideally) a single DNA fragment each are encapsulated together with amplification reagents in droplets of a water-in-oil-mixture, forming microreactors for the following emulsion polymerase chain reaction (emPCR) amplification step. After this amplification step each bead is covered with millions of clonal copies of the template DNA fragment (Figure 2.3 C). The DNA-coated beads are loaded onto a so-called PicoTiterPlate (PTP), a plate with extremely small reaction wells. As the beads are $\sim 20\mu m$ and the wells on the PTP are $\sim 29\mu m$ in diameter on average, it is guaranteed that exactly one bead is loaded into one reaction well (Figure 2.4 A). The beads on the PTP are layered with smaller enzyme beads containing luciferase and sulfurylase, and also DNA polymerase is added. The loaded PTP is placed in the sequencing instrument, and sequencing reagents containing buffers and one of the four nucleotides are flowed across the plate in a fixed order of cycles. The millions of beads on the PTP are sequenced in parallel, each time a polymerase incorporates one or more nucleotides complementary to the template strain, a chemiluminescent light signal is created by the enzyme complex (Figure 2.4 B) and recorded by a CCD (Charge-

[2]Illumina, Inc., San Diego, CA, USA
[3]Life Technologies, Grand Island, NY, USA
[4]Helicos BioSciences Corporation, Cambridge, MA, USA
[5]Pacific Biosciences, Menlo Park, CA, USA

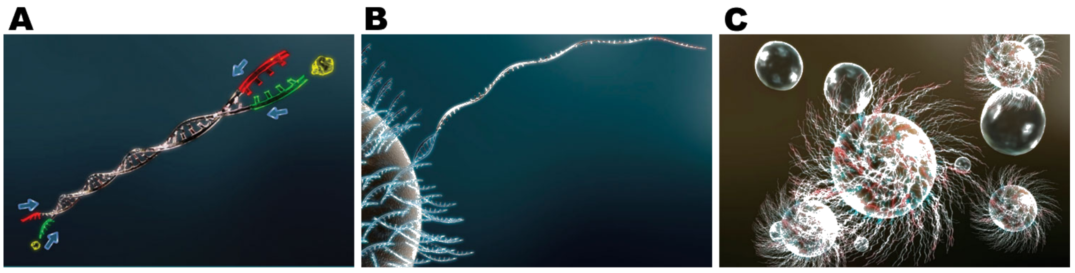


Figure 2.3.: Overview of the template preparation for the 454 pyrosequencing technique. (A) The ends of the fragmented template DNA are ligated with specific sequencing adapters. (B) The template DNA fragments are bound to DNA capture beads using these adapters. (C) After an emulsion PCR amplification step the capture bead is covered with clonal copies of the initial DNA fragment. Source: `www.454.com`

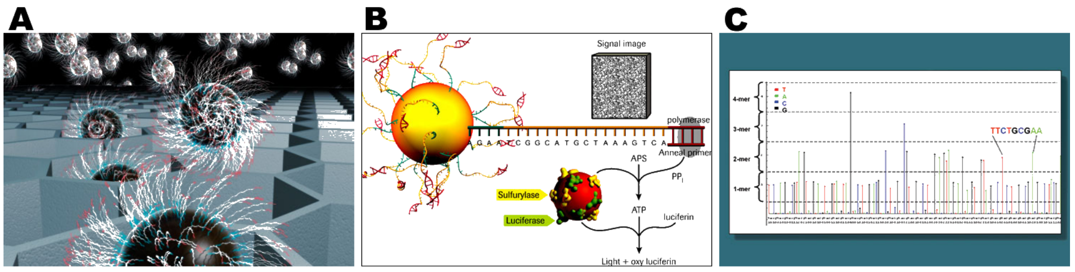


Figure 2.4.: Overview of the sequencing steps for the 454 pyrosequencing technique. (A) The DNA-coated beads are loaded on a PicoTiterPlate, one bead in each well. (B) On the incorporation of one or more nucleotides the sulfurylase-luciferase-reaction creates a light signal that is monitored by a CCD camera. (C) Flowgram resulting from the sequencing process. The diagram shows light signal intensity over time. The signal strength in each cycle is proportional to the number of nucleotides. Source: `www.454.com`

Coupled Device) camera for every well of the PTP, giving one sequencing read per bead. The signal strength is proportional to the number of nucleotides that were incorporated within one cycle (Figure 2.4 C).

2.2.1.2. Characteristics and impact

Pyrosequencing was the first NGS technique to hit the market and thereby was a groundbreaking step forward in whole genome sequencing. When it was introduced in 2005, it brought a dramatic decrease of sequencing costs (see Table 2.1) and thereby made whole genome sequencing an affordable and widespread approach in molecular biology. The main advantage of the 454 technique is that it has the highest read length of all NGS techniques. With reads of 800-1000bp after the FLX+ chemistry update in 2012 pyrosequencing reaches nearly the read length of

Sanger sequencing, which is ideal for *de novo* sequencing. The main drawback is the low throughput compared to other NGS techniques, especially to Illumina and SOLiD, and the higher costs per Megabase (Mb). Although relatively expensive compared to up-to-date short read sequencing techniques, the 454 technique is still superior in the resolution of long repetitive elements. The main drawback of the technique is the tendency to produce erroneous basecalls in longer homopolymer stretches. As described in Section 2.2.1.1, pyrosequencing incorporates more than one nucleotide in one cycle if several consecutive identical nucleotides (homopolymers) are found in the template sequence. A light signal proportional in strength to the number of incorporated nucleotides is emitted and detected by the sequencing system. For longer homopolymer stretches (>8bp), the signal strength is not rising linearly with the number of incorporated bases, making the number of incorporated bases hard to disambiguate (Margulies et al., 2005). This leads to overestimated or - more frequently - underestimated homopolymer lengths and thus to insertions or deletions in reads compared to the original template sequence. Another drawback of pyrosequencing was the inability to sequence organisms with a very high proportion of the bases “G” and “C” compared to the bases “A” and “T”, in the following called “GC content”. As G-C-pairs are bound by three hydrogen bonds these formations are more stable than A-T-pairs with only two hydrogen bonds and the melting temperature is higher. This leads to the formation of very stable secondary structures during the emPCR step in reads with exceedingly high GC content (Schwientek et al., 2011). This secondary structures lead to an insufficient amplification and as a result to an underrepresentation of the respective DNA fragment in the sequencing output. This problem was overcome by an additive to the sequencing chemistry which was identified by Schwientek et al. (2011) to consist of trehalose, which inhibits the self-annealing of the DNA during the emPCR. In summary, this technology provides intermediate read length and price per base compared to Sanger sequencing on one end and short read techniques on the other.

2.2.2. Illumina Solexa: Sequencing by synthesis

2.2.2.1. Technique

The second commercially marketed NGS technique was “sequencing by synthesis”, first marketed by the company Solexa before it was acquired by Illumina. Likewise to pyrosequencing, the sample preparation starts with random fragmentation of the template DNA into fragments of desired size (Figure 2.5, 1.). Specific adapters are ligated to these fragments at both ends, and the single strands of the fragments are bound to an oligo-coated glass surface, the so-called flow cell, at random positions (Figure 2.5, 2.). The adapters are designed to allow free ends of ligated fragments to form bridges to complementary oligos on the flow-cell surface (Figure 2.5, 3.). In the so-called “bridge amplification” step, the complementary strand to the bridged DNA is generated (Figure 2.5, 4.) and then denatured to two single strands (Figure 2.5, 5.), which can form new bridges and be templates for the next doubling step.

By iteratively repeating this procedure, one obtains dense clusters of both strands of identical DNA fragments that are located in close proximity on the glass surface (Figure 2.5, 6.).

After the bridge amplification, the actual sequencing is initiated by adding sequencing primers complementary to the adapters added in the second step, DNA polymerase and special nucleotides, so-called reversible terminator bases (RT-bases) (Figure 2.6, 7.). These RT-bases, $3'$-O-azidomethyl-$2'$-deoxynucleoside triphosphates, terminate the elongation of a DNA strand by the polymerase (Bentley et al., 2008) and are labeled with a different removable dye for each of the four possible bases. RT-bases ensure a stepwise sequencing process. After the first step of the nucleotide incorporation, the fluorophores are excited by a laser and an image is taken. A light signal is generated for each cluster on the flow-cell, and the wavelength of the light signal identifies the incorporated base (Figure 2.6, 8.). Prior to the next sequencing cycle, the fluorescent dyes are removed from the nucleotide, and the $3'$ hydroxyl group is regenerated to allow the next DNA synthesis step. In the next sequencing cycle, the RT-bases are added again (Figure 2.6, 9.) and a new picture is taken with light signals for all clusters (Figure 2.6, 10.). By repeating this cycle several times, the DNA sequence of the fragments is uncovered base-by-base with each cycle (Figure 2.6, 11.). As the reliability of called bases deteriorates with later cycles, the technique was limited to very short reads of 25bp to 35bp at the beginning. By improving the sequencing chemistry and software, Illumina is now capable to sequence reads of up to 150bp on the GAIIx. The benchtop system *MiSeq* is capable to provide 250bp reads with the latest chemistry, although at lower output than the major system.

2.2.2.2. Characteristics and impact

After 454 pyrosequencing opened the market for NGS systems, Illumina/Solexa's sequencing by synthesis was the first established short-read-high-throughput technique. The main advantage of this technique was its huge output of raw sequence data at comparably low costs. The first sequencer released by Illumina in 2007, the "Genome Analyzer", was able to generate 1-2Gb of sequence information within a single 3 days run. The current high-end system by Illumina, the HighSeq2500, can generate 120Gb per day or nearly 1Tb per run. This has dropped the costs for sequencing even large genomes like the human genome below 10,000$, bringing the 1,000$-genome within reach (see Figure 2.7). But the massive output of up-to-date Illumina systems naturally requires equally powerful data storage capacities, making the output of this systems a double-edged feature.

The main drawback of the technique was the initially short read length, but as described in the previous Section 2.2.2.1, the read length was considerably increased in the last years. With 150-250bp reads the technique is now suitable for *de novo* sequencing, and due to the enormous throughput it became the state-of-the-art technique for large sequencing projects. On the one hand, the read length is still inferior to pyrosequencing or Sanger sequencing, and the resolution of repetitive

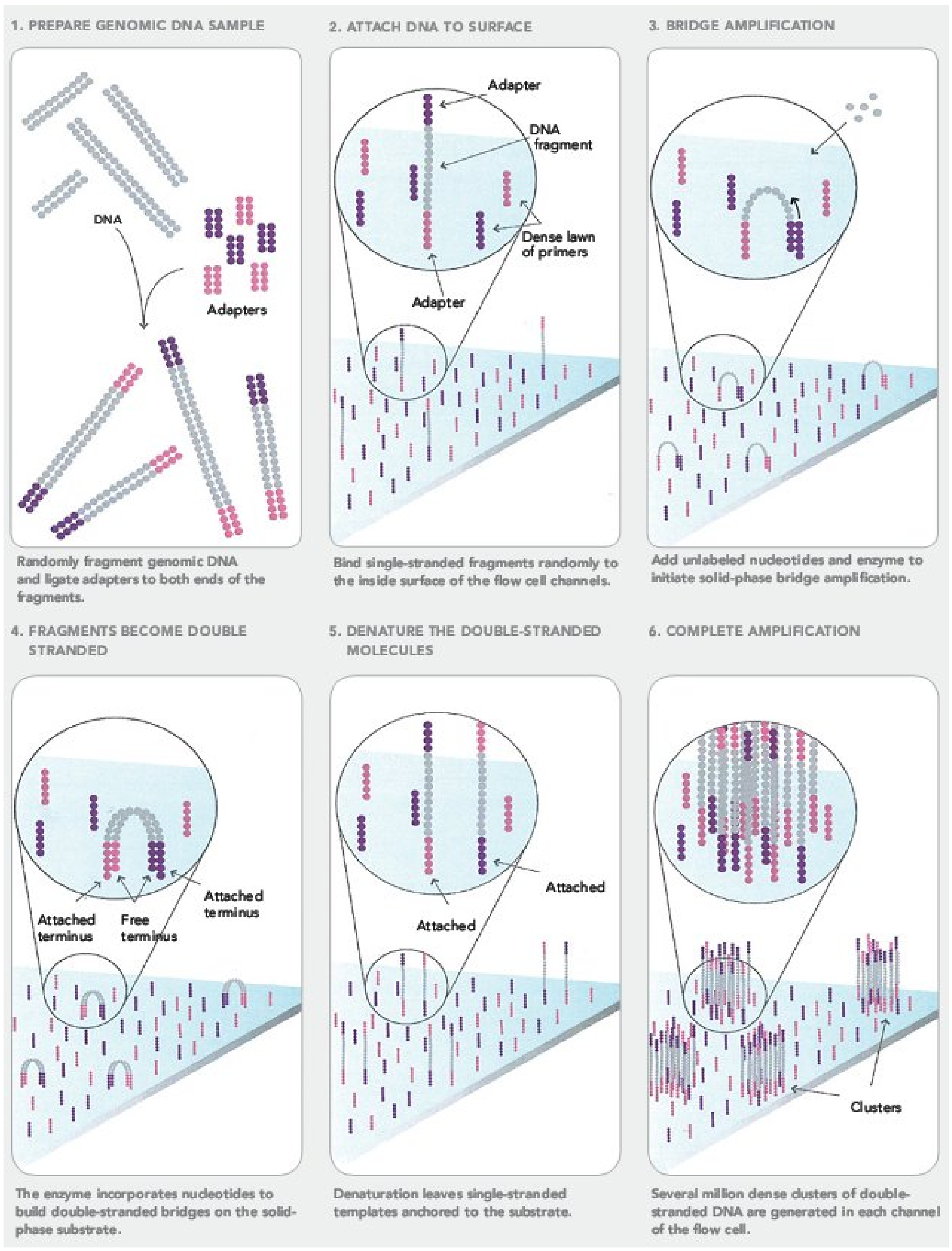

1. PREPARE GENOMIC DNA SAMPLE
DNA
Adapters
Randomly fragment genomic DNA and ligate adapters to both ends of the fragments.
2. ATTACH DNA TO SURFACE
Adapter
DNA fragment
Dense lawn of primers
Adapter
Bind single-stranded fragments randomly to the inside surface of the flow cell channels.
3. BRIDGE AMPLIFICATION
Add unlabeled nucleotides and enzyme to initiate solid-phase bridge amplification.
4. FRAGMENTS BECOME DOUBLE STRANDED
Attached terminus
Free terminus
Attached terminus
The enzyme incorporates nucleotides to build double-stranded bridges on the solid-phase substrate.
5. DENATURE THE DOUBLE-STRANDED MOLECULES
Attached
Attached
Denaturation leaves single-stranded templates anchored to the substrate.
6. COMPLETE AMPLIFICATION
Clusters
Several million dense clusters of double-stranded DNA are generated in each channel of the flow cell.

Figure 2.5.: Overview of the sample preparation and bridge amplification for the Illumina sequencing by synthesis technique. Source: `www.seqanswers.com`.

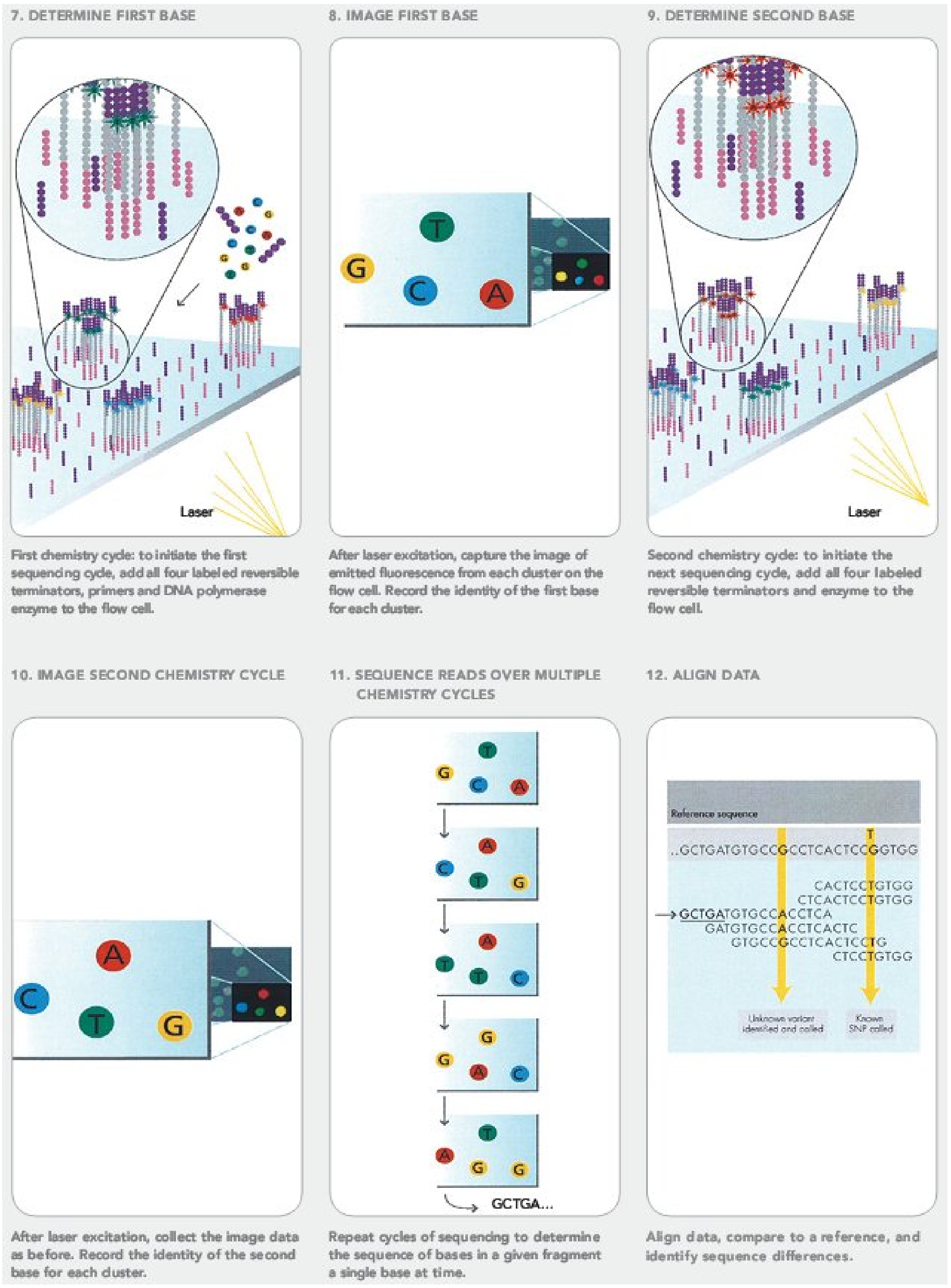

7. DETERMINE FIRST BASE
Laser
First chemistry cycle: to initiate the first sequencing cycle, add all four labeled reversible terminators, primers and DNA polymerase enzyme to the flow cell.
8. IMAGE FIRST BASE
T
G
C
A
After laser excitation, capture the image of emitted fluorescence from each cluster on the flow cell. Record the identity of the first base for each cluster.
9. DETERMINE SECOND BASE
Laser
Second chemistry cycle: to initiate the next sequencing cycle, add all four labeled reversible terminators and enzyme to the flow cell.
10. IMAGE SECOND CHEMISTRY CYCLE
A
C
T
G
After laser excitation, collect the image data as before. Record the identity of the second base for each cluster.
11. SEQUENCE READS OVER MULTIPLE CHEMISTRY CYCLES
GCTGA...
Repeat cycles of sequencing to determine the sequence of bases in a given fragment a single base at time.
12. ALIGN DATA
Reference sequence
T
..GCTGATGTGCCGCCTCACTCCGGTGG
CACTCCTGTGG
CTCACTCCTGTGG
GCTGATGTGCCACCTCA
GATGTGCCACCTCACTC
GTGCCGCCTCACTCCTG
CTCCTGTGG
Unknown variant identified and called
Known SNP called
Align data, compare to a reference, and identify sequence differences.

Figure 2.6.: Overview of the sequencing steps for the Illumina sequencing by synthesis technique. Figure taken from `www.seqanswers.com`

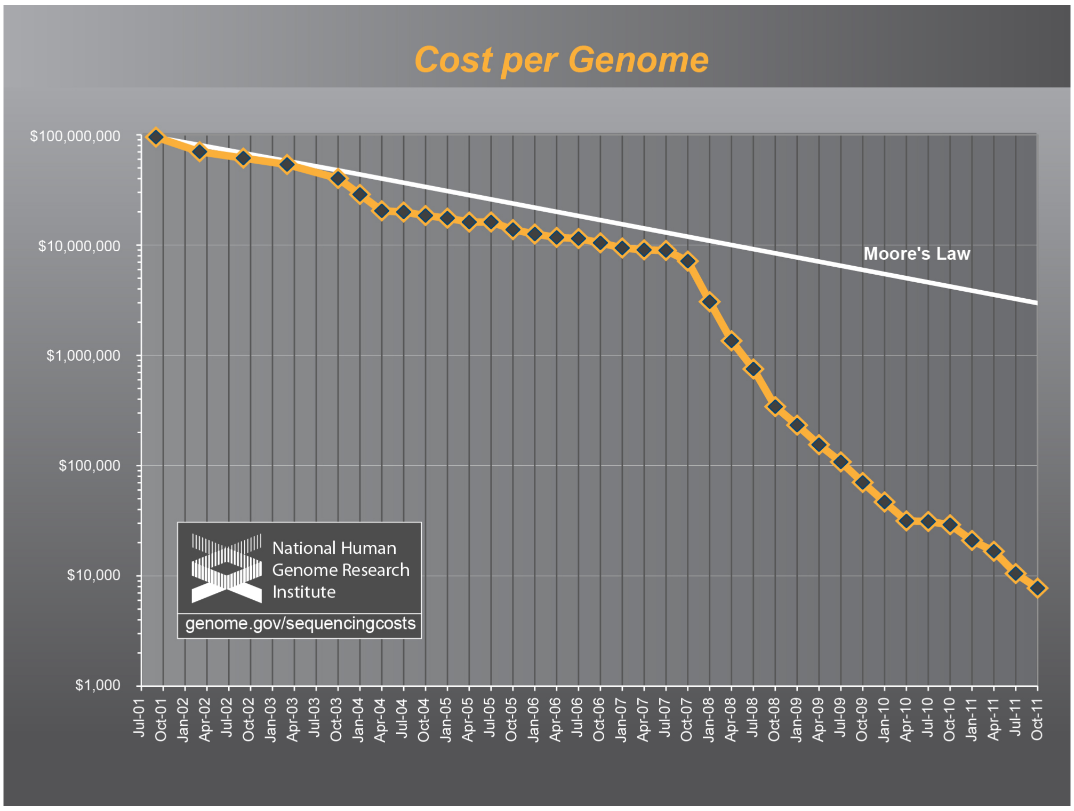


Figure 2.7.: Development of sequencing costs for a human-sized genome since 2001 (Wetterstrand, 2012).

regions is much harder with Illumina's shorter reads. On the other hand the technique is extremely versatile due to the huge number of sequenced reads. While pyrosequencing provides only one million reads per run, an Illumina GAIIx can provide 320 million reads per run (see Table 2.1). This is ideal for techniques like transcriptome sequencing (Wang et al., 2009), TAG sequencing (Porter et al., 2006), or ChIP (chromatin immunoprecipitation) sequencing (Johnson et al., 2007), where the high number of reads and the resulting high resolution is more important than the read length.

2.2.3. Other techniques

2.2.3.1. Helicos

The HeliScope system by Helicos BioSciences was the first commercial system that allowed single molecule sequencing, i.e., the sequencing of a single DNA strand without prior amplification. The HeliScope uses a sequencing by synthesis approach starting with the melting and fragmentation of the template DNA and the ligation of a poly-A-adapter to the single stranded fragments. These fragments are loaded on a proprietary flow cell, and fluorescence-labeled nucleotides are added in

Table 2.1.: Sequencing systems: Overview of the characteristics of historical and up-to-date sequencing systems. Costs do always refer to reagent costs, only. Data taken from (Glenn, 2011) and updated as needed.

Instrument	Run time	reads/run (m)	bases/read	yield(Mb)/run	cost/run	costs/Mb
3730xl capillary	2h	0.000096	650	0.06	96$	1.500$
454 GS Jr. Titanium	10h	0.10	400	50	1.100$	22$
454 FLX Titanium	10 h	1	400	500	6.200$	12.4$
454 FLX+	18-20h	1	800	900	6.200$	7$
Illumina MiSeq	26h	3.4	250	1.020	750$	0.74$
Illumina GAIIx	14d	320	150	96.000	11.524$	0.12$
Illumina HiSeq 2000	8d	1.000	100	200.000	20.120$	0.10$
Illumina HiSeq 2500	10d	$\leq$3.000	100	600.000	23 470$	0.04$
SOLiD - 4	12d	>840	50	71.400	8.128$	<0.11$
SOLiD - 5500	8d	>700	75	77.000	6.101$	<0.08$
SOLiD - 5500xl	8 days	>1.410	75	155.100	10.503$	<0.07$
Helicosf	N/A	800	35	28 000	N/A	N/A
Ion Torrent - 314 chip	2h	0.10	100	>10	500$	<50$
Ion Torrent - 316 chip	2h	1	>100	>100	750$	<7.5$
Ion Torrent - 318 chip	2h	4-8	>100	>1.000	925$	0.93$
PacBio RS	0.5-2h	0.01	860-1.100	5-10	110-900$	11-180$

cycles one at a time. Nucleotides matching the template strand are incorporated by a DNA polymerase, unused nucleotides are washed away. The fluorescence-labeled nucleotides can be detected by the HeliScope system through illumination with a laser. After the imaging step the fluorophores are removed and the next sequencing cycle with the next nucleotide can begin. By tracking the light signals from each spot on the flow cell throughout all cycles, the sequence of each individual template fragment can be estimated. Tracking nucleotide incorporation on each strand determines the exact sequence of each individual DNA molecule. The HeliScope system generates nearly a billion reads per run, but the read length of only 35 bp is very short. Due to the short read length and the high instrument costs (Helicos announced a list prize of 1,350,000$) the system had only limited success in the sequencing market. Today Helicos does no longer sell instruments or reagents, but offers the technique as sequencing service, only.

2.2.3.2. Sequencing by Oligonucleotide Ligation and Detection (SOLiD)

The SOLiD technique is the second short read technique besides Illumina/Solexa. The sample preparation is comparable to the pyrosequencing protocol: A library of template DNA fragments is prepared, attached to magnetic beads, amplified via emulsion PCR, and bound to a glass slide. The amplified fragments all bear a specific *P1* adapter that ensures that the starting sequence for every fragment is identical. The template sequence is then estimated by ligation of di-base probes to it, small probes that are labeled fluorescently according to their first two bases. Starting with a primer complementary to the end of the *P1*, each sequencing cycle a di-base probe is ligated, the fluorescence is detected, unextended strands are capped, and the fluorophore is cleaved off the probe (see Figure 2.8, 1.-4.). These steps are repeated to extend the sequence step by step. As the probe is 5bp long after cleavage (see Fig 2.8, 4.), only the first and second base of each 5bp block is known (Figure 2.8, 5.). Therefore after a defined number of sequencing cycles the synthesized complementary strand is removed and the sequencing process is restarted with a primer starting one base earlier compared to the first primer, called $n-1$ primer (see Figure 2.8, 6.-7.). This procedure is repeated four times, for primers $n-1$, $n-2$, $n-3$, and $n-4$, providing a complete coverage of the template DNA (see Fig 2.8, 8.).

The unique feature of SOLiD is the two base encoding technique. As described, SOLiD uses four fluorophores like other techniques, but a light signal does not only encode one detected nucleotide, but according to the incorporated di-base probe a defined dimer of two nucleotides. The colours used for specific dimers are chosen in a way that an unambiguous sequence can be obtained when the starting base is known, although there are only four colours for 16 possible dimers. Furthermore, the two base encoding allows for the correction of sequencing errors during the base calling process and thereby facilitates a very high sequence quality. The drawback of the SOLiD technique is the short read length, limited to 60-75bp, only.

2.2.3.3. Ion Torrent

The "Ion Torrent" system by Life Technologies is an adaptation of the pyrosequencing technique described in Section 2.2.1.1. The technological improvement lies in the detection system used by Ion Torrent. Like 454's PicoTiterPlate, the Ion Torrent uses a high-density array of micro wells, where each well holds a different DNA template. Furthermore each well comprises of an ion-sensitive layer and an ion sensor. Whenever a nucleotide is incorporated by the polymerase while replicating a template DNA, a hydrogen ion is released. This hydrogen ion changes the pH of the sequencing solution, which is detected by the ion sensor as an electrical signal. The sequencer then sequentially floods the chip with one nucleotide after another, creating an electric signal whenever one or more nucleotides are incorporated. Like in pyrosequencing, the signal strength is proportional to the number of incorporated bases, in this case the voltage of the electric signal. Because of the simple sequencing chemistry - no enzymatic cascade, no fluorescence, no chemiluminescence, no optics are needed - the sequencing process is very fast and the reagent costs are quite low for this system (see Table 2.1).

2.2.3.4. Pacific Biosciences Real-time sequencing

Another new sequencing technology is single molecule real-time (SMRT) sequencing by Pacific Biosciences. The technology monitors the duplication of a template DNA strand by DNA polymerase in real time. Real-time sequencing is based on a nanoscale optical approach, facilitated in so-called zero-mode waveguides (ZMW). A ZMW is a tiny hole in a 100nm film deposited on a glass slide of only a few nanometers in diameter with a single DNA polymerase anchored at the bottom of this hole. As the diameter of the ZMWs is much smaller than the wavelength of visible light ($\sim 400-700nm$), light decays as it enters the ZMW. By shining a laser through the glass slide into the ZMW, only the bottom 30nm of the ZMW are illuminated, narrowing the signal detection area exactly to the area in which the polymerase is located. Fluorophore-labeled nucleotides are flooded over the glass

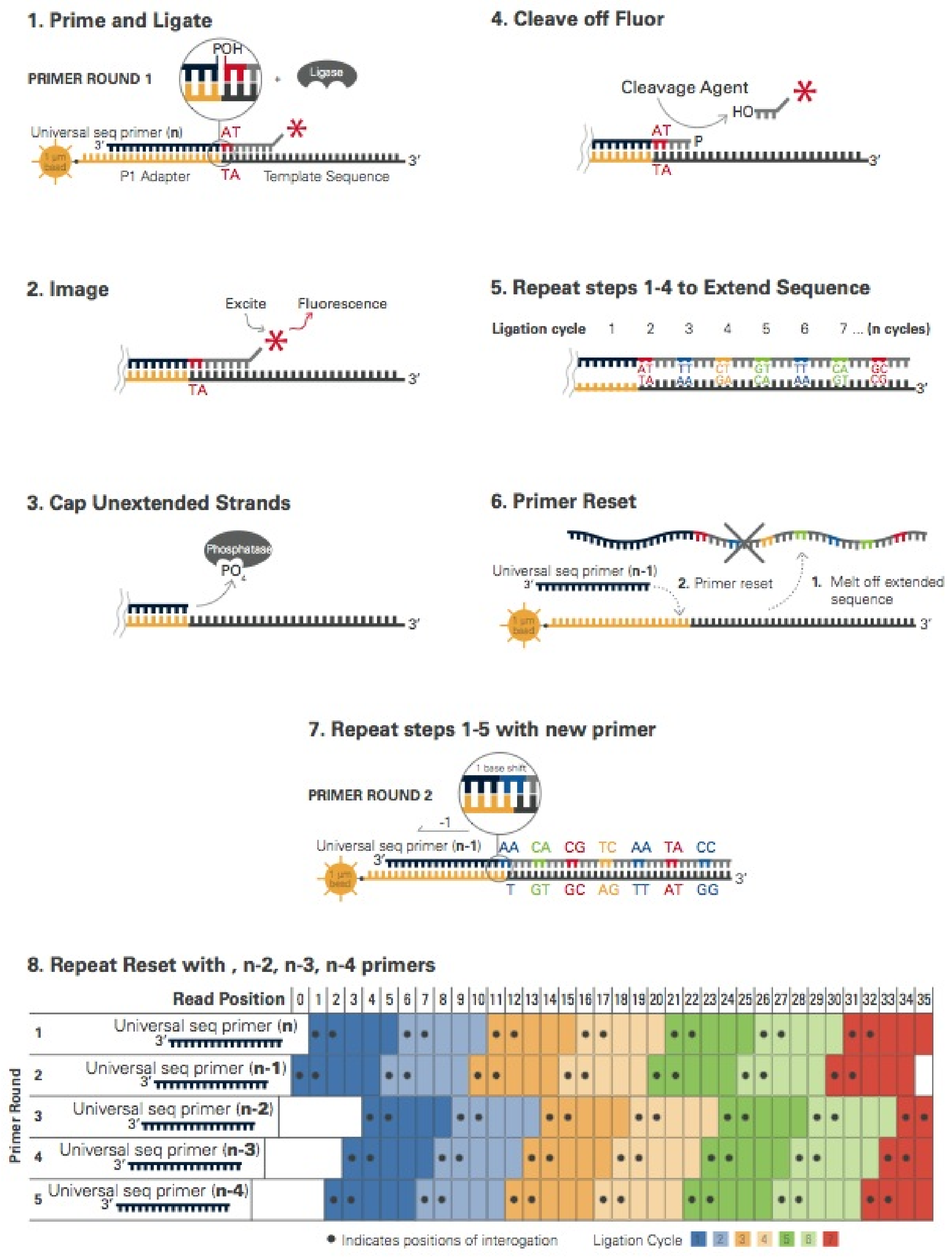

1. Prime and Ligate
PRIMER ROUND 1
POH
Ligase
Universal seq primer (n)
AT
TA
1 μm bead
P1 Adapter
Template Sequence
4. Cleave off Fluor
Cleavage Agent
HO
P
2. Image
Excite
Fluorescence
5. Repeat steps 1-4 to Extend Sequence
Ligation cycle 1 2 3 4 5 6 7 ... (n cycles)
3. Cap Unextended Strands
Phosphatase
PO4
6. Primer Reset
Universal seq primer (n-1)
2. Primer reset
1. Melt off extended sequence
7. Repeat steps 1-5 with new primer
PRIMER ROUND 2
1 base shift
-1
AA CA CG TC AA TA CC
T GT GC AG TT AT GG
8. Repeat Reset with , n-2, n-3, n-4 primers
Read Position
Primer Round
Universal seq primer (n)
Universal seq primer (n-1)
Universal seq primer (n-2)
Universal seq primer (n-3)
Universal seq primer (n-4)
Indicates positions of interrogation
Ligation Cycle

Figure 2.8.: Overview of the sequencing steps for the SOLiD sequencing by ligation technique. Figure taken from `http://seqanswers.com`.

slide with the ZMW array, and if a nucleotide matching the template DNA passes the polymerase it is incorporated. The technique has the ability to detect a single incorporation event.
The system provides a SMRT cell with arrays of $\sim$ 75000 ZMW on glass slides and allows read length of up to 15,000bp. Thereby it is the marketed sequencing technology with the longest read length at the moment. Furthermore, the technique is very fast, allows the sequencing of single DNA molecules, and requires low amounts of template DNA. The drawback of the technique is the low throughput with less than 1Gb output per run. Even more problematic is the low accuracy of the sequencing technique with an error rate of $\sim$ 15% (Eid et al., 2009) which above all consists mostly of insertion and deletion errors.

2.3. Assembly: From reads to complete genomes

As described in Section 2.2 the costs for DNA sequencing decreased dramatically in the last years. But in order to gain a complete genome sequence from the raw sequencing reads, all current sequencing techniques require an assembly. Assembly denotes the process in which a contiguous consensus sequence is formed of the single sequencing reads. There are numerous applications for this purpose, commercial programs as well as free software, which are all based either on overlap graphs (e.g. PHRAP[6], Celera Assembler, MIRA, or CAP3 (Myers et al., 2000; Chevreux, 2005; Huang and Madan, 1999)) or DeBruijn graphs (e.g. Velvet, SOAPdenovo, or ABySS (Zerbino and Birney, 2008; Li et al., 2010; Simpson et al., 2009)). Both assembly approaches are based on the estimation of overlaps between reads and both assembly techniques - to a certain degree - have problems to resolve repetitive regions. Furthermore, the sequencing techniques don't produce a uniform sequencing coverage. As an example, a high GC content of the template DNA fragment may lead to backfolding during the PCR amplification step (as explained in section 2.2.1.2) and thus to an underrepresentation of the respective sequence. This leads to assembly results that do not show one final consensus sequence per replicon, but a number of assembled contiguous sequences (contigs).

2.3.1. From contigs to scaffolds: Paired-end sequencing

A common way to simplify the assembly of whole genome sequencing data is the use of paired sequence information. The basic principle is to assemble not only a set of independent reads, but to take into account knowledge about the distance between and the orientation to each other of pairs of reads. When this technique was first described in 1995 (Roach et al., 1995) the read pairs were obtained by sequencing both ends of clonal template DNA fragments of known size, therefore the technique is called paired end sequencing. If two paired reads are located in different contigs after an initial assembly, this information can be used to bring the

[6]`http://www.phrap.org`

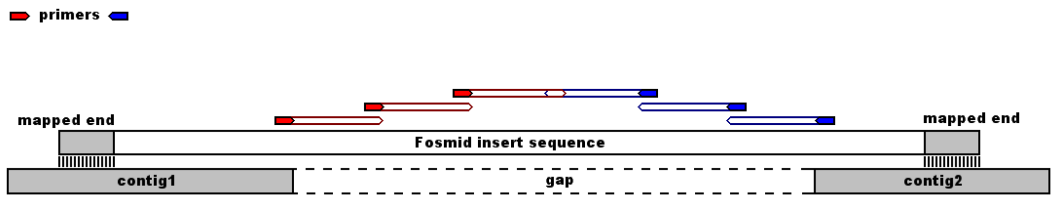


Figure 2.9.: Schematic representation of fosmid walking for genome finishing: The fosmid insert sequence is spanning a gap betwee contig1 and contig2. The size of the gap can be estimated as the insert size of the fosmid is known (usually around 40 kb). For gap closure primers can be designed based on the contig end sequences pointing into the gap. With classical Sanger sequencing one can sequence up to 1000 bp into the gap using the fosmid as template DNA. In the next step primers are designed for the end of the newly sequenced fragment and a further 1000 bp are sequenced. This processed is repeated iteratively until the gap is closed.

contigs into line. The distance between the paired sequencing reads is called the insert size and defines which gap sizes can be spanned by paired end information. All sequencing systems provide protocols for paired end sequencing with different insert sizes, with the exception of Pacific Biosciences where this technique is not needed due to the long read length.

2.3.2. Draft genomes vs. finished genomes

To achieve a single high quality consensus sequence of a complete genome, a further processing step is needed, the so-called finishing. The most established techniques to close the remaining gaps in a genome sequence are gap closure PCRs or fosmid walking. In the first approach, which is mainly used for small gaps, primers are designed on the contig ends pointing out of the contig. If the ordering of the remaining contigs is known, dedicated primer pairs can be designed that flank a gap between two contigs, otherwise all combinations of contig end primers have to be tried. Using the complete genomic DNA as template the compatible primer pairs will lead to the amplification of a DNA fragment spanning the gap, given that the size of the gap is smaller than the maximum sequence length that can be amplified using standard PCR ($\sim$ 1000bp) and sequenced with the Sanger technique afterwards. For longer gaps fosmid walking, a refined primer walking (Kieleczawa et al., 1992) strategy, is the adequate technique. In the first step a library of fosmids, bacterial F-plasmids with an average insert length of $\sim$ 40kb, is created from the genomic DNA template. The ends of the 40kb fosmid insert sequence are decoded using Sanger sequencing and mapped onto the set of contigs. If the two ends map to different contigs, a scaffolding of the contigs is achieved. Based on the scaffolding information primers can be designed for the contig ends and used as starting points for Sanger sequencing into the gap, extending the contig sequence by up to 1000 bp. This can be done from both ends of the gap and is repeated consecutively until the complete gap is bridged (see Figure 2.9).

While the sequencing of a complete genome became more and more affordable over the last years, the cost for the finishing of a genome after the initial assembly remained comparably high as the main finishing techniques still rely on Sanger sequencing. The use of and the need for high quality finished genomes is a matter of passionate discussion since the Sanger era (Fraser et al., 2002; Branscomb and Predki, 2002) and there is still a heated debate. Some experts support the opinion that draft genomes are sufficient to address most scientific question and finishing is not needed (Aury et al., 2008). The main argument for draft genomes are the reduced cost and manpower. Given modern sequencing systems and bioinformatics tools, an unfinished draft genome of good quality can be obtained at a fraction of the costs and manual effort needed for a high quality finished genome. In addition, usually $\sim 99\%$ of the genomic sequence are obtained in the initial assembly, therefore a draft genome allows the analysis of the majority of genomic features of an organism.
A finished genome on the other hand poses a permanent valuable scientific resource, while draft genomes don't provide all information about an organism (Alkan et al., 2010). As all post-genomic analyses have to rely on the genome sequence, it always adds bias to sequence only up to draft status. For example the information about genomic rearrangement events will be lost in draft genomes. In gene content analyses one has to be aware that every remaining gap in a draft genome may destroy one gene and therefore strongly influences the results. Nearly 50% of the genomes uploaded to the NCBI (National Center for Biotechnology Information) databases are only draft genomes[7], therefore they have to be treated with care in all post-genomic analyses. The implications for the methods presented here will be discussed in the respective chapters of this work.

2.3.3. Tools for contig layout and genome finishing

If one wants to take the effort to create a finished genome there are several bioinformatics tools that can help in this task. One of the oldest and most established applications is Consed (Gordon et al., 1998), a tool that can read assembly information and provides a graphical user interface for manual assembly refinement. As Consed takes into account the assembly information of every single read, it struggles with the huge numbers of reads from modern sequencers.
To support clone libraries for genome finishing, e.g., bacterial artificial chromosomes (BACs) or the already mentioned fosmids, the BACcardi tool was developed (Bartels et al., 2005). BACcardi provides a graphical user interface that visualizes a mapping of BAC or fosmid end sequences on an assembled draft genome. BACcardi can be used to identify misassemblies or create a scaffolding of contigs based on the clone map.

[7]http://www.ncbi.nlm.nih.gov/genomes/static/gpstat.html (revised 16.02.2012)

As described in the previous section among the main methods to close gaps in an fragmented assembly are gap closure PCRs. For this methods primers are needed that point from two contig ends into the gap between the contigs. Accordingly, information about the ordering and orientation of the contigs is crucial. The most commonly used method to obtain a contig ordering is to align the contigs to a closely related reference. Several tools were published for this purpose: ABACAS (Assefa et al., 2009) uses MUMmer alignments to order contig according to a reference and also provides an integration of the primer design software Primer3 (Rozen and Skaletsky, 2000). The Mauve aligner looks for locally collinear blocks (LCBs), homologous sequences between the input genomes separated by rearranegment events. The Mauve Contig Mover (MCM) consideres contig boundaries as artificial LCB edges and tries to find a contig layout with minimal number of LCBs (Rissman et al., 2009). OSLay (Richter et al., 2007) uses BLAST or MUMmer ouput and computes a so-called "comparison grid" of partial hits. In a second step a graph of all possible contig connections is calculated based on the grid. The optimal synthetic layout (OSL) is defined as the path through the graph that maximizes the sum of weights of all realized edges in the graph. Projector2 (van Hijum et al., 2005) also uses BLAST comparisons to estimate the order, orientation, and spacing of contigs in relation to a reference. Furthermore Projector2 features repeat masking and primer design. A tool from Bielefeld University is r2cat (Husemann and Stoye, 2010) which uses a "contig adjacency graph" to find the most likely contig layout. A unique feature of r2cat is that it can use several reference genomes and takes into account the phylogenetic distances between the several references. Furthermore r2cat also features primer design using a custom algorithm. All these ordering tools can increase the quality of a draft genome without the need to create any new sequencing data, thus a contig ordering step is strongly recommended if a closely related reference genome is available.

2.4. Comparative genomics

As described in the introduction, the term "comparative genomics" describes the analytical comparison of a number of more or less closely related genome sequences. First attempts to compare complete bacterial genomes have been made since the beginning of the millennium, but the huge amount of data generated since the introduction of pyrosequencing and successive NGS techniques opened up undreamed-of possibilities for large scale comparative studies. This section will give an overview of the historical background of comparative genomics, furthermore the terminology of the field and the most prominent fields of application in which comparative genomics is used will be introduced.

2.4.1. Historical background

In the early days of microbiology the comparison of microbial organisms was limited to morphological or physiological features like gram staining, the existence of an envelope, or the shape of the microbes. Thus, in the mid fifties the definition of the taxonomic term “species” referred to a group of cultures or strains which were accepted by bacteriologists to be sufficiently closely related based on their visible features (Hollricher, 2007). With the discovery of the DNA structure by Watson and Crick (1953) and Franklin and Gosling (1953) the era of molecular biology started, and the value of genetic information for the classification of bacterial species was soon understood (Zuckerkandl and Pauling, 1965). Due to the work of pioneers like Carl Woese, who established 16S ribosomal RNA sequencing in phylogenetic taxonomy and played a substantial role in the definition of the current bacterial taxonomy (Woese et al., 1985; Woese, 1987), the classification of microorganisms was now based on measurable features. In 1987 a more fundamental proposition of the term “species” by the “Ad Hoc Committee on Reconciliation of Approaches to Bacterial Systematics” (Wayne et al., 1987) considered quantities including strains’ DNA molecules reassociation values and phenotypic traits. However, in recent times, these classical approaches are likely to be outdated by future deductions which may be taken from the increasing collection of available whole genome sequences. Now as the complete genome sequences of several organisms have become available, the focus has shifted to comparisons at the whole genome level.

One question that inevitably arises from the rapidly increasing amount of genomic data is if the genetic variability of a species can or should be described by the use of only one single strain. Several studies in the last years negated this question, such as the comparison of *Escherichia coli* strains K12 and O157:H7 by Perna et al. (2001) that revealed 1387 genes to be specific to strain O157:H7. Further studies like the comparison of 17 *Streptococcus pneumoniae* strains by Hiller et al. (2007) or the comparison of 17 lactic acid bacteria by Makarova et al. (2006) showed that a species can be hardly described by the sequence features of a single strain as even strains from the same genus frequently lose existing genes or acquire new ones. Interestingly, even clinical isolates taken at nearby locations from patients with similar symptoms showed divergent genotypes (Hiller et al., 2007).

The key to understanding the evolution and capabilities of certain bacteria is the comparison of the genetic repertoire of the organism of interest with the genomes of closely related strains. For this type of comparison, sequence similarities between the coding sequences of the compared organisms are used to define genomic subsets like the core genome, the pan genome, or singleton genes. In the next section the terminology of the field of comparative genomics will be defined, explaining different concepts of sequence similarity and gene evolution as well as the aforementioned genomic subsets.

2.4.2. Terminology

Whenever one wants to compare the gene content of different genomes, it is important to be aware of the different evolutionary meanings of sequence similarity. Several terms to describe evolutionary relationships between sequences were defined by Walter M. Fitch (Fitch, 1970, 2000) and became basic vocabulary for the description of molecular phylogeny. The following list, taken from the review by Bachhawat (2006), gives an overview of the most important terms as defined by Fitch.

- **Homology** is the relationship of any two characters that have descended, usually through divergence, from a common ancestral character. While every discriminable feature poses a character, in this work genes (DNA stretches encoding an enzyme or RNA) or their respective translated protein sequences are the characters of interest.
- **Homologues** are thus genes that can be attributed to a common ancestor of the two organisms during evolution. Homologues can either be orthologues, paralogues, or xenologues.
- **Orthologues** are homologous genes that have evolved from a common ancestral gene by speciation. They usually have similar functions.
- **Paralogues** are homologues that are related or produced by duplication within a genome. They often have evolved to perform different functions.
- **Xenologues** are homologues that are related by an interspecies (horizontal) transfer of the genetic material for one of the homologues. The functions of the xenologues are quite often similar.
- **Analogues** are non-homologous genes/proteins that have descended convergently from an unrelated ancestor (this is also referred to as “homoplasy”). They have similar functions although they are unrelated in either sequence or structure. This is a case of “non-orthologous gene displacement”.
- **Horizontal (lateral) gene transfer** is the movement of genetic material between species (or genus) other than by vertical descent. In bacteria this process occurs by either natural transformation, conjugation, or transduction (through viruses).

In comparative genomics the identification of orthologous genes is one of the main goals, as orthologous genes usually share a common function and, as Fitch (1970) states, orthologous genes alone allow phylogenetic conclusions:
“Where the homology is the result of speciation so that the history of the gene reflects the history of the species (for example a hemoglobin in man and mouse) the genes should be called orthologous (ortho = exact). Phylogenies require orthologous, not paralogous, genes.” (Fitch, 1970, p. 113)

The reliable identification of orthologous genes based on sequence similarities is a challenging problem and appropriate methods for orthology detection are a matter of passionate discussions. Especially the determination of proper similarity thresholds is often problematic as the thresholds are in most cases selected in a more or less arbitrary way and it is hard to find two studies with comparable threshold values (see Table 2.2). Randomly chosen cutoff values are a problem that makes it hard to compare different comparative studies.

Table 2.2.: Cutoff values from comparative studies: Overview of the orthology thresholds used in six randomly selected comparative studies cited in this work. It becomes obvious that there is no generally accepted method, but there is a wide range of used cutoffs.

Author	Organism	orthology criterion
Hiller et al. (2007)	*S. pneumoniae*	70% identity over 70% length
Lefébure and Stanhope (2007)	*Streptococcus*	evalue $< 1e^{-05}$
Deng et al. (2010)	*L. monocytogenes*	50% positively scored bases
Yukawa et al. (2007)	*C. glutamicum*	evalue $< 1e^{-04}$
Brzuszkiewicz et al. (2006)	*E. coli*	90% identity over 90% length
Tettelin et al. (2005)	*S. agalactiae*	50% identity over 50% length

To overcome this problems several sophisticated orthology estimation methods have been published in the last years, they will be discussed in Section 4.2.1.2 as the identification of orthologous genes is one of the main topics of the EDGAR software. As soon as all orthologous genes between two or a number of strains have been identified, this allows us to analyze the distribution of orthologous genes within a set of compared genomes.

2.4.3. Genomic subsets

As described in the last section the assignment of orthologous genes allows an analysis of their distribution within a set of genomes, which led to the definition of special genomic subsets. The following list provides definitions for the most commonly used genomic subsets as used in EDGAR, mainly based on the trailblazing publications of Tettelin et al. (2005) or Medini et al. (2005).

- **The core genome** is the set of genes that has orthologous genes in all genomes of a comparison set, usually all members of a certain taxonomic group (e.g. genus). These core genes mainly encode vital proteins of the primary metabolism and phenotypical features shared by all strains.
- **Singleton genes** or singletons are the exact opposite of the core genes, i.e. genes that are found in only one single genome of the comparison set. Consequently they encode the unique features of the strain they are found in. Singletons are also called “unique genes” or “orphan genes”.

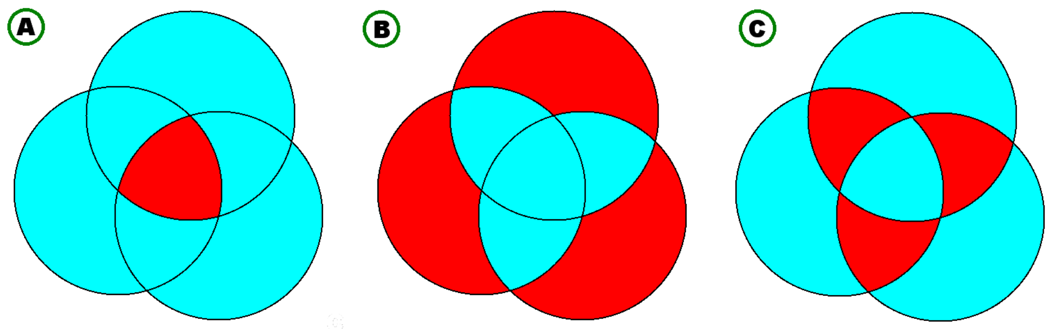


Figure 2.10.: Venn diagram representing the main genomic subsets. Subset of interest marked in red: (A) The core genomes, the set of genes shared by all, in this case three, genomes. (B) The singleton genes which have no orthologs in any other genome. (C) The dispensable genome comprises genes that are found in more than one, but less than all genomes. The genes from sets A, B, and C together build the pan genome.

- **The dispensable genome** denotes the set of genes that has orthologous genes in more than one but less than all genomes. These genes encode phenotypical features only found in a subset of the analyzed genomes.
- **The pan genome**, inspired by the Greek word "pan" for "whole", is the combination of all sets described above and thus comprises of the core genome, all singleton genes and all genes that can be identified in more than one, but not in all compared genomes. It describes the complete genetic repertoire of the analyzed set of genomes.

A graphical representation of the genomic subsets is displayed in Figure 2.10.

Muzzi, Masignani, and Rappuoli pointed out the importance of these concepts, not only to study genetic diversity, but also in terms of medical discoveries and cures (Muzzi et al., 2007). For example in the quest for potential new drugs and vaccines the genes of the core genome of a pathogenic genus are naturally the most promising targets for methods like reverse vaccinology, the screening of the complete genomes of target pathogens for proteins causing immune responses. For all cases where one new organism is compared to a set of already known genomes the singleton genes are without much doubt the most interesting subset. The singleton genes are the unique genes of the new strain and, as all other genes have been observed within the comparison set, are thus the main genes of interest.
The dispensable genome plays an important role in the analysis of genera where subsets of a genus are separated by distinct features, e.g. pathogenicity. There are genera where pathogenic and non-pathogenic members in close evolutionary distance are known, e.g. in the genera *Meningococcus*, *Escherichia,* or *Bacillus.* Genes that are found in pathogenic strains but not in non-pathogenic strains are

the first candidates in the search for virulence factors. The pan genome is mostly interesting in phylogenetic analyses as will be explained in the next section.

2.4.4. The bacterial pan genome

The idea of a pan genome was shaped by Herve Tettelin and Duccio Medini in 2005. Tettelin et al. (2005) sequenced six *Streptococcus agalactiae* strains representing the main serotypes of this pathogenic species using whole-genome shotgun sequencing and compared these six strains to two strains already available in the public databases. In their analyses they found a significant amount of genes not being shared among the compared strains. This led to the definition of the pan genome:
"Comparative analysis [...] suggests that a bacterial species can be described by its "pan-genome" [...], which includes a core genome containing genes present in all strains and a dispensable genome composed of genes absent from one or more strains and genes that are unique to each strain." (Tettelin et al., 2005, p. 13950)
Tetellin *et al.* claimed that in order to understand the global complexity of a bacterial species the genome sequences of multiple, independent isolates are needed. Furthermore, during their analysis of *Streptococcus agalactiae* strains they noticed that even after sequencing eight strains every newly sequenced strain contributed new genes to the pan genome. This led to a differentiation between open and closed pan genomes that was mentioned in the manuscript by Tetellin *et al.* and published later in the same year by Medini et al. (2005). The genomes within some species like *Bacillus anthracis* (Tettelin et al., 2005) or *Buchnera aphidicola* (Tamas et al., 2002) show nearly no gene rearrangements and have a stable, limited gene pool, i.e., after sequencing a sufficient number of genomes the complete pan genome of the species can be estimated. This is denominated a closed pan genome. But the genomes of most species form a so-called open pan genome, that is with every newly sequenced strain new genes are added to the pan genome. In their study Tettelin and Medini estimated the number of newly found genes for every new genome to be 33 for *Streptococcus agalactiae*, for other species this number can reach from just a few to hundreds of new genes per isolate.

2.4.5. Applications in medical, industrial, and fundamental research fields

There are several applications for comparative genomics in medical and industrial research as well as in fundamental scientific questions like evolutionary studies. As described above, in the medical field comparative genomics is extensively used in the comparison of pathogenic and non-pathogenic species strain types as well as in drug design and reverse vaccinology. The profit of comparative analyses of pathogenic species is obvious as it allows to identify the genomic features that constitute pathogenicity or virulence, thus this was among the earliest approaches of comparative genomics (Brzuszkiewicz et al., 2006; Bolotin et al., 2004; Eppinger

et al., 2004). Another emerging trend that comes with the decreasing sequencing cost is the analysis of large sets of genomes from disease outbreaks to analyze point mutations that influence fitness and transmission of the otherwise *de facto* identical genomes (Ford et al., 2011; Niemann et al., 2009). Vaccinology and drug design largely benefit from the availability of multiple genome sequences as this allows to understand the distribution, diversity, and characteristics of potential antigens based on multiple genome sequences (Tettelin, 2009). The virtue of multiple genome screening for reverse vaccinology was demonstrated in several studies (Maione et al., 2005; Groot and Rappuoli, 2004; Liu et al., 2009a).
The comparison of completely sequenced genomes also has several applications in the investigation of industrial relevant bacteria. Industrial relevant production strains like *Corynebacterium glutamicum* (Rückert et al., 2003) or *Escherichia coli* (Blattner et al., 1997) traditionally are target to metabolic engineering (Lee et al., 2005b). To support this process one can compare complete genomes to gain insight into the contribution of certain genes to the efficiency of metabolic pathways and thereby identify candidate genes to be manipulated (Lee et al., 2005a; Yukawa et al., 2007). Existing production strains can be analyzed for differences to type strain on single nucleotide level to identify polymorphisms that cause a desired behavior (Ohnishi et al., 2002).
Finally, there sure are countless application scenarios for comparative studies in fundamental research, especially in the estimation of evolutionary relationships between organisms. The main application is the survey of the evolutionary development of and relationships between several genomes. Examples are studies of the evolution of the core and pan-genome in terms of recombination and genome composition (Lefébure and Stanhope, 2007), analyses of genome clusters and operon conservation (Kant et al., 2011), intraspecific niche expansion and genome diversification in species with closed pan genome (Deng et al., 2010), or the genome dynamics of populations from different locations and the linkage of gene loss and gain to the habitat (Reno et al., 2009). All these examples show the broad field of applications of comparative genomics.

2.5. Parallelization in Biosequence Analysis

As described in this chapter, the output of modern sequencing devices is constantly and rapidly rising. This poses a major challenge for all fields of bioinformatics that deal with sequencing data. Figures 1.1 and 2.7 show that the decrease of sequencing costs can no longer be compensated by the constant increase of computed power according to Moore's law. Thus new approaches are needed to ensure the analysis of the increasing amounts of data in a timely manner. One such approach is the intensified usage of parallel programming.
The basic idea behind parallel programming is to divide a problem into smaller, easier to solve sub-problems, a technique known as "divide and conquer". This approach generates a huge number of small independent tasks by parallelization,

such that each problem can be solved on a different processing unit, either cores of one computer or a compute cluster. Once all calculations are finished they are combined into a solution of the original problem. While server systems with 96 and more CPU (Central Processing Unit) cores are available today and can be used to efficiently speed up multi-threaded software, they still pose a significant capital investment.
Therefore specialized hardware for parallel processing has been employed, such as Celera's[8] GeneMatch™ASIC (Application-specific integrated circuit) approach which uses specialized processors to accelerate several bioinformatics algorithms. A few years later Active Motif[9] developed Field Programmable Gate Arrays (FPGAs) to run adapted versions of the Smith-Waterman algorithm (Smith and Waterman, 1981), BLAST (Altschul et al., 1990), and HMMer (Eddy, 2011) software with significant performance gains. Unfortunately the prices for such optimized special purpose hardware together with appropriate licenses fall in the same expensive investment range as large servers.
Another special purpose hardware solution comes from Convey Computer[10] with the Convey HC ("Hybrid Core") series. These systems combine standard X86 architecture with an FPGA co-processor and allow the execution of code with a standard instruction set as well as with application-specific instructions on the FPGA component. Convey Computer offers several bioinformatics solutions like global alignment algorithms or an implementation of the popular assembly software Velvet (Zerbino and Birney, 2008) which achieve considerable speedups, but likewise to Actice Motif solutions a high initial investment is needed for the hybrid core hardware.
A more cost efficient possibility is the use of existing hardware which can be employed for scientific computing through different frameworks. The MMX technology introduced by Intel in 1997 is a special instruction set that supports SIMD (Single Instruction Multiple Data) approaches, where SIMD is a class of parallel computers that perform identical operations on multiple data points simultaneously. This allows the parallel processing of large data amounts with specialized CPU instructions. The MMX instruction set as well as its successor SSE (Streaming SIMD Extensions) have been used to accelerate implementations of the Smith-Waterman algorithm (Rognes and Seeberg, 2000; Farrar, 2007). In 2008, also a first attempt on non-PC hardware has been published. SWPS3 (Szalkowski et al., 2008) employs the Playstation 3's cell processor to speed up an adapted version of the Smith-Waterman algorithm.

Another promising approach to speed up computational intensive tasks is the usage of graphics hardware scientific computing. To realize the more and more realistic graphics of up-to-date video games recent graphics adapters have a huge compute power (see Figure 2.11 for a comparison of recent GPUs and CPUs), and by design

[8]Celera Corporation, 1401 Harbor Bay Parkway, Alameda, CA, USA
[9]Active Motif, 1914 Palomar Oaks Way, Carlsbad, CA, USA
[10]Convey Computer, 1302 East Collins Boulevard, Richardson, TX, USA

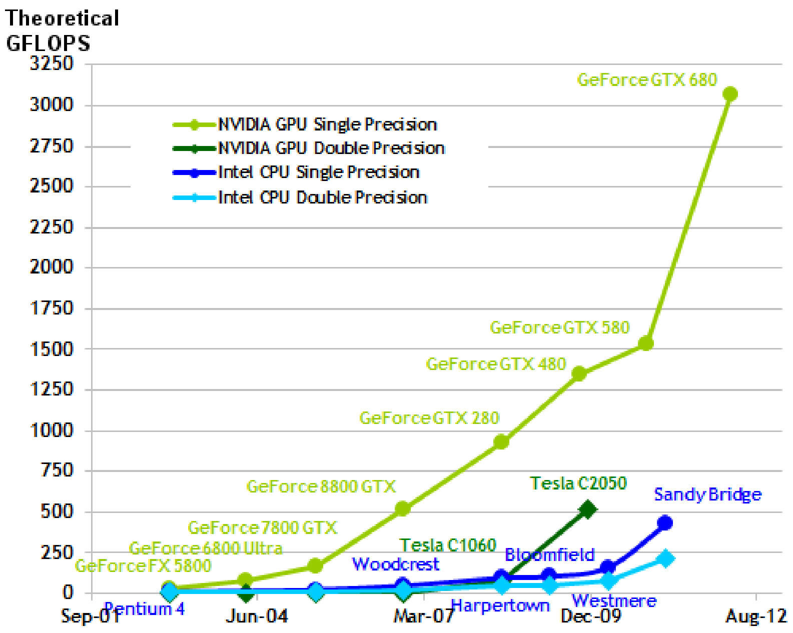


Figure 2.11.: Comparison of the theoretical compute performance of CPUs and GPUs in GFLOPS (FLOPS = Floating-Point Operations Per Second). It is evident that up-to-date GPUs can outperform that latest CPU generation many times over, especially in single-precision calculations. Source: `http://docs.nvidia.com/cuda/cuda-c-programming-guide/index.html`

they support massive parallel execution computation. Although there are some limitations, especially in the amount of available memory (an up-to-date gaming graphics adapter has 2 gigabytes of video RAM (VRAM) which is shared by up to 512 processing units, called "CUDA cores"), if a problem can be split up in small independent tasks GPU programming can provide an extremely efficient solution. First approaches to use graphics cards as hardware accelerators for bioinformatics algorithms (Liu et al., 2006) relied on OpenGL, resulting in a difficult and limited implementation. Today, frameworks simplify software development by hiding the layer of 3D programming behind a more general Application Programming Interface (API). Thus, the focus of development shifts from fitting a given algorithm to OpenGL operations to the development of the best implementation. The CUDA platform (`http://www.nvidia.com/object/cuda_home.html`) developed by the Nvidia cooperation[11] and AMD's[12] STREAM framework (`http://www.amd.com/stream`) are novel approaches to use the huge computational power of modern graphics cards not only for games but also for scientific applications. Contemporary graphics processing units are built as massively parallel computational devices, optimized for floating point operations. In CUDA, the programming is done in CUDA-C/C++,

[11]Nvidia Corporation, 2701 San Tomas Expressway, Santa Clara, CA, USA
[12]AMD Headquarters, One AMD Place, Sunnyvale, CA, USA

a slightly extended version of C/C++ that supports commands to express parallelism and data and thread management. CUDA C/C++ code has to be compiled with `nvcc`, a custom Nvidia LLVM-based C/C++ compiler. Compared to universal central processing units (CPUs) used in every computer, GPUs are specialized for parallel execution of many small tasks while CPUs are designed to execute fewer large tasks sequentially. As such, GPUs are also well suited for the highly parallel computation of small-scale tasks.

CHAPTER 3

Motivation and goals of this work

Comparative genomics is a relatively new and rapidly evolving scientific field, hence there is a urgent need for appropriate software to analyze multi genome datasets and to automate the comparative methods. Especially the rapid development of sequencing technology entails undreamt of opportunities for research in molecular biology. Two applications for comparative genomics are presented in this work, in this chapter the motivation for the development of the two tools EDGAR and SARUMAN will be described and goals will be defined.

3.1. Comparative genomics on gene level

The exponential increase in the number of completely sequenced genomes set in motion by the rapid development of sequencing technology described in chapter 2 makes it feasible to analyze large groups of related genomes in a comparative approach. This allows complex or completely new analyses and opens up the field of comparative genomics. As described in the last chapter there are numerous different studies and application examples, but all of them are either based on comparisons of the gene content of genomes of interest or on the analysis of single nucleotide polymorphisms. Naturally, there is a need for tailored bioinformatics tools to support both experimental ways.

A main task in comparative genomics is the identification of orthologous genes in different genomes and the classification of genes as core genes or singletons. As the application examples presented in section 2.4.5 show, there is a high demand for tools to calculate the genomic characteristics of a species' pan-genome and to reliably identify the described genomic subsets.

Several tools for the comparison of genomes were developed before or in the beginning of the NGS era, comprising for example xBASE or GeConT (Chaudhuri and Pallen, 2006; Chaudhuri et al., 2008; Ciria et al., 2004), but these tools were either focused on only pairwise genome comparisons or were designed with a focus on the comparison of the genomes of different species. To support the comparison of multiple strains of the same species, databases like the Comprehensive Microbial Resource (CMR) or the Microbial Genome Database (MBGD) were designed, but both databases comprise only public genomes from the major sequence repositories. The list of features and the user-friendliness provided by both databases is limited. Furthermore both databases focus on the genomic subsets and do not provide further analysis or visualization features. Thus there is a demand for a software that can calculate the genomic subsets and provide plausible visualizations of the calculated data. Furthermore a proper support for the work with confidential unpublished genomes is needed as well as a support for unfinished draft genomes. Another problem in comparative analyses is the selection of a proper threshold in orthology estimation. As explained in Section 2.4.2 and illustrated in Table 2.2, there is no commonly accepted threshold or thresholding procedure for orthology estimation between the genes of a set of genomes. It would be highly beneficial to have a generic orthology criterion calculated from and tailored to the compared organisms.

3.1.1. Goals

The analyzed problem and shortcomings of existing solutions for comparative genomics on gene level gave the motivation to develop an own software to provide a suitable analysis platform for comparative studies: EDGAR - "**E**fficient **D**atabase framework for comparative **G**enome **A**nalyses using BLAST score **R**atios". The following goals were defined for EDGAR:

- Automatic estimation of an orthology threshold intrinsically calculated from the analyzed data.
- Calculation of genomic subsets for a set of genomes and subsets thereof together with a plausible presentation of the results.
- Proper handling of draft genomes.
- Analysis of public as well as unpublished confidential genomes in a project based approach.
- Preparation of a public database including all suitable genomes from the major sequence repositories to provide an alternative for the above mentioned databases and to simplify the process of obtaining new biological insights into the differential gene content of kindred genomes.

- Support for a phylogenetic analysis of a genome set.
- Support higher level analysis, i.e., the comparison of defined sets of genomes among each other.
- Design of an easy-to-use web based user interface to allow collaborative work of different institutions on the same data.

The resulting software will be presented in chapter 4.

3.2. Comparative genomics on single nucleotide level

For some comparative analyses the resolution of gene content comparisons is just not high enough, e.g., the genomes of pathogenic *Mycobacterium tuberculosis* strains usually differ by not more than a few dozen to a few hundred single nucleotide polymorphisms (SNPs) (Niemann et al., 2009). The standard method to identify such point mutations is a re-sequencing approach using one short read NGS technique. For such a re-sequencing approach the crucial step is the mapping of the sequencing reads to the reference genome. As the short read NGS techniques provide huge numbers of reads (see Table 2.1), computational efficiency is crucial in this step.
There are plenty of tools to create local alignments, most based on the Smith-Waterman algorithm (Smith and Waterman, 1981), but the alignment of a very short read sequence to a very long reference genome is too time consuming to apply it millions of times. Therefore dedicated short read alignment approaches were published, but all early approaches were either too slow for a high throughput computation of read mappings, or they used some kind of heuristic and could therefore not guarantee a complete mapping result, where "complete" means a mapping where all possible alignments for a user-defined error threshold are identified and reported. Furthermore existing mapping software often showed a tendency to especially miss insertions and deletions in the mapping process.

A new trend in bioinformatics is the use of graphics hardware to solve computationally intensive problems. The "Compute Unified Device Architecture" (CUDA[1]) API provided by the graphics card manufacturer NVIDIA allows to use the extreme compute power of modern graphics cards by executing custom `C` code on them. Graphics card programming has huge potential for the solution of parallelizable small scale problems, and thus seems perfectly suited for short read mapping. Although, graphics card programming had hitherto never been applied to the short read mapping problem.

[1]`http://www.nvidia.de/object/cuda_home_new_de.html` (as at 22.07.2012)

3.2.1. Goals

The limited accuracy or unsatisfactory runtime of existing short read alignment software together with the new opportunities provided by graphics card programming were the motivation to create a new short read mapping software called SARUMAN - "**S**emiglobal **A**lignment of short **R**eads **U**sing CUDA and Needle**MAN**-Wunsch". The main design goals for the SARUMAN mapping software were defined as follows:

- Create a software that provides a perfect mapping result in a sense of optimizing a formerly defined goal. This goal for SARUMAN was to find all possible alignment postions of a set of reads to a reference sequence under a given user-defined error threshold. Furthermore one optimal alignment should be reported for each such mapping position (for a formal definition see 5.3.1.1).
- To allow the computation of such a perfect result in an appropriate run time the immense compute power of modern graphic accelerator hardware should be employed.

As the SARUMAN goal definition includes the reporting of an optimal alignment the computation of this alignment is the ideal step to be executed in parallel on the graphics card. As a local alignment against a sequence of genome size would be too memory consuming to be efficiently parallelized on a graphics card, a pairwise global alignment of reads to a read-sized segment of the genome would be preferable. This approach requires two further goals:

- Design of a fast and sensitive filter algorithm to find possible alignment positions and to extract the respective genome segments for the following pairwise alignment step.
- Implementation of an exact Needleman-Wunsch alignment algorithm that can be efficiently executed on graphics hardware and that can align reads to read-sized genome fragments in a highly parallel approach.

The implemented short read matching software SARUMAN will be presented in chapter 5.

CHAPTER 4

Comparative genomics on gene level: EDGAR

As described in the previous chapters, there are numerous applications for comparative genome analyses on gene level. In this chapter, the software EDGAR - "**E**fficient **D**atabase framework for comparative **G**enome **A**nalyses using BLAST score **R**atios" - will be presented. EDGAR provides analysis capabilities for newly sequenced genomes as well as a public repository of precomputed comparisons of more than 1,400 publicly available bacterial genomes.
In the first section of this chapter previous and competing approaches will be introduced. In the second part, which is based on the publication of EDGAR in *BMC Bioinformatics* (Blom et al., 2009), the software will be discussed in full detail. First, the design decision and used methods of the software EDGAR will be discussed. Particular attention will be paid to the special method for orthology estimation used in EDGAR. Subsequently, the data model resulting from goals and design as well as the actual implementation as a web based user interface will be presented. Finally, some examples of successful applications of the EDGAR software in various contexts will bring forth the results of this first main chapter.

4.1. Previous approaches

It is obvious that an automated calculation of the characteristics of a species' pan genome is highly desirable, thus EDGAR is not the only tool in this field. Different tools have been developed to compare the sequences of genomes, with different analytic methods and different foci in their design. In the following section a brief overview of competing software in the field of comparative genomics will be given.

4.1.1. VISTA tools

The VISTA tools (Frazer et al., 2004) are a set of tools for the comparison of genome sequences. The core feature is an alignment of complete genomes sequences, which can then be visualized with additional data like the functional annotation of the genome. The VISTA alignment view shows intuitively the analogies and differences between genome sequences (see Figure 4.1). Started in 2000 as a server based application with precomputed comparisons, the VISTA server now allows to upload and align own sequences, but the upload is restricted to a combined length of not more than 10 Mb. The precomputed alignments for microbial genomes became part of the Integrated Microbial Genomes (IMG) database, a renowned database hosted by the Joint Genome Institute (JGI). The VISTA service is available for all 1635 genomes in the IMG database. Other visualization features are a synteny dot plot and linear synteny viewer to highlight genomic rearrangements. VISTA allows the user to export conserved sequence intervals, but it does not provide any analysis features on actual protein sequences.

4.1.2. CMR - Comprehensive Microbial Resource

The Comprehensive Microbial Resource (CMR) is a database for the search and analysis of publicly available microbial genomes (Davidsen et al., 2010), that was started in 2001 with 31 available genomes (Peterson et al., 2001). Meanwhile it is hosted by the J. Craig Venter Institute (JCVI), and provides numerous comparative tools for the analysis of 723 genomes stored in its database. With 672 genomes the majority of these is of bacterial origin, 48 are archaeal genomes, and 3 are viral genomes, 659 genomes are finished and 64 are draft genomes (as at 27.07.2012). The main comparative analysis feature is the multi-genome homology comparison tool. This tool allows the user to calculate the number of proteins in a reference genome that have hits to up to 15 selected comparison genomes. The estimated homologous genes between the selected genomes are displayed in an interactive circular plot (see Figure 4.2.A), with each ring in the plot representing the homologous genes of one genome compared to the reference genome (Figure 4.2.B). Special sets of these homologous genes like the core genes or the singletons can be observed and exported in a tabular format (Figure 4.2.C).
Furthermore the CMR offers a generator for synteny plots, called “Protein Scatter Plots”, a “Genome Homology Graph” which shows the number of proteins the reference has in common with all genomes included in the CMR database, or the “GC Comparison Graph” which shows a scatter plot of the %GC contents of homologous genes between two selected genomes. In the “Region Comparison” one can search for occurrences of a certain protein locus in the complete CMR database.

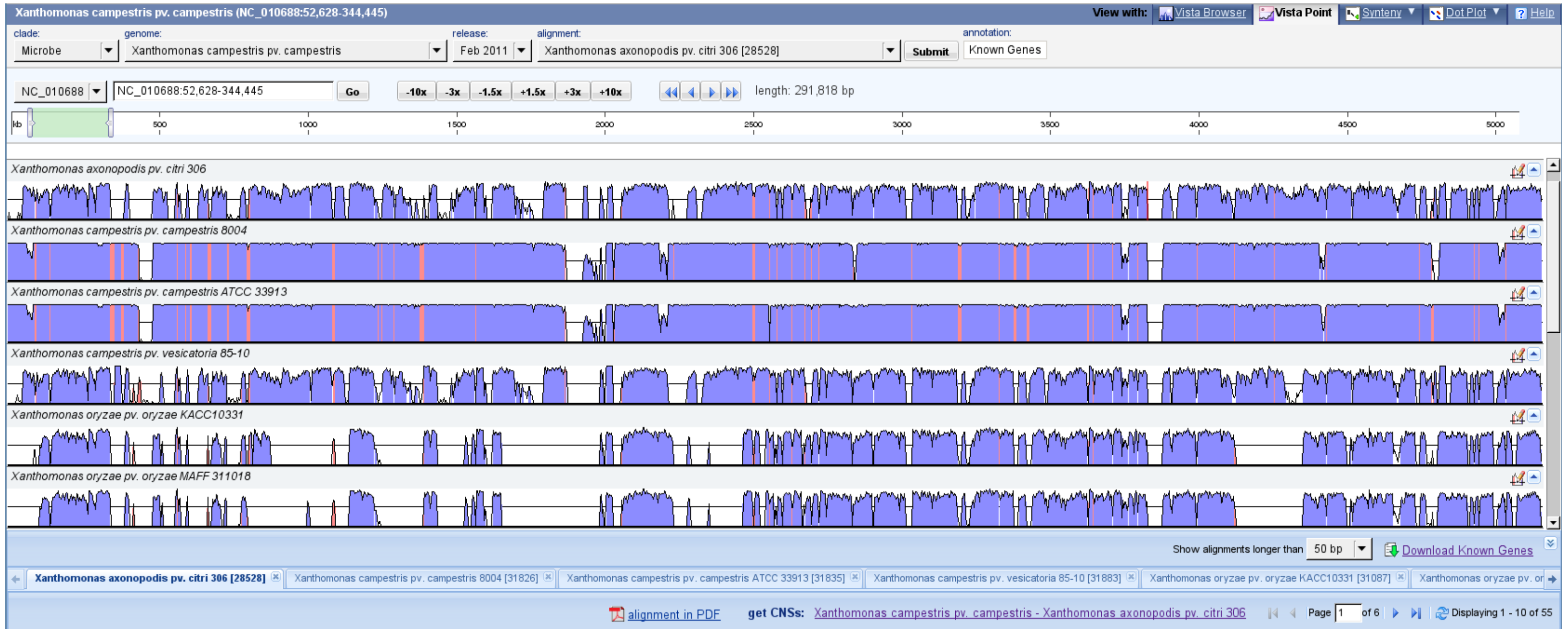
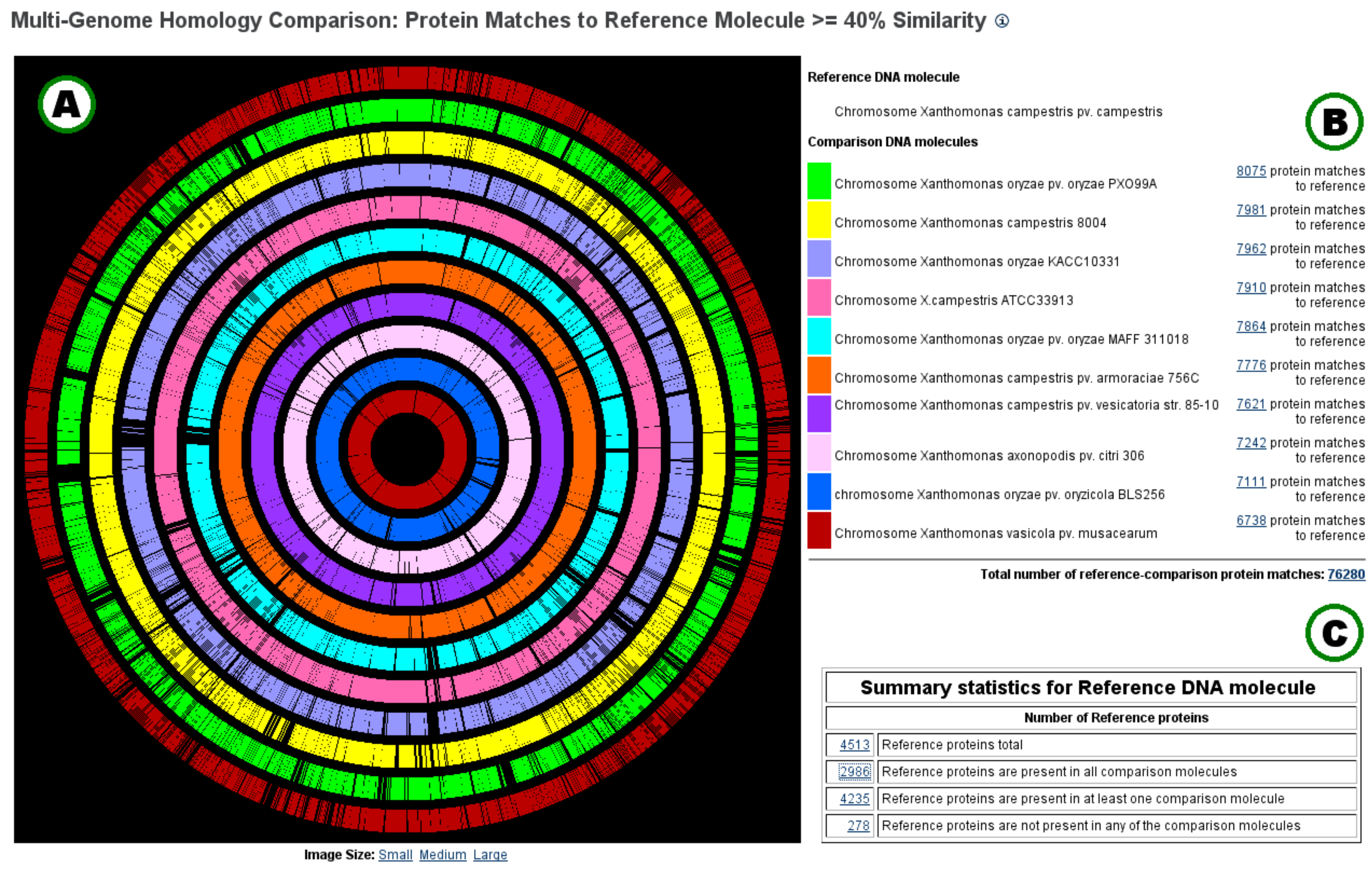

Xanthomonas campestris pv. campestris (NC_010688:52,628-344,445)
View with: Vista Browser | Vista Point | Synteny | Dot Plot | Help
clade: Microbe
genome: Xanthomonas campestris pv. campestris
release: Feb 2011
alignment: Xanthomonas axonopodis pv. citri 306 [28528]
Submit
annotation: Known Genes
NC_010688
NC_010688:52,628-344,445
Go
-10x -3x -1.5x +1.5x +3x +10x
length: 291,818 bp
kb 500 1000 1500 2000 2500 3000 3500 4000 4500 5000
Xanthomonas axonopodis pv. citri 306
Xanthomonas campestris pv. campestris 8004
Xanthomonas campestris pv. campestris ATCC 33913
Xanthomonas campestris pv. vesicatoria 85-10
Xanthomonas oryzae pv. oryzae KACC10331
Xanthomonas oryzae pv. oryzae MAFF 311018
Show alignments longer than 50 bp
Download Known Genes
Xanthomonas axonopodis pv. citri 306 [28528]
Xanthomonas campestris pv. campestris 8004 [31826]
Xanthomonas campestris pv. campestris ATCC 33913 [31835]
Xanthomonas campestris pv. vesicatoria 85-10 [31883]
Xanthomonas oryzae pv. oryzae KACC10331 [31087]
Xanthomonas oryzae pv. or
alignment in PDF
get CNSs: Xanthomonas campestris pv. campestris - Xanthomonas axonopodis pv. citri 306
Page 1 of 6
Displaying 1 - 10 of 55

Figure 4.1.: Screenshot of the VISTA service of the IMG database. The 6 tracks show alignment statistics of 6 *Xanthomonas* genomes against *X. campestris* pv. *campestris* B100. The curve represents a windowed-average identity score calculated over 100bp. The curve is colored if a minimum conservation of 70% was detected over 100bp. The coloring is according to the annotation, where blue coloring denotes coding regions and red coloring shows intergenic regions.



Multi-Genome Homology Comparison: Protein Matches to Reference Molecule >= 40% Similarity

Reference DNA molecule

Chromosome Xanthomonas campestris pv. campestris

Comparison DNA molecules

🟩	Chromosome Xanthomonas oryzae pv. oryzae PXO99A	8075 protein matches to reference
🟨	Chromosome Xanthomonas campestris 8004	7981 protein matches to reference
🟪	Chromosome Xanthomonas oryzae KACC10331	7962 protein matches to reference
🟥	Chromosome X.campestris ATCC33913	7910 protein matches to reference
🟦	Chromosome Xanthomonas oryzae pv. oryzae MAFF 311018	7864 protein matches to reference
🟧	Chromosome Xanthomonas campestris pv. armoraciae 756C	7776 protein matches to reference
🟪	Chromosome Xanthomonas campestris pv. vesicatoria str. 85-10	7621 protein matches to reference
⬜	Chromosome Xanthomonas axonopodis pv. citri 306	7242 protein matches to reference
🟦	chromosome Xanthomonas oryzae pv. oryzicola BLS256	7111 protein matches to reference
🟥	Chromosome Xanthomonas vasicola pv. musacearum	6738 protein matches to reference

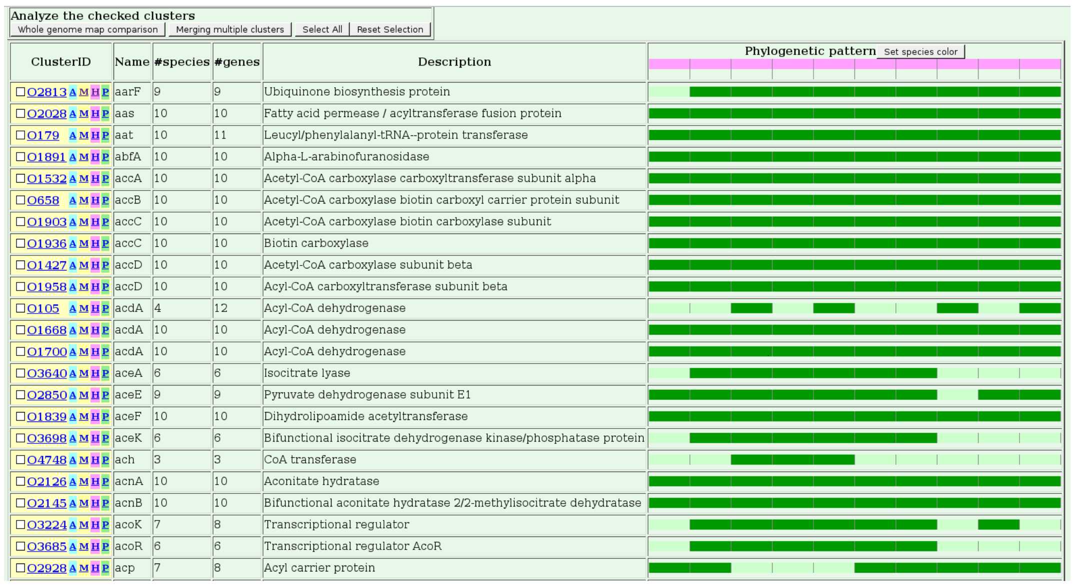
Total number of reference-comparison protein matches: 76280

Summary statistics for Reference DNA molecule	
Number of Reference proteins	
4513	Reference proteins total
2986	Reference proteins are present in all comparison molecules
4235	Reference proteins are present in at least one comparison molecule
278	Reference proteins are not present in any of the comparison molecules

Figure 4.2.: Screenshot of the CMR multi-genome comparison tool (screen elements were compacted to fit into one image). (A) Circular plot of the homologous genes of 11 *Xanthomonas* genomes. The outer circle represents the reference genome, the inner circles denote the 10 genomes compared to the reference. The used colors are shown in the legend (B). Special genomic subsets like the core genome or the singleton genes of the reference genome can be accessed via an extra table (C).

In addition to these gene content/feature comparison tools, the CMR offers a KEGG pathway display tool which allows the user to highlight genes found in one or a set of genomes of interest for certain pathways from the KEGG database (Kyoto encyclopedia of genes and genomes, (Ogata et al., 1999)). An alignment tool features whole genome alignments using the MUMmer software (Kurtz et al., 2004), and the "Attribute Comparison Tools" provide comparisons of genome features like the number of Rho-independent terminators or the number of genes belonging to a certain role category. The CMR represents a convenient resource for genome comparison data and has some useful tools and visualizations, but the drawback of this platform is the fixed set of genomes available for analysis. This problem is exacerbated by the fact that the update intervals of the CMR are very infrequent, the last update dates to the 21.01.2010 (as at 28.07.2012). Private, unpublished genomes can not be analyzed with the CMR platform.

4.1.3. MBGD - MicroBial Genomes Database


Analyze the checked clusters
Whole genome map comparison
Merging multiple clusters
Select All
Reset Selection
Set species color

ClusterID	Name	#species	#genes	Description	Phylogenetic pattern
☐ O2813 A M H P	aarF	9	9	Ubiquinone biosynthesis protein	⬜🟩🟩🟩🟩🟩🟩🟩🟩🟩
☐ O2028 A M H P	aas	10	10	Fatty acid permease / acyltransferase fusion protein	🟩🟩🟩🟩🟩🟩🟩🟩🟩🟩
☐ O179 A M H P	aat	10	11	Leucyl/phenylalanyl-tRNA--protein transferase	🟩🟩🟩🟩🟩🟩🟩🟩🟩🟩
☐ O1891 A M H P	abfA	10	10	Alpha-L-arabinofuranosidase	🟩🟩🟩🟩🟩🟩🟩🟩🟩🟩
☐ O1532 A M H P	accA	10	10	Acetyl-CoA carboxylase carboxyltransferase subunit alpha	🟩🟩🟩🟩🟩🟩🟩🟩🟩🟩
☐ O658 A M H P	accB	10	10	Acetyl-CoA carboxylase biotin carboxyl carrier protein subunit	🟩🟩🟩🟩🟩🟩🟩🟩🟩🟩
☐ O1903 A M H P	accC	10	10	Acetyl-CoA carboxylase biotin carboxylase subunit	🟩🟩🟩🟩🟩🟩🟩🟩🟩🟩
☐ O1936 A M H P	accC	10	10	Biotin carboxylase	🟩🟩🟩🟩🟩🟩🟩🟩🟩🟩
☐ O1427 A M H P	accD	10	10	Acetyl-CoA carboxylase subunit beta	🟩🟩🟩🟩🟩🟩🟩🟩🟩🟩
☐ O1958 A M H P	accD	10	10	Acyl-CoA carboxyltransferase subunit beta	🟩🟩🟩🟩🟩🟩🟩🟩🟩🟩
☐ O105 A M H P	acdA	4	12	Acyl-CoA dehydrogenase	⬜⬜🟩⬜🟩⬜⬜🟩⬜🟩
☐ O1668 A M H P	acdA	10	10	Acyl-CoA dehydrogenase	🟩🟩🟩🟩🟩🟩🟩🟩🟩🟩
☐ O1700 A M H P	acdA	10	10	Acyl-CoA dehydrogenase	🟩🟩🟩🟩🟩🟩🟩🟩🟩🟩
☐ O3640 A M H P	aceA	6	6	Isocitrate lyase	⬜🟩🟩🟩🟩🟩🟩⬜⬜⬜
☐ O2850 A M H P	aceE	9	9	Pyruvate dehydrogenase subunit E1	🟩🟩🟩🟩🟩🟩🟩⬜🟩🟩
☐ O1839 A M H P	aceF	10	10	Dihydrolipoamide acetyltransferase	🟩🟩🟩🟩🟩🟩🟩🟩🟩🟩
☐ O3698 A M H P	aceK	6	6	Bifunctional isocitrate dehydrogenase kinase/phosphatase protein	⬜🟩🟩🟩🟩🟩🟩⬜⬜⬜
☐ O4748 A M H P	ach	3	3	CoA transferase	⬜⬜🟩🟩🟩⬜⬜⬜⬜⬜
☐ O2126 A M H P	acnA	10	10	Aconitate hydratase	🟩🟩🟩🟩🟩🟩🟩🟩🟩🟩
☐ O2145 A M H P	acnB	10	10	Bifunctional aconitate hydratase 2/2-methylisocitrate dehydratase	🟩🟩🟩🟩🟩🟩🟩🟩🟩🟩
☐ O3224 A M H P	acoK	7	8	Transcriptional regulator	⬜🟩🟩🟩🟩🟩🟩⬜🟩⬜
☐ O3685 A M H P	acoR	6	6	Transcriptional regulator AcoR	⬜🟩🟩🟩🟩🟩🟩⬜⬜⬜
☐ O2928 A M H P	acp	7	8	Acyl carrier protein	🟩🟩⬜⬜⬜🟩🟩🟩🟩🟩

Figure 4.3.: Screenshot of the MBGD ortholog cluster table. The table shows the distribution of genes within 10 *Xanthomonas* genomes. A dark green box in the right part of the table denotes the abundance of an orthologous gene. Information in the left part of the table comprises a cluster ID, the gene name, the number of species this cluster was found in, the number of genes in the cluster, and a functional description.

The Microbial Genomes Database (MBGD), established in 2003 (Uchiyama, 2003), is a database for finished microbial genomes that stores genomic information as well as precomputed gene clustering results. The database started in 2003 with only a few dozen genomes (exact numbers not available), with major updates in 2007 (Uchiyama, 2007) and 2010 (Uchiyama et al., 2010). At present, the MBGD

provides comparative analysis features for 1382 finished bacterial genomes. The main feature is a clustering of genes of selected genomes into homologous groups, resulting in tables of orthologous clusters. These tables (see Figure 4.3) represent the pan genome of the analyzed genome set, and the other genomic subsets like the core genome or singleton genes can be extracted from it, too.
In addition to the cluster information the MBGD provides further visualization features, e.g., multiple alignments between clustered genes, circular plots of analyzed genomes, or a map of the genomic context of clustered genes (see Figure 4.4). For newly sequenced, confidential genomes the MBGD provides a service called “My MBGD” which allows the user to upload a private genome. The genes of this genome are then clustered together with a set of selected genomes.

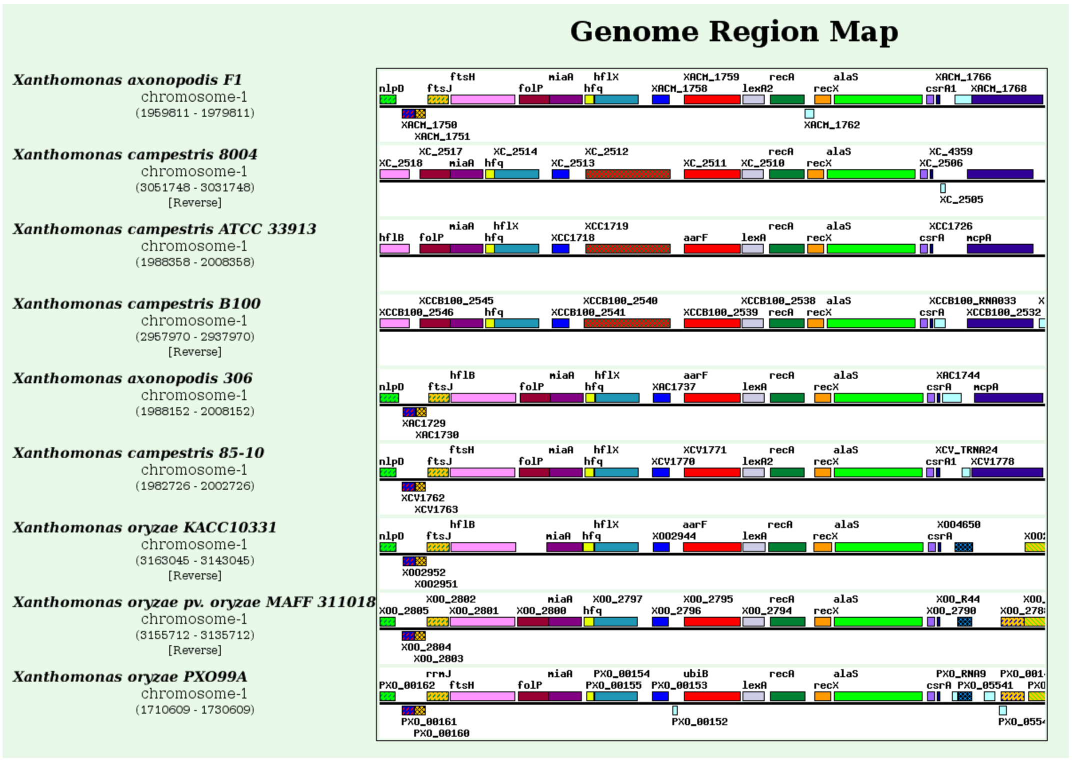

Genome Region Map
Xanthomonas axonopodis F1 chromosome-1 (1959811 - 1979811)
Xanthomonas campestris 8004 chromosome-1 (3051748 - 3031748) [Reverse]
Xanthomonas campestris ATCC 33913 chromosome-1 (1988358 - 2008358)
Xanthomonas campestris B100 chromosome-1 (2957970 - 2937970) [Reverse]
Xanthomonas axonopodis 306 chromosome-1 (1988152 - 2008152)
Xanthomonas campestris 85-10 chromosome-1 (1982726 - 2002726)
Xanthomonas oryzae KACC10331 chromosome-1 (3163045 - 3143045) [Reverse]
Xanthomonas oryzae pv. oryzae MAFF 311018 chromosome-1 (3155712 - 3135712) [Reverse]
Xanthomonas oryzae PXO99A chromosome-1 (1710609 - 1730609)
nlpD
ftsJ
ftsH
folP
miaA
hflX
hfq
recA
lexA
recX
alaS
csrA
hflB
aarF
mcpA

Figure 4.4.: Screenshot of the MBGD Genome Region Map. The map shows the genomic neighborhood of 9 *Xanthomonas* genomes centered around the red colored gene. One can see that there is a high conservation of the gene order. Noticeable differences are e.g. the addition of one gene left to the red gene in the three true *X. campestris* genomes 8004, ATCC 33913, and B100. *X. campestris* 85-10's status as *X. campestris* is debated as it more likely belongs to *X. axonopodis* (see Blom et al. (2009)).

The MBGD presents a comprehensive database of microbial genomes and offers valuable comparative features. Unfortunately the web interface is counterintuitive and the different functions are hard to find. A proper query mechanism to generate genomic subsets of interest is missing. All this together makes to MBGD interface quite inconvenient to use.

4.1.4. Sybil

The Sybil package (Riley et al., 2012) is a set of software tools developed at The Institute for Genomic Research (TIGR) and now hosted by the JCVI. Sybil provides tools to create a web server with several comparative analysis and visualization tools, based on numerous open source bioinformatics tools and databases, e.g., the Chado relational database schema (Mungall et al., 2007), the Apache Batik SVG toolkit[1], or Bioperl[2]. Sybil is still under development and thus it does not provide precomputed datasets (except one demo project) nor any facilities to analyze own data. Sybil only provides the source code for several scripts and tools to perform a comparative analysis and to create a web server presenting the result.
At the moment the following visualizations and analyses are supported:

- Protein/Cluster Search: This feature allows the user to search for genes or gene clusters within the analysed genomes.
- Pan genome analysis: Statistical analysis of the gene repertoire of a set of genomes, extrapolating the expected number of core and singleton genes and the expected pan genome size for large numbers of sequenced strains.
- Genomic comparative view: A view of orthologous genes in their genomic context, comparable to the one shown in Figure 4.4.
- Synteny Gradient Plot: Visualization of conserved genes as a gradient plot. The genes of a reference genome are colored according to a gradient from yellow to blue from start to end. The genes of the comparison genomes are colored according to the color of their orthologous gene in the reference (see Figure 4.5).
- Whole genome display: In this option linear genome plots of several genomes can be created, visualizing different features like genes, tRNAs, signal peptides, etc.
- Shared Cluster Matrix: This feature generates a matrix of the pairwise shared gene clusters between all genomes within a project.

Sybil provides useful insights into the analyzed genomes and high quality visualizations. Unfortunately, the Sybil package has a long list of prerequisites, is completely script based, and needs expertise in database and web server maintenance, so it is only suitable for experienced users with advanced technical skills and most likely too complex for the general biological user. In addition, the

[1]`http://xmlgraphics.apache.org/batik/`
[2]`http://www.bioperl.org/`

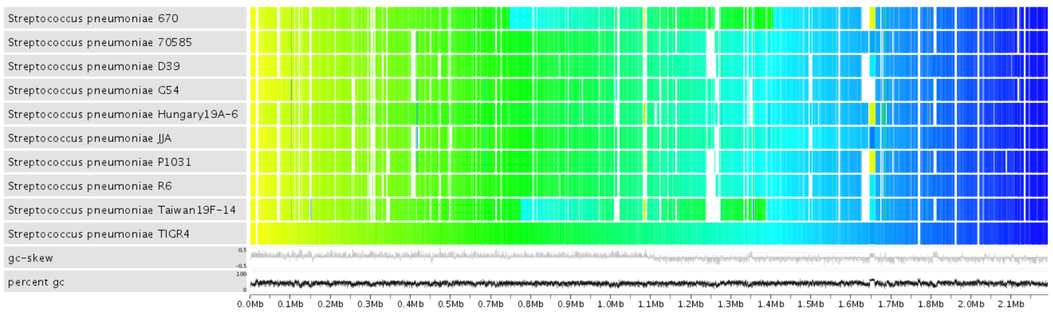


Figure 4.5.: Synteny gradient plot generated by the Sybil package. Genes of the reference genome *Streptococcus pneumoniae* TIGR 4 are colored in a gradient from yellow to blue. The orthologous genes of nine other *S. pneumoniae* genomes are colored according to this gradient. This allows to easily spot genomic rearrangements, e.g., large inversions in the middle of strains 670 and Taiwan19F-14. At the bottom of the gradient plot a GC plot and GC skew are provided.

development of Sybil regrettably seems not to be continued as the last release dates back to 30.10.2008. Only the *Streptococcus pneumoniae* demo project was maintained since then.

4.1.5. Panseq

Panseq (Laing et al., 2010) is a small, web-based tool for the analysis of conserved or novel sequence regions in a set of genomes. It uses BLAST (Altschul et al., 1990) and MUMmer (Kurtz et al., 2004) alignment results to estimate sequence fragments of at least 500bp that are conserved in one or more genomes. The Panseq web server provides all finished bacterial genomes of the NCBI databases for analysis and additionally allows the user to upload up to five own sequences. To start a panseq calculation the user has to select a set of genomes to compare and provide a valid email address. The results are calculated on server-side and the user is informed via email as soon as they are finished. The calculation for a dataset of 10 medium size bacterial genomes takes 3-5 hours, and results are provided as downloadable zip archive and are kept available by the web server for one week.
The results comprise FASTA files of unique regions found in the analyzed genomes, list of conserved sequences, called “core segments”, NEXUS files for the core genomes, the accessory genome, and for parts of the core genomes containing SNPs. These SNPs are also provided as a separate list. The Web server has on option to create graphical representations of the results, but this feature currently has no effect (as at 29.07.2012). Another feature of Panseq is the “Loci Selector”, which allows to use the SNP data calculated in the previously described analyses to generate a list of sequence loci that are best suited for Multi Locus Sequence Typing (Urwin and Maiden, 2003) (MLST), a universal technique widely used in

epidemiology to characterize bacteria. The Panseq web interface is easy-to-use, the provided analyses are beneficial and calculated in moderate time, but with only three main analysis features the capabilities of Panseq are limited. As Panseq is focused on whole genome alignments and works on conserved sequence fragments as main analysis level, comparisons one gene level are not featured by the software.

4.1.6. CoGE

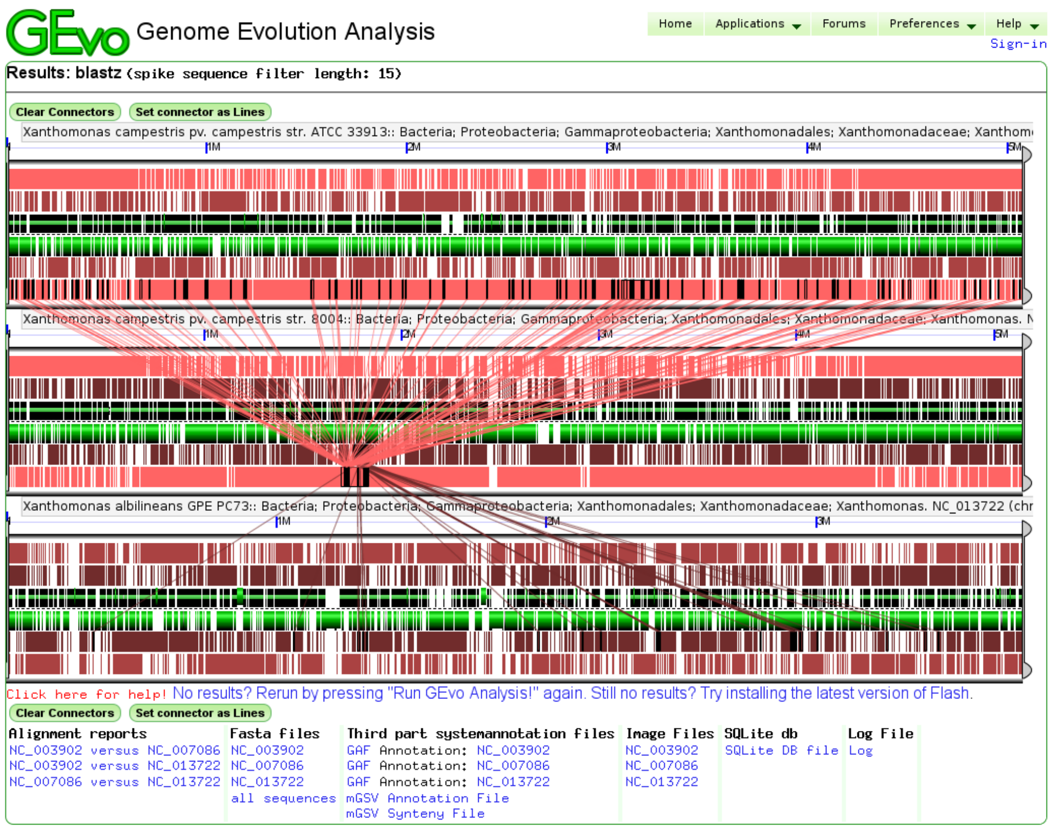
The CoGe (Comparative Genomics) system (Schnable and Lyons, 2011; Lyons and Freeling, 2008) is a web server based analysis platform for all genomes sequenced so far. The focus of the CoGe developers is on plant genomics, but their database features genomes from all taxonomic kingdoms. At present the CoGe database comprises 19,149 genomes of which approximately 7000 are bacterial, including published draft genomes. CoGe provides six main features:

- OrganismView: A search interface for organisms in the CoGe database. Selected organisms can be displayed in a comparison table, showing large scale genomic features like GC-content, number of replicons (or contigs in draft sequences), amino acid usage and codon usage statistics, gene count, etc.
- CoGeBlast: a BLAST interface to align sequences against any set of sequences included in the CoGe database.
- FeatView: Search interface to search for genomic features in the annotations of the stored genomes.
- SynMap: Generator for classical synteny dotplots.
- SynFind: This feature identifies syntenic regions across a set of selected genomes.
- GEvo: The GEvo (for Genome Evolution) interface provides a linear multi genome plot. The annotation of each genome is displayed, and conserved regions can be highlighted (see Figure 4.6).

CoGe is a valuable resource for comparative genomics alone due to the sheer amount of genomic data available. The analysis options concentrate on overall genomic features, an analysis of the genomic subsets or any other in depth analysis of the gene content is not supported by CoGe.

4.1.7. SEED

The SEED project (Overbeek et al., 2005) is an online service focused on the development of curated genomic data to support genome annotation as well as comparative analyses. The SEED project is based on so called "subsystems" and "FIG-fams". "Subsystems" are sets of functional roles that are thought to be related by



Results: blastz (spike sequence filter length: 15)

Xanthomonas campestris pv. campestris str. ATCC 33913:: Bacteria; Proteobacteria; Gammaproteobacteria; Xanthomonadales; Xanthomonadaceae; Xanthom

Xanthomonas campestris pv. campestris str. 8004:: Bacteria; Proteobacteria; Gammaproteobacteria; Xanthomonadales; Xanthomonadaceae; Xanthomonas. N

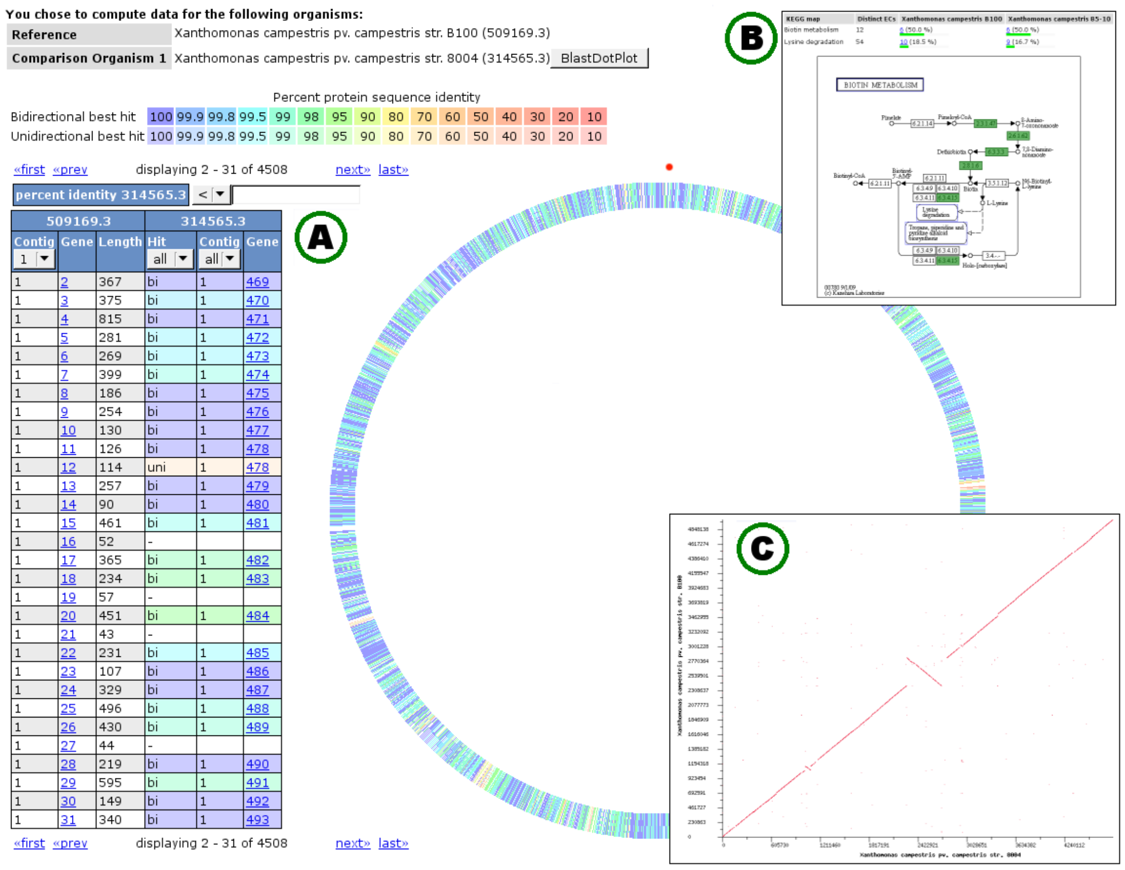
Xanthomonas albilineans GPE PC73:: Bacteria; Proteobacteria; Gammaproteobacteria; Xanthomonadales; Xanthomonadaceae; Xanthomonas. NC_013722 (chr

Click here for help! No results? Rerun by pressing "Run GEvo Analysis!" again. Still no results? Try installing the latest version of Flash.

Alignment reports	Fasta files	Third part systemannotation files	Image Files	SQLite db	Log File
NC_003902 versus NC_007086	NC_003902	GAF Annotation: NC_003902	NC_003902	SQLite DB file	Log
NC_003902 versus NC_013722	NC_007086	GAF Annotation: NC_007086	NC_007086		
NC_007086 versus NC_013722	NC_013722	GAF Annotation: NC_013722	NC_013722		
	all sequences	mGSV Annotation File			
		mGSV Synteny File			

Figure 4.6.: Gevo genome comparison plot generated by the CoGe web server. The plot shows the comparison of three *Xanthomonas* genomes. Annotated genes are highlighted in green. The different reddish colored blocks are conserved regions; by selecting these blocks they are connected to the respective homologous blocks in the other genomes.

an annotator. These sets can be the functional roles of a metabolic pathway, a complex (like the ribosome), or a certain class of proteins. Based on these subsystems and on the correspondence between genes the "Fellowship for the Interpretation of Genomes" (FIG) generates protein families called "FIGfams". Coding sequences that are classified into the same FIGfam are believed to have the same function. Subsystems and FIGfams are both manually curated and thus form a reliable basis for gene classifications based on bidirectional best BLAST hits (BBHs, see Section 4.2.1.1). Via the SEED-Viewer users have read-only access to all curated data which currently comprises 58 Archaea, 921 Bacteria, 562 Eukaryota, 1254 Plasmids, and 1713 Viruses (as at 02.08.2012). For these curated genomes some comparative features are available. A user can perform pairwise genome comparison either on functional level, where the annotated FIGfams of two genomes are compared, or on sequence level where an all against all BLAST comparison of the genes of two genomes is performed and displayed as a circular genome plot (Figure 4.7.A). Furthermore, the SEED-Viewer can generate synteny plots of two genomes (Figure



You chose to compute data for the following organisms:

Reference	Xanthomonas campestris pv. campestris str. B100 (509169.3)
Comparison Organism 1	Xanthomonas campestris pv. campestris str. 8004 (314565.3) BlastDotPlot

Percent protein sequence identity

Bidirectional best hit	100	99.9	99.8	99.5	99	98	95	90	80	70	60	50	40	30	20	10
Unidirectional best hit	100	99.9	99.8	99.5	99	98	95	90	80	70	60	50	40	30	20	10

«first «prev displaying 2 - 31 of 4508 next» last»

percent identity 314565.3 <

509169.3			314565.3		
Contig 1	Gene	Length	Hit all	Contig all	Gene
1	2	367	bi	1	469
1	3	375	bi	1	470
1	4	815	bi	1	471
1	5	281	bi	1	472
1	6	269	bi	1	473
1	7	399	bi	1	474
1	8	186	bi	1	475
1	9	254	bi	1	476
1	10	130	bi	1	477
1	11	126	bi	1	478
1	12	114	uni	1	478
1	13	257	bi	1	479
1	14	90	bi	1	480
1	15	461	bi	1	481
1	16	52	-		
1	17	365	bi	1	482
1	18	234	bi	1	483
1	19	57	-		
1	20	451	bi	1	484
1	21	43	-		
1	22	231	bi	1	485
1	23	107	bi	1	486
1	24	329	bi	1	487
1	25	496	bi	1	488
1	26	430	bi	1	489
1	27	44	-		
1	28	219	bi	1	490
1	29	595	bi	1	491
1	30	149	bi	1	492
1	31	340	bi	1	493

«first «prev displaying 2 - 31 of 4508 next» last»

Figure 4.7.: Collage of SEED-Viewer screenshots. (A) The pairwise sequence based genome comparison. A circular plot as well as a tabular list provide information about BBHs between genes from two genomes. (B) In the KEGG metabolic pathway analysis genes from a certain pathway that are found in a selected set of genomes are highlighted, in this case 6 genes found in the Biotin biosynthesis pathway in two *Xanthomonas* strains. (C) A synteny plot of homologous genes in two *Xanthomonas* genomes.

4.7.B) and can generate KEGG pathway maps where genes are highlighted that are found in all of up to five genome from the SEED database (Figure 4.7.C).

While the SEED project started as a resource for comparative genome analysis, the focus has shifted more and more towards automatic annotation of genomes using the RAST server (Aziz et al., 2008). The comparative features available via the website are limited to the described basic features and only pairwise genome comparisons are possible. However, for expert users the SEED resources and datasets are available for download and can be used to create own analysis tools.

4.1.8. STRING

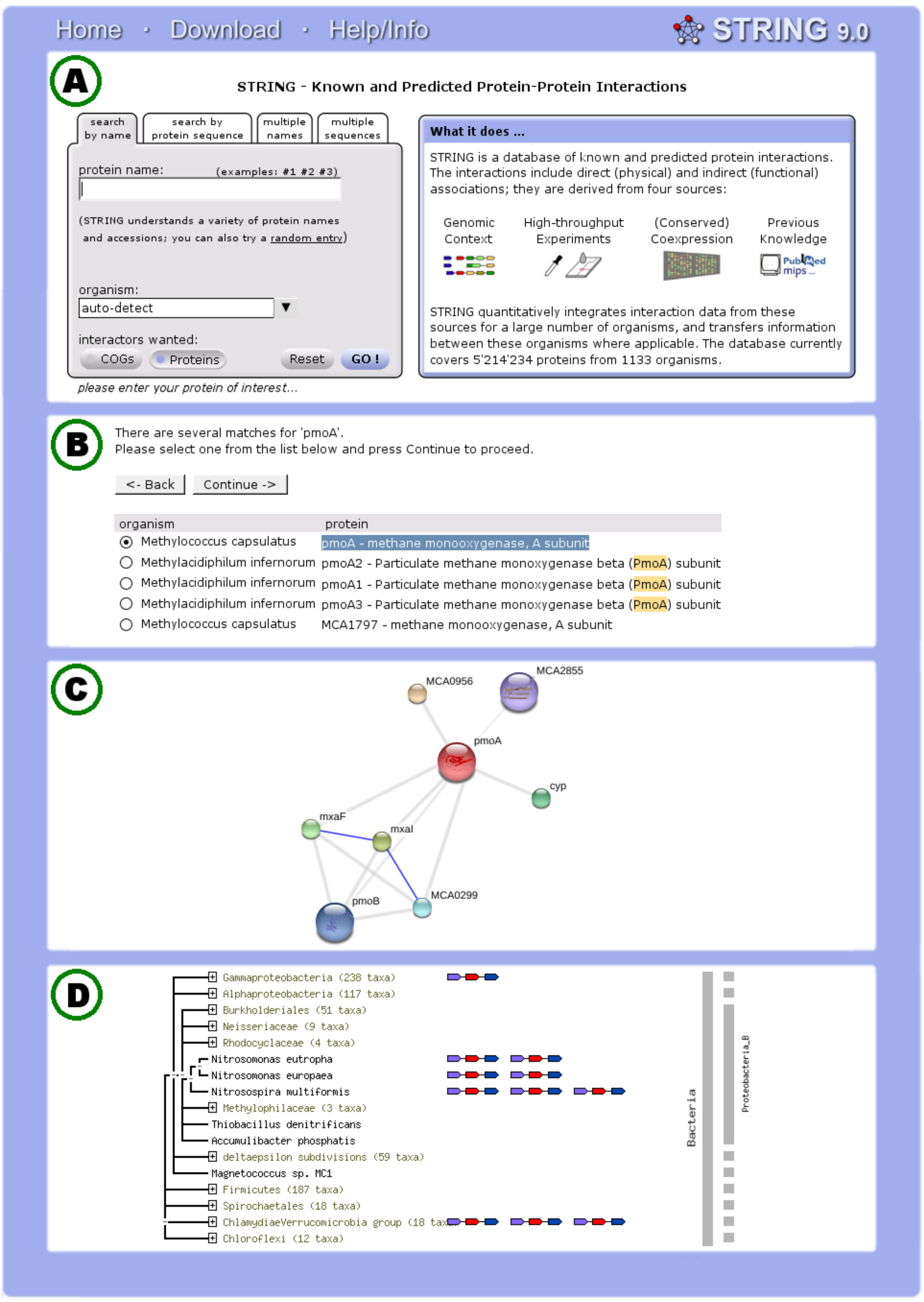
STRING is a public database (Von Mering et al., 2003) of predicted functional associations between genes based on genomic features. It tries to infer functional links between proteins from genomic associations between the genes encoding them, e.g., species coverage, a nearby location on the genome, or the involvement in gene fusion events. STRING was established in 2000 with 59,416 genes from all organisms sequenced at this time (number not given) (Snel et al., 2000). Since this time the database is constantly updated with the latest update to version 9.0 in 2011 (Szklarczyk et al., 2011). In this current version the STRING database hosts information about 5,214,234 proteins from 1,133 organisms. STRING takes homology information from the SIMAP database (Rattei et al., 2010) of pre-calculated protein sequence similarities. STRING allows a user to query a protein of interest in an initial search mask (Figure 4.8.A). For this protein the appearances in all genomes within the database are displayed and a reference organism can be selected (Figure 4.8.B). For the selected organism a graph of functional partner proteins, which show predicted high association scores to the query, is generated (Figure 4.8.C). For the functional partners further analyses are provided, e.g., a phylogenetic tree where occurrences of those partner proteins in close proximity in all stored organisms (most likely operon structures) is highlighted (Figure 4.8.D).

The analyses provided by STRING are highly valuable for scientists interested in only one or a small set of proteins of interests. The web interface is easy to use and fast and allows instant access to a huge collection of data about functionally related proteins. A drawback of the system is that a user has to know his proteins of interest already. STRING does not support any whole genome scale analyses which could help to identify such genes of interest. Furthermore the database is limited to the publicly available organisms included with each update. Indeed, the user can compare an uploaded protein sequence to the database, but a comparison of several unpublished sequences to each other is not possible.

Conclusion

While all presented approaches are valuable contributions to the field, they all have their limitations. VISTA and Panseq feature whole genome alignments and reliably identify conserved regions, but they provide no analyses of the gene content of a set of genomes. Sybil and CoGe provide gene based analysis and descriptive graphical representations, but they do not address the classical genomic subsets. These subsets are featured by the CMR and MBGD databases, but the CMR does not allow to upload own data for analysis, and the MBGD user interface is slow and cumbersome. STRING does not provide complete genome comparisons, but provides analyses of proteins of interest.

If one has a closer look on the features provided by the genome comparison tools some features can be identified to be very popular. All tools except SEED (which is based on pairwise comparisons) provide multi-genome comparisons, all tools



STRING - Known and Predicted Protein-Protein Interactions

protein name: (examples: #1 #2 #3)

(STRING understands a variety of protein names and accessions; you can also try a random entry)

organism: auto-detect

interactors wanted: COGs [✓ Proteins]

please enter your protein of interest...

What it does ...

STRING is a database of known and predicted protein interactions. The interactions include direct (physical) and indirect (functional) associations; they are derived from four sources:

STRING quantitatively integrates interaction data from these sources for a large number of organisms, and transfers information between these organisms where applicable. The database currently covers 5'214'234 proteins from 1133 organisms.

There are several matches for 'pmoA'.
Please select one from the list below and press Continue to proceed.

organism	protein
[✓] Methylococcus capsulatus	pmoA - methane monooxygenase, A subunit
Methylacidiphilum infernorum	pmoA2 - Particulate methane monoxygenase beta (PmoA) subunit
Methylacidiphilum infernorum	pmoA1 - Particulate methane monoxygenase beta (PmoA) subunit
Methylacidiphilum infernorum	pmoA3 - Particulate methane monoxygenase beta (PmoA) subunit
Methylococcus capsulatus	MCA1797 - methane monooxygenase, A subunit

Figure 4.8.: Collage of STRING screenshots. (A) The STRING start screen with the query field. (B) A list of organisms in which the requested protein was found. A reference organism has to be selected for the subsequent analyses. (C) An interactive graph of predicted functional partners of the query protein. (D) Phylogenetic tree with highlighted occurrences of the query protein and co-located functional partners within the genomes of the STRING database.

feature multiple alignments. Synteny plots are also featured by nearly all tools with exception of Panseq and MBGD. Furthermore, most tools provide circular or linear genome plots and an interface to align orthologous gene sets. While VISTA and Panseq work with alignments of complete genome sequences, all other approaches use the gene sequences as basis for their analyses.
There are also analysis features that are found less frequently, e.g., the genomic subsets can be calculated by only three tools (CMR, Panseq, MBGD), and only one tool (Sybil) can provide a statistical analysis of the pan genome development. Three of the tools seem to be not updated any longer (CMR, Sybil, Panseq). Furthermore, only VISTA, CMR, and MBGD allow the upload and analysis of unpublished private genomes. None of the mentioned approaches offers any functionality for collaborative work of scientist at different locations on confidential data, and higher level comparisons of sets of genomes are also not featured by any of the presented applications. Surprisingly, the only tool that provides phylogenetic analyses is STRING, and STRING provides only phylogenies for sets of orthologous genes, not for complete organisms.
Based on the analysis of features of the competing tools, EDGAR was designed to combine all popular analysis features in comparative genomics with novel analysis options like metacontigs for higher level comparison (see Section 4.2.2.4) or whole genome phylogeny. Furthermore, a project based infrastructure should allow collaborative work on confidential data via a web interface.

Another crucial aspect in the analysis of groups of related genomes is the selection of an appropriate similarity cutoff to assign orthologous genes. As already shown in Table 2.2 there is no common agreement on orthology cutoffs, and the same picture arises when comparing the existing software introduced in this chapter. MBGD for example allows the user to choose from 16 parameters like evalue, alignment score, alignment coverage in different combinations and modes, e.g., if only best hits should be used or if unidirectional or bidirectional hits should be considered. The CMR offers three parameters to choose from: Minimum percent similarity (where similarity is the percentage of positively scored amino acids in the global alignment of two proteins), minimum percent identity, or maximum p-value. Panseq lets the user define the parameters for the MUMmer alignment used as data basis for the subsequent calculations, while CoGe offers up to six algorithms to the user, with parameter choices for each of them: BlastN & tBlastX (Altschul et al., 1990), LAstZ (Harris, 2007), Chaos (Brudno et al., 2004), GenomeThreader (Jones, 1999), or Lagan (Brudno et al., 2003). Sybil provides precomputed datasets, and VISTA does not provide gene based analyses, therefore no cutoff is needed.
As a matter of fact, all approaches try to provide appropriate default cutoffs, but naturally a cutoff adapted to the compared data is preferable. A user that does not want to rely on the default parameters has to find the parameters best suited for the genomes he wants to compare by trial and error. An automatic assessment of an adequate homology criterion would be a great easement of comparative analyses.

4.2. EDGAR: Efficient Database framework for comparative Genome Analysis using BLAST score Ratios

As described in the previous section existing software for comparative genomics has several shortcomings, and none of the presented approaches fulfills the requirements defined in Section 3.1. Thus, EDGAR was developed as an easy-to-use software for gene content comparisons and phylogenetic analyses of multiple microbial genomes based on an automatically adjusted generic homology/orthology cutoff tailored for the analyzed genomes. In this section the design and implementation of EDGAR will be described in detail.

4.2.1. Design and methods

The system design of EDGAR follows the classical three-tiered architecture model with a database layer, business logic layer, and presentation layer. The data back-end is realized using a MySQL[3] relational database management system (RDBMS). The business logic layer is implemented mainly in Perl[4], using the DBI package for database access, Bioperl[5] for various parsers, and the in-house "General Project Management System" (GPMS) for user management and access control. The presentation layer is created as a web based user interface implemented in Perl CGI with some additional JavaScript. For complex mathematical operations like curve fitting the open source statistical programming language "R"[6] is used.

4.2.1.1. Orthology estimation

It has been shown in Section 4.1.8 that an adequate orthology criterion is of vital importance for subsequent calculations on gene sets. Following the original definition of orthology by Fitch (see Chapter 2.4.2), two genes are orthologs if they share a common ancestor and diverged through a speciation event. Unfortunately a true orthologous relationship between two genes can not proven computationally, only a resulting similarity between orthologous genes can be observed. Thus, as the assignment of orthologous genes is mainly used to propagate a functional description from one gene to another, the term "ortholog" is often used to describe two genes with highly similar sequences and an assumed common function. The terms "ortholog" and "orthologous" will be used with this meaning in the following if not indicated otherwise. A common way to identify orthologous genes is to use results of the alignment tool BLAST, i.e., the version for protein alignment BLASTP. Sometimes simple unidirectional BLAST hits are used, but a far more common approach uses

[3]`http://www.mysql.com/`
[4]`http://www.perl.org/`
[5]`http://www.bioperl.org/`
[6]`www.r-project.org`

bidirectional best blast hits (BBHs), meaning that two genes have reciprocal blast hits against each other and both hits are the best hits against the respective target genome.
In the last two decades several more sophisticated approaches for orthology estimation have been developed:

- **Clusters of Orthologous Genes (COG)**: The COG database, established in 1997 (Tatusov et al., 1997), is the presumably oldest orthology database and was considered the gold standard for many years. The ortholog clusters in the COG database are based on an all-against-all comparison of proteins using BLAST. The ortholog clusters are defined by triangles of BBHs, called mutually consistent, genome-specific best hits (BeTs) by the authors. Such triangles with mutual sides are iteratively merged to form bigger clusters, and the resulting clusters are manually curated. Unfortunately the last update of the COG database dates back to September of 2003, thus the database has to be considered as no longer maintained.

- **Inparanoid**: The orthology estimation tool Inparanoid (O'Brien et al., 2005) also uses pairwise BLAST similarity scores as basis for the calculations. Bidirectional hits between two genomes are marked as potential orthologs if they exceed a certain cutoff (default: $bitscore \geq 50, overlap \geq 50\%$). The identifying feature of Inparanoid is the resolution of paralogous genes by marking in-species-hits as additional orthologs, so called "in-paralogs".

- **OrthoMCL**: The MCL algorithm used in OrthoMCL (Li et al., 2003) like most orthology tools starts with an all-against-all BLAST comparison of proteins. Reciprocal similar pairs are used to mark potential orthologs or paralogs. Based on the pair results, a similarity matrix is calculated, followed by Markov clustering (van Dongen, 2000), which results in the orthologous groups.

- **RoundUp**: The RoundUp method, recently published in its second version (DeLuca et al., 2012), is based on the reciprocal smallest distance (RSD) algorithm by Wall *et al.* (Wall et al., 2003). This algorithm collects all hits from one gene against a genome exceeding a given divergence or e-value cutoff. Each hit of this collection is aligned with the query in a multiple alignment, and each sequence with more than 80% alignment coverage is used to calculate a maximum likelihood amino acid substitution matrix with the PAML (Yang, 2007) software. The sequence yielding the smallest phylogenetic distance is an orthologous candidate, and the same procedure is repeated in the opposite direction to get a reciprocal result.

- **EggNOG**: The EggNOG database (Powell et al., 2012) - "evolutionary genealogy of genes: Non-supervised Orthologous Groups" - is based on the old COG database. Proteins are assigned to their respective Cluster of Orthologous Genes (COG) using a best alignment hit approach. Proteins that can not be assigned are assembled into non-supervised orthologous groups (NOGs). The formation of NOGs is based on all-against-all Smith-Waterman-alignments (Smith and Waterman, 1981). Duplicated sequences are marked as in-paralogous groups (see Inparanoid). The group formation is realized by iterative joining of triangles of BBHs comparable to the COG approach.

- **OMA**: OMA (Orthologous MAtrix) is a database that identifies orthologs among publicly available, complete genomes (Dessimoz et al., 2005; Altenhoff et al., 2011). The ortholog matrix is constructed from all-against-all Smith-Waterman protein alignments. With the alignment results so called "stable pairs" are identified, verified, and checked for potential paralogous genes. In a last step, cliques of stable pairs are clustered as groups of orthologs.

In 2006 Hulsen *et al.* benchmarked some of these approaches and compared them to the BBH approach (Hulsen et al., 2006). The result was that the simple BBH approach shows a performance comparable to or better than compared approaches (in this study Inparanoid, COG and OrthoMCL). This was confirmed in a second review of orthology estimation methods in 2009. Altenhoff and Dessimoz (2009) compared eleven algorithms and concluded that BBHs are coequal in orthology estimation for the classical phylogenetic orthology by Fitch as well as for the pragmatic functional definition:
"Another surprise is the good overall performance of the simple BBH approach. Although the method is restricted to 1:1 orthologs, the derived relations show good comparative accuracy in terms of Fitch's definition. Orthologs predicted by BBH also show close functional relatedness. This result probably explains why many people use ad-hoc BBH implementations for their analyses rather than a more sophisticated orthology method." (Altenhoff and Dessimoz, 2009, p. 7)
The drawback of BBHs is mentioned in the excerpt, BBHs only allow for 1:1 relationships and thus have problems in orthology assignment if identical paralogous genes are present within a genome. Overall, according to Hulsen *et al.* the more sophisticated approaches may have a higher sensitivity in the detection of orthologs, but BBHs have a higher specificity and should therefore be preferred if specificity is prioritized: "If selectivity (having as few as possible false positives) is more important than sensitivity (having as many as possible true positives) and having only one ortholog per protein is sufficient, the best bidirectional hit approach should give the best results." (Hulsen et al., 2006, p. 6)
The main feature of the EDGAR software is the calculation of genomic subsets, consequently, the assignment of orthologs should be as reliable as possible. Thus, according to Hulsen and Altenhoff, the focus is on specificity and the BBH approach is preferable. Furthermore, the BBH calculation is a fast, straightforward way and can be easily parallelized for calculation on a compute cluster to speed up the processing of the huge amounts of data that can be expected in comparative genomics. The drawback of problematic paralogs is compensated by these advantages. Hence, EDGAR uses BBHs as method for orthology estimation. As discussed in Section

2.4.2 and illustrated in Table 2.2 the cutoff values used for BLAST based orthology estimation are arbitrarily selected. Thus, to find a mechanism for an appropriate automatic thresholding procedure was the next task in the design process.

4.2.1.2. Orthology threshold: BLAST score ratio values (SRVs)

For the high-throughput computation of comparative data intended for EDGAR it is crucial to rely on a generic orthology criterion consistent within the analyzed genome set. For this purpose, EDGAR deploys the so called BLAST Score Ratio Values (SRVs) suggested by Lerat et al. (2003). Instead of using the absolute bit scores provided by the BLAST algorithm, the SRV method uses a normalization approach by relating all bit scores of a protein to the maximum bit score that can be achieved by this protein sequence. The maximum bit score is defined as the BLAST bit score of an alignment of the sequence against itself, as such a BLAST hit always has 100% identity over 100% of the query sequence length and thus gives the maximum bit score possible. So, a SRV is defined as the ratio ($Observed\ score/Maximum\ score$), thus giving a value in the range $[0, 1]$. Finally this value is multiplied by 100 and rounded to gain discrete percentage values.
Lerat et al. (2003) observed that the distribution of bit scores from a all-against-all comparison of the genes of two related genomes shows a bimodal pattern. The first peak at low similarity values was constant among comparisons, thus most probably representing random matches, while the second peak at higher similarities represents true homologous sequences (Lerat et al., 2003).
To remove all nonspecific hits, a cutoff value has to be identified that defines the boundaries of the first peak and thus the set of random hits that should be removed. Lerat *et al.* analyzed the class of γ-Proteobacteria and chose a fix SRV threshold of 30% for orthology estimation based on manual inspection of the bimodal SRV distribution. For EDGAR we wanted to develop an automatic, unsupervised estimation to identify an appropriate threshold.

To find a cutoff for the comparison of two genomes we use a statistical approach based on the beta distribution. The number of BLAST hits with a given SRV is summed up and represented in a histogram for all SRV values in the range $[0, 100]$. As the beta distribution is defined in an interval $[0, 1]$ the SRV histogram is normalized to this interval. Due to the fact that we want to remove the first peak, we only use all values in the left part of the SRV histogram with a normalized SRV < 0.4 and calculate a beta distribution from the mean and standard deviation of the observed SRVs within this interval. The density function of this beta distribution is calculated, and the 97% quantile of this density function is taken as cutoff that defines the end of the first peak. The value of 97% was chosen based on manual inspection of hundreds of SRV distributions. The typically used values of 95% (single standard deviation) or 99% (double standard deviation) were discarded as 95% gave cutoff values that seemed slightly too low and 99% gave values slightly

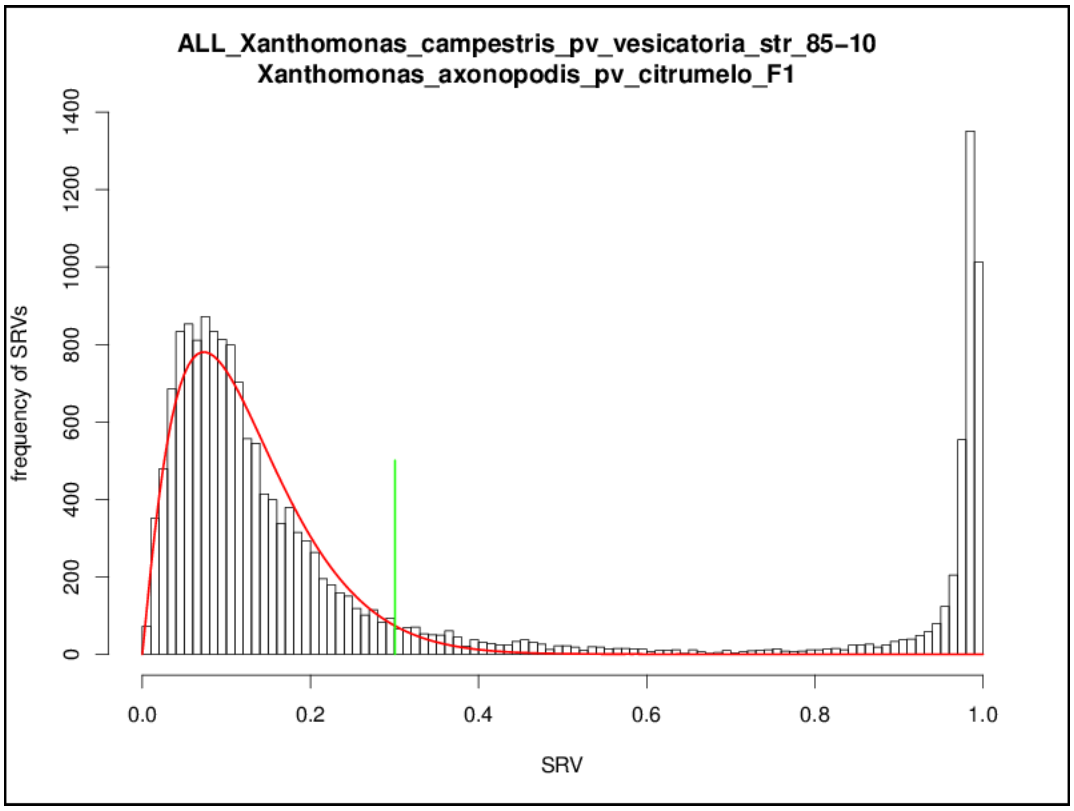


Figure 4.9.: Histogram of the SRVs resulting from the comparison of two *Xanthomonas* genomes. The distribution of the SRVs is clearly bimodal with one peak at 8% and one peak at 99%. The red curve shows a beta distribution calculated for all values $[0, 0.4]$, and the green line shows the 97% quantile of this distribution which is used as cutoff value.

too high. A normalized SRV distribution with the fitted beta distribution density function is shown in Figure 4.9.

This procedure is repeated for all possible combinations of genomes, resulting in n^2 combinations for a set of n genomes. These n^2 cutoffs could be used as threshold for a multi genome comparison by using a distinct cutoff for every comparison of two genomes. This would result in a higher accuracy as for every pairwise comparison a tailored cutoff would be used, but at the same time the basis of comparison for the overall calculations would be lost, e.g, for a comparison of a number of genomes the singletons of each genome would be calculated with different cutoffs. Hence, as distinct cutoffs would give hardly comparable results, the aim of the EDGAR threshold estimation mechanism was to calculate one master cutoff for all genome comparisons. To obtain this the n^2 cutoffs are plotted in a second histogram, which in most cases shows a normal distribution. To get a comparable general threshold for all subsequent calculations, the peak of this histogram is determined and used as the master cutoff for the BBH computation between the compared genomes (see Figure 4.10).

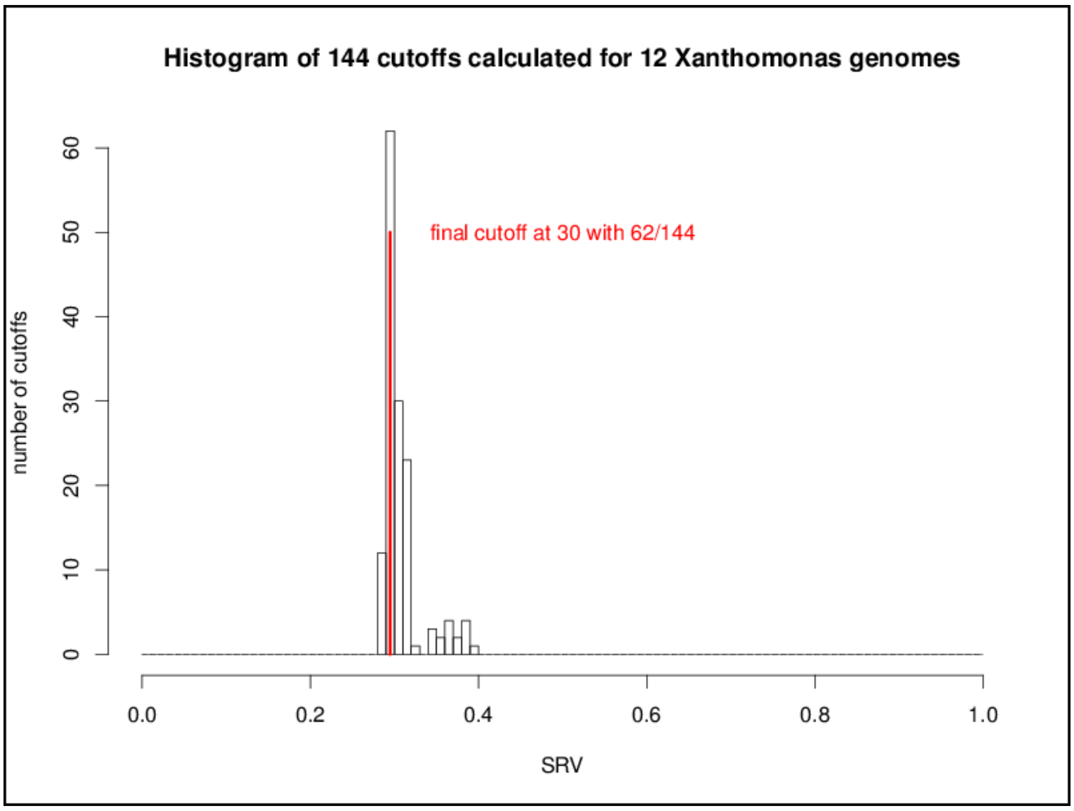


Figure 4.10.: Histogram of the 144 orthology cutoffs calculated independently for 12 *Xanthomonas* genomes. A clear peak at 30 can be observed and is used as final cutoff for all subsequent analyses of these 12 genomes.

The orthology cutoff generated by this approach is quite strict, as all low quality BLAST hits are filtered out. In this way it supports the desired high specificity of the orthology estimation. In one of the biggest EDGAR projects, a private project with 42 organisms from the genus *Erwinia*, the calculated cutoff is 31. Using an initial evalue cutoff of $1e^{-5}$ about 40 million BLAST results are parsed, of which only ~7.3 million or 18.25% pass the SRV filter. The mean percent identity of all hits of 73.5 (median 79.0) and the mean evalue of $6.6e^{-9}$ (median $6.0e^{-103}$) confirm the strictness of the filter. As a consequence of that strict threshold, orthologs found by EDGAR, especially when conserved among numerous genomes like the core genes, could be considered real orthologs, but some potential orthologs might be lost.
To address the problem of paralogous genes with identical protein sequences these genes are filtered out during the project creation. Only one of the paralogous genes is stored as representative gene for the paralog set. This approach circumvents the bias introduced by the appearance of paralogous genes in different order during the calculation. The filtered identical paralogs are not lost but stored in a special data structure in the EDGAR database.

Limitations

In some cases the SRV distribution does not show the expected bimodal shape. This is mostly the case when there is a high variation within the genomes of a genus. One example is the genus *Corynebacterium*, where the genomes are very diverse, leading to an SRV distribution with only one peak for low similarities and a broad plateau of medium scores with a decay at the highest scores. This distribution pattern is not problematic for the cutoff calculation method as the beta distribution still fits the peak at low SRV values very well (see Figure 4.11.A).
Another example is the genus *Buchnera* where the distribution is bimodal, but with the second, higher scoring peak at SRVs around 50%. The accumulation of hits at mediocre scores is problematic as results from the second higher scoring peak contribute to the beta distribution calculated in the interval $[0, 0.4]$ and thereby cause a too high cutoff (see Figure 4.11.B). A third example is the "genus" *Candidatus*. *Candidatus* is an interim taxonomic status assigned to bacterial organisms that can not be cultivated. The members of this interim class are so diverse that in extreme cases there is the low scoring peak alone without sufficiently good enough BLAST results (see Figure 4.11.C). Even in problematic genera the majority of cutoffs is calculated properly and outliers can be ignored in the majority of cases. But if more than 50% of the calculated cutoffs are > 0.4, which is considered wrong as it is outside the boundaries of the beta distribution, a default value is used. This default value was obtained by calculating the cutoff histograms for 130 genera including 1449 genomes and 37622 pairwise genome comparisons as described in Section 4.2.1.2. The results were normalized with regard to the number of genomes within a genus. The resulting cutoff histogram (see Figure 4.11.D) shows a peak at 0.3, so the respective SRV of 30 is used as default value if too many spurious cutoffs are found for a genus. In addition, this value is in accordance with the value proposed by Tettelin et al. (2005).

4.2.1.3. User management and access control

The web-based user interface of EDGAR implies a need for a reliable mechanism to control access to the analysis results as well as to the stored datasets. This is particularly important as EDGAR is intended as an analysis system for unpublished and therefore confidential data. Furthermore, defined roles for each user are desirable as they allow to restrict or extend the functionality available to particular users in individual projects. An existing system to realize the required user management is the General Project Management System (GPMS). GPMS is a proprietary user management system developed and used in several applications of the Bioinformatics Resource Facility (BRF) at Bielefeld University's Center for Biotechnology (CeBiTec), e.g., GenDB (Meyer et al., 2003), EMMA (Dondrup et al., 2009), or MeltDB (Neuweger et al., 2008). The system is specifically designed for user management in distributed projects and is based on roles individually assigned to a user for a certain project of a software. The roles are defined for each software supported

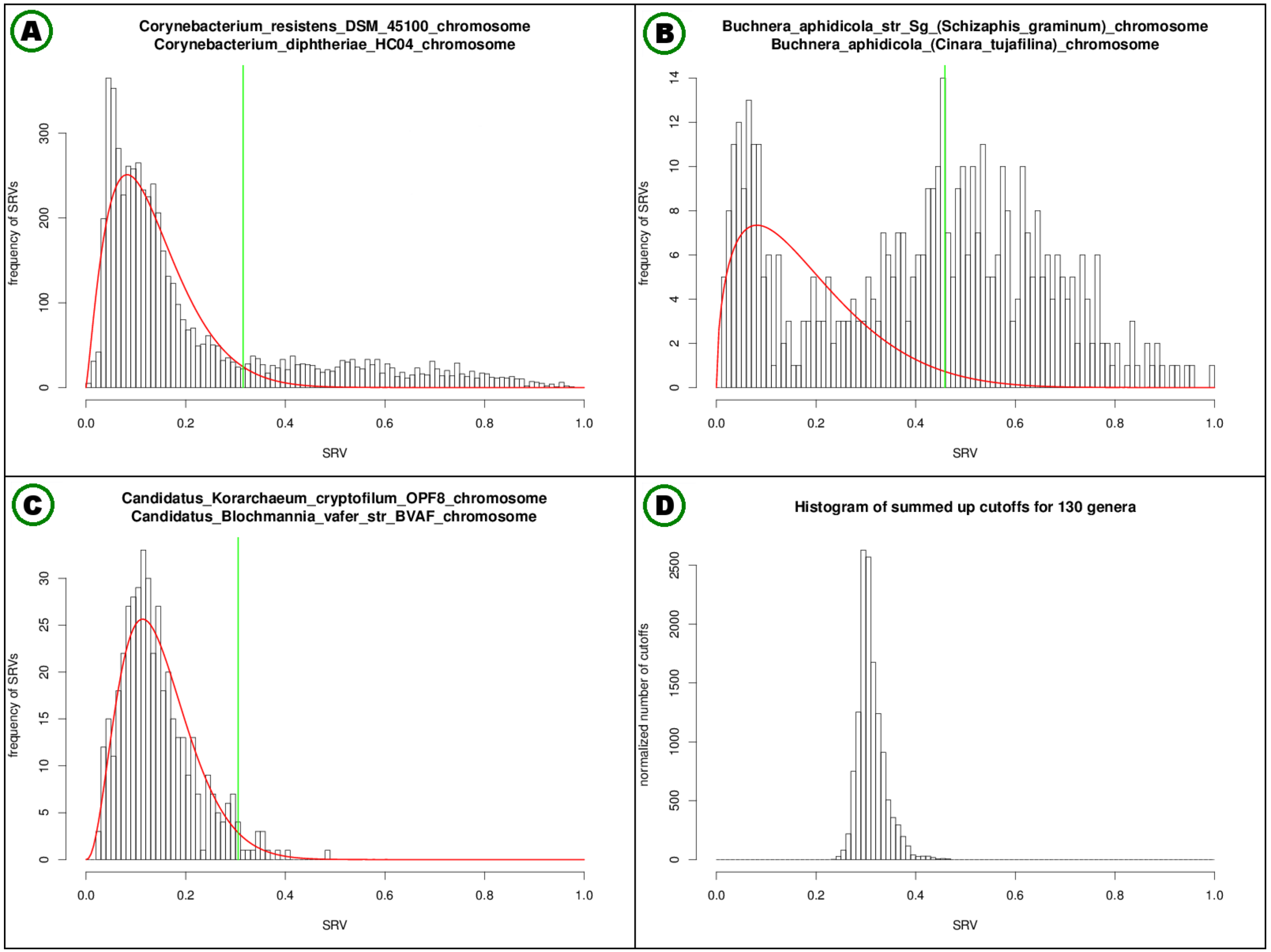


Figure 4.11.: Overview of problematic SRV histograms. (A) SRV histogram for two *Corynebacteria*, the second peak is missing. (B) In the genus *Buchnera* the second peak in some cases occurs between 0.4 and 0.6, thus interfering with the beta distribution. (C) If the taxonomic distance is too big like in this example of two *Candidatus* genomes, only an insufficient number of BLAST results remains. (D) Histogram of the combined cutoffs of 130 genera. Cutoff histograms of every genus were normalized to percent values and then summed up. The peak of this combined histogram is at a value of 0.3, thus the respective SRV value of 30 is used as default cutoff for the cutoff estimation.

by the GPMS system, and for each role a certain set of rights is specified. EDGAR uses four distinct roles with different privileges:

- `Guest:` A `Guest` user has basic access privileges for an EDGAR project. He can access a project and view and analyze stored data associated with a project. As EDGAR analyses rely on precomputed datasets the read-only access of the `Guest` user is sufficient for nearly all analysis features of EDGAR, thus this role is the standard role for EDGAR users.
- `Expert:` A user with the `Expert` role has extended privileges, he may insert or update data in the database. Initially designed as a read-only system, EDGAR was recently extended with an interface that allows users to define groups of genomes (see Chapter 4.2.2.4) and thus requires write privileges. This role allows experienced external EDGAR users the use of this features.
- `Developer:` A user with the `Developer` role may, like an `Expert` user, insert or update data in the database. Furthermore he has the right to change the database layout. This role is, as the name suggests, designed for software developers or project maintainers.
- `Chief:` The `Chief` of a project has all rights of a `Developer` plus additional administrative rights. He is allowed to create new accounts for the GPMS system and is allowed to add GPMS users as members to the project he administrates.

The role-based access control provides a high level of flexibility and significantly simplifies the management of EDGAR projects. The defined user rights and roles guarantee the efficient maintenance of the existing EDGAR system and at the same time allow for easy extension by adding new roles if future developments change the requirements.

4.2.1.4. Data model

As described in the last chapter EDGAR uses a MySQL database backend for data storage and retrieval. The database stores sequence information as well as the results of the all-against-all BLAST searches described in chapters 4.2.1.1 and 4.2.1.2. The data model comprises three hierarchically organized sequence storage tables:

- `Organism:` The `organism` is the superordinate sequence organization table and stores the name of an analyzed organism together with a distinct ID.
- `Contig:` In the table `contig` the single replicons of the organisms are stored. A `contig` stores a replicon name and a distinct ID, as well, but furthermore references the `organism` it belongs to and stores the sequence length and genetic code of the replicon.

- `CDS`: In the table `CDS` the information about the coding sequences (CDS) of the integrated genomes are stored. `CDSs` have a unique name and ID and reference the `contig` they belong to. Additionally the start and stop position of the `CDS`, the sequence, the upstream sequence, and a functional description are stored.

The fundamental information needed for all EDGAR calculations are the BLAST results from the all-against-all comparison of all stored `CDS` filtered according to the described SRV-based cutoff. This information is stored in the table `hits`. In this table the query and hit sequences are stored as references to the according `CDS` entries, and the source contigs of the respective `CDS` are stored as references to the `contig` table. As information about the BLAST hit quality the bit score, the evalue, the percent identity value, and the calculated SRV are saved.
As described in 4.2.1.2 the information about paralogous genes has to be stored in the database. For this purpose the table `paralogs` stores the name and functional description of filtered genes and references the `CDS` that was used as representative during the calculations. For more in depth analyses EDGAR also provides data tables for higher level structures. There are two tables to store arbitrary groups of contigs. The first table `contig_group` stores a name and an ID for every `contig_group` defined by the user. As one `contig_group` comprises several `contigs` and at the same time one `contig` may be assigned to several `contig_groups`, there is a $n : m$ relationship that requires a mapping table. This table `groupmap` stores ID pairs referencing `contig` as well as `contig_group`. This `contig_groups` are simple set unions of the genes of the included `contigs` and are best suited to group replicons of one organisms, and thus may include some or a large number of orthologous genes within a group which causes redundancy.
For cases where this redundancy is troublesome a second higher-ranking data structure to store non-redundant gene sets for multiple `contigs` was created. For a user defined set of `contigs` either the pan or the core genome is calculated and a non-redundant set of genes is extracted, the genes from the reference genome for the core genome or the first abundant gene of every element of the pan genome. A table `metacontig` stores an identifier, a name and the type of the `metacontig` (either pan or core genome based). A mapping table `meta_map` stores the assignment of the non-redundant set of `CDS` to a `metacontig` by referencing the ID fields of the respective tables. The complete database schema of EDGAR is displayed in Figure 4.12
As mentioned in Chapter 4.2.1.3 EDGAR organizes related data in dedicated projects. For each project one database is created based on the described schema. The modular data scheme and the project based approach allow to arbitrarily update single projects of the EDGAR database backend whenever new genomic data becomes available for the respective project.

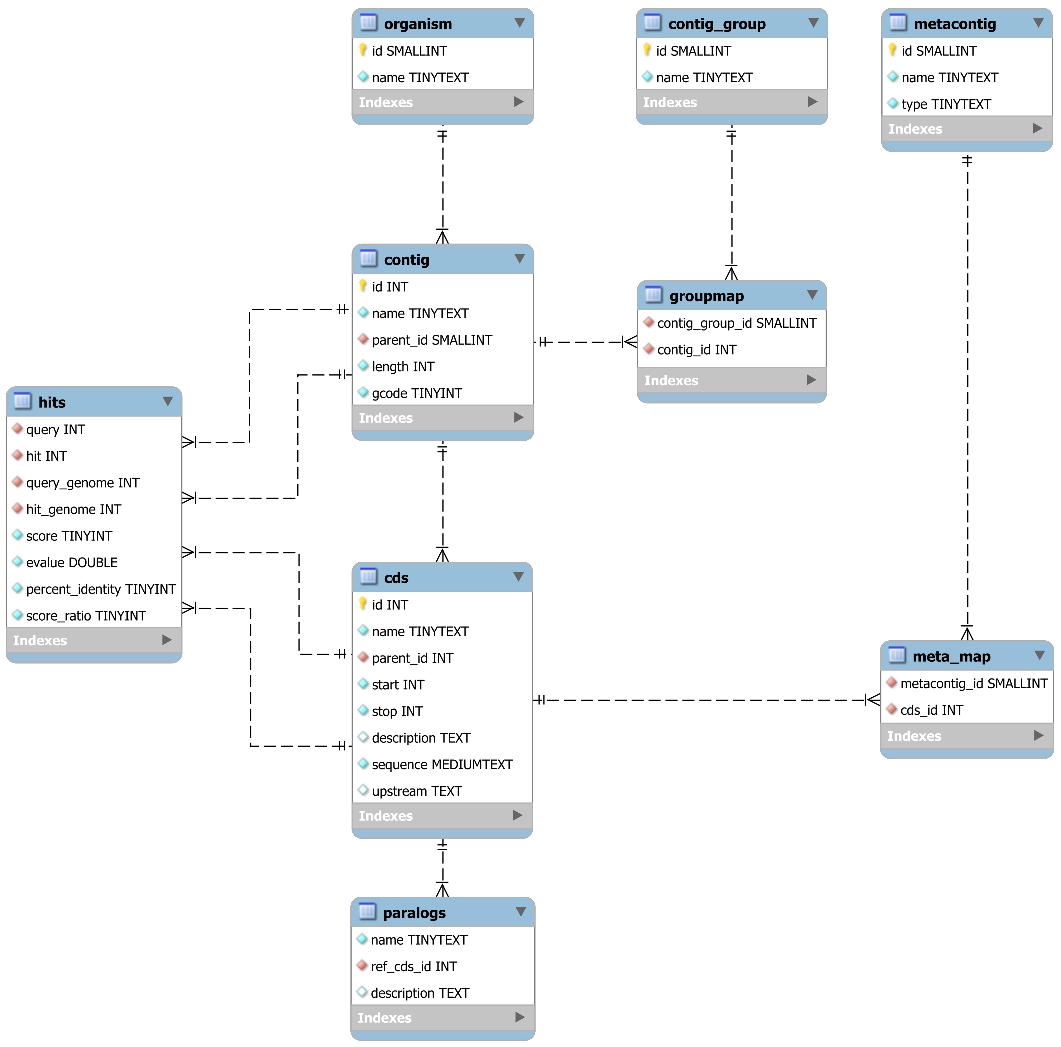


Figure 4.12.: Diagram of the EDGAR database schema. In the center the three main sequence storage tables `organism`, `contig`, and `CDS` are displayed. The fourth table `paralogs` in the lower right part stores the name and description of filtered genes and references the representative CDS that was used in the calculations. On the left the table `hits` is displayed which stores the BLAST hit information. This table references the query and hit CDS as well as the contig of these two CDS by foreign keys. In the right part of the image the tables for higher order data structures are shown with the tables `contig_group` and `metacontig`. The semantics of `contig_group` and `metacontig` are comparable as both aggregate several contigs in one administrative unit, with the difference that `contig_group` aggregates complete contigs while `metacontig` stores nonredundant sets of CDS from several contigs. Both concepts have a $n : m$ relationship to their respective data tables `contig` and `CDS` and thus need association tables. These are realized with the tables `groupmap` and `meta_map`. The connections in the diagram denote $1 : n$ relationships by foreign keys where the referenced field is always "id".

4.2.2. Implementation

Based on the design described in the previous section several comparative features are implemented in EDGAR. This section will first present how an EDGAR project can be created, what types of projects are available, and how a phylogenetic tree can be created for the calculated project. Thereafter, the basic layout and implementation of the EDGAR web interface will be introduced. In the following chapters the different analysis features of EDGAR will be presented and illustrated with screenshots of the respective visualizations from the web interface.

4.2.2.1. Project setup

The creation of an EDGAR project can be based on two different data sources: A GenDB project or a set of Genbank-formatted files[7] (GBKs).

GenDB (Meyer et al., 2003) is an open-source genome annotation system developed and maintained at Bielefeld University. It provides an automatic annotation pipeline, a web interface for collaborative manual annotation and several analysis features for whole genome analysis. It is, like EDGAR, project based and can store several related genomes in one project. Such a project is a perfect data source for EDGAR as it provides all required information about the source organisms together with a high quality automatic functional annotation, or even better a manually curated annotation. The organism, contig and CDS data can be easily extracted from a GenDB project and is used to calculate the all-against-all BLAST comparison and to initialize the EDGAR project. If an EDGAR project is created from an GenDB project the database identifiers from GenDB are cloned such that a CDS in EDGAR has the same ID as the respective CDS region in GenDB. Thereby comparative results from EDGAR can be transferred to GenDB conveniently.

For genomes that are not available in GenDB a second project creation routine is available that uses an arbitrary set of GBKs. This routine needs one GBK for every replicon that should be included in a project, furthermore a mapping file is needed that assigns one or multiple replicons to one organism. All needed sequence and annotation information is extracted from the GBKs using Bioperl. Draft sequences can be included into a project, too, by concatenating the sequence fragments to build an artificial genome.

Apart from the used data source the project creation routines both use the same backbone functionality. The project creation is started as a single Perl script that handles the complete EDGAR pipeline. In the first step the sequence data is parsed, organisms, contigs and CDS are inserted into the database, identical paralogs are filtered and stored in an extra table, and the all-against-all BLAST comparison is prepared. EDGAR uses the NCBI implementation of the protein alignment algorithm BLASTP with the default scoring matrix BLOSUM62 (Henikoff and Henikoff, 1992) and an initial evalue of $1e-5$ to filter insignificant BLAST results and thereby reduce the parsing effort. As an all-against-all BLAST

[7] `http://www.ncbi.nlm.nih.gov/Sitemap/samplerecord`, as at 15.08.2012.

of all genes of a bigger project easily results in millions of alignments that have to be calculated this computations are executed on the CeBiTec compute cluster. This compute cluster with $\sim$ 800 CPUs and more than 4000 CPU cores guarantees that the BLAST calculation step does not take more than one hour even for the biggest EDGAR projects. For management and scheduling of the compute jobs the Sun Grid Engine (SGE[8]) and a Perl DRMAA (Distributed Resource Management Application API) interface are used. When all BLAST calculations are finished, the results will be parsed and score ratio values for all hits will be calculated. Based on the score ratio values an orthology threshold is calculated (see chapter 4.2.1.2) and BLAST results that are above threshold will be stored in the database. Finally, images of the SRV distributions and the final cutoff histogram will be created, and in a clean up step BLAST databases, FASTA files, and any intermediate data will be removed before the project creation is finished. The EDGAR pipeline is illustrated in Figure 4.13.

EDGAR projects can be created either as private projects with access control, or as open projects available for every interested visitor of the EDGAR website. Private projects are created upon request from GenDB projects or from a set of GKBs provided by the user. User management and access control are realized as described in chapter 4.2.1.3. Open projects are realized via a default EDGAR user that is member of all open projects. The majority of open projects is part of the public resource EDGAR provides by processing all complete bacterial genomes from the NCBI nucleotide database. For this purpose a local mirror of all complete genomes in the NCBI database is maintained at the CeBiTec. With a single Perl script GBKs for all genomes within the database are sorted into folders according to their genus if a defined number of genomes is found for a certain genus (default is three genomes). The needed replicon mapping files are automatically generated for each genus, too. With a loop that iterates over the created folders a project for each genus can be created. In this way the complete data inventory of the NCBI can be processed with just two command lines. Another class of open projects are publication-related projects that once were private but were made public together with a publication describing EDGAR results. Private projects can be made public easily at any time by just adding the default user as project member.

4.2.2.2. Phylogenetic trees

Fitch (1970) introduced the definition of "orthologous genes" and postulated that these orthologs alone should be the basis for phylogenetic analyses (see chapter 2.4.2). As the main purpose of EDGAR is the estimation of orthologs among several genomes it is straightforward to infer phylogenetic knowledge from that orthology information. Gontcharov et al. (2004) showed that the combination of several genes gives superior results when compared to phylogenetic trees derived from single genes

[8]`http://www.oracle.com/us/products/tools/oracle-grid-engine-075549.html`, as at 15.08.2012

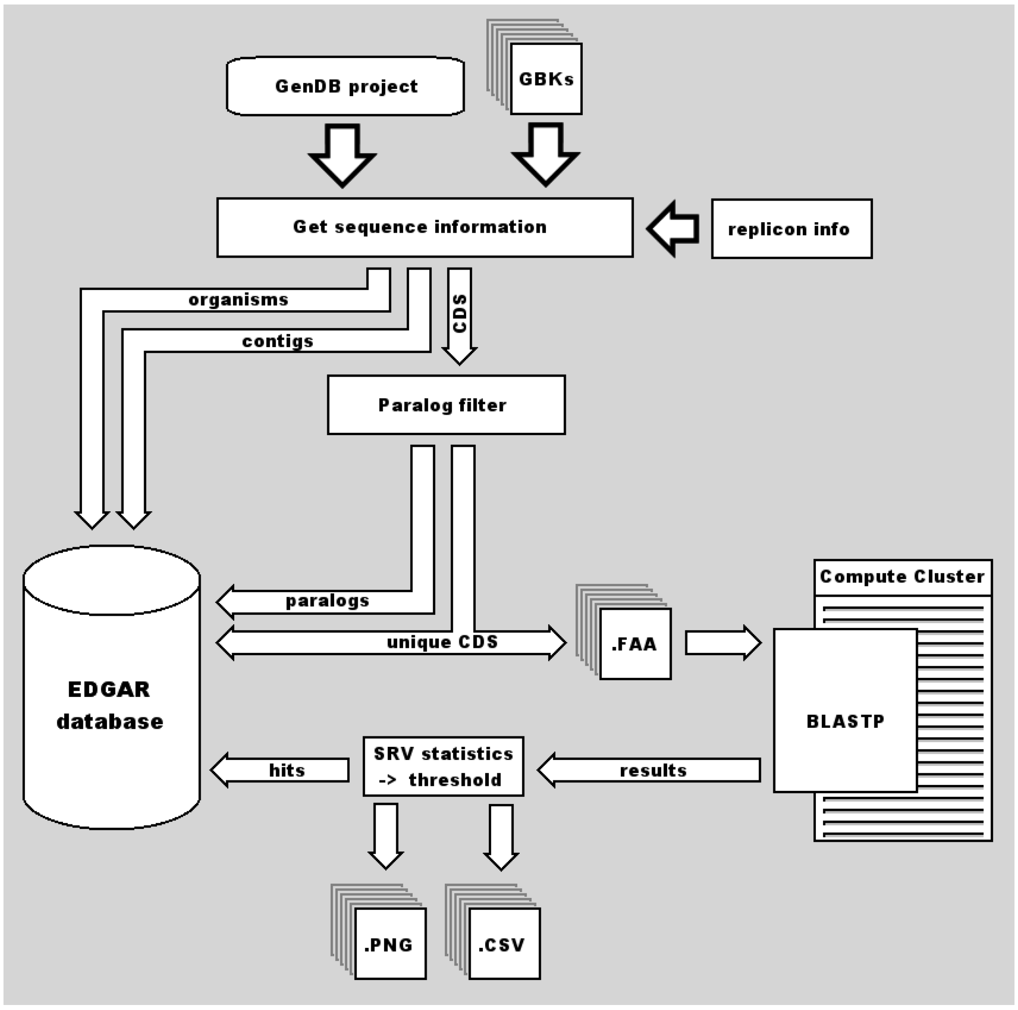
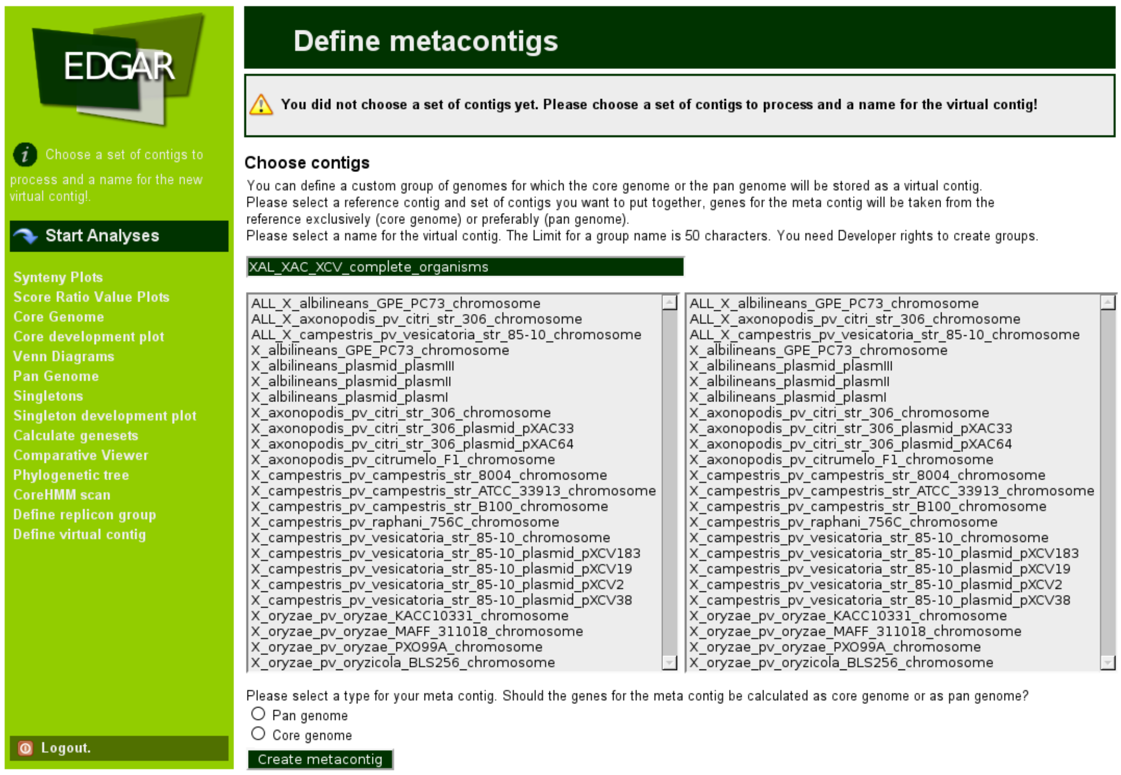


Figure 4.13.: Diagram of the EDGAR project creation workflow. Input data can be either taken from a GenDB project or from a set of GBKs. Furthermore a mapping file is needed that assigns single replicons to their organism. In a first step the sequence information is processed and organism and contig information is inserted into the database, coding sequences (CDS) are first filtered for identical paralogous genes. The resulting unique CDSs and paralogs are inserted into the database, and the unique CDS are exported as protein FASTA file (.FAA) that serves as input and as database for the all-against-all BLASTP comparison. The BLAST results are calculated on the CeBiTec compute cluster, and results are used to generate the SRV statistics described in chapter 4.2.1.2 to generate an orthology threshold. BLAST results with a SRV exceeding this threshold are stored in the "hits" table of the database. During the calculation of the SRV statistics images and flat files of the SRV histograms are created and stored. After the depicted pipeline is finished all temporary data will be removed automatically.

like the popular 16s rRNA genes. Ciccarelli et al. (2006) demonstrated that a set of 31 conserved orthologous genes gives a stable phylogenetic tree even across the three taxonomic domains Bacteria, Archaea, and Eukaryota. In a phylogenetic analysis of 12 insect genomes Zdobnov and Bork (2007) recommend the use of all core genes of a set of genomes to maximize the sequence support for the phylogenetic tree: "To quantify the divergence of these genomes, we focused on the fraction of single-copy orthologs, that is, genes that have exactly one ortholog in each of the genomes. This is the ideal marker set to study rates of evolution as these genes are most likely to retain their ancestral function and thus to evolve under similar constraints." (Zdobnov and Bork, 2007, p. 16)
Zdobnov *et al.* propose a pipeline for the construction of reliable, core genome based phylogenetic trees that is used in EDGAR in a slightly adapted version. The pipeline starts with the calculation of a core genome for a selection of genomes. Every set of orthologous genes found in all genomes is separately aligned using the multiple alignment tool MUSCLE (Edgar, 2004). Non-matching parts of the multiple alignments are masked using GBLOCKS (Talavera, 2007) and removed subsequently. The remaining matching parts of the alignments are concatenated to one huge multiple alignment that can easily be hundreds of thousand of residues long. A distance matrix based on the Kimura distance (Kimura, 1985) is calculated from this alignment and finally a phylogenetic tree is constructed based on this distance matrix using the Neighbor-Joining method by Saitou and Nei (1987). The two latter methods are used in the PHYLIP implementations by Felsenstein (1995). The Neighbor-Joining method was chosen as it is a heuristic approach with a very good computational efficiency, making it well suited for large datasets resulting from the core genome based tree construction. The described pipeline for the construction of phylogenetic trees was created using the Conveyor workflow engine developed by Linke et al. (2011). Conveyor offers a graphical user interface to build complex workflows out of a library of existing implemented processing steps. Generated workflows can be stored in an executable form, but can also be edited at any time using the graphical design tool, e.g., to use another multiple alignment tool or an alternative phylogeny algorithm in the tree construction.

4.2.2.3. User interface

The user interface is based on an Apache Web Server using mod_perl and CGI. The HTML code is organized in static HTML templates, and the graphical layout is implemented using CSS stylesheets. A central CGI script loads Perl modules that fill the respective HTML templates depending on the user selection, and dynamic components are realized using JavaScript. In the following sections the different features of the web interface will be presented.



Define metacontigs

You did not choose a set of contigs yet. Please choose a set of contigs to process and a name for the virtual contig!

Choose contigs

You can define a custom group of genomes for which the core genome or the pan genome will be stored as a virtual contig.
Please select a reference contig and set of contigs you want to put together, genes for the meta contig will be taken from the reference exclusively (core genome) or preferably (pan genome).
Please select a name for the virtual contig. The Limit for a group name is 50 characters. You need Developer rights to create groups.

XAL_XAC_XCV_complete_organisms

```
ALL_X_albilineans_GPE_PC73_chromosome
ALL_X_axonopodis_pv_citri_str_306_chromosome
ALL_X_campestris_pv_vesicatoria_str_85-10_chromosome
X_albilineans_GPE_PC73_chromosome
X_albilineans_plasmid_plasmIII
X_albilineans_plasmid_plasmII
X_albilineans_plasmid_plasmI
X_axonopodis_pv_citri_str_306_chromosome
X_axonopodis_pv_citri_str_306_plasmid_pXAC33
X_axonopodis_pv_citri_str_306_plasmid_pXAC64
X_axonopodis_pv_citrumelo_F1_chromosome
X_campestris_pv_campestris_str_8004_chromosome
X_campestris_pv_campestris_str_ATCC_33913_chromosome
X_campestris_pv_campestris_str_B100_chromosome
X_campestris_pv_raphani_756C_chromosome
X_campestris_pv_vesicatoria_str_85-10_chromosome
X_campestris_pv_vesicatoria_str_85-10_plasmid_pXCV183
X_campestris_pv_vesicatoria_str_85-10_plasmid_pXCV19
X_campestris_pv_vesicatoria_str_85-10_plasmid_pXCV2
X_campestris_pv_vesicatoria_str_85-10_plasmid_pXCV38
X_oryzae_pv_oryzae_KACC10331_chromosome
X_oryzae_pv_oryzae_MAFF_311018_chromosome
X_oryzae_pv_oryzae_PXO99A_chromosome
X_oryzae_pv_oryzicola_BLS256_chromosome
```

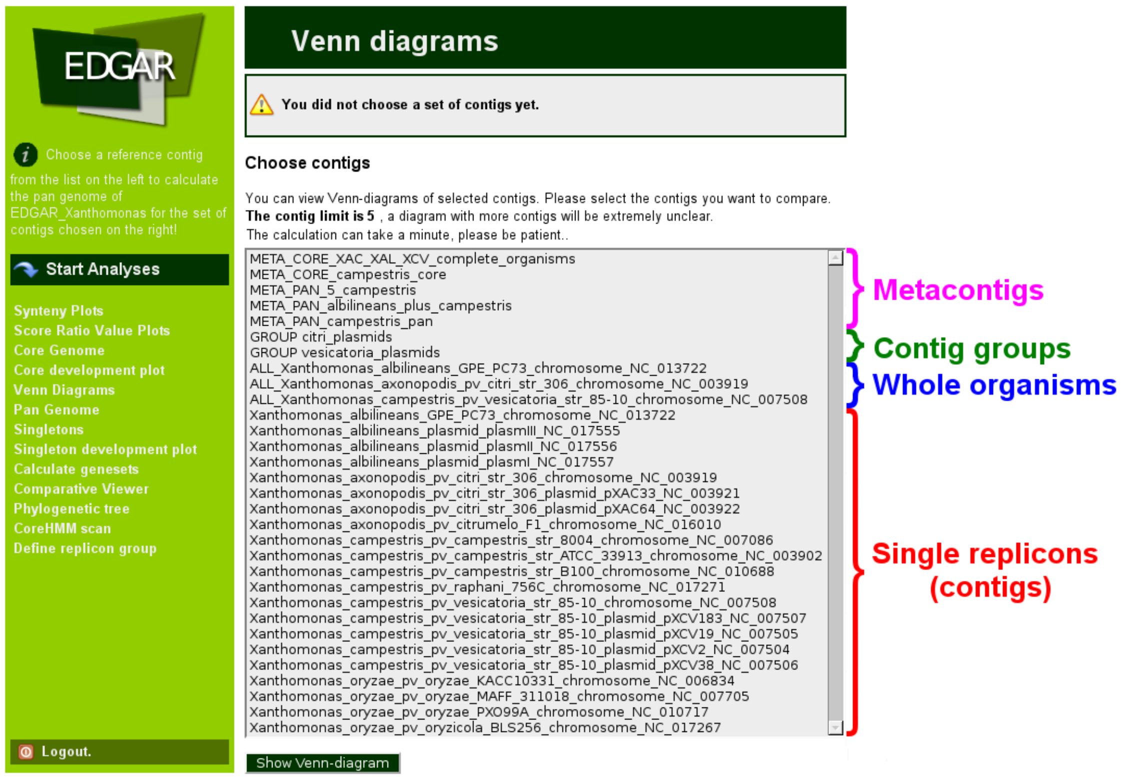
```
ALL_X_albilineans_GPE_PC73_chromosome
ALL_X_axonopodis_pv_citri_str_306_chromosome
ALL_X_campestris_pv_vesicatoria_str_85-10_chromosome
X_albilineans_GPE_PC73_chromosome
X_albilineans_plasmid_plasmIII
X_albilineans_plasmid_plasmII
X_albilineans_plasmid_plasmI
X_axonopodis_pv_citri_str_306_chromosome
X_axonopodis_pv_citri_str_306_plasmid_pXAC33
X_axonopodis_pv_citri_str_306_plasmid_pXAC64
X_axonopodis_pv_citrumelo_F1_chromosome
X_campestris_pv_campestris_str_8004_chromosome
X_campestris_pv_campestris_str_ATCC_33913_chromosome
X_campestris_pv_campestris_str_B100_chromosome
X_campestris_pv_raphani_756C_chromosome
X_campestris_pv_vesicatoria_str_85-10_chromosome
X_campestris_pv_vesicatoria_str_85-10_plasmid_pXCV183
X_campestris_pv_vesicatoria_str_85-10_plasmid_pXCV19
X_campestris_pv_vesicatoria_str_85-10_plasmid_pXCV2
X_campestris_pv_vesicatoria_str_85-10_plasmid_pXCV38
X_oryzae_pv_oryzae_KACC10331_chromosome
X_oryzae_pv_oryzae_MAFF_311018_chromosome
X_oryzae_pv_oryzae_PXO99A_chromosome
X_oryzae_pv_oryzicola_BLS256_chromosome
```

Please select a type for your meta contig. Should the genes for the meta contig be calculated as core genome or as pan genome?
○ Pan genome
○ Core genome

Figure 4.14.: Screenshot of the EDGAR metacontig creation page. A user can select a name for the metacontig, a reference organism/contig from the left list, and a set of contigs/organisms to be grouped together from the right list. At the bottom of the selection lists two options for the formation of a non-redundant gene set can be chosen, a pan genome calculation or a core genome calculation.

4.2.2.4. Multi-replicon organisms, genome groups, and metacontigs

As mentioned in Section 4.2.1.4, EDGAR stores uploaded organisms and the belonging replicons (called `contigs` in EDGAR) in separate data structures. For the organism only a name is stored, while the coding sequences are assigned to the single contigs. This partition of organisms into their replicons can be found in all analysis features in the web interface. It allows the user to perform analyses and comparisons of, for instance, single plasmids, or only chromosomes without the plasmids. To also allow users the analysis of complete organisms the web interface provides menu items for each organism with more than one replicon (single replicon organisms are represented in the list of contigs, already). These menu items have a prefix "ALL_" and will cause all replicons of an organism to be treated as one contig during a selected calculation.
So with the basic data structures EDGAR provides comparisons of single contigs or complete organisms, but nothing in between or beyond. For example, a level of abstraction between organism and contig is needed if someone wants to compare the



Choose a reference contig from the list on the left to calculate the pan genome of EDGAR_Xanthomonas for the set of contigs chosen on the right!

Venn diagrams

You did not choose a set of contigs yet.

Choose contigs

You can view Venn-diagrams of selected contigs. Please select the contigs you want to compare.
The contig limit is 5 , a diagram with more contigs will be extremely unclear.
The calculation can take a minute, please be patient..

```
META_CORE_XAC_XAL_XCV_complete_organisms
META_CORE_campestris_core
META_PAN_5_campestris
META_PAN_albilineans_plus_campestris
META_PAN_campestris_pan
GROUP citri_plasmids
GROUP vesicatoria_plasmids
ALL_Xanthomonas_albilineans_GPE_PC73_chromosome_NC_013722
ALL_Xanthomonas_axonopodis_pv_citri_str_306_chromosome_NC_003919
ALL_Xanthomonas_campestris_pv_vesicatoria_str_85-10_chromosome_NC_007508
Xanthomonas_albilineans_GPE_PC73_chromosome_NC_013722
Xanthomonas_albilineans_plasmid_plasmIII_NC_017555
Xanthomonas_albilineans_plasmid_plasmII_NC_017556
Xanthomonas_albilineans_plasmid_plasmI_NC_017557
Xanthomonas_axonopodis_pv_citri_str_306_chromosome_NC_003919
Xanthomonas_axonopodis_pv_citri_str_306_plasmid_pXAC33_NC_003921
Xanthomonas_axonopodis_pv_citri_str_306_plasmid_pXAC64_NC_003922
Xanthomonas_axonopodis_pv_citrumelo_F1_chromosome_NC_016010
Xanthomonas_campestris_pv_campestris_str_8004_chromosome_NC_007086
Xanthomonas_campestris_pv_campestris_str_ATCC_33913_chromosome_NC_003902
Xanthomonas_campestris_pv_campestris_str_B100_chromosome_NC_010688
Xanthomonas_campestris_pv_raphani_756C_chromosome_NC_017271
Xanthomonas_campestris_pv_vesicatoria_str_85-10_chromosome_NC_007508
Xanthomonas_campestris_pv_vesicatoria_str_85-10_plasmid_pXCV183_NC_007507
Xanthomonas_campestris_pv_vesicatoria_str_85-10_plasmid_pXCV19_NC_007505
Xanthomonas_campestris_pv_vesicatoria_str_85-10_plasmid_pXCV2_NC_007504
Xanthomonas_campestris_pv_vesicatoria_str_85-10_plasmid_pXCV38_NC_007506
Xanthomonas_oryzae_pv_oryzae_KACC10331_chromosome_NC_006834
Xanthomonas_oryzae_pv_oryzae_MAFF_311018_chromosome_NC_007705
Xanthomonas_oryzae_pv_oryzae_PXO99A_chromosome_NC_010717
Xanthomonas_oryzae_pv_oryzicola_BLS256_chromosome_NC_017267
```

Figure 4.15.: A typical selection list in the EDGAR web interface, in this case a selection for the creation of Venn diagrams. The four supported menu items are displayed, single contigs (no prefix), complete organisms (prefix "ALL_", replicon groups (prefix "GROUP_"), or metacontigs (prefix "META_").

gene content of all plasmids of one organism to the genes of all plasmids of another organism. For such cases EDGAR allows users to create groups of contigs of their choice. This groups work fine if they are created of contigs within one organism, but if contigs from different organisms are grouped together the redundant orthologs within the group act as artificial paralogs and prevent a reasonable analysis. Unfortunately, such comparisons of contig sets from different organisms or even of sets of complete organisms can be crucial to answer biological question, e.g, if a researcher wants to compare a set of pathogenic bacteria to a set of non-pathogenic bacteria. Thus, an abstraction level above organism is needed to store non-redundant sets of genes for a group of contigs or organisms. In EDGAR this is realized via so-called metacontigs. A user can define metacontigs via the web interface (see Figure 4.14). First, he has to select organisms and contigs that should be grouped together with a reference genome. Second, a method to remove redundant genes has to be selected, either the pan genome or the core genome calculation. The respective genomic subset will be calculated for the contigs and organisms and a non-redundant set of genes is extracted by using one representative gene of each ortholog set from the result, preferring the gene from the reference. This non-redundant set forms the

metacontig and is stored in the tables `metacontig` and `meta_map` as described in Section 4.2.1.4.
The two user defined groups have their respective menu item in the EDGAR web interface. Contig groups have just the prefix "GROUP_", while metacontigs have a prefix "META_" followed by a second prefix "PAN_" or "CORE_" to indicate what type of metacontig was created. A typical selection list for the calculation of a Venn diagram (see Section 4.2.2.6) with all four types of menu items is shown in Figure 4.15. In the following screenshots this list was removed to reduce image sizes.

4.2.2.5. Calculation of the genomic subsets

The estimation of the genomic subsets defined in Section 2.4.3 is among the most common tasks in the comparative analysis of multiple genomes. Thus, the calculation of these subsets is the core functionality of the EDGAR web interface. It supports the rapid calculation of the core genome, singleton genes, or the pan genome of selected query genomes.

Core genome calculation The core genome is calculated by iterative pairwise comparison of a set of genomes G. A genome A is selected as reference genome, and the gene content of this genome is taken as base set for the following calculations. This set g_A of genes is compared to a set g_B of genes contained in a second genome B of the set G. Each gene in set g_A is checked for reciprocal best BLAST hits against the gene set g_B using the SRV orthology threshold based information stored in the database table `hits`. Every gene from set g_A that has no reciprocal best hit in set g_B is removed from the set. The resulting set $g_{A'}$ is then iteratively compared to the remaining genomes in G, resulting in a final set of genes that have hits in all genomes of G, thus forming the core genome. The iterative process is illustrated in the top section of Figure 4.16.

Singleton calculation The singleton genes are calculated in an iterative approach as well. Again, out of a set of genomes G a genome A is selected and its gene content is taken as base set for the following calculations. The set g_A of genes is compared to a set of genes g_B from genome B out of G. Each gene of g_A is checked for a hit exceeding the SRV based threshold against g_B. If such a hit is found the gene is removed from the intermediate result set. For the singleton calculation simple unidirectional hits are used as a singleton gene is required to show no significant similarity against any other genome. Thus, to avoid false positive singletons the check for reciprocal hits is omitted. The resulting gene set $g_{A'}$ is compared against the remaining genomes in G until the final set of singleton genes is found. The iterative process is illustrated in the bottom section of Figure 4.16.

Pan genome calculation The pan genome is also calculated in a similar iterative way, but in this case the result set of genes is not reduced but extended with every

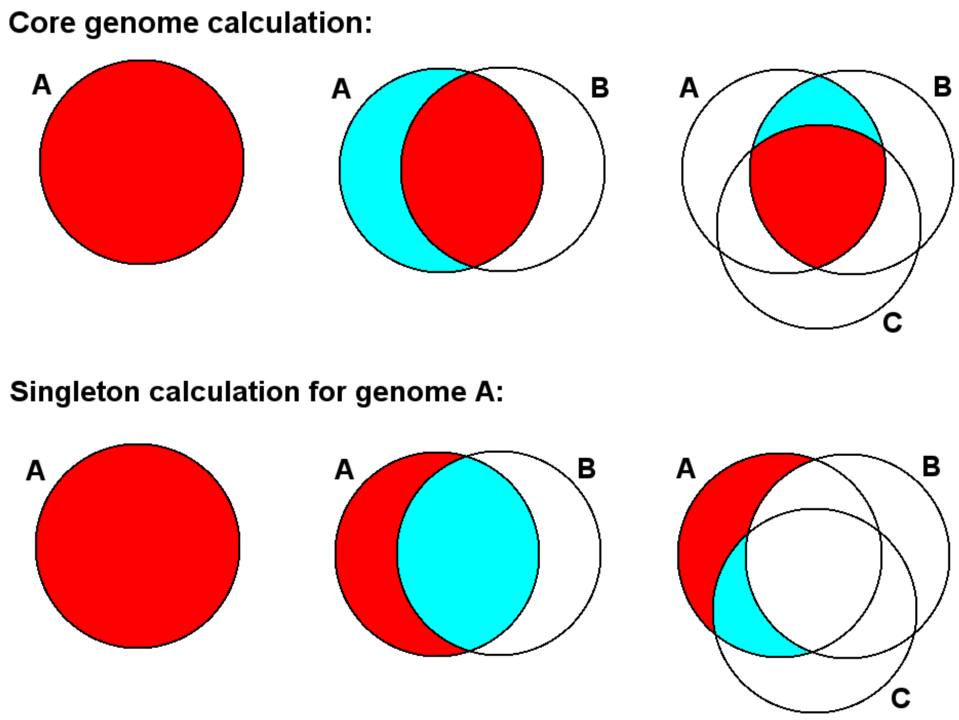


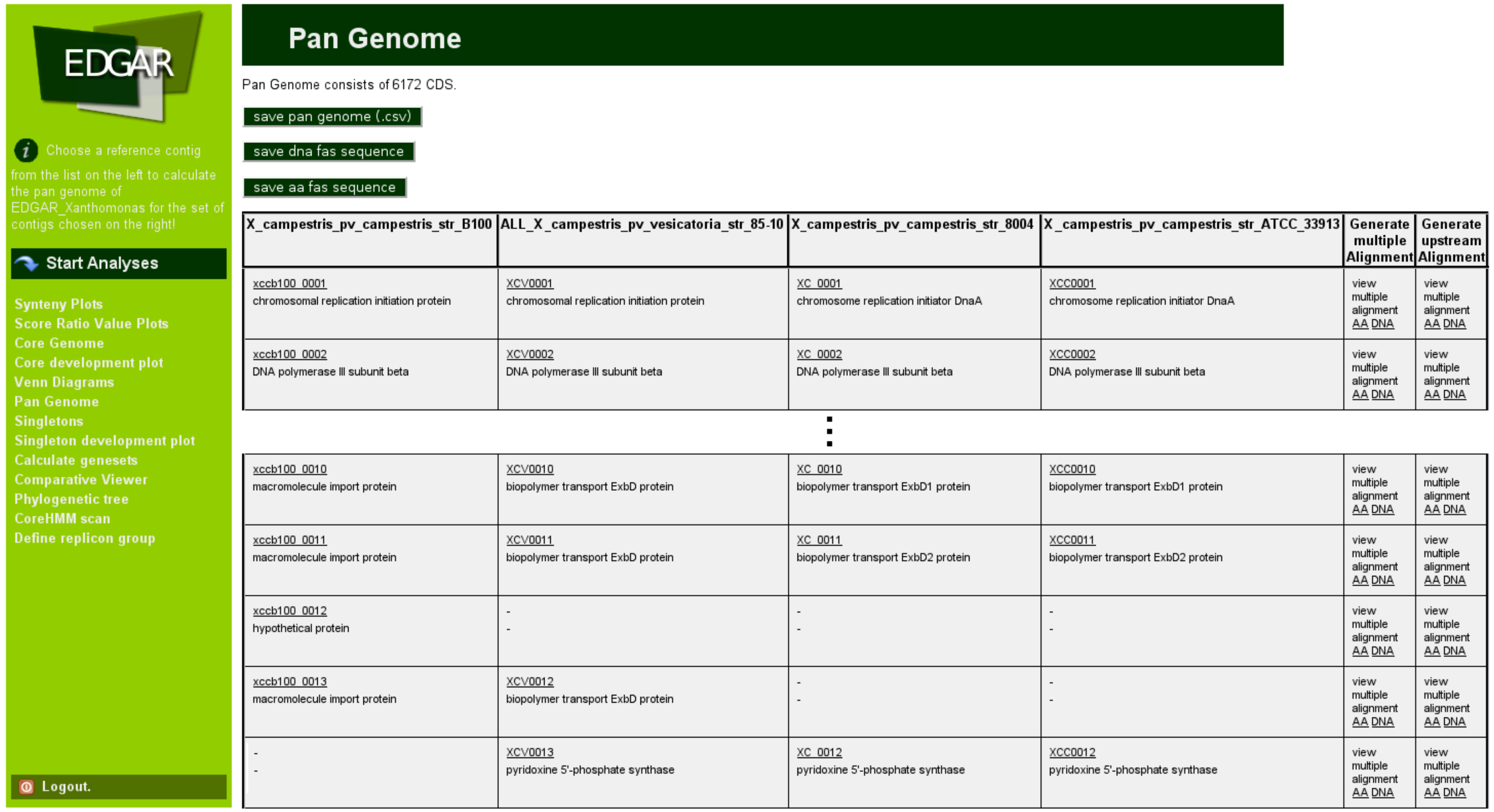
Figure 4.16.: Schematic representation of the iterative calculation of the core genome (top row) and singleton genes (bottom row). Areas colored red are the sought-after result set of genes. The light blue areas are gene sets that are removed from the result set in the current iteration.

new genome iteration. A set g_B of genes is compared to the base set g_A of genes. Every gene of g_B that has no ortholog in g_A is added to the reference set. This process is repeated iteratively for all genomes in the set G, extending the base set g_A step by step to the pan genome. The result table for a pan genome calculation in the EDGAR web interface is displayed in Figure 4.17.

For all three calculations the user has to select from the `contigs/organisms` included in the project a reference genome and a set of genomes it should be compared to. The reference genome serves as starting genome A and will appear in the first column of the result table. To minimize bias due to genes coming into the calculation in different order the remaining genomes are always processed in alphabetical order. The selected reference genome has nearly no impact on the resulting core genome, however, there may be some small bias ($< 1\%$) due to highly similar genes appearing in different order during the calculation.

4.2.2.6. Venn diagrams and set comparisons

As described in the previous section, EDGAR features the automatic calculation of the major genomic subsets. A connatural user requirement may be a general overview of genes shared between a selection of genomes including the intersection



Choose a reference contig from the list on the left to calculate the pan genome of EDGAR_Xanthomonas for the set of contigs chosen on the right!

Pan Genome

Pan Genome consists of 6172 CDS.

X_campestris_pv_campestris_str_B100	ALL_X_campestris_pv_vesicatoria_str_85-10	X_campestris_pv_campestris_str_8004	X_campestris_pv_campestris_str_ATCC_33913	Generate multiple Alignment	Generate upstream Alignment
xccb100_0001 chromosomal replication initiation protein	XCV0001 chromosomal replication initiation protein	XC_0001 chromosome replication initiator DnaA	XCC0001 chromosome replication initiator DnaA	view multiple alignment AA DNA	view multiple alignment AA DNA
xccb100_0002 DNA polymerase III subunit beta	XCV0002 DNA polymerase III subunit beta	XC_0002 DNA polymerase III subunit beta	XCC0002 DNA polymerase III subunit beta	view multiple alignment AA DNA	view multiple alignment AA DNA
		⋮			
xccb100_0010 macromolecule import protein	XCV0010 biopolymer transport ExbD protein	XC_0010 biopolymer transport ExbD1 protein	XCC0010 biopolymer transport ExbD1 protein	view multiple alignment AA DNA	view multiple alignment AA DNA
xccb100_0011 macromolecule import protein	XCV0011 biopolymer transport ExbD protein	XC_0011 biopolymer transport ExbD2 protein	XCC0011 biopolymer transport ExbD2 protein	view multiple alignment AA DNA	view multiple alignment AA DNA
xccb100_0012 hypothetical protein	- -	- -	- -	view multiple alignment AA DNA	view multiple alignment AA DNA
xccb100_0013 macromolecule import protein	XCV0012 biopolymer transport ExbD protein	- -	- -	view multiple alignment AA DNA	view multiple alignment AA DNA
- -	XCV0013 pyridoxine 5'-phosphate synthase	XC_0012 pyridoxine 5'-phosphate synthase	XCC0012 pyridoxine 5'-phosphate synthase	view multiple alignment AA DNA	view multiple alignment AA DNA

Figure 4.17.: Screenshot of the pan genome result table in the EDGAR web interface. The table shows the gene orthologous to each other in four different *Xanthomonas* genomes. For each gene the locus tag and a functional description is displayed. If no ortholog for a certain gene is found in other genomes a "-" will be shown in the table cell. At the end of each line there are links to generate multiple alignments of the genes within the line either based on DNA or amino acid sequence. Such alignments can be created also for the upstream region, the 400bp prior to the translation start site. The pan genome can be saved as multiple FASTA file (DNA or protein sequence) or as a TAB separated flat file of locus tags and functional descriptions. The calculation of a core genome gives a similar tabular result, the calculation of singleton genes results in a simple list.

sets within the dispensable genome. A typical representation of all possible logical relations between a finite number of sets is the Venn diagram. EDGAR allows the creation of Venn diagrams for custom selections of two to five genomes. Figure 4.18.A shows a Venn diagram of 3*rd* order and Figure 4.18.B one of 4*th* order, both created with the EDGAR web interface.

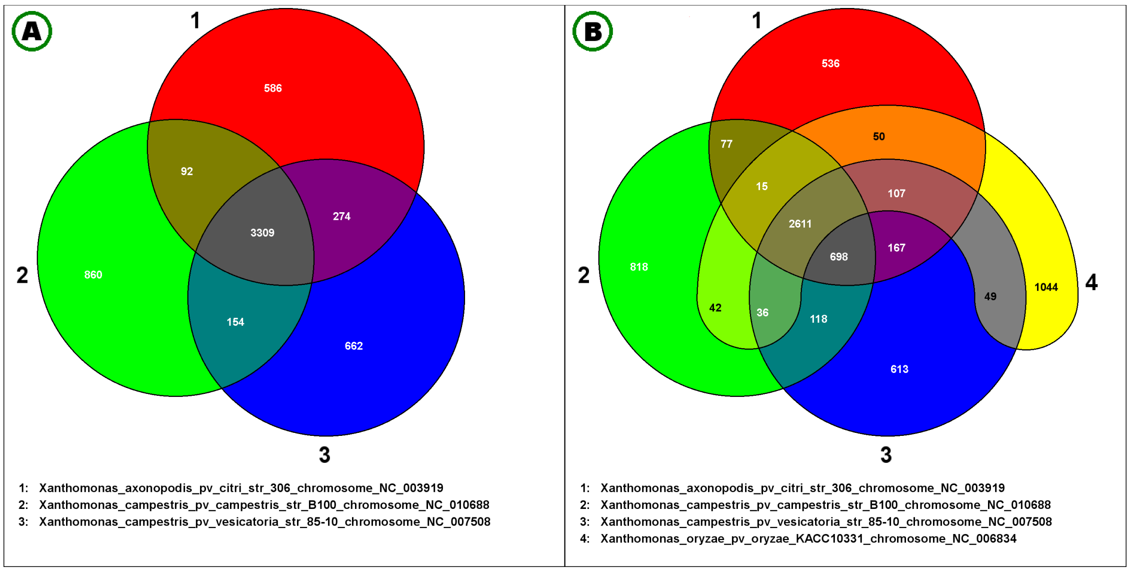
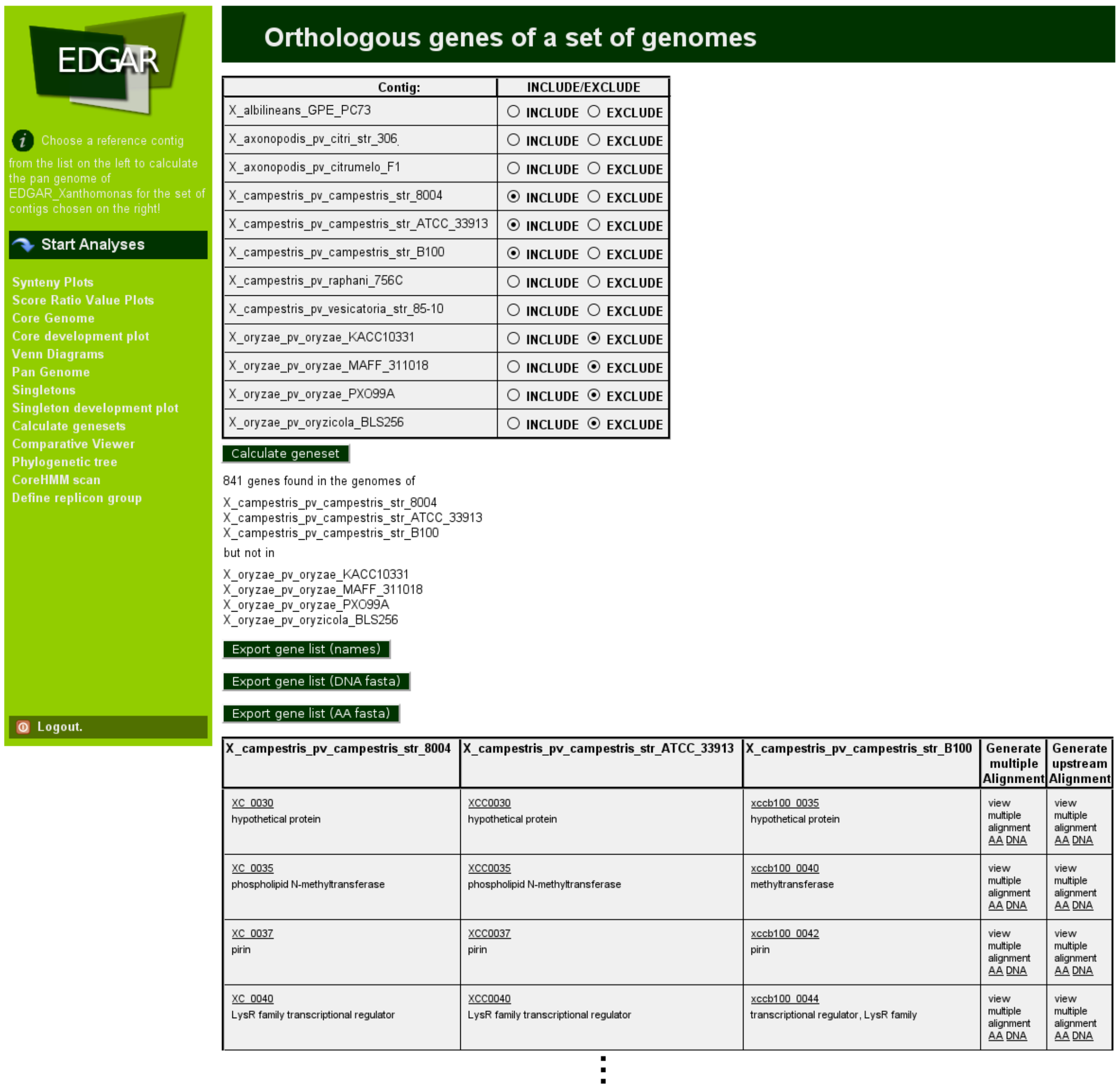


Figure 4.18.: Two Venn diagrams created with the EDGAR web interface. On the left side a Venn diagram of three *Xanthomonas* genomes is shown that illustrates the number of singleton genes (red, green, and blue ares) and the core genome (centered gray shape), but also the intersections in mixed colors. On the right side a Venn diagram of 4*th* order is shown resulting from the addition of another *Xanthomonas* strain. Again the core genome and the singleton genes appear in their respective areas, but all possible subsets of two or three genomes can be observed, too. Each included genome has one basic color that can be seen in the area representing the singletons, all other areas are colored with the combination color of the contributing genomes' colors. The numbers in the single areas are links that forward the user to a table comparable to the one shown in Figure 4.17 listing the genes included in the selected subset.

EDGAR limits the order of Venn diagrams to five although Venn diagrams for six and more genomes are possible. But as the number of regions within a Venn diagram of *nth* order is $2^n - 1$ this results in too many and too small areas in the graphical representation and thus becomes too unclear to be reasonable. To allow the user genome comparisons of higher order EDGAR provides an interface to calculate any possible intersection set of any arbitrary number of genomes. From a list the user can select single genomes as included, excluded, or ignored (by checking neither of the two options). EDGAR will calculate the gene set matching the query (see Figure 4.19) and present the results in tabular form.



Choose a reference contig from the list on the left to calculate the pan genome of EDGAR_Xanthomonas for the set of contigs chosen on the right!

Orthologous genes of a set of genomes

Contig:	INCLUDE/EXCLUDE
X_albilineans_GPE_PC73	INCLUDE EXCLUDE
X_axonopodis_pv_citri_str_306	INCLUDE EXCLUDE
X_axonopodis_pv_citrumelo_F1	INCLUDE EXCLUDE
X_campestris_pv_campestris_str_8004	[✓ INCLUDE] EXCLUDE
X_campestris_pv_campestris_str_ATCC_33913	[✓ INCLUDE] EXCLUDE
X_campestris_pv_campestris_str_B100	[✓ INCLUDE] EXCLUDE
X_campestris_pv_raphani_756C	INCLUDE EXCLUDE
X_campestris_pv_vesicatoria_str_85-10	INCLUDE EXCLUDE
X_oryzae_pv_oryzae_KACC10331	INCLUDE [✓ EXCLUDE]
X_oryzae_pv_oryzae_MAFF_311018	INCLUDE [✓ EXCLUDE]
X_oryzae_pv_oryzae_PXO99A	INCLUDE [✓ EXCLUDE]
X_oryzae_pv_oryzicola_BLS256	INCLUDE [✓ EXCLUDE]

841 genes found in the genomes of

X_campestris_pv_campestris_str_8004
X_campestris_pv_campestris_str_ATCC_33913
X_campestris_pv_campestris_str_B100

but not in

X_oryzae_pv_oryzae_KACC10331
X_oryzae_pv_oryzae_MAFF_311018
X_oryzae_pv_oryzae_PXO99A
X_oryzae_pv_oryzicola_BLS256

X_campestris_pv_campestris_str_8004	X_campestris_pv_campestris_str_ATCC_33913	X_campestris_pv_campestris_str_B100	Generate multiple Alignment	Generate upstream Alignment
XC_0030 hypothetical protein	XCC0030 hypothetical protein	xccb100_0035 hypothetical protein	view multiple alignment AA DNA	view multiple alignment AA DNA
XC_0035 phospholipid N-methyltransferase	XCC0035 phospholipid N-methyltransferase	xccb100_0040 methyltransferase	view multiple alignment AA DNA	view multiple alignment AA DNA
XC_0037 pirin	XCC0037 pirin	xccb100_0042 pirin	view multiple alignment AA DNA	view multiple alignment AA DNA
XC_0040 LysR family transcriptional regulator	XCC0040 LysR family transcriptional regulator	xccb100_0044 transcriptional regulator, LysR family	view multiple alignment AA DNA	view multiple alignment AA DNA

⋮

Figure 4.19.: Screenshot of the gene set calculation interface. In the upper part a table shows all genomes stored in the EDGAR project and a set of select buttons with the options "INCLUDE" and "EXCLUDE" for every genome. The gene set is calculated such that there has to be a set of orthologous genes in all included genomes while there must not be any ortholog to one of the excluded genomes. Genomes where neither "INCLUDE" nor "EXCLUDE" is checked are ignored. In the notation of set theory that means if two genomes A and B are included and a third genomes C is excluded, the subset $A \cap B \setminus C$ is calculated. In this example genes are calculated that are abundant in all *X. campestris* strains (pathogens of cruciferous plants like cabbage and rape) but not in *X. oryzae* strains (pathogens affecting rice). Results are presented in the known tabular form.

4.2.2.7. Synteny plots and comparative view

Synteny describes the physical co-localization of genes on a chromosome. The synteny of genes is a good phylogenetic indicator, as during evolution the ordering of genes is changed by rearrangement events like inversions, deletions, insertions or translocations. Hence, for organisms that are closely related a higher conservation of gene order is expected than for more distantly related organisms. To support the analysis of bacterial synteny, EDGAR provides an interface to create synteny plots of pairs of genomes. Figure 4.20 shows four synteny plots created with the EDGAR web interface. A reference organism, in this case *Xanthomonas campestris* pv. *campestris* B100, is compared to four other *Xanthomonas* genomes. One can see that the level of synteny is decreasing from Figure 4.20.A, where only one small and one larger inversion can be seen, to Figure 4.20.D, where a multitude of rearrangement events can be observed. This reflects the decreasing level of relatedness between the genomes from A to D.

To observe the synteny of genes at a smaller scale the comparative view was developed. In this visualization the orthologs of one selected gene are shown in their genomic neighborhood. This means, if for one reference gene the set of orthologs is computed, the positions of these genes in their respective genomes are aligned, and the genes are shown togther with all other genes in a 10 kb window around them (see Figure 4.21). This visualization allows to easily analyse conserved structures like operons or the general synteny of genomes. Furthermore, if there is a certain level of synteny, missing genes can be easily spotted and it can be checked if they are missing due to defective gene prediction or if a gene was truly deleted. Finally, if an EDGAR project was created from a GenDB project (see Section 4.2.2.1), the ortholog sets can be assigned a consistent functional annotation by copying the annotation from one reference gene to all orthologs. The functional annotation is updated directly in GenDB, but of course this feature is only available if the EDGAR user has `Annotator` rights in the respective GenDB project. The rights management in this case is maintained by the GPMS system described in Section 4.2.1.

4.2.2.8. Statistical extrapolation of genomic subset sizes

A core genome calculated on a certain set of genomes is always only a snapshot of the situation for exactly this genome set. To gain insight into the "real" core genome of a species one can try to extrapolate the number of core genes for a set of genomes of infinite size. This is done by the following calculations based on the approach proposed by Tettelin et al. (2005): One estimates the number of core genes for every possible permutation of k available genomes and stores the number of observed core genes for each particular genome count n – either a mean or a median value, or as distinct single values. For distinct single values Tettelin *et al.*

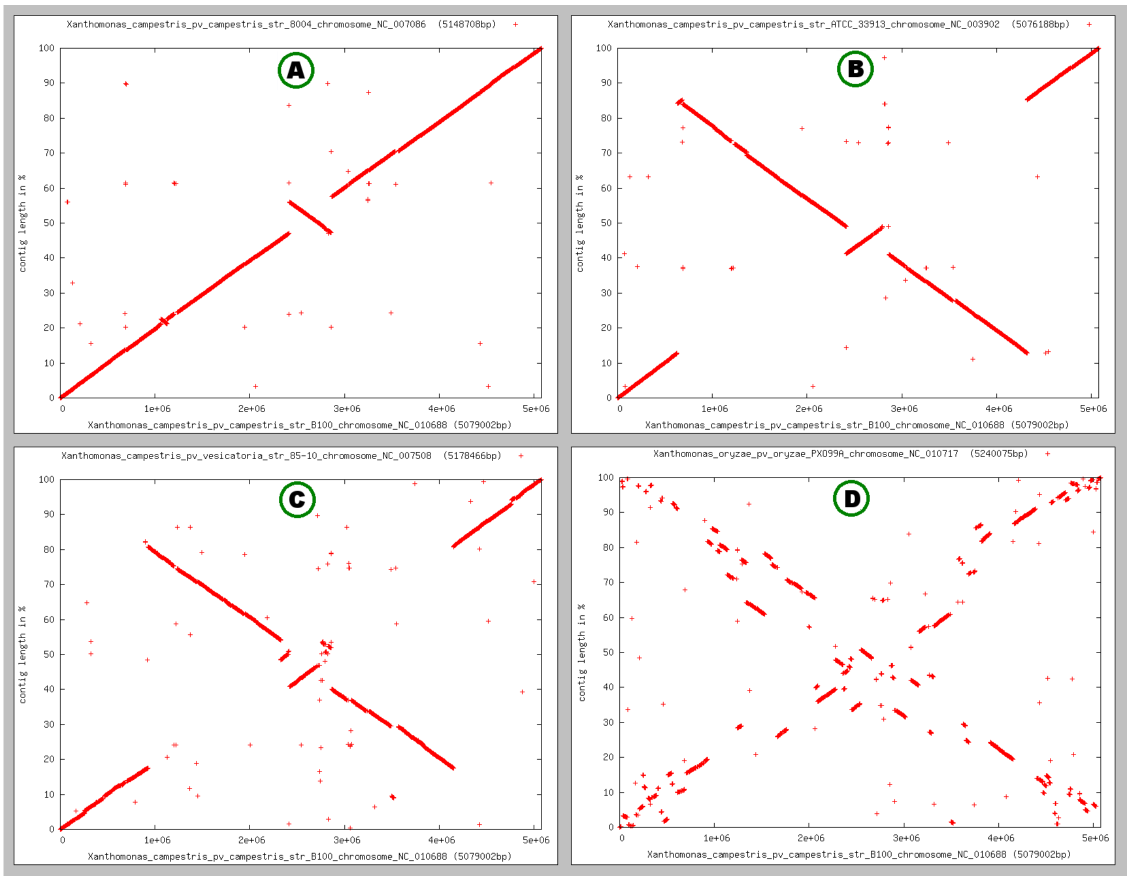
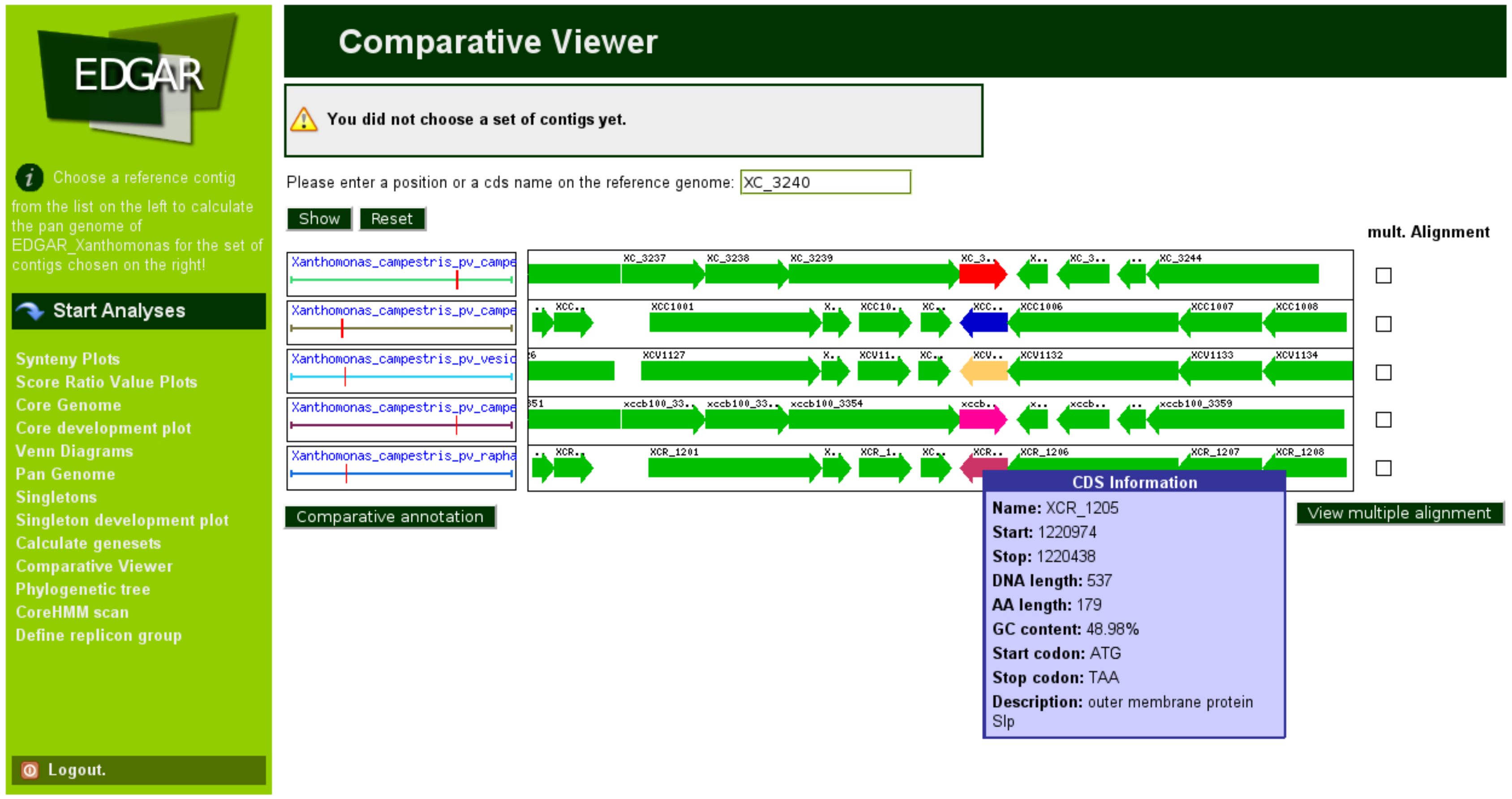


Figure 4.20.: Different synteny plots created with EDGAR: The x-axis shows the position within the reference genome, the y-axis shows the relative position within the compared genome. Red dots reflect the stop positions of genes that were found to be orthologous to each other. The single plots show the synteny of *Xanthomonas campestris* pv. *campestris* (XCC) B100 compared to the XCC strains 8004 (A) and ATCC33913 (B), a strain with differing pathovar *vesicatoria* (C), and a different species in the same genus, *Xanthomonas oryzae* (D). The decreasing relatedness between the compared strains is reflected by the decreasing conservation of gene order.



You did not choose a set of contigs yet.

Please enter a position or a cds name on the reference genome: XC_3240

Name: XCR_1205
Start: 1220974
Stop: 1220438
DNA length: 537
AA length: 179
GC content: 48.98%
Start codon: ATG
Stop codon: TAA
Description: outer membrane protein Slp

Figure 4.21.: Screenshot of the comparative view of the gene neighborhood, showing the location and context of a set of orthologous CDSs. The visualizations show a set of orthologs as colored arrows together with all surrounding genes in a 10 kb window (green arrows). The position of the orthologous genes is marked by red lines in the genome at the left side of the plot. In the example with 5 *Xanthomonas* genomes the gene order seems well conserved, although it is reversed in two of five genomes. Further information about each gene can be obtained by a mouseover information window (blue box). A multiple alignment can be created for the highlighted orthologs. In cases when the EDGAR project was created from a GenDB project, a special interface for comparative annotation allows to copy functional assigments from a reference gene to all orthologs.

also count core genome sizes for identical genome combinations in different order to reflect bias introduced by paralogous genes. Thus, Tettelin *et al.* work with

$$N = \frac{k!}{(n-1)! \cdot (k-1)!} = n \cdot \binom{n}{k} \tag{4.1}$$

permutations of n genomes out of k genomes. As identical paralogs are filtered in EDGAR and furthermore the bias introduced by differing genome order is negligible, EDGAR uses only unique permutations and thus only the simple binomial coefficient with $\binom{n}{k}$ different combinations for n out of k genomes.
Subsequently the number of core genes is plotted as a function of the number of compared genomes and serves as input for a non-linear least squares curve fitting approach. In this step an exponential decay function

$$f_c(n) = k_c \cdot \exp\left(\frac{-n}{\tau_c}\right) + \Omega \tag{4.2}$$

is fitted to the data, where k_c is the amplitude of the exponential function, n is the number of strains, τ_c is the decay constant that defines the speed at which f_c converges against its asymptotic value and Ω is the extrapolated size of the core genome for $n \to \infty$. The Ω value indicates how well the core genome of the currently available genomes reflects the "real" core genome of, e.g., a genus.

In a similar approach the expected number of singletons for each newly sequenced strain can be predicted. Again, for each of the $\binom{n}{k}$ possible genome combinations the number of singleton genes is estimated, but as a set of n genomes results in n singleton counts (each genome of the set has by definition its independent set of singleton genes) this time $n \cdot \binom{n}{k}$ distinct values are registered for n out of k genomes. The curve fitting uses an exponential decay function

$$f_s(n) = k_s \cdot \exp\left(\frac{-n}{\tau_s}\right) + tg(\theta), \tag{4.3}$$

where $tg(\theta)$ is the expected number of singletons observed for every newly sequenced strain and n, k_s, and τ_s have the same meaning as corresponding variables in Equation (4.2).

For the extrapolation of the pan genome size Tettelin et al. (2005) first proposed an additive approach that first estimated the mean number of genes for one single genome out of a set of k genomes and then iteratively added the numbers of expected singleton genes for $2 \ldots k$ genomes. In a more recent publication Tettelin et al. (2008) proposed a more accurate formula to predict the pan genome development using Heaps' power law (Heaps, 1978). The extrapolation of the pan genome size can be seen as a series of measurements where in each single new measurement (added genome) the number of new instances (genes) not observed before is estimated. This continued sampling process is well studied in information retrieval,

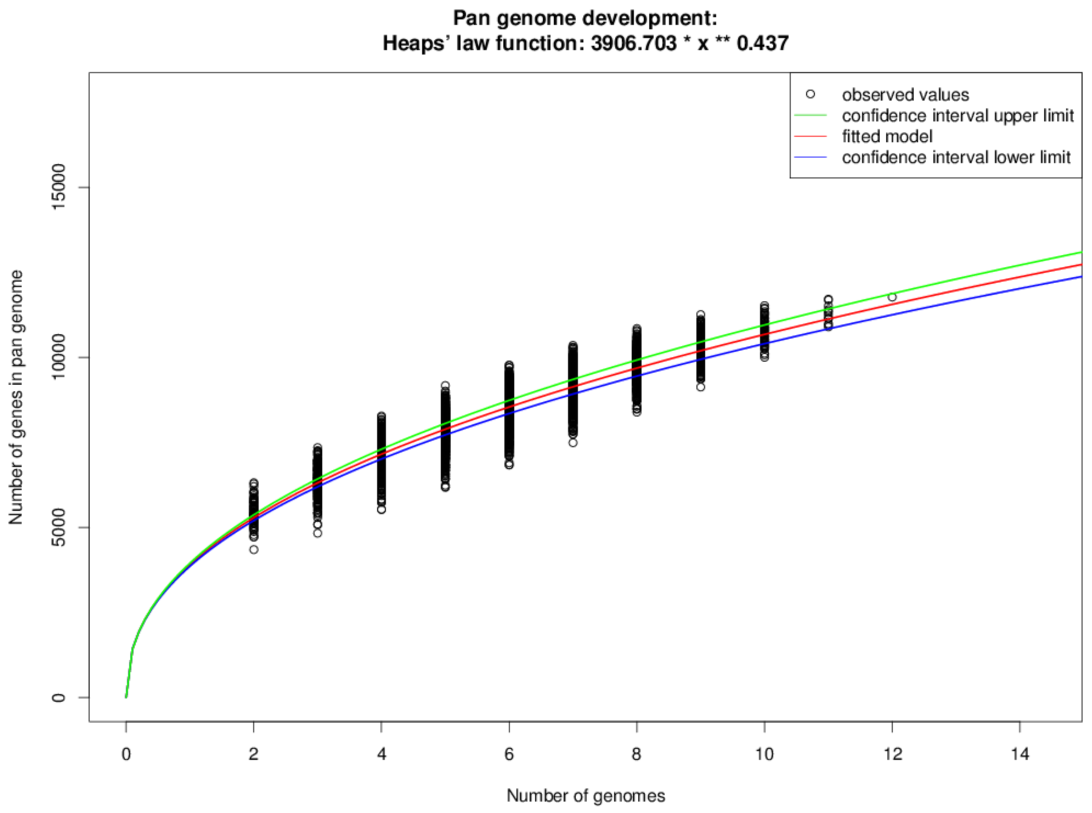


Figure 4.22.: Plot of the observed pan genome sizes and the predicted pan genome development for 12 *Xanthomonas* genomes. The black dots are pan genome sizes calculated for all permutations of the 12 genomes. The red curve is the model calculated according to Heaps' law from this values, the green and blue curves show the 95% confidence interval of the fitted model.

e.g., in text analysis where, when an increasing number of texts is analyzed, the number of different words grows according to a sub-linear power law of the total number of scanned words. The development of the pan genome shows a comparable development and can be extrapolated by a power law of the form $f(n) = k \cdot n^{\gamma}$, where n is the number of compared genomes, k is a proportionality constant and γ the growth exponent. Like in the core genome and singleton statistics the parameters k and γ can be estimated by non-linear least squares curve fitting. An examples of a pan genome size extrapolation for 12 *Xanthomonas* genomes is shown in Figure 4.22. The rate α at which new genes are added to the pan genome is proportional to the growth exponent γ with $n^{\gamma-1} = n^{-\alpha}$, with $\alpha = (1 - \gamma)$. If $\alpha > 1$ ($\gamma < 0$) the pan genome size asymptotically approaches a constant as more and more genomes are sequenced, i.e., the pan genome is closed. Conversely, if $\alpha \leq 1$ (it follows that $0 < \gamma < 1$), the function of the size of the pan genome is unbounded and increasing and thus the pan genome is open.

4.2.3. Results

The introduction of next generation sequencing technology has caused a rapid increase in the number of completely sequenced genomes. This development made it feasible to not only analyze single genome sequences, but to analyze large groups of related genomes in a comparative approach. As described, a main task in comparative genomics is the identification of orthologous genes in different genomes and the classification of genes as core genes or singletons.
With EDGAR a new tool was designed to support these studies and to automatically perform genome comparisons in a high throughput approach. EDGAR provides novel analysis features and significantly simplifies the comparative analysis of related genomes. By providing phylogenetic analyses for every project, EDGAR supports a quick survey of evolutionary relationships. Furthermore, with contig groups and metacontigs it is the only available software that allows comparative analyses on a higher level. EDGAR thus simplifies the process of obtaining new biological insights into the differential gene content of related genomes. Analysis and visualization features, like set calculations, phylogenetic trees, synteny plots, or Venn diagrams, are offered to the scientific community through a web-based and therefore platform independent user interface (`http://edgar.cebitec.uni-bielefeld.de`), where the precomputed data sets can be browsed.

The EDGAR web application

The features of the edgar web interface were described in detail in Sections 4.2.2.3 ff., in this section some statistics about the usage of EDGAR by the scientific community will be provided. As mentioned in Section 4.2.2.1, one intention of EDGAR was to provide a public resource to the scientific community that allows comparative analyses of all public available genomes. For this purpose, all genus groups of the NCBI genomes database with more than three sequenced strains were processed using the EDGAR pipeline, and the resulting orthology information was made available to the scientific community as public projects. When EDGAR was published in 2009 (Blom et al., 2009), the public database comprised 75 projects with 582 genomes in total. After the last update in October 2012, the database provides public projects for 130 genera with 1449 genomes. The use of the public databases is provided as a free service without any access restrictions. Thus, EDGAR has become a valuable resource for scientists all over the world to obtain information about the differential gene content and the phylogenetic relationship of bacteria.

The popularity of EDGAR is also reflected by the increasing number of private projects. Since 2009, more than 100 private projects with confidential unpublished genomes have been created. Currently, there are 101 private projects still in use which contain 1461 genomes[9]. Of these 101 projects, only roughly a third belongs

[9]as at 01.12.2012

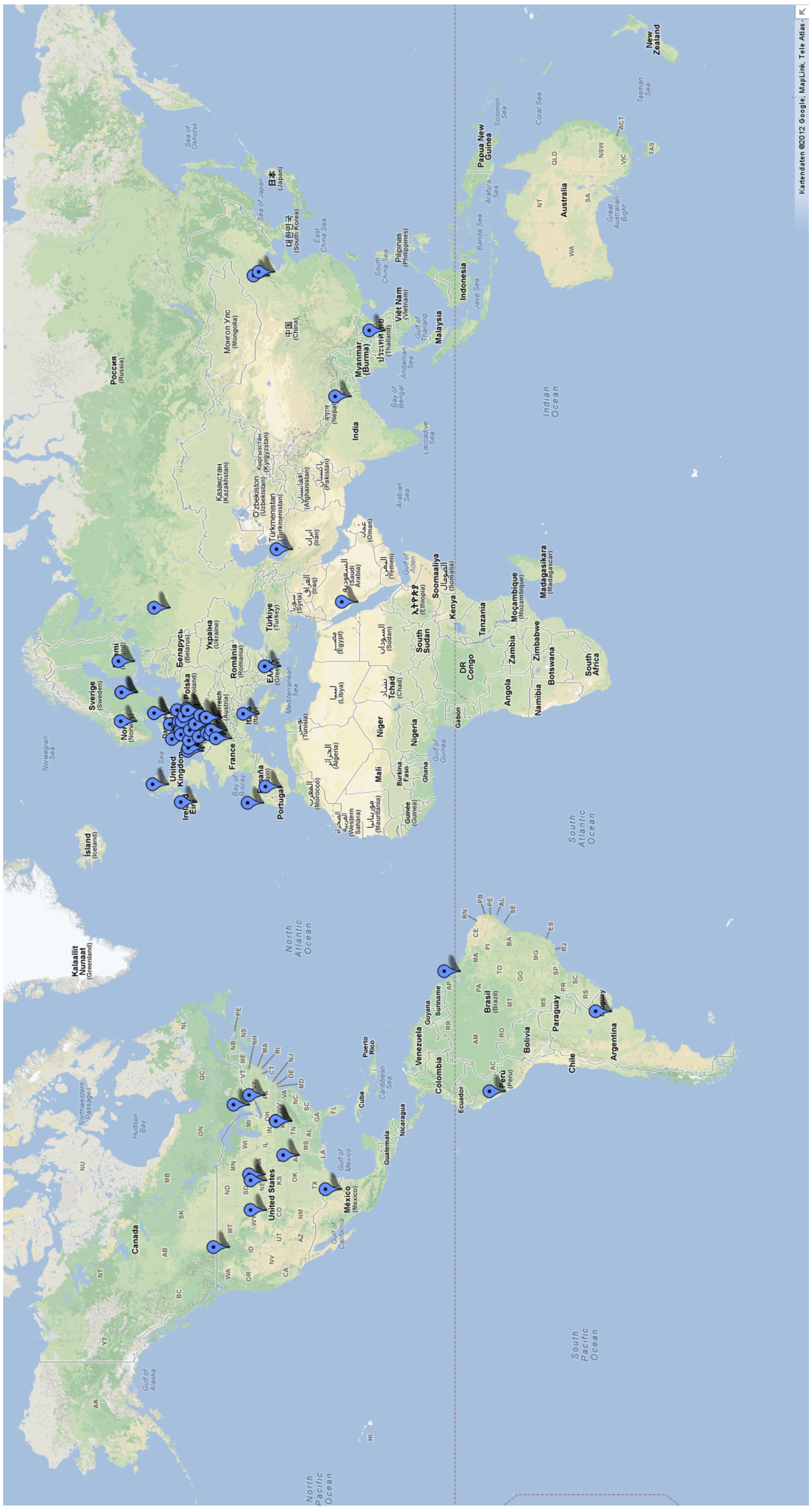


Figure 4.23.: EDGAR users worldwide: The blue symbols on this world map represent institutions that have public EDGAR projects in use. The majority of private projects was created for users in Germany and central Europe, but there are also plenty of projects in North and South America and in Asia. Map data from Google Maps, ©2012 Google, MapLink, TeleAtlas.

to different work groups from the CeBiTec, the rest was requested by scientists from nearly sixty scientific organizations all over the world. This worlwide acceptance of EDGAR is illustrated in Figure 4.23, where the location of external EDGAR users is marked on a world map.
Due to the wide range of scientific institutes using the service, EDGAR was applied to a multitude of different bacterial species and genera. Some of the most remarkable specific application cases will be presented in the following section.

4.2.4. Application cases

4.2.4.1. Corynebacterium diphtheriae

Corynebacterium diphtheriae is a human pathogenic species of bacteria and causative agent of the communicable disease diphtheria, an illness of the upper respiratory tract. The availability of a toxoid vaccine helped to control diphtheria very effectively in developed countries, but with the collapse of the Soviet Union the proportion of immunized people in eastern European countries dropped significantly and the number of diphtheria infections increased dramatically (Galazka et al., 1995).
Despite the great medical importance of *C. diphtheriae*, the knowledge about the molecular mechanisms contributing to the pathogenicity and virulence of *C. diphtheriae* isolates is limited. In a recent study Trost et al. (2012) presented a large scale comparative analysis of 13 *C. diphtheriae* strains isolated from patients with classical diphtheria as well as endocarditis, or pneumonia, complications that can occur during an infection with *C. diphtheriae*. One genome was a previously sequenced reference strain, the remaining 12 genomes were sequenced for this study on a Genome Sequencer FLX with Titanium chemistry (see Section 2.2.1), annotated using GenDB (Meyer et al., 2003) and compared using EDGAR. In a first step EDGAR was used to normalize the functional annotations of the sequenced genomes via the comparative annotation feature described in Section 4.2.2.7. Subsequently the evolutionary relationship of the 13 *C. diphtheriae* strains was analyzed using an MLST approach as well as the core genome based analysis of EDGAR. The comparative analysis of the gene content showed that the pan genome of *C. diphtheriae* comprises 4,768 CDS. Statistical analysis of the pan genome and the singleton development showed an open pan genome. The extrapolation of the pan genome according to Heaps' law (see Section 4.2.2.8) showed a constantly rising pan genome size with a growth exponent $\gamma = 0.31$ and thus a growth ratio of $\alpha = 0.69$. The open pan genome was confirmed by the singleton development plot which predicts 65 new singletons for every newly sequenced genome (see Figure 4.24).
The 4,768 CDSs in the pan genome comprise 1,632 CDSs that build the core genome of the 13 strains as well as 793 genes that are singletons in their respective genome. Thus, the number of genes in the dispensable genome is 2,361. The core genome comprises $\sim$70% of the gene content of one organism, and the pan genome has about three times as many genes as the core genome. This is a quite low value

related to the high genomic stability of corynebacterial species which lack recombination enzymes involved in genomic rearrangements. In comparison, a study on 20 *Lactobacillus* genomes found only 383 core genes which accounts for 15-20% of the gene content, while the pan genome comprised 13382 genes, which is nearly the 35-fold core genome size (Kant et al., 2011). The small pan genome is consequence of the low number of dispensable genes, which represent only $\sim$ 30% of the pan genome size. In a comparison of 61 *Escherichia coli* genomes the proportion of the dispensable genes in the total pan genome was $>$ 90% (Lukjancenko et al., 2010). The number of singletons is as well surprisingly low considering that the sequenced genomes were isolated in different countries, at varying timepoints, and from different diseases.

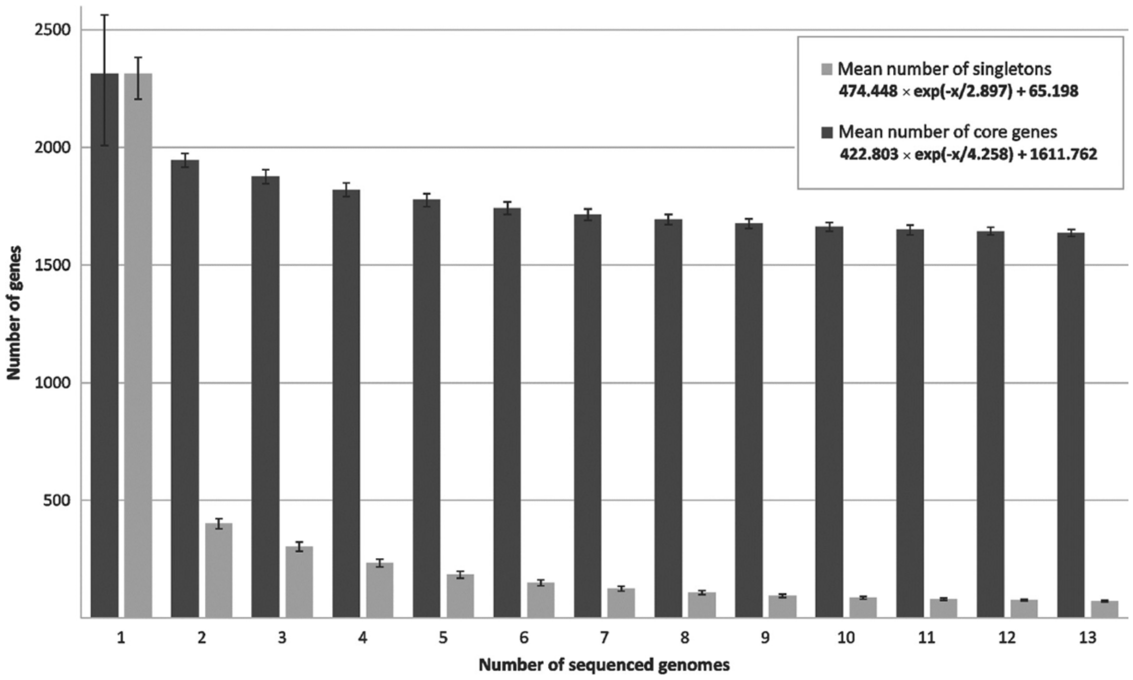


Figure 4.24.: Core genome size and singleton development: The bar plot shows the development of the number of core genes and singletons for 1-13 *C. diphtheriae* genomes. The further development of these numbers was predicted as described in Section 4.2.2.8, the extrapolated functions are shown in a box in the upper right part of the image. Image taken from (Trost et al., 2012).

The most important virulence factor of *C. diphtheriae* is an A-B exotoxin called diphtheria toxin. This toxin is encoded by corynephages, thus toxigenicity of *C. diphtheriae* strains requires a lysogenization by a *tox* corynephage. The expression of the *tox* gene is under bacterial control, although it is part of the phage genome. The sequenced genomes were screened for tox^+ corynephages, and 6 of 13 strains were identified to carry a tox^+ corynephage. For one of the analyzed strains, *C. diphtheriae* 31A, a so far unknown *tox* corynephage was detected. For the strain *C. diphtheriae* Park-Williams no. 8, which is a producing strain for the toxoid vaccine, it was found that a second copy of a *tox* phage was lysogenized. As *tox* is controlled by the regulator DtxR, the DNA binding sites of DtxR were detected by genome wide motif search. A comparative analysis of the DtxR regulons using EDGAR showed considerable differences caused by gained and partially or completely lost genes as well as by DtxR binding site depletion. Pathogenicity islands (PIs) predicted by the software PIPS (Soares et al., 2012) as well revealed characteristics of horizontal gene transfer. Furthermore it was shown that the PIs often encode subunits of adhesive pili, which can be important for the host interaction of *C. diphtheriae*, and that all sequenced isolates contain at least two pilus gene clusters. The pilus gene clusters identified in the 13 *C. diphtheriae* genomes showed a great variety. The CDS encoding the protein components of the pili, i.e., shaft protein, tip pilin, and base pilin, showed very divergent amino acid sequences and were mostly classified as singleton genes. This emphasizes that there is a high diversity among the pilus gene clusters which influence the process of adhesion through variations of the cell surface. A further analysis of these processes is crucial for an understanding of the mechanisms of toxigenicity and the initial steps of infection.

4.2.4.2. Plant pathogenic bacteria

The enterobacterium *Erwinia amylovora* is the causative organism of fire blight, a devastating disease of rosaceous plants. This organism has a huge economic impact as it can affect most pome fruit trees (i.e., apple, pear, and quince) and thus is the most important threat to this fruit production, globally. Consequently, *E. amylovora* has quarantine status outside North America, and there are severe hindrances in fruit trade between affected countries and fire-blight-free countries. Scientists from the Agroscope Changins-Wädenswil (ACW), a research station of the Swiss Federal Department of Economic Affairs, sequenced the complete genome of *E. amylovora* CFBP 1430 and compared it to *Erwinia pyrifoliae* DSM12163, and *Erwinia tasmaniensis* Et1/99 using EDGAR (Smits et al., 2010). In this comparison of the three genomes new virulence factors were identified in all three genomes, even in *E. tasmaniensis* which is considered nonpathogenic. Several virulence factors are found uniquely in *E. amylovora* CFBP 1430 and may contribute to the highly pathogenic phenotype of this strain. Furthermore it was shown that *E. amylovora* and *E. pyrifoliae* share a common set of virulence factors. Although both species have a different host range and virulence, the symptoms caused by the strains are essentially indistinguishable, and thus both strains have to be seen as fire blight pathogens.

Among the specific virulence factors of *E. amylovora* CFBP 1430 are two type 3 secretion systems (T3SSs) on genomic islands that show high homology to the insect pathogen *Sodalis glossinidus* str. *morsidans*. Furthermore, a chitinase-binding domain was found directly downstream of a T2SS in *E. amylovora* CFBP 1430. This may be a hint for an insect associated dispersal of the fire blight pathogens. Finally, several target genes for further analysis were identified using the genome comparisons, among them a T6SS cluster and polyketide synthase (PKS) and nonribosomal peptide synthetase (NRPS) clusters, multi-enzymatic, multi-domain megasynthases involved in the biosynthesis of nonribosomal peptides and polyketides (Ansari et al., 2004).

The successful cooperation with the ACW was continued in several follow-up studies in the field of plant pathogenic *Erwinia* strains. After *E. amylovora* CFBP 1430 the next sequenced strain was *Erwinia amylovora* ATCC BAA-2158, another pathogen restricted to *Rubus* plants (Powney et al., 2011). Another study analyzed the impact of the novel plasmid pEI70 found in several *Erwinia strains*. The removal of this plasmid reduced the aggressiveness of the treated strains, while the introduction in low-aggressiveness strains increased the symptoms caused by these strains (Llop et al., 2011). A comparative study of different *Rubus*- and *Spiraeoideae*-infecting *Erwinia amylovora* strains analyzed a certain pathogenicity island and thereby elucidated the evolution of the analyzed strain (Mann et al., 2012).
At ACW not only the *Erwinia* genomes themselves are studied, but also the biological control agent *Pantoea vagans*. Nevertheless, the use of *P. vagans* as a biocontrol agent is restricted in Europe due to the classification of *Pantoea* strains as opportunistic human pathogens. The strain *P. vagans* C9-1 is an antagonist to *E. amylovora* and in the analysis of the genome antibacterial peptides like pantocin A and dapdiamide E were identified. Virulence factors that could enable an animal or phytopathogenic lifestyle were not found (Smits et al., 2011). To analyze the pathogenic potential of *Pantoea* a clinical isolate of the strain *Pantoea ananatis* LMG 5342 was recently sequenced (De Maayer et al., 2012), the analysis with EDGAR is still in progress.
Another plant pathogenic organism is *Xanthomonas arboricola* pv. *pruni* (XAP), a pathogen of stone fruit plants that was sequenced at the ACW (Pothier et al., 2011). The plasmid pXap41 found in XAP was compared to 15 *Xanthomonas* plasmids and 19 complete genomes. The plasmid was only detected in isolates of the *pruni* pathovar, and a number of putative virulence factors was found on the plasmid, which suggests that the plasmid contributes to the virulence and/or fitness of XAP on *Prunus* species. Another branch of ACW research is the analysis of foodborne strains. In a cooperative project the genome of *Salmonella enterica* subsp. *enterica* serovar Weltevreden isolated from alfalfa sprouts was sequenced (Brankatschk et al., 2011). This genome sequence was compared to other *S. enterica* serovars using EDGAR (Brankatschk et al., 2012). The analysis revealed indications for host specificity, and serovar-specific genomic islands could be identified as well as differences in the presence of a T4SS. A unique plasmid pSW82 was identified that carries an unknown NRPS/PKS cluster, furthermore, an additional carbohydrate metabolism cluster was identified, possibly enabling this strain to survive on plant surfaces.

4.2.4.3. Mycobacterium

An ongoing study is the analysis of a selected set of 27 genomes of the genus *Mycobacterium*. The most well-known member of this genus is the human pathogenic species *Mycobacterium tuberculosis*, causing the tuberculosis disease which accounted for 1.4 million in 2011 alone (World Health Organization, 2012) (more information on this specific species is given in Section 5.3.4.1). Other Mycobacteria can metabolize polycyclic aromatic hydrocarbons (PAHs). PAHs are atmospheric pollutants and many PAHs are carcinogenic. They occur in oil, coal, tar, and as a byproduct of fuel burning. In cooperation with the National Center for Toxicology Research (NCTR) the *Mycobacteria* are currently analyzed in a large scale approach. Using comparative gene content analysis the metabolic potential of the selected *Mycobacteria* is studied with different points of interest:

- The life style of the strains can be free-living, facultative parasitic, or obligate intracellular. It is well known that when bacterial lineages make the transition from free-living or facultatively parasitic life cycles to permanent associations with hosts, they undergo a major loss of genes. This effect is clearly visible in an accumulative core/pan genome analysis (see Figure 4.25).
- *Mycobacteria* can be separated into slow-growing and fast-growing strains. It will be analyzed which genotypical features determine the difference in growth ratios.
- As mentioned there are severe pathogenic bacteria among the *Mycobacteria*, especially *M. tuberculosis*. The pathogenicity of *Mycobacteria* can be naturally linked to the life style of the organisms, furthermore a screening for virulence factors can be simplified by using comparative information.
- Finally, the *Mycobacteria* can be analyzed with regard to their xenobiotic metabolism, especially the potential for PAH degradation will be in focus of these analyses.

Furthermore, the Mycobacteria were grouped based on the results of phylogenetic analysis as well as a principal component analysis (PCA) (Jolliffe, 2005). Three clearly separated groups were identified by this approach, the first containing six PAH degrading *Mycobacteria*, the second consisting seven pathogenic *Mycobacterium tuberculosis* strains, and the last group including two *Mycobacterium leprae* genomes (see Figure 4.26). The first group is of special interest as a PAH degrading potential is not known yet for all genomes in this group, but based on their genomic features they are likely to have a PAH metabolic capability. This is currently verified in wet lab assays.
The metacontigs described in Section 4.2.2.4 were developed to support the analysis of these genome sets. The pan genome and gene set analysis by EDGAR provided a gene list of CDSs unique in the six PAH-degrading *Mycobacteria* that are at the same time not found in the 21 non PAH degrading Mycobacteria. In

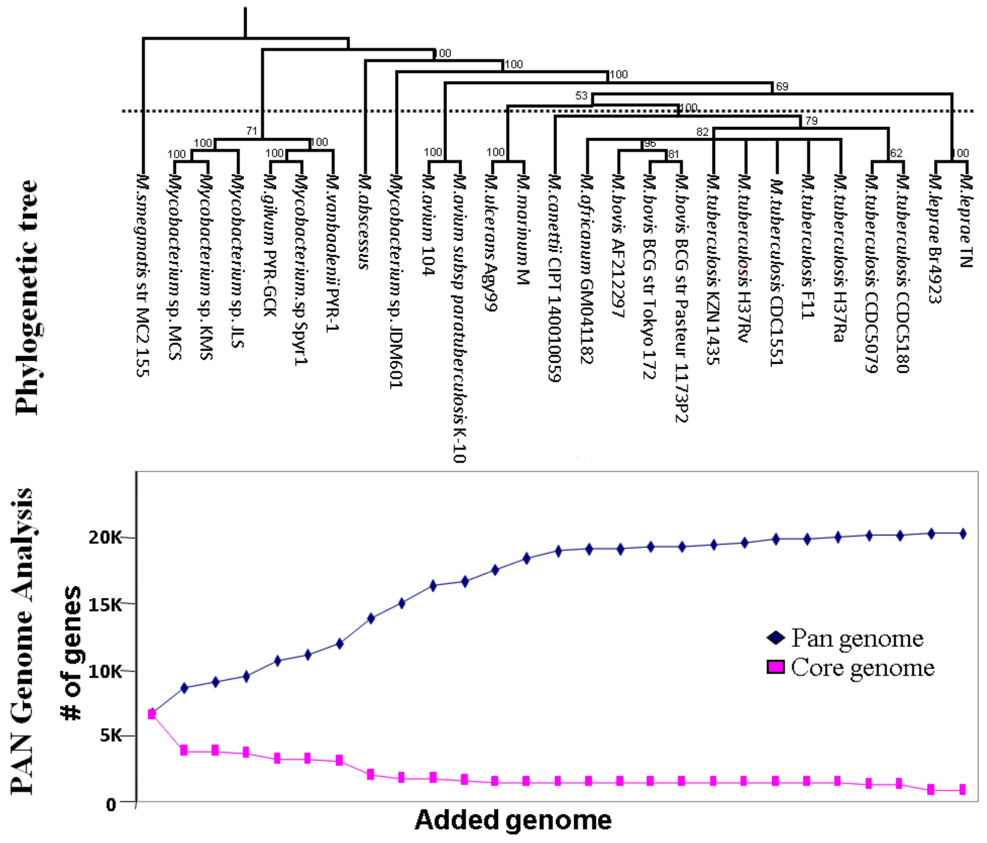


Figure 4.25.: Cumulative pan genome and core genome of 27 *Mycobacterium* genomes. Genomes displayed in the phylogenetic tree at the top are added to the pan/core genome calculation iteratively from left to right, and the observed gene count is plotted. It can be seen that the pan genome size is increasing fast at the beginning, but the number of genes added to the core genome becomes very low when the pathogenic strains of the *M. bovis/M. tuberculosis* clade are added. This is an effect of the genome reduction due to a changed life style as well as of the genomic stability of especially *M. tuberculosis*. This trend is also reflected in the core genome that drops significantly in size after the addition of the first pathogenic bacterium *M. abscessus*. Another noticeable drop in core genome size can be observed for the obligate intracellular *M. leprae* genomes.

a metacontig analysis the pan genome of the 21 non-PAH-degrading *Mycobacteria* was compared to the core genome of the six PAH degrading strains. This provided a list of 248 genes which are found in all genomes of the PAH group, but not in the 21 compared genomes. Logically, these genes are ideal targets for investigation of the PAH degradation potential. The further analysis will combine the data derived from EDGAR with additional genomic data like COG classifications or analyses of horizontal gene transfer with proteomic and metabolomic data as well as protein-protein-interaction (PPI) data.

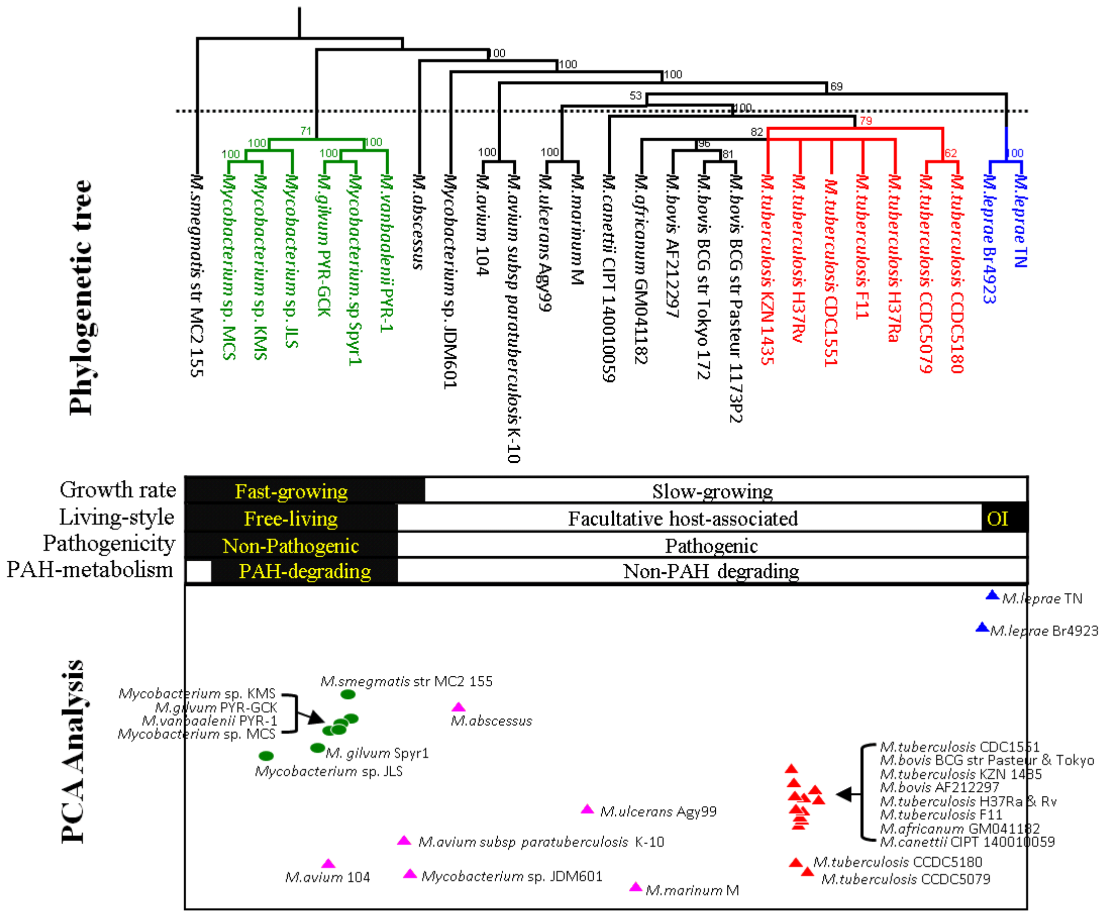


Figure 4.26.: Grouping of 27 *Mycobacterium* genomes based on phylogenetic analyses as well as a PCA. The phylogenetic tree is shown in the upper part above the genome list. Below the list the phenotypical features of the respective strains are shown. At the bottom part of the image the PCA clustering of the genome is shown, the groups are colored in green (PAH degreading bacteria), red (*M. tuberculosis* group), and blue (*M. leprae* group). Genomes that did not cluster together are colored magenta in the PCA image.

4.2.4.4. Further examples

As described in Section 4.2.3, there are dozens of ongoing projects in addition to the three elaborated examples. EDGAR was and is used in several microbial comparative studies, but as a detailed description of all ongoing projects would be to extensive, only finished projects with published results will be presented in the following list:

- ***Acinetobacter baumannii*** is an opportunistic pathogen which frequently causes nosocomial infections and is known for the multi-drug resistance of the main epidemic lineages. A comparative analysis of 12 *A. baumannii* strains was conducted in cooperation with the Istituto Superiore di Sanità (ISS) in Rome and showed that *A. baumannii* has an open pan genome and a high capacity to acquire new genetic elements, which helps the strains to adapt to the clinical environment (Imperi et al., 2011).

- The plant associated bacterium ***Bacillus amyloliquefaciens*** is in commercial use as it is plant growth-promoting and produces antibiotic and antimycotic substances that protect the plant from infection by other bacteria or fungi (Blom et al., 2012). In a phylogenetic study, EDGAR was used together with other techniques like DNA-DNA hybridization, 16S rDNA analysis, or microarray-based comparative genomic hybridization (M-CGH) to analyze the relationship among different Bacillus genomes. Based on the results, a discrimination of *B. amyloliquefaciens* into two subspecies *plantarum* and *amyloliquefaciens* was proposed (Borriss et al., 2011)

- In a recent study EDGAR was used to compare the plasmids of ***Clostridium perfringens*** isolated from chicken with necrotic enteritis. Several pathogenicity loci have been identified in this study (Parreira et al., 2012).

- Besides *Corynebacterium diphtheriae* EDGAR was used in the analysis of other ***Corynebacteria***, e.g. in the analysis of the pathogen *Corynebacterium aurimucosum*, which is associated with complications during pregnancy (Trost et al., 2010), or in the analysis of *Corynebacterium resistens*, a pathogen that is highly resistant to antimicrobial agents (Schröder et al., 2012).

- For the genus ***Lactobacillus***, EDGAR was employed in two different studies. In a first study, *Lactobacillus buchneri*, which belongs to the group of heterofermentative lactic acid bacteria and is a common member of the silage microbiome, was compared to other fully sequenced *Lactobacillus* genomes (Heinl et al., 2012). In a second study in cooperation with Helsinki University, 20 complete *Lactobacillus* genomes were analyzed to provide a comparative review of this genus (Kant et al., 2011).

- ***Neisseria meningitidis*** is a bacterial commensal living in the nasopharynx, that causes severe diseases like meningitis or septicemia in rare cases. A comparison of disease and carriage strains of *N. meningitidis* was the first study that used a script based prototype of EDGAR (Schoen et al., 2008). This study provided evidence that *N. meningitidis* emerged as unencapsulated human commensal and shares a common ancestor with *N. gonorrhoeae* and *N. lactamica*. This study was awarded with the price for molecular sciences on the Bielefeld science fair.

- The whole genome sequencing of ***Saccharothrix espanaensis*** was combined with a comparative analysis of all available *Pseudonocardiaceae*. It showed that *S. espanaensis* has a large number of singletons when compared to the family, of which nearly 50% have no orthologs in the public database (Strobel et al., 2012).

This list shows the diversity of bacterial organisms that were successfully analyzed with EDGAR. But EDGAR is not limited to bacterial species, actually the EDGAR software was integrated into the yeast genome analysis platform RAPYD (Schneider et al., 2011). There it provides the comparative genomics component in an integrated software for the annotation, comparative analysis and metabolic reconstruction of yeast genomes. Alltogether, the presented use cases demonstrate the wide range of application of EDGAR and prove that it became a valuable tool used routinely by scientist in different fields of interest.

CHAPTER 5

Comparative genomics on single nucleotide level: SARUMAN

The previous chapter described methods for comparative genomics on gene level and the development of EDGAR, a software to support such analyses. For some use cases however a comparison of genomes on gene level is not sensitive enough. As described in Section 4.2.4.3, a comparison of all available *Mycobacterium* strains showed nice results, but if one is interested in pathogenic *Mycobacterium tuberculosis* strains only, there are nearly no differences except single nucleotide exchanges (Ford et al., 2011; Niemann et al., 2009). For such cases a more sensitive approach is needed. The advent of the new high-throughput sequencing technologies allows for cost-effective sequencing of complete libraries of different bacterial strains, This progress may provide new insights, for instance into the microevolution of pathogens, but experimental data need to be processed before any conclusion can be drawn. Typically, the generated reads are mapped on a closely related reference genome to perform a targeted re-sequencing. For an in-depth SNP analysis the accuracy of this short read alignment is crucial, and due to the steadily increasing output of sequencing system also speed becomes more and more critical.
In this chapter the software SARUMAN ("**S**emiglobal **A**lignment of short **R**eads **U**sing CUDA and Needle**MAN**-Wunsch"), a fast and accurate short read alignment approach, will be presented. In the first part the meaning of short read alignment in modern sequence analysis will be introduced. In the second part existing approaches for short read alignment will be introduced and the characteristics of the different approaches will be discussed. In the third section of this chapter the development of SARUMAN will be explained in detail. This will include in-depth description of the mapping algorithm of SARUMAN as well as a comprehensive

evaluation in terms of runtime and accuracy. This Section is based on the publication of SARUMAN in Bioinformatics (Blom et al., 2011). The chapter will close with some application examples.

5.1. Short read alignment in modern sequence analysis

As mentioned in the introduction to this chapter, fast and accurate short read alignment is essential in the analysis of single nucleotide differences between closely related genomic sequences. This process is also referred to as "short read mapping" or "short read matching". In the following these terms are used as synonyms. The usual procedure for one of the most common applications, the identification of SNPs, starts with a mapping of all sequenced reads to the reference sequence with a defined error tolerance. The error threshold should be chosen with regard to the intrinsic error of the used sequencing technique - up-to-date sequencing system have error rates of below 1% (Illumina (Minoche et al., 2011)) up to 15% (Pacific Biosciences real-time sequencing, (Eid et al., 2009)) - as well as the expected degree of similarity between reference and template sequence. Reads that can be mapped to the reference are "piled up", and if the majority of reads shows a different base or an indel at a certain position, a SNP can be called (see Figure 5.1).

The approach is similar to re-sequencing, where the reads are piled up on the reference sequence by short read alignment and a consensus of the re-sequenced strain is generated by majority decision for each position. Apart from SNP analysis short read alignment is of vital importance in several other fields of sequence analysis, e.g., in transcriptomics or the analysis of protein-DNA interactions.
RNA sequencing (RNA-seq), the analysis of steady state RNA using next generation sequencing techniques, is a highly efficient technique for transcriptome profiling (Wang et al., 2009). RNA-seq is based on the sequencing of complementary DNA (cDNA) synthesized from the mRNA found in an organism at a given time. The resulting reads can be mapped to the reference genome, and the mapping coverage of annotated genes on the reference shows the expression levels of the respective genes (see Figure 5.2). This technique not only provides a detailed view of the transcription levels of known genes, but also allows the correction of transcription start points (TSPs), the identification of formerly unknown genes, and the detection of operon structures. Furthermore it also enables identification of formerly unknown transcripts like small non-coding RNAs and leader peptides (Raabe et al., 2011). An additional advantage of RNA-seq is that the sequence data can measure expression levels for every gene without bias caused by sequence-specific differences in hybridization efficiency in microarray-based methods and sequence data allow one to measure gene expression more accurately with a much higher dynamic range (Passalacqua et al., 2009).

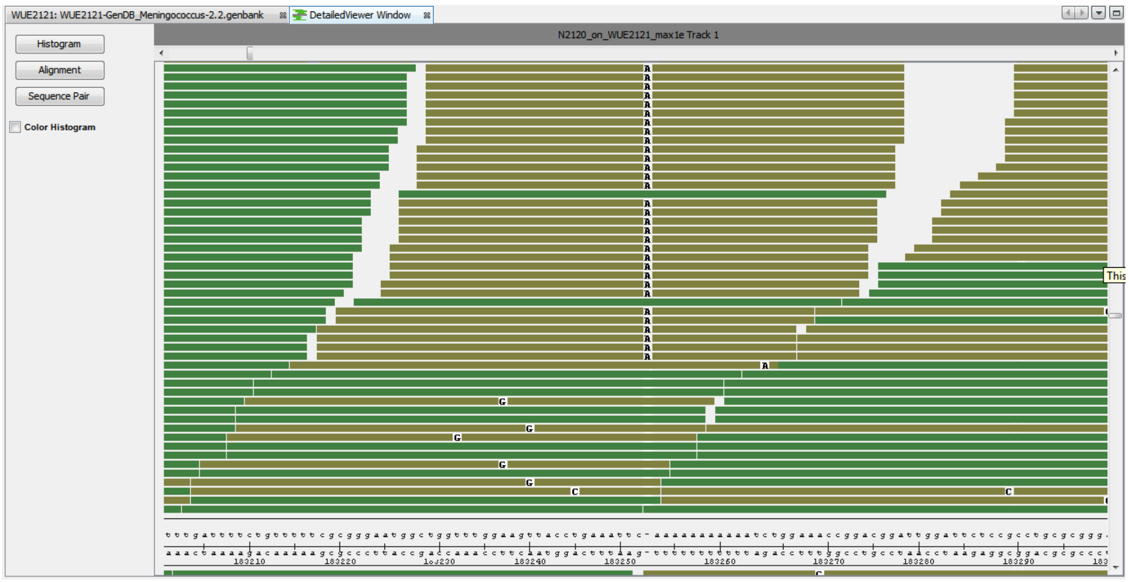

WUE2121: WUE2121-GenDB_Meningococcus-2.2.genbank
DetailedViewer Window
N2120_on_WUE2121_max1e Track 1
Histogram
Alignment
Sequence Pair
Color Histogram

Figure 5.1.: Result of a read mapping and SNP calling visualized as piled-up reads. At the bottom of the plot the reference sequence is displayed. The greenish horizontal bars above reflect reads that were mapped to the respective position. Dark green denotes perfectly mapped reads, while khaki colored reads were mapped with errors. It shows that several reads show an "A" at the identical position (183253), indicating an insertion at this position. Image created with VAMP, an alignment viewer currently developed by Rolf Hilker at the CeBiTec, Bielefeld University

Another application of short read matching is ChIP-sequencing (ChIP-seq)(Johnson et al., 2007), the combination of chromatin immunoprecipitation (ChIP) with high throughput sequencing which is used to identify the binding sites of DNA-associated proteins. Chromatin immunoprecipitation is a technique to purify DNA that interacts with a certain protein of interest. In the ChIP technique a DNA strand is sheared by sonication in vivo, subsequently protein-DNA-complexes are purified from the cell lysate. Such complexes are immunoprecipitated, i.e. captured by bead-attached antibodies specific for a DNA-binding protein of interest. The proteins are unlinked and the DNA fragments, which are supposed to be associated with the protein of interest in vivo, are purified and sequenced. If these fragments are aligned to complete genome sequence of the source organism all binding sites of the analyzed protein can be estimated. ChIP-seq is used to determine the binding sites of transcription factors and other chromatin-associated proteins. Protein-DNA interaction plays a decisive role in gene expression and thus allows the analysis of many biological processes and diseases as well as epigenetic chromatin modifications.
Due to the manifold applications of short read matching, different tools for mapping

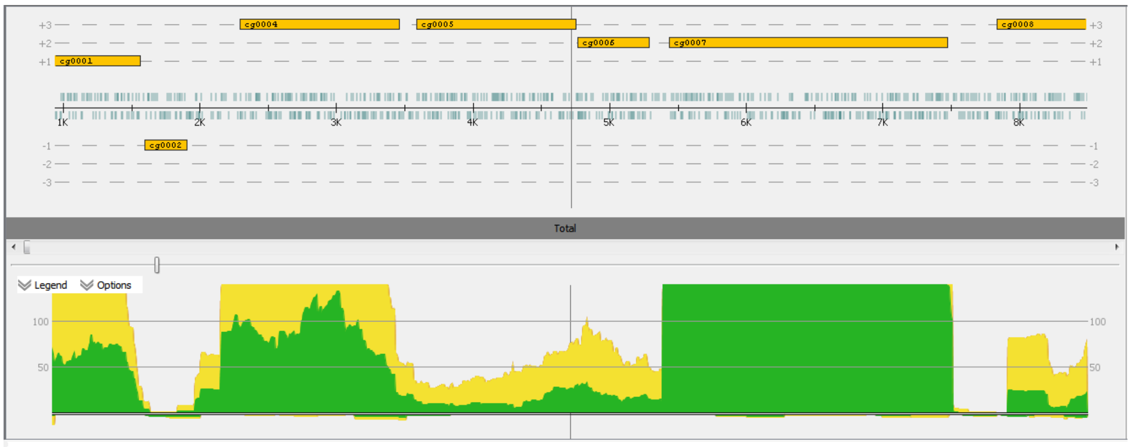


Figure 5.2.: Result of a read mapping of an RNA-seq data set. The upper part of the plot shows the annotated genes of the reference genome, in this case *Corynebacterium glutamicum*, as yellow boxes. The lower part shows the mapping coverage, where green color shows coverage by perfect matches and yellow color shows coverage by reads that have errors. One can see that coverage peaks often match annotated genes very well, e.g., for cg0001, cg0004, and cg0007. For cg0002 there is no coverage, thus this gene seems not to be expressed. There is contiguous coverage for genes cg0004 to cg0007 which indicates that these genes are co-transribed. Image created with VAMP (see Figure 5.1)

reads against reference genomes are available at this time which will be presented in detail in the following section.

5.2. Previous approaches

As described in the previous section, an accurate and reliable short read alignment is crucial for genome comparisons on SNP level as well as for several other applications like re-sequencing, RNA-seq, or ChIP-seq. Thus, it is not suprising that various dedicated short read alignment tools were developed and published since the introduction of the first short read sequencing technique (see Section 2.2.2). With seed-based algorithms and approaches relying on the Burrows-Wheeler transform (BWT), there are two basic principles that are used by the vast majority of short read aligners. This section will present an overview of the most popular short read alignment tools of either technique. All software tools presented in this section will be evaluated in terms of accuracy as well as runtime in Section 5.3.3.2.

5.2.1. Seed based approaches

An established and popular approach for efficient approximate string matching is to split the process into two steps. In the first step small, exact matches of substrings

of the query and the template are searched using exact string matching techniques; such exact partial hits are called "seeds". In a second step these seeds are used as starting point for an extension of the alignment, using e.g. dynamic programming approaches. There are numerous different seeding strategies that differ in the number of required seed matches, the use of contigous or spaced seeds, or the seed length, but most solutions are based on one of the two following principles, or combinations thereof: The *qgram lemma* states that two strings P and S with an edit distance of e share at least t qgrams, that is substrings of length q, where $t = max(|P|, |S|) - q + 1 - q \cdot e$.
That means that every error may destroy up to $q \cdot e$ overlapping qgrams. For non-overlapping qgrams one error can destroy only the qgram in which it is located, which results in the applicability of the *pigeonhole principle.* The pigeonhole principle states that, if n objects (errors) are to be allocated to m containers (segments), then at least one container must hold no fewer than $\lceil \frac{n}{m} \rceil$ objects. Similarly, at least one container must hold no more than $\lfloor \frac{n}{m} \rfloor$ objects. If $n < m$, it follows that $\lfloor \frac{n}{m} \rfloor = 0$, which means that at least one container (segment) has to be empty (free of errors). Moreover, if $n < m$ this holds for at least $m - n$ segments. Seed-based approaches are widely used in all fields of sequence comparison, e.g., the most popular alignment tool BLAST uses a seed-and-extend algorithm. Consequently, the problem of aligning short sequencing reads to long reference sequences was adressed by some seed-based alignment softwares. In the following section three of the most well-known dedicated short read alignment solutions that use seed-and-extend strategies are presented.

5.2.1.1. SHRiMP

SHRiMP, the **SH**ort **R**ead **M**apping **P**ackage (Rumble et al., 2009), relies on three algorithmic principles: spaced seeds (Califano and Rigoutsos, 1993), qgram filters (Rasmussen et al., 2006), and an accelerated Smith-Waterman implementation using SIMD techniques. Spaced seeds are a variation of the classical exact matching seeds that allow mismatches at defined positions of the seed (for a good description of spaced seed see (Ilie and Ilie, 2007)). Spaced seeds are often represented as strings of "1" and "*", where "1" denotes a position where a matching base is required, while "*" denotes a wildcard where a match or mismatch is allowed. As an example, the spaced seed "111**1**111" requires matches at positions 1-3, 6, and 9-11, has length 11 and a weight of 7, where the weight is the number of required matches in a spaced seed. Usually several spaced seeds of identical weight are used at the same time, SHRiMP provides several default sets of spaced seed which are shown in Table 5.1. Q-gram filters, as introduced by Rasmussen et al. (2006), require multiple seed hits in close proximity to identify possible matches. SHRiMP requires two spaced seed hits per 40bp window to define a possible match.

Such possible matches are verified by a vectorized Smith-Waterman implementation comparable to that of Farrar (2007) that can rapidly compute the maximum alignment scores using an SIMD implementation. This SIMD implementation has

Table 5.1.: Spaced seeds: Default spaced seeds used by SHRiMP for seed weights from ten to twelve. Defaults are available for seed weights up to 18.

weight	default spaced seeds
10	"11111**11111", "1111**11***1111", "1111**1**1**1**111", "111**1***1****1**1111"
11	"1111**1111111", "11111**11***1111", "1111**1**1***1**1111", "111**11**1****1**1**111"
12	"1111*1111*1111", "1111*111**1****1111", "1111****11**11*1111"

a three- to five-fold speed-up compared to unvectorized implementations. For candidates with a sufficient alignment score the actual alignments are computed in a final step.

5.2.1.2. PASS

PASS, a "**P**rogram to **A**lign **S**hort **S**equences" (Campagna et al., 2009), uses a classical seed based genome indexing approach in its first algorithmic step: An index of spaced seed words is created for the reference sequence, and reads are scanned for seed words also occurring in the genome index. In a second step PASS tries to extend these initial seeds to full length hits. For this purpose it uses precomputed score tables (PSTs) for all possible short sequences of a given length aligned to each other with defined alignment metrics. For fast access these PSTs are stored in main memory, allowing the rapid comparison of two sequence fragments flanking an initial seed hit. Thus, the execution time rises linearly with the read length as the number of PST queries is dependent on the read length, but at the same time the use of PSTs makes the runtime performance of PASS independent from the number of allowed gaps/errors as no alignments have to be computed. A drawback of PASS is that the PSTs are very memory consuming and can be applied only up to a very limited length. A PST for 8bp sequences needs approximately 4 gigabytes of memory, a 9bp PST would need almost 70 gigabytes.

5.2.1.3. mrFAST & mrsFAST

Two further seed-based short read mapping algorithms are mrFAST (Alkan et al., 2009) and mrsFAST (Hach et al., 2010), where mrsFAST supports only substitutions, while mrFAST also supports indels. Both methods use a classical seed-and-extend approach. Based on the number of allowed errors e, each read is partitioned into $\lceil \frac{readlength}{e+1} \rceil$ non-overlapping segments of length k, called k-mers. An index of all such k-mers in all reads is generated, and a second index is created for all overlapping k-mers found in the reference sequence, storing the positions of the k-mers within the reference.
By the pigeonhole principle there has to be at least one k-mer from a read that has a counterpart in the reference if this read has a match in the reference sequence with not more than e errors. By comparing the read and the reference index, such seed pairs can be found and extended to full alignment hits in a final step. While this algorithm is a standard approach in string matching, the novelty of mrFAST and

mrsFAST is the implementation as a “cache-oblivious” algorithm, where “Cache-oblivious” means, that mrFAST and mrsFAST use a recursive divide-and-conquer technique to split the compute-intensive all-gainst-all comparison of the two indices into smaller sub-problems that fit into the CPU cache. This results in a significantly more efficient execution of the calculation.

5.2.2. Burrows-Wheeler transform approaches

A second popular approach for short read alignment is the use of indices based on the Burrows-Wheeler transform (BWT). The Burrows-Wheeler transform, invented by Michael Burrows and David Wheeler (Burrows and Wheeler, 1994), is a permutation of the characters of a given input string. Although the computation of the BWT itself does not compress the data, the BWT is widely used in compression algorithms, e.g., in bzip2[1]. This is due to the fact that substrings occurring several times in the text lead to sequences of identical characters in the transformed text, which is highly useful for compression. Furthermore the BWT is lossless reversible.

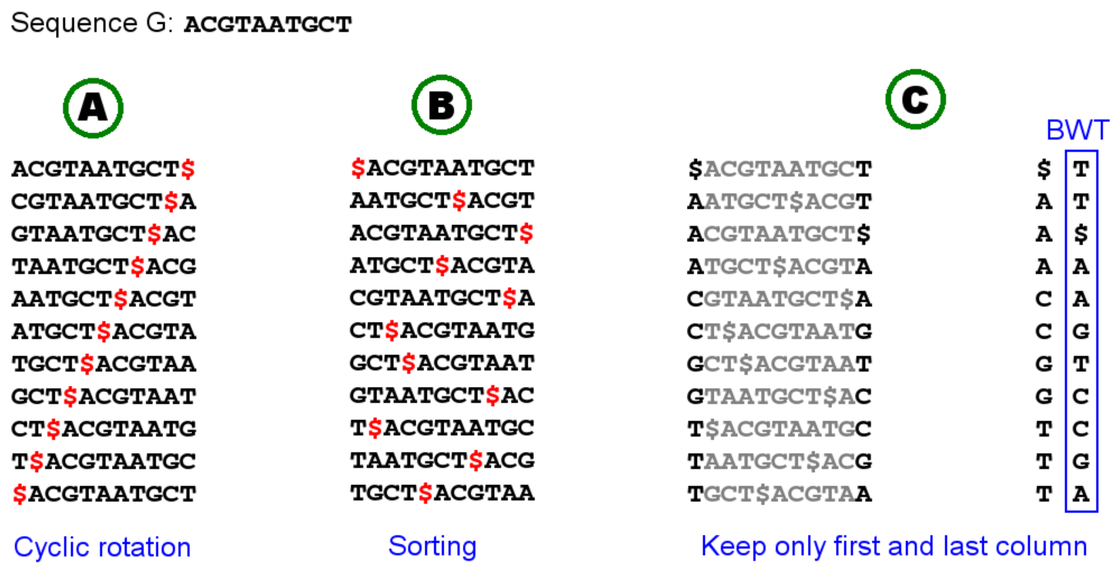


Figure 5.3.: Naive approach to create a Burrows-Wheeler transformation (BWT) of a text, in this case the DNA sequence G: “ACGTAATGCT”. (A) First a special character \$ is appended to the sequence, and a matrix of all cyclic rotations is created. (B) The matrix rows are sorted lexicographically. (C) All columns except the first and the last one are discarded. The last column is the BWT, the first column is needed for basic operation like the decoding of the BWT.

A BWT of a DNA sequence G of length u with the alphabet $\Sigma = A, C, G, T$ can be constructed in 3 steps which are illustrated in Figure 5.3. The described

[1]`http://bzip.org/`

procedure is a naive approach which is satisfactory neither in runtime nor memory requirements, but illustrates the general ideas behind the use of the BWT for pattern matching comprehensively. Efficient implementations are discussed in Section 5.2.3.
In a first step, a special character \$, which has to be lexicographically smaller than every character in Σ, is appended to the end of the sequence. Second, a matrix is created where all rows are permutations of the sequence G in a cyclic rotation, thus representing all possible suffixes of G (Figure 5.3.A). The rows are sorted in lexicographic order using a stable sorting algorithm (Figure 5.3.B). At the same time a suffix array S is created that stores for a suffix in row i of the matrix the starting position in G as $S(i)$. In the third step the last column of the matrix is taken as transformation B (Figure 5.3.C). For all operations on the BWT only the last and the first column are needed, all other columns can be removed.

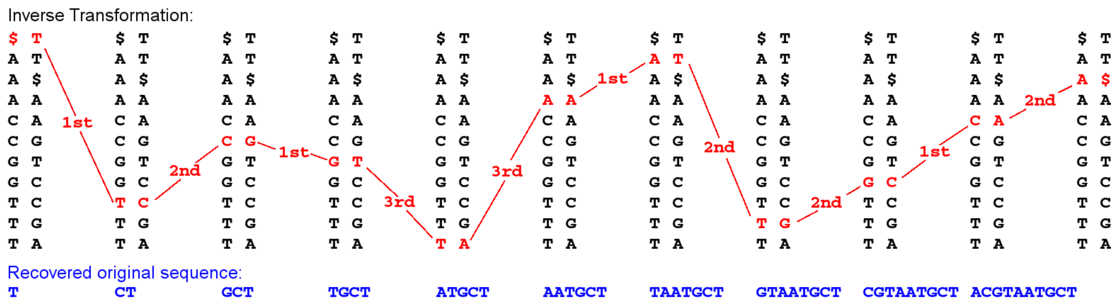


Figure 5.4.: Schematic representation of the decoding of a Burrows-Wheeler transformation (BWT) of sequence G using the first and the last column of the BWT matrix. By design the BWT starts with the last character of the original sequence, in this case a “T”. Due to the LF mapping this “T” in the last column (LC) is the same as the first “T” in the first column (FC). The “C” in the LC is the predecessor of the “T” in G, thus we have decoded the next to last character. The “C” is the second “C” in LC, we can use the LF mapping again to go to the second “C” in FC, uncovering the next preceding character “G”. We continue with this approach until we reach a \$ in LC which tells us that the complete sequence is decoded.

The BWT matrix has a special property called “last-to-first column mapping” or “LF mapping” in short. The LF mapping describes the fact that the ith occurrence of a character x in the last column corresponds to exactly the same character in G as the ith occurrence of x in the first column. This basic property allows the reversal of a BWT as well as the implemenation of string searching algorithms using the BWT. The lossless reversion of a BWT to regain G is depicted in Figure 5.4. Due to the fact that the construction of BWT uses cyclic rotations of G, another important property of the BWT is that the character at position i of the last column (LC) is the predecessor in G of the ith character in the first column (FC). Furthermore,

as by definition \$ has to be lexicographically smaller than every element in Σ we know that \$ will appear at the first position of FC. Thus, the first character of LC is the predecessor of the special character \$, the last character of G. This is our starting point to recreate the original sequence reversely from the end to the start. Using the LF mapping we can estimate the character in FC that corresponds to our start point, in the example in Figure 5.4 the first "T". The character in the LC at this index position is the predecessor of our starting point, in Figure 5.4 a "C". So we have uncovered the second character of G. Using the LF mapping we can jump back to the respective character of FC and estimate the predecessor of this character, and so on. This procedure is repeated until the estimated predecessor is \$, which means that the complete sequence was decoded.
The BWT can also be used for fast exact text matching, again making use of the sorted FC and the LF mapping behavior. Just like the decoding, the pattern search uses the reverse order of characters of search string $P = p_1...p_v$. The search starts with the block in FC that shows the character p_v (see Figure 5.5). In the next step all respective positions in LC are checked if they show the next search character p_{v-1}. All matching positions remain in the result set, and the next position is checked. If the complete search pattern is processed, the position indices from LC will show the matches of P in G. To get from the positions in LC to the original starting positions in G a suffix position index is needed. As it would be too memory consuming to store the starting position in G for every character in LC, the starting position is stored only for every l-th position in compressed form, while the starting positions for all other rows can be computed by LF mapping starting from the closest indexed row. It is possible to store such a compressed suffix position index using $O((u/l) \log u)$ bits of memory.

5.2.3. Full-text minute-space index

As mentioned, the methods for the creation and usage of a BWT described in the previous section represent a naive approach, e.g., one obvious optimization is that the first column has not to be stored actually, but it can be substituted if one counts the number of occurrences of each character within G during the creation of the BWT. As the first row is lexicographically sorted this information is sufficient to virtually recreate the first column.
The theoretical basis for the *efficient* use of Burrows-Wheeler transformations for indexing and compression of text and pattern search was introduced by Ferragina and Manzini (Ferragina and Manzini, 2000), who described a compressed BWT that uses only $O(H_k(G) + o(1))$ bits of memory per input symbol, where H_k is the k-th order empirical entropy of the text. This compression allows to keep an index of human genome size (approx. 3 Gb) in only 1.3 gigabytes of memory and can be created in linear time. In 2005 Ferragina and Manzini proposed the so-called "full-text minute-space index" (FM index) (Ferragina and Manzini, 2005). The FM index is a compressed full-text substring index based on the BWT, it allows fast text search in the compressed index. It allows to identify the number

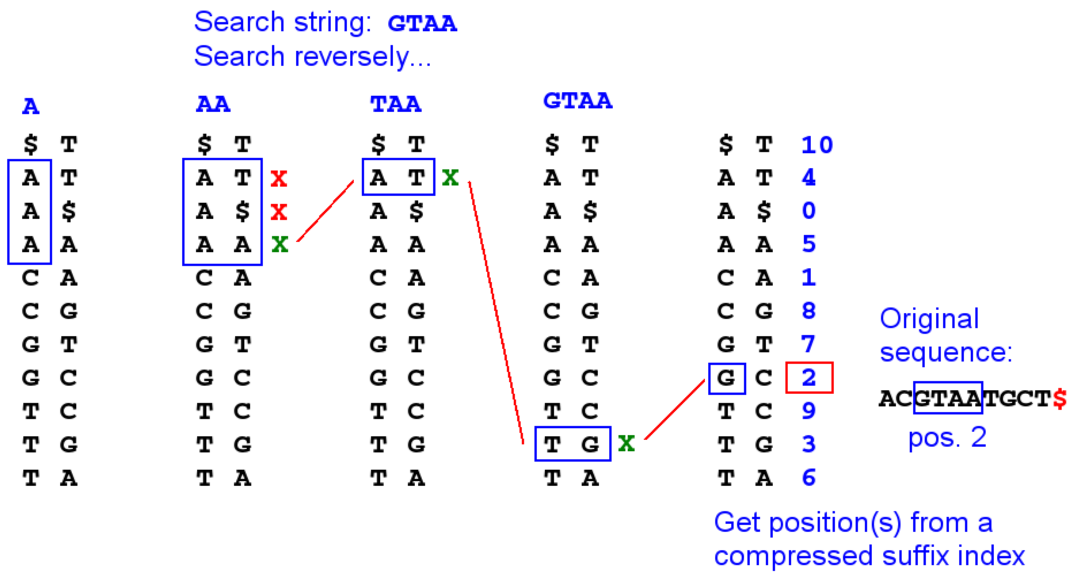


Figure 5.5.: Schematic representation of the exact text search with Burrows-Wheeler transformations. A given search string P to be found in a sequence G is processed from the end to the start. The block in the first column (FC) showing the last letter of the search string is the start point of the text search. For each row of this block the last column (LC) is checked if the character is identical to the second to last of P. For all matching cases, in this example only one after the first iteration, the LF mapping is applied to get the next block of FC position and the search step in LC is repeated. This is done until the complete search pattern is processed or there are no valid characters left. If the complete pattern is processed the position of the search string in G can be obtained from a compressed suffix position index that is stored during the creation of the BWT.

as well as the positions of *occ* occurrences of a text pattern of length v within the compressed text in $O(v + \log u)$ time. Thus it is perfectly suited for short read mapping. Since the introduction of the FM index it is extensively studied in the field of bioinformatics and applied by different short read alignment tools. In the spring of 2009 three of the most popular short read mapping tools were published, all based on BWT indices: Bowtie, BWA, and SOAP2. They will be presented in the following paragraphs together with CUSHAW, a recent approach that can use graphics hardware to accelerate the alignments.

5.2.3.1. Bowtie and Bowtie2

Bowtie, published in March 2009 by Langmead et al. (2009), was the first short read matching software to use the FM index developed by Ferragina and Manzini (2005). Bowtie makes some constraints to achieve a sufficient computing time. If the best

match for a given query sequence (read) is an inexact match, Bowtie can not guarantee to find the highest quality alignment (Langmead et al., 2009). Furthermore in default settings Bowtie reports only one alignment per read. If all alignments are required Bowtie gets significantly slower. The biggest drawback of Bowtie is that it does not support insertions or deletions (indels) within the alignments. This makes the tool unsuitable for all short read mapping applications where accuracy is crucial, e.g., in re-sequencing where frame-shifts would be lost.
To overcome this limitation, Langmead and Salzberg (2012) developed Bowtie2. Bowtie2 combines the BWT-based FM index with a seed-and-extend strategy. The fast and memory efficient FM index is used to find short exact matches in a seeding step. Ungapped seed alignments found by the FM index approach are prioritized, and good scoring seed alignments are extended into full alignments by a dynamic programming approach in an SIMD implementation. Due to this dynamic programming step the accuracy of Bowtie2 is significantly higher compared to the predecessor.

5.2.3.2. SOAP2

SOAP2 (Li et al., 2009b), the successor of the "**S**hort **O**ligonucloetide **A**lignment **P**rogram" SOAP (Li et al., 2008), uses a BWT compressed index as well. The algorithmic strategy of SOAP2 includes an indexing strategy that creates a hash table of the suffixes of the BWT index with a given seed size, e.g. with $4^{13} = 2^{26}$ entries for 13mers. This double indexing strategy allows the rapid identification of match positions for substrings of the given length and of exact read matches in general. For inexact matching that allows mismatches SOAP2 uses a "split-read strategy". This strategy splits the a read into $e + 1$ segments if e mismatches are allowed, such that according to the *pigeonhole principle* it is guaranteed that at least one segment has an exact match. For non-matching segments the BWT decoding steps have to be executed.
Due to its double indexing approach SOAP2 is one of the fastest short read aligners, but at the same time the method is not suited for the detection of indels in the reads. Although Li et al. (2009b) state that the split-read strategy works for mismatches as well as indels, in reality SOAP2 misses the vast majority of reads with indels (see evaluation in Section 5.3.3.2). Thus, SOAP2 obviously sacrifices some accuracy to gain a higher alignment speed.
SOAP3 as the third version of SOAP was published in 2012 (Liu et al., 2012a) and further accelerates the SOAP2 algorithm by the usage of GPU programming. This way SOAP3 achieves a 5-fold to 28-fold speedup above SOAP2 (according to the evaluation in (Liu et al., 2012a)). Unfortunately, the provided binaries of SOAP3 could not be executed on three different test systems, thus it could not be included in the evaluation.

5.2.3.3. BWA

BWA (Li and Durbin, 2009) is another BWT based short read mapping approach that uses an FM-index-like data structure. The unique selling point of BWA is that it has a much better support for reads with indels than Bowtie or SOAP2. BWA uses a heap-like data structure to mimic a breadth-first-search in a prefix trie to find all hits with a given maximal number of differences. The algorithm described in (Li and Durbin, 2009) guarantees to find all alignment positions of a read within a reference up to a defined error threshold, but the BWA developers made various modifications to it. These modifications brought a further speed-up, but at the same time added a heuristic aspect. To limit the search space of this trie search an array D is calculated that stores an estimated lower bound of differences up to the defined position. Based on this distance array certain paths in the prefix trie can be omitted. Furthermore only a limited number of mismatches is allowed in a defined seed sequence at the beginning of the read (default 32 bp).
The described strategy allows BWA to map short reads to a reference sequence with high speed and accuracy, and it is the first BWT-based short read matching application that has a proper indel support. But due to the described speed-up efforts the completeness of the underlying algorithm is forgone.

5.2.3.4. CUSHAW and CUSHAW2

Another recently published BWT-based short read aligner is CUSHAW (Liu et al., 2012b). CUSHAW again is based on the FM index data structure as proposed by Ferragina and Manzini, but the short read mapping is accelerated using graphics hardware via the CUDA framework. The algorithmic approach of CUSHAW is quite similar to BWA, but instead of using a breadth-first-search (BFS) on a prefix trie, CUSHAW uses a depth-first-search (DFS) as a BFS would be too memory consuming to be executed on a graphics card. Furthermore, instead of a prefix trie a complete four-ary tree of all permutations of the read sequence is used. Like BWA, CUSHAW uses a bounded search approach to minimize the search space, but the algorithm in its present form does not support insertions or deletions. To support a more exact alignment CUSHAW2 is available from the same developer team. CUSHAW2 does not support GPU usage via CUDA, but instead supports gapped alignments. Both versions will be evaluated in Section 5.3.3.2.

5.3. SARUMAN: Semiglobal Alignment of short Reads Using CUDA and NeedleMAN-Wunsch

In the previous section several approaches developed during the last years were presented that allow a fast alignment of short sequences to a given template. The majority of these methods use heuristic techniques to gain a speedup of the alignments, thereby missing possible alignment positions. Furthermore, most approaches return only one or a limited number of best hits for every query sequence, thus losing the potentially valuable information of alternative alignment positions with identical scores. As high accuracy and confidence of short read alignment is highly desirable, especially in the analysis of single nucleotide polymorphisms (SNPs) or small-scale structural variations, SARUMAN is designed as an exact short read alignment approach that identifies all match positions for each read under a given error tolerance and furthermore reports one optimal alignment for each of these positions. To obtain an adequate runtime even for large-scale, whole-genome applications, we employ the massively parallel compute power of modern graphics adapters (Graphics Processing Units - GPUs).

5.3.1. Design and Methods

SARUMAN is a seed-based method like the existing approaches described in Section 5.2.1, and the short read alignment process is designed as a two step procedure. The first step is a seed-based filter algorithm that identifies auspicious alignment positions for the reads in the reference sequence. The primary design goal for this filter step was to provide a result that is:

1. **Exact:** All reads that can be aligned to the reference are aligned.
2. **Complete:** All possible alignment positions under a given error threshold are found, not just the n best alignment positions.

With a required sensitivity of 100%, a perfect specificity can not be achieved at the same time at a tolerable runtime. Thus, the filter step gives not only valid results, but naturally also false positive alignment positions are reported. Thus, in a second step possible mapping positions identified by the filter step are verified by an alignment step that by design should be:

3. **Optimal:** For each verified match position one in a mathematical sense optimal alignment is reported.

For this step a CUDA implementation of the Needleman-Wunsch alignment algorithm is used that computes optimal alignments of the candidate read sequences to a clipped part of the reference sequence. The filter algorithm will be described in detail in the following Section 5.3.1.1, the alignment phase is described in Section 5.3.2.2.
Besides these two core components - filter step and alignment step - several additional modules are needed for the short read matching workflow:

- **Data import:** Modules for the import of reads as well as the reference sequence.

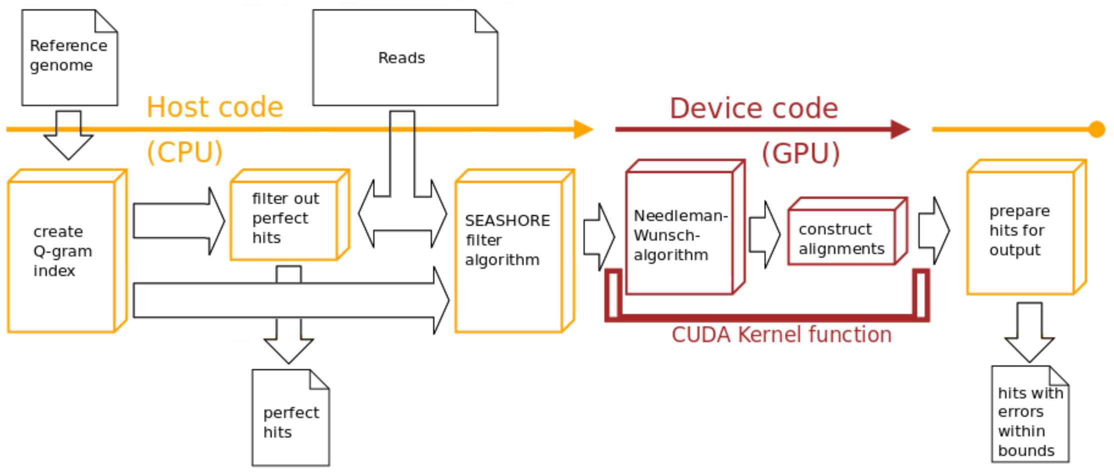


Figure 5.6.: Schematic representation of the SARUMAN workflow. Parts that are yellow colored are executed on the host computer's CPU, parts colored in red are executed on the GPU.
First, the reference genome is imported and an index of all qgrams in the reference of a defined length, which is calculated according to the number of allowed errors, is created. The reads are imported, and with help of the qgram index all perfect matches of reads against the reference genome are identified and exported. Subsequently, the qgram-index and the reads are used in a filter algorithm that searches for promising alignment candidates. These candidates are copied to the graphics card and an optimal alignment is computed using the Needleman-Wunsch algorithm with backtracing. The alignment results are copied back to the host computer's main memory and exported in a tabular format.

- **Qgram index:** For the seed-based filter step a qgram index (see Section 5.2.1) of all substrings of length q is needed.
- **Filter for exact matches:** Exact matches of whole reads against the reference can be identified easily and rapidly. These matches should be filtered out to be omitted in the more sophisticated and thus more compute intensive approximate matching process.
- **Data export:** Modules for the export of the computed short read alignments.

A diagram of the complete workflow of SARUMAN is illustrated in Fig. 5.6. The actual implementation will be described in Section 5.3.2.

5.3.1.1. Algorithm

SARUMAN was designed with the goal to provide an exact, complete and optimal solution for short read alignment (as defined in the previous chapter). A seed-based algorithm providing such a solution will be described in this chapter, starting with a formal definition of the short read matching problem.

Problem definition The short read matching problem can be defined as follows: We have given a short sequencing read f of length $|f| = m$, a (in most cases genomic) reference sequence g of length $|g| = n$, and an error threshold $e \geq 0$ defined by the user. Then we want to calculate all starting positions i in g, such that there exists an alignment of f and a prefix of $g[i...]$ with at most e errors (mismatches and/or indels). The algorithm shall be capable to export an optimal alignment for every such match position.
This problem has been studied widely in the past, and most solutions are based on one of the two following principles, or combinations thereof: The *qgram lemma* and the *pigeonhole principle* (see Section 5.2.1 for a definition of both principles). In our algorithm, presented in the following section, we first use a two-fold application of the pigeonhole principle to find regions of interest in the genome, before we apply parameters derived from the qgram lemma to make our final selection of positions to pass the filter.

Solution **Assumptions and definitions:** As a basic assumption we require a minimal length of reads in relation to the given error threshold: $m > e + 1$. Given this assumption, we calculate the length of the qgrams for our filter algorithm as follows. We define the length q of the qgrams as the largest value below $\frac{m}{e+1}$ such that

$$q' := \lfloor \frac{m}{e+1} \rfloor, \qquad q := \begin{cases} q' \text{ if } (e+1)q' < m, \\ q' - 1 \textit{ otherwise.} \end{cases}$$

This guarantees that a read f can be split into $e + 1$ intervals with an additional non-empty remainder of length $R := m - (e + 1)q$.

Given the calculated qgram length, we create an index I of the starting positions of qgrams in sequence g, such that for each possible qgram x, $I(x)$ contains the starting positions of x in g.
In the following step of the filter algorithm, every read is segmented into pieces of length q (see Fig. 5.7). We choose a set S of $c = \lfloor \frac{m}{q} \rfloor$ segments $S = s_1...s_c$ of length q from f, such that for $i = 1, ..., c : s_i = f[(i-1) * q + 1...i * q]$. As in most cases $q = \lfloor \frac{m}{e+1} \rfloor$ it can easily be shown that c converges against $e + 1$ for increasing read length m. We additionally choose a second set K of c segments $K = k_1...k_c$ of length q, such that for $i = 1, ..., c : k_i = f[(i-1) * q + 1 + R...i * q + R]$, a set shifted by the remainder R.

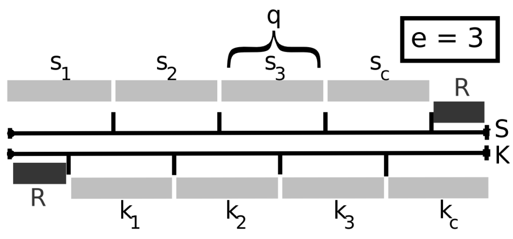


Figure 5.7.: The two sets of $e + 1$ segments of length q, $S = s_1...s_c$ and $K = k_1...k_c$. The two sets are shifted by the distance of R.

The simple version of our algorithm allows for mismatches only, not for insertion or deletions. In this case it can be shown by the following observations that a matching read f must have at least two matching segments from the sets S and/or K. There are two cases, in the first case two segments of set S will match, in the second case one segment from S and one segment from K will match. The term "matching segment" in the following indicates that a starting position for the respective segment can be found in the genome via exact occurrence within index I. The first segment found in the index is used as seed, the second "matching" segment has to be listed in the index in appropriate distance to this seed. The algorithm is based on the assumption that S and K comprise of $c = e + 1$ segments. In cases where $c > e + 1$, we know by the pigeonhole principle that at least two segments of S must match, which fulfills our filter criterion directly. The segment set K is not needed in such cases, and all algorithmic steps after the first can be skipped.

Filter algorithm: By the pigeonhole principle we know that one segment of set S **must** match, as we allow e errors and have $e + 1$ segments. We can identify all match positions of segments $s_i \in S$ for the given read in the qgram index I and use them as starting points for the filter algorithm.

1. For every segment s_i matching at a position b in the genome g we check in I if there is another segment $s_j \in S, j > i$, that starts at the expected position $b + (j - i) * q$. If we find such a segment we identified the two matching segments we expect.

In the case that only one segment $s_i \in S$ matches, there has to be exactly one error in every remaining segment $s_1...s_{i-1}, s_{i+1}...s_c$. Otherwise a second segment of set S would match. We can infer that not more than $c - i$ errors are remaining in the segments to the right of read s_i, but due to the overlapping construction of our segment sets we have one segment more of K to the right of read s_i to check. Hence, if read f is a possible hit, one of these $c - (i - 1)$ segments $k_i...k_c$ has to match, and we can start the checks for set K at position k_i.

2. Check if segment k_i overlapping s_i matches. If it does, we have successfully matched 2 qgrams and passed the filter.

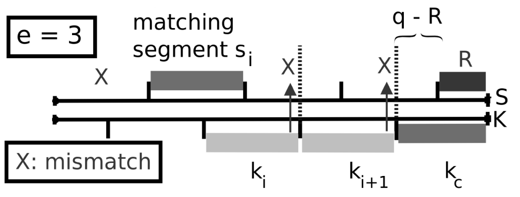


Figure 5.8.: A matching segment $s_i \in S$ where no other segment of S has a match in correct distance. Segments of set K are checked in this case. As the matching segment s_i is the second one of set S, there has to be at least one error prior to s_i, marked by an "X" in the image. If segments k_i and k_{i+1} don't match, two further errors must have been located in the part of the read prior to segment k_c (again marked by "X"), thus reaching the error limit of 3 in this example. If segment k_c does also not match, the read can not match at the respective genome position under the given error constraint.

If segment k_i does not match, we know that k_i overlaps s_i on $q - R$ positions. These positions are free from errors (as s_i matched without errors). Thus the error causing k_i not to match must have been on the last R positions of k_i which are the first R positions of s_{i+1}. As there is exactly one error in every segment of S we can conclude that the last $q - R$ positions of s_{i+1} are free of errors, which are the first $q - R$ positions of k_{i+1}. So if k_{i+1} does not match, the next error is in the first R positions of s_{i+2} and so on.

3. Iteratively check all remaining segments $k_{i+1}...k_c$ until one segment matches and the read f passes the filter or until k_c is reached.

If we reach k_c, that means that k_{c-1} did not match, we know that the error of s_c was in the first R positions. So the last $q - R$ positions of s_c must be correct, and so must be the first $q - R$ positions of k_c. The remaining R positions of k_c must also be correct because all errors are within segments $s_1...s_c$. The last R positions of k_c are not part of one of these segments, so any error within them would be the $e + 1$st one. If k_c does not match, it can be excluded that read f can be aligned to the reference sequence g with a maximum of e errors at the actual position. See Figure 5.8 as illustration of this matching process.

Insertions and Deletions It is possible to allow indels in the matching process by checking every segment not only on one position, but on several positions by shifting the segment to the left or right by up to t positions, where $t = e-(i-1)$ for an initial matching segment s_i as at least $i - 1$ errors have already occurred in the previous $i-1$ segments. Hence, we conclude that if there are only e errors in f, it is always possible to match (1) two segments of S or (2) one segment s_i of S and one segment of K. Algorithm 1 identifies reads as candidates for a Needleman-Wunsch alignment by checking for this condition.

Algorithm 1 (Mapper)

```
Require: A read $f$ of length $m$
Require: A reference sequence $g$ of length $n$
 1: {Set of candidate positions}
 2: $C \leftarrow \emptyset$
 3: {$q$ calculated as described in Section 2.2}
 4: $c \leftarrow \lfloor \frac{m}{q} \rfloor$
 5: $R \leftarrow m - c \cdot q$
 6: for $i \leftarrow 1, \ldots, c$ do
 7:   $u \leftarrow (i-1) \cdot q + 1$
 8:   {initial matching segments}
 9:   for each $b$ in $I(s_i)$ do
10:     $B \leftarrow b - u$
11:     {check remaining segments of $S$}
12:     for $j \leftarrow i+1, \ldots, c$ do
13:       $v \leftarrow (j-1) \cdot q + 1$
14:       {shift by up to $e - (i-1)$ positions}
15:       for $t \leftarrow -(e-(i-1)), \ldots, +(e-(i-1))$ do
16:         {if distance fits, add to alignment queue}
17:         if $B + v + t \in I(s_j)$ then
18:           add $B$ to $C$
19:         end if
20:       end for
21:     end for
22:     if not $B \in C$ then
23:       {check segments of $K$}
24:       for $j \leftarrow i, \ldots, c$ do
25:         $v \leftarrow (j-1) \cdot q + 1 + R$
26:         for $t \leftarrow -e-(j-1), \ldots, e-(j-1)$ do
27:           if $B + v + t \in I(k_j)$ then
28:             add $B$ to $C$
29:           end if
30:         end for
31:       end for
32:     end if
33:   end for
34: end for
35: {Send candidates to alignment}
36: for each $B \in C$ do
37:   align $f$ to $g[B-e, B+m+e]$ and report hit if successful
38: end for
```

Alignment phase As soon as a second qgram hit has been found for a given start index B, the alignment of f to $g[B-e, B+m+e]$ is enqueued for alignment and the rest of the filter phase for this value of B can be skipped. To speed up this alignment procedure in practice, the verification by alignment is computed on graphics hardware. Due to the parallel computation of alignments on graphics hardware, a huge number of possible alignment positions is collected before being submitted to the graphics card. The actual number depends on the amount of memory available on the graphics card (VRAM). The alignment of the read sequence f to the possibly matching part $g[B-e, B+m+e]$ of the reference sequence identified in the filter step is computed by a Needleman-Wunsch algorithm. On both sides of this template sequence, e additional bases are extracted from the reference sequence to allow for indels in an end-gap free alignment, if needed. We use unit edit costs for the alignment matrix. As soon as the error threshold is exceeded in a complete column of the distance matrix, the calculation is stopped, the read is discarded, and backtracing is omitted. If the complete distance matrix is computed, an optimal alignment path is calculated in the backtracing step and the read is reported as hit together with its optimal alignment. SARUMAN does not use banded alignments, yet, this is planned for future releases.

5.3.2. Implementation

SARUMAN was implemented according to the workflow presented in Figure 5.6. To achieve a competitive runtime for SARUMAN the use of GPU programming via the CUDA API was an essential part of the design. The CUDA API provides an extension to the C programming language, thus, SARUMAN was implemented in the C programming language to simplify the integration of CUDA code. The first thing to be implemented was the file import, supporting the file formats FASTA and FASTQ (with standard Sanger quality encoding). SARUMAN does not use the quality values of FASTQ files, but FASTQ is supported as all modern sequencers provide FASTQ as standard output format. While the reads are stored in memory, all sequences containing more than e ambiguous bases (usually represented as `N`) are discarded, due to the fact that an `N` would nevertheless be treated as a mismatch by SARUMAN. The primary data structure in SARUMAN is the qgram index used in the perfect match filter as well as in the filter algorithm described in the previous section. To obtain fast access times a hash table is used to store all occurrences of a certain qgram in the reference sequence. As the C programming language does not natively support hash structures the `uthash` package is used which provides hash tables and linked lists for C structures. `uthash` provides different hash functions, in SARUMAN the “Fowler/Noll/Vo” hash function is used. The perfect match filter performs a look-up in index I for the first qgram s_1 of each read f of length m. For each match position of s_1 in the reference sequence g it is checked if the sequence of length m starting at the match position of s_1 is identical to the sequence of read f. If an identical hit is identified by this procedure, it is exported, and the match position is stored to be skipped in the approximate match filter algorithm which was described in detail in the previous section. The filter algorithm computes potential alignment positions for all reads in the reference genome, which are verified in an optimal, CUDA-accelerated pairwise alignment as described in Section 5.3.1.1. Hence, SARUMAN is divided into two consecutive phases, namely mapping and aligning. Phase one, the creation of the qgram index together with the following mapping of reads through qgrams is completely processed on the host computer. In phase two, CUDA is used to compute the edit distance for candidate hits on the graphics card using a modified Needleman-Wunsch algorithm. Implementation details of this two phases are described in the next two sections.

5.3.2.1. Mapping phase

As mentioned the approximate filter algorithm is based on a qgram index of the reference sequence. Such a qgram index can be very memory consuming, especially if a complete index of all possible qgrams of a given length is created. Thus, SARUMAN hashes only qgrams that are observed within the reference sequence. The memory usage of the qgram index depends on the qgram length and the number of replicates per qgram number, but is mainly proportional to the size of the reference genome. Our tests show that a standard computer with 4 GB of RAM and a recent dual core CPU is able to read and process most bacterial genomes. A benchmark with 75bp reads and a qgram size of 9 showed the memory footprint to be below 4GB for typical bacterial genome sizes of 2-5Mb (see Table 5.2). The largest bacterial genomes known so far, *Sorangium cellulosum* (13,033,779 bp), needed 5.3 GB of RAM, while a small plant chromosome (the first chromosome of *Arabidopsis thaliana*, 32Mb) needed 9GB of RAM. Thus, a desktop PC with 8GB of RAM is sufficient to map all microbial genomes in a single run.

Table 5.2.: Influence of the size of the reference sequence on SARUMAN. The reference organisms were *Neisseria meningitidis* (2.2 Mb), *Escherichia coli* (5Mb), *Sorangium cellulosum* (13 Mb), and the first chromosome of Arabidopsis thaliana (32 Mb). Calculated with a qgram-size of 9 and a read length of 75bp.

Genomesize	2.2 Mb	5 Mb	13 Mb	32 Mb
Memory footprint	2.1 GB	3 GB	5.3 GB	9 GB

To process larger reference genomes with limited resources, the reference sequence can be divided into several chunks that are processed iteratively. Using this technique, it is possible for SARUMAN to run on computers with small amounts of RAM by dividing the qgram index into chunks fitting into available memory, e.g., the *A. thaliana* chromosome would be split into two 16MB chunks. A drawback is that the compute time is increasing with the number of chunks as for each chunk all reads have to be aligned, again. Thus, also with this workaround SARUMAN is not suitable for large eukaryotic genomes as the number of needed iterations would be too high.

Preceding the actual filtering step all reads are preprocessed. During this phase all located start positions of the $2 * c$ qgrams of a read in the reference genome are extracted from the qgram hash index. This list of positions is stored in an auxiliary data structure in order to minimize access to the hash index in later stages and therefore speed up the following filter step. Reads with perfect hits on the reference genome are still further processed as there may exist imperfect hits elsewhere on the reference, but for such reads the starting positions of qgrams representing the perfect hit are removed from the auxiliary data structure.
After this initial step, each read is mapped onto the reference genome using the algorithm described in Section 5.3.1.1. The start positions for the pairwise alignment are dependent on which two qgrams were successfully mapped during the filter mapping phase: A combination of first and last qgram (e.g. s_1 and k_c) exactly determines the start and end position of the read and the respective part of the reference sequence is extracted for alignment. If two "inner" qgrams (e.g. s_2 and k_{c-1}) are mapped, the start and end positions of the read on the reference sequence are unknown, thus in order to find the correct start and stop position e additional bases have to be taken from the reference sequence at both sides to allow for insertions and deletions in the qgrams at the edge of the read., i.e. an end-gap free alignment. Start positions for possible mappings are stored and transferred to the CUDA module in a later stage. In order to not only exploit the parallel architecture of graphics cards but also the availability of multi core CPUs, the matching phase uses two threads. The first thread handles mapping on the sense strand whereas the second thread processes mapping on the antisense strand. In contrast to many other approaches, which employ a 2bit encoding for the four DNA letters, SARUMAN is able to handle `N`s in reads as well as in the reference genome. Since many genomes contain a small number of bases with unknown identity, especially the rising number of unfinished draft genomes (see Section 2.3.2), it is of advantage to correctly treat these bases as `N`. Workarounds used by other approaches like replacing these positions with random or fixed bases to maintain the 2bit encoding may lead to wrong and in case of random replacements even to irreproducible results.

5.3.2.2. Alignment phase

The feasibility of sequence alignments using GPU hardware was demonstrated by different tools like SW-CUDA (Manavski and Valle, 2008) or CUDASW++ (Liu et al., 2009b). Compared to our solution, existing implementations focused on the search for similar sequences in a huge set of other sequences, which corresponds to a BLAST-like use. In contrast, SARUMAN searches for local alignments of millions of short sequences against one long reference sequence. Employing the filtering algorithm described in the previous section all possible alignment positions in the reference genome can be identified, and thereby the problem is reduced to global alignments of a read sequence with a short substring from a reference genome. Thus, compared to SW-CUDA or CUDASW++, SARUMAN does not align sequences

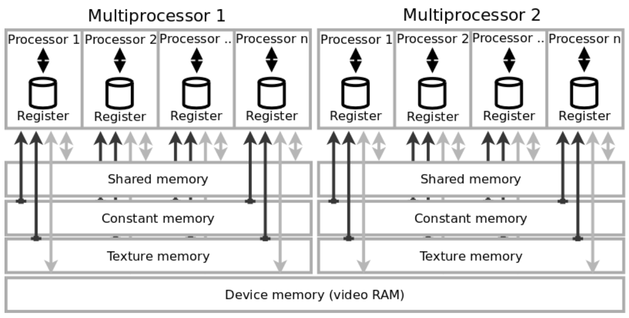


Figure 5.9.: CUDA hardware layout and memory organization. Each multiprocessor consists of 8 processors, each with an own set of registers. Shared, constant, and texture memory can be used by each of the 8 processors, while device memory is globally accessible between multiprocessors.

against a database of templates, but is designed as an alignment application to perform thousands of short pairwise global sequence alignments in parallel.

To efficiently use CUDA, it is of great advantage to understand the underlying hardware of CUDA capable graphics adapters. A GPU consists of a variable number of multiprocessors reaching from one in entry level graphics adapters up to 192 multiprocessors in high end video cards (Nvidia GeForce 600 Series, Kepler architecture released in March 2012). Each of these multiprocessors has access to registers of 8 or 16 kB size and is divided into 8 small processors. The available registers are divided and equally assigned to processors. This small amount of writable memory should be used for data processed in the currently active thread while texture memory and constant memory are read only and can be used to prefetch data from the much slower device memory. An overview of the CUDA hardware and memory model is given in Fig. 5.9.
Implementing code for the execution on a GPU is very similar to standard application development. Some additional keywords extending the C programming language are used to define on which device a function should be executed. A function on the GPU is mapped to one thread on one processor located on the graphics card, whereas one function can and should be executed on many different datasets in parallel. This execution scheme is called SIMT (Single Input Multiple Threads) due to its relation to the similar SIMD (Single Instruction Multiple Data) scheme used in classical parallel programming. Parallel programming with CUDA is transparent and (within NVIDIA products) device independent for developers and users. Launch of a CUDA application is controlled using only a few parameters defining the total number of threads. Those threads are organized hierarchically into *grids*, which themselves consist of threadblocks. Each threadblock is a collection of a given number of threads. A threadblock must be executable in any order and therefore must not have any dependencies on other blocks of the same grid. As the pairwise alignments computed by SARUMAN are completely independent of each other it is not difficult to adhere to this constraint.

The alignment process on the graphics adapter is organized in batches of a defined size. The filter algorithm collects all necessary data of the sequence pairs that should be aligned until a sufficient number of candidate hits has been found. Read and genome sequences are stored together with auxiliary data structures. Subsequently, all required data for the alignment phase is copied to the GPU as one large unit in order to minimize I/O overhead. The maximal number of alignments fitting into the GPU memory heavily depends on read length. The memory available on the graphics adapter is a second crucial limiting factor. SARUMAN automatically calculates an upper limit for the number of parallel alignments. As the number of alignments that can be computed in parallel drastically influences the overall performance, SARUMAN does not use the quality information of reads as this would double up the memory usage and thereby would divide in half the number of parallel alignments. Due to the huge output of modern sequencing systems, for datasets with sufficient coverage and quality it is feasible to simply remove all reads with low quality bases.
For 36bp reads a value of 200,000 alignments (100,000 for each mapping thread and direction) can be achieved on a standard GPU with 1 GB of VRAM. Once all data of the candidate hits has been copied to the GPU for each pair of genome and read sequence, the edit distance is computed using the user-defined values for match and mismatch positions (substitutions, insertions, and deletions have uniform mismatch costs). By comparing the distance with the supplied maximal error rate, all candidates with values above this threshold are discarded. Complete alignments are computed in a second backtracing step only for candidates with a edit distance below the defined threshold e. Typically the alignment phase for one batch takes only a few seconds to complete, including the whole process of copying raw data to and processed alignments from the GPU. Before any output is written, alignments are postprocessed by clipping gaps at the start and end that originate from the end-gap free alignment. For each possible start position of each read the optimal alignment is reported, in contrast to other available tools which only deliver a fixed number of n best positions or do not even guarantee to report any optimal alignment.
A great advantage of the CUDA alignment phase is that it can be executed asynchronously to the filter algorithm on the host computer, i.e., while the graphics card is executing a batch of pairwise alignments, the host CPU simultaneously executes the filter algorithm to collect alignment positions for the next batch. SARUMAN produces tab separated output by default which includes the start and stop position of the mapping together with the edit distance and the complete alignment. The package includes an easy to use conversion tool to generate the widely used SAM alignment format. The SARUMAN software is available for download at `http://www.cebitec.uni-bielefeld.de/brf/saruman/saruman.html`.

5.3.3. Results

SARUMAN was designed to compute exact, complete, and optimal mapping results while still providing a sufficient runtime to match the requirements of modern sequence analysis in times of next-generation sequencing. To prove the efficiency of SARUMAN as a short read matching tool, a detailed evaluation of the runtime as well as the accuracy in comparison to all tools presented in Section 5.2 was performed.

5.3.3.1. Evaluation methods

The evaluation was accomplished on a standard desktop PC with an Intel Core2Duo E8400 3 GHz dual core processor with 8 GB DDR2 RAM and a GeForce GTX280 graphics card with 1 GB of VRAM. All programs were run multi-threaded on both processor cores, the operating system was Ubuntu Linux. For two applications, CUSHAW and mrFAST, this system could not be used. In this cases a server system with 2 Intel Xeon E5620 2.40GHz quadcore HT processors, 72 GB DDR3 RAM and two Tesla 2070 General Purpose GPUs was used. To keep the results as comparable as possible, only two CPU cores and only one GPU were used in these cases. We used four datasets for the performance evaluation, three synthetic read sets of roughly 18 million reads generated from the *Escherichia coli K12 MG1655* genome (GenBank accession NC_000913) with reads of 36, 75, and 100 bp length, and a real dataset with data from one lane of a re-sequencing run of *Corynebacterium glutamicum ATCC 13032* (GenBank Accession BX927147) using the Illumina Solexa GAII sequencer, comprising 18,161,299 reads of 35bp length. The settings for all programs used in the comparisons were adjusted to make the run parameters as comparable as possible. To achieve this, we allowed 2 mismatches/indels for each read and set all programs to support multi-threading. Furthermore, we allowed gapped alignment of reads where possible and adjusted the alignment scoring matrix to simple unit costs.

5.3.3.2. Performance evaluation

In order to prove the exactness and completeness of the presented approach and to measure the discrepancy between exact and heuristic implementations we used the synthetic read sets described in Section 5.3.3.1. Synthetic reads were generated with 36, 75, and 100bp length from both strands with up to two errors of different types, i.e. mismatches, insertions, deletions, and combinations thereof. About three million of the artificial reads contained indels. All reads were mapped to the original source genome *Escherichia coli K12 MG1655*. The datasets are available on the SARUMAN homepage[2]. Table 5.3 compares the mapping ratios of the different tools together with their respective running time for 36bp reads and 100bp reads. Data for the intermediate reads of 75bp is provided in Appendix A.1.

[2]`http://www.cebitec.uni-bielefeld.de/brf/saruman/saruman.html`

Table 5.3.: Artificial data: Sensitivity evaluation with an artificial dataset. 17,984,730 reads of 100bp length and 17,959,086 reads of 36bp length were generated from *Escherichia coli* K12 MG1655 with up to two errors and known position. All reads were mapped to the reference they were created from. MCP (Mapped to Correct Position) denotes the number of mapped reads that had a match to their original position. BMCP (Best Match at Correct Position) denotes the number of reads where the best match was located at the correct position.

36bp reads

	SARUMAN	SOAP2	BOWTIE	BOWTIE2	BWA	CUSHAW	CUSHAW2	MRFAST	SHRIMP	PASS	Reference
Mapped	17,959,086	15,313,052	15,270,158	14,456,612	17,496,445	5,447,496	16,260,296	17,687,472	16,844,510	16,873,044	17,959,086
Not mapped	0	2,646,034	2,688,928	3,502,474	462,641	12,511,590	1,698,790	271,614	1,114,576	1,086,042	0
Perfect	4,999,970	4,999,966	4,999,970	4,999,970	4,999,970	4,999,970	4,999,970	4,999,970	4,999,970	4,999,944	4,999,970
With errors	12,959,116	10,313,086	10,270,188	9,456,642	12,496,475	447,526	11,260,326	12,687,502	11,844,540	11,873,100	12,959,116
1 mismatch	4,999,970	4,999,974	4,999,970	4,999,974	4,999,974	277,626	4,583,221	4,999,974	4,999,972	4,999,908	4,999,970
2 mismatches	4,999,950	4,999,954	4,999,954	2,732,130	4,999,958	30,638	4,185,585	4,999,974	4,999,958	4,999,936	4,999,950
1 Insertion	500,000	72,849	63,014	431,826	478,355	13,938	443,184	500,000	500,000	478,754	500,000
2 Insertions	500,000	5,768	2,212	112,220	314,537	784	387,761	500,000	61,836	496	500,000
1 Deletion	486,328	96,620	96,620	452,086	476,897	96,620	445,293	486,328	486,328	475,916	486,328
2 Deletion	486,242	19,828	19,828	259,568	378,271	19,828	408,232	214,618	198,284	452,714	486,242
1 Ins, and 1 Mism,	499,996	46,643	26,780	211,382	414,662	1,932	402,952	486,630	147,690	18,334	499,996
1 Del, And 1 Mism,	486,630	71,450	61,810	257,456	433,821	6,160	404,098	499,996	450,472	447,042	486,630
MCP	17,792,980	15,152,260	15,132,068	14,273,148	17,059,305	5,122,859	16,151,585	17,591,372	16,548,047	16,577,180	-
BMCP	17,792,950	15,152,260	15,132,052	14,272,990	17,056,553	5,122,859	16,151,585	17,591,332	16,547,287	16,577,173	-
Total Alignments	20,448,078	16,846,483	17,105,082	16,137,490	17,869,188	5,447,496	17,850,259	19,821,714	18,252,830	18,731,420	-
Runtime	06:24 min	04:39 min	15:43 min	06:09 min	08:57 min	02:01 min	07:02 min	04:23 min	32:16 min	12:09 min	-
RAM usage	4,137 Mb	635.9 Mb	14.7 Mb	42.09 Mb	92.4 Mb	730.7 Mb	1,200 Mb	7,044 Mb	1,313 Mb	389.6 Mb	-

100bp reads

	SARUMAN	SOAP2	BOWTIE	BOWTIE2	BWA	CUSHAW[a]	CUSHAW2	MRFAST[b]	SHRIMP	PASS	Reference
Mapped	17,984,730	15,105,693	15,091,462	17,830,600	17,805,133	5,148,292	17,368,107	13,569,796	17,984,730	16,925,234	17,984,730
Not mapped	0	2,879,037	2,893,268	154,124	179,597	12,836,432	616,617	4,414,928	0	1,059,496	0
Perfect	4,999,892	4,999,892	4,999,890	4,999,892	4,999,892	4,999,892	4,999,892	3,123,810	4,999,892	4,999,892	4,999,892
With errors	12,984,838	10,105,801	10,091,572	12,830,708	12,805,241	148,400	12,368,215	10,445,986	12,984,838	11,925,342	12,984,838
1 mismatch	4,999,898	4,999,898	4,999,898	4,999,898	4,999,898	100,360	4,849,320	3,868,132	4,999,898	4,999,898	4,999,898
2 mismatches	4,999,882	4,999,880	4,999,882	4,999,882	4,999,882	3,910	4,703,342	4,330,222	4,999,882	4,999,882	4,999,882
1 Insertion	499,982	26,229	22,802	496,040	492,382	5,238	479,173	387,688	499,982	490,088	499,982
2 Insertions	499,992	692	240	421,146	428,811	90	459,486	433,600	499,992	240	499,992
1 Deletion	494,928	34,964	34,964	494,024	491,454	34,964	480,229	374,884	494,928	493,580	494,928
2 Deletion	495,110	2,810	2,810	466,518	448,730	2,810	465,721	196,214	495,110	465,032	495,110
1 Ins, and 1 Mism,	499,978	16,014	21,944	472,772	468,627	264	465,292	433,508	499,978	13,772	499,978
1 Del, And 1 Mism,	495,068	25,314	9,032	480428	475,457	764	465,652	421,738	495,068	462,850	495,068
MCP	17,924,380	15,051,451	15,045,428	17,757,594	17,530,771	5,002,786	17,327,254	13,498,866	17,821,643	16,703,548	-
BMCP	17,924,376	15,052,277	15,045,426	17,757,554	17,528,745	5,002,786	17,327,254	13,498,864	17,820,604	16,703,541	-
Total Alignments	19,855,624	15,052,277	16,412,582	19,383,970	18,056,003	5,148,292	18,731,727	14,757,742	19,374,452	18,397,754	-
Runtime	22:15 min	08:28 min	22:00 min	21:03 min	19:18 min	03:55 min	32:20 min	09:00 min	149:43 min	37:40 min	-
RAM usage	3,832 Mb	702.9 Mb	14.3 Mb	46.00 Mb	125.4 Mb	772.6 Mb	3,600 Mb	14,738 Mb	1,313 Mb	421.1 Mb	-

[a] Due to memory requirements of up to 14Gb MRFAST was validated on the server system described in Section 5.3.3.1

[b] CUSHAW requires a GPU with Fermi architecture which is the successor of the GTX280 of the test PC, thus CUSHAW was validated on the server system described in Section 5.3.3.1

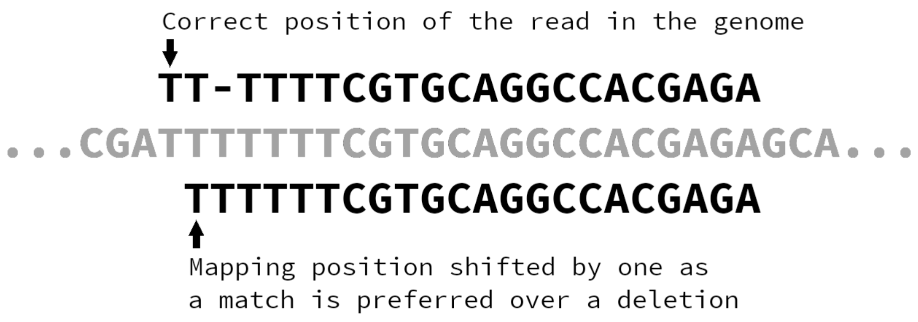

Correct position of the read in the genome
TT-TTTTCGTGCAGGCCACGAGA
...CGATTTTTTTCGTGCAGGCCACGAGAGCA...
TTTTTTCGTGCAGGCCACGAGA
Mapping position shifted by one as
a match is preferred over a deletion

Figure 5.10.: Example for an ambiguous mapping. The read has a deletion in a poly-T region compared to the reference. As a match of course is prefered over a deletion during the backtracing this results in a shift of the mapping position by one base. Without knowledge about the construction of the read it would not be possible to determine in which position the deletion event happened.

The goal was to map all artificial reads on the genome at the exact position without missing any mappings. As expected, SARUMAN was able to map all artificial reads to the genome for 36bp reads as well as for 100bp reads, thus the exactness of the approach is demonstrated. Furthermore, nearly all reads were mapped to the correct position in the reference genome. In some rare cases (0.34% (100bp) to 0.93% (36bp)) SARUMAN returned optimal alignments that are shifted from the reads' original positions by up to e bases (see Figure 5.10). Such cases can not be resolved, this is a general problem of the edit cost function and not a flaw of SARUMAN. Additionally, SARUMAN reported a large number of other matches on different sites of the genome. Among them were 30 (36bp) and accordingly 4 (100bp) matches that placed a read to an alternative position in the reference genome with a better score than the alignment to the original position. In this cases the incorporation of errors led to a read that just by chance fits better to a wrong genomic position. While alternative hits can be identified as misplaced in synthetic data, this behavior is preferable for real data as one can not determine the correct mapping position among several equally good locations. Both effects are more likely the shorter the reads are.

The comparison of other tools shows huge differences in the mapping performance of the evaluated programs, depending on different parameters. An important factor is the ability to handle gaps properly. Neither SOAP2 nor Bowtie nor CUSHAW are able to perform gapped alignments, all three tools were not able to map more than a small portion of sequences generated with indels to their correct position by using mismatches instead of gaps. The CUDA accelerated CUSHAW seems to have a problem even with mismatches, in the artificial datasets the mapping efficiency of CUSHAW is deficient for all but the perfect matching reads. With BWA, BOWTIE2, CUSHAW2, MRFAST, PASS, SHRiMP, and SARUMAN, all other tools are capable of aligning reads containing gaps, although PASS shows a poor mapping ratio for reads with more than one indel. CUSHAW2 performs significantly better than CUSHAW, although it still misses ∼600,000 (100bp) to 1,600,000 (36bp) reads. BWA shows a very good performance, but still misses ∼180,000 (100bp) to more than 450,000 reads (36bp). The results of SHRiMP, MRFAST, and Bowtie2 are dependent of the read length. SHRIMP shows a complete mapping of all reads for 100bp reads, although it places less reads than SARUMAN to the correct position. For 36bp reads SHRiMP loses its completeness and shows a poor mapping efficiency for reads with insertions, just like PASS. MRFAST shows a very good exactness for 36bp reads with only ∼270,000 reads missed, for the 100bp reads it misses more than 4 million alignments. Bowtie2 drastically improves the mapping performance of its predecessor for 100bp reads, with only ∼150,000 missed alignments it has the best accuracy after the exact approaches SARUMAN and SHRiMP. For 36bp reads it misses more than 3.5 million reads, mostly reads with 2 errors.

SARUMAN also shows the best performance on the real *C. glutamicum* dataset presented in Table 5.4. As this dataset originates from a re-sequencing of the identical strain, only a very low number of errors is expected. Therefore the differences between the tools are quite small. Nevertheless SARUMAN shows the highest mapping ratio of all tools and furthermore produces the highest number of valid alignments, independent from the error rate.
In the comparison of the other tools, a surprising finding is that Bowtie performs comparable or even slightly better than its successor Bowtie2 on the real-life data. Bowtie, Bowtie2, BWA, CUSHAW2 and PASS show comparable good mapping results, mapping only slightly less reads than the exact approach SARUMAN. BWA shows the lowest number of missed alignments after SARUMAN, but has a quite low number of total alignments. Bowtie and Bowtie2 map slightly less reads than BWA, but have a higher number of total alignments as they allow more hits per read. CUSHAW2 has less hits than the first three tools, but still more alignments than BWA. CUSHAW and MRFAST produced no alignments for more than 800,000 reads on all three error threshold, thus showing a poor mapping performance. The finding from the artificial dataset that SHRiMP performs poor an short reads is confirmed on the 35bp reads of the real dataset where SHRiMP missed more than 1.6 million reads for one and two allowed errors and can not compete with SARUMAN also at three allowed errors.

Besides sensitivity, another major requirement of short read alignment approaches is the performance in terms of running time. The runtime evaluation of the artificial reads shows that for 36bp reads most approaches map the dataset in less than 10 minutes. CUSHAW is the fastest with only ∼2 minutes runtime, but has a very poor mapping ratio. SOAP2 has a runtime of 04:39 min, but misses nearly 3 million possible mappings. MRFAST performs very well on the short reads with a runtime of 04:23 and only ∼270,000 missed reads, the second best value after SARUMAN. SARUMAN, BWA, Bowtie2 and CUSHAW2 are in the same range with runtimes of six to nine minutes (SARUMAN: 06:24 min). Bowtie

Table 5.4.: Real dataset: In-depth analysis of reported mappings of 18,161,299 *C. glutamicum* re-sequencing reads with different error thresholds of 1, 2, and 3. Perfect matches reported by SOAP2 include reads containing "N"s, which are treated as mismatch by other programs.

1 error allowed

	SARUMAN	SOAP2[a]	BOWTIE	BOWTIE2[b]	BWA	CUSHAW	CUSHAW2	MRFAST	SHRIMP	PASS
Mapped	17,868,892	18,001,767	17,847,705	17,864,272	17,864,263	17,172,193	17,712,833	17,193,212	16,468,563	17,864,502
Not mapped	292,407	159,532	313,594	297,027	297,036	989,106	448,466	968,087	1,692,736	296,797
1 alignment	17,291,581	17,467,485	17,274,166	17,288,091	17,578,942	17,172,193	17,187,550	16,636,138	15,975,056	17,287,079
2 alignments	168,942	153,627	168,194	168,491	168,289	0	150,803	162,973	141,743	169,751
3 alignments	57,267	44,237	56,687	56,994	56,713	0	44,033	55,543	41,414	58,242
$\geq$ 4 alignments	351,102	336,418	348,658	350,696	60,319	0	330,447	338,558	310,350	349,430
Alignments total	19,735,492	19,755,348	19,694,819	19,725,521	18,326,935	17,172,193	19,436,340	18,993,301	17,883,842	19,720,101
Runtime	04:52 min	06:09 min	04:25 min	08:10 min	05:35 min	03:21 min	05:52 min	09:49 min	31:41 min	19:28 min

2 errors allowed

	SARUMAN	SOAP2	BOWTIE	BOWTIE2	BWA	CUSHAW	CUSHAW2	MRFAST	SHRIMP	PASS
Mapped	18,025,584	18,001,767	17,999,961	17,984,048	18,023,109	17,292,124	17,820,286	17,301,357	16,558,248	18,008,908
Not mapped	135,715	159,532	161,338	177,251	138,190	869,175	341,013	859,942	1,603,051	152,391
1 alignment	17,406,040	17,467,485	17,388,188	17,382,398	17,732,158	17,293,270	17,292,124	16,712,622	16,057,777	17,392,887
2 alignments	184,224	153,627	181,476	177,024	171,385	0	151,810	174,661	144,175	183,259
3 alignments	72,679	44,237	73,534	65,619	58,298	0	44,328	66,704	42,699	75,193
$\geq$ 4 alignments	362,641	336,418	356,763	359,007	61,268	0	332,024	347,370	313,597	357,569
Alignments total	20,006,760	19,755,348	19,936,446	19,917,433	18,494,894	17,293,270	19,552,540	19,185,165	17,991,074	19,954,089
Runtime	06:08 min	09:29 min	19:40 min	08:10 min	08:30 min	03:19 min	06:39	10:20 min	32:03 min	19:42 min

3 errors allowed

	SARUMAN	SOAP2	BOWTIE	BOWTIE2	BWA	CUSHAW	CUSHAW2	MRFAST	SHRIMP	PASS
Mapped	18,082,983	18,001,767	18,057,089	18,026,244	18,066,847	17,345,771	17,849,688	17,344,318	17,853,159	18,015,869
Not mapped	78,316	159,532	104,210	135,055	94,452	815,528	311,611	816,981	308,140	145,430
1 alignment	17,421,035	17,467,485	17,410,328	17,408,659	17,774,056	17,345,771	17,187,550	16,731,135	17,279,255	17,401,227
2 alignments	203,034	153,627	195,440	183,958	172,931	0	150,803	185,796	168,216	181,740
3 alignments	88,419	44,237	88,077	71,095	58,490	0	44,033	75,347	56,639	72,874
$\geq$ 4 alignments	370,495	336,418	363,244	362,532	61,370	0	330,447	352,040	349,049	360,028
Alignments total	20,155,731	19,755,348	20,067,556	19,995,242	18,540,868	17,345,771	19,436,340	19,278,419	19,470,872	19,971,466
Runtime	06:12 min	09:26 min	34:41 min	08:10 min	08:40 min	03:59 min	05:53 min	10:22 min	32:49 min	19:46 min

[a] Please note that the evaluation results for SOAP 2 are identical independently from the parameters used. The error-parameters used seem to have no effect on the output of SOAP 2.

[b] BOWTIE2 does not allow to define an error threshold but provides predefined sensitivity settings. Only one run with maximum sensitivity was performed and results were filtered for one, two or three errors. Thus, the runtime is identical for all three error ratios.

and PASS need more than 10 minutes while SHRiMP needs more than 30 minutes for the mapping of the 36bp reads. In summary, for the 36bp reads only CUSHAW, SOAP2 and MRFAST have a better runtime than SARUMAN, but CUSHAW has a very poor accuracy and SOAP2 does not allow gapped alignments. MRFAST can not provide an exact result, but is ∼33% faster than SARUMAN and thus clearly the second best tool in this evaluation.
The image changes, though, for 100bp reads. SOAP2, CUSHAW, and MRFAST are still the fastest approaches with 08:28 min, 03:55 min, and 09:00 min, respectively, but the accuracy of SOAP2 and CUSHAW is still poor, and for the longer reads the accuracy of MRFAST drops dramatically, too. MRFAST misses more than 4 million alignments, showing the worst accuracy after CUSHAW for 100bp reads. SARUMAN, Bowtie, Bowtie2 and BWA have nearly identical runtimes, ranging from ∼19-22 minutes. PASS needs ∼37 minutes, while SHRiMP is far behind with almost 150 minutes. Thus, for longer reads SARUMAN shows the best accuracy as well as a runtime comparable to or better than all except the least accurate tools. The best result of the other tools is achieved by Bowtie2 and BWA with good mapping accuracy and runtimes comparable to SARUMAN.
For the *C. glutamicum* re-sequencing dataset, SARUMAN is among the fastest tools for all three error thresholds with running times of 04:52, 06:08, and 06:12 minutes. CUSHAW (03:21, 03:19, and 03:59 minutes) and CUSHAW2 (05:52, 06:39, and 05:53 minutes) are slightly faster, but miss significantly more reads. The runtime for Bowtie is highly dependent on the number of allowed errors, rising from 04:25 minutes to 34:41 minutes from one to three errors. Runtimes for SOAP2, Bowtie2, BWA, and MRFAST are all higher, but in the same range as SARUMAN. PASS and SHRiMP are the slowest approaches on real data with approximately 20 (PASS) or more than 30 (SHRiMP) minutes runtime independent of the error threshold.

Considering all calculated datasets, CUSHAW and SOAP2 show the best runtime performance of all compared approaches, being faster than SARUMAN in most cases, but at the same time they are the approaches with the lowest sensitivity. BWA and Bowtie2 show runtimes comparable to SARUMAN in most cases as well as a good accuracy, while Bowtie is highly dependent on the used error ratio. For low error rates it can compete with other approaches, for higher error rates the runtime rises significantly. CUSHAW2 is far better than CUSHAW, it shows a mediocre accuracy on the artificial dataset, but a good runtime at acceptable accuracy on the real data. MRFAST is fast and accurate for real short reads, but becomes very insensitive for longer reads. SHRiMP and PASS are the slowest of the compared approaches for all evaluated datasets. SHRiMP shows a poor mapping accuracy for 36bp reads, but provides an exact result like SARUMAN for longer reads (100bp as well as 75bp, see Appendix A.1), although the result is not complete in a way that not all possible alignment positions are found.

5.3.3.3. Performance of filter and alignment components

As described in Section 5.3.2, SARUMAN has two main components, the filter algorithm and the CUDA accelerated alignment module. To find out how each component contributes to the runtime of SARUMAN under different circumstances the components will be evaluated independently in this section.
The performance of the filter algorithm mainly depends on the read length and the qgram length. To test for the influence of the read length we used the artificial datasets described in Section 5.3.3.1. We compared the number of alignment candidates identified by the filter algorithm and transferred to the graphics card with the number of alignments that were actually successfully verified in the alignment step. The results are shown in Table 5.5. For 36bp reads ∼35% of the alignment candidates are discarded, while for 100bp reads only ∼3.5% of the alignment candidates do not result in a valid alignment. Thus, it shows that the efficiency of the filter algorithm is slightly increasing with longer read lengths due to the bigger qgram size. The performance of the alignment step is highly dependent on the read length as longer reads are more memory consuming on the graphics adapter, thus the read length defines how many alignments can be computed in parallel with the VRAM available on the used graphics card. Consequently, the performance of the alignment step is decreasing for longer reads. As a consequence the ratio of filter time to alignment time shifts with increasing read length. For 36bp reads the filter algorithm takes nearly eight times as long as the alignment step, while for 100bp reads the alignment step takes more than three times as long as the filter algorithm.
Another question is how much runtime performance is achieved by the usage of CUDA programming. To evaluate this we executed the SARUMAN alignment algorithm module on the GPU as well as on the CPU. To avoid bias by different implementations the same code was used, compiled once to be executed on the GPU, once to be executed on the CPU. For short reads of 36bp the GPU version of the alignment module is ∼26 times faster than the CPU version. As expected, for longer reads the advantage of the GPU implementation decreases, but even for 100bp reads the speed-up due to the usage of graphics cards is significant, the GPU implementation still shows a more than five-fold speedup compared to the CPU implementation of the same algorithm.

5.3.3.4. Summary

The evaluations in the previous sections showed that SARUMAN can provide the required optimal result independent of the tested parameters and thus has the highest accuracy of all evaluated methods. Furthermore, in comparison to the other approaches the runtime efficiency of SARUMAN is competitive or even superior. The interaction of an efficient filter algorithm with a CUDA implementation of the Needleman-Wunsch alignment works perfectly in a sense that for shorter reads the

Table 5.5.: Sensitivity of the filter algorithm and performance gain of the GPU implementation, tested on *E. coli* artificial data. The upper part shows the runtime of the filter step, the alignment candidates passing the filter and the number of reads that where successfully aligned. The middle part compares the runtime of the filter step with the runtime of the alignment step. The lower part compares the runtime of the alignment step on a GPU vs. the runtime on a CPU.

Read length	36 bp	75 bp	100 bp
Candidates	23,040,503	15,405,284	14,991,106
Aligned	14,918,066	14,553,824	14,451,680
Filter step	326 s	293 s	289 s
Filter step : Alignment	7.76 : 1	1 : 1.47	1 : 3.62
Alignment on GPU	42 s	430 s	1046 s
Alignment on CPU	1085 s	3282 s	5814 s
GPU:CPU	1 : 25.83	1 : 7.63	1 : 5.55

speedup through the usage of graphics adapters for the alignment step is tremendous, while for longer reads, where the performance gain of the CUDA implementation decreases, the efficiency of the filter algorithm increases. Nevertheless, for read lengths exceeding 100bp the performance of SARUMAN will significantly drop as the increase in alignment time can not be compensated by the speedup of the filter step. However, it is arguable if a *short* read alignment program should be used for such reads.
In summary, SARUMAN is the only approach that provides exact and complete results, while still being nearly as fast or even faster than all compared approaches. Thereby it shows the best overall performance for short read alignment against prokaryotic genomes. The good performance of SARUMAN comes with some limitations. SARUMAN is a short read alignment solution dedicated for short microbial genomes, for large mammalian or plant genomes the memory requirements of the genome index are to high and they can not be processed in admissible time. In the following section some use cases of the SARUMAN software will be presented.

5.3.4. Application cases

5.3.4.1. Mycobacterium tuberculosis microevolution

The classical application case of SARUMAN is the analysis of single nucleotide polymorphisms in re-sequencing assays. A nice example of such a study was given by Roetzer et al. (2013) with an analysis of 86 *Mycobacterium tuberculosis* outbreak isolates. As mentioned in Section 4.2.4.3 already, *M. tuberculosis* is the causative agent of the infectious disease tuberculosis (TB). According to the global tuberculosis report 2012 (World Health Organization, 2012), 8.7 million new TB cases were detected in 2011, and 1.4 million people died from the disease.

Furthermore, a third of the global population carries the bacterium as a latent infection (Luciani et al., 2009). In addition to the high infection rate and death toll of TB, recent years have seen the emergence of multidrug-resistant (MDR) or even extensively drug-resistant (XDR) strains which are resistant to almost all available antibiotic treatments. Thus, an understanding of *M. tuberculosis* transmission is crucial to optimize control strategies for this pathogen. In classical epidemiology strain typing methods like IS*6110* DNA RFLP (Restriction Fragment Length Polymorphism) fingerprinting (Van Embden et al., 1993) and 24-locus MIRU-VNTR (Mycobacterial Interspersed Repetitive Unit-Variable Number Tandem Repeat) genotyping (Allix et al., 2004) are used, but they lack the discriminatory resolution to survey local outbreaks of highly similar strains. Roetzer et al. (2013) used NGS for identification of transmission chains in an outbreak in Hamburg with samples collected over 14 years.
In a epidemiological surveillance from 1997 to 2010 comprising a total of 2301 patients *M. tuberculosis* strains were isolated and subjected to classical strain typing. The largest strain cluster identified by classical strain typing consisted of 86 strains that showed nearly no difference in IS*6110* DNA RFLP or 24-locus MIRU-VNTR genotyping.
To study the potential of NGS for accurately tracing particular outbreak clones and to reveal the patterns of spread of the cluster strain over time, these 86 strains were sequenced. To have a reference that reflects the initial genome, an early strain *M. tuberculosis* 7199/99 from 1999 was completely sequenced using the 454 pyrosequencing technique (see Section 2.2.1). The further 85 genomes were sequenced on an Illumina GAIIx sequencing system with 50bp read length and 3-10 million reads per isolate. The resulting reads were mapped to the reference strain *M. tuberculosis* H37rv using SARUMAN with an error threshold of 3. SNPs for all 86 isolates were extracted based on the reads mapped by SARUMAN if the reference genome was covered by at least ten reads and at the same time the differing allele haa a minimum frequency of at least 80%.
In total, 85 SNPs were detected in the outbreak isolates, which were all verified by Sanger sequencing. Various further SNPs were detected in repetitive regions, but in a grab sample of 15 SNPs, none of these could be verified by Sanger sequencing, thus SNPs in repetitive regions were ignored. Of the 85 reliable SNPs, seven were detected outside of coding sequences. Of the remaining 78 SNPs within coding sequences 31 were synonymous SNPs and 47 were non-synonymous SNPs. The sequenced strains were clustered by their SNP profile, resulting in seven clusters with two to 24 isolates and 36 unique SNP profiles. To get a detailed image of the population and spreading of the Hamburg clone, a minimum spanning tree was calculated from an alignment of 85 verified SNPs (see Figure 5.11). In this tree, 72 of 86 isolates were located in a single clade, which was termed the “Hamburg clone”. The NGS analysis of the 86 isolates disclosed the presence of distinct genotypes, a result that is in contrast with the classical genotyping and proves the better resolution of NGS sequencing in population studies.

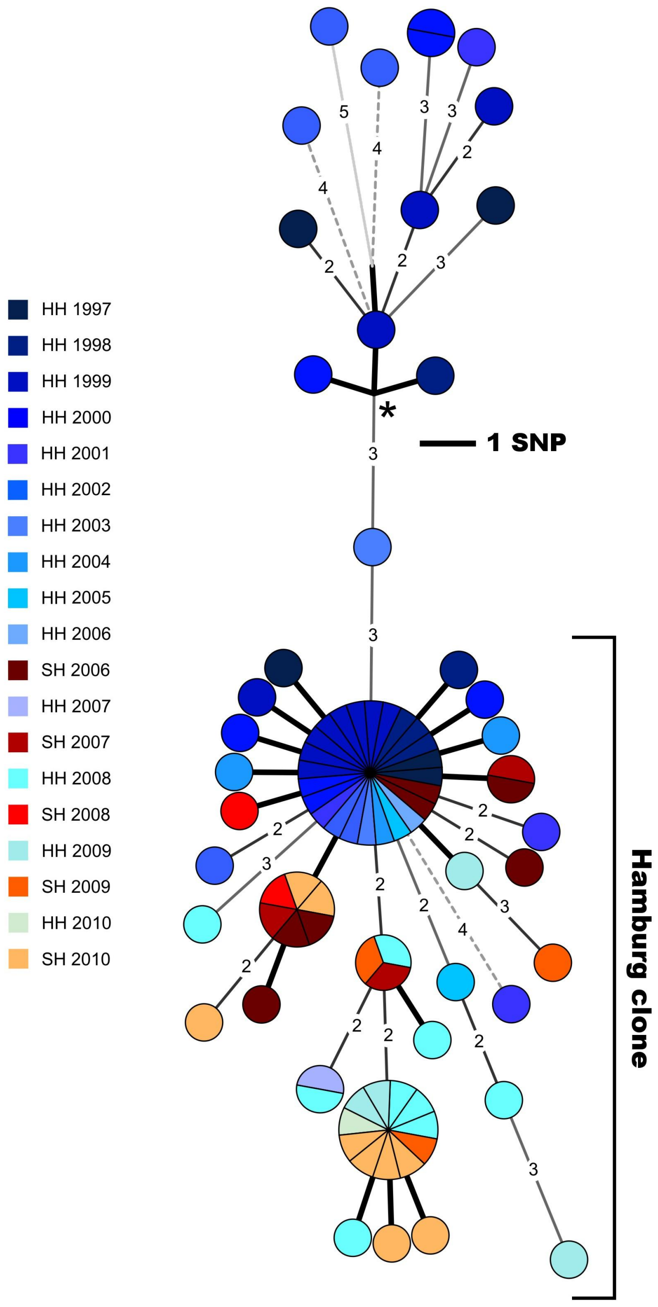


Figure 5.11.: Minimum spanning tree of 86 *M. tuberculosis* strains isolated in Hamburg and Schleswig Holstein from 1997 to 2010. The color of nodes reflects to year and place of the isolate, the legend can be found on the left. Blueish colored strains were isolated in Hamburg, reddish/yellowish strains reflect infections spread to Schleswig Holstein. The large clade reflecting the “Hamburg clone” is clearly visible, furthermore there is one large cluster of 24 isolates which have identical SNP patterns. Numbers at the edges show the number of SNPs between the respective isolates/clusters, bold edges reflect a single SNP, only. Image taken from (Roetzer et al., 2013).

The SNP data was furthermore correlated with human-to-human transmission chains that were confirmed by surveillance and contact investigation work performed by the Public Health Offices. A maximum number of three SNPs was identified in the eight confirmed transmission chains which involved transmission links between 31 patients. This allowed to estimate the evolutionary rate of *Mycobacterium tuberculosis* as 0.4 mutations per genome and year. The SNP data furthermore correlates well with the two known spatio-temporal spreads to Schleswig-Holstein, as seen by two distinct clusters in Figure 5.11.
One major finding of this work was that the level of genomic variation in *M. tuberculosis* transmission chains was very low with not more than three SNPs. Still, the resolution of the presented NGS approach was high enough to resolve the outbreak and identified the "Hamburg clone" that could not be detected with classical genotyping methods. Thus, re-sequencing is a powerful new tool in the surveillance and control of infectious diseases like tuberculosis.

5.3.4.2. Chinese Hamster miRNAs

While the *Mycobacterium* use case demonstrates the classical application of SARUMAN for SNP detection, the use of the software is not limited to genomic references, but also artificial reference sequences can be processed. An example is the analysis of microRNAs (miRNAs) of chinese hamster ovary (CHO) cells using SARUMAN on a database of known miRNAs (Hackl et al., 2011).
CHO cells are widely used in biological and medical research and are the most important cell line for the commercial production of recombinant therapeutic proteins, e.g., monoclonal antibodies. Functional genomics and proteomics have been employed to identify promising cellular pathways as well as specific genes that could serve as targets for genetic engineering of the CHO cells for higher productivity. When the CHO miRNA study started, no genome sequence of a CHO cell line was available (the first one was published by Xu et al. (2011)).

MicroRNAs are small, non-coding RNAs that are transcribed in the nucleus of the eukaryotic cells. They are processed by a special RNaseIII (RNAse III Drosha) and exported to the cytoplasm as hairpins of $\sim$70bp. These hairpins are cleaved by another RNAse (RNAse III Dicer) into short double stranded fragments of 20-25bp length which form the mature miRNAs (Carthew and Sontheimer, 2009). One strand of the miRNA is incorporated into the RNA-induced silencing complex (RISC), which binds to partially complementary regions in the 3' untranslated region (UTR) of target mRNAs and degrades or represses their translation. A miRNA can bind to many different mRNA-UTRs, and in each UTR there can be several binding sites for different miRNAs. Thus there are realms of combinations and interactions that allow a complex translational regulation.
The post-transcriptional regulation of gene expression in CHO cells by miRNAs is of highest interest as a potential tool for the characterization and genetic engineering of CHO cell lines as the regulation influences the complete life cycle of CHO cells. In the absence of a genomic sequence, the exploration of miRNAs had to be tackled

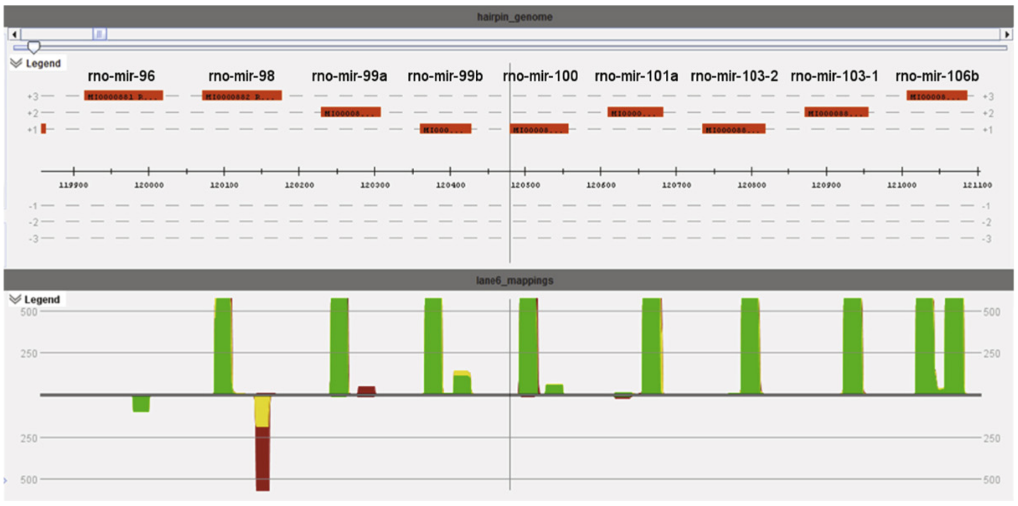


Figure 5.12.: Detail of the CHO miRNA mapping showing the coverage on 11 miRNA hairpin sequences from miRBase. It shows that for all but the first miRNA hairpin sequences there are one or two short regions that have a high coverage, either on the 5' or on the 3' section of the hairpin sequence. The coverage is colored according to the mapping quality of the respective reads. Green denotes perfect matches, yellow shows matches with errors that have no better match position, red is used for reads that map better to a different position in the reference.

in a *de novo* approach. For this purpose, total RNA from 6 different CHO cell lines was isolated using Trizol[3] reagent, and small RNA fragments of 18-36bp were purified from the total RNA. The purified RNA fragments were sequenced on a Illumina GaIIx (see Section 2.2.2.1) with ∼16 million 36bp reads per cell line. The reads were trimmed for adapter fragments, and reads with poly-A low complexity regions or overall low quality were discarded. Identical reads were merged to one single read and the number of occurrences of each unique read was stored. The reads processed in this way were mapped to an artificial reference sequence that was created by a concatenation of all known miRNA hairpin sequences available in the miRBase database (Griffiths-Jones et al., 2006). To prevent mappings that span more than one known miRNA, the miRBase sequences were concatenated with a spacer consisting of 50 "N"s[4] which are treated as mismatches by SARUMAN. This resulted in a 1.6Mb artificial genome named comprehensive miRNA hairpin reference (CMR) that served as a reference for mapping of the unique reads using SARUMAN with an error tolerance of 3. The result was a short read alignment showing numerous reads aligning to non-overlapping blocks of 22bp length at the 5' or 3' arm of the hairpin reference, representing the mature miRNA (see Figure 5.12).

[3]Invitrogen, Carlsbad, CA, USA

[4]For the subsequent mapping with 3 allowed errors a spacer of 4 "N"s would have been sufficient, but the 50"N"s also served as spacers for the visualization displayed in Figure 5.12.

In total, 235 hairpin sequences showed hits with at least 5-fold coverage, of which 130 showed hits on the 5' positions as well as on the 3' position while 105 showed coverage only on one end of the hairpin sequence. Thus, altogether 365 highly conserved mature miRNA sequences could be identified by this study.

5.3.4.3. Arabidopsis thaliana ChIP-seq data

A further application field for SARUMAN is the analysis of ChIP-seq data. In his PhD thesis, Oliver Jahns describes the establishment of ChIP-seq protocols for the analysis of protein-protein interactions in *Arabidopsis thaliana* (Jahns, 2012). The plant *A. thaliana* (mouse-ear cress) is a widely used model organisms in plant research as it has a short life cycle and a relatively small genome of 125Mb in five chromosomes. The genome of *A. thaliana* was the first plant genome that was completely sequenced. A field of special interest in *A. thaliana* research is the investigation of flavonoid biosynthesis. Flavonoids are plant secondary metabolites that fulfill various functions in plants, e.g., flower coloration (the name flavonoid originates from the Latin word “flavus” meaning yellow), UV filtration, symbiotic nitrogen fixation, chemical messaging, physiological regulation, or inhibition of plant pathogens.
The regulation of the gene expression in flavonoid biosynthesis is mainly controlled by transcription factors. To identify genome-wide the binding sites of transcription factors the ChIP-seq technique as described in Section 5.1 is the ideal approach. ChIP-seq works with selected antibodies that target specific transcription factors, thus in this study three transcription factors PAP1 (Production of Anthocyanin Pigments), TT8 (Transparent Testa8), and EGL3 (Enhancer of GLABRA3) were selected as they are known to be involved in the anthocyanin biosynthesis (anthocyanins are a class of pigments).
Chromatin immunoprecipitation was performed for three datasets, and the resulting DNA fragments were sequenced on an Illumina GaIIx system with 36bp read length. One lane of the GaIIx flow cell was sequenced per sample, and an additional lane for each sample was used for a negative control. The reads achieved by sequencing were trimmed (the first and the last base of each read were removed), quality filtered, and subsequently mapped onto the *A. thaliana* chromosomes with an error threshold of 2 using the server system described in Section 5.3.3.1. The analysis of the mapping results showed that PAP1 could not be enriched by ChIP, but TT8 and EGL3 were successfully enriched. As the mapping for TT8 and EGL3 showed a high noise level due to reads mapping to various positions, the alignment results were filtered for unique perfect matches, i.e., reads that were mapped without errors and only to one position (see Figure 5.13).

The mapping results were analyzed with the PeakRanger software (Feng et al., 2011), a peak caller software package that detects enriched regions and can resolve closely-spaced peaks. For TT8, 1441 enriched regions could be identified using this approach, of which 59% were located in coding regions and another 13% in promotor regions. For EGL3 5653 enriched regions were detected, of which 48%

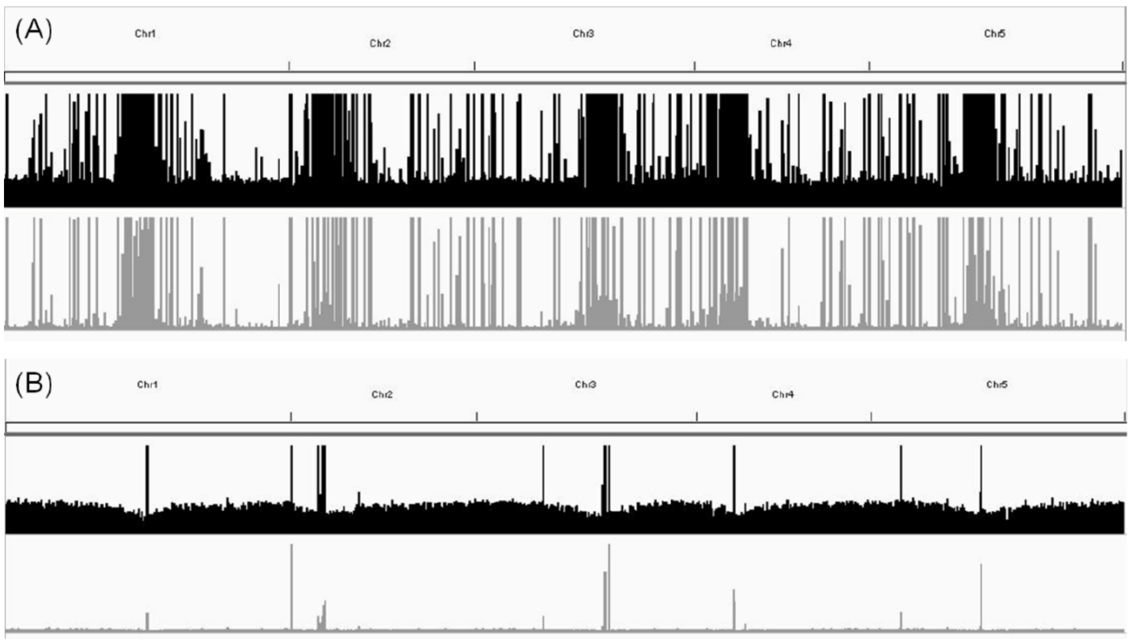


Figure 5.13.: Coverage plot of a mapping of ChIP-seq reads to the five *A. thaliana* chromosomes. The black plots are the ChIP-seq datasets, the grey plots are the negative controls. The upper two plots show the mapping of all quality filtered reads. The lower two plots show the coverage for perfect unique reads, only. One can see that the upper plots are much more noisy. Image courtesy of (Jahns, 2012).

were in exon regions while 20% were located in promotor regions. It could be shown that TT8 as well as EGL3 are involved in the regulation of anthocyanin-specific regulons. The binding of TT8 and EGL3 to known loci of anthocyanin regulator was confirmed, and hundreds of potential binding sites were identified as target regions for further research.

5.3.4.4. Ongoing work: RNA-seq analysis in promotor analysis

A promotor is the DNA sequence that initiates the transcription of a gene by specifying the binding site for the RNA polymerase (RNAP) and thus the transcriptional start point (TSP). Promotors are located upstream, on the same strand and in close proximity to their target gene. Exact knowledge of the promotor location is crucial for the characterization of promotor activity under different environmental conditions. At the CeBiTec RNA-seq is used to identify promotor sequences and transcriptional start points (TSPs) of *Corynebacterium glutamicum* (Patek et al., 2013).
In Section 5.1 the RNA-seq technique was described as an efficient technique in the analysis of transcriptional regulation in microorganisms. For the specific analysis of promotors a modification of RNA-seq termed “differential RNA-seq” (Sharma et al., 2010) is used. Like the classical approach it starts with the isolation of total RNA

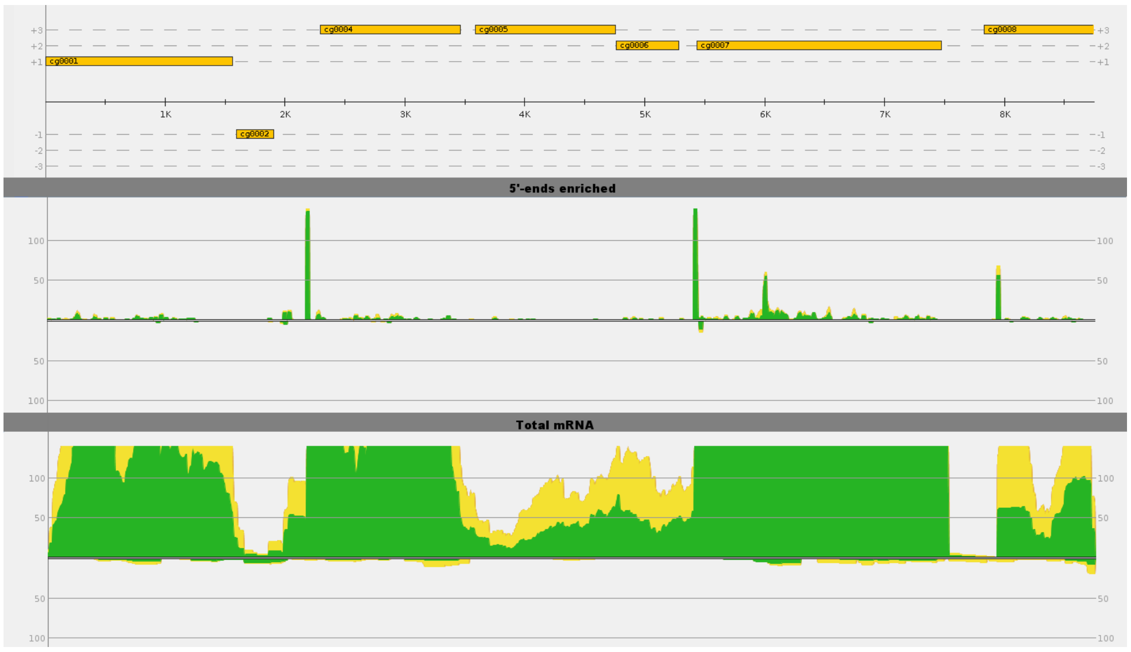


Figure 5.14.: Combination of classical RNA-seq data with differential RNA-seq. The upper part of the plot shows the annotated genes of the reference genome, the middle part shows the alignment coverage of the 5'-enriched RNA-seq data, the bottom part shows coverage with classical RNA-seq data. The meaning of the colors is as described in Figure 5.2. The transcriptional pattern is nearly identical to Figure 5.2, again showing the set of co-transcribed genes cg0004-cg0007, but with the addition of the 5'-enriched data it becomes obvious that cg0004 and cg0007 have independent promotors, while cg0005 and cg0006 are transcribed together with cg0004.

from target cells. The total RNA comprises more than 95% of stable RNA (rRNAs and tRNAs) that have to be removed from the samples. Then cDNA is created from the enriched mRNA by reverse transcription and amplified using polymerase chain reaction (PCR). The resulting ds-cDNA fragments can be sequenced subsequently with a NGS technique of choice, at the CeBiTec usually using an Illumina system. In differential RNA-seq an additional step is added prior to sequencing in which a terminator exonuclease degrades processed transcripts (which have an 5'P end), but does not affect native 5' ends (5'PPP) and hence enriches these native transcripts. No matter if classical RNA-seq or differential RNA-seq is used, after the sequencing the resulting reads have to assigned to their position in the genome by short read alignment. In Figure 5.14 a combination of classical RNA-seq with differential RNA-seq is displayed for the same part of the *C. glutamicum* genome that was already shown in Figure 5.2.

RNA-seq is the ideal tool for the massive parallel detection and characterization of promtors in microbial organisms. Moreover, it allows not only the precise localization of the promotors but also provides quantitative information on gene expression and promotor activity. SARUMAN is the ideal tool for short read mapping in RNA-seq experiments on bacteria as it combines maximal mapping accuracy and very good runtime for bacterial species as was demonstrated in Table 5.4 for the target organism of this use case, *C. glutamicum*. Thus SARUMAN is used in all ongoing bacterial RNA-seq studies at the CeBiTec as well as in other studies like bacterial re-sequencing or ChIP-seq analyses.

CHAPTER 6

Discussion

In the previous chapters two applications have been presented that address two different aspects of comparative genomics pointed out in Section 1.1. EDGAR - "**E**fficient **D**atabase framework for comparative **G**enome **A**nalyses using BLAST score **R**atios" - supports large scale gene content analysis of dozens of bacterial genomes. SARUMAN - "**S**emiglobal **A**lignment of short **R**eads **U**sing CUDA and Needle**MAN**-Wunsch" - provides a fast and exact short read alignment software and thereby enables a comparison of bacterial genomes on single nucleotide level. This final chapter will give a summary of the results of this work, and the presented tools will be discussed with regard to their advantages and disadvantages. Furthermore, an outlook on the future of the presented tools and the field of comparative genomics in general will be given.

6.1. Gene content analysis with EDGAR

EDGAR is a web-based analysis platform for gene content comparisons of microbial genomes. All analyses are based on a generic orthology criterion, i.e., the threshold that defines two genes as orthologous genes is computed from the analyzed genomes themselves based on the distribution of BLAST score ratio values (see Section 4.2.1.2). No user-provided parameters are needed and thus no bias is introduced into the analyses, which is one of the big advantages of EDGAR.
The data management is project based, i.e., for every set of genomes that is compared a dedicated EDGAR project is created. As the comparison of the gene content of several genomes is a compute intensive task, the needed calculations are performed on a compute cluster, and the results are stored in the project database.

The access to every EDGAR project can be controlled via a user management system. Based on the granted permissions EDGAR users obtain access to different visualization and analysis features via a web interface described in Sections 4.2.2.3 ff. The access controlled web interface provides a convenient platform for the collaborative analysis of genomic data by scientists from different institutions distributed around the world.

On the one hand, EDGAR provides access to public projects for 130 genera with 1449 genomes that can be accessed without any restrictions. These 130 projects pose a valuable and frequently used scientific resource that is provided as a free non-commercial service. On the other hand, there are 101 private projects with 1461 genomes that are used to analyze confidential, unpublished genomes. These projects are as well provided free of charge on collaborational basis, and individual scientists, scientific institutions, or consortia from all over the world used this service in the last year as is demonstrated in Figure 4.23. In addition, the use cases presented in Section 4.2.4 prove that EDGAR can contribute important results to biological analyses on different species and genera, which is also reflected by more than 50 peer reviewed publications citing EDGAR[1].

The set of features provided by EDGAR is unique in its comprehensivenes. While several tools provide different forms of gene content analysis, the combination of genomic subset calculations with phylogenetic and statistical analyses is not found in any other tool. Furthermore EDGAR is the only tool that allows higher level comparisons by genome groups and metacontigs. In addition, the web interface is designed to be as simple and user friendly as possible, for most analysis features no user input is needed except a selection of organisms of interest. For the public projects, the available genomes are sorted by genus. For private projects the included genomes are selected by the user, thus it is ensured that all needed genomes are available while no superfluous genomes slow down the analyses.

The huge number of projects provided in the EDGAR framework is only possible because the project creation and maintenance was simplified as far as possible. The creation of an individual project is realized with a single command line, and as EDGAR allows mainly read-only database access, the effort for the maintenance of an existing project is minimal. As the computation of an EDGAR project relies only on the amino acid sequences of the genes, the work with unfinished draft genomes is unproblematic as long as a gene prediction is available. EDGAR was successfully used with draft genomes, but one has to be aware of the fact that every gap in the genome sequence may “destroy” one gene and thus the gene content of draft genomes may be biased towards missing genes.

In summary, the goals defined in Section 3.1 were fully achieved, and EDGAR is established as an accepted, convenient, and widely used tool for comparative genomics on gene level.

[1]According the Google Scholar and Web of Knowledge, as of 15.01.2013

Outlook: While EDGAR proved to be a useful tool for comparative gene content analyses, there are several aspects that can or even have to be addressed in the future. One point that will become more and more important in the future is the bias introduced by draft genomes. As nowadays nearly 50% of the genome sequences uploaded to the NCBI databases are permanent drafts (see Section 2.3.2), a mechanism to get a maximum of information of the draft genomes is needed. Large gaps in the genome sequences may cause complete CDS to be missed, but often the gaps in draft genomes are relatively short, and thus the affected CDS are only truncated. Such truncated genes would have to be treated separately, e.g. by lowering the orthology cutoff for genes at the boundaries of sequencing contigs.
Another problem that mostly influences big projects is the update mechanism of EDGAR. Up to now project updates always recalculate the entire project database as new genomes may influence the SRV statistics and thus the orthology threshold. Most EDGAR projects contain 3-10 genomes, for these cases a complete update is not critical, but for the bigger projects like the public Escherichia project with 56 organisms and more than 250,000 genes a complete recalculation is too much effort for the addition of just one or two genome. As the addition of one or two genomes to such a big project should influence the SRV statistics only marginally, a mechanism to add genomes to big projects without changing the SRV threshold would be a considerable option.
The most severe problem that the EDGAR service has to face in the future is the rapid increase in available genome sequences. While the increasing computational effort during project setup can at least partially be compensated by extended compute resources or special purpose hardware like FPGA based TimeLogic BLAST accelerator cards, the Perl/CGI based web interface will comes to its limits in the medium term. Complex operations like the calculation of large pan genomes cause a lot of workload on the web server, especially if several projects are active at the same time, and the computation may take more time than an EDGAR session allows. Furthermore, for large datasets also the result visualization capacities of a browser page come to a limit, e.g., a pan genome of more than 20,000 gene sets can be hardly visualized in a HTML table. Thus, the EDGAR user interface will be replaced with an up-to-date stand-alone Java application. A prototype for such a user-interface, named “jEDGAR”, was developed by Dominik Vahrenhorst in his Master thesis under my supervision which was finished at 28.09.2012. jEDGAR is a client-server solution based on the Java Platform, Enterprise Edition (Java EE) and NetBeans Platform. The jEDGAR client sends requests via REST to the server, the server fetches the required data from the database, performs the requested calculations, and sends the result to the client in XML format via the Java Architecture for XML Binding (JAXB). The jEDGAR client does not yet provide all analysis features that the web interface offers, but in the foreseeable future it will replace the EDGAR web interface.

6.2. Short read alignment with SARUMAN

In Sections 3.2 and 5.1 the various applications of short read mapping in modern sequence analysis like RNA-seq, Chip-seq, and genome re-sequencing have been discussed. For all these applications an accurate mapping is crucial. Furthermore, with the ever increasing output of next-generation sequencing systems, the need for high performance solutions in short read alignment is obvious.
With SARUMAN this work presents a non-heuristic short read alignment software that by design guarantees that:

1. All reads that can be aligned to the reference are aligned.
2. All possible mapping positions under a given error constraint are found.
3. For each mapping position one optimal alignment is reported.

Thus, SARUMAN provides an in a mathematical sense optimal mapping result. At the same time, SARUMAN is one of the fastest short read mapping softwares available. In a comprehensive performance evaluation in Section 5.3.3.2 in comparison with nine competing applications (see Section 5.3.3.2) it was demonstrated that SARUMAN is comparable fast or even faster than heuristic approaches and several times faster than existing exact approaches. The good performance of SARUMAN was proven on artificial data as well as a real data, and for different parameter settings and read lengths.
The optimal results achieved by SARUMAN are the result of a specific filter algorithm that rapidly identifies possible alignment positions of short reads within a reference sequence using an index of this reference and a qgram-based algorithm that relies on the double application of the *pigeonhole principle* (see Section 5.3.1.1). The good runtime performance is achieved due to the use of graphics card programming via the CUDA API (see Sections 2.5 and 5.3.2.2) for the alignment step used to verify the alignment positions found by the filter step. The usage of graphics hardware for general purpose computing is an emerging trend in scientific programming. Several popular algorithms and programs like Smith-Waterman-Alignment, HMMer, or MUMmer (Liu et al., 2009b; Ganesan et al., 2010; Schatz et al., 2007) have been ported to graphics hardware in the last years. SARUMAN was the first short read matching program to make use of this promising technique, and the alignment module of SARUMAN was the first implementation of the Needleman-Wunsch algorithm on graphics hardware. The trend was quickly adapted by other short read matching softwares, in 2012 CUSHAW and SOAP3 were published, both using GPU programming for short read alignment.
SARUMAN does not natively support paired end sequencing data, but as all possible alignments are returned, paired end hits in correct distance and orientation can be easily extracted from the mapping results in a postprocessing step. Furthermore, SARUMAN ignores quality values from FASTQ files. The limiting factor to the speed of the alignment step is the memory usage of one read in the graphics

card's main memory. As quality values would double the memory requirements, the usage of these values is not an option for SARUMAN. Due to the huge number of reads produced by modern NGS systems, a better solution would be to discard all reads with deficient qualities prior to any processing.
SARUMAN is a short read alignment software that is tailored to show a superior performance on microbial genomes. The drawback of this approach is that SARUMAN is not suitable for large eukaryotic genomes. Due to the high accuracy constraints that permitted memory saving techniques like two-bit-encoding, the genome index used in the filter algorithm is very memory consuming, thus reference sequences that are bigger than small plant chromosomes can not be processed in a single step on a standard desktop computer. Another limitation of SARUMAN is that the performance will decrease with increasing read length as the number of reads that can be aligned in parallel on the graphics card is reduced. There is no fixed limit to the read length, the software was tested successfully with 400bp reads, but the performance gain by the use of GPU programming is dwindling for longer reads. As a rule of thumb, with up-to-date graphics adapters a read length of 150bp is the limit at which SARUMAN can be used with reasonable running times (but this hardly depends on the available VRAM).
In summary, the goals defined in Section 3.2 for the second part of this work were also achieved to full extend. With SARUMAN a CUDA-accelerated software was developed that provides an accurate and fast short read alignment for microbial genomes.

Outlook: Besides the limitations in the length of reads and reference genomes that can be processed, there are several other aspects in which SARUMAN still can be improved. Obviously, it would be a great advantage to make SARUMAN applicable for large genomes. One option to do so would be to use an index for the filter algorithm that is based on the FM index as this data structure combines fast access times with a low memory footprint. The low memory usage of such an index would allow the usage of SARUMAN also for larger genomes. Another option that would come with a drastically smaller genome index would be to implement a new filter algorithm that can be executed on graphics hardware like the alignment step. The fact that CUSHAW and SOAP3 recently demonstrated the implementation of FM index-like datastructures on graphics hardware encourages this idea.
Another planned extension of SARUMAN, the native support for the Sequence Alignment/Map (SAM) format (Li et al., 2009a), would be highly desirable. The SAM format has become the widely accepted standard for short read alignment results. Unfortunately, during the development of SARUMAN this was not yet foreseeable, thus SARUMAN provides a proprietary TAB-separated output format. SARUMAN output can be converted to SAM format easily, but for a native support of the SAM format the alignment result has to be represented as a CIGAR string, a compressed representation of an alignment. To create such CIGAR string, major

changes to the CUDA alignment module would be needed, thus the SAM support is not a trivial task.

6.3. Conclusion

All in all, the presented tools EDGAR and SARUMAN are important contributions to the field of comparative genomics. EDGAR is a convenient tool for the in-depth analysis of differential gene contents in microbial genomes. It has one of the most comprehensive sets of analysis features of all available genome comparison tools. SARUMAN is a maximal accurate and fast short read alignment program for small genomes. For microbial genomes it can be considered the short read alignment program with the best overall performance in terms of runtime and accuracy. Both applications have the potential to make an important contribution to the field of comparative genomics and to microbial genome research in general.
The gene set comparisons of EDGAR bear great potential for reverse vaccinology and drug design as well as for the comparative analysis of pathogenic or industrially important bacteria. The easy determination of genes of interest simplifies the identification of virulence factors in pathogens or targets for genetic engineering in production organisms. The phylogenetic features and geneset statistics of EDGAR are perfect tools to deepen our understanding of bacterial evolution.
Other application cases in comparative genomics need a more fine grained approach, e.g., bacterial re-sequencing or a SNP analysis of closely related strains of the same species. For these application cases SARUMAN represents the perfect mapping tool, as its exact and complete mapping result ensures the highest possible sensitivity in the analysis of smallest genomic differences. Furthermore there are several other fields in genome research were an accurate read mapping is crucial, e.g., RNA-seq or ChIP-seq analyses.
As discussed in the previous sections, the constant technical progress of sequencing technology will bring new challenges in the future, but with a foresightful further development both tools bear the potential to stay valuable tools in the field of comparative genomics.

APPENDIX A

Appendix

A.1. Artificial dataset: 75bp reads

Evaluation of SARUMAN using a synthetic read set of roughly 18 million reads generated from the *Escherichia coli K12 MG1655* genome (GenBank accession NC_000913) with reads of 75bp length. Reads were generated from both strands with up to two errors of different types, i.e. mismatches, insertions, deletions, and combinations thereof. About three million of the artificial reads contained indels. The settings for all programs used in the comparisons were adjusted to make the run parameters as comparable as possible. Reads were mapped with 2 allowed errors/mismatches, all programs were set to support multi-threading. Furthermore, gapped alignments were supported if possible, and the alignment scoring costs were set to simple unit costs. Results are shown in Table A.1.

Table A.1.: Artificial reads 75bp: Sensitivity evaluation with an artificial dataset of 17,980,142 reads (75bp) generated from *Escherichia coli K12 MG1655*. %MCP (Mapped to Correct Position) denotes the percentage of mapped reads that had a match to their original position in the source genome. %BMCP (Best Match at Correct Position) denotes the percentage of reads where the best match was located at the correct position.

	75bp reads									
	SARUMAN	SOAP2	Bowtie	Bowtie2	BWA	CUSHAW	CUSHAW2	MRFAST	SHRiMP	PASS	Reference
Mapped	17,980,142	15,142,908	15,123,838	17,777,306	17,746,484	5,200,726	17,159,304	14,960,950	17,980,142	16,873,044	17,980,142
Not mapped	0	2,837,234	2,856,304	202,836	233,658	12,779,416	820,838	3,019,192	0	1,107,098	0
Perfect	4,999,944	4,999,942	4,999,944	4,999,944	4,999,944	4,999,944	4,999,944	3,429,166	4,999,944	4,999,944	4,999,944
With errors	12,980,198	10,142,966	10,123,894	12,777,362	12,746,540	200,782	12,159,360	11,531,784	12,980,198	11,873,100	12,980,198
1 mismatch	4,999,908	4,999,906	4,999,908	4,999,908	4,999,908	133,534	4,800,130	4,325,550	4,999,908	4,999,908	4,999,908
2 mismatches	4,999,936	4,999,936	4,999,936	4,999,936	4,999,936	6,934	4,605,513	4,728,484	4,999,936	4,999,936	4,999,936
1 Insertion	499,992	34,648	29,900	494,774	489,788	6,682	472,711	433,604	499,992	478,754	499,992
2 Insertions	499,998	1,243	496	396,332	405,563	190	445,554	473,858	499,998	496	499,998
1 Deletion	493,560	46,740	46,740	492,342	491,454	46,740	473,645	413,626	493,560	475,916	493,560
2 Deletion	493,354	4,828	4,828	455,916	433,605	4,828	454,587	226,090	493,354	452,714	493,354
1 Ins. & 1 Mism.	500,000	21,374	12,308	464,486	458,957	478	453,680	473,276	500,000	18,334	500,000
1 Del. & 1 Mism.	493,450	34,291	29,778	473,668	467,329	1,396	453,540	457,296	493,450	447,042	493,450
MCP	17,899,092	15,070,286	15,061,178	17,679,766	17,432,842	5,022,325	17,104,859	14,886,244	17,785,567	16,577,180	-
BMCP	17,899,086	15,070,286	15,061,178	17,679,712	17,430,632	5,022,325	17,104,859	14,886,238	17,784,713	16,577,173	-
Total Alignments	19,971,674	16,425,886	16,542,168	19,437,878	18,022,317	5,200,726	18,602,453	16,350,910	19,423,932	18,447,418	-
Runtime	12:03 min	06:40 min	18:56 min	15m:50 min	15:09 min	03:11 min	20:28 min	07:09 min	95:06 min	27:42 min	-
RAM usage	3,375 Mb	702.9 Mb	14.4 Mb	44.8 Mb	117.2 Mb	750.6 Mb	3,012 Mb	11,734 Mb	1,313 Mb	399.3 Mb	-

Bibliography

C. Alkan, J.M. Kidd, T. Marques-Bonet, G. Aksay, F. Antonacci, F. Hormozdiari, J.O. Kitzman, C. Baker, M. Malig, O. Mutlu, S.C. Sahinalp, R.A. Gibbs, and E.E. Eichler. Personalized copy number and segmental duplication maps using next-generation sequencing. *Nature Genetics*, 41(10):1061–1067, 2009.

C. Alkan, S. Sajjadian, and E.E. Eichler. Limitations of next-generation genome sequence assembly. *Nature Methods*, 8 (1):61–65, 2010.

C. Allix, P. Supply, and M. Fauville-Dufaux. Utility of Fast Mycobacterial Interspersed Repetitive Unit-Variable Number Tandem Repeat Genotyping in Clinical Mycobacteriological Analysis. *Clinical Infectious Diseases*, 39(6):783–789, 2004.

A.M. Altenhoff and C. Dessimoz. Phylogenetic and functional assessment of orthologs inference projects and methods. *PLoS Computational Biology*, 5(1):e1000262, 2009.

A.M. Altenhoff, A. Schneider, G.H. Gonnet, and C. Dessimoz. OMA 2011: orthology inference among 1000 complete genomes. *Nucleic Acids Research*, 39(suppl 1):D289–D294, 2011.

S.F. Altschul, W. Gish, W. Miller, Myers. E.W., and D.J. Lipman. Basic local alignment search tool. *Journal of Molecular Biology*, 215(3):403–410, 1990. doi: 10.1006/jmbi.1990.9999.

M. Ansari, G. Yadav, R.S. Gokhale, and D. Mohanty. NRPS-PKS: a knowledge-based resource for analysis of NRPS/PKS megasynthases. *Nucleic Acids Research*, 32(suppl 2):W405–W413, 2004.

S. Assefa, T.M. Keane, T.D. Otto, C. Newbold, and M. Berriman. ABACAS: algorithm-based automatic contiguation of assembled sequences. *Bioinformatics*, 25(15):1968–1969, 2009.

J.M. Aury, C. Cruaud, V. Barbe, O. Rogier, S. Mangenot, G. Samson, J. Poulain, V. Anthouard, C. Scarpelli, F. Artiguenave, and P. Wincker. High quality draft sequences for prokaryotic genomes using a mix of new sequencing technologies. *BMC Genomics*, 9(1):603, 2008.

R. Aziz, D. Bartels, A. Best, M. DeJongh, T. Disz, R. Edwards, K. Formsma, S. Gerdes, E. Glass, M. Kubal, F. Meyer, G.J. Olsen, R. Olson, Osterman A.L., Overbeek R.A., L.K. McNeil, D. Paarmann, T. Paczian, B. Parrello, G.D. Pusch, C. Reich, R. Stevens, O. Vassieva, V. Vonstein, A. Wilke, and O. Zagnitko1. The RAST Server: rapid annotations using subsystems technology. *BMC Genomics*, 9(1):75, 2008.

A.K. Bachhawat. Comparative genomics. *Resonance*, 11(8):22–40, 2006.

D. Bartels, S. Kespohl, S. Albaum, T. Drüke, A. Goesmann, J. Herold, O. Kaiser, A. Pühler, F. Pfeiffer, G. Raddatz, et al. BACCardI - a tool for the validation of genomic assemblies, assisting genome finishing and intergenome comparison. *Bioinformatics*, 21(7):853–859, 2005.

D.R. Bentley, S. Balasubramanian, H.P. Swerdlow, G.P. Smith, J. Milton, C.G. Brown, K.P. Hall, D.J. Evers, C.L. Barnes, H.R. Bignell, et al. Accurate whole human genome sequencing using reversible terminator chemistry. *Nature*, 456 (7218):53–59, 2008.

F.R. Blattner, G. Plunkett III, C.A. Bloch, N.T. Perna, V. Burland, M. Riley, J. Collado-Vides, J.D. Glasner, C.K. Rode, G.F. Mayhew, J Gregor, N.W. Davis, H.A. Kirkpatrick, M. A. Goeden, D.R. Rose, B. Mau, and Y. Shao. The complete genome sequence of *Escherichia coli* K-12. *Science*, 277(5331):1453–1462, 1997.

J. Blom, S.P. Albaum, D. Doppmeier, A. Pühler, F.J. Vorhölter, M. Zakrzewski, and A. Goesmann. EDGAR: a software framework for the comparative analysis of prokaryotic genomes. *BMC Bioinformatics*, 10(1):154, 2009. ISSN 1471-2105.

J. Blom, T. Jakobi, D. Doppmeier, S. Jaenicke, J. Kalinowski, J. Stoye, and A. Goesmann. Exact and complete short-read alignment to microbial genomes using Graphics Processing Unit programming. *Bioinformatics*, 27(10):1351–1358, 2011.

J. Blom, C. Rückert, B. Niu, Q. Wang, and R. Borriss. The Complete Genome of *Bacillus amyloliquefaciens* subsp. *plantarum* CAU B946 Contains a Gene Cluster for Nonribosomal Synthesis of Iturin A. *Journal of Bacteriology*, 194 (7):1845–1846, 2012.

A. Bolotin, B. Quinquis, P. Renault, A. Sorokin, S.D. Ehrlich, S. Kulakauskas, A. Lapidus, E. Goltsman, M. Mazur, G.D. Pusch, M. Fonstein, R. Overbeek, N. Kyprides, B. Purnelle, D. Prozzi, K. Ngui, D. Masuy, F. Hancy, S. Burteau, M. Boutry, J. Delcour, A. Goffeau, and P. Hols. Complete sequence and comparative genome analysis of the dairy bacterium *Streptococcus thermophilus*. *Nature Biotechnology*, 22(12):1554–1558, 2004.

R. Borriss, X.-H. Chen, C. Rückert, J. Blom, A. Becker, B. Baumgarth, B. Fan, R. Pukall, P. Schumann, C. Sproer, H. Junge, J. Vater, A. Pühler, and H.-P. Klenk. Relationship of *Bacillus amyloliquefaciens* clades associated with strains DSM 7T and FZB42T: a proposal for *Bacillus amyloliquefaciens* subsp. *amyloliquefaciens* subsp. nov. and *Bacillus amyloliquefaciens* subsp. *plantarum* subsp. nov. based on complete genome sequence comparisons. *International Journal of Systematic and Evolutionary Microbiology*, 61(Pt 8), 2011.

K. Brankatschk, J. Blom, A. Goesmann, T.H.M. Smits, and B. Duffy. Genome of a European fresh-vegetable food safety outbreak strain of *Salmonella enterica* subsp. *enterica* serovar *weltevreden*. *Journal of Bacteriology*, 193(8): 2066–2066, 2011.

K. Brankatschk, J. Blom, A. Goesmann, T.H.M. Smits, and B. Duffy. Comparative genomic analysis of *Salmonella enterica* subsp. *enterica* serovar *weltevreden* foodborne strains with other serovars. *International Journal of Food Microbiology*, 155(3):247–256, 2012.

E. Branscomb and P. Predki. On the high value of low standards. *Journal of Bacteriology*, 184(23):6406–6409, 2002.

M. Brudno, C.B. Do, G.M. Cooper, M.F. Kim, E. Davydov, E.D. Green, A. Sidow, and S. Batzoglou. LAGAN and Multi-LAGAN: efficient tools for large-scale multiple alignment of genomic DNA. *Genome Research*, 13(4):721–731, 2003.

M. Brudno, R. Steinkamp, and B. Morgenstern. The CHAOS/DIALIGN WWW server for multiple alignment of genomic sequences. *Nucleic Acids Research*, 32(suppl 2):W41–W44, 2004.

E. Brzuszkiewicz, H. Brüggemann, H. Liesegang, M. Emmerth, T. Ölschläger, G. Nagy, K. Albermann, C. Wagner, C. Buchrieser, L. Emődy, G. Gottschalk, J. Hacker, and U. Dobrindt. How to become a uropathogen: comparative genomic analysis of extraintestinal pathogenic *Escherichia coli* strains. *Proceedings of the National Academy of Sciences*, 103(34):12879–12884, 2006.

M. Burrows and D. J. Wheeler. A block-sorting lossless data compression algorithm. Technical Report 124, DEC System Resource Center (SRC), 1994.

A. Califano and I. Rigoutsos. FLASH: A fast look-up algorithm for string homology. In *Computer Vision and Pattern Recognition, 1993. Proceedings CVPR'93., 1993 IEEE Computer Society Conference on*, pages 353–359. IEEE, 1993.

D. Campagna, A. Albiero, A. Bilardi, E. Caniato, C. Forcato, S. Manavski, N. Vitulo, and G. Valle. PASS: a program to align short sequences. *Bioinformatics*, 25(7):967, 2009. ISSN 1367-4803.

R.W. Carthew and E.J. Sontheimer. Origins and mechanisms of miRNAs and siRNAs. *Cell*, 136(4):642–655, 2009.

R.R. Chaudhuri and M.J. Pallen. xBASE, a collection of online databases for bacterial comparative genomics. *Nucleic Acids Research*, 34(Database issue):D335–D337, 2006.

R.R. Chaudhuri, N.J. Loman, L.A.S. Snyder, C.M. Bailey, D.J. Stekel, and M.J. Pallen. xBASE2: a comprehensive resource for comparative bacterial genomics. *Nucleic Acids Research*, 36(Database issue):D543–D546, 2008.

B. Chevreux. *MIRA: an automated genome and EST assembler*. PhD thesis, German Cancer Research Center Heidelberg, 2005.

F.D. Ciccarelli, T. Doerks, C. Von Mering, C.J. Creevey, B. Snel, and P. Bork. Toward automatic reconstruction of a highly resolved tree of life. *Science*, 311(5765):1283–1287, 2006.

R. Ciria, C. Abreu-Goodger, E. Morett, and E. Merino. GeConT: gene context analysis. *Bioinformatics*, 20(14):2307–2308, 2004.

A.C.E. Darling, B. Mau, F.R. Blattner, and N.T. Perna. Mauve: multiple alignment of conserved genomic sequence with rearrangements. *Genome Research*, 14(7):1394–1403, 2004.

T. Davidsen, E. Beck, A. Ganapathy, R. Montgomery, N. Zafar, Q. Yang, R. Madupu, P. Goetz, K. Galinsky, O. White, and G. Sutton. The comprehensive microbial resource. *Nucleic Acids Research*, 38(suppl 1):D340–D345, 2010.

P. De Maayer, W.Y. Chan, F. Rezzonico, A. Bühlmann, S.N. Venter, J. Blom, A. Goesmann, J.E. Frey, T.H.M. Smits, B. Duffy, and T.A. Coutinho. Complete Genome Sequence of Clinical Isolate *Pantoea ananatis* LMG 5342. *Journal of Bacteriology*, 194(6):1615–1616, 2012.

T.F. DeLuca, J. Cui, J.Y. Jung, K.C.S. Gabriel, and D.P. Wall. Roundup 2.0: enabling comparative genomics for over 1800 genomes. *Bioinformatics*, 28(5):715–716, 2012.

X. Deng, A.M. Phillippy, Z. Li, S.L. Salzberg, and W. Zhang. Probing the pan-genome of *Listeria monocytogenes*: new insights into intraspecific niche expansion and genomic diversification. *BMC Genomics*, 11(1):500, 2010.

C. Dessimoz, G. Cannarozzi, M. Gil, D. Margadant, A. Roth, A. Schneider, and G.H. Gonnet. OMA, a comprehensive, automated project for the identification of orthologs from complete genome data: Introduction and first achievements. *Lecture Notes in Computer Science*, 3678:61–72, 2005.

M. Dondrup, S. Albaum, T. Griebel, K. Henckel, S. Jünemann, T. Kahlke, C. Kleindt, H. Küster, B. Linke, D. Mertens, V. Mittard-Runte, H. Neuweger, K.J. Runte, A. Tauch, F. Tille, A. Pühler, and A. Goesmann. EMMA 2 - A MAGE-compliant system for the collaborative analysis and integration of microarray data. *BMC Bioinformatics*, 10(1):50, 2009.

S.R. Eddy. Accelerated profile HMM searches. *PLoS Computational Biology*, 7(10):e1002195, 2011.

R.C. Edgar. MUSCLE: multiple sequence alignment with high accuracy and high throughput. *Nucleic Acids Research*, 32(5):1792, 2004.

J. Eid, A. Fehr, J. Gray, K. Luong, J. Lyle, G. Otto, P. Peluso, D. Rank, P. Baybayan, B. Bettman, A. Bibillo, K. Bjornson, B. Chaudhuri, F. Christians, R. Cicero, S. Clark, R. Dalal, A. deWinter, J. Dixon, M. Foquet, A. Gaertner, P. Hardenbol, C. Heiner, K. Hester, D. Holden, G. Kearns, X. Kong, R. Kuse, Y. Lacroix, S. Lin, P. Lundquist, C. Ma, P. Marks, M. Maxham, D. Murphy, I. Park, T. Pham, M. Phillips, J. Roy, R. Sebra, S. Shen, J. Sorenson, A. Tomaney, K. Travers, M. Trulson, J. Vieceli, J. Wegener, D. Wu, A. Yang, D. Zaccarin, P. Zhao, F. Zhong, J. Korlach, and S. Turner. Real-time DNA sequencing from single polymerase molecules. *Science*, 323(5910):133–138, 2009.

M. Eppinger, C. Baar, G. Raddatz, D.H. Huson, and S.C. Schuster. Comparative analysis of four Campylobacterales. *Nature Reviews Microbiology*, 2(11):872–885, 2004.

M. Farrar. Striped Smith-Waterman speeds database searches six times over other SIMD implementations. *Bioinformatics*, 23(2):156–161, 2007. doi: 10.1093/bioinformatics/btl582.

J. Felsenstein. PHYLIP (Phylogeny Inference Package), version 3.57 c, 1995. Distributed by the author.

X. Feng, R. Grossman, and L. Stein. PeakRanger: A cloud-enabled peak caller for ChIP-seq data. *BMC Bioinformatics*, 12(1):139, 2011.

P. Ferragina and G. Manzini. Opportunistic data structures with applications. In *Foundations of Computer Science, 2000. Proceedings. 41st Annual Symposium on*, pages 390–398. IEEE, 2000.

P. Ferragina and G. Manzini. Indexing compressed text. *Journal of the ACM*, 52(4):552–581, 2005.

W.M. Fitch. Distinguishing homologous from analogous proteins. *Systematic Biology*, 19(2):99–113, 1970.

W.M. Fitch. Homology: a personal view on some of the problems. *Trends in Genetics*, 16(5):227–231, 2000.

C.B. Ford, P.L. Lin, M.R. Chase, R.R. Shah, O. Iartchouk, J. Galagan, N. Mohaideen, T.R. Ioerger, J.C. Sacchettini, M. Lipsitch, J.L. Flynn, and Fortune S.M. Use of whole genome sequencing to estimate the mutation rate of *Mycobacterium tuberculosis* during latent infection. *Nature Genetics*, 43(5):482–486, 2011.

R.E. Franklin and R.G. Gosling. Molecular configuration in sodium thymonucleate. *Nature*, 171(4356):740–741, 1953.

C.M. Fraser, J.A. Eisen, K.E. Nelson, I.T. Paulsen, and S.L. Salzberg. The value of complete microbial genome sequencing (you get what you pay for). *Journal of Bacteriology*, 184(23):6403–6405, 2002.

K.A. Frazer, L. Pachter, A. Poliakov, E.M. Rubin, and I. Dubchak. VISTA: computational tools for comparative genomics. *Nucleic Acids Research*, 32(Web Server issue):W273–W279, 2004.

A.M. Galazka, S.E. Robertson, and G.P. Oblapenko. Resurgence of diphtheria. *European Journal of Epidemiology*, 11 (1):95–105, 1995.

N. Ganesan, R.D. Chamberlain, J. Buhler, and M. Taufer. Accelerating HMMER on GPUs by implementing hybrid data and task parallelism. In *Proceedings of the First ACM International Conference on Bioinformatics and Computational Biology*, pages 418–421. ACM, ACM, 2010.

T.C. Glenn. Field guide to next-generation DNA sequencers. *Molecular Ecology Resources*, 11(5):759–769, 2011.

A.A. Gontcharov, B. Marin, and M. Melkonian. Are combined analyses better than single gene phylogenies? A case study using SSU rDNA and rbcL sequence comparisons in the Zygnematophyceae (Streptophyta). *Molecular Biology and Evolution*, 21(3):612–624, 2004.

D. Gordon, C. Abajian, and P. Green. Consed: a graphical tool for sequence finishing. *Genome Research*, 8(3):195–202, 1998.

S. Griffiths-Jones, R.J. Grocock, S. Van Dongen, A. Bateman, and A.J. Enright. miRBase: microRNA sequences, targets and gene nomenclature. *Nucleic Acids Research*, 34(suppl 1):D140–D144, 2006.

A.S.D. Groot and R. Rappuoli. Genome-derived vaccines. *Expert Review of Vaccines*, 3(1):59–76, 2004.

F. Hach, F. Hormozdiari, C. Alkan, F. Hormozdiari, I. Birol, E.E. Eichler, and S.C. Sahinalp. mrsFAST: a cache-oblivious algorithm for short-read mapping. *Nature Methods*, 7(8):576–577, 2010.

M. Hackl, T. Jakobi, J. Blom, D. Doppmeier, K. Brinkrolf, R. Szczepanowski, S.H. Bernhart, C.H. Siederdissen, J.A.H. Bort, M. Wieser, R. Kunert, S. Jeffs, I.L. Hofacker, A. Goesmann, A. Pühler, N. Borth, and J. Grillari. Next-generation sequencing of the Chinese hamster ovary microRNA transcriptome: Identification, annotation and profiling of microRNAs as targets for cellular engineering. *Journal of Biotechnology*, 153(1):62–75, 2011.

R.S. Harris. *Improved pairwise alignment of genomic DNA*. ProQuest, 2007.

H.S. Heaps. *Information retrieval: Computational and theoretical aspects*. Academic Press, Inc., 1978.

S. Heinl, D. Wibberg, F.G. Eikmeyer, R. Szczepanowski, J. Blom, B. Linke, A. Goesmann, R. Grabherr, H. Schwab, A. Pühler, and A. Schlüter. Insights into the completely annotated genome of *Lactobacillus buchneri* CD034, a strain isolated from stable grass silage. *Journal of Biotechnology*, 161(2):153–166, 2012.

S. Henikoff and J.G. Henikoff. Amino acid substitution matrices from protein blocks. *Proceedings of the National Academy of Sciences*, 89(22):10915, 1992.

N.L. Hiller, B. Janto, J.S. Hogg, R. Boissy, S. Yu, E. Powell, R. Keefe, N.E. Ehrlich, K. Shen, J. Hayes, K. Barbadora, W. Klimke, D. Dernovoy, T. Tatusova, J. Parkhill, S.D. Bentley, J.C. Post, G.D. Ehrlich, and F.Z. Hu. Comparative genomic analyses of seventeen *Streptococcus pneumoniae* strains: insights into the Pneumococcal Supragenome. *Journal of Bacteriology*, 189(22):8186–8195, 2007.

M. Höhl, S. Kurtz, and E. Ohlebusch. Efficient multiple genome alignment. *Bioinformatics*, 18(suppl 1):S312–S320, 2002.

K. Hollricher. Microbial systematics - Species Don't Really Mean Anything in the Bacterial World. *Lab Times*, 5:22–25, 2007.

X. Huang and A. Madan. CAP3: A DNA sequence assembly program. *Genome Research*, 9(9):868–877, 1999.

T. Hulsen, M.A. Huynen, J. De Vlieg, and P.M.A. Groenen. Benchmarking ortholog identification methods using functional genomics data. *Genome Biology*, 7(4):R31, 2006.

P. Husemann and J. Stoye. r2cat: synteny plots and comparative assembly. *Bioinformatics*, 26(4):570–571, 2010.

L. Ilie and S. Ilie. Multiple spaced seeds for homology search. *Bioinformatics*, 23(22):2969–2977, 2007.

F. Imperi, L. Antunes, J. Blom, L. Villa, M. Iacono, P. Visca, and A. Carattoli. The genomics of *Acinetobacter baumannii*: Insights into genome plasticity, antimicrobial resistance and pathogenicity. *IUBMB Life*, 63(12):1068–1074, 2011.

O. Jahns. *ChIP-basierter Screen und Untersuchungen zu Protein-DNA-Interaktionen ausgewählter Regulatoren der Flavonoidbiosynthese in Arabidopsis thaliana.* PhD thesis, Bielefeld University, 2012.

D.S. Johnson, A. Mortazavi, R.M. Myers, and B. Wold. Genome-wide mapping of in vivo protein-DNA interactions. *Science*, 316(5830):1497, 2007.

I. Jolliffe. *Principal component analysis.* Wiley Online Library, 2005.

D.T. Jones. GenTHREADER: an efficient and reliable protein fold recognition method for genomic sequences. *Journal of Molecular Biology*, 287(4):797–815, 1999.

R. Kant, J. Blom, A. Palva, R.J. Siezen, and W.M. de Vos. Comparative genomics of Lactobacillus. *Microbial Biotechnology*, 4(3):323–332, 2011.

J. Kieleczawa, J.J. Dunn, and F.W. Studier. DNA sequencing by primer walking with strings of contiguous hexamers. *Science*, 258(5089):1787–1791, 1992.

M. Kimura. *The neutral theory of molecular evolution.* Cambridge University Press, 1985.

S. Kurtz, A. Phillippy, A.L. Delcher, M. Smoot, M. Shumway, C. Antonescu, and S.L. Salzberg. Versatile and open software for comparing large genomes. *Genome Biology*, 5(2):R12, 2004.

C. Laing, C. Buchanan, E. Taboada, Y. Zhang, A. Kropinski, A. Villegas, J. Thomas, and V. Gannon. Pan-genome sequence analysis using Panseq: an online tool for the rapid analysis of core and accessory genomic regions. *BMC Bioinformatics*, 11(1):461, 2010.

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, 2001.

B. Langmead and S.L. Salzberg. Fast gapped-read alignment with Bowtie 2. *Nature Methods*, 9(4):357–359, 2012.

B. Langmead, C. Trapnell, M. Pop, and S.L. Salzberg. Ultrafast and memory-efficient alignment of short DNA sequences to the human genome. *Genome Biology*, 10(3):R25, 2009.

S.J. Lee, D.Y. Lee, T.Y. Kim, B.H. Kim, J. Lee, and S.Y. Lee. Metabolic engineering of *Escherichia coli* for enhanced production of succinic acid, based on genome comparison and in silico gene knockout simulation. *Applied and Environmental Microbiology*, 71(12):7880–7887, 2005a.

S.Y. Lee, D.Y. Lee, and T.Y. Kim. Systems biotechnology for strain improvement. *Trends in Biotechnology*, 23(7): 349–358, 2005b.

T. Lefébure and M.J. Stanhope. Evolution of the core and pan-genome of Streptococcus: positive selection, recombination, and genome composition. *Genome Biology*, 8(5):R71, 2007.

E. Lerat, V. Daubin, and N.A. Moran. From gene trees to organismal phylogeny in prokaryotes: the case of the gamma-Proteobacteria. *PLoS Biology*, 1(1):E19, 2003.

H. Li and R. Durbin. Fast and accurate short read alignment with Burrows-Wheeler transform. *Bioinformatics*, 25(14): 1754–1760, 2009.

H. Li, B. Handsaker, A. Wysoker, T. Fennell, J. Ruan, N. Homer, G. Marth, G. Abecasis, R. Durbin, and 1000 Genome Project Data Processing Subgroup. The Sequence Alignment/Map format and SAMtools. *Bioinformatics*, 25(16): 2078–2079, 2009a.

L. Li, C.J. Stoeckert, and D.S. Roos. OrthoMCL: identification of ortholog groups for eukaryotic genomes, 2003.

R. Li, Y. Li, K. Kristiansen, and J. Wang. SOAP: short oligonucleotide alignment program. *Bioinformatics*, 24(5): 713–714, 2008.

R. Li, C. Yu, Y. Li, T.-W. Lam, S.-M. Yiu, K. Kristiansen, and J. Wang. SOAP2: an improved ultrafast tool for short read alignment. *Bioinformatics*, 25(15):1966–1967, 2009b.

R. Li, H. Zhu, J. Ruan, W. Qian, X. Fang, Z. Shi, Y. Li, S. Li, G. Shan, K. Kristiansen, L. Li, H. Yang, Ji. Wang, and Ju. Wang. De novo assembly of human genomes with massively parallel short read sequencing. *Genome Research*, 20 (2):265–272, 2010.

B. Linke, R. Giegerich, and A. Goesmann. Conveyor: a workflow engine for bioinformatic analyses. *Bioinformatics*, 27 (7):903–911, 2011.

C.M. Liu, T. Wong, E. Wu, R. Luo, S.M. Yiu, Y. Li, B. Wang, C. Yu, X. Chu, K. Zhao, et al. SOAP3: ultra-fast GPU-based parallel alignment tool for short reads. *Bioinformatics*, 28(6):878–879, 2012a.

L. Liu, G. Cheng, C. Wang, X. Pan, Y. Cong, Q. Pan, J. Wang, F. Zheng, F. Hu, and J. Tang. Identification and experimental verification of protective antigens against *Streptococcus suis* serotype 2 based on genome sequence analysis. *Current Microbiology*, 58(1):11–17, 2009a.

W. Liu, B. Schmidt, G. Voss, A. Schroder, and W. Muller-Wittig. Bio-sequence database scanning on a GPU. In *Parallel and Distributed Processing Symposium, 2006. IPDPS 2006. 20th International*, page 8, 2006.

Y. Liu, D.L. Maskell, and B. Schmidt. CUDASW++: optimizing Smith-Waterman sequence database searches for CUDA-enabled graphics processing units. *BMC Research Notes*, 2:73, 2009b.

Y. Liu, B. Schmidt, and D.L. Maskell. CUSHAW: a CUDA compatible short read aligner to large genomes based on the Burrows-Wheeler transform. *Bioinformatics*, 28(14):1830–1837, 2012b.

P. Llop, J. Cabrefiga, T.H.M. Smits, T. Dreo, S. Barbé, J. Pulawska, A. Bultreys, J. Blom, B. Duffy, E. Montesinos, and M. M Lopez. *Erwinia amylovora* novel plasmid pEI70: Complete sequence, biogeography, and role in aggressiveness in the fire blight phytopathogen. *PLoS One*, 6(12), 2011.

F. Luciani, S.A. Sisson, H. Jiang, A.R. Francis, and M.M. Tanaka. The epidemiological fitness cost of drug resistance in *Mycobacterium tuberculosis*. *Proceedings of the National Academy of Sciences*, 106(34):14711–14715, 2009.

O. Lukjancenko, T.M. Wassenaar, and D.W. Ussery. Comparison of 61 sequenced *Escherichia coli* genomes. *Microbial Ecology*, 60(4):708–720, 2010.

E. Lyons and M. Freeling. How to usefully compare homologous plant genes and chromosomes as DNA sequences. *The Plant Journal*, 53(4):661–673, 2008.

D. Maione, I. Margarit, C.D. Rinaudo, V. Masignani, M. Mora, M. Scarselli, H. Tettelin, C. Brettoni, E.T. Iacobini, R. Rosini, N. D'Agostino, L. Miorin, S. Buccato, M. Mariani, G. Galli, R. Nogarotto, V. Nardi-Dei, F. Vegni, C. Fraser, G. Mancuso, G. Teti, L.C. Madoff, L.C. Paoletti, R. Rappuoli, D.L. Kasper, J.L. Telford, and G. Grandi. Identification of a universal Group B streptococcus vaccine by multiple genome screen. *Science*, 309(5731):148–150, 2005.

K. Makarova, A. Slesarev, Y. Wolf, A. Sorokin, B. Mirkin, E. Koonin, A. Pavlov, N. Pavlova, V. Karamychev, N. Polouchine, et al. Comparative genomics of the lactic acid bacteria. *Proceedings of the National Academy of Sciences*, 103 (42):15611–15616, 2006.

S.A. Manavski and G. Valle. CUDA compatible GPU cards as efficient hardware accelerators for Smith-Waterman sequence alignment. *BMC Bioinformatics*, 9 Suppl 2:S10, 2008.

R.A. Mann, J. Blom, A. Bühlmann, K.M. Plummer, S.V. Beer, J.E. Luck, A. Goesmann, J.E. Frey, B.C. Rodoni, B. Duffy, and T.H.M. Smits. Comparative analysis of the Hrp pathogenicity island of Rubus- and Spiraeoideae-infecting *Erwinia amylovora* strains identifies the IT region as a remnant of an integrative conjugative element. *Gene*, 2012.

M. Margulies, M. Egholm, W.E. Altman, S. Attiya, J.S. Bader, L.A. Bemben, J. Berka, M.S. Braverman, Y.J. Chen, Z. Chen, S.B. Dewell, L. Du, J.M. Fierro, X.V. Gomes, B.C. Godwin, W. He, S. Helgesen, C.H. Ho, C.H. Ho, G.P. Irzyk, S.C. Jando, M.L.I. Alenquer, T.P. Jarvie, K.B. Jirage, J.-B. Kim, J.R. Knight, J.R. Lanza, J.H. Leamon, S.M. Lefkowitz, M. Lei, J. Li, K.L. Lohman, H. Lu, V.M. Makhijani, K.E. McDade, M.P. McKenna, E.W. Myers, E. Nickerson, J.R. Nobile, R. Plant, B.P. Puc, M.T. Ronan, G.T. Roth, G.J. Sarkis, J.F. Simons, J.W. Simpson, M. Srinivasan, K.R. Tartaro, A. Tomasz, K.A. Vogt, G.A. Volkmer, S.H. Wang, Y. Wang, M.P. Weiner, P. Yu, R.F. Begley, and J.M. Rothberg. Genome sequencing in microfabricated high-density picolitre reactors. *Nature*, 437(7057): 376–380, 2005.

A.M. Maxam and W. Gilbert. A new method for sequencing DNA. *Proceedings of the National Academy of Sciences*, 74(2):560, 1977.

D. Medini, C. Donati, H. Tettelin, V. Masignani, and R. Rappuoli. The microbial pan-genome. *Current Opinion in Genetics & Development*, 15(6):589–594, 2005.

F. Meyer, A. Goesmann, A.C. McHardy, D. Bartels, T. Bekel, J. Clausen, J. Kalinowski, B. Linke, O. Rupp, R. Giegerich, and A. Pühler. GenDB - an open source genome annotation system for prokaryote genomes. *Nucleic Acids Research*, 31(8):2187–2195, 2003.

A.E. Minoche, J.C. Dohm, and H. Himmelbauer. Evaluation of genomic high-throughput sequencing data generated on Illumina HiSeq and Genome Analyzer systems. *Genome Biology*, 12(11):R112, 2011.

C.J. Mungall, D.B. Emmert, and The FlyBase Consortium. A Chado case study: an ontology-based modular schema for representing genome-associated biological information. *Bioinformatics*, 23(13):i337–i346, 2007.

A. Muzzi, V. Masignani, and r. Rappuoli. The pan-genome: towards a knowledge-based discovery of novel targets for vaccines and antibacterials. *Drug Discovery Today*, 12(11-12):429–439, 2007.

E.W. Myers, G.G. Sutton, A.L. Delcher, I.M. Dew, D.P. Fasulo, M.J. Flanigan, S.A. Kravitz, C.M. Mobarry, K.H.J. Reinert, K.A. Remington, E.L. Anson, R.A. Bolanos, H.-H. Chou, C.M. Jordan, A.L. Halpern, S. Lonardi, E.M. Beasley, R.C. Brandon, L. Chen, P.J. Dunn, Z. Lai, Y. Liang, D.R. Nusskern, M. Zhan, Q. Zhang, X. Zheng, G.M. Rubin, M.D. Adams, and J.C. Venter. A whole-genome assembly of Drosophila. *Science*, 287(5461):2196–2204, 2000.

H. Neuweger, S.P. Albaum, M. Dondrup, M. Persicke, T. Watt, K. Niehaus, J. Stoye, and A. Goesmann. MeltDB: a software platform for the analysis and integration of metabolomics experiment data. *Bioinformatics*, 24(23):2726–2732, 2008.

S. Niemann, C.U. Köser, S. Gagneux, C. Plinke, S. Homolka, H. Bignell, R.J. Carter, R.K. Cheetham, A. Cox, N.A. Gormley, P. Kokko-Gonzales, L.J. Murray, R. Rigatti, V.P. Smith, F.P.M. Arends, H.S. Cox, G. Smith, and Archer J.A.C. Genomic diversity among drug sensitive and multidrug resistant isolates of *Mycobacterium tuberculosis* with identical DNA fingerprints. *PLoS One*, 4(10):e7407, 2009.

P. Nyrén. Enzymatic method for continuous monitoring of DNA polymerase activity. *Analytical Biochemistry*, 167(2): 235–238, 1987.

K.P. O'Brien, M. Remm, and E.L.L. Sonnhammer. Inparanoid: a comprehensive database of eukaryotic orthologs. *Nucleic Acids Research*, 33(Database Issue):D476–D480, 2005.

H. Ogata, S. Goto, K. Sato, W. Fujibuchi, H. Bono, and M. Kanehisa. KEGG: Kyoto encyclopedia of genes and genomes. *Nucleic Acids Research*, 27(1):29–34, 1999.

J. Ohnishi, S. Mitsuhashi, M. Hayashi, S. Ando, H. Yokoi, K. Ochiai, and M. Ikeda. A novel methodology employing *Corynebacterium glutamicum* genome information to generate a new L-lysine-producing mutant. *Applied Microbiology and Biotechnology*, 58(2):217–223, 2002.

R. Overbeek, T. Begley, R.M. Butler, J.V. Choudhuri, H.Y. Chuang, M. Cohoon, V. de Crécy-Lagard, N. Diaz, T. Disz, R. Edwards, et al. The subsystems approach to genome annotation and its use in the project to annotate 1000 genomes. *Nucleic Acids Research*, 33(17):5691–5702, 2005.

I. Pagani, K. Liolios, J. Jansson, I.M.A. Chen, T. Smirnova, B. Nosrat, V.M. Markowitz, and N.C. Kyrpides. The Genomes OnLine Database (GOLD) v. 4: status of genomic and metagenomic projects and their associated metadata. *Nucleic Acids Research*, 40(D1):D571–D579, 2012.

V.R. Parreira, M. Costa, F.G. Eikmeyer, J. Blom, and J.F. Prescott. Sequence of Two Plasmids from *Clostridium perfringens* Chicken Necrotic Enteritis Isolates and Comparison with *C. perfringens* Conjugative Plasmids. *PLoS One*, 7(11), 2012.

K.D. Passalacqua, A. Varadarajan, B.D. Ondov, D.T. Okou, M.E. Zwick, and N.H. Bergman. Structure and complexity of a bacterial transcriptome. *Journal of Bacteriology*, 191(10):3203–3211, 2009.

M. Patek, J. Holatko, T. Busche, J. Kalinowski, and J. Nesvera. *Corynebacterium glutamicum* promoters: A practical approach. *Microbial Biotechnology*, 2013. accpeted, in press.

N.T. Perna, G. Plunkett, V. Burland, B. Mau, J.D. Glasner, D.J. Rose, G.F. Mayhew, P.S. Evans, J. Gregor, H.A. Kirkpatrick, G. Pósfai, J. Hackett, S. Klink, A. Boutin, Y. Shao, L. Miller, E.J. Grotbeck, N.W. Davis, A. Lim, E.T. Dimalanta, K.D. Potamousis, J. Apodaca, T.S. Anantharaman, J. Lin, G. Yen, D.C. Schwartz, R.A. Welch, and F.R. Blattner. Genome sequence of enterohaemorrhagic *Escherichia coli* O157:H7. *Nature*, 409(6819):529–533, 2001.

J.D. Peterson, L.A. Umayam, T. Dickinson, E.K. Hickey, and O. White. The comprehensive microbial resource. *Nucleic Acids Research*, 29(1):123–125, 2001.

D. Porter, J. Yao, and K. Polyak. SAGE and related approaches for cancer target identification. *Drug Discovery Today*, 11(3-4):110–118, 2006.

J. F. Pothier, T. H. Smits, J. Blom, F.-J. Vorhölter, A. Goesmann, A. Pühler, and B. Duffy. Complete genome sequence of the stone fruit pathogen *Xanthomonas arboricola* pv. *pruni*, 2011. Conference Abstract.

S. Powell, D. Szklarczyk, K. Trachana, A. Roth, M. Kuhn, J. Muller, R. Arnold, T. Rattei, I. Letunic, T. Doerks, L.J. Jensen, C. von Mering, and P. Bork. EggNOG v3. 0: orthologous groups covering 1133 organisms at 41 different taxonomic ranges. *Nucleic Acids Research*, 40(D1):D284–D289, 2012.

R. Powney, T.H.M. Smits, T. Sawbridge, B. Frey, J. Blom, J.E. Frey, K.M. Plummer, S.V. Beer, J. Luck, B. Duffy, and B. Rodoni. Genome Sequence of an *Erwinia amylovora* Strain with Pathogenicity Restricted to Rubus Plants. *Journal of Bacteriology*, 193(3):785, 2011.

C.A. Raabe, C.H. Hoe, G. Randau, J. Brosius, T.H. Tang, and T.S. Rozhdestvensky. The rocks and shallows of deep RNA sequencing: Examples in the *Vibrio cholerae* RNome. *RNA*, 17(7):1357–1366, 2011.

K.R. Rasmussen, J. Stoye, and E.W. Myers. Efficient q-gram filters for finding all ε-matches over a given length. *Journal of Computational Biology*, 13(2):296–308, 2006.

T. Rattei, P. Tischler, S. Götz, M.A. Jehl, J. Hoser, R. Arnold, A. Conesa, and H.W. Mewes. SIMAP - a comprehensive database of pre-calculated protein sequence similarities, domains, annotations and clusters. *Nucleic Acids Research*, 38(suppl 1):D223–D226, 2010.

M.L. Reno, N.L. Held, C.J. Fields, P.V. Burke, and R.J. Whitaker. Biogeography of the *Sulfolobus islandicus* pan-genome. *Proceedings of the National Academy of Sciences*, 106(21):8605, 2009.

D.C. Richter, S.C. Schuster, and D.H. Huson. OSLay: optimal syntenic layout of unfinished assemblies. *Bioinformatics*, 23(13):1573–1579, 2007.

D.R. Riley, S.V. Angiuoli, J. Crabtree, J.C.D. Hotopp, and H. Tettelin. Using Sybil for interactive comparative genomics of microbes on the web. *Bioinformatics*, 28(2):160–166, 2012.

A.I. Rissman, B. Mau, B.S. Biehl, A.E. Darling, J.D. Glasner, and N.T. Perna. Reordering contigs of draft genomes using the Mauve aligner. *Bioinformatics*, 25(16):2071–2073, 2009.

J.C. Roach, C. Boysen, K. Wang, and L. Hood. Pairwise end sequencing: a unified approach to genomic mapping and sequencing. *Genomics*, 26(2):345–353, 1995.

A. Roetzer, R. Diel, T.A. Kohl, C. Rückert, U. Nübel, J. Blom, T. Wirth, S. Jaenicke, S. Schuback, S. Rüsch-Gerdes, P. Supply, J. Kalinowski, and S. Niemann. Whole genome sequencing vs. traditional genotyping for investigation of a *Mycobacterium tuberculosis* outbreak: a longitudinal cohort study. *PLoS Medicine*, 2013. accepted.

T. Rognes and E. Seeberg. Six-fold speed-up of Smith-Waterman sequence database searches using parallel processing on common microprocessors. *Bioinformatics*, 16(8):699–706, 2000.

M. Ronaghi, S. Karamohamed, B. Pettersson, M. Uhlén, and P. Nyren. Real-time DNA sequencing using detection of pyrophosphate release. *Analytical Biochemistry*, 242(1):84–89, 1996.

S. Rozen and H. Skaletsky. Primer3 on the WWW for general users and for biologist programmers. *Methods in Molecular Biology*, 132(3):365–386, 2000.

C. Rückert, A. Pühler, and J. Kalinowski. Genome-wide analysis of the l-methionine biosynthetic pathway in *Corynebacterium glutamicum* by targeted gene deletion and homologous complementation. *Journal of Biotechnology*, 104(1): 213–228, 2003.

S.M. Rumble, P. Lacroute, A.V. Dalca, M. Fiume, A. Sidow, and M. Brudno. SHRiMP: accurate mapping of short color-space reads. *PLoS Computational Biology*, 5(5):e1000386, 2009.

N. Saitou and M. Nei. The neighbor-joining method: a new method for reconstructing phylogenetic trees. *Molecular Biology and Evolution*, 4(4):406–425, 1987.

F. Sanger and A.R. Coulson. A rapid method for determining sequences in DNA by primed synthesis with DNA polymerase. *Journal of Molecular Biology*, 94(3):441–446, 1975.

F. Sanger, G.M. Air, B.G. Barrell, N.L. Brown, A.R. Coulson, J.C. Fiddes, P.M. Slocombe, and M. Smith. Nucleotide sequence of bacteriophage (D X174 DNA. *Nature*, 265(5596):687–695, 1977.

M.C. Schatz, C. Trapnell, A.L. Delcher, and A. Varshney. High-throughput sequence alignment using Graphics Processing Units. *BMC Bioinformatics*, 8(1):474, 2007.

J.C. Schnable and E. Lyons. Comparative genomics with maize and other grasses: from genes to genomes! *Maydica*, 56 (2):183, 2011.

J. Schneider, J. Blom, S. Jaenicke, B. Linke, K. Brinkrolf, H. Neuweger, A. Tauch, and A. Goesmann. RAPYD-Rapid Annotation Platform for Yeast Data. *Journal of Biotechnology*, 155(1):118–126, 2011.

C. Schoen, J. Blom, H. Claus, A. Schramm-Glück, P. Brandt, T. Müller, A. Goesmann, B. Joseph, S. Konietzny, O. Kurzai, C. Schmitt, T. Friedrich, B. Linke, U. Vogel, and M. Frosch. Whole-genome comparison of disease and carriage strains provides insights into virulence evolution in *Neisseria meningitidis*. *Proceedings of the National Academy of Sciences*, 105(9):3473–3478, 2008.

J. Schröder, I. Maus, K. Meyer, S. Wördemann, J. Blom, S. Jaenicke, J. Schneider, E. Trost, and A. Tauch. Complete genome sequence, lifestyle, and multi-drug resistance of the human pathogen *Corynebacterium resistens* DSM 45100 isolated from blood samples of a leukemia patient. *BMC Genomics*, 13, 2012.

P. Schwientek, R. Szczepanowski, C. Rückert, J. Stoye, and A. Pühler. Sequencing of high G+C microbial genomes using the ultrafast pyrosequencing technology. *Journal of Biotechnology*, 2011.

C.M. Sharma, S. Hoffmann, F. Darfeuille, J. Reignier, S. Findeiß, A. Sittka, S. Chabas, K. Reiche, J. Hackermüller, R. Reinhardt, P.F. Stadler, and J. Vogel. The primary transcriptome of the major human pathogen *Helicobacter pylori*. *Nature*, 464(7286):250–255, 2010.

J.T. Simpson, K. Wong, S.D. Jackman, J.E. Schein, S.J.M. Jones, and I. Birol. ABySS: a parallel assembler for short read sequence data. *Genome Research*, 19(6):1117–1123, 2009.

T.F. Smith and M.S. Waterman. Identification of common molecular subsequences. *Journal of Molecular Biology*, 147 (1):195–197, 1981.

T.H.M. Smits, F. Rezzonico, T. Kamber, J. Blom, A. Goesmann, J.E. Frey, and B. Duffy. Complete Genome Sequence of the Fire Blight Pathogen *Erwinia amylovora* CFBP 1430 and Comparison to Other Erwinia spp. *Molecular Plant-Microbe Interactions*, 23(4):384–393, 2010. ISSN 0894-0282.

T.H.M. Smits, F. Rezzonico, T. Kamber, J. Blom, A. Goesmann, C.A. Ishimaru, J.E. Frey, V.O. Stockwell, and B. Duffy. Metabolic versatility and antibacterial metabolite biosynthesis are distinguishing genomic features of the fire blight antagonist *Pantoea vagans* C9-1. *PloS One*, 6(7):e22247, 2011.

B. Snel, G. Lehmann, P. Bork, and M.A. Huynen. STRING: a web-server to retrieve and display the repeatedly occurring neighbourhood of a gene. *Nucleic Acids Research*, 28(18):3442–3444, 2000.

S.C. Soares, V.A.C. Abreu, R.T.J. Ramos, L. Cerdeira, A. Silva, J. Baumbach, E. Trost, A. Tauch, R. Hirata, A.L. Mattos-Guaraldi, A. Miyoshi, and V. Azevedo. PIPS: pathogenicity island prediction software. *PloS One*, 7(2): e30848, 2012.

T. Strobel, A. Al-Dilaimi, J. Blom, A. Gessner, J. Kalinowski, M. Luzhetska, A. Pühler, R. Szczepanowski, A. Bechthold, and C. Rückert. Complete genome sequence of *Saccharothrix espanaensis* DSM 44229T and comparison to the other completely sequenced Pseudonocardiaceae. *BMC Genomics*, 13(1), 2012.

A. Szalkowski, C. Ledergerber, P. Krähenbühl, and C. Dessimoz. SWPS3 - fast multi-threaded vectorized Smith-Waterman for IBM Cell/B.E. and x86/SSE2. *BMC Research Notes*, 1:107, 2008.

D. Szklarczyk, A. Franceschini, M. Kuhn, M. Simonovic, A. Roth, P. Minguez, T. Doerks, M. Stark, J. Muller, P. Bork, L.J. Jensen, and C. von Mering. The STRING database in 2011: functional interaction networks of proteins, globally integrated and scored. *Nucleic Acids Research*, 39(suppl 1):D561–D568, 2011.

G. Talavera. Improvement of Phylogenies after Removing Divergent and Ambiguously Aligned Blocks from Protein Sequence Alignments. *Systematic Biology*, 56(4):564–577, 2007.

I. Tamas, L. Klasson, B. Canbäck, A.K. Näslund, A.-S. Eriksson, J.J. Wernegreen, J.P. Sandström, N.A. Moran, and S.G.E. Andersson. 50 million years of genomic stasis in endosymbiotic bacteria. *Science*, 296(5577):2376–2379, 2002.

R.L. Tatusov, E.V. Koonin, and D.J. Lipman. A genomic perspective on protein families. *Science*, 278(5338):631–637, 1997.

A. Tauch, I. Homann, S. Mormann, S. Rüberg, A. Billault, B. Bathe, S. Brand, O. Brockmann-Gretza, C. Rückert, N. Schischka, C. Wrenger, J. Hoheisel, B. Möckelf, K. Huthmacher, W. Pfefferle, A. Pühler, and J. Kalinowski. Strategy to sequence the genome of *Corynebacterium glutamicum* ATCC 13032: use of a cosmid and a bacterial artificial chromosome library. *Journal of Biotechnology*, 95(1):25–38, 2002.

H. Tettelin. The bacterial pan-genome and reverse vaccinology. *Genome Dynamics*, 6:35, 2009.

H. Tettelin, V. Masignani, M.J. Cieslewicz, C. Donati, D. Medini, N.L. Ward, S.V. Angiuoli, J. Crabtree, A.L. Jones, A.S. Durkin, R.T. Deboy, T.M. Davidsen, M. Mora, M. Scarselli, I. Margarit y Ros, J.D. Peterson, C.R. Hauser, J.P. Sundaram, W.C. Nelson, R. Madupu, L.M. Brinkac, R.J. Dodson, M.J. Rosovitz, S.A. Sullivan, S.C. Daugherty, D.H. Haft, J. Selengut, M.L. Gwinn, L. Zhou, N. Zafar, H. Khouri, D. Radune, G. Dimitrov, K. Watkins, K.J.B. O'Connor, S. Smith, T.R. Utterback, O. White, C.E. Rubens, G. Grandi, L.C. Madoff, D.L. Kasper, J.L. Telford, M.R. Wessels, R. Rappuoli, and C.M. Fraser. Genome analysis of multiple pathogenic isolates of *Streptococcus agalactiae*: implications for the microbial "pan-genome". *Proceedings of the National Academy of Sciences*, 102(39):13950–13955, 2005.

H. Tettelin, D. Riley, C. Cattuto, and D. Medini. Comparative genomics: the bacterial pan-genome. *Current Opinion in Microbiology*, 11(5):472–477, 2008.

E. Trost, S. Götker, J. Schneider, S. Schneiker-Bekel, R. Szczepanowski, A. Tilker, P. Viehoever, W. Arnold, T. Bekel, J. Blom, et al. Complete genome sequence and lifestyle of black-pigmented *Corynebacterium aurimucosum* ATCC 700975 (formerly *C. nigricans* CN-1) isolated from a vaginal swab of a woman with spontaneous abortion. *BMC Genomics*, 11(1):91, 2010. ISSN 1471-2164.

E. Trost, J. Blom, S. de Castro Soares, I-H. Huang, A. Al-Dilaimi, J. Schröder, Se. Jaenicke, Fernanda A Dorella, Flavia S Rocha, Anderson Miyoshi, Vasco Azevedo, Maria P Schneider, Artur Silva, Thereza C Camello, Priscila S Sabbadini, Cintia S Santos, Louisy S Santos, Raphael Jr Hirata, Ana L Mattos-Guaraldi, Androulla Efstratiou, Michael P Schmitt, Hung Ton-That, and Andreas Tauch. Pangenomic Study of *Corynebacterium diphtheriae* That Provides Insights into the Genomic Diversity of Pathogenic Isolates from Cases of Classical Diphtheria, Endocarditis, and Pneumonia. *Journal of Bacteriology*, 194(12), 2012.

I. Uchiyama. MBGD: microbial genome database for comparative analysis. *Nucleic Acids Research*, 31(1):58–62, 2003.

I. Uchiyama. MBGD: a platform for microbial comparative genomics based on the automated construction of orthologous groups. *Nucleic Acids Research*, 35(suppl 1):D343–D346, 2007.

I. Uchiyama, T. Higuchi, and M. Kawai. MBGD update 2010: toward a comprehensive resource for exploring microbial genome diversity. *Nucleic Acids Research*, 38(suppl 1):D361–D365, 2010.

R. Urwin and M.C.J. Maiden. Multi-locus sequence typing: a tool for global epidemiology. *Trends in microbiology*, 11 (10):479–487, 2003.

S.M. van Dongen. *Graph clustering by flow simulation*. PhD thesis, Utrecht University, 2000.

J.D. Van Embden, M.D. Cave, J.T. Crawford, J.W. Dale, K.D. Eisenach, B. Gicquel, P. Hermans, C. Martin, R. McAdam, and T.M. Shinnick. Strain identification of *Mycobacterium tuberculosis* by DNA fingerprinting: recommendations for a standardized methodology. *Journal of Clinical Microbiology*, 31(2):406–409, 1993.

S.A.F.T. van Hijum, A.L. Zomer, O.P. Kuipers, and J. Kok. Projector 2: contig mapping for efficient gap-closure of prokaryotic genome sequence assemblies. *Nucleic Acids Research*, 33(suppl 2):W560–W566, 2005.

C. Von Mering, M. Huynen, D. Jaeggi, S. Schmidt, P. Bork, and B. Snel. STRING: a database of predicted functional associations between proteins. *Nucleic Acids Research*, 31(1):258–261, 2003.

D.P. Wall, H.B. Fraser, and A.E. Hirsh. Detecting putative orthologs. *Bioinformatics*, 19(13):1710–1711, 2003.

Z. Wang, M. Gerstein, and M. Snyder. RNA-Seq: a revolutionary tool for transcriptomics. *Nature Reviews Genetics*, 10 (1):57–63, 2009.

J.D. Watson and F.H.C. Crick. Molecular structure of nucleic acids. *Nature*, 171(4356):737–738, 1953.

L.G. Wayne, D.J. Brenner, R.R. Colwell, P.A.D. Grimont, O. Kandler, M.I. Krichevsky, L.H. Moore, W.E.C. More, R.G.E Murray, E. Stackebrandt, M.P. Starr, and H.G. Trüper. Report of the Ad Hoc Committee on Reconciliation of Approaches to Bacterial Systematics. *International Journal of Systematic Bacteriology*, 37(4):463–464, 1987.

K.A. Wetterstrand. DNA Sequencing Costs: Data from the NHGRI Large-Scale Genome Sequencing Program. `www.genome.gov/sequencingcosts`, 2012. Accessed 03.09.2012.

C.R. Woese. Bacterial evolution. *Microbiological Reviews*, 51(2):221, 1987.

C.R. Woese, E. Stackebrandt, T.J. Macke, and G.E. Fox. A phylogenetic definition of the major eubacterial taxa. *Systematic and Applied Microbiology*, 6(2):143–151, 1985.

World Health Organization. Global tuberculosis report 2012. Technical Report 17, World Health Organization (WHO), 2012. URL `http://www.who.int/tb/publications/global_report/en/`.

X. Xu, H. Nagarajan, N.E. Lewis, S. Pan, Z. Cai, X. Liu, W. Chen, M. Xie, W. Wang, S. Hammond, M.R. Andersen, N. Neff, B. Passarelli, W. Koh, H.C. Fan, J. Wang, Y. Gui, K. Lee, M.J. Betenbaugh, S.R. Quake, I. Famili, B.O. Palsson, and J. Wang. The genomic sequence of the Chinese hamster ovary (CHO)-K1 cell line. *Nature Biotechnology*, 29(8):735–741, 2011.

Z. Yang. PAML 4: phylogenetic analysis by maximum likelihood. *Molecular Biology and Evolution*, 24(8):1586–1591, 2007.

H. Yukawa, C.A. Omumasaba, H. Nonaka, P. Kós, N. Okai, N. Suzuki, M. Suda, Y. Tsuge, J. Watanabe, Y. Ikeda, et al. Comparative analysis of the *Corynebacterium glutamicum* group and complete genome sequence of strain R. *Microbiology*, 153(4):1042–1058, 2007.

E.M. Zdobnov and P. Bork. Quantification of insect genome divergence. *Trends in Genetics*, 23(1):16–20, 2007.

D.R. Zerbino and E. Birney. Velvet: algorithms for de novo short read assembly using de Bruijn graphs. *Genome Research*, 18(5):821–829, 2008.

E. Zuckerkandl and L. Pauling. Molecules as documents of evolutionary history. *Journal of Theoretical Biology*, 8(2): 357–366, 1965.

Acknowledgments

First of all I wish to thank Prof. Dr. Jens Stoye and Dr. Alexander Goesmann for the opportunity to do this PhD thesis under their supervision. I really appreciate the excellent mentoring by both supervisors who provided expert advice, inspiration, and encouragement as well as - if needed - constructive criticism.
Also, I want to thank the people of the BRF, the Genome Informatics group, and the Computational Genomics group for technical support and advice, proof-reading, assistance, and last not least the amicable work atmosphere.
I would like to acknowledge the German Federal Ministry of Education and Research for funding this work (grants 0313805A "GenoMik-Plus" and 0315599B "GenoMik-Transfer"). In this context I have to thank Dr. Alexander Goesmann again for his tireless efforts to ensure the funding of me and my colleagues of the BRF.
In the course of this work several Bachelor and Master thesis have been supervised by me. I would like to thank Daniel Doppmeier and Dominik Vahrenhorst for their contributions to EDGAR, and I am much obliged to Tobias Jakobi for his major contribution to SARUMAN by realizing the CUDA implementation in his Master thesis. Furthermore I have to thank Corinna Ernst who worked on the statistical analyses in EDGAR as a student assistant.
I wish to thank all the people of the CeBiTec, especially the workgroups "Medical Microbiology and Genomics" of PD Dr. Andreas Tauch and "Microbial Genomics and Biotechnology" of Dr. Jörn Kalinowski for the pleasant collaboration, fruitful discussions, and excellent feedback that helped on the improvement of EDGAR. I also want to thank Theo Smits, Joel Pothier, and Brion Duffy of the Agroscope ACW in Wädenswil, Switzerland, for the productive collaboration.
Special thanks go to my colleagues Dr. Burkhard Linke and Sebastian Jaenicke who share an office with me for more than six years now. Both supported the development of EDGAR as well as SARUMAN with many hints and advices and in countless discussions, and both pushed me whenever I started to procrastinate during the writing of this thesis. Thank you guys!
I would like to express my gratitude to all my friends who helped me on the way, especially the PAX guys who accompanied me from the beginning of my studies in Bielefeld up to now: No one could wish for better friends!
I am eternally grateful to my parents for the love, trust, and support I could always count on throughout my studies.
Finally, I wish to thank my girlfriend Karin for her amazing and invaluable support.

Bielefeld, den 21. Januar 2013

Jochen Blom

ERKLÄRUNG

Ich, Jochen Blom, erkläre hiermit, dass ich die Dissertation selbständig erarbeitet und keine anderen als die in der Dissertation angegebenen Hilfsmittel benutzt habe.

Bielefeld, den 21. Januar 2013

Jochen Blom