Bioinformatic methods for eukaryotic RNA-Seq-based promoter identification

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

von

Tobias Jakobi

Juni 2014

Supervisors: Prof. Dr. Alfred Pühler
Prof. Dr. Alexander Goesmann

Tobias Jakobi
Am Hange 22
58119 Hagen
tjakobi@cebitec.uni-bielefeld.de

Printed on non-aging paper according to DIN-ISO 9706.

Contents

CHAPTER 1

Introduction

1.1 The Chinese hamster

The Chinese hamster (*Cricetulus griseus*) is a member of the genus *Cricetulus*, consisting of seven species, whom tend to differ from common hamsters in a more ratlike appearance. Native to the deserts in northern China and Mongolia, members of this species typically reach sizes up to 12 cm, weights of up to 45 grams and their span of live covers about two to three years. While female animals found use as pets, male hamsters were mainly used as laboratory animals until hamsters where gradually replaced by other rodents like mouse (*Mus musculus*) or rat (*Rattus norvegicus*), which exhibit easier breeding and keeping properties.

In the early 1920s Chinese hamsters were used for typing *Streptococcus pneumoniae* [Jayapal and Wlaschin, 2007]. Shortly after typing studies it came to the researchers attention that Chinese hamsters were well suited as a transmission vector for visceral leishmaniasis, also known as "black fever" or "Dumdum fever". The disease is caused by protozoan parasites, infests several internal organs like liver, spleen, or bone marrow and accounts for 500,000 infections per year [Desjeux, 2001], while an infection is nearly always lethal when no treatment is administered . Thus, *Cricetulus griseus* became an important part of epidemiology research.

Figure 1.1: A Chinese hamster (*Cricetulus griseus*).

Since 1948 the Chinese hamster found use in United States research laboratories for breeding purposes. It was also in the mid-20th century, when Yerganian [1972, 1985] discovered a tendency for hereditary diseases during breeding experiments. During this time, focus shifted from *Cricetulus griseus* as a laboratory animal to cell lines derived from different tissues of the animal. Indeed, when shifting from organism level down to cell level, different properties of the research subject become more important. *Mus musculus* has a chromosome number of 20, *Rattus norvegicus* possesses 21 chromosomes and *Homo sapiens* 24 chromosomes. In contrast, the Chinese hamster only has 11 chromosomes (2n=22). This very low complement of chromosomes makes Chinese hamster cells an ideal model for tissue culture or radiation studies [Tjio and Puck, 1958]. The history of Chinese hamster cell lines continues in the late 1950s when Puck et al. received a female Chinese hamster from the Boston Cancer Research foundation and extracted ovary cells. These cells formed the first Chinese Hamster Ovary (CHO) cell line and blazed a trail for many derived cell lines with immense impact on modern biotechnology.

1.2 CHO cell lines

Although *Cricetulus griseus* was superseded by mouse and rat in its function as laboratory animal, derived cell lines still play a key role in modern biotechnology, emphasised by a quotation from Puck [1985], who described CHO cells as "the mammalian equivalent of *Escherichia coli*". A time line showing the development from the ancestral Puck cell line to today's production and research cell lines is shown in Figure 1.2. The genetic material analysed in this work was extracted from a culture of CHO-K1 CCL-61 cells. The genealogical tree of CHO lines however, can only show a subset of available cultures, since most laboratories working with CHO cells developed lines fine-tuned to specific requirements and especially companies from within the biotechnology sector typically do not uncover enhancements or modifications they made due to intellectual property (IP) issues.

Isolation, characterisation, and cultivation of mammalian cells always has been a challenge, partly owed to the fact that mammalian cells are diploid in general. Diploidity can be vital for cells, since defects in one chromosome can be corrected by the second member of the pair. However, when introducing new genetic constructs, such like genes to be expressed, this fail safe mechanism is known to cause problems. CHO cells in contrast are normally hemizygous and therefore prone to manipulation of the genetic material. Chasin and Urlaub [1975] as well as Simon and Taylor [1982] were able to show that this hemizygosity is mainly induced by gene inactivation. Due to this unique property for a mammalian cell line, CHO cells found their way in medical studies, cell biology studies, and toxicology studies.

The diversification of CHO cell lines started with mutagenesis experiments conducted with ethyl methanesulfonate (EMS) or gamma ray bombardment. Those treatments induce genetic mutations by chance and may produce cells with new functions or loss-of-function mutants due to altered DNA. During one of these experiments a loss-of-function mutant with a defect in the Dihydrofolate reductase (DHFR) gene was discovered. Since cells of this mutant need to be supplied with glycine, hypoxanthine, and thymidine their auxotrophy can be used as a selection marker and subsequently a new expression system was developed originating from this mutant.

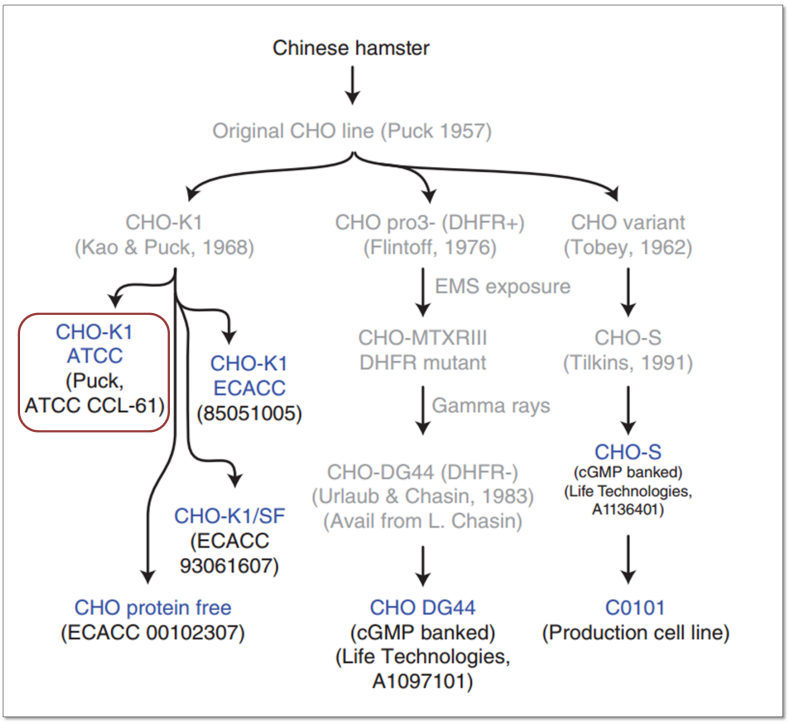


Figure 1.2: Overview of the development of CHO cell lines starting from the original line derived by Puck in 1957. Graphics from Lewis et al. [2013]. CHO-K1 ATCC CCL-61 (marked red) is the cell line used as source for genetic material analysed within this manuscript.

Expression systems within the prokaryotic world have a long history and are well established. However, in the mammalian environment proteins normally require post translational modifications which cannot be carried out by prokaryotic host systems. These modifications, are crucial to ensure the biological activity as well as human compatibility in case of pharmaceutical products. Another important factor for host systems is their scalability. Several types of cell lines are only able to grow as monolayers in petri dishes which may be sufficient for small scale experiments but becomes a major drawback in industrial production. The CHO cell line is able to grow in suspension and therefore even in large bioreactors accommodating

thousands of liters of cell culture suspension. Product safety is another crucial point in the process of choosing the right host system, especially within the sector of human pharmaceuticals. Contaminations of the product have to be avoided by all means. During the last two decades purification techniques have evolved, resulting in no more than picograms of possible CHO DNA left in the product [Wiebe et al., 1989]. Even more important than removal of residual host DNA is the lack of any human pathogens, specifically viruses. Indeed, it could be shown that only a fraction of human pathogenic viruses are able to replicate within CHO cells [Wurm, 2005].

1.2.1 Applications

Name	Type	Exemplary indications	Sales ($ billion)
Humira	mAb	Rheumatoid arthritis	9,266
Enbrel	Protein	Rheumatoid arthritis	7,967
Rituxan	mAb	Rheumatoid arthritis	7,049
Herceptin	mAb	Breast cancer, gastric cancer	6,188
Avastin	mAb	Colorectal / cell lung cancer	6,059
Avonex	Protein	Multiple sclerosis	2,913
Rebif	Protein	Multiple sclerosis	2,408

Table 1.1: Selection of CHO based pharmaceuticals from the top-ten-selling biologic drugs 2012 [Huggett, 2013]. Two types of drugs are shown: proteins and monoclonal antibodies (mAb). Different kinds of arthritis, cancer and multiple sclerosis are subject to treatment with CHO derived products. Total sales in 2012 sum up to nearly 42 billion dollars.

A brief gaze on the list of top ten selling biologic drugs stresses the importance of CHO as production host for monoclonal antibodies and proteins, since 7 out of 10 drugs (shown in Table 1.1) are produced by CHO cell lines, accumulating to a total sales value of 41.85 billion dollar in 2012 [Huggett, 2013]. Two categories of drugs produced by CHO cells can be distinguished: those that are produced in protein form and monoclonal antibodies. Monoclonal antibodies have a specific binding affinity e.g. for receptors of cancer cells. Antibodies are able to mark targeted cells in a way that a patients immune system can recognize the signal and can attack the malfunctioning cell. Several antibody-based cancer therapies work this way; therapeutic proteins may inhibit the expression of genes responsible for specific diseases.

1.2.2 Recent development

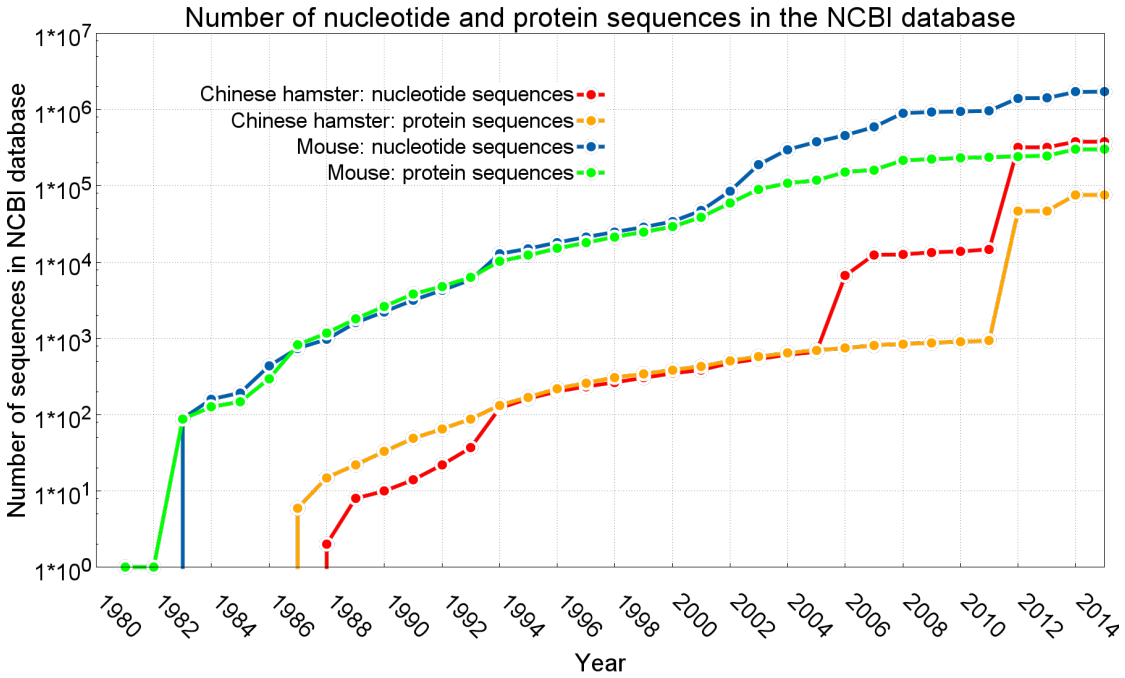


Figure 1.3: Total number of protein and nucleotide sequences in the NCBI (National Center for Biotechnology Information) database assigned to *Cricetulus griseus* and CHO cells or *Mus musculus* respectively. The y-axis is log scaled and shows the total number of sequences in the NCBI database for the given organism. Corresponding years are shown on the x-axis. The number of mouse related sequences per year exceeds the number of Chinese hamster assigned sequences by one or more orders of magnitude. This is in sharp contrast to the aforementioned industrial importance of CHO cell lines.

Given the importance of the CHO cell line, which was highlighted in the previous section, it would be safe to assume that there were numerous efforts to generate large amounts of sequence data. However, most of the sequence data for *Cricetulus griseus* or Chinese hamster is less than two years old. First sequences for *Mus musculus* became available in 1982, for the Chinese hamster first data was added in 1987. Despite the short delay and the slow growth rate of sequence databases in the 1980s, the amount of newly generated sequence data originating from mouse exceeds novel Chinese hamster related data by far. In 1996, ten years after first data was submitted over 18,000 nucleotide sequences from mouse stand vis-à-vis to roughly 300 sequences of *Cricetulus griseus*. An overview of 30 years of sequence database development for mouse and Chinese hamster is depicted in Figure 1.3. Additions of nucleotide sequences for the hamster occur in a linear way but also show at least two significant impulses: the first from 2005 to 2007, the second in 2012. Screening of the NCBI database for publications that added large amounts of sequences resulted in two patents, filed in 2005 [Melville et al., 2005] and 2006

[Melville et al., 2006]. Both of the patents describe thousands of oligonucleotide sequences which can be used to monitor gene expression via microarray technologies. However, the probably most important year by means of hamster sequence data is 2011. For the first time a draft genome of the CHO-K1 cell line with a size of 2,4 Gb was made available to the scientific community by Xu et al. [2011], resulting in the addition of more than 200,000 genomic and transcriptomic sequences. Later in the same year, nearly 70,000 additional transcriptomic sequences could be contributed by Becker et al. [2011], thus paving the way for further studies of the CHO cell line on genetic level. After negligible updates in 2012, three publications added another 60,000 sequences to the pool; 25,000 of them being *in silico* predicted piRNAs originating from a study by Gerstl et al. [2013]. Two years after Xu et al. [2011] presented a draft version of the CHO-K1 genome, the source organism's genome was sequenced and assembled using chromosome separation techniques by Brinkrolf et al. [2013]. The *Cricetulus griseus* draft genome has a size of 2,33 Gb and proved that, despite of decades in cell culture the CHO-K1 line is not missing major parts of the Chinese hamster genome. Latest addition to the hamster data pool is a second assembly of the *Cricetulus griseus* wild type by Lewis et al. [2013]. The study also highlights genetic differences between the wild type and several different CHO cell lines like DG-54 or CHO-S.

Within the last ten years the amount of available sequence data for the Chinese hamster has grown by three orders of magnitude, one alone during the last two years. A broad basis of sequence data, may it be in form of protein or nucleotide sequences is the foundation for nearly all further research topics. Cross comparisons of cell lines are only possible with at least one reference genome and the discovery of single nucleotide polymorphisms (SNPs) also requires a genetic reference as data source. Biotechnological engineering of CHO cells requires detailed knowledge of gene structures and the ordering of genes on the chromosomes. However, when comparing the amount of available sequence data with mouse, rat, or human several challenges still have to be overcome for the Chinese hamster.

1.3 About this work

For many eukaryotic genomes, including the Chinese hamster no sequenced genomes were available at the start of this project. Due to the industrial relevance of the CHO cell lines stressed in the previous Section, these cell lines are important targets for genetic engineering. Modifications especially include promoter regions which are key factors for the yield of arbitrary protein coding genes. In order to increase the production capacity of these eukaryotic hosts, promoter constructs have to be screened and assessed for their potential use as production promoters. In a first step a bioinformatics pipeline able to extend incomplete CHO transcripts into promoter carrying regions was developed to acquire knowledge of the yet un-

charted CHO promoter regions. However, as sequenced genomes became available a second, RNA sequencing-based approach was employed to locate transcription start sites in reference genomes by read mapping techniques. The located transcription start sites act as indicator for nearby promoter regions and subsequently allow for detailed analyses of the important regulatory regions for specific elements possibly enhancing the expression profile of the corresponding promoters.

1.4 Scientific work

Throughout the last years, several scientific contributions were made, starting with parts of the Bachelor thesis [Henckel et al., 2009], followed by the publication of SARUMAN, a short read mapping software [Blom et al., 2011] as Master thesis. A transition to eukaryotic genome assembly and related bioinformatic problems began simultaneously with the start as Ph.D. student [Hackl et al., 2011; Becker et al., 2011; Hackl et al., 2012]. Since then, the aforementioned Chinese hamster in general and the CHO-K1 cell line in special became the foundation for this work. During an internship at Illumina Inc. based in Cambridge, UK, several contributions to BEETL, a BWT-based sequence compression library were made [Cox et al., 2012a,b].

Publications as first author

- T. Jakobi, K. Brinkrolf, A. Tauch, T. Noll, J. Stoye, A. Pühler, and A. Goesmann, Discovery of transcription start sites in the Chinese hamster genome by next-generation RNA sequencing, *J. Biotechnol.*, submitted
- 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*, vol. 27, no. 10, pp. 1351-8, May 2011. (* contributed equally)

Publications as co author

- A. J. Cox, T. Jakobi, G. Rosone, and O. B. Schulz-Trieglaff, Comparing DNA Sequence Collections by Direct Comparison of Compressed Text Indexes, Proceedings of the 12th International Workshop on Algorithms in Bioinformatics, Lecture Notes in Computer Science Volume 7534, pp. 214-224, Sep. 2012.
- A. J. Cox, M. J. Bauer, T. Jakobi, and G. Rosone, Large-scale compression of genomic sequence databases with the Burrows-Wheeler transform, *Bioinformatics*, vol. 28, no. 11, pp. 1415-1419, May 2012.
- M. Hackl, V. Jadhav, T. Jakobi, O. Rupp, K. Brinkrolf, A. Goesmann, A. Pühler, T. Noll, N. Borth, and J. Grillari, Computational identification of

microRNA gene loci and precursor microRNA sequences in CHO cell lines., *J. Biotechnol.*, vol. 158, no. 3, pp. 151-5, Apr. 2012.

- J. Becker, M. Hackl, O. Rupp, T. Jakobi, J. Schneider, R. Szczepanowski, T. Bekel, N. Borth, A. Goesmann, J. Grillari, C. Kaltschmidt, T. Noll, A. Pühler, A. Tauch, and K. Brinkrolf, Unraveling the Chinese hamster ovary cell line transcriptome by next-generation sequencing., *J. Biotechnol.*, vol. 156, no. 3, pp. 227-35, Dec. 2011.

- M. Hackl, T. Jakobi, J. Blom, D. Doppmeier, K. Brinkrolf, R. Szczepanowski, S. H. Bernhart, C. H. Z. 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., *J. Biotechnol.*, vol. 153, no. 1-2, pp. 62-75, Apr. 2011.

- J. Blom, R. Hilker, T. Jakobi, K. Pfeifer, C. Rückert, J. Kalinowski, and A. Goesmann, RNA-Sequenzierung in der funktionellen Genomforschung: Methoden zur Analyse von RNASeq-Datensätzen am Bielefelder Centrum für Biotechnologie, *GENOMXPRESS*, 3.11 , p. 4-6, Mar. 2011

- K. Henckel, K. J. Runte, T. Bekel, M. Dondrup, T. Jakobi, H. Küster, and A. Goesmann, TRUNCATULIX–a data warehouse for the legume community., *BMC Plant Biol.*, vol. 9, p. 19, Jan. 2009.

Conference posters

- T. Jakobi, K. Brinkrolf, A. Wippermann, A. Pühler, and T. Noll, Exploring the promoter landscape of the Chinese hamster by next -generation RNA sequencing (Poster abstract), *Cell Culture Engineering XIV*, Quebec City, Canada, May 2014.

- T. Jakobi, A. Goesmann, A. Pühler, and J. Stoye, Seed based, reference-less and targeted assembly on eukaryotic scale (Poster abstract), *Workshop on Algorithms in Bioinformatics 2012*, Ljubljana, Slovenia, Sep. 2012.

- T. Jakobi, J.Blom, D. Doppmeier, S. Jaenicke, J. Kalinowski, J. Stoye, and A. Goesmann, Einsatz von Grafikhardware in der Genomforschung (Poster abstract), *DECHEMA-Jahrestagung der Biotechnologen*, Aachen, Germany, Aug. 2010.

- T. Jakobi, J.Blom, D. Doppmeier, S. Jaenicke, J. Kalinowski, J. Stoye, and A. Goesmann, SARUMAN - Using GPU programming for short read mapping (Poster abstract), *International Conference on Intelligent Systems for Molecular Biology*, Boston, USA, Jul. 2010.

Parts of this work (Chapter 5) were submitted to the *Journal of Biotechnology* as:

T. Jakobi, K. Brinkrolf, A. Tauch, T. Noll, J. Stoye, A. Pühler, and A. Goesmann, Discovery of transcription start sites in the Chinese hamster genome by next-generation RNA sequencing.

1.5 Manuscript structure

In Chapter 3, an overview on state of the art techniques commonly used in genome assembly and promoter analysis is given and limitations of those approaches are pointed out. Background knowledge about DNA sequencing, genome assembly, and promoter analysis is mediated in Chapter 2. Within Chapter 4 the new SATYR pipeline for targeted assembly is presented. The analysis pipeline for CHO promoter sequences is discussed in chapter 5 and described together with the experimental setup. Within Chapter 6 the results are summarised, the accomplishments are reviewed, and an outlook into possible further developments is given.

CHAPTER 2

Background

2.1 DNA sequencing

Deoxyribonucleic acid, or DNA for short, is a molecule which has the ability to store information on different levels. Every living organism, with the exception of several viruses, uses DNA as main storage for genetic information. In general, DNA has a double stranded appearance, consisting of two anti-parallel strands. The direction of each strand can be derived from the position of the 3rd and 5th carbon atom within the sugar molecule and is therefore referred to as 3' and 5'. The DNA can be imagined as a ladder with two phosphate-deoxyribose stiles and treads made of two of the four nucleobases: adenine (A), thymine (T), guanine (G), and cytosine (C) [Levene, 1921]. The two valid combinations of nucleotides are adenine + thymine and cytosine + guanine. The base pair combinations are implicated by the number of hydrogen bonds the nucleotides can form: two for AT and three for CG. Classifications into pyrimidines and purines are based on their chemical properties. This basic structure of the DNA molecule was discovered by Watson and Crick [1953a,b] and Franklin and Gosling [1953] and eventually resulted in a Nobel prize for Watson and Crick.

Several years later, the term "central dogma of molecular biology" was coined by Crick [1958, 1970]. The term describes the flow of information between three different kinds of biopolymers, namely from DNA to RNA (ribonucleic acid). Information transfer from protein level back to DNA or RNA level is not allowed, however, DNA and RNA may interchange information in both directions.

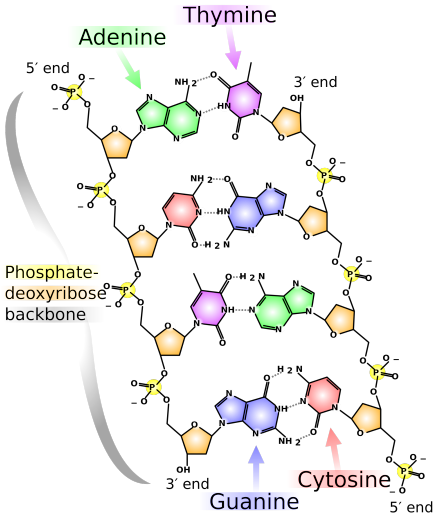


Figure 2.1: Blueprint of the DNA molecule. The DNA consists of two phosphate-deoxyribose backbones and pairs of the four nucleobases adenine (A), thymine (T), guanine (G), and cytosine (C). Image by Madeleine Price Ball.

The creation of RNA from a DNA template is commonly referred to as transcription. And indeed, a transcription takes place during the process since thymine is replaced by uracil while the sequence of nucleotides is copied from the template DNA strand onto a newly synthesised RNA molecule. At this point directions of strands play a key role during transcription. While an enzyme called RNA-polymerase slides over the template strand in 3' to 5' direction, a new RNA strand is synthesised from 5' to 3' [Solomon, 2005]. Several kinds of RNA classes are known in organisms: messenger RNA (mRNA) which codes for proteins or non-coding RNA families like micro RNA (miRNA), small interfering RNA (siRNA), transfer RNAs (tRNA), or ribosomal RNAs (rRNA) [Krebs et al., 2012].

In a second step the newly created messenger RNA (mRNA) may be converted into a protein during the translation process. Again, the mRNA strand is processed from 5' to 3', while the chain of amino acids is generated. Proteins subsequently are folded in such a way that they can interact with different compounds in form of biologically active enzymes. Enzymes are critical for every organism as they are integrated in most chemical reaction in a cell [Smith, 1997].

While transcription and translation are crucial operations for each organism, the replication process is used during each cell division when the genetic material has to be copied, so that each cell receives exactly the same DNA sequence as the source cell [Alberts, 2002; Albà, 2001]. This enzymatic reaction of the replication process is also important during genome sequencing. During genome sequencing, the order of base pairs of given sample DNA is analysed, a process which is typically connected with the synthesis of a DNA strand (see Section 2.1.1 and following).

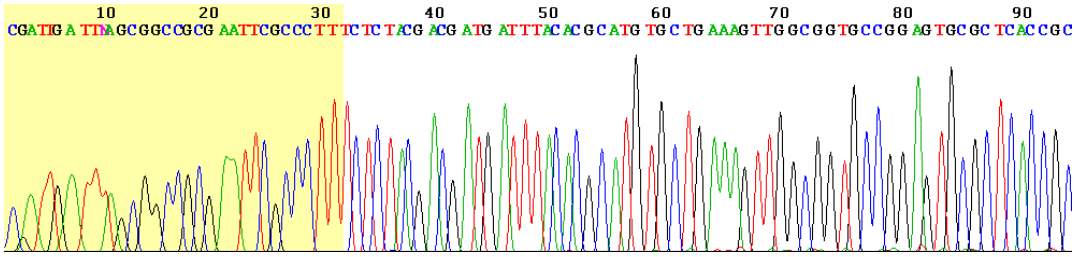


Figure 2.2: Sample output of a Sanger sequencing read (bases 1-95 shown). The sequence can be seen in coloured letters directly below the base index. Bases are shown in different colours: A (green), C (blue), G (black), and T (red). Readout data from the laser excitation is shown in the curve. A yellow area highlights the start of the read which is normally removed during post processing due to low base quality.

End product of this sequencing process is the sequence of bases encoding for all properties of the source organism, which may vary in length from a few thousand bases (for small viruses [Sanger et al., 1977b]) up to billions of bases (for complex eukaryotic life forms [Pellicer et al., 2010]).

2.1.1 Sanger sequencing

The era of genome sequencing started in 1977 when the genome of the bacteriophage phi X 174 was deciphered. The virus has a small genome of roughly 5,000 bp and therefore was well suited for establishing the new technique developed by Sanger et al. [1977a]. This first method later became known as "Sanger sequencing" after his initial developer. Prerequisite is a sufficient amount of template DNA which can be achieved by using clone libraries or by PCR (polymerase chain reaction) amplification. Single stranded template DNA is combined with a primer needed to start the replication process. Four different reactions are carried out, one for each nucleotide. Each reaction is provided with DNA polymerase enzyme, unlabelled deoxynucleotides (dNTP: dATP, dCTP, dGTP, dTTP) and one fluorescently or radioactively labelled dideoxynucleotide (ddNTP: ddATP, ddCTP, ddGTP, ddTTP). Once started, the sequencing reaction stops when a ddNTP is encountered, causing the polymerase to stop strand elongation. After a denaturing phase the process is repeated several times to ensure that for each position of the template strand one prematurely terminated new DNA molecule has been created, therefore the method is also called chain termination sequencing. Newly synthesised strands are separated based on their length by gel electrophoresis and scanned by a laser (in case of fluorescent labelled ddNTPs) to extract the sequence of bases for the template (Figure 2.2). Although Sanger sequencing was developed more than three decades ago it still is present in today's research. Sequencing automatons

Phred score	Probability of incorrect base call	Base call accuracy
10	1 in 10	90 %
20	1 in 100	99 %
30	1 in 1000	99.9 %
40	1 in 10000	99.99 %
50	1 in 100000	99.999 %

Table 2.1: Summary of phred quality scores. These quality scores, introduced with the phred program are used to measure the confidence of a base produced by many sequencing machines.

like the ABI 3730 xl[1] can sequence up to 384 samples in parallel, produce reads of more than 1000 bases and have a maximal throughput of 2 million bases per day. Today these machines are mainly used for gap closure in sequencing projects as well as for short areas of interest which need to be sequenced. However, until next generation sequencing hit the market even large genomes like the human genome were analysed using classical Sanger sequencing [Lander et al., 2001].

2.1.2 Assessment of sequencing quality

The quality of single bases is usually denoted by a so called phred score, originating from the equally named base calling tool [Ewing et al., 1998]. The phred tool was initially used to convert the fluorescence trace files produced by Sanger sequencing machines (Figure 2.2) back into a series of bases, hence the name "base caller". Phred introduced quality scores to determine the quality or confidence of a base. Whereas in times of non-automated Sanger sequencing trace files could be verified semi-automatically, large sequencing projects, such as the Human Genome Project demanded for fully automated base calling and quality assessment due to the sheer amount of sequencing data. Phred came to extensive use during the Human Genome Project and remained de-facto standard for sequencing quality measurement since then. Higher scores indicate a high quality, whereas lower scores suggest questionable confidence of a base (Table 2.1). These score values become important during the assembly process (see Section 2.2.1), as the software can recognise putative sequencing errors and consider them during the assembly.

[1]Life Technologies, Carlsbad, California, USA, http://www.appliedbiosystems.com

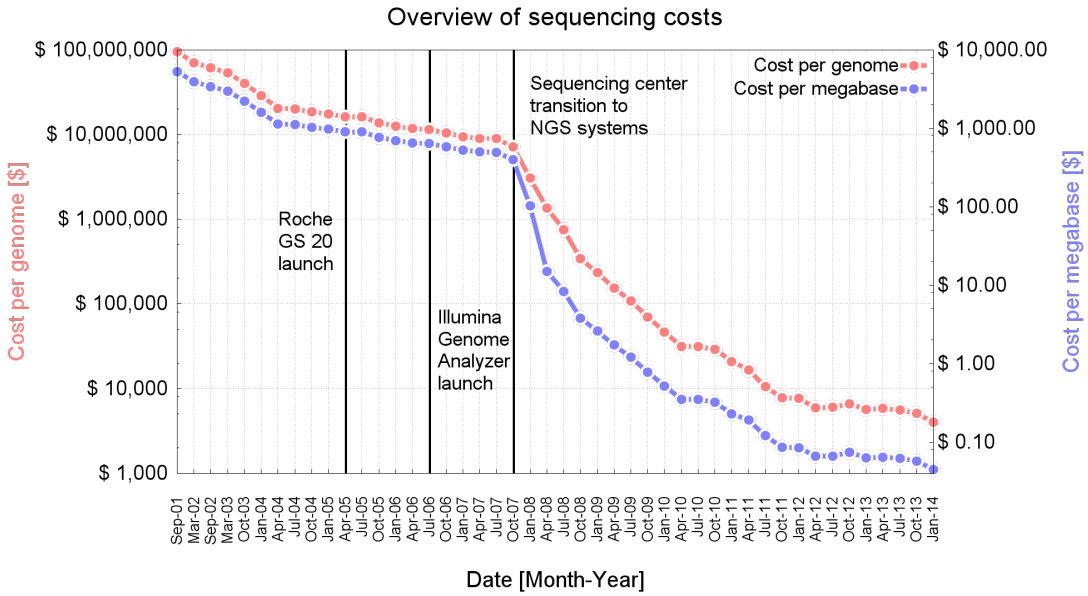


Figure 2.3: A brief history of sequencing costs, divided into "Costs per genome" on the first y-axis (red) and "costs per megabase" on the second y-axis (blue). Corresponding time points for are shown on the x-axis. Both y-axis are log-scaled. After the introduction of the first two commercial next generation sequencing systems in 2005 and 2006 the transition of sequencing centres from Sanger sequencing to NGS techniques took another year. Starting from October 2007 prices per megabase and genome dropped dramatically and are still continuously decreasing. Data taken from [Wetterstrand, 2014].

2.1.3 Next generation sequencing

The human genome project started in 1990 and a first draft sequence was published in 2001 [Lander et al., 2001]. While it may be experimentally possible to use Sanger sequencing for large genomes, the investment of over $3,000,000,000 makes it impossible to acquire a large variety of genome sequences, especially for species with large genomes, like mammals and plants. Starting in the early 2000s several new technologies emerged from research laboratories. The following pages will cover systems by Roche and Illumina in more detail, since data from those systems was used throughout the work, other systems and new developments will be introduced in a more briefly manner. It is due to the next generation sequencing systems that sequencing costs per megabase and per genome significantly decreased during the last years (Figure 2.3), forming the 1000 $ human genome catchphrase which was finally achieved with the introduction of Illumina's HiSeq X Ten system in early 2014 [Illumina Inc.].

2.1.3.1 Roche 454

First efforts in the search for new sequencing techniques led to the development of pyrosequencing by Ronaghi et al. [1996, 1998]. Licensed by 454 Life Sciences[2] the first device called GS20 was released in 2005. Roche Diagnostics[3] bought 454 in 2007 but continued to operate 454 as a independent business unit. Pyrosequencing differs fundamentally from Sanger sequencing in the way in which bases generate a signal, which is translated into an A, C, G, or T. Additionally the time consuming step of cloning is circumvented, since the template DNA can be sequenced directly. However, a fragmentation step is necessary to generate smaller units of DNA by nebulization. Those fragments are subsequently connected to capture beads by ligation. During the next step these template DNA strands are amplified by an emulsion PCR [Margulies et al., 2005] which works within a drop of water-oil-emulsion that contains adapters, nucleotides, primers, and enzymes for the reaction, thus creating beads occupied by thousands of single stranded DNA templates. The Sequencing reaction itself takes place on a so called PicoTiterPlate which carries millions of wells with a diameter of 29 μm, thus perfectly fitting the $\sim$ 20 μm diameter of a bead with connected DNA templates. When ideally every well is occupied by one bead a second enzyme mixture containing Luciferase and DNA-polymerase is added to the plate and therefore each well. The actual sequencing step is performed in cycles, each exclusively adds A, C, G, or T to the sequence and is followed by a washing step. The method, also called "sequencing by synthesis", is based on the fact that during each nucleotide incorporation a weak light signal is emitted by the Luciferase enzyme, which is detected by a very sensitive imaging device. The complete PicoTiterPlate is monitored, resulting in millions of signals from each well. This parallel step is another difference to Sanger sequencing where the most advanced machines can only sequence up to 384 samples in parallel. Initial the GS20 sequencer in 2005 was able to generate reads of 100 bp length, while achieving a throughput of 20 Mb. When compared to an ABI 3730 xl with 2 Mb per day 454 achieved a 10-fold higher throughput already in the first expansion stage. The GS20 was superseded by a new sequencer, the 454 GS FLX. While still based on the same principle, the throughput was increased to 700 Mb per day distributed through 1 million 700 nt reads. Later the "Titanium" chemistry increased the maximal read length to 1,000 nt. Main advantage of the 454 system are long reads combined with a relatively fast sequencing process. But due to the design of the method there are two drawbacks. On the one hand, reading out consecutive identical bases causes problems in base calling, since extracting the exact amount of bases from the cumulative signal is only precise up to 8 nt. Above this limit it should be assumed that the actual number of consecutive bases is underestimated or sometimes overestimated. On the other hand many organisms do not have an equal rate of G+C and A+T pairs throughout the genome, resulting in a higher "G+C ratio". During the last years it has come to attention that areas

[2] 454 Life Sciences, Branford, Connecticut, USA, http://www.my454.com
[3] Roche Diagnostics Corporation, Indianapolis, Indiana, USA, http://www.roche.com

with significant higher G+C ratios (≥ 70 %) introduce problems in the sequencing process. As remarked in Section 2.1 G+C form three hydrogen bonds which may cause very robust secondary structures induced by folding of the DNA molecule. Those structures cannot be read by the polymerase and yield incorrect readings or no readings at all. A study by Schwientek et al. [2011] analysed an additive supplied by 454 to overcome the GC-ratio induced difficulties and identified the compound as trehalose, which reduces the number of hydrogen-bonds forming. In spring 2010 Roche launched a second sequencer, called GS Junior, which is a size reduced benchtop sequencing system based on the same chemistry as the GS FLX system. The GS Junior was an attempt to overcome the pricing pressure induced by Illumina's different system design, but eventually was not able to stop the loss of market share to Illumina.

2.1.3.2 Illumina

In 1998 Solexa was founded as a startup company that emerged from Cambridge University. The company developed a novel sequencing technology based on fluorescently labelled, reversible terminated bases, and a technique used to generate clusters of DNA on a solid surface [Mayer et al., 2007, 1998]. In 2006, shortly after Roche's 454 system hit the market Solexa launched their first sequencing system, called "Genome Analyzer". Solexa was subsequently bought by Illumina Inc.[4] in 2006, hence the system is now known as Illumina sequencing. In principle a sequencing-by-synthesis approach, the whole process can be divided into three consecutive phases. DNA libraries are prepared in a series of steps, starting with fragmentation of the source DNA, in order to obtain sequences short enough to be sequenced. Afterwards adapters are ligated to the fragments, which are subsequently selected by size. During cluster generation the selected fragments are bound to fitting adapters located on a flow cell. Once the sequences are fixed on the flow cell an amplification process is repeated several times to generate a dense forest of clonal sequences in near proximity. Reverse complementary strands are washed away after each cycle to ensure strand specificity of sequencing. When enough copies of each sequence are synthesised primers for the actual sequencing reaction are hybridised to the fragments. In contrast to the 454 system, which uses Luciferase to generate a chemiluminescent signal for all bases in different cycles, Illumina uses four differently labelled terminator dyes for strand extension during the sequencing process. In each cycle the suitable base is incorporated, thus also adding specificity because the different bases compete against each other during strand synthesis. Once a base has been attached the process stalls owed to the termination properties of the modified bases. Synthesis is followed by excitation with a laser to trigger a fluorescence reaction which afterwards is measured by a highly sensitive imaging device. This step differs from the 454 imaging approach, since Illumina records four different bases at the same time while 454 only measures one

[4]Illumina, Inc., San Diego, California, USA, http://www.illumina.com

signal release by the Luciferase. Image capturing concludes with a washing step which also removes the terminator properties from the last base inserted and allows for further synthesis. During image analysis the correct base for each cluster is extracted from the colour detected at the cluster's location. The sequencing process is finished after a specified number of cycles has been reached and results in millions of reads with a uniform length. Read length however, is one of the weak spots in Illumina's sequencing system - the last stage of expansion resulted in a maximal read length of 300 nt per read and allows two reads to be combined to one 600 nt read[5]. Like Roche, Illumina offers several different sequencing systems from entry level (MiSeq) to genome sequencing centre level (HiSeq 2500); only the MiSeq benchtop sequencer is able to reach 300 nt per read, all other platforms are limited to 150 nt at maximum. Although read length cannot compete with the 454 platform - at least on very high throughput machines - the number of sequenced bases is unmatched in sequencing business. The Genome Analyzer, introduced in 2006, on the one hand was able to produce reads of only 30 nt, but on the other hand reached a throughput of up to 1 Gb per run - 50 × more than Roche's GS 20. Output was increased with each chemistry update and by introducing new systems, the Genome Analyzer IIx started with 10 Gb and 35 nt reads and today is able to produce up to 95 Gb with 2 × 150 nt reads. The HiSeq product line is capable of a maximum throughput of 600 Gb in one run over 11 days, thus resulting in 3 billion reads of 100 nt. Even though the run length of up to 11 days is much longer than one day required for a 454 setup 90 % of today's sequencing experiments are carried out on Illumina machines. This is primarily due to the high cost per run on a GS FLX+ Titanium system but also founded on the massive amount of sequence information generated by a single Illumina run.

2.1.3.3 SoLiD

The SoLiD (Sequencing by Oligonucleotide Ligation and Detection) system by Applied Biosystems was the last of the three major next generation sequencing approaches and entered commercial service in 2008. Applied Biosystems, known for their gold standard Sanger sequencing machines (see Section 2.1.1), put a strong emphasis on high accuracy, therefore the system today is typically used for detecting variations in resequencing, targeted resequencing, and transcriptome sequencing. The technology was initially bought from Agencourt Bioscience in 2006 and developed to a market ready state during the next two years [McKernan et al., 2009]. As for the competitors approaches the first step is fragmentation of the DNA source material, resulting in shorter sequences ready for sequencing preparation. Preliminarily to emulsion PCR two adapters, P1 and P2, are ligated to the fragments which are necessary for binding to the amplification beads during the next step. Emulsion PCR is very similar to the protocol in 454 systems, a sequence is attached to a bead through a covalent binding to the P1 adapter and

[5]http://www.illumina.com/products/miseq-reagent-kit-v3.ilmn

an water-oil-emulsion with enzymes and oligonucleotides is added. Afterwards a selection for enriched beads is performed to minimize the number of empty beads during sequencing and therefore maximize the efficiency. Once the beads are bound to a flowcell, the sequencing by ligation process - in contrast to the sequencing by synthesis approach of Illumina and Roche - can be started. A generic primer is bound to the P1 adapter of the sequences to start the process, followed by the first octamer for ligation. The octamers start with a nucleotide dimer at 3' , followed by four degenerated (N) bases and two degenerated bases carrying one of four fluorescent dyes, thus resulting in 16 octamers. After ligation the colour of the inserted dye is registered and the next octamer is ligated, which is typically repeated seven times and generates 35 nt reads. However, the whole process is repeated again five more times with modified primers shifting the start position by one base during each cycle. As dinucleotides are used during complementation each nucleotide of the template is covered by two shifted dinucleotides in different cycles. Since there are 16 different dinucleotides but only four available colours each colour represents four different dinucleotides, which Applied Biosystems calls 2-base encoding. The special encoding has the unique property, that it is able to distinguish between simple sequencing errors, in which case only one of the two covering dinucleotides differ from the reference, and real SNPs (Single Nucleotide polymorphisms) in which case both dinucleotides differ in their colourcode from the reference [Smith et al., 2008]. Another property owed to the 2-base encoding is the error profile of the SoLiD system, which is noticeable due to the very low number of insertion or deletion errors during sequencing and for the high accuracy of up to 99.99 % in general. The queue of colour codes (referred to as "colour space") is decoded into normal bases during post processing for use in nucleotide based tools such as many mapping tools. Additionally several genomics tools are available which can directly work on colour space data and benefit from the additional data layer. The 5500xl System from Applied Biosystems is the latest SoLiD machine and was launched in 2010. With current chemistry (V4) a throughout of up to 20 Gb per day can be achieved using read lengths from 35 to 75 nt.

System / Manufacturer	Time	Million reads/run	Read length	Yield
Applied Biosystems				
3730xl	2h	0.000096	$\sim$ 650 nt	0.0006 Gb
SOLiD v4	12d	$>$ 840	50 + 35 nt	71.4 Gb
SOLiD 5500	8d	$>$ 700	75 + 35 nt	77 Gb
SOLiD 5500xl	8d	$>$ 1,410	75 + 35 nt	155 Gb
Illumina				
GAIIx	14d	320	2 $\times$ 150 nt	96 Gb
MiSeq	65h	25	2 $\times$ 300 nt	15 Gb
HiSeq 1000 & 1500	8.5d	1,500	2 $\times$ 100 nt	300 Gb
HiSeq 1500 (rapid run)	40h	300	2 $\times$ 100 nt	90 Gb
HiSeq 2000 & 2500	11d	6,000	2 $\times$ 100 nt	600 Gb
HiSeq 2500 (rapid run)	40h	600	2 $\times$ 100 nt	180 Gb
Roche				
GS Junior	10h	0.10	$\sim$ 400 nt	0.035 Gb
GS FLX Titanium XLR70	10h	1	$\sim$ 450 nt	0.450 Gb
GS FLX Titanium XL+	23h	1	$\sim$ 700 nt	0.7 Gb
Life Technologies				
Ion 314 v2 (PGM system)	2-4h	0.55	$\sim$ 200 nt	0.1 Gb
Ion 316 v2 (PGM system)	3-5h	3	$\sim$ 200 nt	1 Gb
Ion 318 v2 (PGM system)	4-7h	5.5	$\sim$ 200 nt	2 Gb
Ion PI v2 (Proton system)	2-4h	80	$\leq$ 200 nt	10 Gb
Pacific Bioscience				
PacBio RS II (P4-C2 chemistry)	0.5-3h	0.05	$\sim$ 5,500 nt	0.275 Gb
PacBio RS II (P5-C3 chemistry)	0.5-3h	0.05	$\sim$ 8,500 nt	0.375 Gb
Helicos BioSciences				
Helicos	N/A	1,000	35 nt	35 Gb
Oxford nanopore				
MiniIon	Early access program launched 11/2013			
GridIon	Under development as of 03/2014			

Table 2.2: Overview of the sequencing system landscape in early 2014. Historical data taken from Glenn [2011], for systems and updates after 2011 manufacturer's specifications have been used. Systems are sorted by manufacturer. One classical Sanger sequencing machine is dyed in light red, 2nd generation systems are shown in light blue, approaches between the 2nd and 3rd generation are labelled orange, actual 3rd generation systems have been coloured green.

2.1.4 Post next generation sequencing

2.1.4.1 Helicos

Helicos BioSciences[6] was the first company to make a 3rd generation sequencing technology commercially available [Braslavsky et al., 2003]. In principle a "sequence-by-synthesis" approach, the HeliScope sequencing system uses single molecule sequencing to obtain the nucleotide sequence of the sample. As a first step template DNA has to be prepared for sequencing by fragmentation and size selection; only sequences below 1,000 nt (ideally 100-200 nt) are suitable for further processing. Within the next phase, fragments are bound to a flowcell by using an oligo(dT)50 / oligo(dA)50 pairing. Poly-A tails are ligated to the sequences prior to fixation mediating a covalent binding. For sequencing, enzymes and one kind of fluorescence-labelled nucleotides are added. The procedure is carried out in cycles similar to the 454 pyrosequencing approach. Each time a base is incorporated, a weak light signal is detected by an imaging device and leftover nucleotides from the last cycle are washed away, thus paving the way for the next nucleotide. The process is repeated 35 × 4 (number of bases) times to generate 35 nt reads and run for each fixated DNA molecule on the flowcell in parallel. A main difference compared to other sequencing solutions is the missing template amplification step, which had the potential to significantly speed up the sequencing process. In its last version the HeliScope system was able to produce up to 1 billion 35 nt reads, corresponding to 35 Gb of sequence data. The HeliScope machine could never generate a significant market share, primarily due to the very limited read length and a very high price for the instrument itself (last price: $ 999,000 in 2009). Sales of instrumentation or reagents was ceased in 2010 after only 20 units sold[7] and the company started to offer sequencing services based on Helicos technology. However, in November 2012 Helicos BioSciences filed for Chapter 11 bankruptcy and neither sequencing services nor reagents or systems are available any longer.

2.1.4.2 Ion Torrent

Life Technologies[8] released their "Ion Torrent Personal Genome Machine"(PGM) System in 2010 after licensing core technologies from DNA Electronics Ltd. and improving the technology until market readiness. The cycle-wise sequencing approach has borrowings from the pyrosequencing technique, it features an unique way of base incorporation detection, deviating from other systems. Like almost all other systems Ion Torrent systems need fragmented template DNA as input, furthermore a PCR step to amplify fragmented sequences has to be conducted - in this case an emulsion-PCR very similar to the one used by 454 instruments (see Section 2.1.3.1). Contrary to other competitors, the Ion Torrent system does not

[6] Helicos BioSciences, Cambridge, Massachusetts, USA, http://www.helicosbio.com (out of business)

[7] http://www.cd-genomics.com/About-Bankruptcy-Helicops.html

[8] Life Technologies, Carlsbad, California, USA, http://www.lifetechnologies.com

use any fluorescent or radioactive dyes nor optical detection systems, but instead detects changes in the pH-gradient during base incorporation events. A flowcell with wells small enough to house only one bead from the emulsion-PCR, together with micro pH-meters fabricated in the bottom of each well is employed for the sequencing reaction. During sequencing, all four bases are floated over the flowcell in cycles, each cycle followed by a washing step. If a nucleotide is incorporated into the newly synthesised DNA strand a H^+ ion is released, thus changing the pH value in the well. In case of longer stretches of consecutive bases the intensity of the signal can be tracked back to the number of nucleotides. This technique however, does only work up to 6-8 consecutive bases and thus introduces 454 typical homopolymer errors. Ion Torrent sells two different systems; the PGM, released in 2010 and updated with newer sequencing chips is capable of producing up to 2 Gb of data out of 5.5 million reads, with an average read length of 200 nt (see Table 2.2 for details). The new benchtop instrument, called Proton was recently launched and starts with the capacity to deliver up to 10 Gb of data in 80 million reads. Unfortunately the length of the reads does not exceed 200 nt, but this flaw can be counterbalanced with a very short runtime (only 2-7 hours) and low costs per run/Gb.

2.1.4.3 Single molecule real time sequencing (SMRT)

Pacific Biosciences[9] was founded in 2004 shortly after an initial proof of concept study was published by Levene et al. [2003]. So called Zero-Mode Waveguides (ZMWs) are minuscule reaction chambers which are used to attach and observe single DNA polymerase enzymes during processing. Only 70 nm $\times \sim$ 100 nm in size, the entry diameter is 6-10 times smaller than the wavelength of visible light (420 to 680 nm [Laufer, 1996]) in order to detect only fluorescence produced during the sequencing process. The detection sensitivity is further increased by the fact that the barrel-formed chamber is illuminated through the glass column plate by a laser, resulting in an illumination focused on the first 30 nm behind the glass bottom - exactly the position of the fixated DNA polymerase. The sequencing itself follows a classical "labelled nucleotide" approach with the exception, that the fluorescence label is located at the phosphor site of the nucleotide compared to the sugar location normally used. All four nucleotides are labelled in different colours and reactions take place without the need of washing steps or other forms of cycles. Each time a nucleotide is incorporated, the specific fluorescence signal is emitted and detected by an optical subsystem. The DNA strand produced by the polymerase is natural and does not possess any chemical modifications originating from the initial nucleotide labeling. Pacific Biosciences launched its first system, the PacBio RS in 2010, while first data was already published in 2009 Eid et al. [2009]. In 2013 the new PacBio RS II system was introduced, followed a chemistry update (P5-C3) in late 2013. With current SMRT-cells the PacBio RS II generates roughly 50,000 reads per

[9]Pacific Biosciences, Menlo Park, California, USA, http://www.pacbio.com

run, which is 120,000 times less than bleeding edge Illumina systems can produce. While read count and coverage cannot compete with any 2nd generation sequencing system the average read length and maximal read length are superior to any other available system, including Sanger sequencing. With standard P4-C2 chemistry 5,500 bases are possible on average and up to 24,000 bases can be achieved in terms of maximal read length. With latest improvements, the fraction of smaller reads could be lowered significantly, allowing for an increased average 8,500 nt read length and more than 30,000 nt at maximum. However, this comes at the cost of a high error rate; roughly 15% errors are introduced during sequencing and signal processing, the major fraction being insertion and deletion errors.

2.1.4.4 Oxford Nanopore Technologies

Oxford Nanopore Technologies [10] started in 2005 as spin off from Oxford University. Nanopores, as the name suggests, are very tiny holes of only a few nanometers in diameter and the key to the company's new sequencing approach. Fundamental research within this field started with work by Kasianowicz et al. [1996] and although various companies licensed the new technology, more than 15 years had to pass until first commercial products hit the market. The novel sequencing system employs a protein-made nanopore (α-hemolysin) embedded within a membrane, where an electrical current is applied. The diameter of the nanopore is hardly sufficient to be passed by a single stranded DNA molecule and as such, only one molecule may be present in the channel at a given time. During the passage variations in the current can be read out and assigned to specific bases therefore resulting in the molecule's DNA sequence. In theory, read length is limited only by experimental runtime, thus allowing very long reads, possibly even longer than those generated by SMRT-sequencing (Section 2.1.4.3). The read error rate was announced to be initially around 4% but should drop to 0.1% once the first commercial systems are sold. The yield per day is expected to be in the order of tens of Gb, therefore competing with smaller systems by Illumina and larger machines by Pacific Bio-Sciences. Two different systems, the disposable "MiniIon" and a cluster-capable "GridIon" system are planned. The MiniIon system is available through an early access program since November 2013.

2.2 Promoter analysis

Genome sequencing has become an irreplaceable tool in today's biotechnology and is involved in a many fields of research, reaching from genome sequencing projects up to transcriptome studies or expression analysis. In the following, an introduction into the functionality of promoters will be given, as promoters are one of the key components in the regulation of gene expression and therefore an auspicious target for sequencing based research.

[10] Oxford Nanopore Technologies, Oxford, UK, http://www.nanoporetech.com

2.2.1 The eukaryotic transcription process

In each organism, the expression of genes combined with a fine grained regulation of this process is an integral part throughout the complete life cycle. While large parts of this regulation machinery could be uncovered within the last decades (reviewed by Butler and Kadonaga [2002]), there are still several gears and springs of unknown function which remain subject of ongoing research.

The transcription process, which is performed by the RNA-polymerase enzyme complex and accounts for rewriting genomic DNA to messenger RNA (mRNA) is the first stage of gene expression. In eukaryotes three slightly differing RNA-polymerase enzymes exist, each fulfilling a specialised task. While RNA-polymerase I produces 45S pre-rRNA (ribosomal RNA) later involved in ribosome forming [Grummt, 1998], RNA-polymerase II synthesises precursors of mRNA, microRNAs, and snRNA (small nuclear RNA) [Lee et al., 2004]. Transfer RNA (tRNA), 5S rRNA and small RNAs (sRNA) are produced by RNA-polymerase III [Willis, 1994]; in plants additionally RNA-polymerase IV [Herr et al., 2005] and RNA-polymerase V [Wierzbicki et al., 2009] are known. The perhaps best studied polymerase however, is RNA-polymerase II, since it is responsible for the synthesis of all protein coding genes within the organism.

In order to start the transcription process, the polymerase has to be positioned in vicinity of the transcription start site of the gene, which, in turn is located at the 5' end of the gene. The so called promoter region includes specific DNA sequences, which are able to bind transcription factors. These transcription factors subsequently provide a guidance system to exactly position the polymerase complex. After this initiation process, the actual transcription is performed. Promoters are heavily influenced by additional regulatory regions like enhancers or silencers and therefore are among the most important concepts in the process of transcription level regulation.

2.2.2 Promoters in industrial biotechnology

In order to achieve optimal efficiency in eukaryotic production cell lines, a high expression level of the specific protein is an important prerequisite. Today promoter sequences with viral heritage are a typical choice, since they deliver very high expression levels under most conditions [Qin et al., 2010; Makrides, 1999]. Here, especially two promoters should be introduced, namely CMV and SV40. The Simian vacuolating virus 40 (SV40) early promoter, isolated in 1960 from rhesus monkey (*Macaca mulatta*) kidney cells [Eddy et al., 1961], showed very high transcription rates when cloned in front of genes of interest [Banerji et al., 1981]. Together with the *Cytomegalovirus* (CMV) immediate early promoter, originating from the *Herpesviridae* humanpathogenic virus family, which exhibited similar effects [Boshart

et al., 1985], both SV40 and CMV became standard promoters for use in eukaryotic expression systems. However, there are significant drawbacks linked with the usage of viral promoters instead of native eukaryotic promoters. The list of possible interferences includes unfolded protein response (UPR) [Isler et al., 2005], endoplasmic reticulum (ER) stress [Tirosh et al., 2005], induced apoptosis, and dependencies of the promoter onto the cell cycle. A solution to this problem would be the use of endogenous promoters which generally should not introduce any side effects due to their optimisation for the host organism. First attempts with human endogenous promoters can be dated back to 1990, when Kim et al. [1990] presented the elongation factor 1α as a well suited tool for protein expression in mammalian expression systems. Later, this system could also be adapted to the Chinese hamster's elongation factor 1α [Deer and Allison, 2004]. Indeed, the search for suitable high yield endogenous promoters, specifically for the Chinese hamster and the derived CHO cell line has just begun.

2.2.3 The eukaryotic core promoter

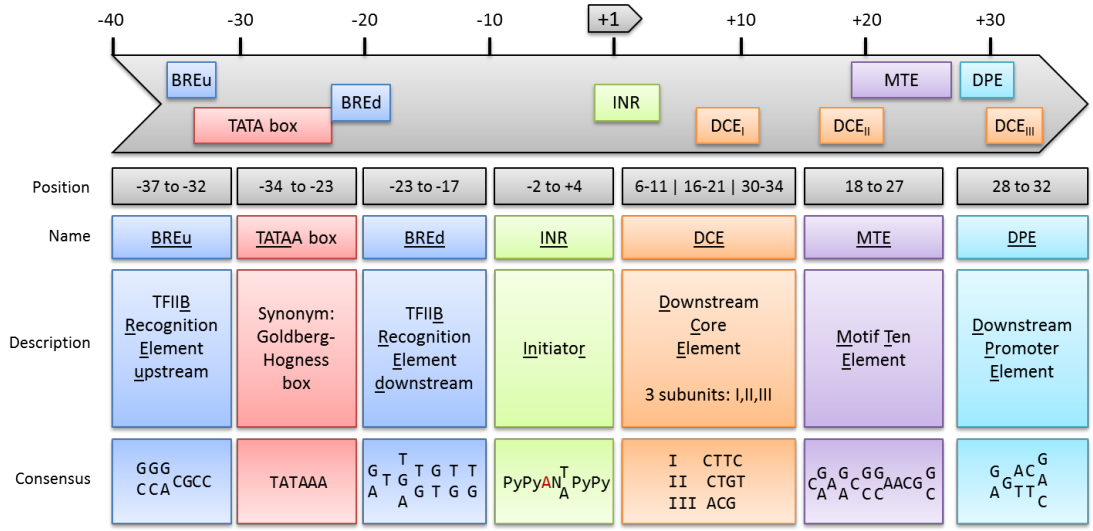


Figure 2.4: The promoter landscape from -40 to +35 bases relative to the transcription start site (TSS) is shown. All elements are colour coded, size is roughly to scale. The first row contains the approximate location, row two to four include name, abbreviation and consensus sequence. The red A within the INR denotes the +1 position, Py is used for pyrimidine bases (C/T, IUPAC[1] code Y). Data from Maston et al. [2006]; Gershenzon et al. [2006].

The eukaryotic core promoter for RNA polymerase II is based on a small set of regulatory elements, typically arranged from -40 to +35 relative to the transcription start site (Figure 2.4) [Juven-Gershon, 2006]. Consensus sequences of these

[10]International Union of Pure and Applied Chemistry

elements are generally rather small, with a size ranging from 3 to 12 nt.

The B recognition element upstream (BREu) was discovered by Lagrange et al. [1998] and is the first element of the core promoter, typically located from -37 to -35. As the name suggests, the motif is bound by transcription factor $TF_{II}B$ and recognised through a helix-turn-helix motif. Crystallographic structure analysis lead to the presumption that it can either increase or decrease the transcription rate of the corresponding gene [Lagrange et al., 1998; Littlefield et al., 1999], the precise working mechanism however remains unresolved. The exact consensus sequence is shown in Figure 2.4.

The TATA Box (synonymously Goldberg-Hogness box), located from -34 to -23 and partly covered by BREu is probably one of the most prominent members of the core promoter. Initially described in *Drosophila melanogaster* by Goldberg [1979], the element received its name from the concise TATAA consensus sequence, which is also known as Pribnow box [Pribnow, 1975] from bacterial promoters. The TATA box binding protein (TBP), its corresponding transcription factor, becomes part of the pre-initiation complex (PIC) after binding. Promoters with functional TATA elements are associated with developmental regulation and and differentiation processes [Carninci et al., 2006]. While first studies assumed that TATA box carrying promoters are the rule rather than the exception, this pictures was relativised by later studies, reducing the estimated amount of TATA promoters down to 25 % [Suzuki et al., 2001] and later to only 10 % [Carninci et al., 2006].

Downstream of the TATA box, from -23 to -17 a second B recognition element, termed BREd, was confirmed by Deng and Roberts [2005]. Although BREu and BREd share the same transcription factor ($TF_{II}B$), their consensus sequences pose hardly any similarity (see Figure 2.4 for a comparison) which is emphasised by the fact that BREd does not rely on the BREu typical helix-turn-helix binding domain. Both, BREd and BREu can be understood as extensions of the TATA box [Juven-Gershon, 2006] and may exhibit either increasing or decreasing effects on transcription levels [Deng and Roberts, 2005; Lagrange et al., 1998].

The area surrounding the actual transcription start site from -2 to +4 is known as the initiator region (INR) and was discovered as one of the first eukaryotic promoter elements in human HeLa cells [Scherer et al., 1953] by Corden et al. [1980]. The consensus sequence of the INR motif is very pyrimidine-rich and features typically an adenine at the +1 position. Deletion studies showed a broader range of transcription start site locations combined with reduced transcription rates [Grosschedl and Birnstiel, 1980]. Further work additionally revealed reciprocal, effects ranging from synergistic influence on transcription levels in case of spacings from 25 to 30 nt between TATA box and INR to independent functions of TATA and INR if their spacing exceeds 30 nt [O'Shea-Greenfield and Smale, 1992]. The corresponding transcription factor of INR is $TF_{II}D$ which is suggested by the fact

that in absence of $TF_{II}D$ INR does not exhibit any regulatory effects [Smale, 1997].

The upstream region of the eukaryotic promoter is dominated by three different elements, the first one being the downstream core element (DCE). The DCE itself features three subunits (DCE S_I to DCE S_{III}), scattered though large parts of the core promoter upstream region and positioned from +6 to +34 (Figure 2.4) [Lewis et al., 2000]. In contrast to other regulatory elements, the individual subunits of the DCE are very short sequence tags of 3 to 4 bases with only minimal variations [Lewis et al., 2000]. Common for DCE S_I and DCE S_{II} is the almost exclusive use of pyrimidines. DCE S_{III} uses only one pyrimidine base and in contrast in to S_{II} and S_{III} able to function on its own [Lewis et al., 2000]. In order to influence the transcription process, DCE elements utilise the $TF_{II}D$ transcription factor. Functional characterisation of the DCE however, is still based on assumptions, reaching from changes in promoter specificity or involvement in special regulatory networks.

Co-localized with DCE S_{II}, the motif ten element (MTE, +18 to +27) requires only a functional INR and a correct spacing between both elements. The MTE is therefore able to promote transcription in the vast majority of promoters missing TATA boxes. However, if a TATA box is present, the combination of MTE and TATA shows significant synergistic effects [Lim et al., 2004]. Originally discovered in *Drosophila melanogaster*, the MTE motif was also detected in higher eukaryotes up to human and mouse [Lim et al., 2004].

Currently, the farthermost located downstream motif of the eukaryotic promoter is the the downstream processing element (DPE) [Burke and Kadonaga, 1996; Juven-Gershon, 2006] (+28 to +32). While DPE together with INR can account for basal transcription if spacing between elements is correct it is also found in many TATA less promoters where it is suspected to take over the role of the TATA box and partially in promoters with TATA boxes in place [Kutach and Kadonaga, 2000]. Like most other motifs, DPE is recognised by $TF_{II}D$ (subunits TAF6 and TAF9).

2.2.4 Methods of transcription start site identification

In order to gather knowledge of promoter regions and their general architecture, it is crucial to obtain precise location information, if possible exact on nucleotide level. On the one hand, *in silico* predictions based on sophisticated models and algorithms can be used to search DNA sequence on genome or local scale for auspicious positions of transcription start sites. On the other hand, biological sequencing experiments and subsequent data analyses may either verify *in silico* predictions or add TSS positions not recognized by computational methods.

2.2.4.1 Computational methods

First approaches for eukaryotic promoter prediction started around 1995 with PromoterScan [Prestridge, 1995] and PromFind [Hutchinson, 1996] (Table 2.3). During this phase different computational methods were evaluated, including structural features like hexamer frequency differences between coding regions and promoter areas [Hutchinson, 1996], Markov chains [Audic and Claverie, 1997], TATA box position weight matrices (PWMs) [Prestridge, 1995], or transcription factor binding site densities [Prestridge, 1995]. However, these tools rely in large portions on extrinsic data sources like transcription factor databases or verified TATA box consensus sequences to build reasonable models. Even though these first bioinformatics approaches paved the way for more sophisticated implementations, none of the first generation predictors achieved sensitivity values >60 %, while most tools not do exceed 30 % [Fickett and Hatzigeorgiou, 1997].

Over the years, molecular biological knowledge of promoter structures, transcription process and DNA sequence features increased and allowed for the development of novel *in silico* approaches. Due to sequencing technology advances, the human genome set the new gold standard for promoter and TSS prediction, effectively rendering most previous software tools infeasible. Two novel approaches subsequently appeared, the first designed for genome scale application, the second also able to work on single gene level. With the advent of large genomes, such as the human genome, a common approach is the screening and scoring of each nucleotide of the underlying genome, while the scoring is mostly realised through classification algorithms with cross-validation [Abeel et al., 2009]. Typical representatives of this class are ARTS [Sonnenburg et al., 2006], ProSOM [Abeel et al., 2008b], and EP3 [Abeel et al., 2008a] (Table 2.3). Another possibility to detect promoter regions or TSSs employs a much more local scope and does not accumulate scores throughout the whole genome. Therefore only auspicious start/stop positions for promoter regions, potentially combined with a confidence scores, are reported. This method is used for instance in PromoterExplorer [Xie et al., 2006] and a proposed software by Wu et al. [2007].

In order to assess performance and accuracy of this second generation promoter and transcription start site prediction tools, a first proposed gold standard was established by Abeel et al. [2009]. The study was able to confirm a bias for most tools towards CpG containing promoters, commonly associated with housekeeping genes [Carninci et al., 2006], while other promoters not exhibiting CpG islands seem to be under-represented. Further bias is caused by over-represented promoters of highly transcribed genes compared to promoters of relatively weak expressed genes.

Tool	Method description
PromFind	Based on differences between hexamer frequencies in promoter regions, coding, and non-coding regions [Hutchinson, 1996]
TSSG & TSSW	Linear discriminant function combines TATA box scores and triplet preferences around the TSS [Solovyev and Salamov, 1997]
PromoterScan	Uses TATA PWMs (position weight matrices) and densities of transcription factor binding sites [Prestridge, 1995]
Nameless tool	Promoter recognition algorithm based on Markov transition matrices [Audic and Claverie, 1997]
PromoSer	Promoter and transcription start site identification, web based, source genome data dates to 2003 [Halees, 2003]
CoreBoost_HM	TSS prediction based on histone modification signals, web based, 100 Kb maximal input [Wang et al., 2009]
NNPP2.2	Neural network based, utilises difference between TSS and translation start site (TLS) [Burden et al., 2005]
MotifLab	Combines several data sources like chromatin accessibility and epigenetic state of the cell [Klepper and Drabløs, 2013, 2010]
McPromoter	Based on stochastic segment models (SSMs) and interpolated Markov chains [Ohler et al., 2000; Ohler, 2006]
EP3	Uses large scale DNA structural features to predict promoters [Abeel et al., 2008a]
Eponine	Based on a hybrid machine-learning algorithm, developed for mammalian genomes [Down and Hubbard, 2002]
GPMiner	Meta tool, identifies TSSs and regulatory features, uses McPromoter, Eponine, and NNPP2.2 [Lee et al., 2012]
ProSOM	Facilitates unsupervised clustering by using self-organizing maps to recognise promoter regions [Abeel et al., 2008b]
ARTS	Employs Support Vector Machines (SVMs) with advanced sequence kernels [Sonnenburg et al., 2006]

Table 2.3: Selection of TSS / promoter prediction tools. The first four tools have been chosen exemplarily as representatives for the first generation of prediction tools [Fickett and Hatzigeorgiou, 1997]. The second part of the table features web-based implementations, part three is dedicated to more recent works. An extensive review of *in silico* solutions for promoter and TSS discovery was conducted by Narlikar and Ovcharenko [2009].

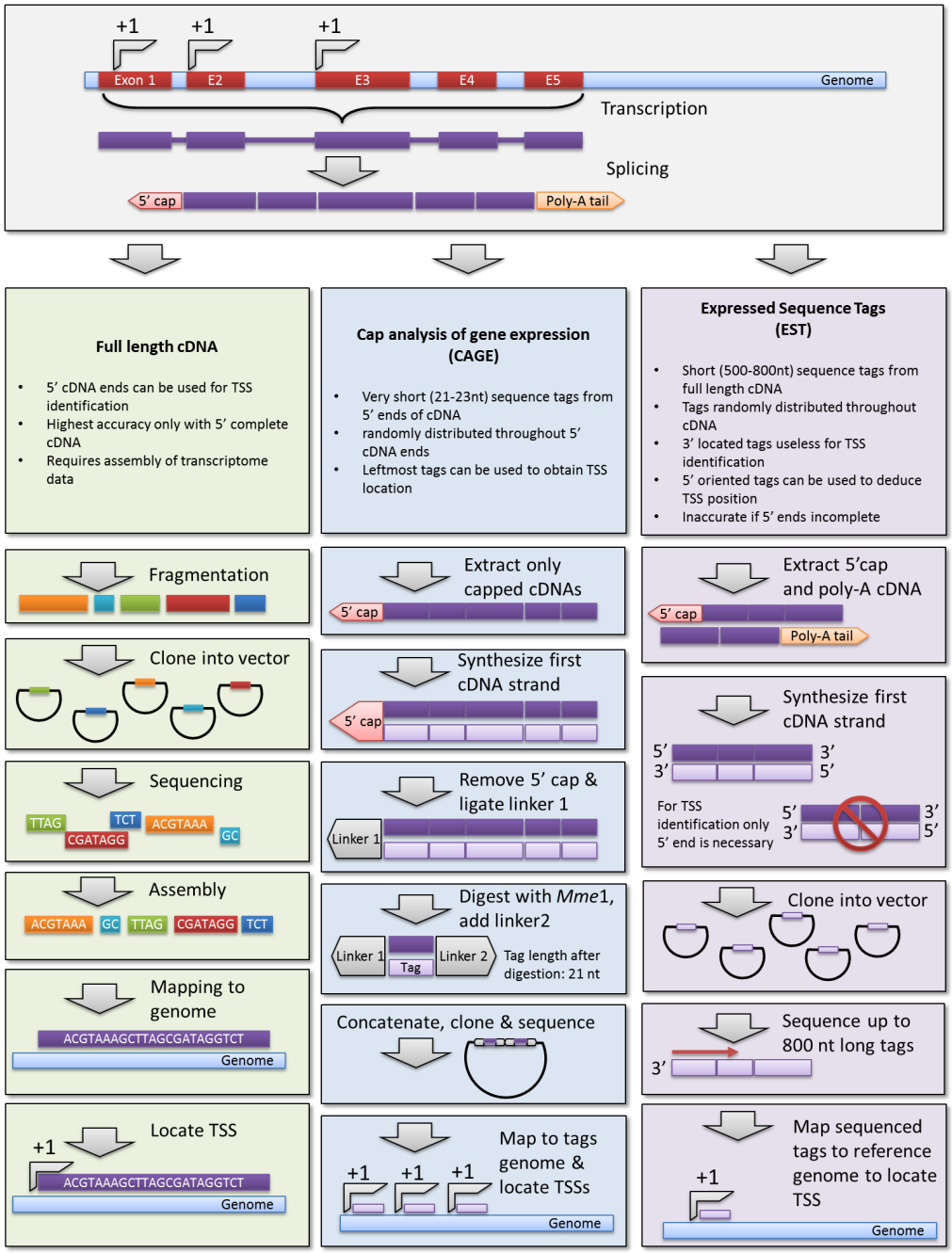


Figure 2.5: Overview of transcription start site (TSS) detection methods. **Left** (green): TSS can be detected by full length cDNA sequencing. After assembly of sequencing reads, full length cDNA sequences are mapped onto a suitable reference genome. The leftmost (5') mapping position corresponds to the TSS (given the cDNA assembly yielded a full length sequence). **Middle** (blue): Cap analysis of gene expression is an approach more focused on 5' mapping since several 5' end tags ($\approx$ 21 nt) of different genes are fused and sequenced in one read, therefore increasing the overall throughput of detectable TSS. After sequencing the tags have to be mapped onto a suitable reference. **Right** (purple): Expressed sequence tags (ESTs) are randomly distributed tags much longer than CAGE tags (about 500-800 nt). Therefore not all tags can be used for TSS mapping while additionally more sequencing is performed for non 5' specific tags, thus lowering the overall TSS yield.

2.2.4.2 Biotechnological methods

As shown, in many cases results obtained by *in silico* methods can give first insights and a general idea about TSS positions and possible promoter regions. However, all these methods are biased in one way or another and as such will not be able deliver a complete and correct picture of the TSS landscape of eukaryotic organisms. Although computational methods are typically more cost efficient due to the fact that no expensive reagents are required, biological experiments can provide new results or findings which cannot be predicted by computational methods since algorithms will usually only report those results which are related to their programming.

Full length cDNA Generally, transcriptome sequencing focuses on the reconstruction of complete complementary DNAs (cDNA), in order to gain information about the protein structure and therefore possible functions. As such, ideally the cDNA is sequenced completely from the 5' UTR up to the 3' UTR and later assembled into its prior form. In order to precisely locate the transcription start site of a given transcript, it is mapped against a reference genome with BLAST [Altschul et al., 1990] or similar tools. Transcription start site identification in this case is a byproduct and not the intended use case for this strategy. Full length cDNA sequencing was introduced in times of Sanger sequencing, hence the throughput of this technology is very limited. Additionally the vast majority of sequencing information is used for coding parts of the mRNA rather than to identify as many TSSs as possible. All processing steps of this method are summarised in Figure 2.5

Expressed sequence tags - ESTs As previously mentioned, full length cDNA methods has two significant drawbacks. First, the amount of required sequencing data is relatively high. Second, a subsequent assembly process is mandatory to obtain a correct transcript which can be mapped back to the reference genome. Indeed, both disadvantages were addressed in a study by Adams et al. [1991] presenting an effort to get a broad overview of transcripts in a large number of samples by using the limited Sanger technology. In contrast to full length sequencing which employs several reads per transcript to fully cover its sequence, so called expressed sequence tags (ESTs) consist of one read only, yielding a typical length of 500-800 nt. This single read starts either from the 5' or the 3' end of the transcript, leaving large portions of the transcript untouched (seed Figure 2.5 for a graphical summary). However, reads longer than 150 nt are already sufficient for similarity searches and genome mapping on a human genome scale [Adams et al., 1991]. Although ESTs were intended as a tools for expression profiling and are widely used even today, due to their transcript end focused sequencing strategy they proofed to be an excellent tool for TSS identification purposes.

Cap analysis of gene expression - CAGE The approach of non-complete sequencing was enhanced and optimised to further increase the possible yield of transcription start sites. The typical tag size of EST sequencing was reduced by using

restriction enzymes and varies between 21 - 23 nt. These tags are concatenated into a single vector where several tags can be sequenced in a serial way. Compared to previous approaches cap analysis of gene expression (CAGE) [Kodzius et al., 2006; Shiraki et al., 2003] was able to reduce costs while at the same time increasing overall yield of tags. This came at the price of a negligible coverage of the original transcript. The approach therefore results in much higher throughput, since only concatenated tags are sequenced rather than full transcripts. Sequenced tags can be mapped onto a suitable reference genome, which in turn reveals transcription start sites due to the 5' aligned location of the CAGE tags. Recent studies showed however, that the CAGE protocol is prone to non-specific G at the tag's 5' end, therefore leading to flawed mapping positions within the reference genome [Zhao et al., 2011] and finally to bogus tag to gene mappings.

2.3 Genome sequence assembly

Genome sequencing, with its different strategies and approaches as described in Section 2.1, is the first step towards an organisms genome. Until sequencing of whole chromosomes in form of single molecules will become possible, the first step during each genome project has to be a fragmentation of the source DNA. The initial technique, called *shotgun sequencing*, was developed by Sanger et al. [1982] when sequencing the 48 Kb genome of the λ-bacteriophage.

Shotgun sequencing uses BACs (Bacterial Artificial Chromosomes) to amplify large regions of DNA in bacterial fertility plasmids (F-plasmids). Those BACs are mapped back to the source genome to establish a so called 'tiling path', a list of BACs needed to cover the genome sequence. The DNA integrated into the chosen BACs ('insert') is fragmented further by an undirected shearing process (therefore the shotgun metaphor) which results in many smaller fragmented DNA sequences ready to be sequenced. Several years later, this technique could be employed to sequence the much larger genome of *Haemophilus influenzae* (1.83 Mb, $\approx$ 38 times larger than the λ-bacteriophage, [Fleischmann et al., 1995]). BACs have the advantage that by mapping to the organism's genome a first impression of the order of the cloned fragments can be estimated, which eases the subsequent assembly process by adding positional meta information to the sequenced DNA fragments. However, after sequencing all fragments have to be arranged in proper order to resemble the organisms genome correctly. This process, typically envisioned as a large jigsaw puzzle, is referred to as *genome assembly*. Although shotgun sequencing paved the way for today's genomics, there are shortcomings, such as problems during amplification, possibly due to toxicity for the host bacteria or other unintended side effects [Pop et al., 2002].

Over time, the initial shotgun approach evolved into *whole genome shotgun sequencing*, or WGSS. Compared to is predecessor, WGSS does employ BACs but

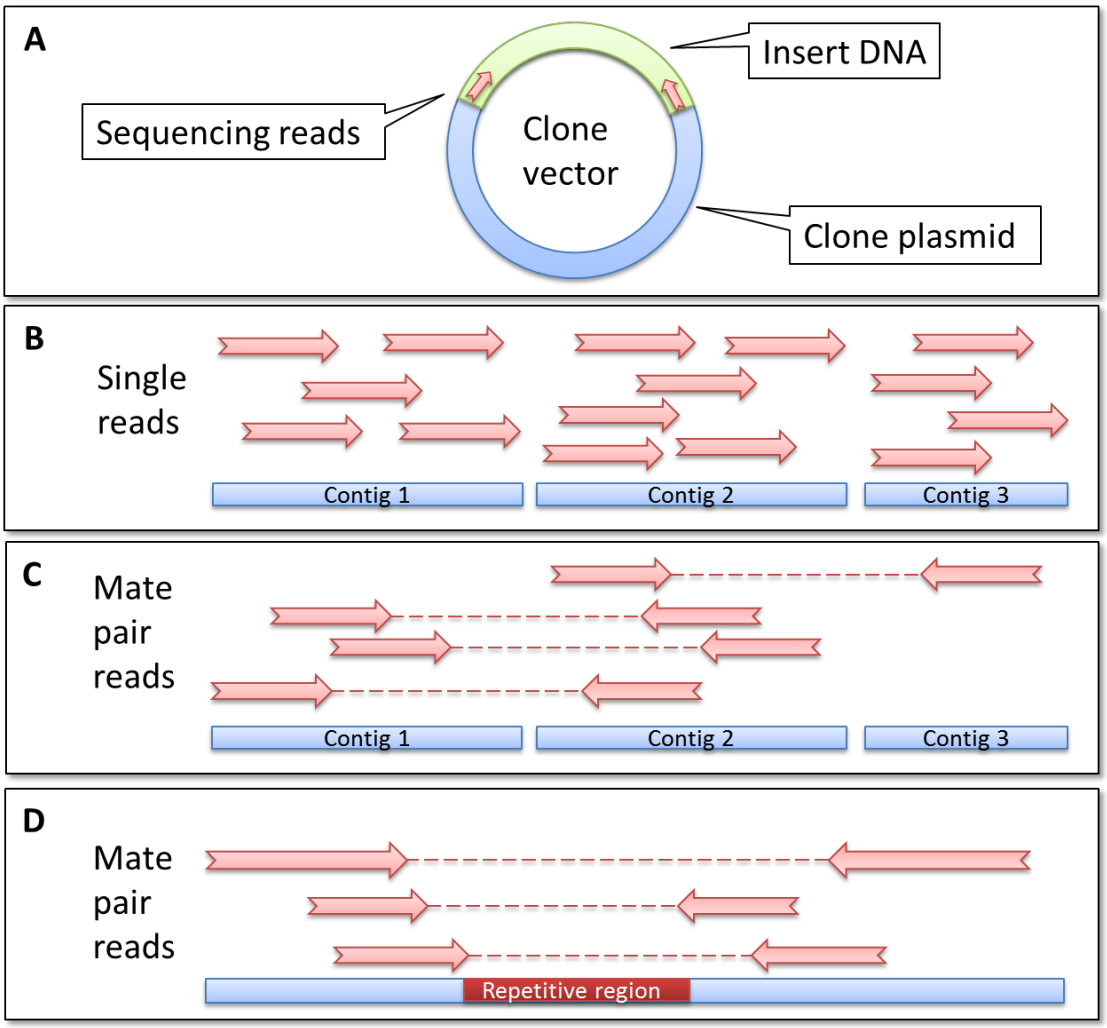


Figure 2.6: Overview of clone based sequencing. **A)** Fragments of source DNA are introduced into cloning vectors, which are amplified in host organisms like *Escherichia coli*. The insert part is sequenced after amplification either from one side, therefore producing single reads or from both sides, resulting in paired end/mate pair reads. **B)** If sequencing is performed only from one end of the source DNA, the single reads are the end product. Those reads can be assembled to form contigs, however, assigning an order to those contigs with only single reads available is hardly possible, since no reads are spanning gaps between contigs. **C)** In case of mate pair sequencing, each read pair carries additional information in form of the insert size. This means that for each pair, the distance between both reads is known. As this insert size is much larger than the read length itself, a pair anchored in two different contigs may be used as a linker, thus establishing an order among the contigs during scaffolding. **D)** Many repetitive regions are longer than even the longest sequencing reads. Repeats therefore pose a complex problem during assembly which may be avoided by mate pair reads. A pair of two reads might span the repetitive region completely, thus giving an estimate of its length and ease the assembly process.

instead the DNA is fragmented into specific sizes, for example 2, 3, 8, and 10 Kb. This fragmented DNA is sequenced after cloning into host cells and amplification. Sequencing takes place from both ends, thus generating two reads for each clone which usually do not overlap (due to the maximum read length of about 1 kb of the Sanger sequencing approach). This strategy, also referred to as mate pair sequence or paired end sequencing depending on the sequencing solution makes it possible to span repetitive regions (Figure 2.7) and aids during later scaffolding steps (see Figure 2.6). The genome assembly of the fruit fly *Drosophila melongaster* (135 Mb) [Myers et al., 2000] was a first milestone, achieved without any information from BAC clones, while using an assembly software specifically tailored to deal with large genomes and therefore large amounts of sequencing data. This software, known as WGS-assembler or Celera-assembler [Myers et al., 2000], was subsequently used during the human genome project in 2001. It is important to mention that a certain sequence *coverage* is needed for a genome assembly, where coverage refers to the degree of oversampling, i.e. how many times a specific base of the genome is theoretically covered by a sequencing read. The required coverage varies depending on sequencing technique and the organism which is sequenced. As a rule of thumb, more coverage yields an easier assembly process, although at a given point more coverage will not improve the assembly any further or may even start to generate biases and side effects. Typical values for prokaryotic organisms are within the range of 25 - 30× [Aury et al., 2008], for large large mammalian genomes 100× and more are definitely reasonable [Li et al., 2010]. This fact also offers an explanation for the high costs of sequencing projects in the late 1990s and early 2000s when Sanger sequencing still was the only available sequencing technology. The human genome project had an average coverage depth of 12×, resulting in 12 × 3.3 Gb[11] of sequence information to be processed and overall cost for the whole project aggregated over $3,000,000,000. However, even if the genome is statistically covered above a given threshold, there will always remain regions which will not be covered at all, may it be areas of high G+C content (see Section 2.1.3.1), highly repetitive regions (Figure 2.7), or centromeric regions of chromosomes.

2.3.1 The sequence assembly problem

Starting with the first genome sequencing efforts a new branch of bioinformatics software emerged that, even after over 30 years, is in very active development. Genome assembly programs start with nothing more but a set of reads produced by a sequencing machine and try to reconstruct the original genome as close as possible. The assembly process, ordering, connecting, and merging reads together back into one genome can be expressed in a more formal way - the shortest common superstring problem (SCS) [Tarhio and Ukkonen, 1988].

[11] Human genome assembly GRCh37, Jun 2013, http://www.ensembl.org/Homo_sapiens/Info/Annotation

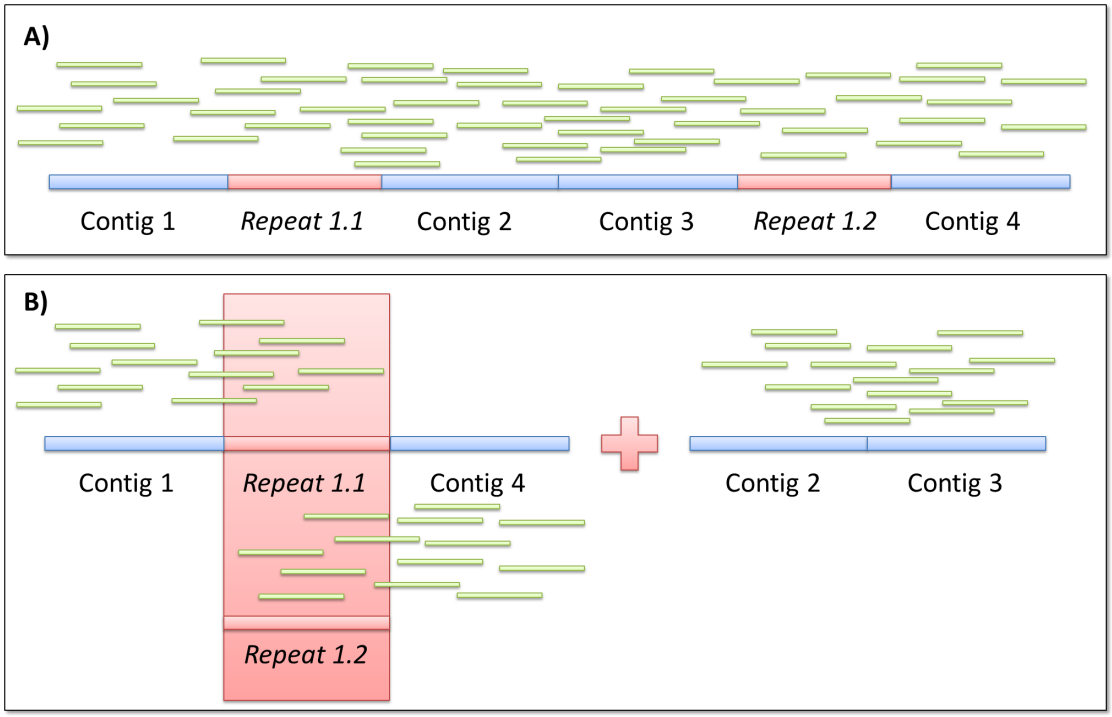


Figure 2.7: Typical assembly error induced by repetitive regions with no read pair information. **A)** Several contigs and repetitive regions are covered by multiple reads. The order shown here is the original ordering as found in the source genome. The two repetitive regions 1.1 and 1.2 are assumed to contain the same sequence, which is common e.g. for satellite DNA. **B)** If only single reads without mate information are available, misassemblies are expected. Contig 1 is connected to Repeat 1.1 by a few reads sharing the transition sequence between these segments. However, since Repeat 1.1 and 1.2 share the same sequence, some reads belonging to 1.2 are clustered together with reads from 1.1, thus creating a crossover with Contig 4 (instead of Contig 2). Contig 2 and 3 are subsequently merged correctly, but do not have any connection to the repetitive regions their they where supposed to separate.

Given a finite set of sequences $S = \{S_1, S_2, S_3, \ldots, S_n\}$ over an alphabet $\Sigma = \{A, C, G, T\}$[12] , find a shortest sequence G, such that every read $S_i \in S$ is contained in G. The problem may be reformulated to better match the "real" assembly process in such a way, that errors within the sequences are allowed. This reformulated problem is known as the reconstruction problem [Kececioglu and Myers, 1995]. Given all symbols introduced above while adding an error rate $0 < \varepsilon < 1$, the minimal number of insertions, deletions, or substitutions $d(X, Y)$ between two strings X *and* Y, and the reverse complement $\overline{X}$ of a string X, find a shortest sequence G, such that for every read $S_i \in S$ there is a substring X_i of G such that

$$min(d(S_i, X_i), d(\bar{S}_i, X_i)) \leq \varepsilon \cdot |X_i| \,. \tag{2.1}$$

[12]additional letters of the IUPAC code may also be part of Σ

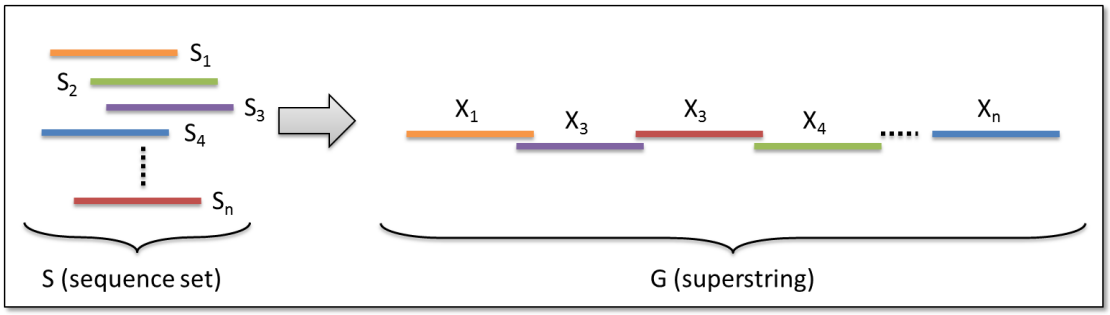


Figure 2.8: Visual representation of the shortest common superstring problem.

Both problems, however, are known to be NP-complete and therefore infeasible to be solved exactly for an arbitrary number of sequences [Maier, 1978]. Therefore, different heuristics have been developed in order to solve the problem in polynomial time, each adapted to current sequencing technologies and optimised to exploit continuously evolving computing hardware. The sacrifice for this approximation lies in the not necessarily optimal solution - the returned superstring is not guaranteed to be the shortest one i.e. the correct genome assembly.

2.3.2 Greedy algorithm based methods

First specialised tools for sequence assembly were developed in the late 1980s and early 1990s and their lineage goes back to sequence alignment programs developed even earlier. This is owed to the general paradigm of sequence assembly, i.e. find two overlapping reads and merge those reads together to a larger fragment if the overlap score is above a given threshold. During early days of genome assembly and sequencing, the number of reads produced rarely exceeded several thousands or tens of thousands of reads. The *greedy algorithm* first computes and scores all possible overlaps between the reads. In a second step reads are merged based on their overlap scores until no more reads are unconnected. Obviously this strategy works best for a limited number of more or less unresembling reads, as too many very similar reads introduce a high level of ambiguity during overlap computation. Examples for tools based on the greedy algorithm are phrap [Green, 1994], gap4 [Bonfield et al., 1995], TIGR [Sutton et al., 1995], and CAP3 [Huang and Madan, 1999].

These programs became standard tools for genome assembly workflows during the next years, as Sanger sequencing was not superseded until NGS technologies emerged. An advantage of greedy assemblers is the relatively low implementation complexity compared to other contemporary approaches. However, due to their simple base scheme, problems arise if repetitive regions occur as these may introduce false positive overlaps which eventually lead to misassemblies. Normally, no mate pair information is processed during the assembly, thus introducing addi-

tional difficulties regarding contig orientation and placement. Another drawback of greedy implementations is the high demand for memory - one gigabyte of RAM for each megabase of genome [Pop et al., 2002] - which rendered these tools useless for larger genomes back in the late 1990s when even high end server systems rarely were equipped with more than 32GB of RAM ($\approx$ 32 Mb maximal genome size).

With the advent of NGS sequencing, several tools were revived and improved the greedy algorithm for use with Illumina and 454 data. The first explicit short read assembler was SSAKE [Warren et al., 2007]; SHARCGS [Dohm et al., 2007] added several preprocessing and postprocessing steps to SSAKE, while VCAKE [Jeck et al., 2007] is able to work with non-perfect overlaps, thus accounting for sequencing errors and possible SNPs (single nucleotide polymorphism) in polyploid organisms. Some of these assemblers later found their way into hybrid assembly pipelines, which allow for a combination of different sequencing technologies within one genome project.

2.3.3 Overlap graph based methods

As sequencing projects began to grow from small bacterial genomes to larger eukaryotic organisms like yeasts and plants, hitherto assembly strategies had to be adapted for larger amounts of longer reads produced by the now automated Sanger procedure. Greedy assemblers were phased out stepwise by a new approach called *overlap-layout-consensus* (OLC) [Kececioglu and Myers, 1995].

As the name suggests, the new assembler generation works in a three-tier process starting with the computation of pairwise overlaps between the reads (overlap phase), equivalent to the previous greedy implementation. Reads are segmented into k-mers prior to the alignment step in order to obtain a list of promising candidates (seed-and-extend technique). A k-mer is a substring of length k of the original sequence, where the next consecutive k-mer starts exactly 1 position shifted to the right. That way, not all pairwise alignments have to be computed and thus the overall runtime and memory footprint is streamlined. The choice of a suitable k-mer value, i.e. the k-mer length is crucial for further processing as both to large and to small values have negative impact on the final assembly.

The second step differs from the greedy algorithm in such a way, that OLC-based assemblers use an abstract graph-based data structure during the layout phase in contrast to the at most implicit graph concept of greedy tools (Figure 2.9 D and E). The overlap graph consists of nodes representing reads and directed edges representing an alignment between these two reads. During layout phase the overlap graph is simplified and errors are removed. The overlap graph itself does not have to include all information for each read, therefore the graph structure can

be implemented more memory efficient. The algorithm has to visit each vertex of the graph once during consensus generation, thus forming an Hamiltonian cycle.

During the last phase a consensus sequence is computed out of multiple sequence alignments (MSA) of overlapping reads. As MSAs are computational expensive operations (NP-complete, [Wang and Jiang, 1994]), it is important to simplify the graph in the previous step as far as possible to keep the complexity of the consensus step on a reasonable level. The MSA can also be divided into multiple pairwise alignments, thus reducing the problem complexity.

Several implementations of this OLC blueprint have been published and improved throughout the 2000s, starting with the Celera-assembler (also WGS assembler) [Myers et al., 2000; Istrail et al., 2004; Busam et al., 2006; Miller et al., 2008]. Other well known and widely used OLC-based tools are Arachne [Swan et al., 2002], PCAP [Huang et al., 2003], and Edena [Hernandez et al., 2008]. Indeed, one of the probably most used assemblers, Newbler [Margulies et al., 2005] is also based on the OLC concept. Newbler is part of the software suite that is delivered with sequencing machines sold by Roche and therefore heavily optimised on 454 based input data. The OLC concept, however, has not been abandoned, as still new methods such as clever string indexing and compression are combined with the classical OLC technique, for example in form of the SGA assembler [Simpson and Durbin, 2012].

2.3.4 de Bruijn graph based methods

Although de Bruijn graphs were not specifically designed in a bioinformatics context, a novel usage scenario [Pevzner, 1989] was found within the then new technique of sequencing by hybridisation (SBH) [Lysov et al., 1988; Southern, 1988]. This microarray-related technique works by hybridisation of sequences against an array of all possible k-mers of a given length. Even if this approach was not able to become another competitive sequencing technology, it paved the way for a k-mer/sequencing relation in bioinformatics. When constructing a de Bruijn graph, the k-mer length has to be chosen. This variable has a dramatic impact on assembly performance and normally has to be evaluated for each new assembly project to achieve optimal results. Reads are segmented into k-mers (Figure 2.9 B and C) and subsequently inserted into the graph. In detail, a de Bruijn graph in sequence assembly context has an edge for every k-mer of the input reads and each edge corresponds to a k-1 overlap between two nodes (Figure 2.9 F). Therefore, the two nodes, which each edge is connected to represent the $(k-1)$ prefix and suffix of the edge.

In contrast to overlap graphs, de Bruijn graph (DBG) [de Bruijn and Erdos, 1946] assemblers generally represent sequencing reads as edges. However, mixed forms, combining labeled vertices with the k-mer approach of a classical de Bruijn graph have also be implemented [Compeau et al., 2011]. The difference between

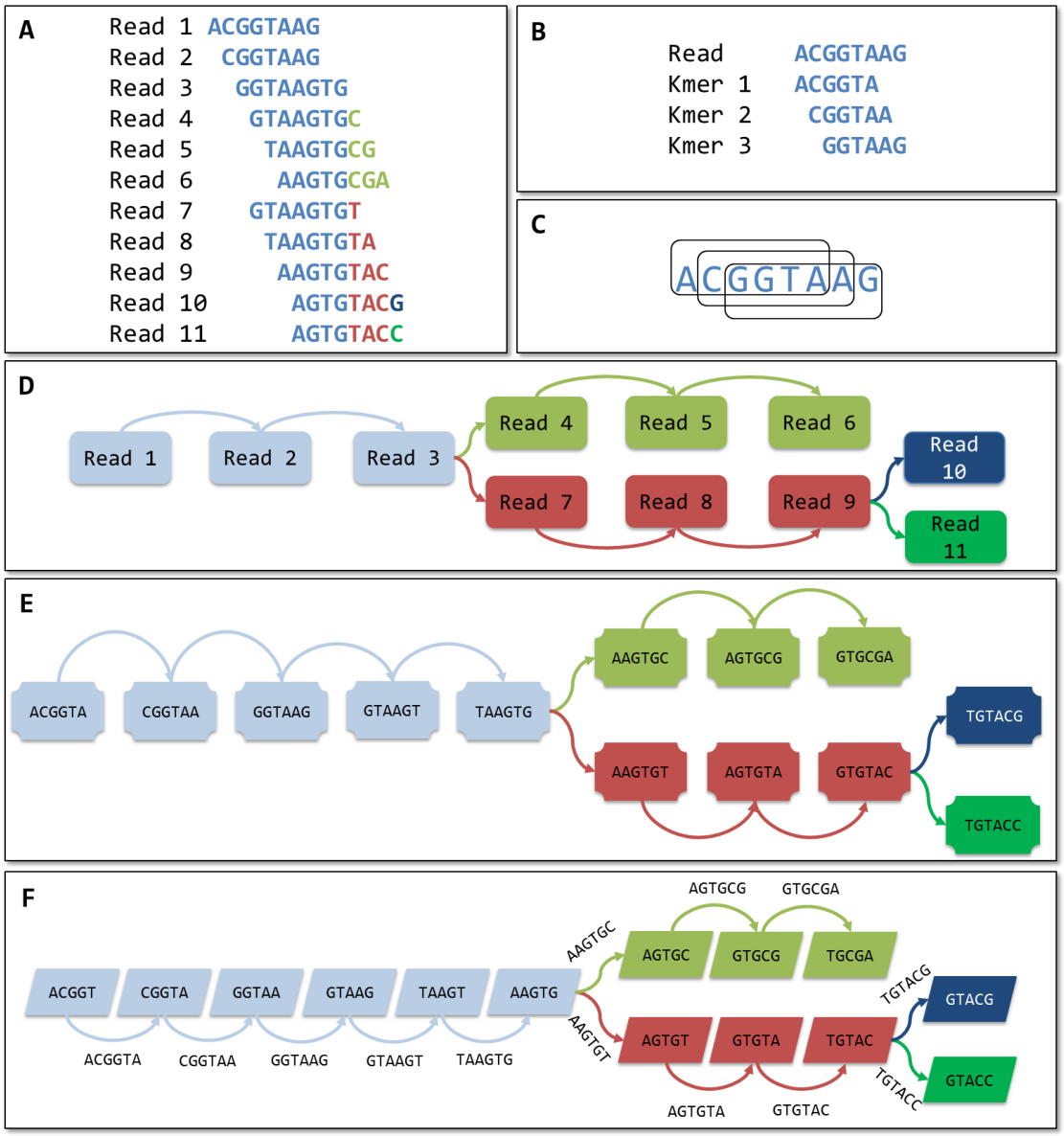


Figure 2.9: Examples for graph structures of different assembly approaches. **A)** List of reads used in further examples. Differences indicating new branches are color-coded. **B)** Segmentation of the first read into three k-mers for $K = 6$. **C)** Another representation of the k-mer segmentation process, showing the three different frames. **D)** Typical graph of an OLC-based assembler. Nodes represent complete reads, directed edges between nodes imply an alignment of these two reads. **E)** A mixed k-mer/overlap approach. Each node contains one k-mer. As previous, edges here only indicate a pairwise alignment. To find a consensus sequence, each vertex has to be visited once, therefore we face an Hamiltonian cycle. **F)** A de Bruijn graph representation, used by modern short read assemblers. Nodes are $(k-1)$-mers, edges are labeled with k-mers. Each edge needs to be visited once to generate a consensus sequence, resulting in an Eulerian cycle.

both approaches lies within the consensus generation, which involves finding the Eulerian cycle [Euler, 1741] for genuine de Bruijn graphs and the search for the Hamiltonian cycle for mixed approaches. The applicability of de Bruijn graphs for genome assembly purposes was first proposed in form of the EULER software [Pevzner et al., 2001], the first DBG-based assembly approach.

The DBG representation is well suited to seize large amounts of short and heavily redundant reads produced by Illumina sequencers, even if the memory consumption can become critical for very large projects [Miller et al., 2010]. Since the market share of Illumina sequencers became dominating, the current focus for assembler development is clearly 'short' read assembly. Following EULER, several competing assemblers have been published, including Velvet [Zerbino et al., 2009], ABySS [Simpson et al., 2009], ALLPATH-LG [Butler et al., 2008], SOAPdenovo [Li and Homer, 2010] and an improved EULER version, EULER-SR [Chaisson et al., 2009].

Even if today's Illumina protocols nearly compete with 454 Titanium in terms of read length, the challenge of handling gigabases of data currently favours de Bruijn based assemblers. This might change when extremely long reads of 10 kb or more are available in larger quantities as long reads are a domain of OLC-based tools.

2.4 Prerequisites for nucleotide-level promoter analysis

Within the chapter several core techniques were introduced, all of them in one way or another prerequisites for promoter analyses on nucleotide level. Mature sequencing techniques such as state of the art protocols are required to generate sequencing data of either genomic or transcriptomic origin. The raw data however requires assembly software which turns the vast amount of short reads back into a genome or transcriptome. While a genome is essential for the promoter analysis pipeline described in Chapter 5, the transcriptome can be used to improve the *in silico* gene prediction of gene starts. The promoter analysis is able to suggest a range of potential endogenous promoters with suitable expression profiles for a given product together with the set of known regulatory elements present in each promoter. Due to negative effects of viral promoters commonly used for overexpression in eukaryotic systems, these high-yield endogenous promoters are of great interest for the biotechnological community.

CHAPTER 3

Motivation and thesis aims

The CHO cell line has become a valuable source for many different biopharmaceuticals and is of crucial interest for industrial biotechnology. Given this context, it seems only logical to collect relevant data from all possible sources. Indeed, the amount of sequence information available for Chinese hamster related cells has increased over the past years, but additional important fields have not yet been established within the CHO community. One of these fields is the promoter landscape of the Chinese hamster, which is, in contrast to the mouse or human promoter structure, a relatively blank spot on the map. This becomes even more unintelligible given the rise of 2nd and 3rd generation sequencing techniques and advanced lab protocols. Suitable promoters are a fundamental step in cell line engineering due to their contribution to a preferable high transcription level. Additionally, viral promoters in eukaryotic production hosts are more and more superseded by endogenous promoters, thus increasing the demand for endogenous Chinese hamster high yield promoters.

The main goal of this thesis is therefore a first in-depth analysis of the CHO promoter landscape and a list of auspicious promoter candidates which may be used later as basis for high level expression experiments or production systems. Since the last extensive study of promoter architecture in mammalians, several generations of sequencing systems were developed, thus an up-to-date sequencing protocol which exploits recent high throughput techniques is required. Therefore, a specialised dual-library RNA sequencing approach was chosen as data source.

In a next step, a specialised bioinformatics pipeline has to be developed. This pipeline needs to be able to analyse NGS data within the magnitude of human

genome scaled experiments and will be implemented as a single workflow, leading from preprocessing through data analysis to graphical summaries and data output in an easy accessible format. Several analyses, such as transcription start site identification, TSS annotation, promoter analysis on a per gene level and on a genome-wide scale as well as mapping tools for pathway reconstructions have to be featured in the software.

In addition to the new RNA-seq sequencing project a CHO cDNA dataset from an earlier study was available. On the one hand, many of these assembled cDNA sequences did not reach the full length of the original transcript. One the other hand however, genomic sequence data originating from a CHO-K1 cell line is also available. Therefore, a decision to combine both datasets was made in order to obtain new knowledge of possibly non-full length transcripts in terms of transcription start site positions. The process of combining those two datasets is known as targeted assembly, although this niche of bioinformatics software is not yet well developed. Therefore, as a second goal for this thesis, a software, able to perform targeted assembly for large data sets of several gigabases needs to be developed from scratch. In an iterative fashion, cDNAs will be used as seeds, whereas genomic reads act as an extensions of the seed in 5' and 3' direction.

In the following, a summary of the main goals of this is thesis presented:

Chinese hamster promoter analysis pipeline

1. Selection of a sequencing strategy suitable for transcription start site identification in eukaryotic organisms
2. Design and implementation of a bioinformatics pipeline capable of NGS data processing and TSS analyses
3. Assignment of transcription start sites to corresponding genes
4. Promoter analysis performed genome-wide and on a per gene basis

   The final goal is to construct a list of promoters exhibiting very high transcription rates as well as a detailed analysis of known regulatory elements within the promoter region

cDNA extension pipeline

1. Pipeline design must only rely on intrinsic data, since it should be able to work without prior knowledge of the genome
2. The software is not designed as a *de novo* assembly replacement but specifically as a targeted assembly tool
3. Incomplete cDNA sequences acting as seeds need to be extended in 5' and 3' direction Main goal is a set of extended sequences possibly reaching into the 5' and 3' untranslated regions

CHAPTER 4

Targeted assembly with SATYR

4.1 Introduction

In section 2.3 state of the art *de novo* assembly strategies were introduced in depth. However, certain use cases for sequencing data do not require typical *de novo* assemblies, as only specific information is of interest. Examples for this use case include tumour specific DNA changes in cancer cells, detection of common repetitive elements, or observation of gene fusion events [Chen et al., 2012; Peterlongo and Chikhi, 2012].

Due to advances in sequencing techniques (Chapter 2.1) on the one hand and the development of versatile and robust assembly algorithms on the other hand, in many cases sequencing and assembly are just one step in a larger experimental setup. This is especially the case for organisms with relatively small and simple genomes, originating, for example from bacterial lifeforms. Given a reasonable data quality and a decent sequencing setup, assembly of those genomes today is a straightforward task. However, exceptions to this statement exist for instance in form of organisms with unusual high genomic G+C content.

Starting with larger and more complex genomes from yeast & fungi, mammalians, or plants, sequencing and especially the assembly process still require significant time and expertise in order to generate satisfying results. Of course, these tasks may be outsourced to commercial sequencing centres, which in turn puts pressure onto the project budget. Therefore, methods which enable scientists to perform genome assemblies in a more targeted way, thus avoiding the need for resource-intensive *de novo* assemblies of whole genomes may pave the way for

genetic analysis of organisms without the need for large scale sequencing efforts linked with high experimental costs.

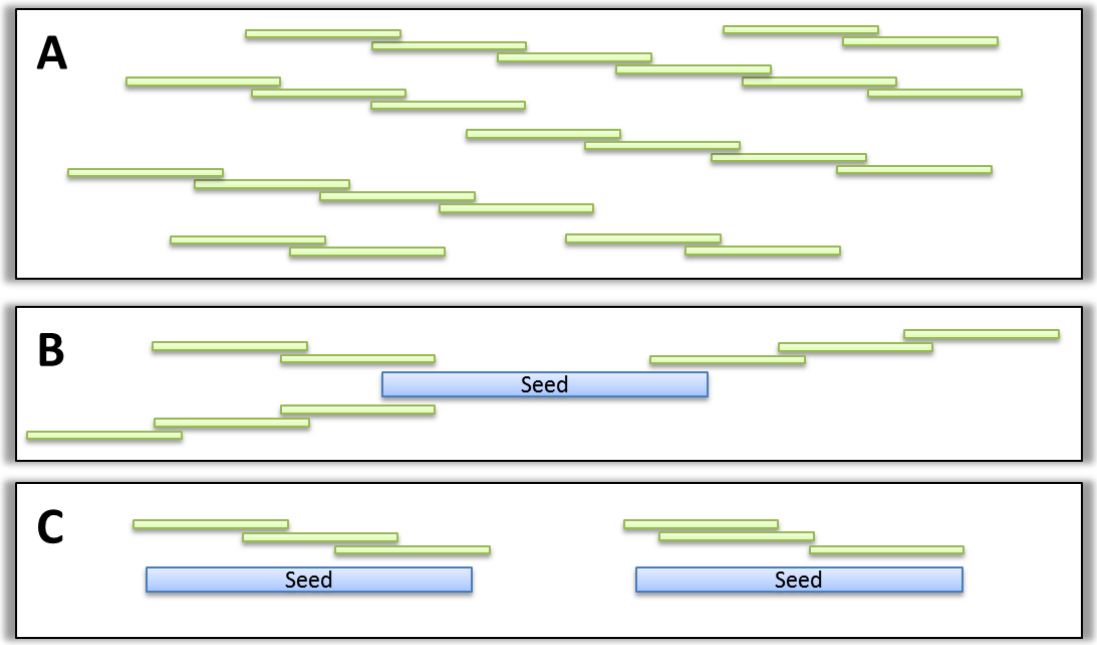


Figure 4.1: Comparison of *de novo* assembly and a targeted assembly approach. **A)** Overlap graph-based *de novo* assembly, where reads (green) with similar ends (5' and 3') are joined into larger contig structures. **B)** In the targeted approach, overlaps are also computed, however, a set of starting points, called seeds (blue) exists in this implementation. Each seed is extended during the assembly process, normally until no further suitable overlaps are found within the read set. **C)** A variant of the targeted assembly approach where only the seed sequences are reconstructed from sequencing reads. This variant has analogies to the read mapping problem.

Targeted approaches are applicable in cases when no reference genome exists, only a fraction of the sequence coverage required for *de novo* assemblies is available, or only specific regions, e.g., several kilobases surrounding a gene, are of interest. In contrast to whole genome shotgun setups, where assembly starts with a set of reads without any reference points, targeted assembly usually uses seed sequences as a starting point for locally limited assemblies (Figure 4.1 A). Both 5' and 3' end of the seeds are compared to the set of sequencing reads in order to find possible overlaps, which is comparable to a localised version of overlap layout consensus assemblers (OLC-based, Figure 4.1 B).

Initially, a collection of 29,184 cDNA sequences originating from the CHO-K1 cell line [Becker et al., 2011] formed the starting point for the development of the targeted assembly software SATYR (**S**eed **A**ssisted **T**argeted assembly of **Y**ield increasing **R**egions). Aim of the tool is the targeted assembly of the mostly missing

5' regions of the cDNA sequences, due to their relevance for transcription regulation. Upstream of the protein coding genes corresponding promoter region may be found. Knowledge of this region may therefore lead to insights into each genes expression, which in turn is of great interest in biotechnological applications, where specific proteins are overexpressed (see Section 2.2.1). Work on this software started in 2010, when no CHO or Chinese hamster reference was publicly available, thus an iterated assembly with genomic sequence reads was conceived. A first Perl-based prototype acted as blueprint for the subsequently developed SATYR software. Sequence data for this new pipeline was provided in form of an Illumina sequencing run on an GAIIx system which was used for whole genome shotgun sequencing of the CHO-K1 genome (see Section 4.4.2.1).

4.1.1 Previous work

The niche of this relatively narrow field has not yet spawned as many software solutions as *de novo* assembly software, which still is a field of very active development, not at least stimulated by continuously evolving sequencing technologies.

SHORTY The SHORTY assembler [Hossain et al., 2009] was a first approach to seed-based assembly. SHORTY uses a small number (5-10) of relatively short seeds (300-500 nt) as a skeleton to guide a *de novo* assembly of ultrashort 35 nt reads, produced by the SOLiD (see Section 2.1.3.3) sequencing system. The software especially exploits paired-end information of reads to accurately estimate inter-contig distances in subsequent assembly steps. Although SHORTY is seed-based, the ultimate goal is a complete *de novo* assembly, which "crystallises" around the given seed sequences. In a first step, reads are stored within a trie data structure [Fredkin, 1960]. A trie is special kind of tree structure for string storage, in which each node stores a certain prefix, starting with one character and increasing in length with each additional level (Figure 4.2 A). This trie is used as central search construct for reads based on kmers to find related mates and their corresponding kmers.

After read storage, reads are anchored onto a seed sequence by read mapping (Figure 4.2 B). Due to the mate information extracted from the trie, the read's mate sequence and approximate distance is known. The mate set obtained by trie lookups is afterwards converted into a contig by overlapping the reads. If these contigs exceed a given threshold, they are used as seed sequences on their own in the next assembly iteration. Using this strategy, a whole *de novo* assembly can be performed. In intermediate steps contigs are screened for repeats, palindromes, and other sequence anomalies which are removed.

SHORTY may be seen as the first approach using seeds to start an assembly process. However, the tool aims for complete *de novo* assemblies of bacterial and viral genomes and the software is not suited for today's sequencing technologies

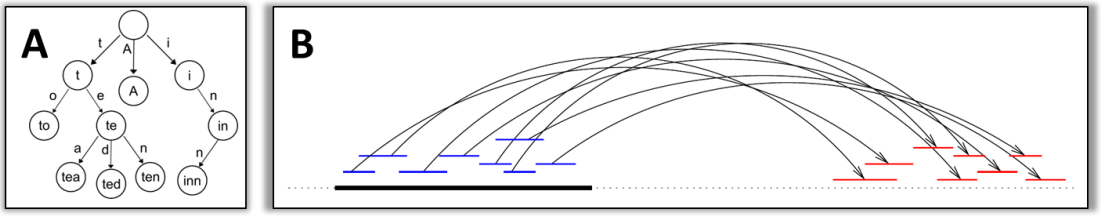


Figure 4.2: Read storage and seed processing within the SHORTY software. **A)** A trie data structure, in this case used for storage of the words "A, to, ted, tea, ten, in, and inn. Strings are stored starting with the first character, whereas each additional layer in the trie structure adds another letter to the node. Therefore, for strings consisting of three letters, three layers of nodes are needed. Graphic modified from [Wikipedia, 2014]. **B)** Overview of seed processing in SHORTY. Reads (blue) mapping to the initial seed (thick black line) have a mate associated, whereas the distance between both mates is known. Overlaps between the red mates are computed, resulting in a new contig. Graphic from [Hossain et al., 2009].

with their increasing read length and throughput. Although targeted assembly with SHORTY is possible with some constraints, namely sequencing technology and read length, it is not the intended use case of the tool, especially when dealing with eukaryotic genomes.

TASR Two years later, TASR [Warren and Holt, 2011], a Perl-based offspring of the well known SSAKE assembler [Warren et al., 2007] was published. The tool is optimised for Illumina data and abandoned the initial aim to reconstruct the whole genome in favour of relatively short areas of interest. TASR receives a list of seed sequences which may be either biologically verified or artificially constructed. The target sequences should be relatively short in order to obtain optimal results. TASR has a fixed parameter of 15 nt long kmers used during the assembly process. While the targets are divided into all possible 15-mers, each occurring kmer is stored with a hash table. In a second step all input reads are screened for these kmers and only reads containing corresponding kmers are retained for the subsequent assembly process.

Compared to SHORTY, TASR pursuits a slightly different approach in focussing on a high quality reconstruction of the supplied target sequences, in other words a locally very limited, targeted *de novo* assembly. The use cases described within the TASR publication therefore focus on detailed analysis of the reconstructed seed sequences to discover fusion transcripts (Figure 4.3) or single nucleotide polymorphisms/variants (SNPs, SNVs) which would potentially be missed during a standard *de novo* assembly. Due to the fixed kmer size of 15 nt, the maximal extension size not covered by seed sequences is limited to (read length - 15) bases. While this limitation does not interfere with the projected use case of the software, which is targeted reconstruction of seed sequences, it significantly limits the possibility to

discover sequence features several hundreds or thousands of nucleotides upstream or downstream of the initial target sequence.

PRICE The PRICE software [Ruby et al., 2013], published in 2013 and implemented in C++, is the latest addition to the group of targeted assembly software solutions. Again, the tool was developed with a set of specific use cases in mind which deviate from classical *de novo* assembly and therefore require for new software solutions.

While TASR is strictly focused on reconstruction of the supplied seed sequences with reads from the input set, PRICE aims for an extension approach, meaning that the seed sequences should be extended in both directions as far as possible. Traditional *de novo* assembly software generally is optimised on two hypotheses, first, sequencing coverage throughout the genome is assumed to be equally distributed and second, all sequencing data originates from a single genome. This is true for most classical assembly use cases, but not for metagenomic sequencing projects, where DNA originates from a pool of organisms. Here, neither a single genome is sequenced nor is the sequence coverage equally distributed. Generally, the coverage is also not equally distributed throughout the group of organisms which were sequenced. Due to these restrictions, classical *de novo* assemblers are not well suited for metagenomic assemblies [Peng et al., 2011] and specialised versions of established tools like Velvet [Zerbino and Birney, 2008] and IBDA [Peng et al., 2010] were developed in form of Meta-Velvet [Namiki et al., 2012] and Meta-IBDA [Peng et al., 2011].

The paired-read iterative contig extension (PRICE) software is tailored to deliver assemblies of specific, underrepresented organisms within the DNA mixture of a metagenomic sequencing setup, for example viruses or pathogens. As the name suggests, it is especially developed to exploit paired end data, as overlapping paired end reads can be combined into a longer single reads, thus reducing the number of required overlap computations by up to 50% (assuming all paired end pairs exhibit sequence overlaps).

The general workflow of PRICE starts with the mapping of reads to seeds or existing contigs (Figure 4.4 A, Step 1), followed by a localised assembly of 5' and 3' extensions of the seed sequences (Figure 4.4 A, Step 2). The last step involves the construction of scaffolds from potential overlaps of extended contigs (Figure 4.4 A, Step 3) and closes with a removal of spurious extensions from the assembly. These steps are iterated as many times as desired until the genome is assembled or no further extending reads are found. Paired end reads are of special meaning during the assembly process, as the they may be able to extend the seed sequence for short insert values (Figure 4.4 B, green arrows) or may be employed to establish

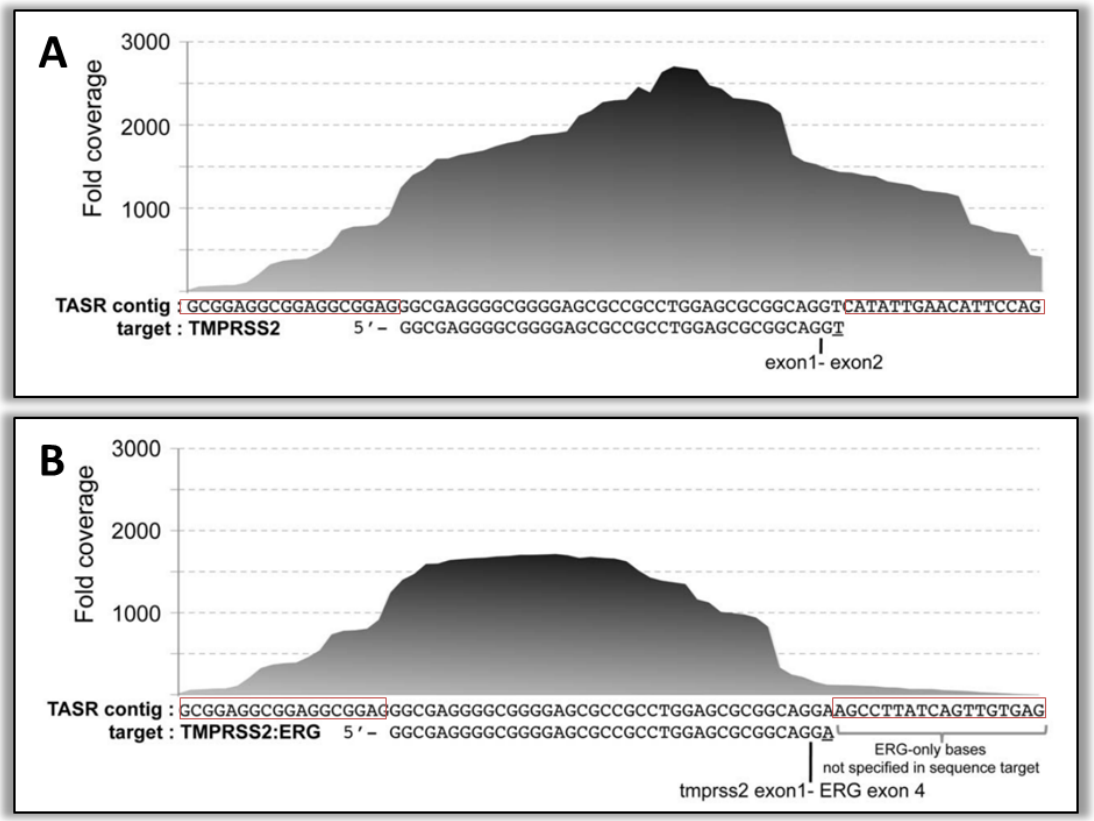


Figure 4.3: Two contigs reconstructed by TASR. Bases of the assembled DNA sequence are shown on the x-axis, read coverage is shown on the y-axis. The seed sequence is centred under both contigs, assembled sequences extending the initial seed are marked with red boxes. Both assemblies only provide short glimpses into neighbouring sequence areas, which are no longer than 20 nt. **A)** The original seed sequence, showing the linkage between exon 1 and exon 2 of TMPRSS2. A slight decrease in coverage is observable at the transition from seed sequence to newly assembled sequence. **B)** The seed sequence was altered by introducing a A at the 3' end (underlined), therefore yielding a different set of spanning reads, which confirm a fusion of the TMPRSS2 exon 1 to ERG exon 4. The coverage drop between seed and extended sequence is much pronounced as in the previous example, thus showing that those fusions account for a small amount of altered transcripts. Image modified from [Warren and Holt, 2011].

the correct order of contigs among each other for long insert values (Figure 4.4 B, yellow arrows).

Mapsembler Mapsembler was the first tool specifically designed for seed-based and targeted *de novo* assembly and was published in 2012 by Peterlongo and Chikhi [2012]. The tool uses given seed sequences only as a starting point for the assembly process, but does not try to reconstruct the seed sequences from reads of

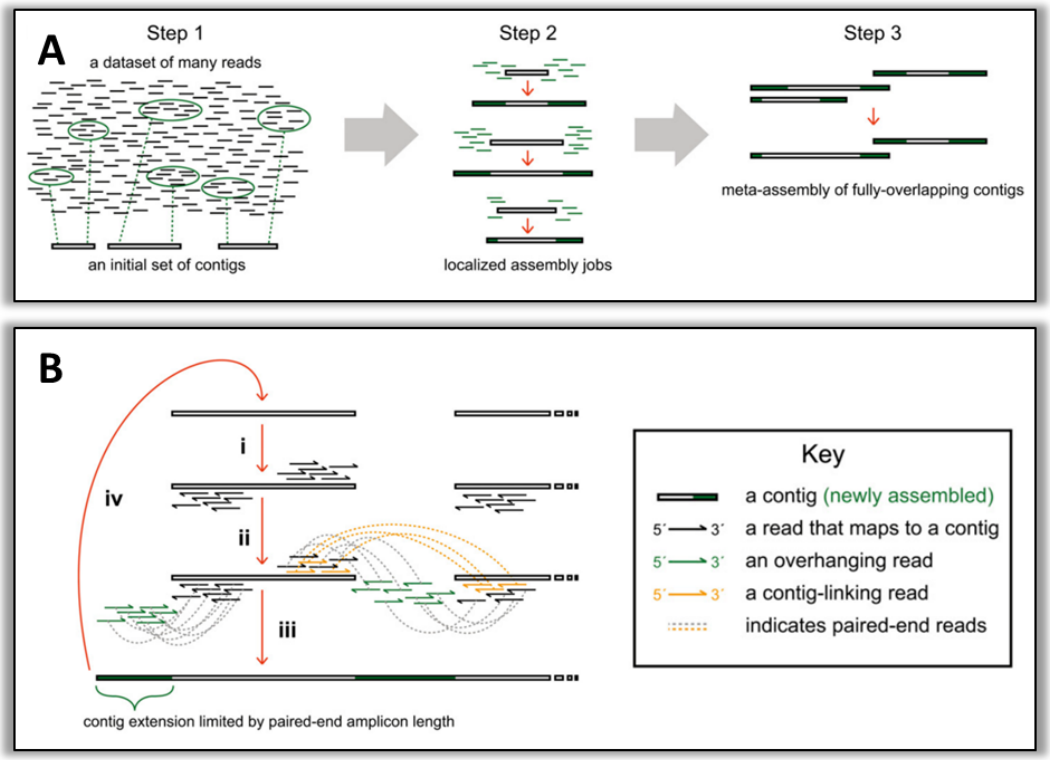


Figure 4.4: Overview of the assembly strategy employed by PRICE. **A)** Step 1) For each seed sequence, overlapping reads from the set of input reads are detected. Step 2) Local assembly joins reads extending a common seed into a specific direction into a new contig. Step 3) The new contigs are scanned for overlaps and - if possible - merged. **B)** Detailed view of steps 1 & 2 from Figure A). Paired end information of reads is used to anchor neighbouring contigs (yellow). Image modified from [Warren and Holt, 2011].

the input data set. Therefore, only 5' and 3' end of the seed sequences are extended as far as possible in order to obtain novel upstream and downstream sequence information (Figure 4.5 A). With clever designed target sequences however, similar results compared to those of TASR can be obtained.

The published version of Mapsembler does not employ fixed indices, which may be saved from one run to another. Instead, during preprocessing, all occurring seed kmers are stored in a runtime index, together with the position of the occurrence. All available reads are screened for according kmers and are only retained, if any of the read kmers is found within the seed index, thus indicating a possible overlap.

In contrast to TASR, the only parts of the seed sequence which are reconstructed during the extension phase are the initial 5' and 3' ends which are needed to anchor a first set of reads onto the seed. The software also features a majority vote-based error correction, known from earlier *de novo* assembly software [Zerbino and

Birney, 2008], to account for sequencing errors within the input reads (Figure 4.5 B).

After error correction, possible overlaps are written into a graph structure, as not every extension step is unambiguous and a simple FASTA file would not be able to store different assembly branches. Nevertheless, Mapsembler allows to abort the extension phase if a branching is detected, thus only exporting confirmed extension sequences. The new sequence ends processed in this iteration again are the starting point for overlap detection in the following step. This process is repeated until no further extensions are possible.

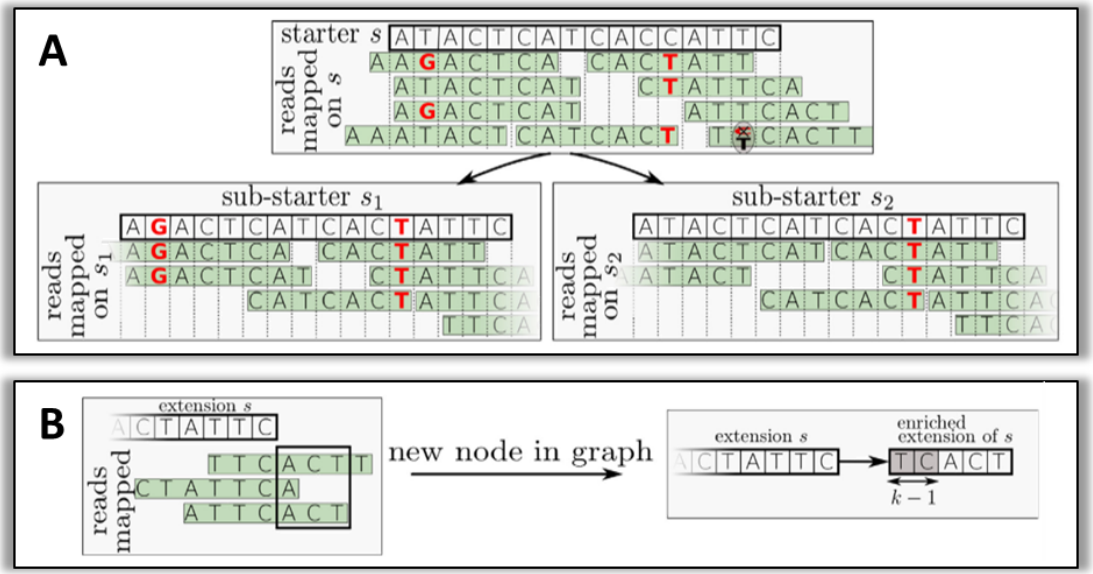


Figure 4.5: Two different stages within the assembly process of the Mapsembler software. **A)** Reads are positioned above the initial starter sequence s. Red bold letters indicate read bases deviating from the reference sequence (the seed). In the lower right a base of a read is processed by error correction. Due to two different possible variations indicated by input reads, two sub-starters with the consensus sequence of aligning reads are created. **B)** Three reads are forming an extension (`ACT`) of the seed sequence. The single overhanging 'T' is not stored since the minimal coverage is two. Image modified from [Peterlongo and Chikhi, 2012].

Both tools TASR and Mapsembler share that they work in an iterative fashion, that minimizes memory usage and allows for execution on standard hardware such as laptops. However, program runtime increases especially for large datasets originating from eukaryotes and mammals as the typical amount of reads used in such setups usually reaches higher orders of magnitude compared to prokaryotic sequencing experiments.

Changes have been introduced into the current development version of Mapsembler, also indicated by the new major version number. Mapsembler2 has not been published by now, but is accessible through the tool's homepage[13]. In order to improve performance for larger datasets, the memory-minimizing strategy employed in the first version was exchanged for fixed read indices, which are written on disk for later reuse. Additionally, the assembly backend now uses data structures from the Minia de Bruijn graph-based short read assembler developed by Chikhi and Rizk [2013]. In essence, these changes transforms Mapsembler2 into a locally limited version of the Minia *de novo* assembler.

4.1.1.1 Conclusion

In the preceding section different software solutions related to targeted and seed-based assembly were introduced. Most of the presented tools however aim at very specific questions and are of limited use only for the seed-and-extend approach outlined in Section 4.1 and Figure 4.1 B. An overview of all existing solutions is summarised in Table 4.1 in form of a feature matrix. As the comparison suggests, Mapsembler is the only tool out of the very narrow field of published software solutions to address the exact issue of targeted assembly as defined in this work, although some performance bottlenecks remain. Thus, Mapsembler in its second, unpublished version was chosen as a suitable reference for benchmarks and evaluation of the software solution developed throughout this work in order to deal with targeted assembly on state of the art sequencing data scale although later evaluations will reveal that these bottlenecks become a serious issue for very large sets of reads.

4.1.2 Software requirements

As stated previously, one of the most important points to consider during the development phase of a new software, aiming at targeted assembly is data handling and especially the handling of several tens or hundreds of gigabases produced by today's sequencing systems. The new software should be able to work on such datasets within a reasonable time frame. The seed sequences required for a seeded assembly approach should not be limited in any way, neither in number nor in the maximal seed length. As for competing software no assumptions can be made about origin of the seed sequence, thus either mRNA, genomic sequences or artificial constructs have to be accepted. Two distinct *modi operandi* exist for targeted assembly software, namely reconstruction (Figure 4.1 B) or extension (Figure 4.1 C). The focus of this implementation clearly is the extension of seed sequences into previous uncharted 5' and 3' regions with the help of assembled reads, thus reconstruction of the seed sequence itself is of lesser interest within this work and will not

[13] http://colibread.inria.fr/mapsembler2

Software	*Targeted*	*Seed-based*	*NGS capable*	*Seed extension*	Reference
SHORTY	✓	✓	–	✓	[Hossain et al., 2009]
TASR	✓	✓	✓	–	[Warren and Holt, 2011]
PRICE	✓	✓	⊖	✓	[Ruby et al., 2013]
Mapsembler	✓	✓	⊖	✓	[Peterlongo and Chikhi, 2012]

Table 4.1: Feature matrix of selected targeted assembly tools. ✓: feature available, ⊖ : feature available with limitations, – : not available. SHORTY accepts SOLiD data, but is not able to deal with amount of data produced by state of the art Illumina high throughout systems. TASR focuses on reconstruction of seed sequences and only performs minimal extension of the seeds. PRICE is specialised on metagenomic datasets and further specialised on assembly of low abundant specimens. Mapsembler in principal is NGS data capable, but shows performance bottlenecks for human genome sized projects.

be implemented. It is reasonable to assume that running the program with different parameters and different sets of seed sequences are common use cases. In conjunction with the already mentioned amount of data it is wise to facilitate a disk based index structure to store indices constructed during the program runtime for later use. Indeed, creation of these data structures consumes significant compute and memory resources, which, however amortises after several program calls depending on input data size. Quality of the assembled extensions should be comparable to existing *de novo* algorithms to depict a detailed, biologically as correct as possible version of the sequence area surrounding the seed. The length of the extension should be user controllable or, if no value is given, only be limited by the availability of suitable reads in the given input read set. In most cases, more than one possible order of reads will represent possible extensions. In such cases, additional information such as (average) coverage, read errors or specific sequences features should be considered to discard unsuitable forks within the extension graph. This requires that possible extensions are stored in graph structures, implemented with adequate, possible external libraries. After extension has finished, the most probable path within the assembly graph should be constructed and stored in form of FASTA files for further usage, whereas the graph should be saved in a standardised graph file format. For performance and portability reasons, the program should be implemented in C / C++ and only use standardised libraries. As the vast major-

ity of bioinformatics tools the software should run on Linux and other Unix-based systems like Solaris.

Overview: Requirements

- Next generation sequencing data capable
- Targeted and seeded assembly
- Reusable, disk based index structure
- Extension sequence quality comparable to established *de novo* assemblers
- Scalable, portable, and performance orientated implementation

4.2 Principal software design and methods

Module	Description
`I/O`	Reading and verification of seed and read input files in FASTQ, FASTA, or SEQ format.
`Utility`	Provides a multiplicity of helper functions used by other modules, such as Hamming distance calculation or sequence conversion function
`Storage`	Storage facility for read data. Burrows-Wheeler transform based, exchange layer between application and BEETL library
`Index`	Provides index functions (`locate`, `count`, `extract`) to the BWT transformed reads through the `Storage` module.
`Extension`	Implementation of the iterated assembly algorithm used to extend seed sequences in 5' and 3' direction

Table 4.2: Overview of SATYR's core modules and their specific functions within the software.

SATYR was developed from scratch in a modular manner and the functionality was divided into five main modules, namely Input/Output module, utility module, extension module, storage module, and index module (Figure 4.6 and Table 4.2). Additional functionality was added through three libraries, the Standard Template Library (STL) [Plauger et al., 2000], the Boost library [Siek, 2002; Schaeling, 2011],

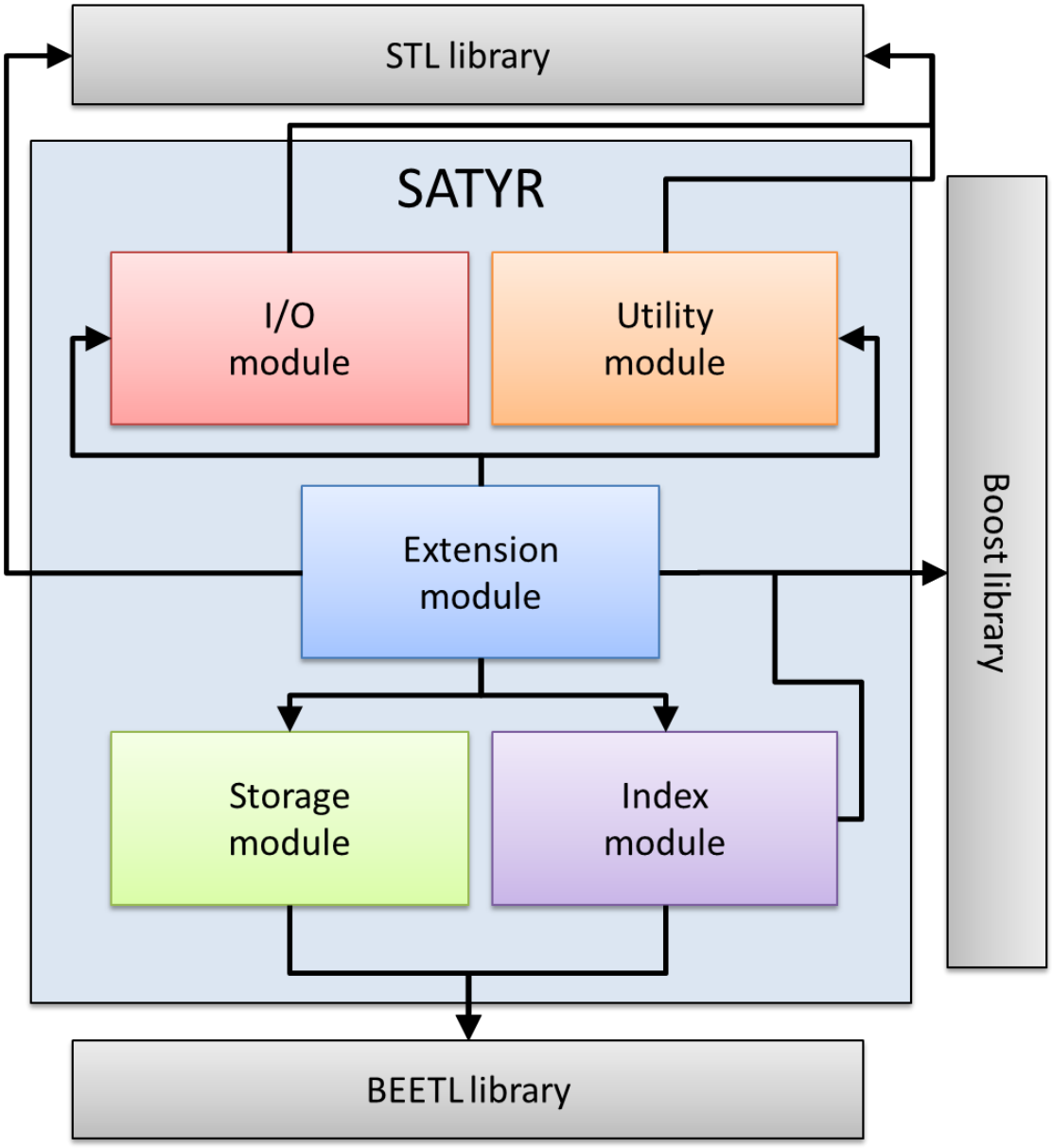


Figure 4.6: Simplified depiction of the internal design of SATYR. Five major modules are shown in red, orange, blue, green, and purple inside the light blue box indicating the functionality of SATYR. External libraries are shown as grey boxes separated from SATYR modules. Connecting arrows in black indicate relations and access patterns between modules.

and the BEETL library [Cox et al., 2012a] (Figure 4.6, surrounding grey boxes and Table 4.3). The software project is implemented in C/C++ and controlled by a standard Makefile, allowing for compiling and running on all Linux-based systems as well as Solaris. The program's source code together with supplementary scripts for analysis and evaluation is available on request through a Mercurial repository[14] hosted on the Bielefeld University Bioinformatics Server (BiBiServ)[15]. The software is purely command line-based and features no graphical user interface, since its normal use will more likely include server systems without connected displays rather than desktop or laptop systems.

Library	Description
STL [Plauger et al., 2000]	Includes a large set of useful functions and data structures (e.g. `vector`, a static contiguous array or `deque`, a dynamic contiguous array)
BEETL[16] [Cox et al., 2012a]	Implements a Burrows-Wheeler transform for sequencing read sets and provides additional compression backends
Boost[17] [Schaeling, 2011]	A large collection of libraries, reaching from multithreading over graph representation to unit testing, currently contains over 80 libraries

Table 4.3: Overview of external libraries employed by SATYR.

4.2.1 Software workflow

Initially, SATYR is supplied with a list of files containing sequencing reads and another file containing seed sequences which are used as start points for the assembly process. Two distinct work flows, the first one dedicated to read processing, the second one responsible for seed processing are initiated, given all command line parameters were verified and accepted, otherwise meaningful error messages are issued. For read files, the Burrows-Wheeler transformation process is initiated if no previously created BWT is found on disk and subsequently loaded into RAM for quick access (Figure 4.7, green coloured steps). Seed sequences are processed differently, since in contrast to the sequencing reads only fragments of the seed's sequence are of interest. Therefore 5' and 3' ends corresponding in length to the

[14] ssh://hg@hg.cebitec.uni-bielefeld.de/bioinfo/satyr
[15] http://bibiserv.techfak.uni-bielefeld.de
[16] http://www.github.com/BEETL/BEETL
[17] http://www.boost.org

chosen kmer value are extracted from the seed and stored in memory. These initial kmers are used as start point for iterative assembly to find reads of similar sequence (Figure 4.7, red and orange coloured steps).

The second phase of the approach is dominated by the iterative assembly (Figure 4.7, central grey box). In a first step, reads including kmers previously extracted from the seeds are located through the BWT index. Suitable reads are reconstructed to their full length using the in-RAM BWT read set, while to short or error-prone reads are discarded. Remaining reads are grouped and temporary consensus sequences corresponding to potential ambiguous extensions (forks within the assembly graph) are generated. For each of these possible extensions, sequences of length k are extracted from 5' or 3' end to server as start points for the next iteration of the assembly process. It is either possible to carry on extension as far as possible or to supply a fixed number of iterations after which extension should cease.

Once the limit has been reached or no further extension are possible, a final consensus has to be generated. To retrieve the consensus each node along a path from the last added node back to the root of the assembly graph has to be visited and its DNA sequence has to be recovered from BWT-index. Afterwards overlapping sequence stubs are removed and sequences are concatenated until a single consensus sequences emerges. However, there is no guarantee that this consensus is the correct one, since forks within the assembly graph introduced by sequencing errors or repetitive regions interfere with the graph traversal step.

The workflow finishes with the export of assembled regions as FASTA files, separated into 5' extension, seed, and 3' extension. Additionally to the sequence output files the complete assembly graph with coverage information for each node is exported as GraphViz-compatible[18] ".dot" file. These files can be visualised by a broad range of tools on all operating systems while also allowing for different graph layouts. The log file generated by the software contains supplementary statistics of the assembly, including average extension length, maximal length, number and ID's of non-extended seeds as well as other runtime metrics.

4.2.2 The Burrows-Wheeler transform

4.2.2.1 Introduction

More than 20 years ago, during the mid 1990s, portable music was most commonly found in form of compact disc players or still in form of cassette-based walkmans. However, these two gimmicks vanished a few years later, starting in the early 2000s and were replaced by much smaller players with internal hard disks, able to carry hundreds of songs instead of the 90 minutes of a standard CD. But how was is

[18] http://www.graphviz.org

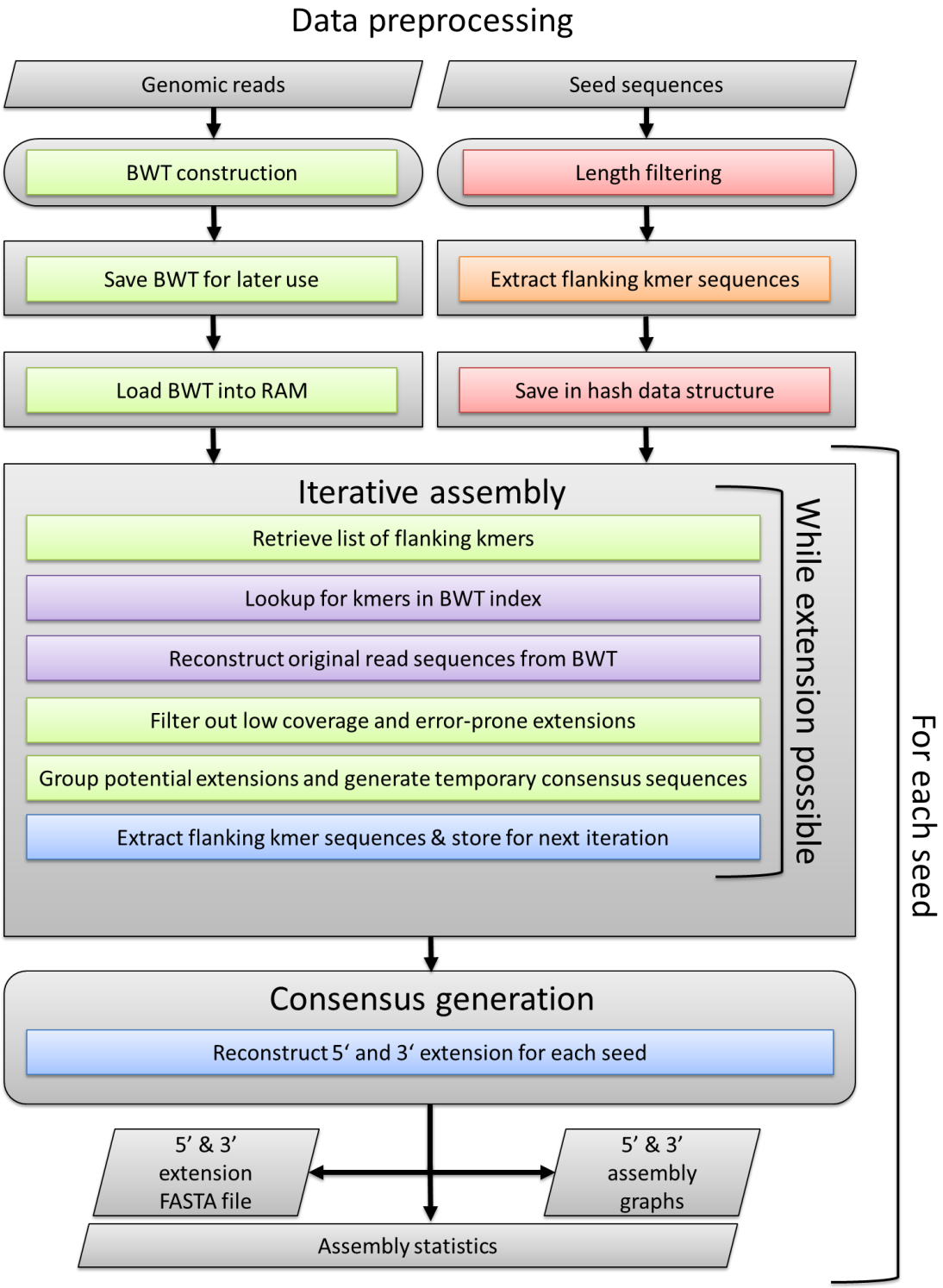


Figure 4.7: Classical flow diagram of SATYR. Start steps are indicated as rectangles with rounded corners, data input/output is depicted as parallelogram. Each processing step is shown as rectangle, whereas the flow direction is indicated by black arrows. Each step is colour coded to match the module's colour shown in Figure 4.6.

.

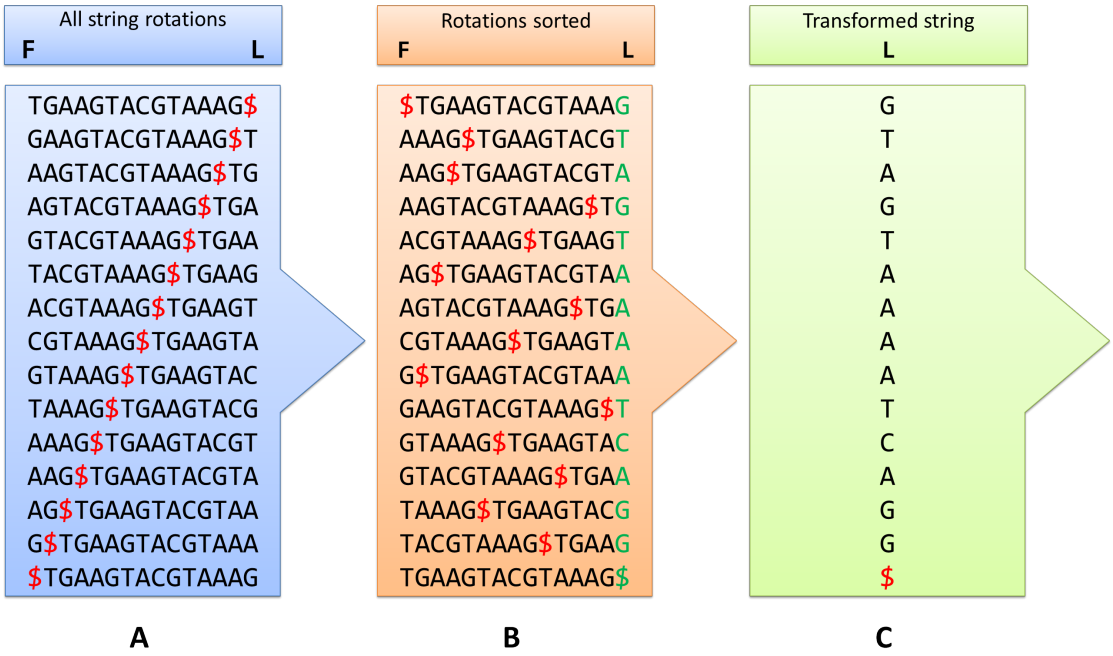


Figure 4.8: The naive approach to the Burrows-Wheeler transform, divided into 3 steps. The sentinel character \$ is marked red. **A)** Input string of length n is written in the first row of a matrix, the remaining $(n-1)$ rows are filled with cycled rotations of the string. **B)** Rows of the matrix are sorted lexicographically. **C)** The transformed string can be read in the last column of the sorted matrix.

.

possible to fit these amounts of audio data onto a small hard disk?

The answer to this question, of course, is the MP3 (MPEG-2 Audio Layer III) file format, which is able to compress audio data to a fraction of its original size. While MP3 is a lossy compression algorithm, meaning that part of the information of the source is lost during compression, there are also lossless compression methods, used e.g. to compress files to send them as email. The bzip2 format[19], first released in 1996 is an example of lossless compression and itself based on the Burrows-Wheeler transform (BWT). This text transformation, originally developed in 1983 but not published before 1994 [Burrows and Wheeler, 1994] does not change the size of the input data, but instead permutates the order of characters in such a way, that similar characters are more likely to follow each other, making them easier to compress [Adjeroh et al., 2008].

[19]http://www.bzip.org

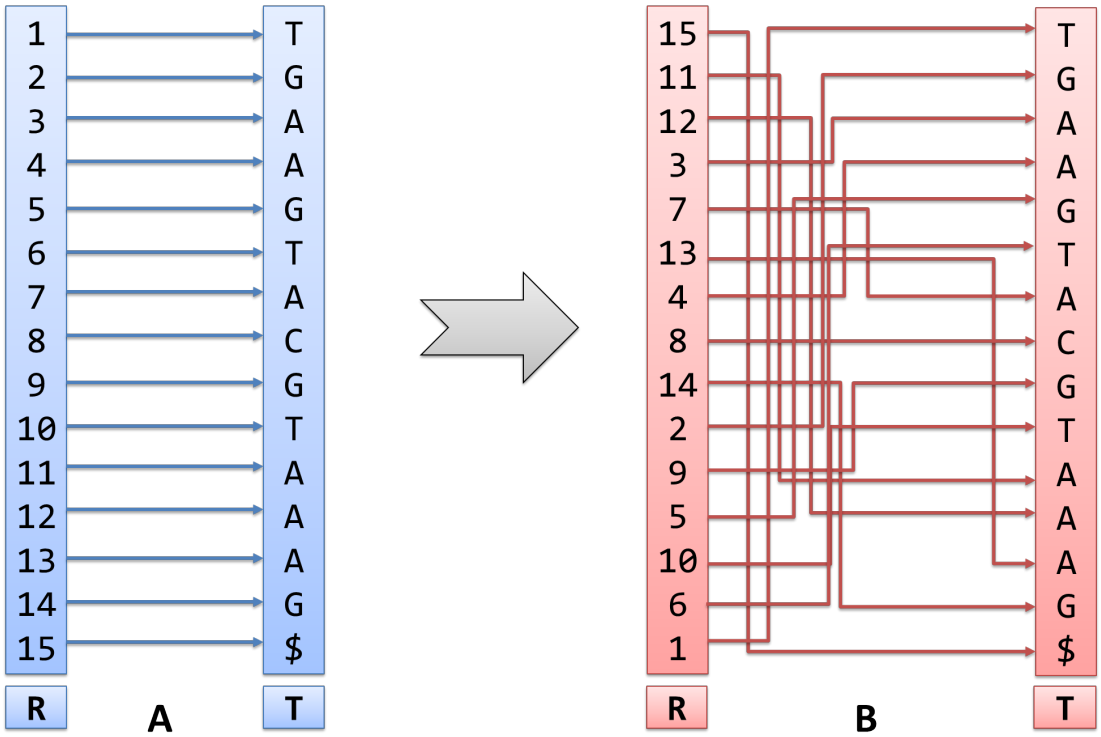


Figure 4.9: Reference based sorting of substrings during the second step of BWT generation. **A)** Initialisation of R with numbers from $1 \ldots n$. **B)** Assignment of substrings in relation to the original sorting.

4.2.2.2 Encoding

The process of text transformation from its source form to the BW-transformed string is also called encoding. This encoding takes three steps and, in this special case is limited to the alphabet of DNA, therefore $\sum = \{\$, \mathrm{A}, \mathrm{C}, \mathrm{G}, \mathrm{T}\}$, whereas \$ is used as a sentinel character to mark the end of the string[20]. Additionally the sentinel \$ is defined to be lexicographically smaller than A,C,G, and T. The sample string used throughout this section is "TGAAGTACGTAAAG$" and the Burrows-Wheeler transform starts by writing this string of length 15 into the first row of a 15×15 matrix. The next 14 rows of the first matrix are filled with cyclic rotated versions of the source string, each shifted by one character as distinguishable by the pattern created by the red sentinel character (Figure 4.8 A).

Within the second step, the cyclic rotated string of the first matrix are sorted by their lexicographical ordering (Figure 4.8 B). The BW-transformed string can be read from the last column L of the sorted matrix (Figure 4.8 C). While the naive approach shown may be suitable for understanding the algorithm, its complexity is

[20]The sentinel character is not mandatory for the basic BWT implementation, however, the sentinel helps to visualise the BWT process and is actually used in the implementation of the BEETL library.

not recommend for implementation. Since n strings of length n have to be compared to each other, $\mathcal{O}(n^2)$ space is required for this version [Adjeroh et al., 2008]. Since for large texts the space requirement can quickly become infeasible an improved version using only a reference based array R was proposed. The array is initialised with entries from $1 \ldots n$ (Figure 4.9 A) corresponding to the original ordering of substrings in the input text. Subsequently the R array is sorted. The transformed string, previously read from the last column L now can be extracted using the character at the specific position, e.g. $T[R[i]]$. The usage of a single array reduced the space and time complexity down to $\mathcal{O}(n)$, the time complexity for sorting itself may be assessed with $\mathcal{O}(n \log n)$ given a quicksort-based sorting method. The estimate might however be inaccurate, due to already sorted input text, yielding $\mathcal{O}(n^2)$ worst-case complexity for quicksort [Adjeroh et al., 2008]. To avoid this pitfall, Burrows and Wheeler [1994] proposed a modified quicksort algorithm which employs a preceding radix-sort to identity out potential repeating text.

4.2.2.3 Decoding

One of the main reasons the Burrows-Wheeler transform is so popular is its reversibility. The decoding process requires more steps than the initial encoding but again features an elegant and simple algorithm. Decoding starts with nothing more but the BW-transformed string which corresponds to the L column of the BWT matrix (Figure 4.10 A). Since the in the second encoding step all cyclic rotations of the input string were sorted lexicographically, in turn, the first column, F, must consist of all characters of the input string (or the transformed string) in sorted order (Figure 4.10 B). Due to the cyclic rotations during encoding, the strings of the BWT matrix wrap around at the end of the row, therefore a character in a specific row of column L must be followed by the character in column F. The contents of column F are also known, since F must contain all characters of the input string in sorted order. Now, columns L and F are combined, therefore creating a list of tuples (Figure 4.10 C). The sorting of this list of tuples yields reconstructed column 2 (Figure 4.10 D). This process is iterated until the matrix is completely filled and in each round the existing columns are prefixed with the characters from the fixed L column (Figure 4.10 E & F). Once the matrix is filled, the source string can be read in the first row of the matrix (Figure 4.10 G). Although the presented algorithm is an evidence for the simplicity of this approach, it is also clear that, due to the used matrix the space complexity of the algorithm is $\mathcal{O}(n^2)$. This however, renders the approach infeasible for very large strings such as genomes.

The naive algorithm however can be improved by the addition of several auxiliary array structures holding a number of indices to reduce the overall space complexity and to speed up decoding for multiple strings [Adjeroh et al., 2008]. As stated, the naive approach retains the complete matrix with all cyclic rotated and sorted input strings, therefore $\mathcal{O}(n^2)$ space is required. A logical step would be the removal of the complete matrix and the introduction of smaller auxiliary arrays. When

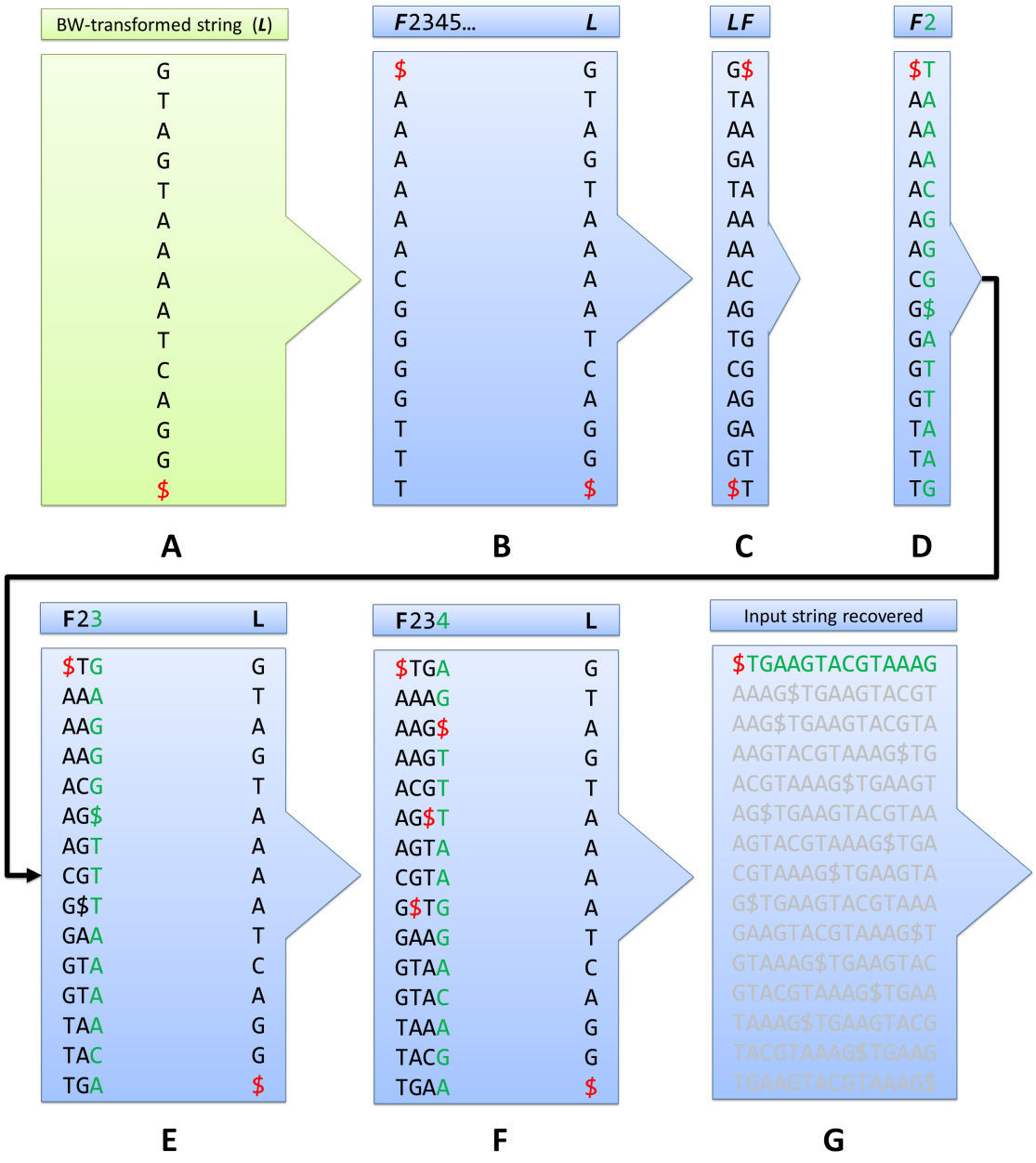


Figure 4.10: Naive decoding algorithm for a BW-transformed string. **A)** The encoded string as read from the L column during the last step of the encoding process. **B)** Column F can be constructed from column L by simple lexicographical sorting. **C)** Contents of column 2 are revealed by creating a list of pairs from column L and F. **D)** Sorting this list again lexicographically, leads to the reconstructed column 2 when reading the second character of each tuple. **E) & F)** In each iteration a new character is added to the reconstructed string, the principle of concatenating columns and sorting them as lexicographically as in D) remains the same. **G)** The reconstructed string can be read from the first line of the matrix after the last column has been reconstructed.

inspecting column F it becomes clear, that the information content of the column could easily be reduced by counting the number of different characters as well as their first occurrence within the sorted column. This modification of the algorithm replaces the $\mathcal{O}(n)$ column F by two new matrices, M and K of $\mathcal{O}(|\sum|)$ size each. The remainder of the initial matrix is not needed, only a temporary array Q used to store the intermediate result after each iteration and the column L containing the transformed string both of size $\mathcal{O}(n)$ have to be kept additionally.

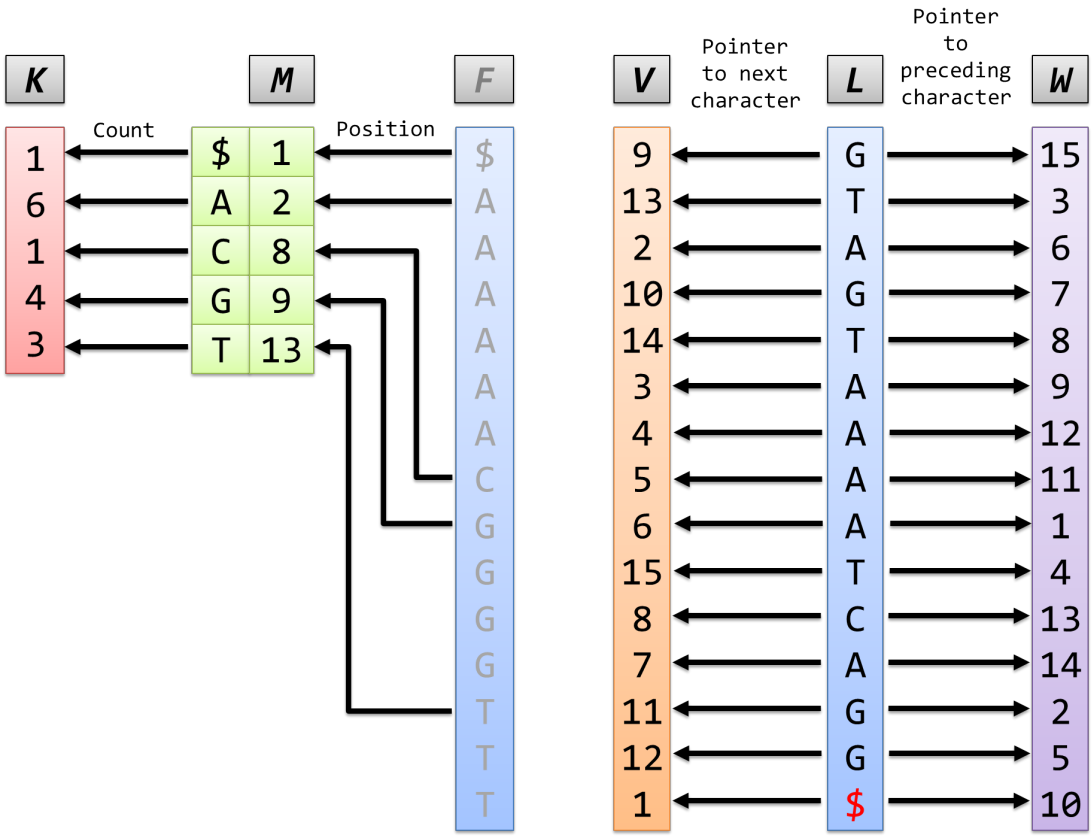


Figure 4.11: More complex implementation of the reverse BWT, including several additional arrays used to lower the overall memory consumption as well as run time. Column ***F*** may be replaced by a combination of array ***M*** which records the start position of a new character in ***F*** and ***K***, holding the total count for each character of the alphabet. This is possible due to the lexicographical sorting of ***F*** and the fact that is contains all characters of the source string. The original matrix used in Figure 4.8 is not needed for this approach, as well as column ***F*** (displayed greyed out). Especially in case of multiple decoding calls, arrays ***V*** and ***W*** become important, as they represent a mapping between ***F*** and ***L*** in forward and reverse direction and allow for backward or reverse decoding. Both arrays are constructed with help of ***M*** (shown here in its initial state). ***M*** is modified during the construction.

As the space complexity could be reduced, a next step aims for a speedup of the actual decoding process. It should be noted that with the current setup the

Algorithm 4.1 Construct V and W from M and L [Adjeroh et al., 2008].

```
1: for i = 1; i ≤ n; i++ do
2:     V[i] = M[L[i]]
3:     W[M[L[i]]] = i
4:     M[L[i]] = M[L[i]]+1
5: end for
```

source string is computed in reverse owed to the naive approach. While, especially in bioinformatics context reverse versions of strings are not unfamiliar, a possibility to decode in forward direction would significantly improve the algorithm. For this reason Burrows and Wheeler [1994] added two further auxiliary arrays to the approach, V and W. While $V[i]$ holds a pointer to the value of the character following a specific character $L[i]$, $W[i]$ contains a pointer to the character preceding $L[i]$. Both columns can be constructed in linear time in one pass using Algorithm 4.1. Due to the pre-computation of V and W it is possible to seamlessly decode a BW-transformed string into both directions by simply following the pointers in those two arrays. Therefore, these arrays may be understood as a kind of cache which speeds up decoding of multiple instances of strings. The overall space requirements can be summarised as follows:

$$\mathcal{O}(\overbrace{(n)}^{\text{Arrays C \& K}} + \underbrace{\left|\sum\right|}_{\text{Arrays L, C, V, \& W}})$$

4.2.2.4 Counting and locating patterns

Motivated by the possibility to efficiently encode and decode texts using the Burrows-Wheeler transform different approaches were published, aiming at the use of the BWT as an (compressed) full text index useful for pattern matching. Ferragina and Manzini [2000] developed methods to count the number of occurrences of a pattern p within a compressed text T in linear time and to exactly locate the position of each hit within the original text. Their proposed index structures was coined "Full-text index in Minute space" or FM-index for short [Ferragina and Manzini, 2005].

Counting patterns in the BWT Counting of hits is performed using the BW_search function (Algorithm 4.2) which depends on an auxiliary function `Occ`. It is remarkable that the only input required for counting and location are a pattern P and column L containing the transformed string. The helper function `Occ(c,1,k)` returns the number of occurrences of character c in the prefix $L[1, k]$ in constant time. Therefore the total time required to count the number of hits for a pattern is linear with respect to its length. The idea behind BW_search is to narrow down the number of auspicious suffixes (Figure 4.12, left) in each iteration which is achieved

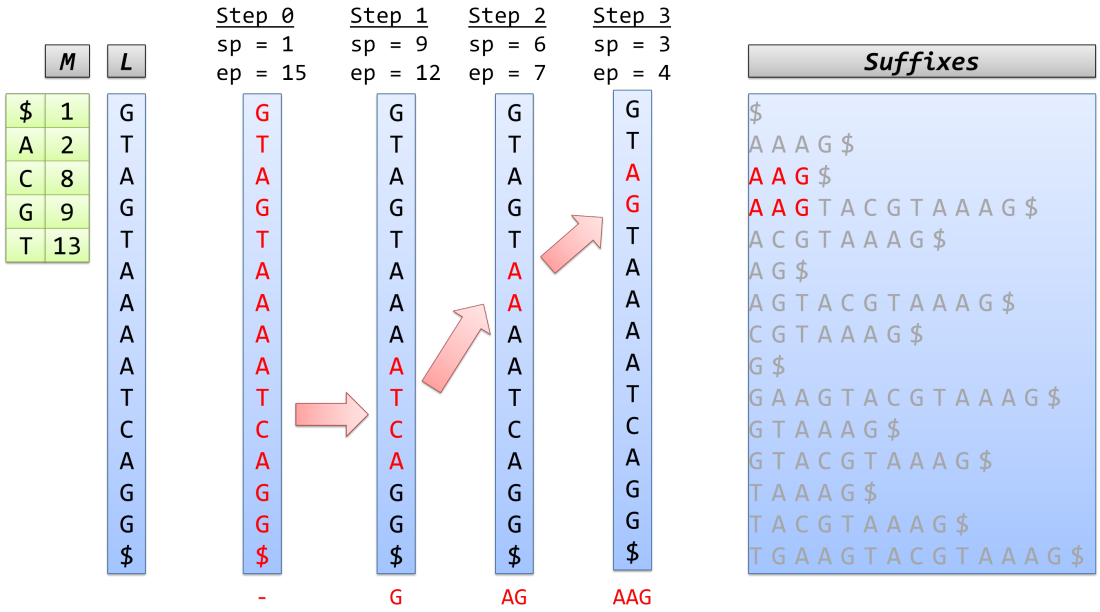


Figure 4.12: Stepwise strategy of the BW_search algorithm (Algorithm 4.2) for the BW-transformed string of "TGAAGTACGTAAAG$" and pattern "AAG". Columns ***M*** and ***L*** are given as input. In each of the four steps the pattern is extended by one character from right to left (therefore backward search). For each step the portion of characters marked in red indicates the position of the *sp* and *ep* pointers employed by the algorithm. Step three is the terminating step due to the length of the pattern, yielding a hit count of $(4 - 3 + 1) = 2$.

through two pointer variables *sp* and *ep* for the first occurrence of the suffix and the last occurrence of the current suffix. The function is terminated when either the *ep* pointer passes the *sp* pointer, indicating that the query pattern is not part of the BWT string or if the iteration counter reaches the length of the query pattern. In case the pattern is found the number of hits can be derived directly from the distance between the *sp* and *ep* pointers, as each row of the virtual BWT matrix comprises a suffix starting with the pattern in question.

Locating patterns in the BWT While previously the check for existence of a specific suffix and the corresponding count of hits was the focus, the locate operation returns the position within the original string for a chosen index of the L column (Figure 4.13). To allow for a quicker search process, a position mapping from L to T is constructed, in order to provide a balance between speed and memory requirements only a fraction of the positions of L are included into this mapping, for example each 1000th position in L. In case the BWT position of question is such an index, the position in T can directly be returned. Otherwise, a detour via the next nearest index is necessary. To reach the nearest indexed entry of L the

Algorithm 4.2 BW_Search
Counts the number of occurrences of pattern $P[1, p]$ in a BWT text $T[1, u]$ [Ferragina and Manzini, 2000].

```
 1: c = P[p]
 2: i = p
 3: sp = M[c] +1                          ▷ First occurrence of initial 1-letter suffix
 4: ep = M[c+1]                           ▷ Last occurrence of initial 1-letter suffix
 5:
 6: while sp < ep && i > 2  do
 7:     c = P[i-1]
 8:     sp = M[c] + Occ(c,1,sp-1) + 1                          ▷ Update sp pointer
 9:     ep = M[c] + Occ(c,1,ep)                                ▷ Update ep pointer
10:     i = i - 1
11: end while
12:
13: if ep < sp then
14:     return 0                                               ▷ No hits found
15: else
16:     return (ep − sp + 1)          ▷ hits = number of suffixes between sp and ep
17: end if
```

LF-mapping property is used, i.e. the i-th character c in L corresponds to the i-th character c in F.

4.2.2.5 BWT on multi sequence sets

The previous functions and properties of the BWT are by now only defined for one string T, for example a book or, in bioinformatics context, a genome. By employing the presented functions, it is possible to construct read mapping algorithms able to align millions of reads against very large compressed and indexed reference genomes. Well established tools include BWA [Li and Durbin, 2009], SOAP2 [Li and Durbin, 2009] and Bowtie2 [Langmead and Salzberg, 2012]. In various use cases it might however be of interest to be able to transform a set of n sequences $S = \{S_1, S_2, S_3, \ldots, S_n\}$ instead of a single sequence. In this context the sentinel character \$, introduced in Section 4.2.2.2 changes its relevance from an optional character used for presentation purposes to an integral part of the concept. First ideas to extend the classical BWT to an extended BWT of string sets were published by Mantaci et al. [2005] and previous work by Gessel and Reutenauer [1993] on permutation count in cyclic structures was used as foundation for the novel approach. In essence, instead of using the sentinel only once at the end of the string, a set of identifiable sentinel characters $\$ = \{\$_1, \$_2, \$_3, \ldots, \$_n\}$ is attached to the end of each string. This is necessary since otherwise it would not be possible to assign a given suffix to a specific string, therefore resulting in ambiguity within the BWT.

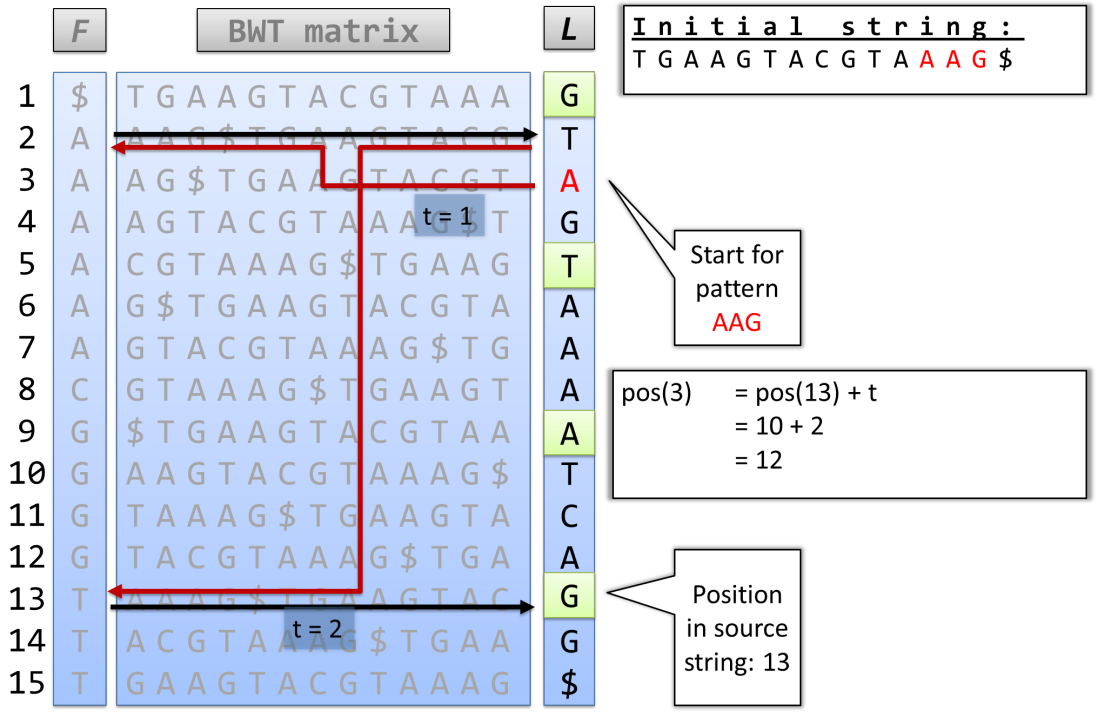


Figure 4.13: Mode of operation for the location function within a BWT. The position within the initial string of first occurrence of pattern "AAG", located in row 3 of the L column (see Figure 4.12) should be recovered. Since 'A' at position 3 is not indexed, the L-F mapping leads to the first 'A' in F (red arrow) and therefore to the first 'T' in L (black arrow). Since this ''T' is indexed neither, the L-F mapping redirects to the first 'T' in F (red arrow), which in turn leads to the 3 'G' of L, which is indexed with original string position 13. As $t = 2$ steps had to be made, "AAG" is located at position $10 + 2 = 12$ in the original string.

Several years after the initial approach, Bauer et al. [2011] further pursued the extended BWT idea by developing two algorithms able to store, index and process large amounts of Illumina-based reads in a BWT structure. This prototype finally matured to BEETL, a BWT C++ library [Cox et al., 2012a], featuring compression techniques like compression of the BWT with runlength encoding or Huffman encoding [Huffman, 1952]. The classical BWT uses a string to store the transformed characters. The BEETL library divides all characters among 5 different piles, one for each letter ($,A,C,G,T) and corresponding to the current suffix (Figure 4.14). These individual piles are virtually compiled into one string which is used for further processing. Therefore all operations discussed previously are available for the extended BWT.

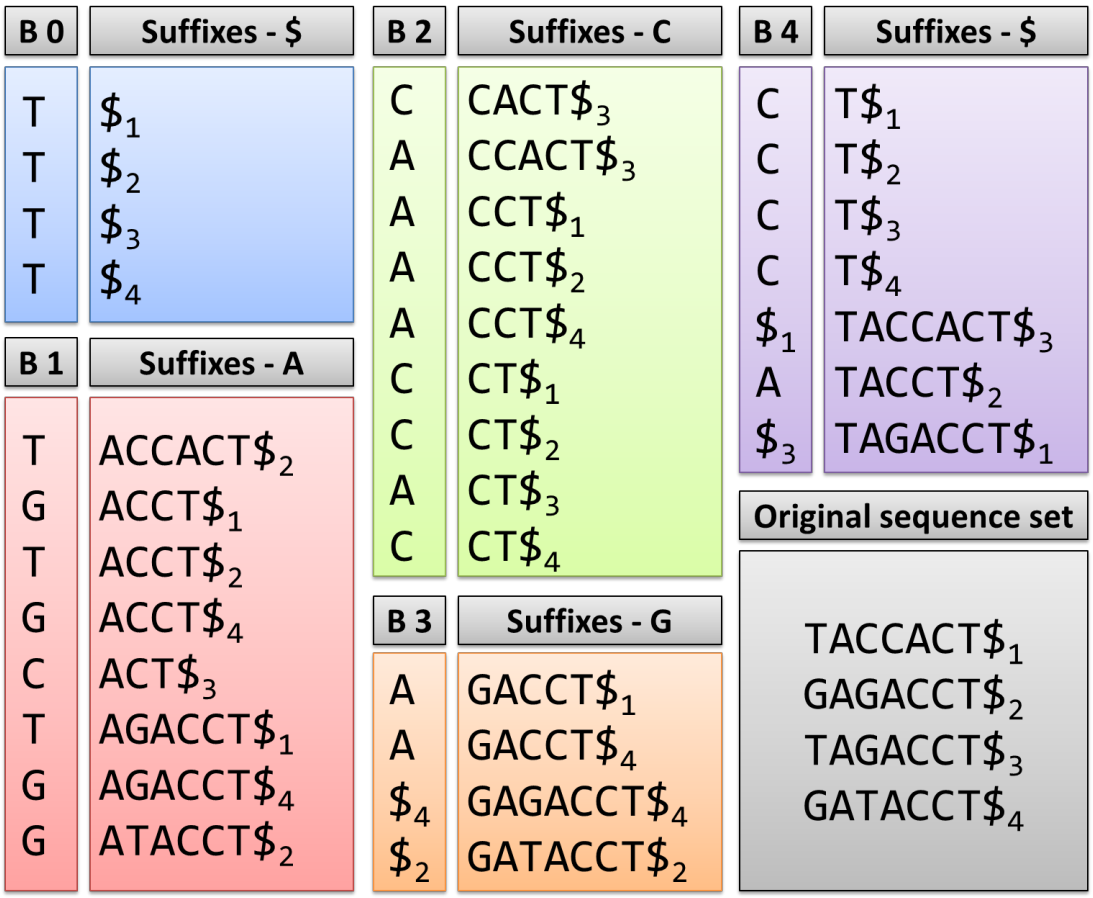


Figure 4.14: Burrows-Wheeler transformed set of strings. The original sequence set is shown in the silver box, coloured boxes contain parts of the BW-transformed string, sorted by suffixes into 5 piles ($,A,C,G,T). Note the usage of the identifiable sentinel characters $\$_1$ to $\$_4$. Example taken from Cox et al. [2012a].

4.3 Implementation of SATYR

The implementation of SATYR follows the requirements outlined in Section 4.1.2 and the principal design decisions made in Section 4.2. In the following each of the modules is introduced in depth while setting the focus on actual implementation details like employed libraries and data structures. The section is divided into three parts corresponding to the organisation of the modules. While housekeeping and maintenance functions are carried out by the `I/O` and `Utility` module respectively, both `Storage` and `Index` module are focused on BWT interactions, and the `Extension` module is solely responsible for the actual assembly.

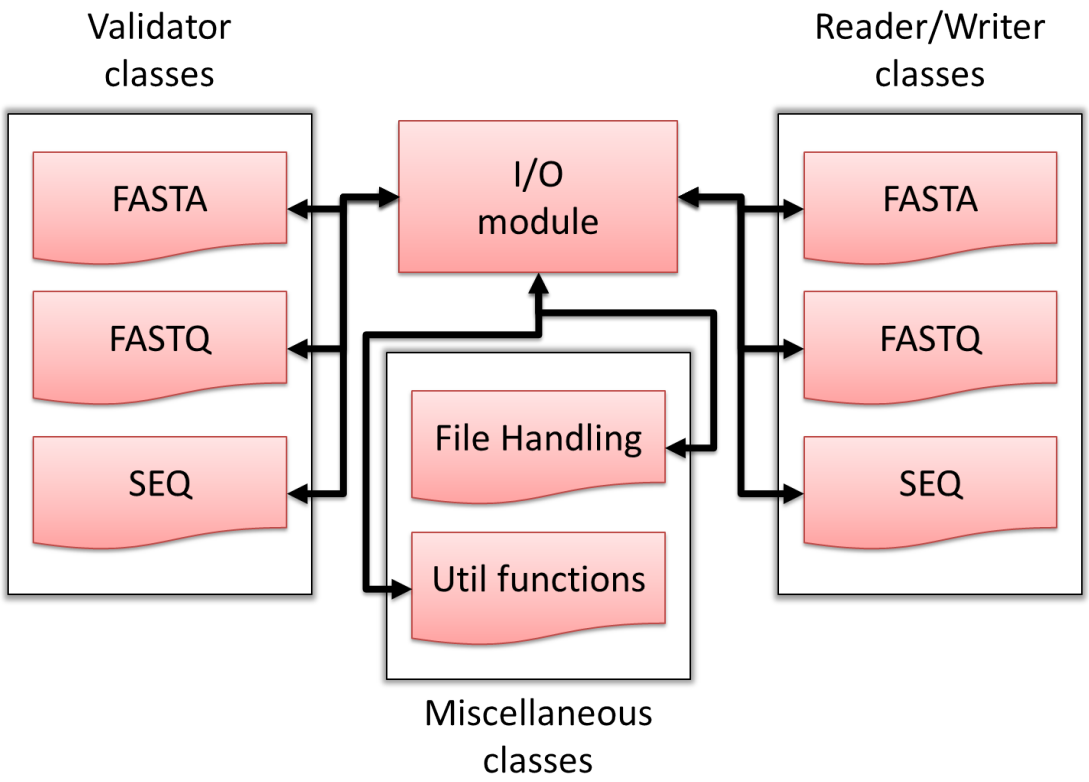


Figure 4.15: Internal organigram of the `I/O` module. Colours are in correspondence with the overall program design (Figure 4.6). The module provides direct read & write access to files, especially to read and seed files in different formats (FASTA, FASTQ, and SEQ).

4.3.1 I/O and supplementary functions

4.3.1.1 I/O module

Main responsibility of the `I/O` module is to provide read and write access to all files used within the program flow. This includes files containing reads, files containing seeds and in case the BWT for the read set was already constructed the direct access to the BWT files. For seed and read files different formats are accepted, covering the most widespread bioinformatics formats. FASTA files [Lipman and Pearson, 1985] include only a information tag in the header and the sequence information. The FASTQ format [Cock et al., 2010], containing additionally quality information for each base is generally found as final output of sequencing runs. SATYR also supports SEQ-files which only contain sequence information, in the form of one sequence per line. It may be used to either save disk space or in cases where meta information for each read is not required. In order to ensure a correct program flow all file types are checked for errors during import and unsuitable reads are

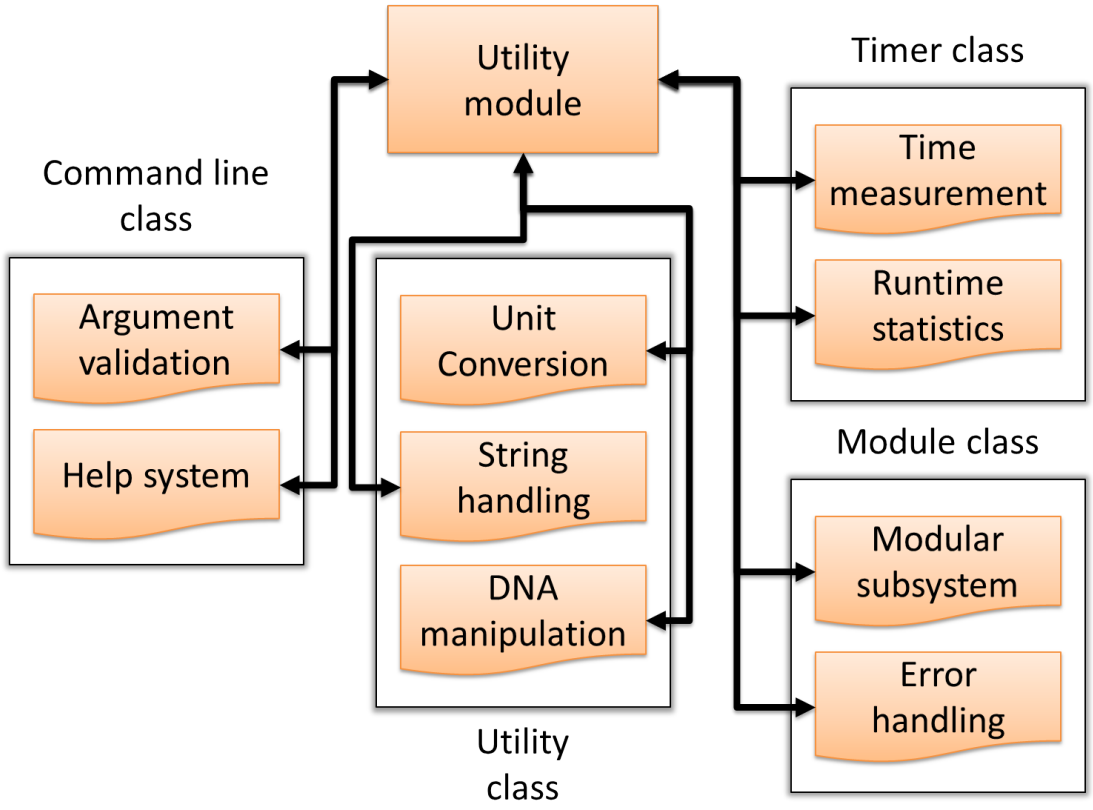


Figure 4.16: Internal organigram of the `Utility` module. Colours are in correspondence with the overall program design (Figure 4.6). The module provides a broad range of utility functions, reaching from string helper functions over command line parsing to time measurement for each module.

automatically discarded. Several helper functions allow to obtain read length and several statistics like average seed length and G+C content.

4.3.1.2 Utility module

The `Utility` module (Figure 4.16) was designed to provide a range of globally used helper functions in order to avoid redundant implementations and code to ease code maintenance. Several different classes are joined within the module, each one focussing on a different scope. The command line class ensures that all user supplied command line arguments, such as file paths, kmer-sizes, or error thresholds are of correct format and within a valid range. The scope of the utility class comprises of a set of versatile helper functions, whereas most of these are related to DNA-specific transformations or distance measurements, such as reverse complement, complement, or Hamming distance. Additional string-related functions are included which are not covered by the C++ standard library. For benchmarking and evaluation purposes, SATYR possesses its own timing system to independently keep track of time spent within different modules for different tasks. Therefore it is

possible to output the time needed to load/create the BWT index, the seed preprocessing or the iterative seed extension phase. Error handling and display of suitable error and help messages is also performed by the `Utility` module.

4.3.2 Working with the BWT

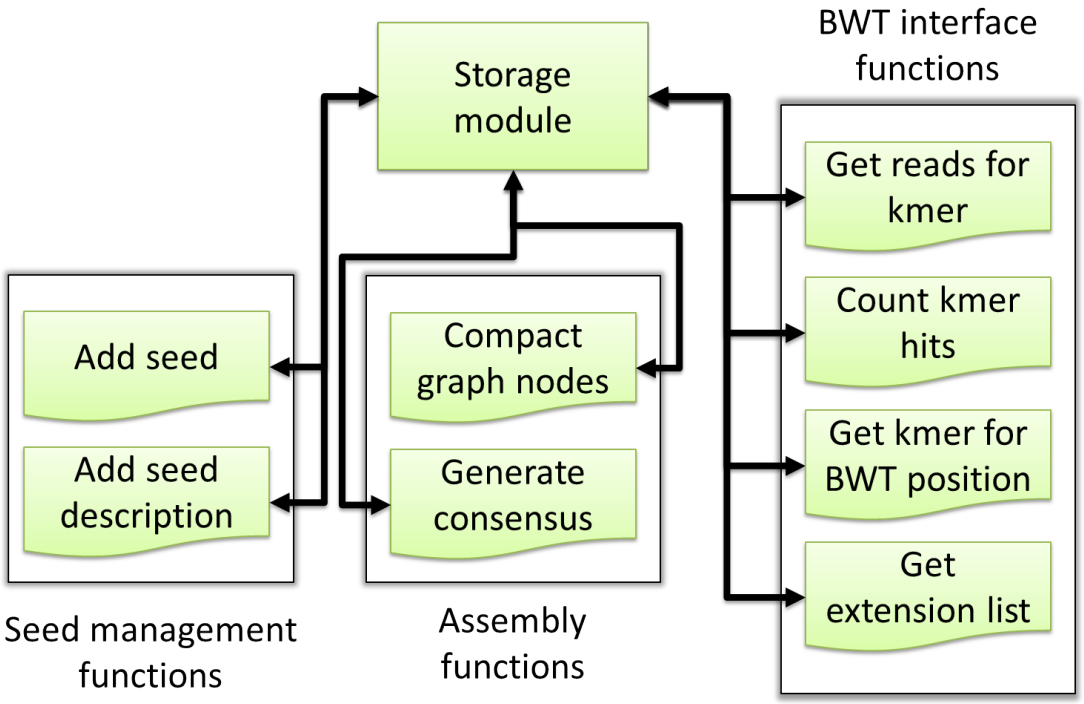


Figure 4.17: The `Storage` module with its three main fields of duty. Main focus of the module is the BWT interface with provides several functions for BWT interaction. Additionally, seed sequences and meta information are processed in the module and a pair of helper functions for BWT-related assembly tasks were implemented in the `Storage` module.

The BEETL library [Cox et al., 2012a] is referenced within the program as external library and dynamically linked. Functions of the library are used to generate the BWT of the input reads if no BWT is present yet. `Index` and `Storage` module are implemented without using function calls from the BEETL library and work independent.

4.3.2.1 Storage module

The `Storage` module provides different functions, ranging from seed management over BWT interface function up to assembly related helper functions (Figure 4.17). Within the program flow of a SATYR run, the first task of the `Storage` module is the accommodation of the set of seed sequences. Since typically the number

of seed sequences will not exceed a few thousand, all seeds accompanied by meta information such as seed name and seed ID are stored within a hashmap. While SATYR was implemented in C++, parts of the algorithms and functionality are kept in plain C99 for optimal performance. As C99 does not feature any kind of hashmap data structure, an external approach was employed. UThash [Hanson, 2009] is a BSD licensed, well documented, and actively maintained hashmap for use in C and is supplied as a stand alone header file providing all necessary functions without the need for further linking.

Although the `Storage` module provides several non-BWT related functions, its main duty is to act as interface to the BWT files on disk. For this task, the module employs a number of auxiliary data structures and caches to speed up queries. BWT functions implemented within the `Storage` module allow for example counting the occurrences of specific kmers (see Algorithm 4.2) which translates into the question 'how many possible extensions for a given 5' or 3' end exist'. While reconstruction of these hits is performed in the `Index` module, the `Storage` module is able to retrieve all reads including a kmer of choice. This list is used later within the extension phase. Each kmer in the BWT text can be directly accessed using its unique BWT position. Using an auxiliary hash structure, pairs of recently searched kmers and their corresponding BWT positions are kept in cache to reduce the number of queries send to the BWT.

The module also contains two assembly related functions due to dependency constraints within the program. Usually, the query for reads given a kmer returns a certain amount of hits. Since the reads originate from sequencing experiments are therefore not expected to be completely error-free, a mechanism is needed which accounts for single sequencing errors and yet is able to separate the hits into different classes corresponding to their overall sequence similarity. After the initial call of the `compact` function which acts as a wrapper, the consensus sequence for the set of reads extending the current kmer has to be generated generated. Ideally all extending reads would be grouped into a single bin, yielding a a single, distinct extension. In reality, due to sequencing errors mentioned and repeating kmers several bins, each corresponding to a different direction in the assembly graph exist. While some of these bins can be discarded because of coverage values deviating from the average others have to be included in the graph and add a certain level of ambiguity to the assembly process.

4.3.2.2 Index module

While the `Storage` module implements several helper functions and takes care of seed storage and management, the `Index` module is solely responsible for index management and queries against the BWT index (Figure 4.18). The BWT construction is performed directly by the BEETL library through wrapper functions implemented within the `Storage` module. Index creation however works without

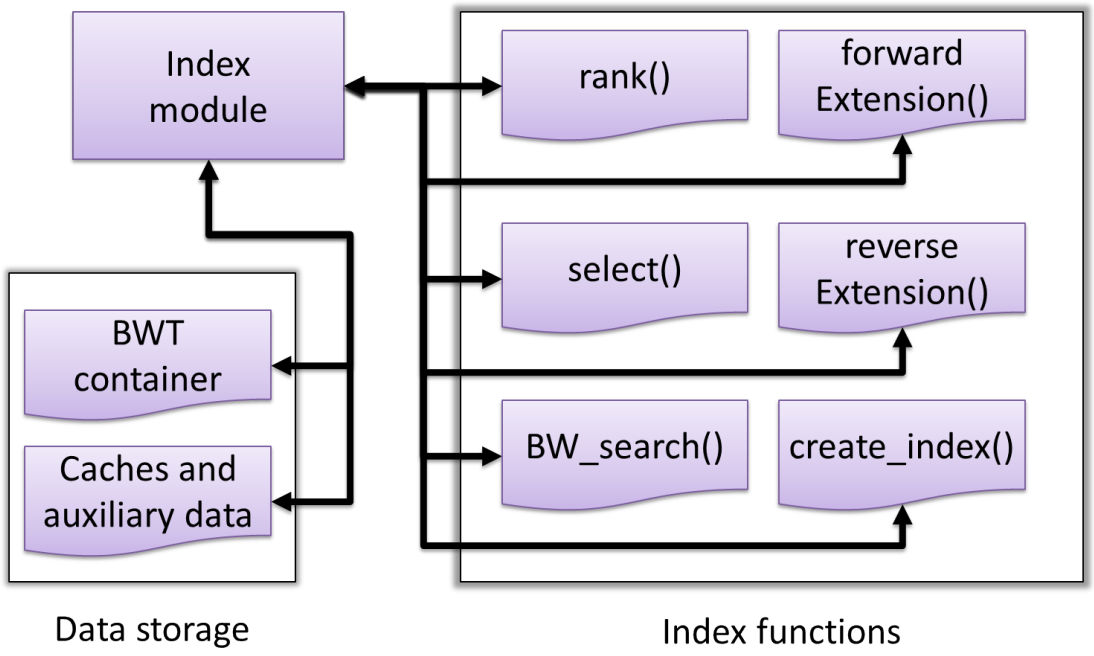


Figure 4.18: Internal structure of the `Index` module. While other modules typically provide more than one function, the `Index` module is restricted to the eponymous index. The module holds the entire BWT of several tens or hundreds of gigabyte in RAM and possesses several caching structures used to work with the BWT. It also implements all relevant functions described in Section 4.2.2.

referencing any BEETL functions and is therefore fully integrated into SATYR. Initially, all so called piles have to be preprocessed (Figure 4.19 A). Six piles together form a virtual BWT string, whereas each pile contains all suffixes starting with a specific character. In contrast to the five piles shown in Figure 4.14, the BEETL library adds an additional sixth pile for characters not in the alphabet set, named "Z-pile". All piles are read into memory, one after another in chunks equal to the chosen BWT block size which defaults to 1,000 characters (Figure 4.19 B) & C). During the transfer from disk into main memory, different counter variables used in later stages to keep track of borders between the different piles are initiated. The BWT block size has direct influence on the performance of the index, as these blocks act as markers within the BWT. A block size of 1 means direct access to each element combined with a very large amount of memory consumed by marker structures. To large values for the block size on the contrary translate to lower memory usage but also included much longer run times, since from a given point within the BWT index a way back or forward to the nearest index has to be found via $L - F$ mapping (see Section 4.2.2.4). For each block, a set of 1,000 sub blocks is created and filled whereas each sub block itself contains an integer array of size 255 (Figure 4.19 D). Since the size of 255 entries corresponds exactly to the size of the standard ASCII alphabet each sub block is used to establish a mapping

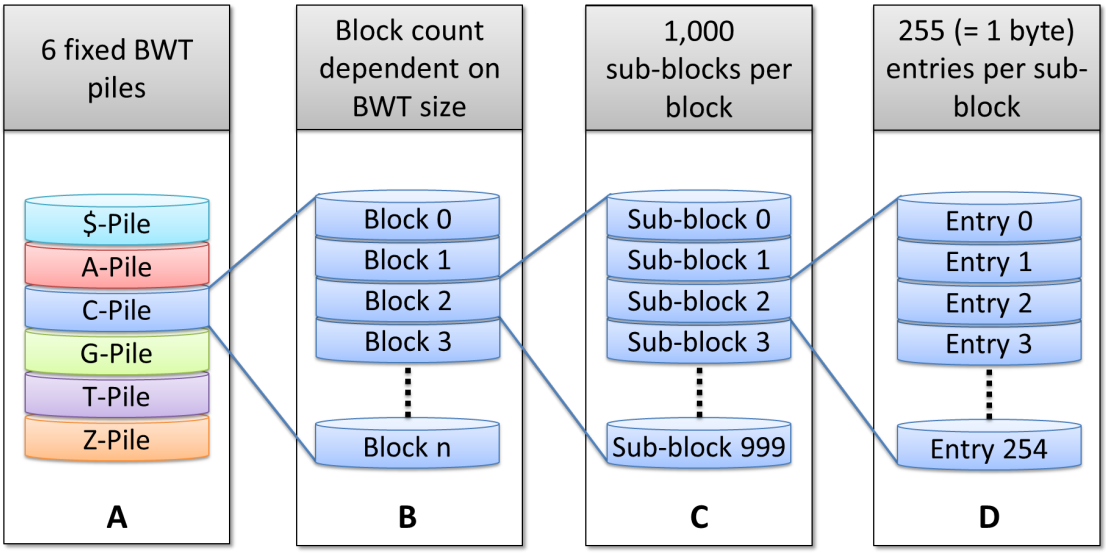


Figure 4.19: Representation of the BWT within the library. **A)** The complete BWT string is divided into six different piles, which act as one virtual BWT string. **B)** Each of the six piles has a varying amount of blocks, dependent on the number of suffixes allocated in this pile. **C)** The sub-block size is normally set to 1,000 as a compromise between speed and memory consumption. **D)** A Sub-block has 255 entries, corresponding to one byte or in other words the range of all ASCII characters.

between ASCII code of a character and its count within the BWT. The letter 'A' has a decimal code of 65, therefore `array[65]` would be increased by 1 in case an 'A' is encountered within the BWT.

Once the BWT is available in memory and all counters are initialised the index structure is ready for queries from other modules. In order to answer queries, the `Occ()` and `BW_search()` functions as well as procedures for forward and reverse reconstruction of reads from the read set were implemented and modified to work with multiple sequence BWTs (Figure 4.18). Whereas the initial version of the FM-index employed the `Occ()` function to find the number of occurrences of a given symbol letter until a given BWT position [Ferragina and Manzini, 2005], the `Index` module additionally features the `select()` function which is able to to find the position of the X-th occurrence of given letter symbol in the BWT. As SATYR tries to extend in either 5' or 3' direction, it is necessary to reconstruct reads from the BWT in both directions. Therefore, 5' extensions are handled by the `forward_reconstruction()` function and 3' reconstructions are performed by `reverse_reconstruction()`. This ensures that only those parts of the read which are relevant for an extension of the seed are reconstructed, while the opposite direction is skipped.

4.3.3 Iterated assembly

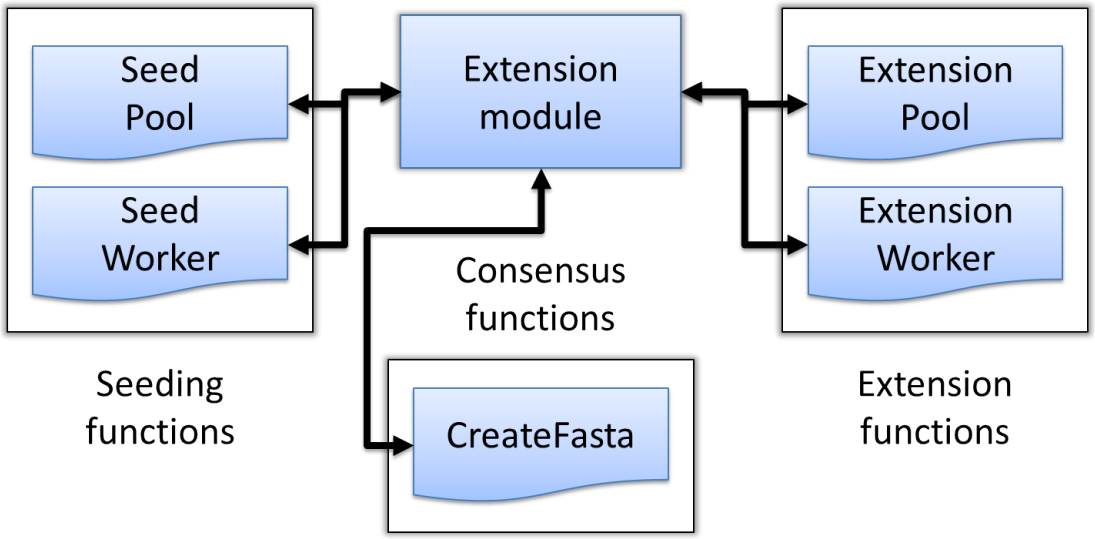


Figure 4.20: Organisation of the `Extension` module. The extension phase starts by calling the seeding functions while the seed pool controls a set of seeding workers. The seeding phase is very similar to the subsequent extension phase, however, during seeding several data structures used for extension are initialised. The extension functions provide the framework for the assembly process, while several functions located throughout other modules perform single steps of the assembly. After the assembly process has finished, consensus sequences are derived from the assembly graphs.

4.3.3.1 Extension module

The `Extension` module basically runs the initial seeding phase as well as the iterative assembly and the consensus generation (Figure 4.20). The so called seeding phase is the first step of the iterative assembly and tries to locate first overlaps between the seed sequence and the input read set. Due to the initialisation of several data structures used during the assembly process the first overlap step was implemented separated from the subsequent main assembly stage. For all seeds that report relevant overlaps the extension phase is started right after the seed step. Both, seeding and extension phase are multi-threaded and possess thread pools which assure that at each time point a given number of threads is active. Each seed runs in its own thread context, whereas global data structures such as the BWT are shared between all threads and synchronised where necessary. Main function of the `Extension` module is to control the program flow of the assembly process, functions related to the BWT, the BWT index, graph simplification, and filtering steps are implemented within their corresponding modules.

While the functionality required for the assembly process does not reside in the module, all relevant data structures do. With respect to the assembly, the

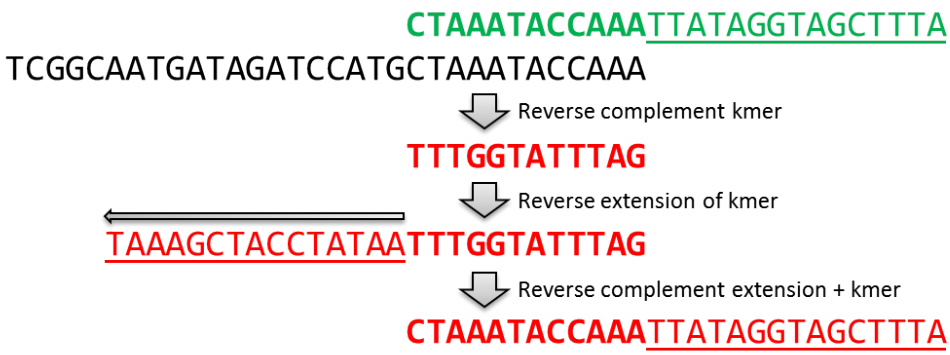


Figure 4.21: Visualisation of the extension process for the 3' end of an seed (black). The forward extension process is shown in green, the kmer for which the BWT index is queried for is shown in bold font, the derived extension of the kmer is underlined. The process for the second strand is depicted in red. The reverse-complemented kmer is shown in bold whereas the extension, this time acquired through reverse extension is again underlined. To simplify work, all results form the second strand are reverse completed to match the results from the first strand.

assembly graph is the most important structure within the program, as it stores the network of overlapping reads for each seed and therefore is the direct precursor of the final consensus produced by SATYR. In order to reproduce the topography of the assembly graph the Boost Graph library (BGL) [Trobin, 2002] was chosen as it provides a very mature and well documented set of graph data structures, functions, and algorithms. From the various kinds of graph structures the BGL offers, the `adjacency_list` was selected due to support for directed edges. The functions defined on its vertices and edges include the count of ingoing and outgoing edges, the possibility to add custom data structures to nodes and direct access to each of the graph's nodes and vertices.

In a first step, the list of flanking kmers stored by either the initial seeding or the last iterative assembly step is retrieved by the `Storage` module (see Figure 4.7). In case of the first extension step, this list contains only one entry per direction (5' and 3'), while the reverse complement of the overlapping sequences is implicitly computed throughout the program and subsequently translated into results of the sense strand. For each seed and therefore in each thread two distinct units perform the extension in 5' and 3' direction.

The `Index` module takes the aforementioned list as input and queries the BWT index for existence and position of these kmers in any of the input reads. Depending on the direction of the current extension and the strand of the current kmer, hits

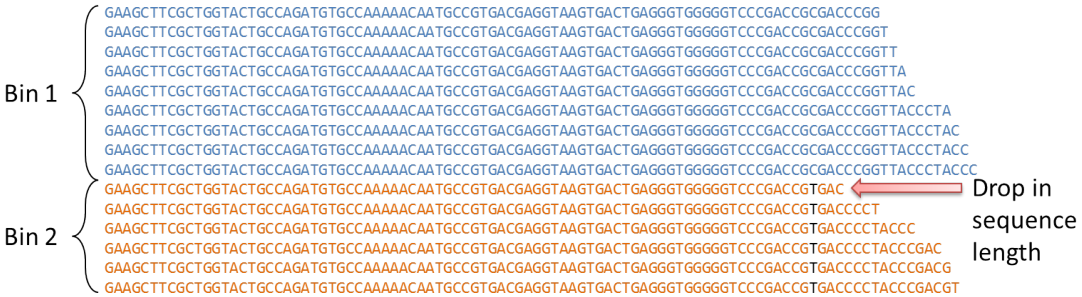


Figure 4.22: The binning mechanism used by SATYR is based on the inherent properties of the BWT. Due to the strict lexicographical sorting of all hits found by the search and their corresponding extensions, single base changes (indicated in black) create drops in the sequence length distribution.

are extended using the `forwardExtension()` and `reverseExtension()` functions supplied by the `Index` module (Figure 4.21).

Once the extensions from both strands have been filtered for to short extension ($\leq 2\times$ kmer size) a sequence similarity-based binning is performed. Due to the properties of the BWT, all hits and extensions extracted from the BWT are lexicographically sorted. It it therefore possible to detect larger variations within the set of extension by drops in the sequence length distribution in the ordered extension list (Figure 4.22). Although the binning does not work flawlessly in every case, it produces bins with enough similarity to produce significant consensus sequences for each bin, since sequencing errors are typically outvoted during the majority vote-based consensus generation by other members of the bin. For each bin's consensus sequence, the 5' or 3' flanking kmer is extracted and stored in the kmer extension list.

The BGL-based assembly graph is updated in such a way that based on the current node (initially the root node) for each bin a new node is created and linked via a directed edge to the parent node. Within the node's data structure, coverage information, the node ID, and its extension sequence is stored. Since the sequence is also stored in the BWT and for each bin a representative read is chosen which allows for exact location of the extension sequence there is no obviously no need to store the extension sequence a second time within the node. Indeed, benchmarks during SATYR's implementation showed performance benefits when storing the sequence in the graph compared to the BWT only storage, while the additional memory consumption stays reasonable.

After the last extension step, induced by either the absence of suitable extensions or by reaching the user-supplied limit of cycles the final consensus generation is carried out by the `Extension` module. The process works by backtracking the path from the final node inserted in the graph back to the root node, as this will construct the longest extension. At each node the extension sequence is retrieved

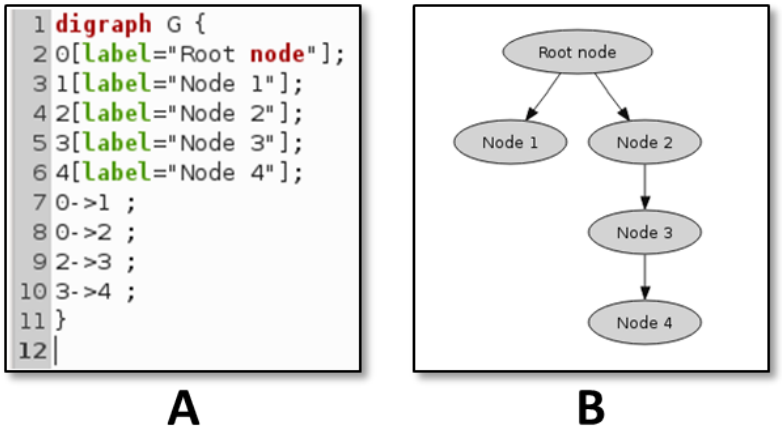


Figure 4.23: The ".dot" format used by the Boost graph library during export. **A)** Source format exported by SATYR. **B)** Visualisation of the corresponding .dot file using the `dot` layout algorithm[22].

and added to the consensus while clipping possible overlaps between the nodes.

Depending on the input reads and the initial seed, structures of the assembly graph show a great range of variation, reaching from very simple and linear graphs (Figure 4.24 A) to variants including simple branches which quickly terminate (Figure 4.24 B) and to complex, highly ambiguous layouts with many internal branches (Figure 4.24 C). Additionally to consensus sequence for 5' and 3' extension both assembly graphs are exported in the widely supported ".dot" format used by the GraphViz software package[21] (Figure 4.23).

4.4 Results

SATYR was implemented to meet the requirements outlined previously, while its main focus was set to a seed-based, targeted assembly of next generation sequencing data, to provide insights into the flanking regions of the seed as far as possible. As such, the length of the produced extensions is the main criteria under which SATYR was evaluated, the accuracy ranks only second in comparison to contig length. This section therefore evaluates the presented tool in comparison with Mapsembler by Peterlongo and Chikhi [2012], as both tools share similar use cases and requirements.

For evaluation purposes two different data sets were used. The first set from the K12 MG1665 *Escherichia coli* strain and the second read set, a genomic CHO-K1 cell line Illumina sequencing experiment, were evaluated on a high performance

[21] http://www.graphviz.org
[22] Part of the Graphviz package

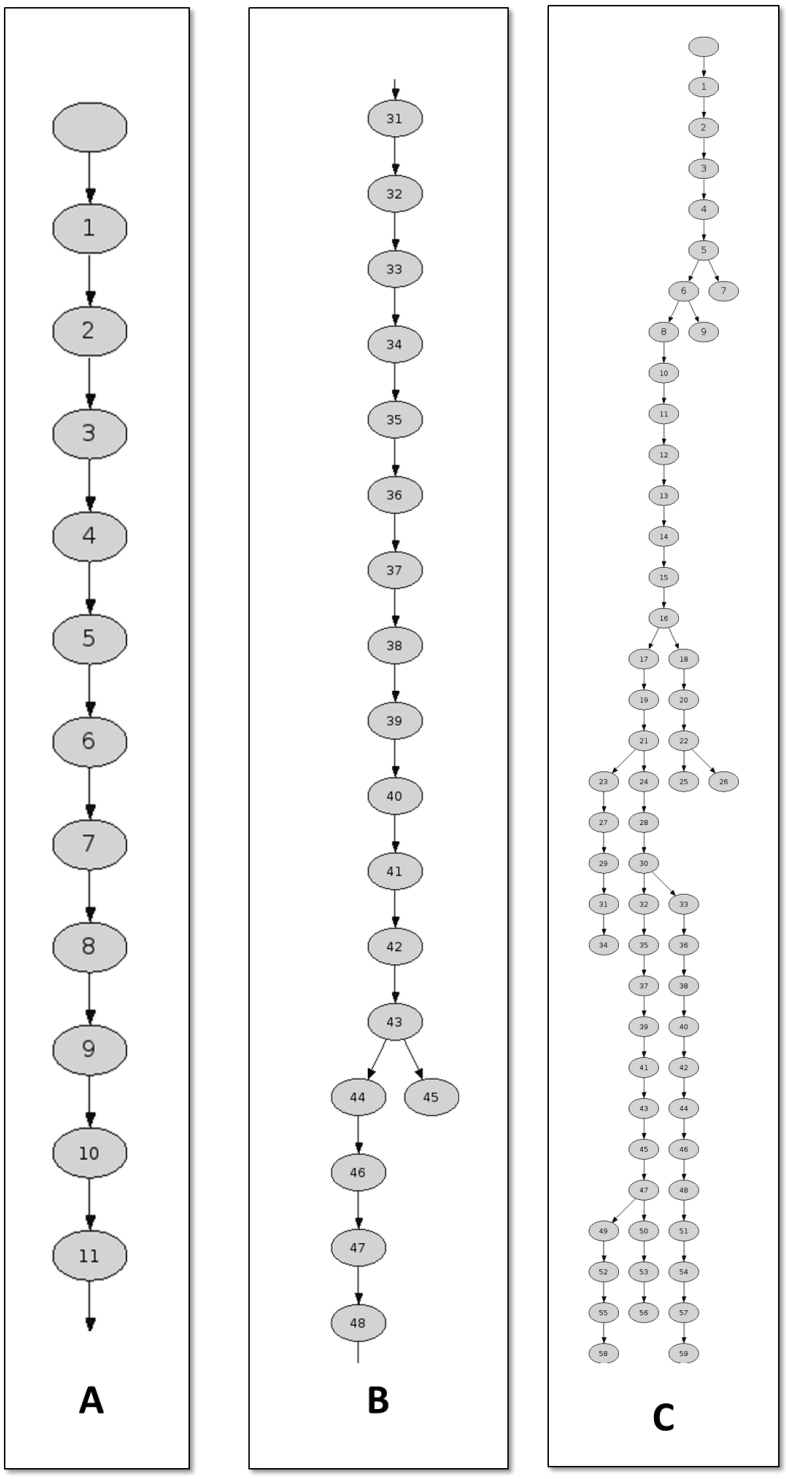


Figure 4.24: Three different example assembly graphs produced by SATYR and exported using the Boost-graph library. The numbers within the vertices represent the node ID. Layouts are partly cropped to fit into the figure. **A)** Ideal graph with no branching, each extension only overlaps with one specific read set. **B)** Assembly graph with simple branch. The branching most probably was related to a sequencing error, as no further extensions were found for this read set. **C)** Heavy branching in the graph. Typically occurs in repetitive regions or in case of a too small kmer value, thus leading to multiple and contrary read sets.

server carrying eight X7542 (2.67GHz) Xeon CPUs and 1.0 TB of RAM. The machine employs the Fedora Linux distribution (release 18).

4.4.1 Prokaryotic Escherichia coli K12 MG1665 dataset

The genome of the *Escherichia coli* K12 MG1665 strain (GenBank accession NC_000913) was sequenced in 2008 on an Illumina Genome Analyzer using a 200 bp insert library. In total 20,708,709 paired end reads were generated, yielding 749 Mb of sequence data ($\approx$ 162$\times$ coverage) which are publicly accessible (Short Read Archive [SRA] ID SRX000429). A set of 10,000 seeds was generated by a Perl script that randomly samples 36 bp seed sequence from the MG1665 reference genome. The Mapsembler software was compiled and evaluated using the most recent 2.1.2 version and the supplied Makefile with no additional settings. Program parameters were chosen to match SATYR's method of operation as close as possible. In correspondence with the read size of 36 nt a suitable range of kmer sizes was restricted to values from $k = 17$ to $k = 29$. Only values of odd kmers were used for evaluation, as Mapsembler does not allow even kmer values due to possible palindromes, therefore resulting in the final kmer set $\{17, 19, 21, 23, 25, 27, 29\}$. Since the reference sequence is exactly known it is used as gold standard against which all similarity comparisons are performed with BLAST.

4.4.1.1 Assembly analyses - similarity

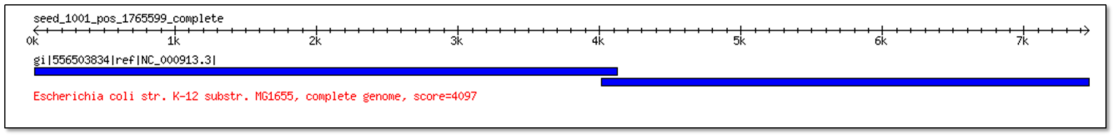


Figure 4.25: A misassembled extension produced by SATYR. Shortly after position 4,000, a second alignment (blue) is shown, indicating that this assembled regions most probably originates from a different site within the genome. The assembly quality however, is not affected by this misassembly.

In a first step the quality of assembled sequences was assessed by BLAST comparisons against the reference genome. To allow for minor variations discontiguous MegaBLAST was used [Zhang et al., 2000; Ma et al., 2002], as this BLAST implementation is specialised for search of similar but not exactly identical sequences. The coverage term used throughout the evaluation is defined as the percentage of bases in the BLAST reference covered by bases of assembled regions. The assembled sequences consisting of 5' extension, seed sequence, and 3' extension were used as query, while the analysis was repeated for each of the seven kmer values. For smaller kmer values reaching from 17 to 21 Mapsembler performs optimal, i.e. the average coverage is 99.99% or higher (Figure 4.26). Starting from $k = 23$, SATYR constantly loses coverage reaching its lowest value of 79%

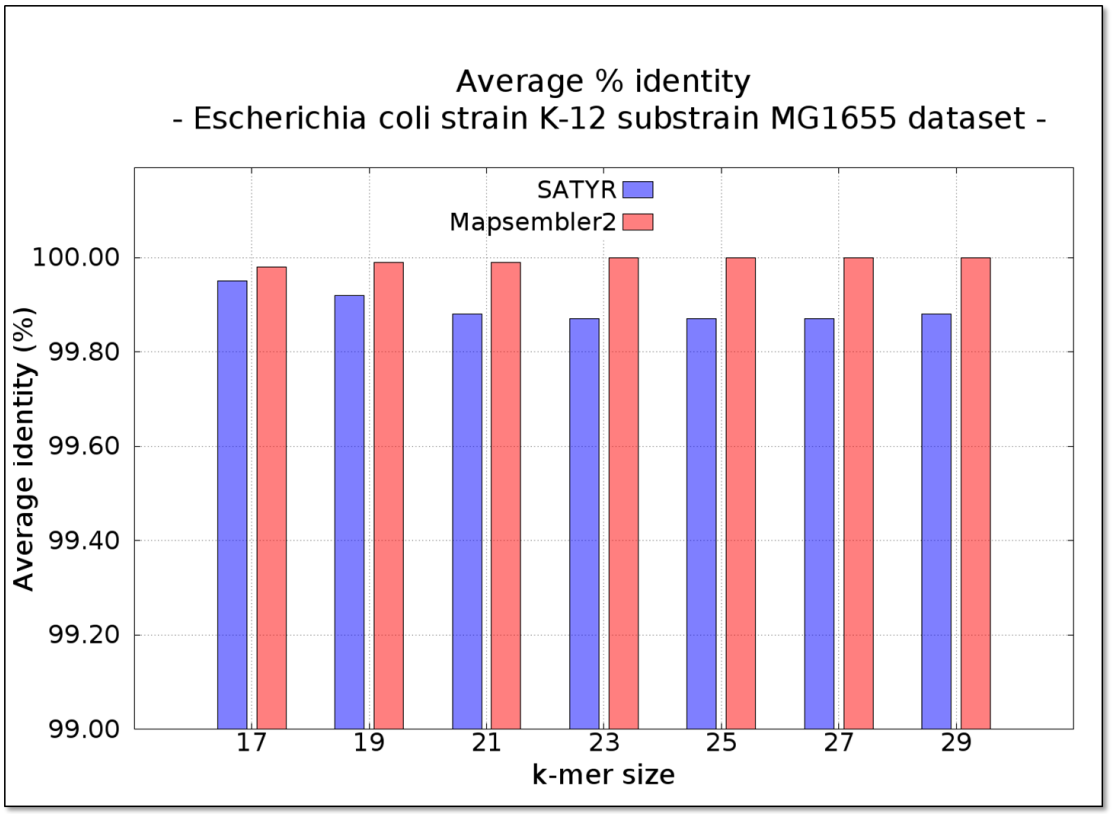


Figure 4.26: Graphical representation of the average reference coverage of all extensions. The coverage of the reference sequence provides information about the structural accuracy of the assembly.

for $k = 29$. This decrease in reference coverage is caused by SATYR's internal handling of the extension process. On the one hand, for small kmer values, usually many matching reads are found which are subsequently grouped into a few large bins (Figure 4.22) due to high similarity between the reads. On the other hand, long kmers typically result in fewer hits, since the probability of finding an exact match decreases for growing kmer values. Often, hits reported by longer kmers show a certain amount of variation, therefore introducing more different bins in relation to the hit count than hits of short kmers. While for short kmers one bin is typically dominating the set in terms of members, for longer kmers the number of entries per bin is more evenly distributed, yielding a higher chance of choosing a wrong way in the assembly graph. It is important to point out that these misassemblies do not contain randomly assembled reads but correctly assembled regions originating from other parts of the genome (Figure 4.25), as blasting the corresponding regions of SATYR assemblies leads to high quality database matches.

The previous statement concerning quality of misassembled regions is confirmed by the average percentage of identity of all alignments, shown in Figure 4.27. Again,

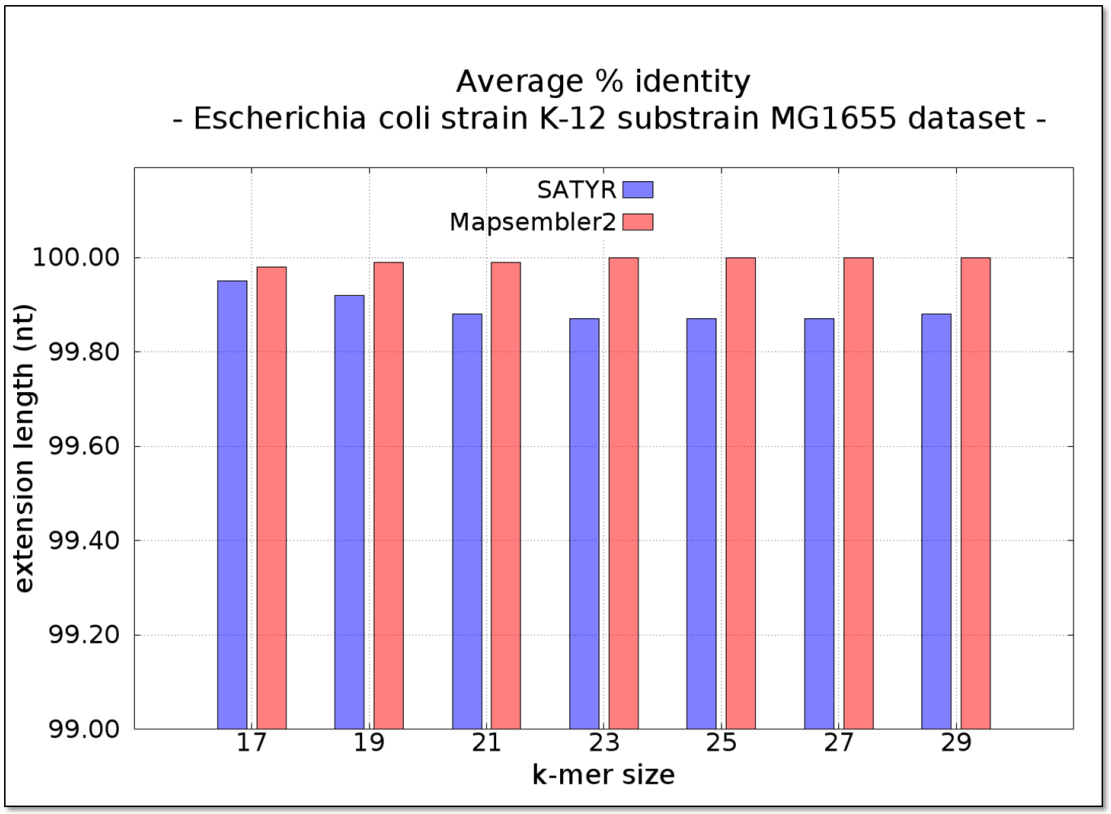


Figure 4.27: Graphical representation of the overall percentage of identity for Mapsembler and SATYR. Higher values indicate more similarity to the reference genome. The identity value can be used to assess an assembly on nucleotide level, compared to the structural assessment of coverage values.

Mapsembler performs well with qualities of 99.99% and above for all kmer values. However, the distance to the values achieved by SATYR is relatively small as suggested by the y-axis scaling of the diagram. The overall average percentage of identity for SATYR is 99.89%, resulting in a very small quality decrement of only 0.1%.

4.4.1.2 Assembly analyses - length

In a next step, the focus was set to assembly performance in terms of extension length. In contrast to the query coverage and percentage of identity, SATYR performs very well throughout all tested kmer sizes. For small kmer values, SATYR was able to reliably generate extensions of 2,000 nt and more, while Mapsembler crosses the 2,000 nt threshold not before $k = 23$ (Figure 4.28). For no kmer value Mapsembler is able to reach the average extension length achieved by SATYR. Averaging through all kmers, SATYR's extension are 3.35 fold longer than the sequences generated by Mapsembler, although for large kmer sizes the distance between both approaches decreases. The average extension length reaches

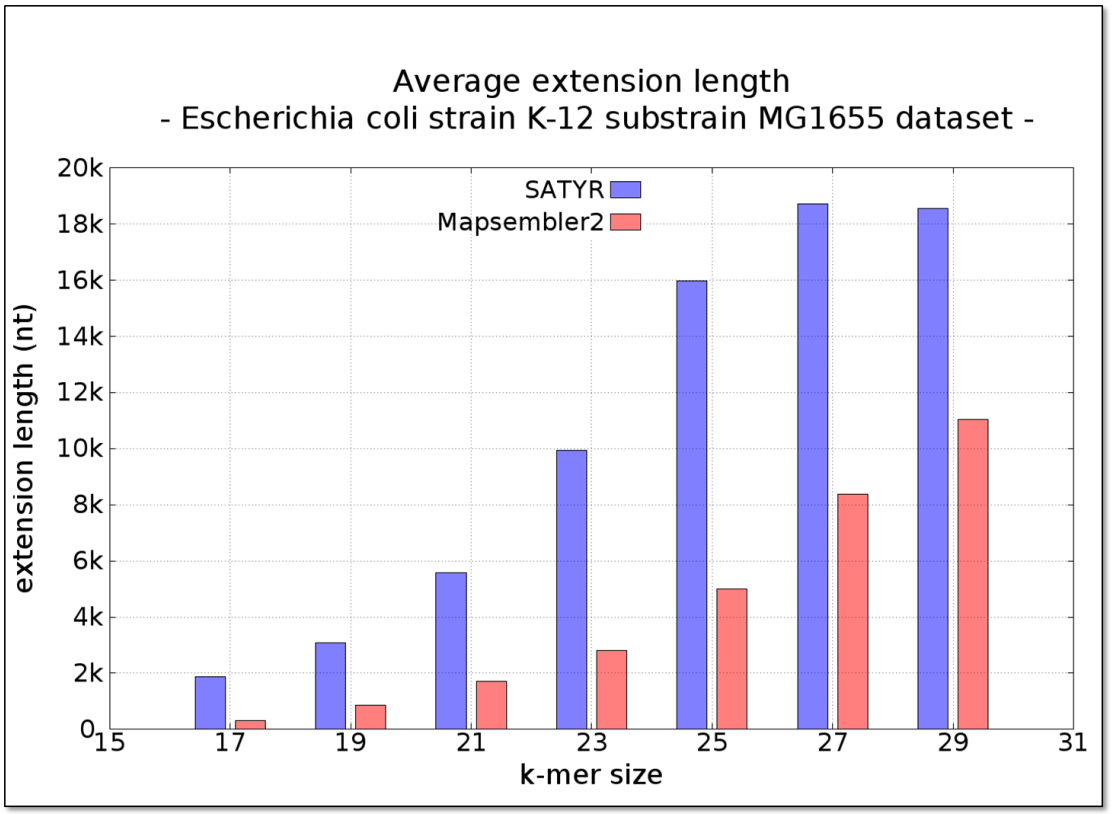


Figure 4.28: Diagram showing the average length of the assembled extensions for Mapsembler and SATYR.

a maximum value at $k = 27$, as for $k = 29$ no further increase is recognisable. The tendency visible for Mapsembler does not show signs of a similar stagnation for up to the largest kmer value.

The difference between SATYR and Mapsembler demonstrated in average extension length becomes more apparent for the maximal extension lengths achieved by the two tools (Figure 4.29). While the maximal length for SATYR surpasses 10,000 nt directly with the smallest kmer, Mapsembler requires a kmer size of at least 21 to generate extensions of similar length. For $k = 21$ SATYR is able to double its maximum output from $\approx 19,000$ nt to more than 40,000 nt. From $k = 23$ on, the maximal obtained length increases at slower rate, reaching roughly 65,000 nt as peak value for $k = 29$. In comparison, the development of Mapsembler's maximal extension length shows a slower increment, reaching 10,000 nt not before $k = 21$. Mapsembler reaches its maximum extension length exactly at 20,060 nt for $k = 27$, while 25 and 27 show similar values with minor deviations. Further evaluations with several different setups confirmed a hard limit for the maximal extension at 20,060 nt, as under no parameter choice longer extensions were reached.

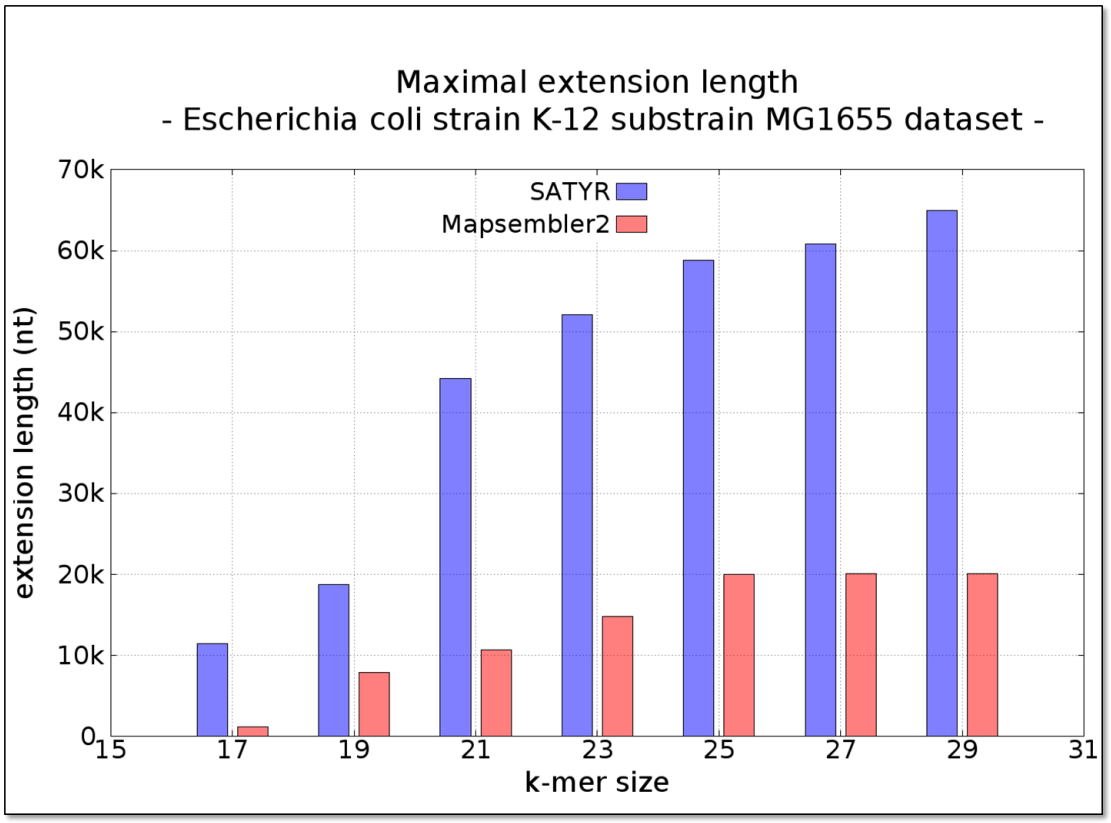


Figure 4.29: Diagram of the maximal length reached by assembled extension for Mapsembler and SATYR. Note the stagnation of Mapsembler for $k \geq 25$ at 20,000 nt.

4.4.1.3 Distribution of extension lengths

To further investigate the assembled extensions produced by Mapsembler and SATYR, for each tested kmer value a histogram containing the number of sequences and the corresponding length of these extension was generated. The histograms (Figure 4.30) provide insight into the composition of the extended sequences, e.g. how many of the 10,000 seed extensions are shorter or longer than 10,000 nt. For the histograms a bin size of 200 was chosen to allow for a fine grained resolution of the analysis. For $k = 17$ Mapsembler is only able to produce about 7,500 very short extensions which do not exceed 200 nt in length. With $k = 19$ the distribution becomes more heterogeneous with more bins, however no extensions reach more than 10,000 nt in length. Starting from $k = 21$ up to $k = 25$, the length of extensions generally increases and first sequences can be extended further than 10,000 nt. The image changes for kmer values ≥ 27; the bin counts for longer sequences increase while two dominant bins, one for sequences reaching about 10,000 nt, and a second one containing extensions of roughly 20,000 nt contain most extensions. Although these bins contain more extensions than all other bins, their content only contains 15% of all extensions. The diagram also confirms the previous statement, that no extension produced by Mapsembler reaches more than 20,000 nt length as no bins

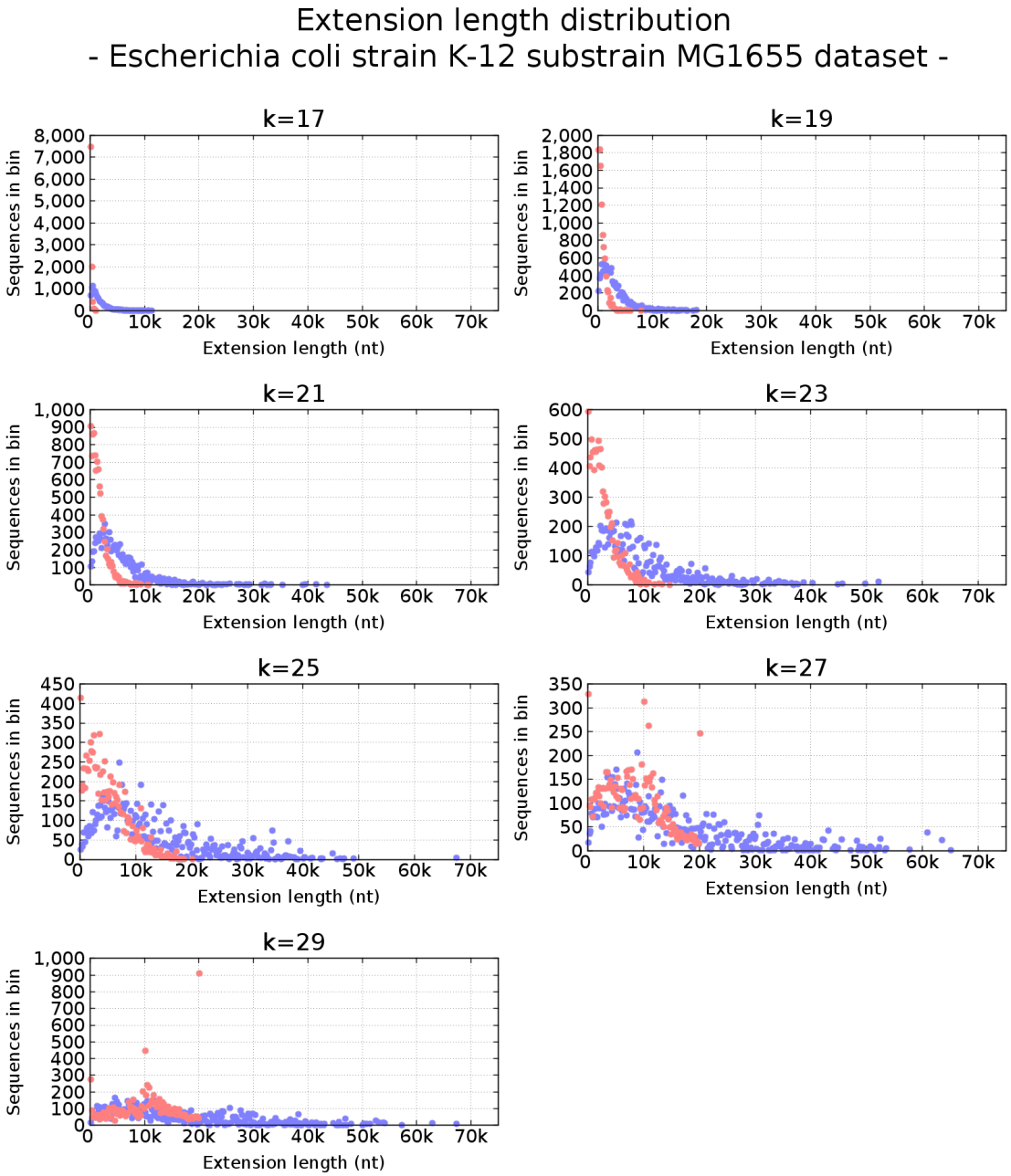


Figure 4.30: Histogram of the extensions length for different kmer sizes. The x-axis shows the number of sequences in the corresponding bin (bin size: 200 nt length), the y-axis holds the extension length. Red dots represent Mapsembler extensions, blue dots indicate SATYR extensions.

are identifiable past the 20 kb border for $k \geq 27$. SATYR in comparison, shows a different development throughout the different kmer sizes. SATYR quickly starts to extend seeds longer than Mapsembler, while the majority of extensions is con-

centrated in bins below 20,000 nt length. This distribution changes for kmer sizes ≥ 23, when more and more bins with lengths $\geq$ 20,000 nt are identifiable and again for $k = 27$ when more than half of the bins contain extensions $\geq$ 20,000 nt. For the largest tested kmer 29, the count of sequences in bins $\leq$ 20,000 nt decreases significantly, leading to a relatively even distribution of extensions of various length.

4.4.1.4 Ratio between shorter and longer extensions

In a last step, the evaluation focuses on the number of extensions reaching at least a certain length threshold. Thus, three different thresholds (500 nt, 5,000 nt, and 10,000 nt) were examined (Figure 4.31), in order to assess the amount of very short and therefore most probably undesired extensions. As for the previous analyses, Mapsembler does not generate reasonable extensions for kmer sizes ≤ 17, while SATYR can extend nearly half of the seeds to 500 nt or more. Mapsembler subsequently catches up with SATYR's results, although it is not before $k = 29$ that Mapsembler can outperform SATYR.

For mid-sized extensions of at least 5,000 nt the portion of longer extensions decreases for both tools. Again Mapsembler does not produce significant amounts of suitable extensions for $k \leq 23$, while SATYR is able to deliver reasonable amounts of long extensions for $k \geq 19$ and continues to increase the number of extensions up to a maximum of almost 80% of all extensions. As for the previous threshold, Mapsembler is not able to catch up before the last kmer is reached.

Following, the threshold was set to 10,000 nt which corresponds to 50% of Mapsembler's extension limit. For this threshold, Mapsembler first starts to generate long extensions for $k = 25$, while SATYR is able to produce roughly the same portion already for smaller kmer values ($k \geq 21$). The limit for both tools is located slightly below $\approx$ 6,000 nt. As for both previous thresholds, for $k = 29$ Mapsembler is able to outperform SATYR for the first time, producing also slightly more extensions of the given threshold.

4.4.1.5 Program run times and memory consumption

Mapsembler's memory consumption averaged around 40 MB of RAM, which is less than the size of the input reads and owed to the iterative build up of the index structures in such a way, that at no point the whole set of reads is present in memory. The program run times varied between 20 minutes for the smallest kmer size and 25 minutes for the largest kmer, where the time for the initial kmer index creation is already included in the overall running time.

Memory consumption for SATYR is stable at roughly 3.7 GB of RAM. Index creation for SATYR is independent from the kmer size, therefore the index had to be created only one time which took 8:50 minutes. Run times for SATYR however

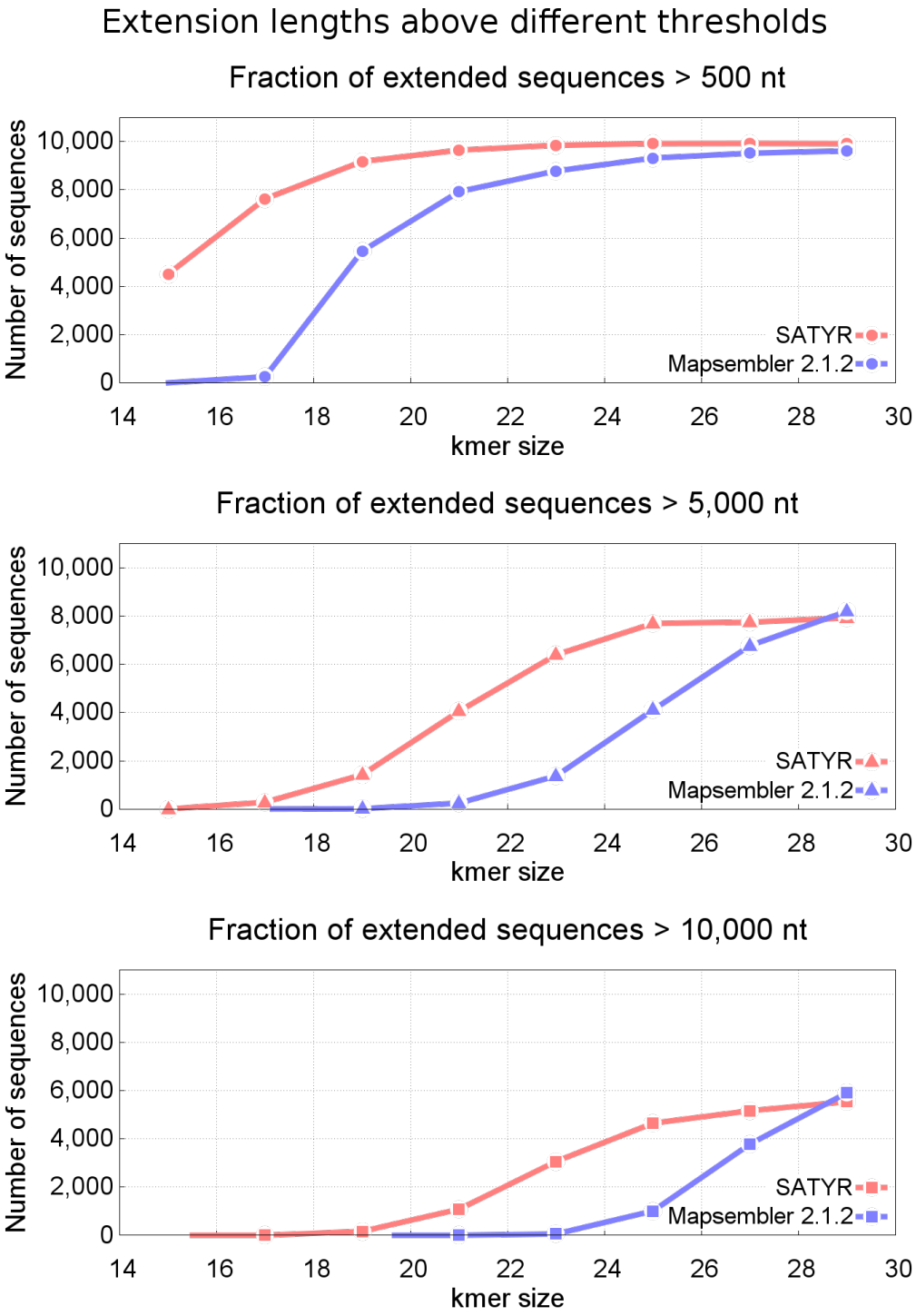


Figure 4.31: Portions of reads larger than three chosen thresholds (500 nt, 5,000 nt, and 10,000 nt). The x-axis shows the kmer size, the y-axis contains the corresponding number of sequences.

are heavily affected by the number of seeds. Due to the high number of seeds (10,000 samples) used for evaluation run times for larger kmer values increase up to 17 hours. Indeed, the amount of seeds employed for benchmarking will typically not be used for actual use cases, therefore greatly reducing the run time of SATYR.

4.4.2 Eukaryotic CHO-K1 ATCC CCL61 dataset

As the initial aim of SATYR was the extension of incomplete cDNAs originating from a CHO-K1 cell line, genomic sequencing reads for this cell line had to be generated in order to perform a targeted assembly as proposed by SATYR. The source cell line was the same for the supplied cDNA seed sequences and the new genome sequencing experiment, thus DNA sequence compatibility between both experiments can be assumed.

4.4.2.1 DNA library setup and sequencing

Lane	Library name	DNA concentration	Fragment length	Read overlap
1	K2 CHO PE	8 pM	339 nt	+27 nt
2	K3 CHO PE	6.5 pM	575 nt	-209 nt
3	R2 CHO PE	8 pM	314 nt	+89 nt
4	R3 CHO PE	8 pM	530 nt	-138 nt
6	R2 CHO PE	8 pM	314 nt	+89 nt
7	R3 CHO PE	7 pM	530 nt	-138 nt

Table 4.4: Flowcell allocation of the sequencing run. The flowcell has 8 lanes; lane 5 was used for Illumina's recommended quality control with the PhiX phage. Lane 8 was used for a different sequencing experiment. In total 6 lanes (1-4,6-7) of this flowcell were used for CHO paired end sequencing. Four different libraries were employed: two with a negative overlap size (K3 CHO PE, R3 CHO PE, Figure 4.32 A), resulting in a gap between the two adjacent reads, and two libraries with positive overlap values (K2 CHO PE, R2 CHO PE, Figure 4.32 B), yielding an overlap between the two reads which therefore can be merged into longer reads.

Prior to sequencing several DNA libraries were prepared by Karina Brinkrolf. In a first step genomic DNA was extracted from CHO-K1 cells followed by purification and washing steps (Figure 4.33 A). The obtained DNA molecules were subsequently fragmented into shorter samples (4.33 B) by mechanical sheering. To allow for later sequencing, two different adapter and sequencing primer pairs are ligated to the fragmented, linear DNA molecules (Figure 4.33 C), while each of these pairs

consists of the adapter/primer construct on the one strand (coloured boxes) and a non-functional sequence on the other strand (grey boxes). The ligation process is followed by a binding step which attaches both adapter sequences to complementary sequences on the flowcell (Figure 4.33 D), resulting in a characteristic U-shaped construct. The still double stranded DNA molecules are separated by a temperature shift which leaves only the adapter sequences attached to the flow cell (Figure 4.33 E). The resulting lawn of DNA strands on the flow cell can now be sequenced by employing the specific sequencing primers (Figure 4.33 F and G). The produced libraries (Table 4.4) were subsequently sequenced on an in-house GAIIx sequencing system with 125 cycles using the Illumina standard paired end protocol (Figure 4.32), which corresponds to a read length of 125 nt.

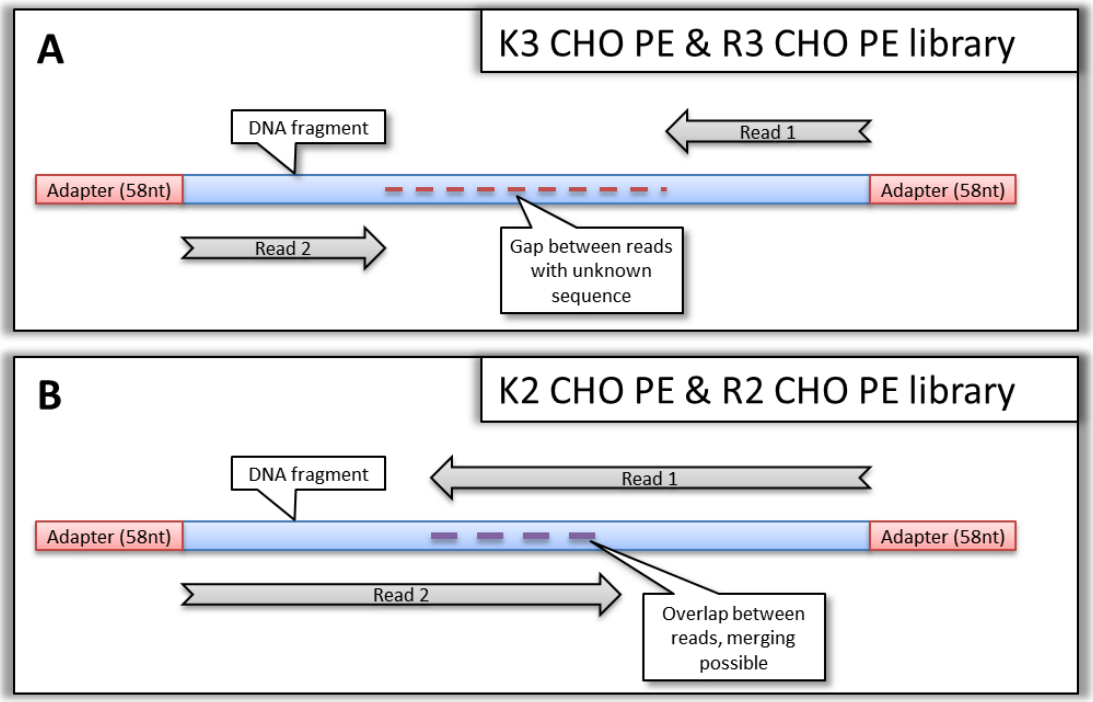


Figure 4.32: Overview of the libraries used for sequencing. Due to the paired end sequencing protocol, where the DNA fragment is sequenced from both directions, two different outcomes are possible. The fragment size refers to the size of the DNA template (blue) without any linker sequences (red). Dotted lines indicate areas of overlaps or gaps. **A)** Libraries K3 CHO PE and R3 CHO PE have larger fragment sizes (see Table 4.4), therefore the sequenced reads will not produce an overlapping sequence, resulting in a gap between both reads. **B)** Libraries K2 CHO PE and R2 CHO PE contain fragments of smaller size, therefore overlaps between read 1 and 2 of each pair are expected.

The reads produced by this experiment were initially intended for a *de novo* assembly of the CHO-K1 genome, thus the paired end protocol was employed to maximise the effective read length (Figure 4.32 B) and to produce additional

Library name	# Read pairs sequenced	Yield [Gb]	Used pairs [%]
K2 CHO PE	36,726,297	9.181	85.67
K3 CHO PE	30,095,262	7.523	83.01
R2 CHO PE	37,724,343	9.431	88.66
R3 CHO PE	32,107,071	8.026	84.86
R2 CHO PE	38,131,469	9.532	88.51
R3 CHO PE	31,108,180	7.777	84.52
	$\sum$ 205,892,622	$\sum$ 51.47	⌀ 85.88

Table 4.5: Results of the sequencing run using the prepared libraries. In total, six lanes were sequenced, yielding 205,892,622 read pairs (2×125 nt), which cumulates to 51.47 gigabases of data in total. In order to obtain optimal read quality and to discard non-matching reads for subsequent analyses, reads were mapped against the Chinese hamster ovary cell line (CHO-K1) reference genome [Xu et al., 2011] . The number given in the last column thus refer to the percentage of reads used for further analyses.

valuable meta information (Figure 4.32 A). While state of the art sequencing and assembly techniques would employ primarily mate pair reads (Section 2.1.3.2), in 2010 only a limited set of suitable assembly software was available and the scientific community had not yet developed mature protocols for *de novo* sequencing and assembly of higher eukaryotic organisms.

In total 51.47 gigabases of sequencing data distributed through 205,892,622 read pairs (Table 4.5) were produced. Quality control of raw reads was performed with the FastQC software, which allows for checks of quality values, linker residues or vector fragments [Andrews, 2012]. The results of the initial quality checks for each library are summarised in Figure 4.34 and 4.35. The analyses showed reasonable quality values for the first 100 bases, with a significant decrease for the last 10-25 bases for most libraries. The libraries generally show decreasing quality tendencies for each following library and additionally a degradation from read one to read two. While the degradation of the second read is owed to the chemistry and protocol employed for paired end reads, the decreasing qualities for the last 25 nt agree with previous experiences with the in-house GAIIx system. In order to compensate for the quality issues a mapping against the CHO-K1 draft genome sequence [Xu et al., 2011] was performed to discard faulty reads which may interfere with later assembly steps. Only reads mapping onto the CHO-K1 reference genome were used for further processing, all other reads were discarded (Table 5.2).

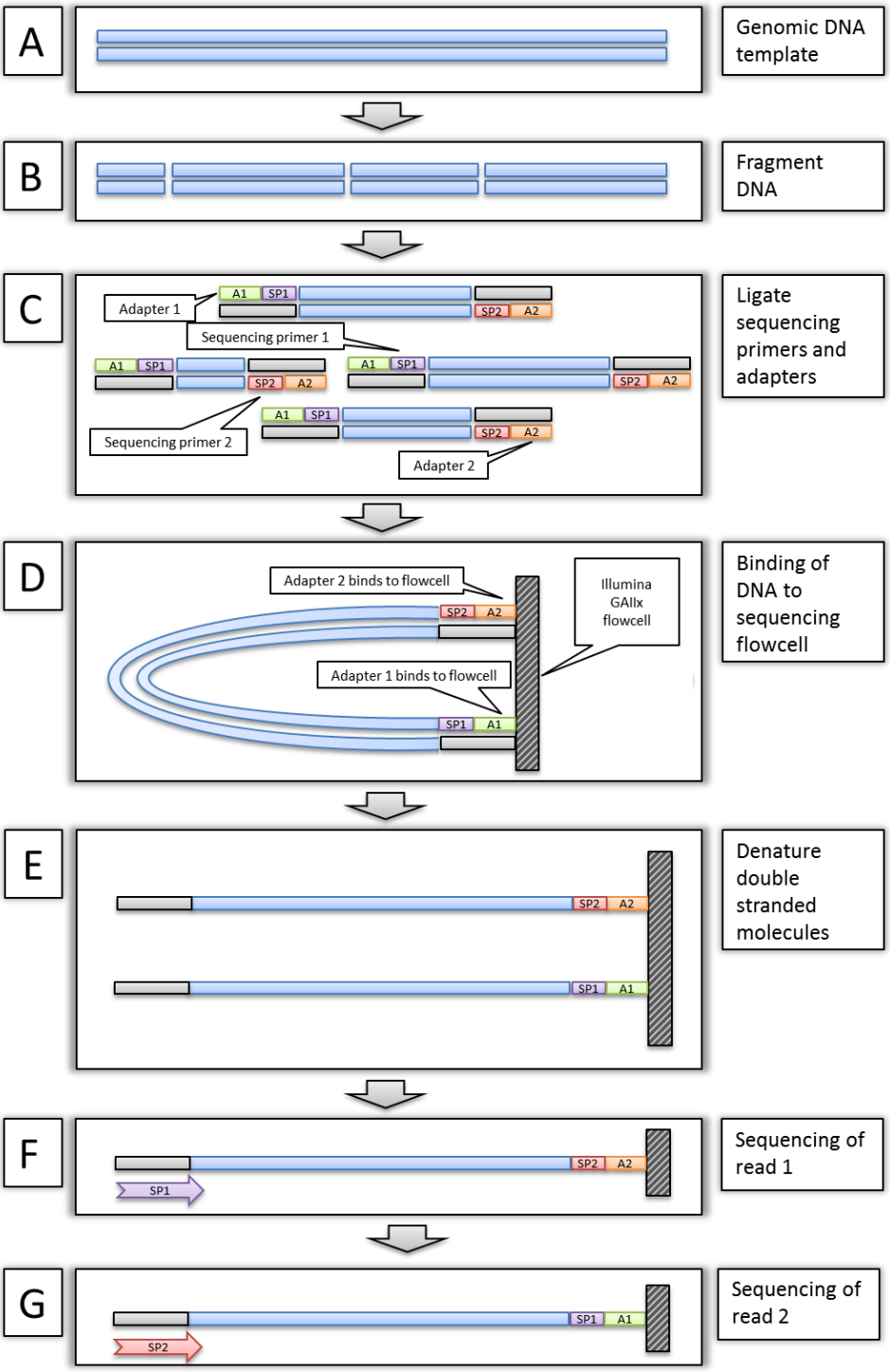


Figure 4.33: Library preparation for the CHO-K1 based sequencing run on the Illumina GAIIx sequencing system. **A)** Double stranded genomic DNA is the source material used for sequencing. **B)** The source DNA is fragmented into shorter sequences. **C)** Adapter 1 (green) and adapter 2 (orange), later anchoring the DNA molecule onto the flowcell, are ligated to the DNA samples together with two sequencing primers (SP1 and SP2, purple and red) required to start the sequencing process. **D)** Adapters ligated in the previous step retain the samples on the flowcell. **E)** The double stranded construct is denatured, leading to single stranded sequences fixated on the flowcell. **F)** Read 1 of 2 is sequenced from 5' to 3' using SP1 **G)** Read 2 of 2 is sequenced from 5' to 3' using SP2.

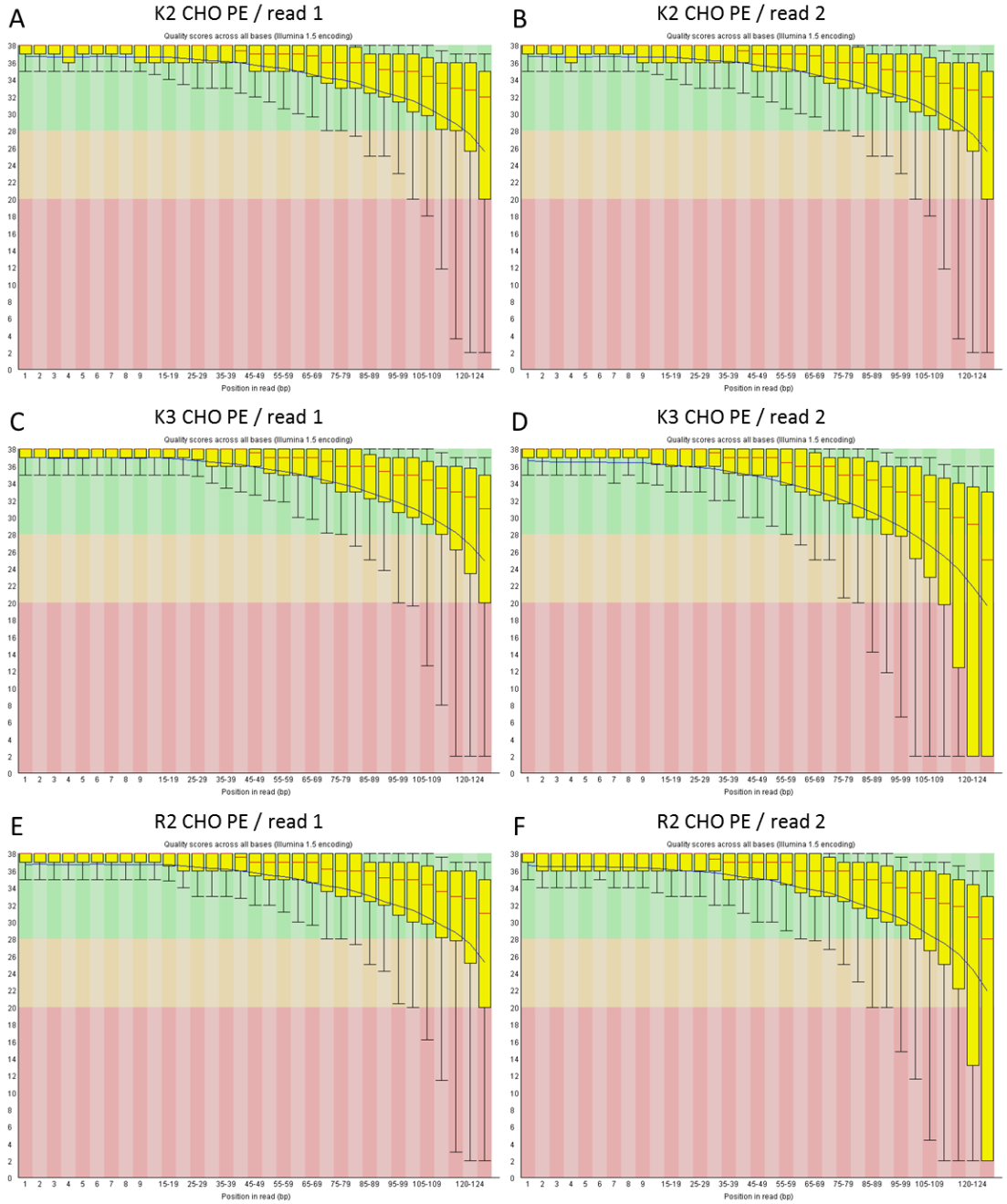


Figure 4.34: Sequencing quality assessment of the first three lanes performed with FastQC [Andrews, 2012]. The first column shows the quality for the first read of the pair, the second column contains the quality of the second read. The x-axis contains the position within the read (starting from 1 to 125), the y-axis contains Phred-based quality values (see Section 2.1.2). Green areas in the graph represent a 'good' quality, yellow reports average quality, and red regions indicate poor base quality. Again within the Box-Whisker plot, the red line is the median value, the yellow boxes represent the inter-quartile range (25-75%), upper and lower whiskers represent the 10% and 90% points and the blue line finally represents the mean quality. The library name is shown centred above each plot. In general, the second read always shows slightly lesser quality the first read, owed to the sequencing reaction. Especially **D)** and**F)** show significant outliers in the quality of the last base.

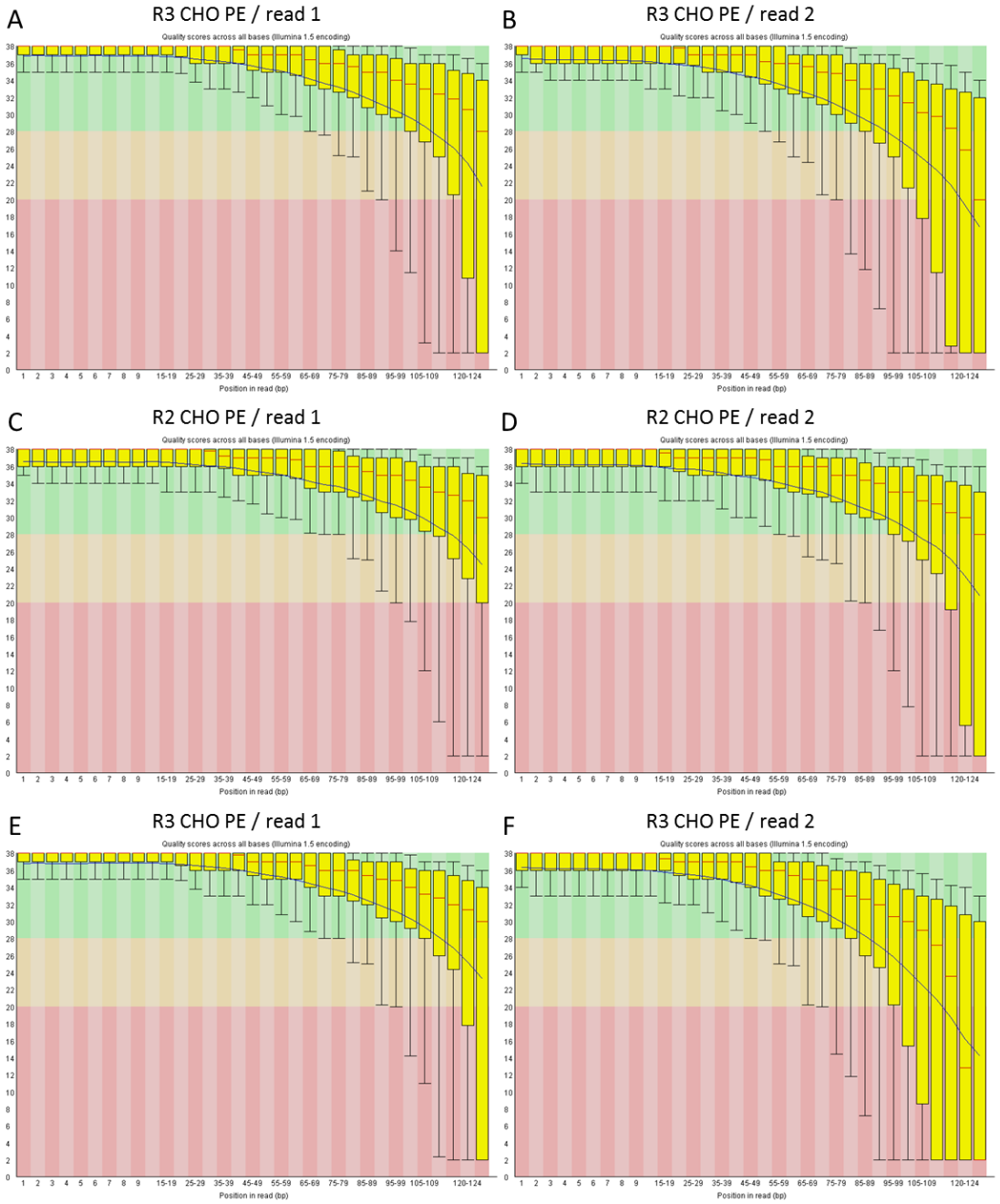


Figure 4.35: Legend identical to Figure 4.34. Compared to the first three lanes, quality for the second reads shows a negative trend leading to up to 6 bases with arguable reliability (**F**).

4.4.2.2 Evaluation of SATYR with an eukaryotic dataset

The CHO dataset is used to assess the performance of Mapsembler and SATYR with datasets typically found in eukaryotic genome projects. Unfortunately, Mapsembler in its most recent version (2.1.2) failed repeatedly with the CHO dataset after approximatively 4 weeks running time with a program error and without producing any assembly output. Analyses of the program during runtime with the `strace` debugging utility revealed that even after 4 weeks Mapsembler still was in the process of read indexing. As such only the results for the SATYR software developed throughout this thesis are presented and discussed within this section.

As SATYR requires seed sequences to start the assembly process, a suitable set of seeds was selected. These sequences were generated in a work by Becker et al. [2011] and represent 3,533 partially incomplete cDNA sequences originating from the same cell line as the genomic DNA sequenced within this work. SATYR was set up to extend all seeds as far as possible with a minimal coverage of two reads. To accelerate the assembly process for this larger set SATYR's multi-threading capability was exploited to use all of the 48 available processor cores.

The general setup for the CHO dataset evaluation is the same as for the previous prokaryotic dataset. However, the seed sequences used for the CHO dataset are significantly longer with an average length of 22,584 nt and a maximal length of up to 174,210 nt compared to the fixed seed length of 36 nt for the artificial *Escherichia coli* dataset. Therefore, instead of considering the complete construct (5' extension, seed, and 3' extension) analyses were performed for 5' and 3' extension while discarding the already known seed sequence from further processing.

4.4.2.3 Assembly analyses - similarity

In a first step of the evaluation the focus was set on sequence similarity and coverage of respective reference sequence which was evaluated with the BLAST software suite. Compared with the first evaluation employing an *Escherichia coli* dataset with 162-fold coverage first impressions show lower average coverage values for the CHO dataset (Figure 4.36). Although the CHO reads were preprocessed to eliminate interference with the assembly, this pre-filtering also reduced the overall genome coverage to approximatively 16-fold for the CHO dataset. This is a factor of 10, not even including the much more complex genomic structure of the Chinese hamster. While 80-fold coverage represented the lower border for the *Escherichia coli* dataset, this mark is not reached by the most-covered regions for $k = 15$. Generally, extensions for 3' and 5' perform similar without larger deviations. While the overall trend throughout the increasing kmer values is decreasing, for a single kmer (92) a peak in coverage can be observed. This peak also stresses

the importance of the right choice for the kmer value and the fact that often no clear trend for the correct kmer choice can be extrapolated from a few test runs of assembly software.

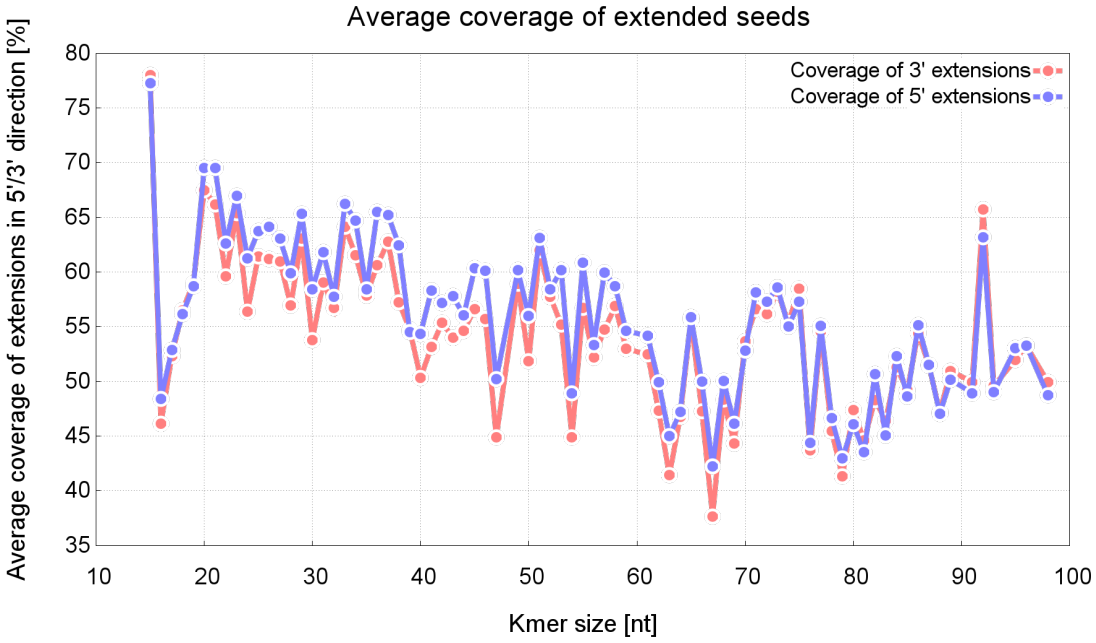


Figure 4.36: Graphical representation of the average coverage for all 5' and 3' extensions for a given kmer size. The coverage of the reference sequence provides information about the structural accuracy of the assembly.

The above statement is additionally validated by the graphical representation of the percentage of identity (Figure 4.37). The values shown in the representation indicate the level of basepair similarity between the covered parts of the reference (Figure 4.36) and the reference sequence. Similarity generally is located between nearly 100% for $k = 15$ and several longer kmer sizes (70, 80, 81) and 97% for $k = 45$. There are, however several outliers which show similarity measurements below 96% ($k = 42$ *and* 61). Although 84 different kmer values were tested for this evaluation, no clear trend for similarity can be observed apart from the increasing tendency to produce more outliers for larger kmer sizes.

4.4.2.4 Assembly analyses - extension length

In contrast to the similarity measurements where no clear trend could be observed, the analyses of the extension lengths shows more concrete effects of the kmer size on the length of extensions (Figure 4.39). Kmer sizes ≤ 20 on average produce very short extensions of less than 500 nt. The average length increases further to up to 800 nt for $k = 29$, starts to decrease for $k = 37$ while reaching a level similar

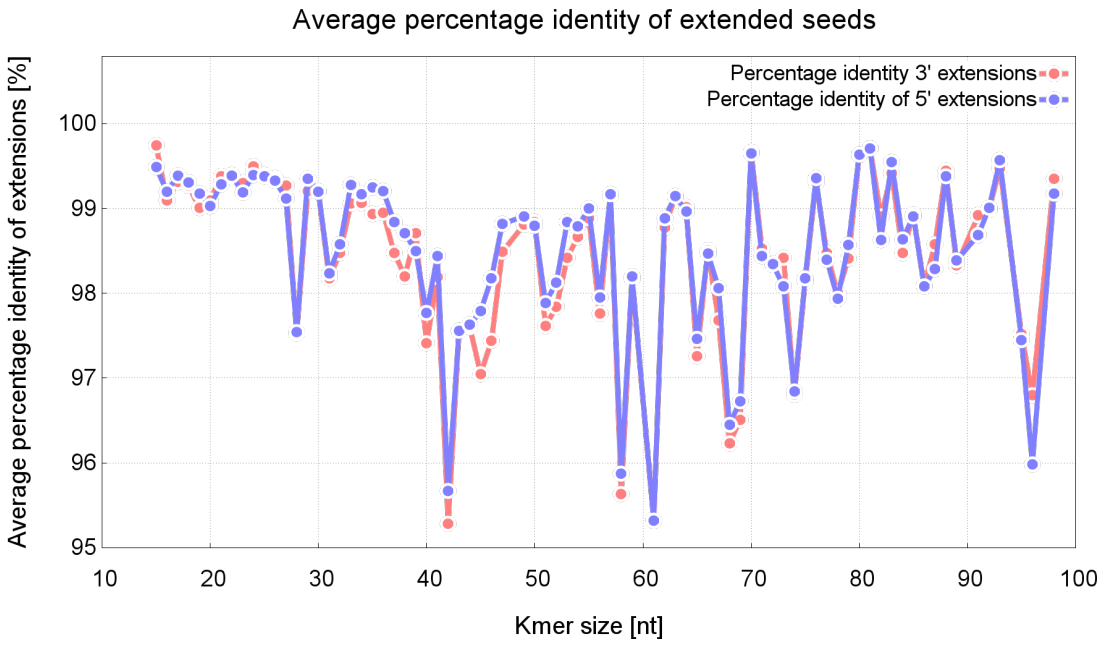


Figure 4.37: Graphical representation of the average percentage of identity for all 5' and 3' extensions for a given kmer size. Higher values indicate more similarity to the reference genome. The identity value can be used to assess an assembly on nucleotide level, compared to the structural assessment of coverage values.

to the start level around $k = 65$. Interestingly, the number of outliers for $k \geq 50$ decreases, which may be related to the lower number of reads returned by the BW_search algorithm due to the large kmer values and therefore most extension terminate after a few iterations.

The trend established by the average extension length is continued to a certain degree by the maximal extension length. As expected, very small kmer sizes produce relatively short extensions, owed to the higher probability of reaching the maximal hit threshold. For medium kmer sizes from $= 30$ to $k = 70$ the rate of outliers increases, while for large kmer sizes the number of outliers as well as the maximal extension length decrease. This decrease can be explained in by the smaller number of reads possible to extend and therefore a higher probability of breaking the iterated assembly within a few cycles. While the maximal measured extension length of 14,024 nt was produced with a kmer size of 33, 14 kb correspond approximatively to the average extension length achieved by larger kmers of the *Escherichia coli* dataset.

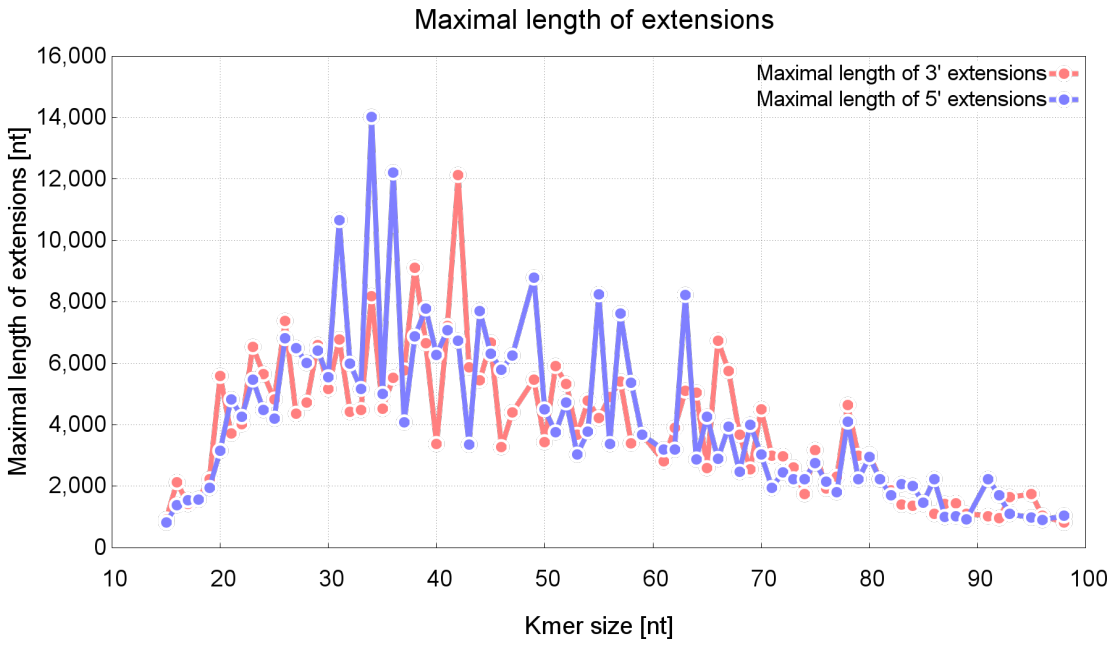


Figure 4.38: Diagram of the maximal length reached by assembled extension for SATYR. The kmer size is shown on the x-axis, the y-axis shows the length of the maximal extension.

4.4.2.5 Number of extended seeds

Due to the large deviation in coverage and genome complexity, the most obvious question is how many of the supplied seeds actually can be extended. While for the *Escherichia coli* dataset in nearly all cases 100% of the seeds could be extended this should not be assumed for the CHO dataset. Additionally the seed sequences consisting of cDNAs represent spliced versions of the underlying genome sequences, therefore 5' and 3' end sequences of the seeds may contain splice junctions, which make it impossible to find matching genomic reads spanning those regions.

4.4.2.6 Program run times and memory consumption

Mapsembler was not able to complete the evaluation. The execution of the program terminated after roughly 4 weeks of running time with a constant memory usage of 3.2 GB. During this time the Mapsembler process constantly occupied one CPU and was confirmed to still import reads.

SATYR required 161 GB of main memory to store the whole read index, which is the same for all kmer sizes. While the creation of the index took around 90 hours on a dedicated system, the running time for the actual assembly process averaged around 10 minutes per kmer size. This is in contrast to the run times measured for

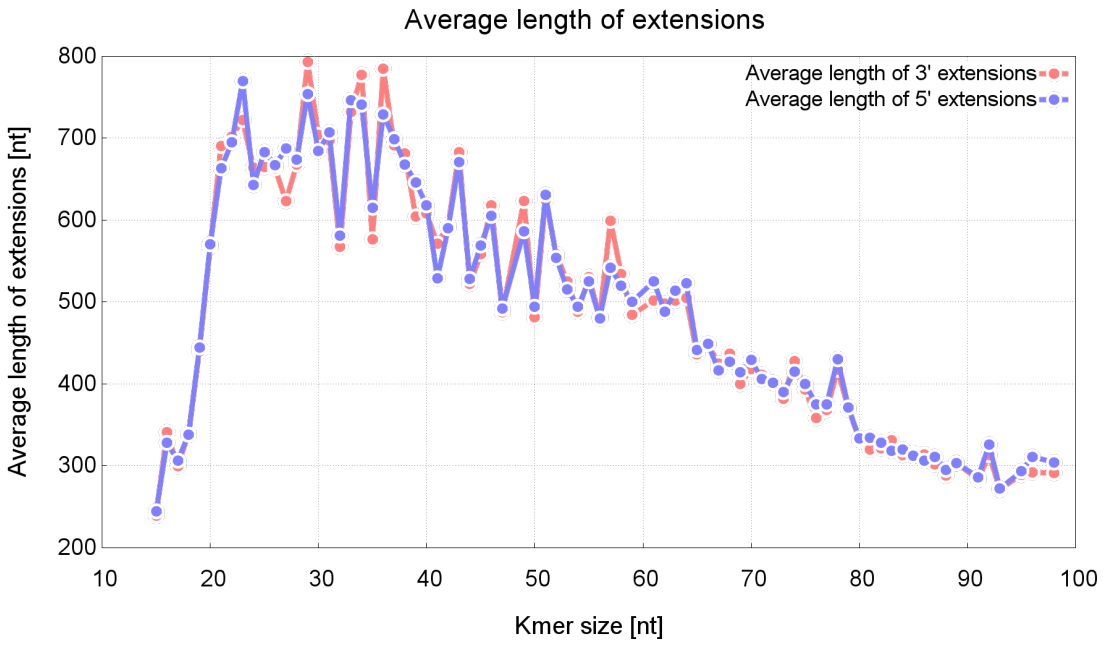


Figure 4.39: Diagram of the average length reached by assembled extension for SATYR. The kmer size is shown on the x-axis, the y-axis shows the length of the maximal extension.

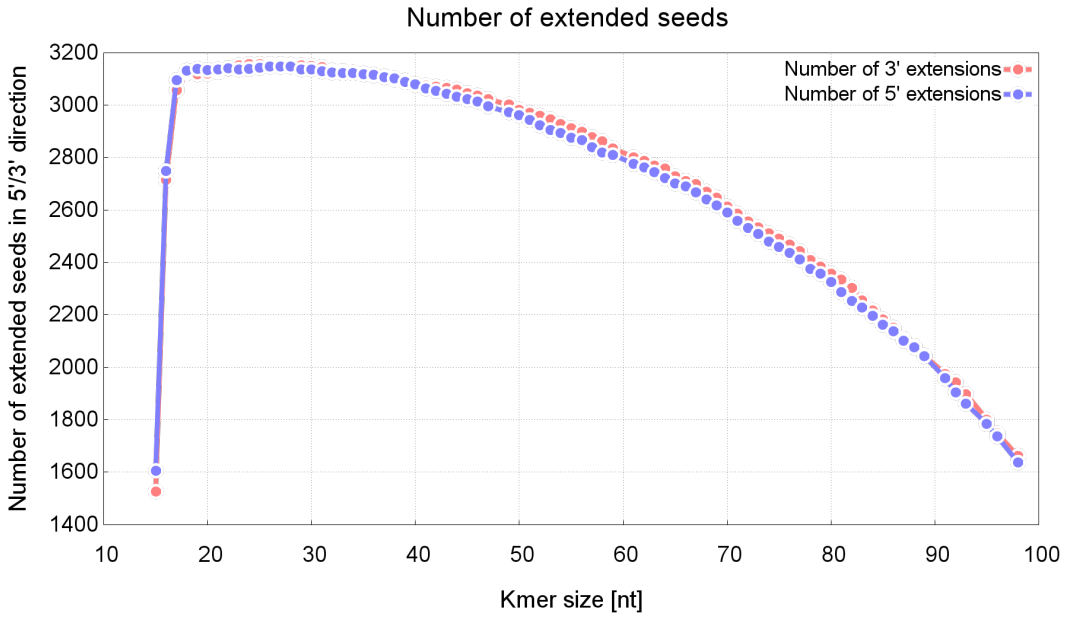


Figure 4.40: Graphical representation of the average coverage for all 5' and 3' extensions for a given kmer size. The coverage of the reference sequence provides information about the structural accuracy of the assembly.

the *Escherichia coli* dataset, where the number of seed sequences was more than three times higher.

4.5 Targeted assembly on eukaryotic scale

Several targeted assembly-related tools were introduced throughout the chapter and their applicability to the given requirements was assessed in detail. Out of all presented tools, only Mapsembler developed by Peterlongo and Chikhi [2012] fulfilled all four requirements (targeted assembly, seed-based assembly, NGS capable, extension of seeds). The SATYR (**T**argeted assembly of **Y**ield increasing **R**egions) software tailored specifically to extend given cDNA seeds in 5' and 3' direction, while the capability for large next generation sequencing data sets was ensured by using a Burrows-Wheeler transform-based index structure comprising all sequencing reads. SATYR was able to extend seeds of an *Escherichia coli* dataset significantly longer than Mapsembler, but in turn showed lower similarities than Mapsembler. Although Mapsembler's publication contains a use case with eukaryotic sequencing data, the software was not able to cope with the $\geq$ 50 gigabase Chinese hamster dataset which in contrast could be processed with SATYR. Within the evaluation it became evident that the quality and quantity of available sequencing data have significant impact on assembly performance and consensus quality, therefore requiring scalable algorithms able to deal with the required amounts of data.

CHAPTER 5

Identification of transcription start sites in the Chinese hamster genome by next-generation RNA sequencing

5.1 Previous approaches

Over the years several efforts tackling the problem of transcription start site detection using different analytical methods have been published. The fraction of organisms for which in-depth studies of promoter regions and transcription start sites have been carried out, however, is still limited and has a focus on *Mus musculus* and *Homo sapiens* as mammalian representatives [Sandelin et al., 2007], *Drosophila melanogaster* for insects [Ohler et al., 2002], as well as several plant species [Kumari and Ware, 2013]. Following successful studies with the cap analysis of gene expression (CAGE, see Section 2.2.4.2) protocol for instance by Carninci et al. [2006], several refinements of the initial CAGE method have been developed. While nanoCAGE dramatically lowers the amount of RNA required for analysis, CAGEscan allows for the identification of sequences upstream of the transcription start site which may not have been been assigned solely on basis for the 21-23 nt short CAGE tags [Plessy et al., 2010]. The DeepCAGE protocol is an adaptation of the original protocol to 454, Illumina and SOLiD sequencing systems [de Hoon and Hayashizaki, 2008; Valen et al., 2009] and aims at a further increase of throughput. The most recent development in terms of CAGE was the so called HeliScopeCAGE [Kanamori-Katayama et al., 2011] approach based on the equally named single molecule sequencing system, which however is not available any more since to Heli-

cos BioSciences went out of service in 2012. Common to all these works is the fact that so far no study was conducted for the industrially relevant promoter regions of the Chinese hamster.

5.2 A dual-library RNA sequencing approach for TSS detection

Due to the importance of promoters in biotechnological context and the lack of significant public contributions specifically for CHO cells a state of the art technique for TSS identification in large mammalian genomes was required. Since all currently available sequencing systems offer more than enough output in order to easily sequence whole transcriptomes within one run (see chapter 2.1 for a comparison of sequencing systems), the importance of a suitable sequencing protocol proved to be essential. After careful consideration, the choice was made for dual-library setup as offered by Vertis Biotechnologie AG (Freising, Germany). Basically two 5' enriched cDNA libraries were constructed. Library one, in the following referred to as "peak library" contains only cDNA which had its 5' cap removed by tobacco acid pyrophosphatase (EC 3.6.1.-), while library two, the "background library", was left untreated. The removal of the 5' cap in one of the two libraries is crucial, as the presence of the cap inhibits the amplification and therefore the sequencing of correct 5' mRNA ends, thus leading to a library consisting only of background mRNAs. In combination with the TAP-treated peak library a bioinformatics pipeline is able to remove false positive TSS peaks identified by peaks in both, background and peak library. In comparison to previous CAGE or SAGE protocols the overall library setup was simplified, as no restrictions and ligations of tags are needed.

5.2.1 Cell line and culture conditions

Cultivation and harvesting was performed by Stefan Northoff, member of the Institute for Cell Culture Technology at Bielefeld University. CHO-K1 cells (ATCC CCL61) adapted to serum-free media and cultivation in suspension were cultured in 250 mL Erlenmeyer flasks using TC-42 (TeutoCell AG, Bielefeld, Germany), a serum free, chemically defined media, containing a recombinant protein measured below 1 mg/L. The medium was inoculated with 6 mM glutamine originating from a 200 mM stock solution. Cultivation parameters were set to 37°C, 5% CO_2 content, and a relative humidity of 80%, where the shaking device worked at an oscillation of 5 cm at 185 rpm and the initial volume was 130 mL. Samples with a cell count of $4{\times}10^7$ were taken on days 2 to 8, centrifuged at 115 × g and the pellets were stored at -80°C. For all analyses within manuscript, material from day 2, 6, and 8 was pooled (black dotted time points on black lines in Figure 5.1).

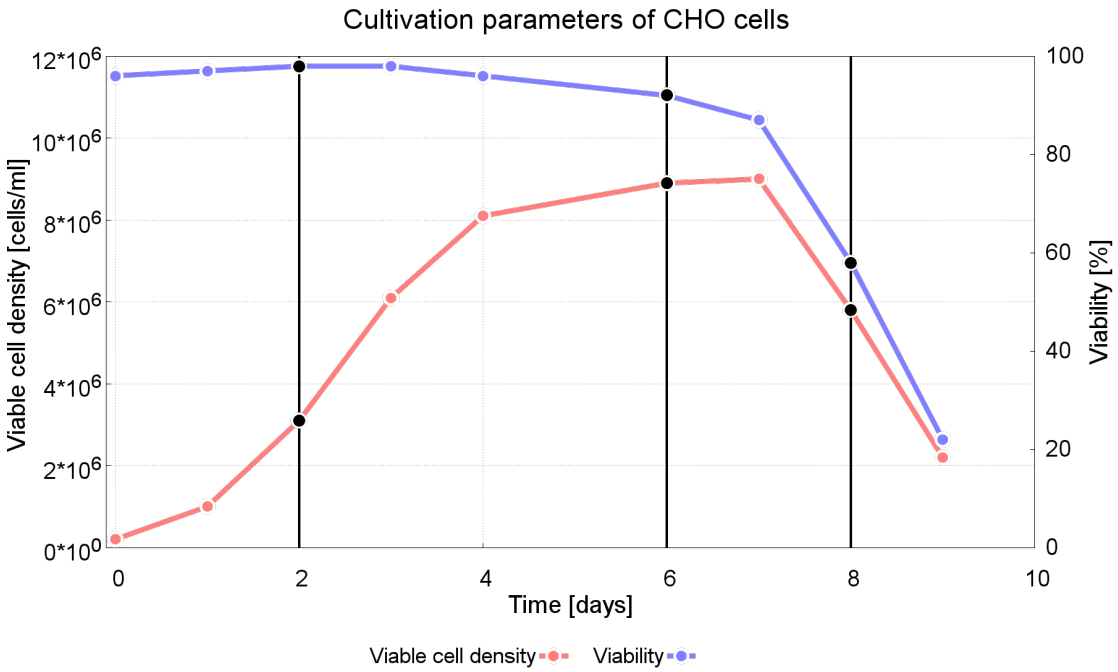


Figure 5.1: Cultivation of CHO cells (ATCC CCL-61) used as source for cDNA samples throughout this experiment. Viable cell density is shown on the left y-axis in cells/ml, the corresponding time point of sampling is shown on the x-axis, the right y-axis shows the viability of the cells throughout the cultivation. Black dots indicate sampling points (days 2, 6, and 8) used for sequencing.

Sample	Peak library	Background library
TAP treatment	yes	no
Barcode	CAACTA	GTGAAA
PCR cycles	26	26
cDNA concentration	41 $^{ng}/_{\mu l}$	16 $^{ng}/_{\mu l}$
% of pool	70	30

Table 5.1: Summary of results from RNA sequencing library preparation for both libraries prior to sequencing on a HiSeq 2000 machine.

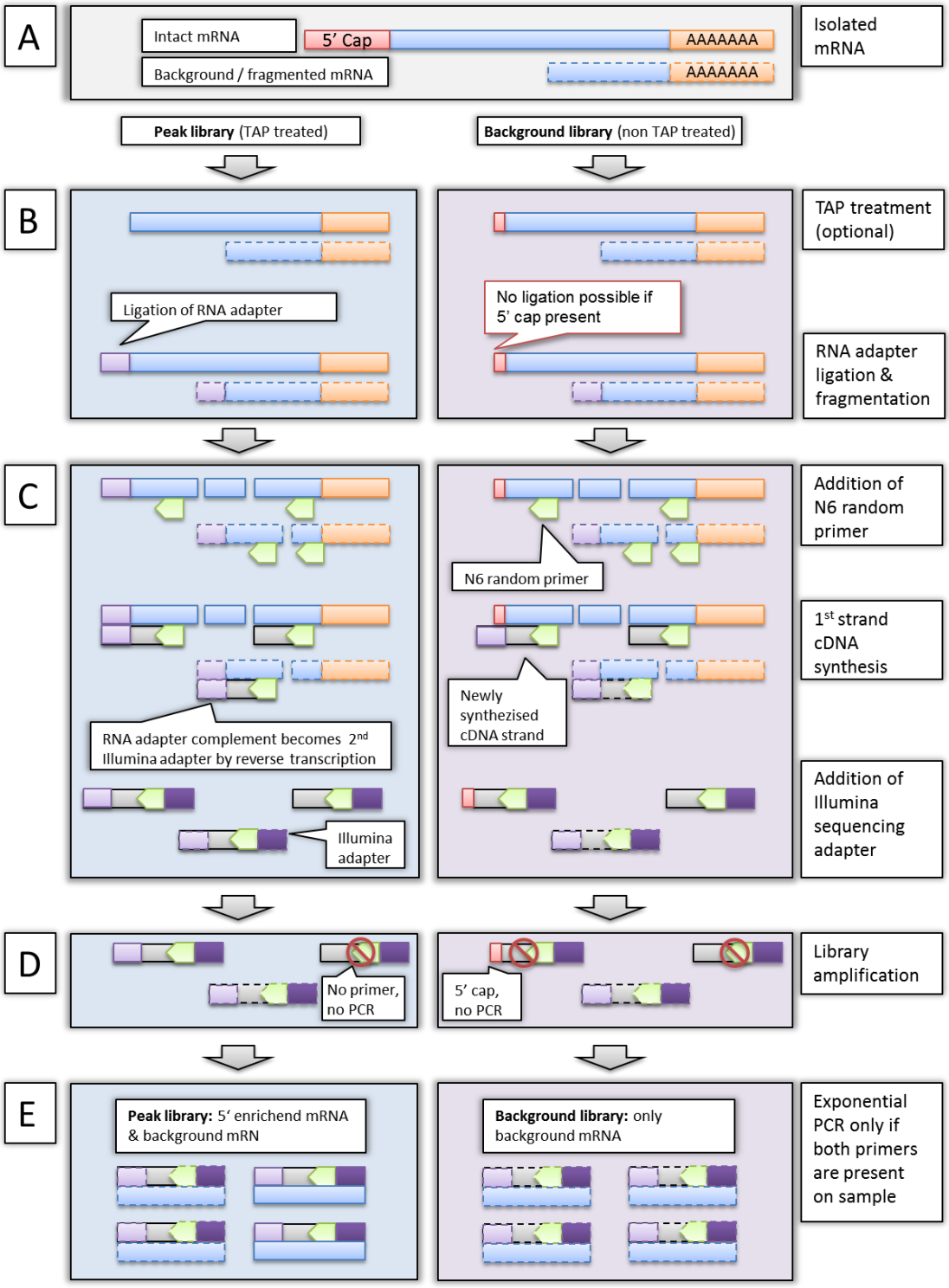


Figure 5.2: Two sequencing libraries were prepared in order to distinguish between true TSSs and background noise. The peak library was treated with tobacco acid pyrophosphatase (TAP) and includes sequences representing true TSSs, as well as sequences originating from fragmented mRNAs. The background library was not treated with TAP and includes only sequences from fragmented mRNAs.

5.2.2 Library construction and sequencing

Instead of a single library containing all data, a dual library approach was chosen, in which the sample DNA is distributed to two libraries subsequently receiving different treatments. These two 5' enriched cDNA libraries were constructed at Vertis Biotechnology AG (Freising-Weihenstephan, Germany). All samples were combined into one cDNA pool in order to receive a broader perspective of active transcripts during different phases of cultivation. Afterwards, total RNA was extracted from the pool using a mirVana miRNA isolation kit (Ambion/Life Technologies GmbH, Darmstadt, Germany). RNA preparation lead to an RNA concentration of 989 $^{ng}/_{\mu l}$ and an total amount of 36.6 µg. In order to start 5' cDNA synthesis all RNA molecules exhibiting a mono-phosphate at the their 5' end were degraded by a terminator exonuclease (Epicentre, Madison, U.S.A.). Thereafter, the sample pool was divided in a 70:30 ratio, while the 70 % fraction was treated with tobacco acid pyrophophatase (TAP) and the remaining 30% were left untreated (Figure 5.2, step B). A first adapter, later becoming the second Illumina sequencing adapter when reverse transcribed is ligated to the 5' ends of the sequences, succeeded by a fragmentation step (2 pulses, 30 seconds, 4°C, Figure 5.2, step B). A N6 random primer was added to start the 1st strand cDNA synthesis carried out by M-MLV reverse transcriptase (Ambion/Life Technologies GmbH, Darmstadt, Germany), followed by the addition of the Illumina sequencing adapter to the 5' ends of the antisense strands (Figure 5.2, step C). The subsequent PCR (polymerase chain reaction) conducted 26 cycles; exponential amplification only was achieved for sequences carrying both sequencing adapters and hence sequences with remaining 5' caps (background library) did not reach a significant amplification (Figure 5.2, step D). In a last step, both samples were again pooled and cDNAs ranging from 250-500 bp were selected from agarose gel. Sequencing was eventually carried out on an Illumina HiSeq 2000 sequencing system. All library data available before sequencing is summarized in Table 5.1.

5.3 Bioinformatics analysis pipeline

5.3.1 Overview of pipeline modules

By the time this manuscript is written, no specialized bioinformatics pipeline for the analysis of promoter-centric experimental data was publicly available. The customized nature of the dual-library sequencing approach in combination with the large amount of data associated with genomes in the gigabase-range, as the Chinese hamster, made the development of such a pipeline one of the main targets of the study. The pipeline itself can be divided into seven distinct steps grouped into three modules. Each of the modules has a specific task, handled by a tool chain specific for the module. Module A (Figure 5.3 top) conducts preprocessing tasks and outputs a list of candidate transcription start sites, module B (Figure 5.3

middle) handles transcription start site annotation, and module C finally performs a set of promoter analyses (Figure 5.3 bottom).

5.3.2 General implementation

The majority of the software was implemented in Perl, the practical extraction and report language, widely used for bioinformatics as well as many other data manipulation intensive tasks. Perl is one of the constants of scripting languages being in active development for nearly 30 years by now. As Perl is a typical "glue-language" used for building pipelines of other software tools, it is very well suited for the purpose of this work. Some parts of the well known BioPerl library [Stajich et al., 2002] were employed for graphical output handling. Additionally a small set of Bash scripts was included mainly as wrappers for external programs from within the pipeline. The flowchart [Booch, 2006] of the pipeline is shown in Figure 5.3. Input RNA sequencing reads are shown as green parallelogram, the first and last steps are shown as rounded rectangles, all other steps are represented by grey rectangles. Smaller orange rectangles display the executed program, red rectangles include the kind of analysis carried out, blue parallelograms indicate generated output. The central decision point has purple colour, all steps are interconnected by black arrows determining the direction of information flow. Although the pipeline was developed on Linux systems, it is also fully functional on Solaris or any other Unix-based operating system providing the required Perl environment. The software is not dependent on any kind of database neither server nor flat-file based. All output is generated in form a tab separated values (TSV) and as such human readable and easy to parse. All graphical output is produced in form of resolution independent scalable vector graphics (SVG).

5.3.3 Preprocessing

Due to maintenance and usability reasons the pipeline has three separated stages distributed through several scripts. Within module A, all preprocessing needed for further analysis of the raw read data is carried out. Several external tools are employed for this module, reaching from raw read modification tools to read mapping tools and postprocessing software. Prior to any further analysis the clipper component of the FASTX-Toolkit [Andrews, 2012] scans for remaining sequence fragments originating from Illumina linker or primer sequences, as contaminated reads introduce erroneous mapping positions in later steps. Usually this purging removes a few percent of all reads, leaving a large portion of reads suitable for further processing. Subsequently low quality reads are filtered out using the filter component of the FASTX software suite, as these would also interfere with later analyses.

Mapping of the reads against a reference genome is one of the central tasks in transcription start site identification. Given a read length of 36 or even 50 nt an

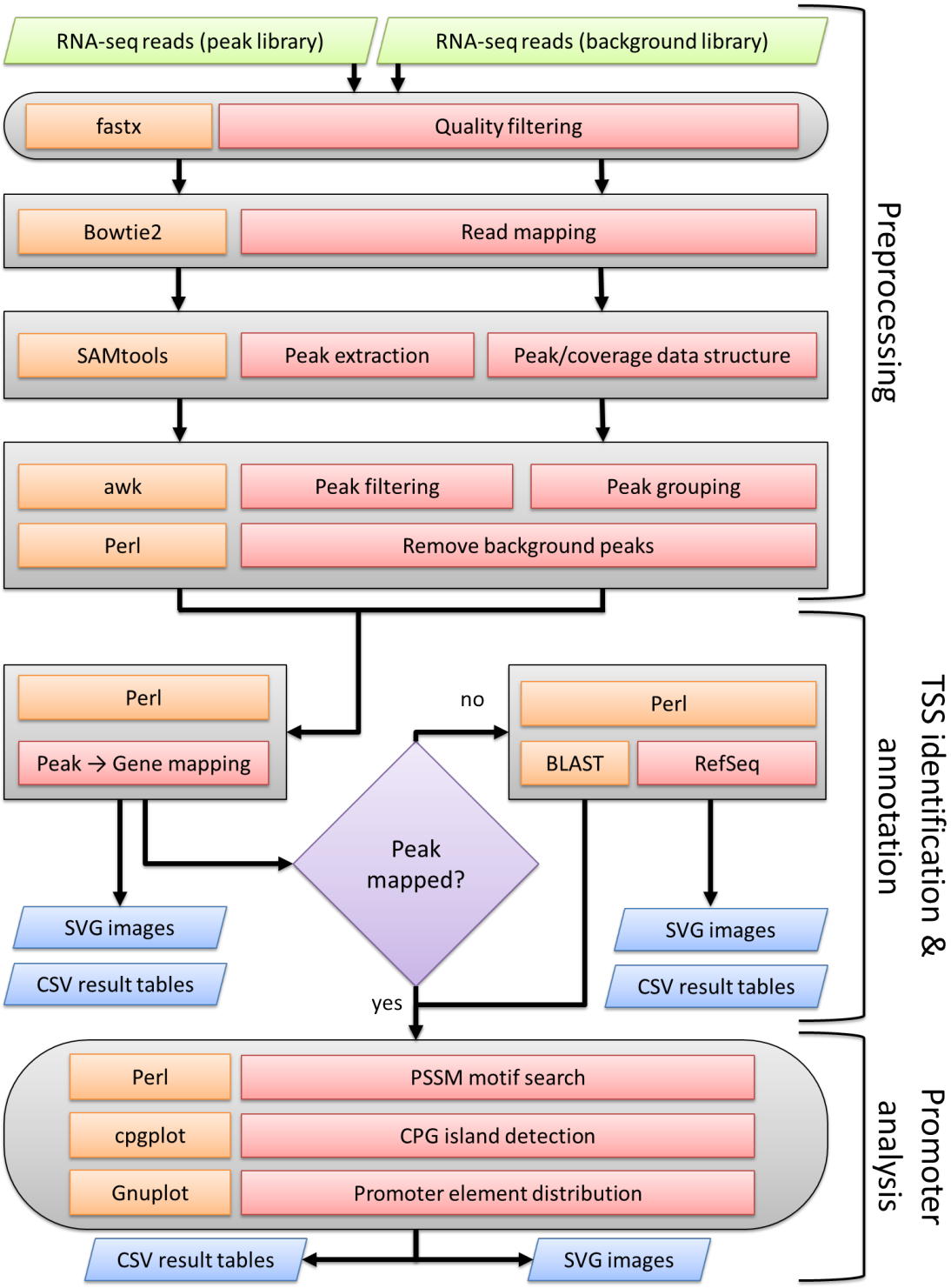


Figure 5.3: RNA sequencing bioinformatics pipeline. The pipeline has seven consecutive steps that are divided into three modules: preprocessing of raw data, TSS identification and annotation, and promoter analysis. Analyses of each step are summarized in grey boxes. Further colours used: Green (data input steps), orange (programs executed), red (kind of analysis), blue (output steps), purple (points of decision), and black arrows (direction of flow of information).

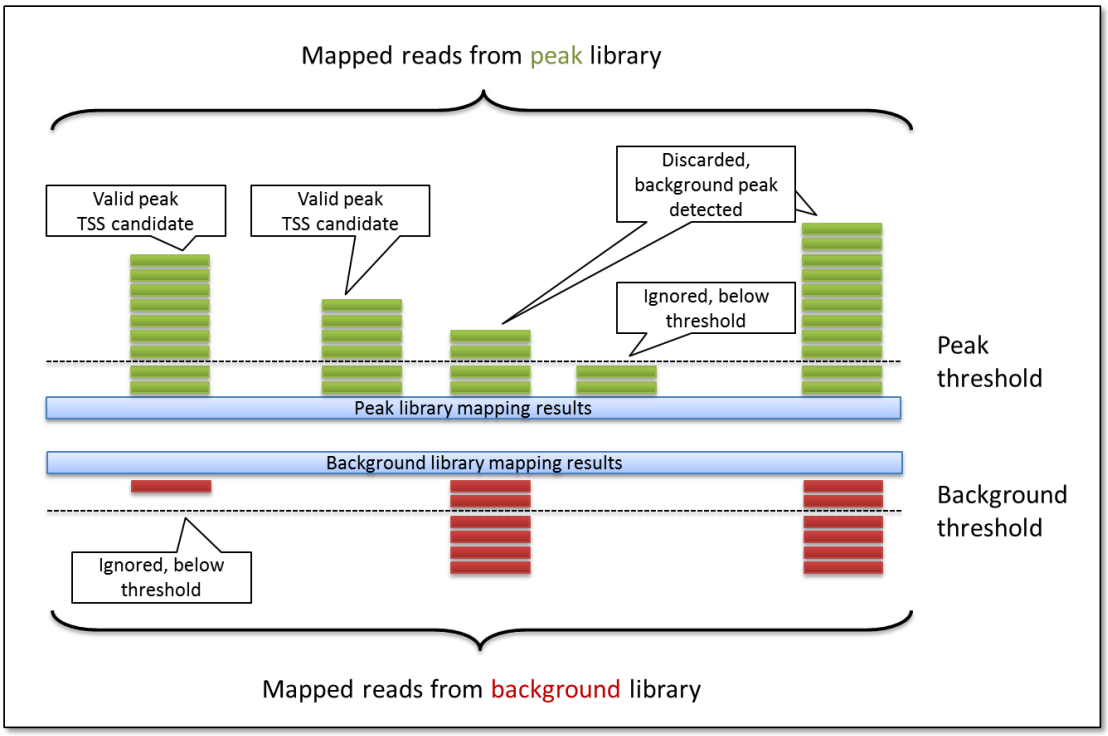


Figure 5.4: Peak merging process within the preprocessing module. Reads from background (red) and peak library (green) are processed separately until peak merging. Thresholds (dotted lines) filter out possible random mappings. For each read stack of the peak library the list of background peaks is checked for a corresponding peak. If no such peak exists a potential TSS is found, otherwise the peak is discarded.

unique assignment to one specific position in the genome is generally feasible, although in special cases like repeats or orthologous genes non-unique mappings may occur. Read mapping within this pipeline is carried out by Bowtie2 [Langmead and Salzberg, 2012]. The software is robust, fast, and well established [Fonseca et al., 2012]. The tool is also able to process the genome of the Chinese hamster as reference, which can be classified within the same size category as the human genome (Chinese hamster: 2.3 Gb [Brinkrolf et al., 2013], human genome: 3.2 Gb [Istrail et al., 2004]).

In a next step, the SAMtools [Li et al., 2009] software package is employed to extract features from the mapping results that are essential for further analyses. Start and stop positions of each mapped read are extracted, together with strand information to assign TSSs to the correct strand. In the following, reads are grouped according to their start and stop position in combination with strand information. Essentially, this step generates a list of read stacks, while coverage information allows for a coarse relative ordering of the stacks in terms of transcription activity.

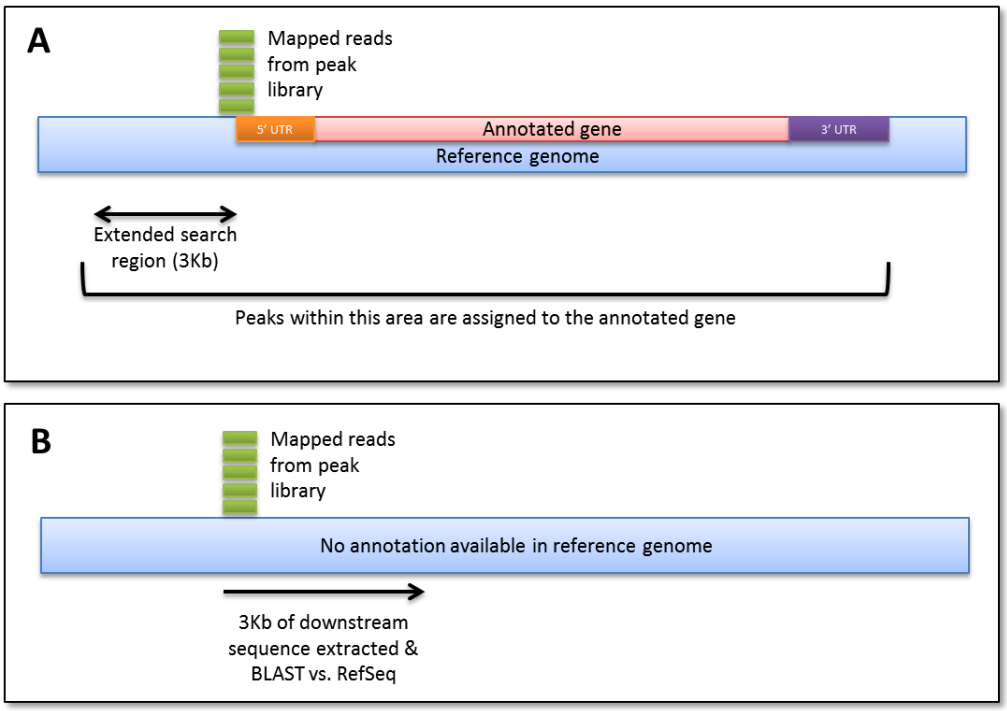


Figure 5.5: The two-phased peak annotation process within the equally named module. **A)** In a first step a region starting 3,000 nt in front of annotated genes and stretching until the end of the annotated region is screened for potential TSS peaks. The 3,000 nt extra space compensate for mispredicted 5' untranslated regions (UTR) and possible wrong annotated gene starts. **B)** In several cases TSS peaks will not be assigned to any CH annotated gene due to the draft status of the Chinese hamster reference genome. Therefore a BLAST vs. RefSeq search is conducted with 3,000 nt of sequence extracted in downstream direction starting from the TSS position.

Preprocessing concludes with merging of stacks from peak and background library (Figure 5.4). Reads of both libraries were preprocessed separately, but are now merged to a final set of reads stacks or "peaks". The list of stacks from the peak library is processed iteratively, in which the coverage of reads must exceed a given threshold. For each peak above the threshold, the background peak list is checked for corresponding peaks within the same start/stop range. In case such a background peak is found, the initial peak is discarded. If no background reads were found to map at that specific position or if the number of background reads does not exceed the threshold, the peak is saved for later analysis and is a potential TSS candidate. Final output of the preprocessing module is a data structure consisting of candidate TSS positions, their coverage, and strand information.

5.3.4 Transcription start site identification and annotation

The promoter analysis module takes the list of candidate TSS positions generated by the preprocessing module and assigns annotations based on available reference data and extrinsic approaches. The first phase of the module, depicted in Figure 5.5 A) uses the supplied reference genome to add as much information to a peak as possible. The GenBank format [Benson et al., 2008] stores sequence information as well as additional features of the sequence, such as coding regions, assembly gaps or gene functions.

After caching the annotation information each candidate peak is processed. Start and stop position of the peaks are screened for matches within the cached positions for annotated regions. In order to compensate for incorrect *in silico* gene predictions or to short 5' untranslated regions (UTRs) a 3,000 nt safety margin is added in front of each annotated gene (see Figure 5.5 A for details). It is important, to not only screen the 5' regions for TSS peaks as the experiment might also have generated peaks originating from alternate start positions within the gene. This might happen when a secondary transcript variant does not start with the first (annotated) exon, but with a posterior exon. In case of a successful hit for the TSS position, the second annotation phase is skipped, as all information already is available. However, for numerous TSS candidate peaks no corresponding gene annotated for the Chinese hamster was found, primarily owed to the draft status (i.e. incompleteness) of the CH genome.

It is reasonable to assume that many of the genes missing an annotation in the hamster genome can be found in related species, such as *Rattus norvegicus* or *Mus musculus*. As such, a BLAST (version 2.2.28+) [Altschul et al., 1990] database was set up from files containing data from the RefSeq database (release 62, [Pruitt et al., 2007]). RefSeq is a collection of curated, non-redundant transcript sequences of various organisms (also available for genomes and proteins), therefore assuring a certain level of quality for the entries. In this second phase 3,000 nt of sequence is extracted from the reference genome, starting downstream of the candidate TSS position. These 3,000 nt are then screened for sequence similarities to transcripts in the BLAST database. The e-value threshold used is based on input sequence length and varies between 1×10^{-5} for fragments $\leq$ 500 nt and 1×10^{-20} for fragments $\leq 2,000$ nt. Results of the query are analysed and the most probable record is chosen as representative for the corresponding peak. Additional information such as gene name, functional annotation or pathway affiliation are extracted if available.

Peaks not receiving an annotation from neither the reference nor from RefSeq similarities are either false positive hits, or genes unique to the Chinese hamster but missing from all other RefSeq-listed organisms. This probability, however, is relatively low, as such these genes are neglected. In contrast to the preprocessing module, for the TSS identification graphical output is provided. The SVG-based

images show each TSS peak with its characteristic peak shape within its genomic neighbourhood of annotated genes.

5.3.5 Promoter analysis

The third module of the bioinformatics pipeline enriches hitherto results with several promoter-centric analyses. As for the TSS identification, base data from the preprocessing module is used as starting point for all analyses. Although the typical core promoter extends from -40 to +35 relative to the TSS (Figure 2.4, Chapter 2.2.1) 100 nt of upstream and downstream sequence based on the identified position of the transcription start were extracted from the reference genome. Indeed, localisation of the TSS on a basepair-exact basis is not possible in all cases, due to different shapes of transcription start sites [Carninci et al., 2006]. Therefore an adequate safety margin is added to compensate for deviating peak shapes.

5.3.5.1 Position weight matrices (PWMs)

To identify possible regulatory elements within the core promoter, a filter methodology is required to classify sequences. One solution to this problem, developed by Stormo and Schneider [1982] are position weight matrices (PWMs). PWMs are constructed from a set of input sequences and were developed as an alternative to consensus sequences. Although a consensus sequence may represent a set of sequences, a consensus is normally not able to conserve all information of the original set due to its "compressed" kind of illustration. The use of PWMs in contrast, regains almost all features of the original sequence set, taking also underrepresented sequence features into account. A PWM is a $\mathcal{A}\times l$ matrix, with $\mathcal{A} = A, C, G, T$ in the case of DNA sequences and l is the length of the longest sequence in the sequence set S the PWM is constructed of. Given a set S of $N = 10$ sequences of length $l = 9$ (Equation 5.2), Equation 5.1 is used to compute each cell of the matrix M (Equation 5.3), while $i \in (1 \ldots N)$, $k \in \mathcal{A}$, and $j \in (1 \ldots l)$. Figure 5.6 additionally shows a graphical representation of matrix M.

Figure 5.6: A graphical representation of M, created with the WebLogo [Crooks and Hon, 2004] software. Bases are distinguishable by colour, the size of each base corresponds to its relative frequency at the given position in the sequence set S.

$$M_{k,j} = \frac{1}{N}\sum_{i=1}^{N} f(S_{i,j}), \text{ where } f(S_{i,j}) = \begin{cases} 1 & \text{if } j = k \\ 0 & \text{otherwise} \end{cases} \tag{5.1}$$

$$S = \begin{bmatrix} \texttt{GAGGTAAAC} \\ \texttt{TCCGTAAGT} \\ \texttt{CAGGTTGGA} \\ \texttt{ACAGTCAGT} \\ \texttt{TAGGTCATT} \\ \texttt{TAGGTACTG} \\ \texttt{ATGGTAACT} \\ \texttt{CAGGTATAC} \\ \texttt{TGTGTGAGT} \\ \texttt{AAGGTAAGT} \end{bmatrix} \tag{5.2}$$

$$M = \begin{array}{c} \\ \mathbf{A} \\ \mathbf{C} \\ \mathbf{G} \\ \mathbf{T} \end{array} \begin{bmatrix} \mathbf{1} & \mathbf{2} & \mathbf{3} & \mathbf{4} & \mathbf{5} & \mathbf{6} & \mathbf{7} & \mathbf{8} & \mathbf{9} \\ 0.3 & 0.6 & 0.1 & 0.0 & 0.0 & 0.6 & 0.7 & 0.2 & 0.1 \\ 0.2 & 0.2 & 0.1 & 0.0 & 0.0 & 0.2 & 0.1 & 0.1 & 0.2 \\ 0.1 & 0.1 & 0.7 & 1.0 & 0.0 & 0.1 & 0.1 & 0.5 & 0.1 \\ 0.4 & 0.1 & 0.1 & 0.0 & 1.0 & 0.1 & 0.1 & 0.2 & 0.6 \end{bmatrix}. \tag{5.3}$$

Matrix M contains a probabilities for each base at each position, the columns values sum up to 1 (i.e. 100%). Compared to the example shown, in reality much more sequences, typically extracted from experimentally verified promoter regions are used to construct reliable PWMs. A PWM assumes statistical independence, i.e. the probability for each base is calculated independently. Thus, the overall probability of a given sequence, e.g. "TAGGTAAGT" can be calculated given the probabilities of M by simple multiplication of each bases probability:

$$p(TAGGTAAGT|M) = 0.4 \times 0.6 \times 0.7 \times 1.0 \times 1.0 \times 0.6 \times 0.7 \times 0.5 \times 0.6 \tag{5.4}$$

$$= 0.021168 \tag{5.5}$$

The position weight matrices for regulatory elements used within this work were collected from different sources. The TATA box PWM used within the pipeline was published in the Eukaryotic Promoter Database (EPD) [Dreos et al., 2013], while the INR model is based on work from Chalkley and Verrijzer [1999]. DPE, BRE_u and MTE definitions were used as published previously by Jin et al. [2006]. For DCE and BRE_d reference data from the original publication was used to create suitable PWMs [Lewis et al., 2000; Deng and Roberts, 2005].

The algorithm to locate appearances of potential elements was originally published by Cartharius et al. [2005] as MatInd and MatInspector. It was ported to a stand alone version, as the original algorithm is only available via web-interface and not suitable for high throughput analyses. MatInd constructs an additional utility vector from a given PWM, while MatInspector performs the actual probability calculation for each sequence (Figure 5.7). For each k-mer of the 200 nt broad sequence window, nine different scores for each regulatory element are computed. In case of a probability ≥ 0.75 the sequence is defined as hit.

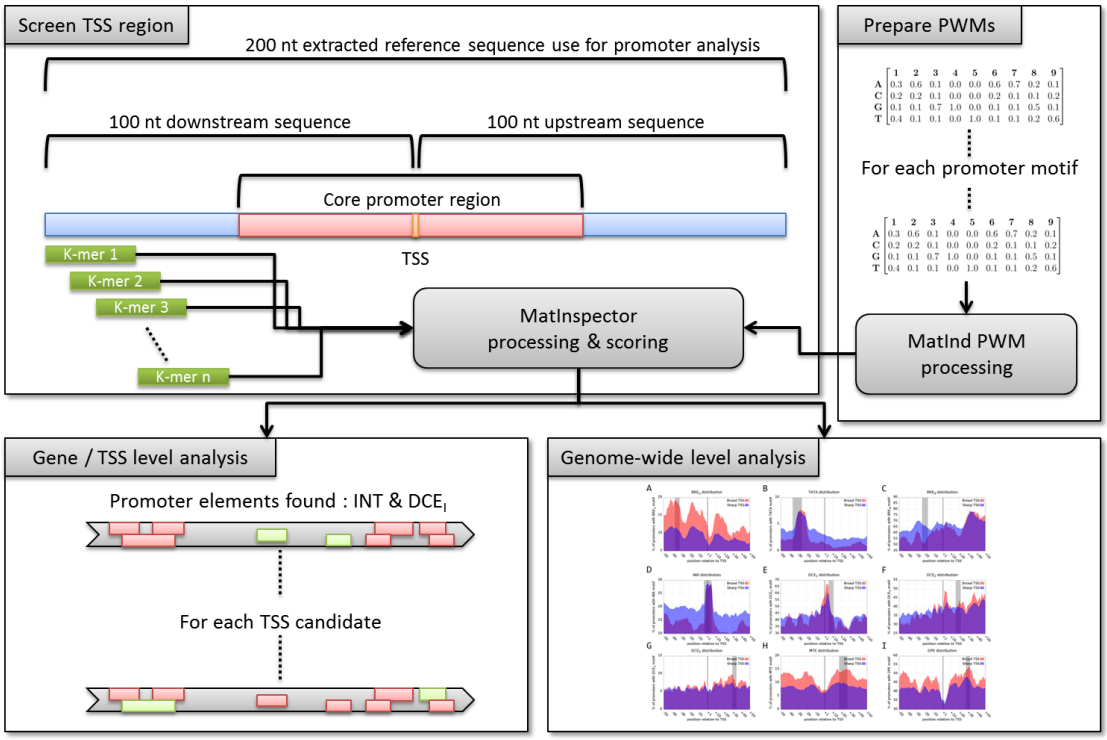


Figure 5.7: Overview of the promoter analysis process. A sequence reaching from -100 to +100 relative to the TSS is extracted from the reference genome. A sliding window screens the whole sequence for potential promoter motifs using PWMs and a local MatInspector implementation. Results are stored on a genome-wide and per gene basis.

If a probability ≥ 0.75 is measured within the defined borders of a specific promoter motif, the core promoter of the corresponding TSS is assumed to possess this particular promoter element. To compensate for different transcript start site peak shapes, up to 5 nt deviation in both 5' and 3' direction are tolerated. Values ≤ 0.75 are not considered for per-gene analysis if they appear outside of their expected location. However, these findings are included in the genome-wide analysis. The genome-wide approach sheds light on the distribution of motifs throughout the core promoter region and therefore answers questions concerning the general position and spacing of motifs. In contrast, the gene-centric view provides information about the set of motifs for specific genes.

Additionally to the nine promoter motifs, other sequence features are analysed during processing. CpG islands are stretches of guanine and cytosine nucleotides connected by a phosphodiester bond (hence CpG) reaching over ≥ 200 nt [Gardiner-Garden and Frommer, 1987]. On the one hand, these islands can be used as criteria for promoter discovery, as 40% of mammalian promoter and 70% of human promoters exhibit a significantly higher G+C content within their promoter regions [Fatemi et al., 2005]. On the other hand, a study by Saxonov et al. [2006] showed,

that CpG islands may also be used to distinguish between two promoter classes, since one class exhibits a much higher CpG density than the second one. A CpG analysis was integrated into the bioinformatics pipeline, based on the `cpgplot` tool and part of the Staden bioinformatics software suite [Staden, 1996; Larsen et al., 1992].

It is known that the -1/+1 position has a preference for pyrimidine/purine (PyPu) dinucleotides [Carninci et al., 2006], which, at least in parts is triggered by the INR motif located around this position and exhibits a strong PyPu peak in its centre. These dinucleotide peaks are also captured by an additional filter, together with general statistics concerning nucleotide usage within core the promoter region and beyond, as these pattern might reveal effects of base pair composition on promoter type and shape.

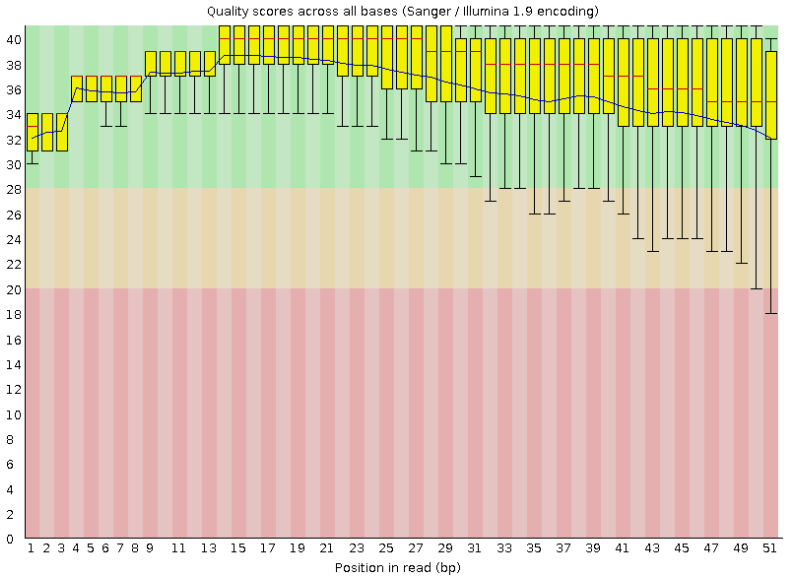
5.4 Results

5.4.1 Preprocessing

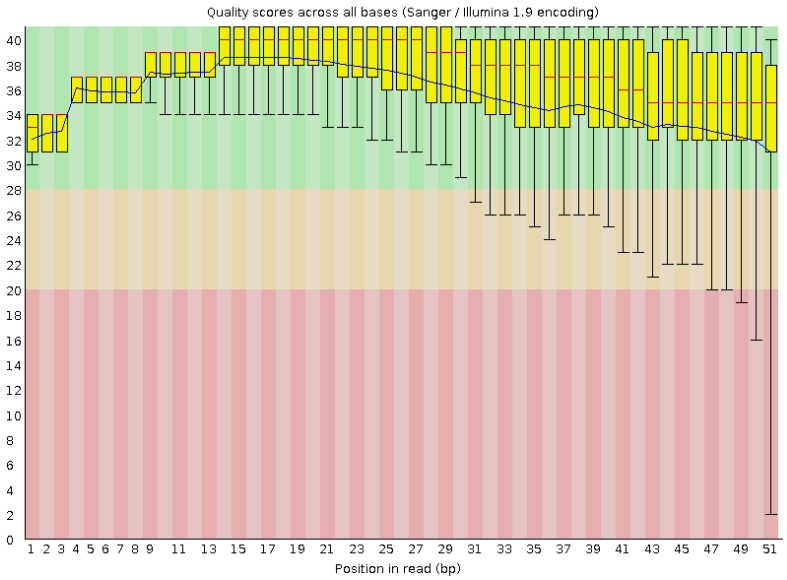
The sequencing process of both libraries on a HiSeq 2000 machine was successful and achieved a total yield of 4.67 Gb for the run, divided into 3.43 Gb for the peak library and 1.26 Gb for the background library. The imbalance between both sequencing results is owed to the 70:30 ratio of peak and background library introduced before sequencing. 111,980,314 and 44,686,574 reads were generated for peak and background library respectively (Table 5.2).

Prior to any further analysis quality checks were performed in order to eliminate potential difficulties in following downstream analyses. Quality control of sequencing reads was realised through the FastQC software which allows for a standard repertoire of checks, including quality values, checks for linker and vector fragments, or biased base composition of the reads [Andrews, 2012]. A graphical representation showing the per base quality of both libraries is shown in Figure 5.8. The background library (Figure 5.8a) displays a flawless quality up to base 32, when first outliers reach values below 28. Even the last base, shows reasonable quality values, however, due to outliers below phred scores of 20 the last three bases (48 to 51) were clipped from all background reads before further analysis. The image for reads originating from the peak library (Figure 5.8b) is very similar and only deviating in the fact, that the last five bases (46 to 51) were clipped due to outliers below phred score 20. Clipping removed a total of 560 Mb of sequence information from the peak and 134 Mb from the background library, which is negligible.

Directly after clipping, quality filtering was carried out to remove any residual linker sequences, which may cause interference during read mapping. Therefore, all reads were screened for fragments of suitable adapter sequences used during



(a) Sequencing quality of the background library. Generally, reads achieve reasonable quality scores of not less than 31, although base 51 includes outliers reaching down to phred 20.



(b) Sequencing quality of the peak library. Reads achieve qualities comparable to the background library. Bases starting at 49 however show decreasing quality values for outliers.

Figure 5.8: Quality overview of sequencing runs for background (5.8a) and peak library (5.8b) produced produced by the FastQC quality control tool. The x-axis shows the base position within the read, the y-axis indicates the phred quality score.

	Peak library	Background library
Yield	3.43 Gb	1.24 Gb
Read length	51 nt	51 nt
Read length (clipped)	46 nt	48 nt
No. of Reads	111,980,314	44,686,574
Discarded reads (quality filtered)	565,212	418,503
Remaining reads	111,415,102	44,121,362

Table 5.2: Results of the RNA sequencing run conducted on a Illumina HiSeq 2000 mashine. The remaining reads in the last row are the data source for all further analyses.

Illumina sequencing. Indeed, a fraction of the reads showed residual linker patterns, whereupon the whole read was removed instead of only removing the linker sequences from the read. A removal of adapter sequences yields very short remaining reads due to the initial read length of only 50 nt and was therefore considered infeasible. Adapters were found in 565,212 and 418,503 reads belonging to peak and background library (Table 5.2). These numbers were within the expected range (below 1%) and were removed.

5.4.1.1 Evaluation of different read mapping tools

The basis for all following analyses, TSS identification as well as promoter analyses is a correct mapping of as many reads as possible onto a chosen reference genome. Only if reads are mapped back to their correct original position a precise location of transcription start sites is possible

miRNA filtering As this work's focus are protein coding genes synthesised by RNA polymerase II, other non-coding mRNA types, such as miRNA should be filtered out prior to further steps. miRBase[23] [Kozomara and Griffiths-Jones, 2014], a central, curated registry of miRNA stemloops and mature miRNA sequences was selected a sources for miRNA reference data. The 20th database release, published in June 2013, contains 24,521 precursor miRNAs and 30,424 mature miRNA products. In order to maximise filter efficiency, both stemloops and mature sequences were converted for use as read mapping reference. This mapping resulted in 2,074,740 (1.86%) peak reads and 1,329,483 (3.00%) background reads that produced significant mappings for the miRBase reference. Those reads were excluded from further processing.

[23] http://www.mirbase.org

Read mapping against CHO-K1 and Chinese hamster references Initially, read mapping was performed with the Bowtie (version 0.12.7) [Langmead et al., 2009] read mapper. The achieved mapping rates however, were not satisfactory (Figure 5.9). Since the RNA samples used within this work originate from a CHO-K1 cell line, a logical choice for a reference genome was the published CHO-K1 draft genome sequence [Xu et al., 2011]. Howsoever, with trimmed and filtered reads Bowtie was only able to map 47.06 % of peak reads and 13.39 % of all background reads to the reference genome. Low mapping rates as those reported by Bowtie may indicate a general problem in sequencing, such as contaminations, sequencing artefacts or large ratio of spliced RNAs in the sample, which cannot be mapped with simple mapping software such as Bowtie.

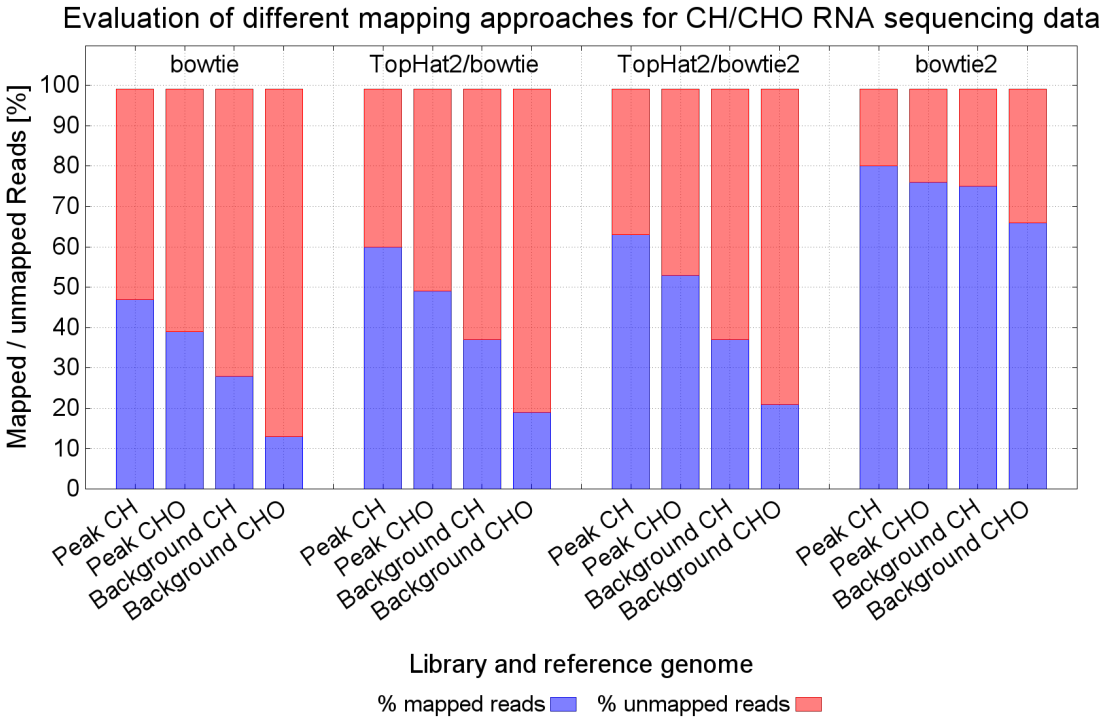


Figure 5.9: Graphical representation of the different mapping approaches evaluated. CH is used as abbreviation for Chinese hamster, CHO represents the CHO-K1 draft genome. The continuous increase of mapped reads and the decrease of unmapped reads is easily visible from approach to approach. Mapping rates for each tool in format peak CH/peak CHO, background CH/background CHO: Bowtie: 47.06 %/39.10 %, 28.33 %/13.09 %; TopHat2/Bowtie: 60.29 %/49.09 %, 37.25 %/19.92 %; TopHat/Bowtie2: 63.82 %/53.08 %, 37.77 %/21.57 %; Bowtie2: 80.81 %/76.20 %, 75.02 %/66.72 %.

In a first step a random sample of reads was extracted from both libraries and screened for possible contaminations with BLAST [Altschul et al., 1990] against the nt database. Most of the reads had their best hit within the CHO-K1 genome. However, a significant amount of reads did not score any hit in CHO-K1 but in

mouse, rat or human instead. No contaminations such as *Escherichia coli* or PhiX could be confirmed. Further analysis of reads scoring hits in rat and mouse showed perfect hits, suggesting that those reads are indeed from a CHO cell but probably located in non-assembled areas of the CHO genome, which still has draft status. Since the CH draft genome is now publicly available [Brinkrolf et al., 2013; Lewis et al., 2013] a second read mapping with identical setup was employed to assess the effect on mapping ratios.

A direct comparison between both mappings is given in first section of four columns of Figure 5.9. Although the number of reads successfully mapped increases by 8 % to 47.06 % for peak reads and another 15 % increase to 28.33 % is visible for background reads both rates are still below 50 % and require further improvement to increase the yield of recovered reads. As pointed out earlier, it is possible that a certain amount of reads could not be mapped due to splice sites within the reads. Although the read length of only 50 bp makes this scenario unlikely, read mapping was conducted with TopHat2 (version 2.0.4) [Kim et al., 2013], a splicing site aware transcriptome mapping extension that is based on the Bowtie mapper family, to assess the influence of splice sites on the mapping. Again, CHO-K1 and CH provide the reference indices, while peak and background reads of both libraries were mapped separately. Two setups differing in the underlying Bowtie version (0.12.7 and 2.0.0-beta6) were prepared to evaluate potential improvements added in Bowtie2. Analysis of TopHat2's output showed promising results already for the Bowtie backend, yielding about 11.5 % gain for peak and 7.5 % for background. However, differences between both setups do not exceed 4 % for the peak library and 2 % respectively for background library. In a last mapping setup, Bowtie2 was employed using default parameters with the exception of the so called "localmode". This mode adds the possibility to clip 5' and/or 3' end of the read in order to obtain suitable alignments. In some cases, this ability might be able to recover additional, otherwise not mapping reads. Prior to sequencing, the first cDNA strand was synthesised using random N6 primers (see Section 5.2.2). Indeed, a recent study by Hansen et al. [2010] showed cases of biases in Illumina transcriptome sequencing which were caused by N6 random priming during library preparation. Due to the possibility of clipping non-aligning regions of the read the percentage of mapped reads may increase significantly. The difference between the best TopHat2/Bowtie2 run (Figure 5.9, third group of columns) and the Bowtie2 run with localmode enabled (Figure 5.9, last group) is easily visible and resulted in mapping rates over 80 % for peak reads and 75 % for background reads (given the CH reference), while both mappings with CHO-K1 reference performed about 4 % and 9 % inferior compared to the CH run. Overall, the mapping results seem to verify the assumptions on potentially biased reads produced by the RNA sequencing setup. Further analyses of mappings generated by Bowtie2 showed a significant prevalence for soft clipping events at the 5' end of the reads (56.38 % vs 18.20 % for background reads, 46.54 % vs 37.75 % for peak reads, based on the CH mapping), which correlates with the 5' bias suggested by Hansen et al. [2010].

5.4.2 Identification of transcription start sites

Read mapping typically produces BAM files containing all information about the mapping process. The initial peak extraction generates a preliminary list of 51,892 peak entries and 20,931 background entries (Figure 5.10, first row). However, these numbers represent unfiltered data, which requires additional processing steps. All peaks, which are supported by < 10 reads are removed from the set, leading to 15,876 and 3,458 remaining peaks from peak and background library, while a large portion of the removed peaks was supported by one read only and therefore deemed as not reliable for further processing (Figure 5.10, second row). Subsequently consecutive peaks were grouped, as many transcription start sites are defined by more than one peak, normally separated by several nucleotides. This grouping reduces the number of peaks by roughly 50 %, down to 7,462 (Figure 5.10, third row). In a last step false positive peaks are removed from the peak set, identified by peaks for background and peak library at the same position. In such a case, the peak is removed only if the background read coverage corresponds to at least 10‘% of the peak library coverage. In total, 915 background peaks could be removed from the peak set, leaving a final set of 6,547 potential TSSs (Figure 5.10, fourth row).

5.4.3 Annotation of transcription start sites

Annotation using the Chinese hamster reference genome The annotation module received a set of 6,547 potential transcription start sites which require further functional annotation. A majority of transcription start sites should be assignable to annotated CH genes, even if the hamster reference genome still is in draft status in 2014. As such a mapping of all TSSs to genes of the CH reference genome is performed in a first annotation step. 4,320 transcription start sites were directly assigned to 3,808 different hamster genes as depicted in the lower part of Figure 5.10, while the difference between TSS count and gene count is caused by multiple alternative TSSs per gene. For 2,227 remaining peaks no genes were found within the threshold of 3,000 nt (Section 5.3.4). However, due to the draft status of the reference genome, a large portion of these TSS candidates is expected to find corresponding genes in closely related genomes. Therefore, the remaining 2,227 TSSs were screened for potential target genes by means of sequence similarities.

Annotation by RefSeq sequence similarity search The transcript variant of the RefSeq database was used in combination with BLAST as described in Section 5.3.4 in which three different search results are distinguished (shown in the yellow area of Figure 5.10). In case no hit is found for the extracted 3,000 nt sequence downstream of the TSS, the TSS remains without annotation. In most cases, multiple hits for single gene symbols are found in the database, corresponding to several different source organisms. This status of sequence conservation is a strong indicator for a reliable assignment and 1,581 of the 2,227 TSSs could be classified into this category, representing 542 unique genes and thus increasing

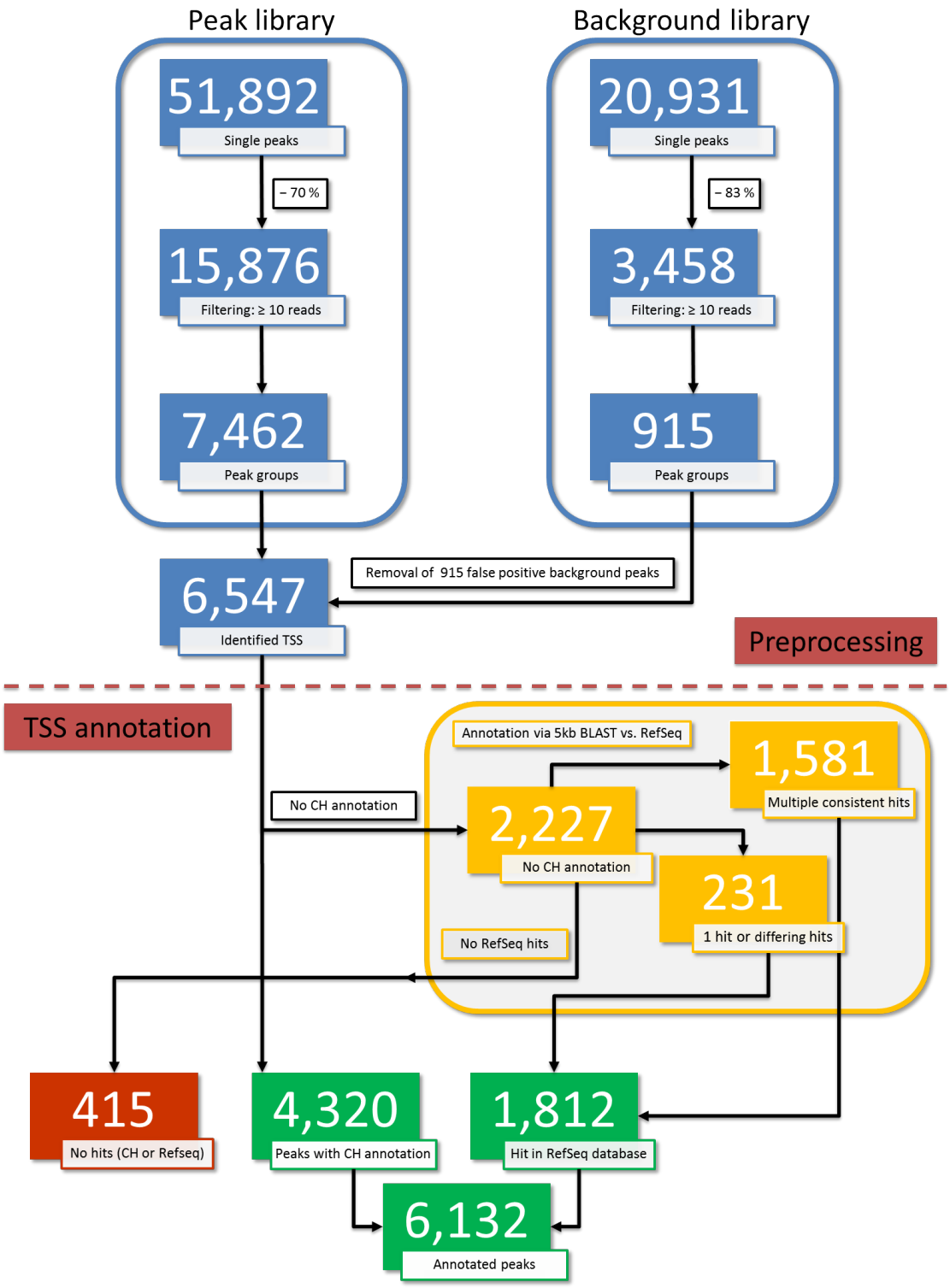


Figure 5.10: Graphical representation of preprocessing and TSS identification results. 51,892 and 20,931 peaks were initially identified for peak and background library. Filtering for reads ≥ 10 removed 70 % and 83 % of noise peaks. 7,462 peaks remained for the peak library, 915 peaks were recognised as background peak. 6,547 transcription start sites were located, of these 4,320 could directly associated with Chinese hamster genes. 1,812 of 2,227 TSSs were assigned to non Chinese hamster genes, 1,581 were classified as trustworthy hits, and 231 received a putative rating. In total, 6,035 TSSs (93.66 %) could be assigned to genes, 415 TSSs (6.33 %) could not be assigned to any gene.

the count of successfully annotated TSS to 5,901. A third category comprises of less well conserved hits within the RefSeq database, caused by non-unique gene names for similar transcripts or single hits in only one organism. 231 of these less confident hits were found and assigned to 174 genes during the analysis, resulting a total count of 6,132 annotated TSSs (93.66 %) assigned to 4,437 unique gene names. Only 415 TSS (6.33 %, shown red in the lower part of Figure 5.10) could be assigned by neither reference-based nor sequence similarity methods.

To further characterize the set of 415 unassigned TSS, the average and median peak heights were computed for assigned and unassigned peaks, resulting in a nearly identical average peak height of 3,644 reads for the assigned peaks and 3,613 reads for the unassigned peaks. However, median values deviated significantly, showing 1,577 reads for assigned peaks and only 339 reads for unassigned peaks. Therefore, most of the unassigned peaks seem to possess a significantly lower coverage with only a few outliers showing considerably higher read counts. This data suggests a relatively large fraction of weaker expressed background peaks in the 415 unconfirmed TSS.

5.4.4 Gene Ontology classification of transcription start sites

The successful assignment of genes to experimentally verified transcription start sites also added access to the complete functional annotation of these genes. In order to validate the obtained results in terms of functional annotation and to estimate the range of functions associated to the genes of the detected promoters a functional analysis was performed. The required information is provided in form of specialized tags or terms, assigned by the Gene Ontology (GO) consortium [Ashburner et al., 2000]. By defining a common set of terms, classified into three super categories, function, localisation, and involved biological processes can described precisely. Following both gene assignment steps, all unique gene symbols (e.g. *Bop1*, *Gtf2h2*, ...) were extracted from the final mapping, resulting in a list of 2,241 gene names. This set defined the input data for processing with the DAVID Functional Annotation Tool [Huang et al., 2009a,b], provided as an online analysis tool. The tool is based on a massive set of functional databases and mappings between those databases, helping to establish links between different annotations. As the tool does not yet incorporate *Cricetulus griseus* as source genome, *Mus musculus* was selected as substitution and 2,126 entries (94.87 %) of the set of 2,241 unique genes could be identified in the mouse genome. GO terms were extracted and classified into the three main categories. 1,602 (75.6 %) of the 2,126 genes mapped to mouse possess one or more GO terms for biological processes, 1,781 (84.0 %) have a cellular component assigned, and 1,675 (78.9 %) are involved in molecular functions according to GO.

For each category, all top level terms were selected for graphical representation (Figure 5.11). For the molecular function category (Figure 5.11, red bars), the

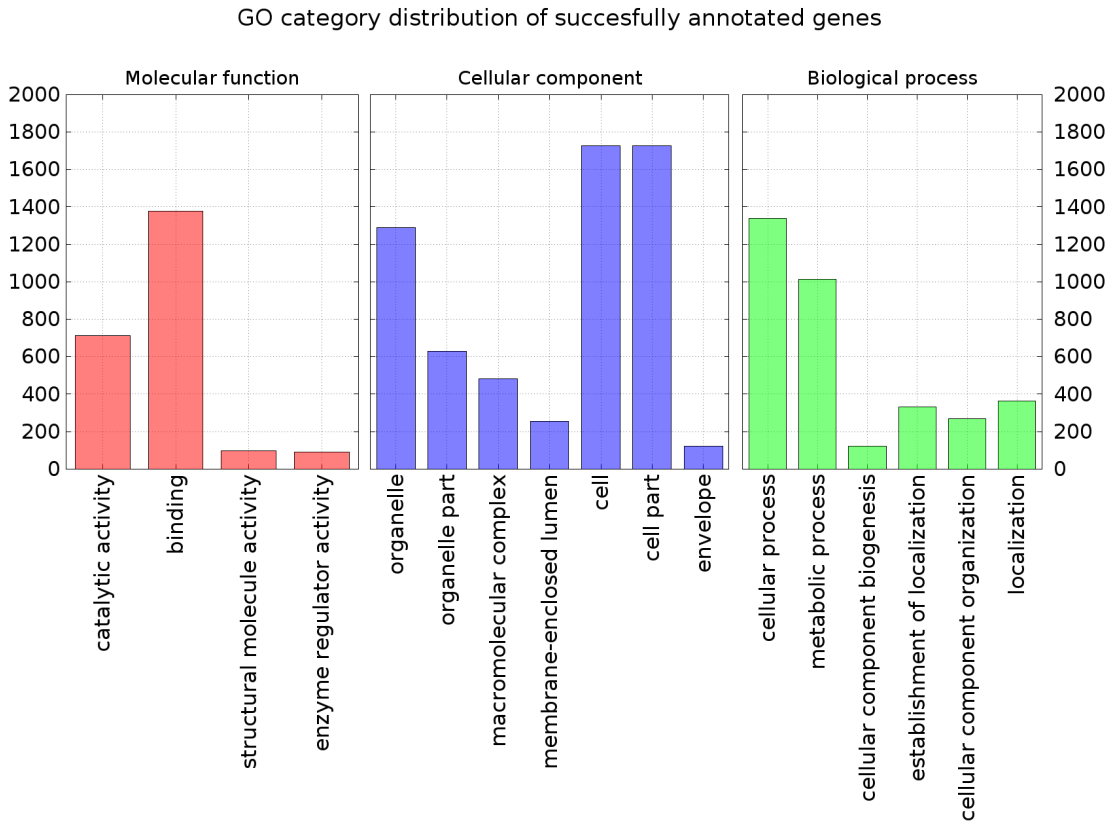


Figure 5.11: Gene Ontology based mapping of genes assigned to TSSs. A list of 4,437 unique genes names was screened for associated Gene Ontology (GO) [Ashburner et al., 2000] terms in the three categories "Molecular function" (red), "Cellular component" (blue), and "Biological process" (green). The x-axis shows the name of the assigned GO term, on the y-axis the number of genes assigned to this term are shown. For each class, only the 20 most abundant go terms were selected. The gene name ⇒ GO term mapping was performed through the DAVID web suite [Huang et al., 2009a,b].

four GO terms are dominated by binding functions followed by catalytic activity in the first column, while regulatory activity and molecule activity only play minor roles. Location information for the gene set is delivered by the cellular component category (Figure 5.11, blue bars). Here, the two most abundant terms represent location in the cell or parts of the cell, followed by cell organelles. Assignment of genes to biological processes (Figure 5.11, green bars) shows clear overrepresentation of cellular and metabolic processes, followed after a gap to other processes such as localisation, cellular component organisation of cellular component biogenesis.

Functional classifications of the presented approach into GO terms were compared to the results obtained by Rupp et al. [2014] for a CHO-K1 cell line genome assembly. Although the deviation in the number of gene clusters detected by Rupp

et al. [2014] and the number of assigned genes of this work partially reaches one order of magnitude, the general distribution of functional assignments stays comparable. As such it is reasonable to assume that the set of genes assigned to observed promoters in this study does show signs of strong biases for specific functions, locations or processes.

5.4.5 KEGG based mapping of genes of transcription start sites

In industrial context the systematic overexpression of specific genes is one of most common use cases within cell line optimisation to obtain a desired protein. In order to maximize the yield of such processes, in-depth knowledge of involved pathways is crucial. To demonstrate the industrial relevance of the approach presented in this work, a mapping of peaks to genes of a selected KEGG pathway was performed [Kanehisa and Goto, 2000]. KEGG, the Kyoto Encyclopedia of Genes and Genomes provides a database of enzymes, uniquely identifiable through EC (Enzyme Commission) numbers. Such a number comprises of four letters divided by dots and six different major categories (1.-.-.- to 6.-.-.-). The naming scheme is hierarchical, meaning that 1.1.-.- is the parent group of 1.1.2.- and 1.1.3.- and as such the latter two are more specific enzymes of the parent group. Annotation data in form of EC numbers was extracted from input GenBank files and combined with corresponding expression heights. A logarithmic colour scale adjusted to fit the minimal and maximal values of the data set is automatically generated and allows precise assignment of expression heights. On the basis of the generated scale, a CSV file is generated, suited for upload within the KEGG "colour pathway" web tool[24]. Overall 463 different EC numbers were extracted from the conducted functional annotation of the transcription start sites. In Figure 5.12, the mouse reference pathway for glycolysis is shown. It is possible to reveal most of the pathway, only a few genes coding for alternative enzymes could not be found in the data set. For glycolysis 18 different EC numbers were mapped onto the pathway and their corresponding genes are scattered throughout the genome on all chromosomes. Although most of the enzymes are marked with average expression strength, the high dynamic range of the RNA sequencing approach is able to reveal very weakly expressed genes like *Galm* (EC number 5.1.3.3, between 30 and 50 reads) as well as genes with a significantly higher expression strength like *Akr1a1* (EC number 1.1.1.2, >140,000 reads). Again, the difference between lowest and highest expression level in the glycolysis pathway alone covers five orders of magnitude, which would not be feasible with traditional microarray experiments due to saturation effects [Zhao et al., 2014].

5.4.6 Insights into the Chinese hamster promoter landscape

6,132 of the 6,547 peaks were successfully assigned to annotations of the Chinese hamster or they were correlated to homologous sequences of related species, indi-

[24] http://www.genome.jp/kegg/tool/map_pathway2.html

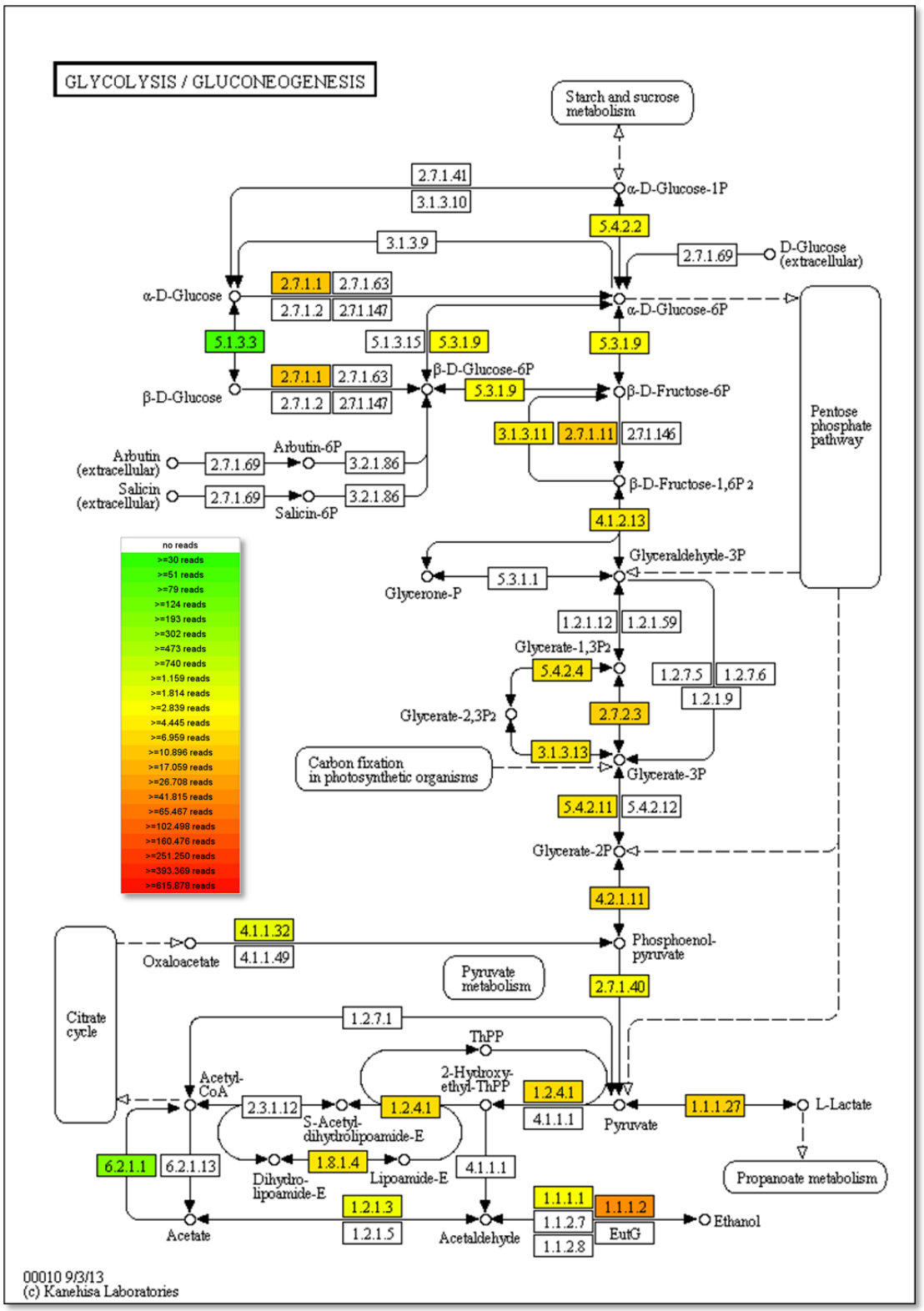


Figure 5.12: Mapping of TSS peaks to KEGG pathways. Export of the glycolysis KEGG pathway with colour coded transcription levels. TSS peaks were assigned to genes coding for specific enzymes involved in glycolysis. Most of the entries show a medium transcription level of 2,000 - 10,000 reads, however, selected genes show very high expression levels (1.1.1.2, >100.000 reads) or very low expression levels (5.1.3.3., >30 reads).

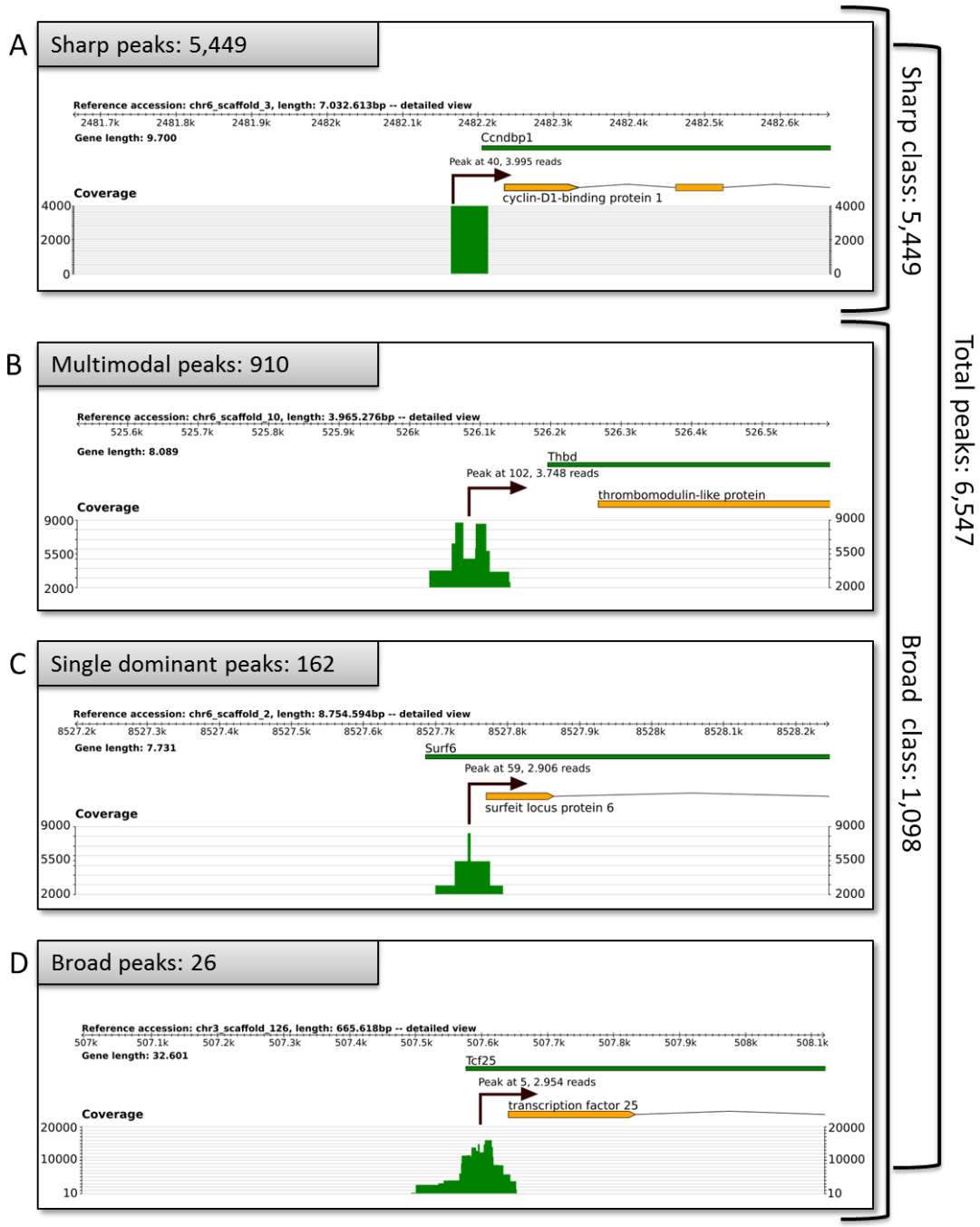


Figure 5.13: Overview of different peak shapes. (A) Sharp peaks. These peaks are characterized by one narrow peak, typically not much wider than the actual read length. (B) Multimodal peaks. These broad promoters are characterized by a plateau of basic transcription over more than one read length, accompanied by at least two distinct peaks outreaching significantly the basal transcription level. (C) Single dominant peaks. These broad promoters show a basal transcription similar to multimodal peaks but possess only one distinct peak. (D) Broad peaks. They are generally longer than one read length without any dominant distinct peaks. Green bars, annotated genes; yellow bars, exons; green peaks, mapped reads.

cating which genes were transcribed in the CHO-K1 culture under the experimental conditions used. Due to the setup of this approach, a list of exact TSSs is available for 4,437 genes, paving the way for in-depth analyses of promoter regions. These analyses are implemented and grouped in the promoter analysis module of the bioinformatics pipeline (Figure 5.3). Studies of the visualized data (Figure 5.13 and Figure 5.14) showed highly diverse patterns of the promoter anatomy, which correspond to similar findings in mammalians [Sandelin et al., 2007; Carninci et al., 2006] as well as in Drosophila melanogaster [Rach et al., 2009] and plants [Kumari and Ware, 2013]. This heterogeneity is reflected by the two main TSS classes "broad" (including three sub-classes) and "sharp". Broad TSSs exist, because in many cases a TSS cannot be pinpointed to a specific base on the genome, owing to the fact that transcription is initiated over a range of up to 150 nt [Carninci et al., 2006]. These TSSs show a plateau-like distribution of mapped reads and are generally associated with ubiquitous spatial and temporal expression patterns [Rach et al., 2009]. In contrast, sharp start sites exhibit a sharp, well-defined peak, combined with tightly regulated spatial and temporal expression of their corresponding genes [Rach et al., 2009]. Out of the 6,547 peaks previously identified, 5,449 peaks (83.23%) were categorized as sharp (Figure 5.13 A), while 1,098 peaks (16.77%) were assigned to the class of broad TSSs (Figure 5.13 B - D). Peaks assigned to the broad set were further split into one of the three sub categories (i) multimodal (MU), (ii) single dominant (PB), and (iii) broad (BR), defined by visual features of the peaks [Carninci et al., 2006]. The majority of broad peaks belongs to the sub-class multimodal (910, 13.90%), meaning that at least two clear peaks are visible above a basic transcription level (Figure 5.13 B). Another 162 peaks (2.47%) were classified as dominant (Figure 5.13 C) and thus also possess a basic plateau-like transcription activity, complemented by a single sharp peak. The sub class of classical TSSs featuring a plateau-like appearance without any other distinctive features (Figure 5.13 D) was assigned to only 26 peaks (0.40%). In order to uncover the potential impact of the peak type on the existence of regulatory elements, all subsequent analyses were calculated for broad and sharp TSS classes separately.

5.4.7 Promoter landscapes of the Chinese hamster

Another feature of module C of the bioinformatics pipeline is the analysis of promoters on the global scale. The core promoter regions between -40 and +35 were screened for nine common regulatory elements of eukaryotic promoters (Figure 2.4). Since these elements can have severe impact on transcription activity, an in-depth analysis of each regulatory element of the core promoter was conducted.

For the "TFIIB recognition element upstream" (BRE_u) [Lagrange et al., 1998] three characteristic and decreasing summits can be identified (Figure 5.14 A), which is consistent with previous observations in mouse and human [Carninci et al., 2006]. Although the assumed BRE_u location is covered with little offset, it is obvious that broad start sites seem to prefer BRE_u usage before sharp TSSs

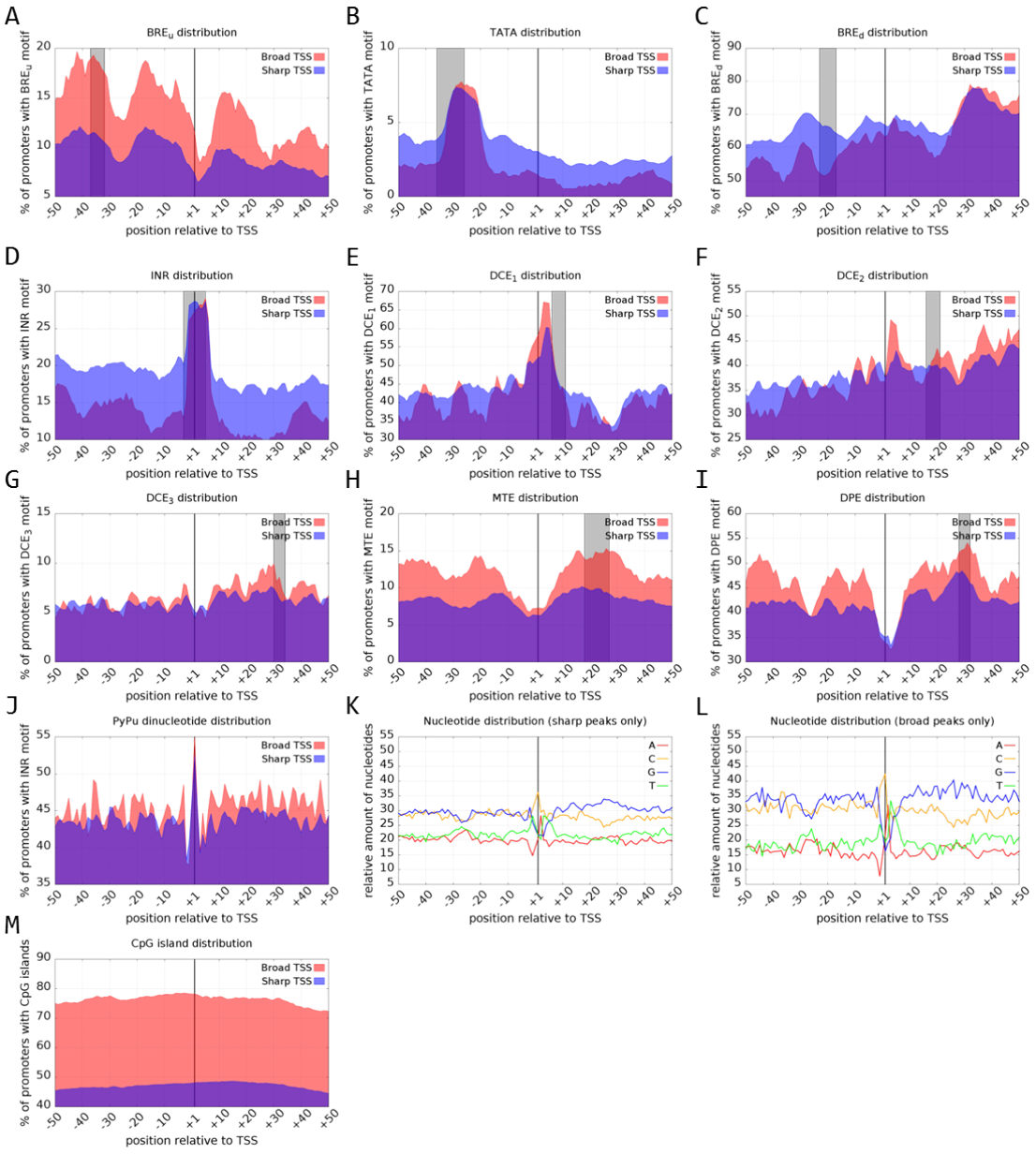


Figure 5.14: Matrix showing screened regulatory elements. Position relative to the TSS (+1, black line) is shown on the X-axis, the Y-axis is labelled with the global percentage of promoters having the corresponding feature at this position. Broad (red) and sharp (blue) TSSs are shown separately, grey boxes indicate the expected position of the maximal peak.

with a difference of 4 % to 6 %. The striking drop observed around position +1 may be due to the fact that region -2 to +4 is dominated by the "initiator" (INR) element [Corden et al., 1980]. The INR motif exhibits high pyrimidine content combined with an 'A' fixed at position +1 in contrast to the BRE_u consensus, which is completely free of pyrimidine and therefore should not yield any signals at the INR location.

The TATA box location fits perfectly to its assumed position and is slightly preferred by broad over sharp TSSs (Figure 5.14 B) [Goldberg, 1979]. Thereby, the average difference is only about 3 % to 4 %. The fact that only 10 % of all (broad) TSSs possess a TATA box is consistent with comparable studies initially relativizing the number of TATA box dependent promoters to 25 % [Suzuki et al., 2001] and later on down to only 10 % [Carninci et al., 2006].

In contrast to the data for BRE_u, the "TFIIB recognition element downstream" (BRE_d) does not fit well to the predicted position (Figure 5.14 C) [Deng and Roberts, 2005]. A first accumulation is located at -30, more than 10 bp upstream of its presumed location (-23 to -17). More than 70 % of all sharp promoters and 60 % of the broad promoters show this first peak. A second peak for 80 % of all promoters was identified at position +30. However, similar studies in human cells [Albert et al., 2010] showed likewise inconclusive results with no detectable abundance at the predicted BRE_d location [Deng and Roberts, 2005]. Given the generally high level of noise within the BRE_d data set, it seems plausible that the shifted peak at -30 may in fact be an artefact. Additionally, the strong broad/sharp cluster starts at +20, which might be attributable to the fact that the BRE_d consensus sequence exhibits a high G+C ratio, which also is common for the gene start [Saxonov et al., 2006].

The INR motif is very tightly located around the +1 position, with only minimal abbreviations in upstream or downstream direction (Figure 5.14 D), which correlates well to the fact that the INR motif operates efficiently only when correctly positioned [Smale and Baltimore, 1989]. Although the general level of background activity is in the order of 12 % to 15 %, no significant secondary clusters were observed. INR usage does not seem to be related to the TSS type, as the difference between sharp and broad start sites is negligible. Roughly 25 % of all TSSs analysed have a correctly positioned INR motif, which is below the findings for human and mouse, where up to 50 % of all promoters involve INR motifs [Maston et al., 2006].

Additionally to previous regulatory motifs, we included the three "downstream core elements" (DCE) DCE S_I, DCE S_{II}, and DCE S_{III} to receive a complete picture of the Chinese hamster promoter landscape [Lewis et al., 2000]. The three subunits of DCE are scattered over large parts of the core promoter positioned from +6 to +34 (Figure 2.4). In contrast to other regulatory elements, the individual subunits

of DCE are very short sequence tags of 3 to 4 bases with only minor variations [Lewis et al., 2000]. Common for DCE S_I and DCE S_{II} is the almost exclusive use of pyrimidines, resulting in massive clusters around +1, thus mimicking the INR peak and creating a complement to the BRE_u, MTE, and DPE gaps (Figure 5.14 E - F). However, apart from a small secondary culmination, DCE S_I does not seem to be specifically overrepresented in our data set. Within the expected position of DCE S_{II}, another secondary cluster can be identified, but hardly exceeds the general noise level of around 22 %. A similar situation is visible in the graph for DCE S_{III}, with the exception that the background level varies around 6 % to 8 % (Figure 5.14 G). Again, as for DCE S_I and DCE S_{II}, no local maxima distinguishable from the background frequency are located at the correct position.

The "motif ten element" (MTE) is co-localized with DCE S_{II}, while MTE and INR are able to operate on a synergistic basis given a correct spacing between both elements [Lim et al., 2004]. The signal distribution is bimodal, resulting in two peaks of equal height, where the first peak from -30 to -10 is less wide than the main cluster including the MTE location (Figure 5.14 H). The groove around +1 for the MTE is larger but similar to the one of BRE_u. The most probable cause for this large MTE signal free area is the complete lack of pyrimidines strictly required for the INR motif. The expected location of the MTE (+18 to +27) is completely covered by the second cluster, yielding over 15 % of MTE promoters for the broad group and 11 % for the sharp group.

Distribution of clusters for the "downstream processing element" (DPE) data can be described as relatively heterogeneous with one dominant accumulation at the proposed DPE location (+28 to +32) (Figure 5.14 I) [Burke and Kadonaga, 1996]. As for BRE_u and MTE, a significant drop of positive promoter signals is visible around the +1 region, similarly explainable by the pyrimidine-rich INR motif residing around +1. Although the DPE consensus comprises up to four pyrimidine bases, the probability that all of them are part of the motif is considerably lower, resulting in only few overlaps with the INR motif. Based on DPE data, about 55 % of all broad promoters and 40 % to 45 % of the sharp TSSs possess a DPE.

A feature related to the INR is a pyrimidine/purine (PyPu) dinucleotide peak starting directly in front of the +1 position. Due to the consensus sequence of the INR motif, a striking increase of PyPu dinucleotides should be observable in at least 25 % of the promoters, as each positively identified INR motif is expected to show such a peak. And indeed, following a PyPu depletion at -3, -2 and -1 show a significant increase of PyPu dinucleotides, resulting in one very sharp peak which reaches his maximal value at 55 % for both, broad and sharp TSSs (Figure 5.14 J), which also corresponds to previous findings by Sandelin et al. [2007]. The dinucleotide peak may also be seen as an indicator for correct placement of the TSSs screened for this element, as the primary peak shows no signs of shadow

peaks shifted by several nucleotides.

In order to complement the previous motif-based analyses, several structure-based views were integrated. For all TSS regions the nucleotide composition at each position (from -50 to +50) was recorded and separated by TSS type (Figure 5.14 K and L). Scaling for both views was chosen equally to easily identify deviations between both types. G+C and A+T curves show linkage to a certain degree, which is not related to complementary base pairs, but to sequence similarities within the core promoter region. Both diagrams show a decrease of G+C and an increase of A+T in the region from -30 to -20, which is accurately the position of the A+T-rich TATA box. A second general pattern can be found around the +1 region and corresponds to the already mentioned PyPu peak. Directly upstream of +1 a significant drop of adenosine, followed by a strong increase at +1 is observable, consistent with the INR consensus which includes an A at +1. Although both TSS types share a similar curve pattern, broad TSSs exhibit higher G+C ratios and vice-versa lower A+T ratios, resulting in higher amplitudes within base composition changes. Common to both types is also an increased fraction of guanine in the region from +10 to +40 that fits to the guanine-rich consensus sequences of MTE, DPE and DCE, all of which located within this core promoter region.

In a last step, the core promoter region was screened for CpG islands, areas with increased G+C content. As for other analyses, both TSS types were screened individually. In conformity to a study by Saxonov et al. [2006], the average difference between both types is $\geq$ 25 %, while nearly 80 % of broad TSSs show CpG islands, compared to 45-50 % of sharp TSSs with CpG islands (Figure 5.14 M). Therefore, the consideration of additional structural features, such as CpG content may further improve the classification process for transcription start sites.

5.4.8 Analysing Chinese hamster promoters on the gene level

The above identified promoter elements, as well as combinations of them, have the potential to significantly increase or decrease transcription activity. As such, a global view on promoter architecture can give hints about general usage patterns. However, with regard to a biotechnological usage of promoters for the pharmaceutical industry, specific knowledge of the regulatory promoter elements on the gene level is of special interest. In this context, analyses from module C of our bioinformatics pipeline (Figure 5.3) can also be used to demonstrate the characteristics of every single promoter detected. This includes the expression levels of the corresponding genes, which is possible due to the strict limitation on unique mappings of reads. Due to the non-normalized nature of the chosen library approach and the stringent mapping criteria (uniquely mapping reads only), the data presented in this work have a very high information density, as shown by the >90 % mapping rate from peaks to genes. Information about expression strength

for each peak further increases this density.

Table 5.3 summarizes the 20 genes with the highest peaks detected in this approach, along with the type of TSS, regulatory elements detected, and chromosomal location. 55 % (11) of these peaks could be assigned to ribosomal-like genes, another 25 % (5) were identified as histone-like proteins and only 20 % (4) peaks were annotated otherwise. This is consistent with expectations, as histone and ribosomal genes are typically transcribed at very high rates. Most of the top expressed genes are located on chromosomes 1, 2, and 3. This is not unexpected, as chromosome sizes and therefore the number of genes located within the chromosomes drop from 563 Mb (6,161 genes) for chromosome 1 to 278 Mb (3,969 genes) for chromosome 3, and 54 Mb (1,102 genes) for chromosomes 9+10 combined [Brinkrolf et al., 2013]. Focussing on the two main promoter classes it is noticeable that 75 % (15 of 20) of the top 20 expressed genes belong to the sharp type promoters, whereof seven possess TATA boxes. From the eight remaining TATA-less sharp promoters, four are equipped with DPEs and only two sharp promoters do not appear to possess any regulatory elements. On the global scale, only 10 % of the analyzed promoters in this study do not seem to include any of the regulatory elements common to eukaryotic promoters. DPE appears to be one of the most commonly used motifs, especially in combination with BRE_d, BRE_u, MTE, and INR. As predicted by previous studies, TATA and TATA-combined promoters are rather exceptions and account for only a fraction of all promoters [Suzuki et al., 2001; Carninci et al., 2006].

5.5 A successful combination

The innovative dual-library sequencing approach enables the effective removal of false positive in an early stage of the pipeline, thus reducing unnecessary computations and sophisticated removal of false positive by *in silico* measures. The bioinformatics pipeline was specifically tailored to meet the requirements of the dual-library sequencing strategy and thus profits from the library design. Although the pipeline requires a reference genome, results showed that even draft status genomes are capable of delivering a wide range of information. In case of annotated genomes with gene predictions, a precise assignment of transcription start sites to genes is possible. However, the pipeline also reported by now unknown transcription start sites, either within genes or in previously uncharted genome regions. The presented pipeline's output provides promoter information on different scopes, reaching from nucleotide-level to genome-wide observations for known regulatory elements.

Reads	Assigned annotation	Chr.	Class	Promoter element								
				BRE_u	TATA	BRE_d	INR	DCE_I	DCE_{II}	DCE_{III}	MTE	DPE
921,977	L28-like protein	9/10*	SP	-	✓	-	-	✓	✓	-	-	-
911,238	H4-like protein	1	SP	-	-	✓	-	-	✓	-	-	-
849,029	H4-like protein	3	SP	-	-	✓	✓	-	-	-	-	-
719,851	S24-like protein	1	SP	✓	✓	-	-	✓	✓	✓	-	-
635,045	H1,3-like protein	3	MU	-	-	✓	-	-	✓	-	-	✓
627,213	histone H2B type	3	SP	-	-	-	✓	✓	-	-	-	✓
490,529	S26-like protein	2	SP	-	-	✓	-	✓	-	-	-	✓
466,529	S18-like protein	1	SP	-	✓	✓	✓	-	-	-	-	-
376,649	S17-like prneotein	3	PB	-	✓	✓	-	-	-	-	-	✓
357,815	45S RNA	X	SP	-	-	-	-	-	-	-	-	✓
315,382	L3-like protein	2	SP	-	✓	-	-	-	✓	-	-	✓
286,123	glucose-regulated precursor	6	SP	-	✓	✓	✓	-	✓	-	-	✓
285,072	L22e containing protein	2	SP	✓	✓	-	-	✓	✓	-	-	-
278,889	L13 protein	3	PB	-	✓	-	-	-	✓	-	-	✓
277,277	clusterin-like protein	1	MU	-	✓	✓	-	-	✓	-	✓	✓
218,538	H4-like protein	8	SP	-	✓	✓	-	✓	✓	-	-	-
205,055	S16-like protein	9/10*	SP	-	-	-	-	-	✓	✓	✓	✓
201,381	transketolase-like protein	1	SP	-	-	-	-	✓	✓	-	-	-
188,862	S15a-like protein	9/10*	SP	-	-	-	-	✓	✓	-	-	✓
181,819	RIKEN D130020L05 cDNA	1	SP	-	✓	✓	-	-	-	✓	-	✓

Table 5.3: List of highly expressed genes in the dataset, sorted by number of mapped reads. The number of reads mapped to the TSS are shown in column 1, column 2 shows the annotation, column 3 the chromosomal location. Type and detected regulatory elements (Figure 2.4) are outlined in columns 4 - 9. Four different peak types are shown: SP - sharp peak, PB - single dominant peak, MU - multimodal peaks, BR - broad peaks (see Figure 5.13 for corresponding peak shapes). Rows are colour coded, ribosomal-like genes are shown in red, histone-related genes have blue colour, all other gene annotations have green background. *: Due to size restrictions, chromosome 9 and 10 were not separated prior to sequencing [Brinkrolf et al., 2013].

CHAPTER 6

Discussion

This thesis presented two approaches focussing on different aspects of transcription start site analyses in eukaryotic genomes. While the SATYR (**S**eed **A**ssisted **T**argeted assembly of **Y**ield increasing **R**egions) software is a targeted assembly approach which can be used to reconstruct the incomplete 5' and 3' regions of the cDNAs, the bioinformatics pipeline used to process dual library RNA sequencing runs employs also external data sources such as BLAST to annotate verified transcription start sites. Within this last chapter will review both approaches, summarise the results and will highlight advantages as well as possible disadvantages of both approaches.

6.1 TSS identification in the Chinese hamster by RNA sequencing

By employing a two-library-based RNA sequencing approach and a specifically tailored bioinformatics pipeline, in Chapter 5 light was shed on the previously unstudied promoter regions in the Chinese hamster genome on the global scale. For this purpose, EST and CAGE based approaches from previous studies were replaced with a state-of-the-art RNA sequencing technique. This cost-effective method of TSS exploration combined with a specific dual-library setup is ideally suited, because the sequence information is enriched directly at potential transcription start sites rather than distributed throughout complete transcripts. The modular bioinformatics pipeline developed for this study automates sequence data preprocessing, TSS discovery, TSS annotation, and TSS visualization in one workflow. The software detected 6,547 TSSs and assigned 93.66% of these to

known genes. Furthermore, it uncovered 2,227 transcription start sites of genes not yet annotated for the Chinese hamster genome. This fact emphasizes the current draft status of the Chinese hamster genome, especially when compared to the high-quality annotation status of the mouse and the human genome.

An advantage of the approach presented here in this work is that a single experiment can be used to provide insights into several aspects. Notwithstanding that TSS identification is the primary goal, the CHO community can now be supplied with promoter structures for several thousand genes, including promoter types, regulatory elements, expression height, and exact locations of the TSSs. Both knowledge of expression strength and regulatory elements, such as TATA box or DPE, are also valuable parameters when searching for potential high yield promoter constructs. Here, further experiments have to be conducted for those constructs, eventually resulting in a list of endogenous CHO promoters able to replace classical SV40 and similar constructs together with their unintended side effects. In addition to a gene-centric promoter view, this work also took genome wide promoter architecture into account. It was possible to verify motif patterns for seven of the nine tested regulatory elements, including important motifs such as TATA box and INR. Difficulties occurred for the three DCE subunits as well as for BREd, possibly caused by imprecise PSSMs or a lack of activity given the conditions used for this approach.

The work carried out within this project represents a first step in global promoter studies within the Chinese hamster, which may contribute to a more exact and verified annotation of transcription start sites. The combination of experimental and bioinformatics setup has proven to deliver data with high information density usable in several scenarios and is expected to provide even deeper insights when performed on larger input data sets.

Outlook

The initial RNA sequencing run used to detect possible transcription start sites was based on a pooled DNA sample, therefore representing a virtual state of the cell with various combined parameters. It would be of great interest to employ the developed pipeline in conjunction with RNA sequencing experiments based on several conditions and perform a kind of differential promoter study. The results could be used to generate lists of promoters either active in a series of conditions, e.g. in a series of pH concentrations or only active und certain parameters. Such a strategy would allow for the detection of possible inducible promoters which are of great use in biotechnological productions environments.

In direct contrast to the separated approach outlined above, the pooling strategy could be extended, thus combining as many conditions as possible into one RNA sequencing experiment in order to increase the number of detected transcription

start sites. This way, the annotation of the Chinese hamster reference genome in terms of transcription start sites could further be improved.

This however, brings up a question concerning the dynamic range of the sequencing experiment. As discussed, RNA sequencing offers a very broad range of detection, starting from a few reads up to several million mapped reads. While the naive method to increase the number of detected TSSs would only require to an increase of the sequencing output, the question remains if the additional sequencing coverage provided is able to capture a portion of very low expressed genes or if the sequencing coverage is accumulated within the highly expressed genes. In the latter case, no significant increase of the overall TSS count is expected and the number of reads generated for this work can be used as an upper bound for further sequencing experiments.

Following bioinformatics analyses within this work, the verification of the obtained results should be performed using biotechnological methods. These experiments involve the cloning of several of the "Top 20 promoters" (Table 5.3) into CHO cells and the combination of these promoters with reporter genes. This validation process has already started and is currently performed by Anna Wippermann as part of her Ph.D. thesis. First results show that the cloned promoters are indeed active and produce significant amounts of transcripts. However, in comparison to the classical CMV promoters which is used as control promoter, the initial set of endogenous CHO promoters only reaches $\approx 10\%$ of the CMV activity.

6.2 Targeted assembly with SATYR

SATYR was developed from scratch as a seed based, targeted assembly tool. As such, a BWT based index structure was employed to cope with the challenges of next generation sequencing datasets. When the development of SATYR started, no tools with a similar focus were available and simple script based approaches using Python or Perl cannot be scaled up in a reasonable way to handle next generation sequencing datasets of several gigabases. In Chapter 5 it was shown that given a draft reference genome and a single RNA sequencing experiment it is feasible to perform in-depth promoter analyses. The massive reduction of sequencing costs combined with easy-to-use assembly software made the process of constructing draft genomes of even mammalian genome sizes less cost-intensive and as such one of the use cases of SATYR, the targeted assembly of reads without a reference genome became at least in parts obsolete. SATYR however, can still be applied to a wide range of use cases, e.g. to discover regions surrounding a sequence of interest in organisms without any reference genome. The use of the Burrows-Wheeler transformation makes SATYR an ideal tool to deal with data from next generation sequencing projects, as the produced index is both re-usable and fast to access. Concerning its performance, SATYR is able to outclass the only

competing software with similar set of features, Mapsembler, with respect to the average and maximal extension length for the *Escherichia coli* derived test dataset. Mapsembler is able to perform at higher similarity levels, owed to the underlying de Bruijn-graph based *de novo* assembly algorithm. The main flaw of Mapsembler is its limitation to a maximal output of 20 kb per seed, which may limit the tool's usefulness in situations where length of the assembled regions matters more than high accuracy.

The evaluation of the Chinese hamster dataset revealed deficiencies for both tools, Mapsembler and SATYR. While Mapsembler does not seem to be suited to properly handle more than 50 gigabases of sequence data, SATYR is able to perform a targeted assembly on the dataset. Analyses of the CHO-based assemblies provided by SATYR showed that the relatively short length of the extensions was in nearly all cases caused by a lack of supporting reads. As a rule of thumb, for reasonable *de novo* assemblies the sequencing coverage should not be $\leq 40\times$. However, the CHO dataset even before any filtering does not reach this value. As such, the retrieved sequencing coverage of the CHO genome should be assessed as at least fragmentary and the short extensions combined with a decrease of the quality of the assembled sequences can be attributed to lack of coverage and the quality of the sequence reads.

Outlook

As stated above, the observed drop in extension length and sequence similarity is caused by deficiencies of the initial dataset. Therefore further experiments with state of the art Illumina-based sequencing data combined with a considerably increased coverage of $60\times$ to $120\times$ would represent a great opportunity to evaluate the capabilities of SATYR under optimal circumstances.

SATYR was developed with modern multi-core server systems in mind, therefore supporting multi-threaded execution. Benchmarks showed that while assemblies for small to medium numbers of seeds (up to 5,000) perform well and are processed relatively fast, assemblies with a huge number of seeds ($\geq$ 10,000) tend to have a negative impact on the program runtime which is independent from the size of the BWT index and therefore the number of input reads but depends purely on the number of seeds. Additional work with this focus could contribute to a noticeable increase in the program runtime.

Since both datasets used for evaluation revealed minor shortcomings in terms of sequence similarity when compared to reference sequences, additional work in this direction could effectively position SATYR in front of Mapsembler for the majority of use cases currently better covered by Mapsembler. Work on a replacement for the consensus generation functionality has already started and showed promising results for the introduced *Escherichia coli* dataset.

6.3 Final remarks

Throughout this manuscript, the rapid development of biotechnology was recognisable in each chapter and reaches from sequencing techniques to bioinformatic methods of data analysis. Still, there are many unanswered questions and processes of which we only have a basic understanding. Promoters, especially those of eukaryotic organisms are well studied for several decades bus still today new regulatory elements are found, partially with severe effects on the transcription process. The two methods for promoter identification proposed in this work contribute in different ways to eukaryotic promoter research and may help to correct *in silico* predictions, annotate new promoters or to examine previously unassembled regions of interest related to regulatory features. As the both methods are built in a modular way, future developments within promoter research may be integrated into the software.

Bibliography

T. Abeel, Y. Saeys, E. Bonnet, P. Rouzé, and Y. Van de Peer. Generic eukaryotic core promoter prediction using structural features of DNA. *Genome research*, 18 (2):310–23, February 2008a. 28, 29

T. Abeel, Y. Saeys, P. Rouzé, and Y. Van de Peer. ProSOM: core promoter prediction based on unsupervised clustering of DNA physical profiles. *Bioinformatics*, 24(13):i24–31, July 2008b. 28, 29

T. Abeel, Y. Van de Peer, and Y. Saeys. Toward a gold standard for promoter prediction evaluation. *Bioinformatics*, 25(12):i313–20, June 2009. 28

M. D. Adams, J. M. Kelley, J. D. Gocayne, et al. Complementary DNA sequencing: expressed sequence tags and human genome project. *Science (New York, N.Y.)*, 252(5013):1651–6, June 1991. 31

D. Adjeroh, T. Bell, and A. Mukherjee. *The Burrows-Wheeler Transform: Data Compression, Suffix Arrays, and Pattern Matching*. Springer, 2008. 58, 60, 63

M. Albà. Replicative DNA polymerases. *Genome Biology*, 2(1):REVIEWS3002, January 2001. 12

T. K. Albert, K. Grote, S. Boeing, and M. Meisterernst. Basal core promoters control the equilibrium between negative cofactor 2 and preinitiation complexes in human cells. *Genome Biology*, 11(3):R33, January 2010. 126

B. Alberts. *Molecular biology of the cell*. Garland Science, New York, 2002. 12

S. F. Altschul, W. Gish, W. Miller, E. W. Myers, and D. J. Lipman. Basic local alignment search tool. *Journal of Molecular Biology*, 215(3):403–410, October 1990. 31, 108, 115

S. Andrews. FastQC. *A quality control tool for high throughput sequence data*, 2012. http://www. bioinformatics.bbsrc.ac.uk/projects/fastqc/. 89, 91, 104, 112

M. Ashburner, C. Ball, J. Blake, and D. Botstein. Gene Ontology: tool for the unification of biology. *Nature Genetics*, 25(may):25–29, 2000. 119, 120

S. Audic and J. M. Claverie. Detection of eukaryotic promoters using Markov transition matrices. *Computers & chemistry*, 21(4):223–7, January 1997. 28, 29

J.-M. Aury, C. Cruaud, V. Barbe, et al. High quality draft sequences for prokaryotic genomes using a mix of new sequencing technologies. *BMC Genomics*, 9:603, January 2008. 34

J. Banerji, S. Rusconi, and W. Schaffner. Expression of a beta-globin gene is enhanced by remote SV40 DNA sequences. *Cell*, 27(2 Pt 1):299–308, December 1981. 24

M. J. Bauer, A. J. Cox, and G. Rosone. Lightweight BWT Construction for Very Large String Collections. In *CPM 2011*, volume 6661, pages 219–231. Springer, 2011. 66

J. Becker, M. Hackl, O. Rupp, et al. Unraveling the Chinese hamster ovary cell line transcriptome by next-generation sequencing. *Journal of Biotechnology*, 156 (3):227–35, December 2011. 6, 7, 44, 93

D. A. Benson, I. Karsch-Mizrachi, D. J. Lipman, J. Ostell, and D. L. Wheeler. GenBank. *Nucleic acids research*, 36(Database issue):D25–30, January 2008. 108

J. Blom, T. Jakobi, D. Doppmeier, et al. Exact and complete short-read alignment to microbial genomes using Graphics Processing Unit programming. *Bioinformatics*, 27(10):1351–8, May 2011. 7

J. K. Bonfield, K. F. Smith, and R. Staden. A new DNA sequence assembly program. *Nucleic acids research*, 23(24):4992–9, December 1995. 36

G. Booch. *Object Oriented Analysis & Design with Application*. Pearson Education India, 2006. 104

M. Boshart, F. Weber, G. Jahn, et al. A very strong enhancer is located upstream of an immediate early gene of human cytomegalovirus. *Cell*, 41(2):521–30, June 1985. 24

I. Braslavsky, B. Hebert, E. Kartalov, and S. R. Quake. Sequence information can be obtained from single DNA molecules. *Proceedings of the National Academy of Sciences of the United States of America*, 100(7):3960–4, April 2003. 21

K. Brinkrolf, O. Rupp, H. Laux, et al. Chinese hamster genome sequenced from sorted chromosomes. *Nature Biotechnology*, 31(8):694–5, August 2013. 6, 106, 116, 129, 130

S. Burden, Y.-X. Lin, and R. Zhang. Improving promoter prediction for the NNPP2.2 algorithm: a case study using Escherichia coli DNA sequences. *Bioinformatics*, 21(5):601–7, March 2005. 29

T. W. Burke and J. T. Kadonaga. Drosophila TFIID binds to a conserved downstream basal promoter element that is present in many TATA-box-deficient promoters. *Genes & Development*, 10(6):711–724, March 1996. 27, 127

M. Burrows and D. J. Wheeler. A block-sorting lossless data compression algorithm. *Systems Research*, (124), 1994. 58, 60, 63

D. Busam, T. Feldblyum, S. Ferriera, et al. A Sanger/pyrosequencing hybrid approach for the generation of high-quality draft assemblies of marine microbial genomes. *Proceedings of the National Academy of Sciences*, 103(43), 2006. 38

J. E. F. Butler and J. T. Kadonaga. The RNA polymerase II core promoter: a key component in the regulation of gene expression. *Genes & development*, 16(20): 2583–92, October 2002. 24

J. Butler, I. MacCallum, M. Kleber, et al. ALLPATHS: de novo assembly of whole-genome shotgun microreads. *Genome research*, 18(5):810–20, May 2008. 40

P. Carninci, A. Sandelin, B. Lenhard, et al. Genome-wide analysis of mammalian promoter architecture and evolution. *Nature Genetics*, 38(6):626–35, June 2006. 26, 28, 99, 109, 112, 124, 126, 129

K. Cartharius, K. Frech, K. Grote, et al. MatInspector and beyond: promoter analysis based on transcription factor binding sites. *Bioinformatics*, 21(13):2933–42, July 2005. 110

M. J. Chaisson, D. Brinza, and P. a. Pevzner. De novo fragment assembly with short mate-paired reads: Does the read length matter? *Genome research*, 19(2): 336–46, February 2009. 40

G. Chalkley and C. Verrijzer. DNA binding site selection by RNA polymerase II TAFs: A TAFII150 complex recognizes the Initiator. *The EMBO journal*, 18 (17):4835–4845, 1999. 110

L. Chasin and G. Urlaub. Chromosome-wide event accompanies the expression of recessive mutations in tetraploid cells. *Science*, (11):1091–1093, 1975. 2

K. Chen, J. W. Wallis, C. Kandoth, et al. BreakFusion: targeted assembly-based identification of gene fusions in whole transcriptome paired-end sequencing data. *Bioinformatics*, 28(14):1923–4, July 2012. 43

R. Chikhi and G. Rizk. Space-efficient and exact de Bruijn graph representation based on a Bloom filter. *Algorithms for molecular biology : AMB*, 8(1):22, January 2013. 51

P. J. a. Cock, C. J. Fields, N. Goto, M. L. Heuer, and P. M. Rice. The Sanger FASTQ file format for sequences with quality scores, and the Solexa/Illumina FASTQ variants. *Nucleic acids research*, 38(6):1767–71, April 2010. 68

P. E. C. Compeau, P. a. Pevzner, and G. Tesler. How to apply de Bruijn graphs to genome assembly. *Nature Biotechnology*, 29(11):987–91, November 2011. 38

J. Corden, B. Wasylyk, a. Buchwalder, et al. Promoter sequences of eukaryotic protein-coding genes. *Science (New York, N.Y.)*, 209(4463):1406–14, September 1980. 26, 126

A. J. Cox, M. J. Bauer, T. Jakobi, and G. Rosone. Large-scale compression of genomic sequence databases with the Burrows-Wheeler transform. *Bioinformatics*, 28(11):1415–1419, May 2012a. 7, 55, 66, 67, 70

A. J. Cox, T. Jakobi, and G. Rosone. Comparing DNA Sequence Collections by Direct Comparison of Compressed Text Indexes. *Algorithms in Bioinformatics*, pages 214–224, 2012b. 7

F. Crick. On Protein Synthesis. *Symposia of the Society for Experimental Biology*, XII:139–163, 1958. 11

F. Crick. Central Dogma of Molecular Biology. *Nature*, 227:561–563, August 1970. 11

G. Crooks and G. Hon. WebLogo: a sequence logo generator. *Genome research*, pages 1188–1190, 2004. 109

N. G. de Bruijn and P. Erdos. A combinatorial problem. *Koninklijke Nederlandse Akademie v. Wetenschappen*, 49(49):758–764, 1946. 38

M. de Hoon and Y. Hayashizaki. Deep cap analysis gene expression (CAGE): genome-wide identification of promoters, quantification of their expression, and network inference. *Biotechniques*, 44(5):627–8, 630, 632, April 2008. 99

J. Deer and D. Allison. High Level Expression of Proteins in Mammalian Cells Using Transcription Regulatory Sequences from the Chinese Hamster EF1α Gene. *Biotechnology progress*, (3):880–889, 2004. 25

W. Deng and S. G. E. Roberts. A core promoter element downstream of the TATA box that is recognized by TFIIB. *Genes & development*, 19(20):2418–23, October 2005. 26, 110, 126

P. Desjeux. The increase in risk factors for leishmaniasis worldwide. *Transactions of the Royal Society of Tropical Medicine and Hygiene*, 95(3):239–43, 2001. 1

J. C. Dohm, C. Lottaz, T. Borodina, and H. Himmelbauer. SHARCGS, a fast and highly accurate short-read assembly algorithm for de novo genomic sequencing. *Genome research*, 17(11):1697–706, November 2007. 37

T. a. Down and T. J. P. Hubbard. Computational detection and location of transcription start sites in mammalian genomic DNA. *Genome research*, 12(3):458–61, March 2002. 29

R. Dreos, G. Ambrosini, R. Cavin Périer, and P. Bucher. EPD and EPDnew, high-quality promoter resources in the next-generation sequencing era. *Nucleic acids research*, 41(Database issue):D157–64, January 2013. 110

B. E. Eddy, G. S. Borman, W. H. Berkeley, and R. D. Young. Tumors Induced in Hamsters by Injection of Rhesus Monkey Kidney Cell Extracts. *Experimental Biology and Medicine*, 107(1):191–197, May 1961. 24

J. Eid, A. Fehr, J. Gray, et al. Real-Time DNA Sequencing from Single Polymerase Molecules. *Science*, 323(5910):133–138, 2009. 22

L. Euler. Solutio problematis ad geometriam situs pertinentis. *Commentarii academiae scientiarum Petropolitanae*, 8:128–140, 1741. 40

B. Ewing, L. Hillier, M. C. Wendl, and P. Green. Base-Calling of Automated Sequencer Traces UsingPhred. I. AccuracyâAssessment. *Genome research*, 8(3): 175–185, March 1998. 14

M. Fatemi, M. M. Pao, S. Jeong, et al. Footprinting of mammalian promoters: use of a CpG DNA methyltransferase revealing nucleosome positions at a single molecule level. *Nucleic acids research*, 33(20):e176, January 2005. 111

P. Ferragina and G. Manzini. Opportunistic data structures with applications. *Proceedings 41st Annual Symposium on Foundations of Computer Science*, pages 390–398, 2000. 63, 65

P. Ferragina and G. Manzini. Indexing compressed text. *Journal of the ACM (JACM)*, 52(4):552–581, 2005. 63, 73

J. Fickett and A. Hatzigeorgiou. Eukaryotic promoter recognition. *Genome research*, pages 861–878, 1997. 28, 29

R. Fleischmann, M. Adams, and O. White. Whole-genome random sequencing and assembly of Haemophilus influenzae Rd. *Science*, 1995. 32

N. a. Fonseca, J. Rung, A. Brazma, and J. C. Marioni. Tools for mapping high-throughput sequencing data. *Bioinformatics*, 28(24):3169–77, December 2012. 106

R. E. Franklin and R. G. Gosling. Molecular Configuration in Sodium Thymonucleate. *Nature*, 171:740–741, April 1953. 11

E. Fredkin. Trie memory. *Communications of the ACM*, 1960. 45

M. Gardiner-Garden and M. Frommer. CpG islands in vertebrate genomes. *Journal of Molecular Biology*, 196(2):261–82, July 1987. 111

N. I. Gershenzon, E. N. Trifonov, and I. P. Ioshikhes. The features of Drosophila core promoters revealed by statistical analysis. *BMC Genomics*, 7:161, January 2006. 25

M. P. Gerstl, M. Hackl, A. B. Graf, N. Borth, and J. Grillari. Prediction of transcribed PIWI-interacting RNAs from CHO RNAseq data. *Journal of Biotechnology*, 166(1-2):51–7, June 2013. 6

I. M. Gessel and C. Reutenauer. Counting permutations with given cycle structure and descent set. *Journal of Combinatorial Theory, Series A*, 64(2):189–215, November 1993. 65

T. C. Glenn. Field guide to next-generation DNA sequencers. *Molecular ecology resources*, 11(5):759–69, September 2011. 20

M. L. Goldberg. *Sequence analysis of Drosophila histone genes*. PhD thesis, Stanford University, 1979. 26, 126

P. Green. Phrap. *Unpublished, available for download at http://www. genome. washington. edu/UWGC/analysistools/phrap. htm*, 1994. 36

R. Grosschedl and M. Birnstiel. Identification of regulatory sequences in the prelude sequences of an H2A histone gene by the study of specific deletion mutants in vivo. *Proceedings of the National Academy of Sciences*, 77(3):1432–1436, 1980. 26

I. Grummt. Regulation of mammalian ribosomal gene transcription by RNA polymerase I. *Progress in nucleic acid research and molecular biology*, 62:109–154, 1998. 24

M. Hackl, T. Jakobi, J. Blom, et al. 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 (1-2):62–75, April 2011. 7

M. Hackl, V. Jadhav, T. Jakobi, et al. Computational identification of microRNA gene loci and precursor microRNA sequences in CHO cell lines. *Journal of Biotechnology*, 158(3):151–5, April 2012. 7

a. S. Halees. PromoSer: a large-scale mammalian promoter and transcription start site identification service. *Nucleic Acids Research*, 31(13):3554–3559, July 2003. 29

K. D. Hansen, S. E. Brenner, and S. Dudoit. Biases in Illumina transcriptome sequencing caused by random hexamer priming. *Nucleic acids research*, 38(12): e131, July 2010. 116

T. D. Hanson. A hash table for C structures. *http://uthash.sourceforge.net/index.html*, 2009. 71

K. Henckel, K. J. Runte, T. Bekel, et al. TRUNCATULIX–a data warehouse for the legume community. *BMC plant biology*, 9:19, January 2009. 7

D. Hernandez, P. François, L. Farinelli, M. Osterå s, and J. Schrenzel. De novo bacterial genome sequencing: millions of very short reads assembled on a desktop computer. *Genome research*, 18(5):802–9, May 2008. 38

a. J. Herr, M. B. Jensen, T. Dalmay, and D. C. Baulcombe. RNA polymerase IV directs silencing of endogenous DNA. *Science (New York, N.Y.)*, 308(5718): 118–20, April 2005. 24

M. S. Hossain, N. Azimi, and S. Skiena. Crystallizing short-read assemblies around seeds. *BMC Bioinformatics*, 10 Suppl 1:S16, January 2009. 45, 46, 52

D. W. Huang, B. T. Sherman, and R. a. Lempicki. Systematic and integrative analysis of large gene lists using DAVID bioinformatics resources. *Nature protocols*, 4(1):44–57, January 2009a. 119, 120

D. W. Huang, B. T. Sherman, and R. a. Lempicki. Bioinformatics enrichment tools: paths toward the comprehensive functional analysis of large gene lists. *Nucleic acids research*, 37(1):1–13, January 2009b. 119, 120

X. Huang and A. Madan. CAP3: A DNA Sequence Assembly Program. *Genome research*, 9(9):868–877, September 1999. 36

X. Huang, J. Wang, S. Aluru, S.-P. Yang, and L. Hillier. PCAP: a whole-genome assembly program. *Genome research*, 13(9):2164–70, September 2003. 38

D. Huffman. A Method for the Construction of Minimum-Redundancy Codes. *Proceedings of the IRE*, 40(9):1098–1101, September 1952. 66

B. Huggett. Public biotech 2012 - the numbers. *Nature Biotechnology*, 31:697–703, 2013. 4

G. B. Hutchinson. The prediction of vertebrate promoter regions using differential hexamer frequency analysis. *Computer applications in the biosciences : CABIOS*, 12(5):391–8, October 1996. 28, 29

Illumina Inc. HiSeq X Ten System overview. *www.illumina.com/systems/hiseq-x-sequencing-system/system.ilmn*. 15

J. Isler, A. Skalet, and J. Alwine. Human cytomegalovirus infection activates and regulates the unfolded protein response. *Journal of Virology*, 79(11), 2005. 25

S. Istrail, G. G. Sutton, L. Florea, et al. Whole-genome shotgun assembly and comparison of human genome assemblies. *Proceedings of the National Academy of Sciences of the United States of America*, 101(7):1916–21, February 2004. 38, 106

K. Jayapal and K. Wlaschin. Recombinant protein therapeutics from CHO cells-20 years and counting. *Chemical engineering progress*, pages 40–47, 2007. 1

W. R. Jeck, J. a. Reinhardt, D. a. Baltrus, et al. Extending assembly of short DNA sequences to handle error. *Bioinformatics*, 23(21):2942–4, November 2007. 37

V. X. Jin, G. a. C. Singer, F. J. Agosto-Pérez, S. Liyanarachchi, and R. V. Davuluri. Genome-wide analysis of core promoter elements from conserved human and mouse orthologous pairs. *BMC Bioinformatics*, 7:114, January 2006. 110

T. Juven-Gershon. Perspectives on the RNA polymerase II core promoter. *Biochemical Society*, (July):1051–1054, 2006. 25, 26, 27

M. Kanamori-Katayama, M. Itoh, H. Kawaji, et al. Unamplified cap analysis of gene expression on a single-molecule sequencer. *Genome research*, 21(7):1150–9, July 2011. 99

M. Kanehisa and S. Goto. KEGG: kyoto encyclopedia of genes and genomes. *Nucleic acids research*, 28(1):27–30, January 2000. 121

J. J. Kasianowicz, E. Brandin, D. Branton, and D. W. Deamer. Characterization of individual polynucleotide molecules using a membrane channel. *Proceedings of the National Academy of Sciences of the United States of America*, 93(24): 13770–3, November 1996. 23

J. Kececioglu and E. Myers. Combinatorial algorithms for DNA sequence assembly. *Algorithmica*, pages 7–51, 1995. 35, 37

D. Kim, G. Pertea, C. Trapnell, et al. TopHat2: accurate alignment of transcriptomes in the presence of insertions, deletions and gene fusions. *Genome Biology*, 14(4):R36, April 2013. 116

D. Kim, T. Uetsuki, Y. Kaziro, N. Yamaguchi, and S. Sugano. Use of the human elongation factor 1α promoter as a versatile and efficient expression system. *Gene*, 91:217–223, 1990. 25

K. Klepper and F. Drabløs. PriorsEditor: a tool for the creation and use of positional priors in motif discovery. *Bioinformatics*, 26(17):2195–7, September 2010. 29

K. Klepper and F. Drabløs. MotifLab: a tools and data integration workbench for motif discovery and regulatory sequence analysis. *BMC Bioinformatics*, 14: 9, January 2013. 29

R. Kodzius, M. Kojima, H. Nishiyori, et al. CAGE: cap analysis of gene expression. *Nature Methods*, 3(3):211–22, March 2006. 32

A. Kozomara and S. Griffiths-Jones. miRBase: annotating high confidence microRNAs using deep sequencing data. *Nucleic acids research*, 42(1):D68–73, January 2014. 114

J. E. Krebs, E. S. Goldstein, and S. T. Kilpatrick. *Lewin's GENES XI*. Jones & Bartlett Learning, 2012. 12

S. Kumari and D. Ware. Genome-Wide Computational Prediction and Analysis of Core Promoter Elements across Plant Monocots and Dicots. *PloS one*, 8(10): e79011, January 2013. 99, 124

A. Kutach and J. Kadonaga. The downstream promoter element DPE appears to be as widely used as the TATA box in Drosophila core promoters. *Molecular and cellular biology*, 20(13):4754–4764, 2000. 27

T. Lagrange, a. N. Kapanidis, H. Tang, D. Reinberg, and R. H. Ebright. New core promoter element in RNA polymerase II-dependent transcription: sequence-specific DNA binding by transcription factorÂ IIB. *Genes & Development*, 12 (1):34–44, January 1998. 26, 124

E. S. Lander, L. M. Linton, B. Birren, et al. Initial sequencing and analysis of the human genome. *Nature*, 409(6822):860–921, February 2001. 14, 15

B. Langmead and S. L. Salzberg. Fast gapped-read alignment with Bowtie 2. *Nature Methods*, 9(4):357–360, March 2012. 65, 106

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. 115

F. Larsen, G. Gundersen, R. Lopez, and H. Prydz. CpG islands as gene markers in the human genome. *Genomics*, 13(4):1095–107, August 1992. 112

G. Laufer. *Introduction to Optics and Lasers in Engineering*. Cambridge University Press, 1996. 22

T.-Y. Lee, W.-C. Chang, J. B.-K. Hsu, T.-H. Chang, and D.-M. Shien. GPMiner: an integrated system for mining combinatorial cis-regulatory elements in mammalian gene group. *BMC Genomics*, 13 Suppl 1(Suppl 1):S3, January 2012. 29

Y. Lee, M. Kim, J. Han, et al. MicroRNA genes are transcribed by RNA polymerase II. *The EMBO journal*, 23(20):4051–60, October 2004. 24

M. J. Levene, J. Korlach, S. W. Turner, et al. Zero-mode waveguides for single-molecule analysis at high concentrations. *Science (New York, N.Y.)*, 299(5607): 682–6, January 2003. 22

P. A. Levene. The Structure Of Yeast Nucleic Acid. *Studies from the Rockefeller Institute for Medical Research: Reprints*, 36:183, 1921. 11

B. a. Lewis, T. K. Kim, and S. H. Orkin. A downstream element in the human beta-globin promoter: evidence of extended sequence-specific transcription factor IID contacts. *Proceedings of the National Academy of Sciences of the United States of America*, 97(13):7172–7, June 2000. 27, 110, 126, 127

N. E. Lewis, X. Liu, Y. Li, et al. Genomic landscapes of Chinese hamster ovary cell lines as revealed by the Cricetulus griseus draft genome. *Nature Biotechnology*, 31(8):759–65, August 2013. 3, 6, 116

H. Li and R. Durbin. Fast and accurate short read alignment with Burrows-Wheeler transform. *Bioinformatics*, 25(14):1754–1760, July 2009. 65

H. Li and N. Homer. A survey of sequence alignment algorithms for next-generation sequencing. *Briefings in bioinformatics*, 11(5):473–83, September 2010. 40

H. Li, B. Handsaker, A. Wysoker, et al. The Sequence Alignment/Map format and SAMtools. *Bioinformatics*, 25(16):2078–2079, August 2009. 106

R. Li, W. Fan, G. Tian, et al. The sequence and de novo assembly of the giant panda genome. *Nature*, 463(7279):311–7, January 2010. 34

C. Y. Lim, B. Santoso, T. Boulay, et al. The MTE, a new core promoter element for transcription by RNA polymerase II. *Genes & development*, 18(13):1606–17, July 2004. 27, 127

D. J. Lipman and W. R. Pearson. Rapid and sensitive protein similarity searches. *Science (New York, N.Y.)*, 227(4693):1435–41, March 1985. 68

O. Littlefield, Y. Korkhin, and P. B. Sigler. The structural basis for the oriented assembly of a TBP/TFB/promoter complex. *Proceedings of the National Academy of Sciences of the United States of America*, 96(24):13668–73, November 1999. 26

I. Lysov, V. L. Florent'ev, A. A. Khorlin, K. R. Khrapko, and V. V. Shik. Determination of the nucleotide sequence of DNA using hybridization with oligonucleotides. A new method]. *Doklady Akademii Nauk SSSR*, 303(6):1508, 1988. 38

B. Ma, J. Tromp, and M. Li. PatternHunter: faster and more sensitive homology search. *Bioinformatics*, 18(3):440–5, March 2002. 79

D. Maier. The complexity of some problems on subsequences and supersequences. *Journal of the ACM*, 25(2):322–336, April 1978. 36

S. C. Makrides. Components of vectors for gene transfer and expression in mammalian cells. *Protein expression and purification*, 17(2):183–202, November 1999. 24

S. Mantaci, A. Restivo, G. Rosone, and M. Sciortino. An extension of the Burrows Wheeler transform and applications to sequence comparison and data compression. In *Combinatorial Pattern Matching*, pages 427–463. Springer, 2005. 65

M. Margulies, M. Egholm, W. E. Altman, et al. Genome sequencing in microfabricated high-density picolitre reactors. *Nature*, 437(7057):376–380, September 2005. 16, 38

G. a. Maston, S. K. Evans, and M. R. Green. Transcriptional regulatory elements in the human genome. *Annual review of genomics and human genetics*, 7:29–59, January 2006. 25, 126

P. Mayer, L. Farinelli, G. Matton, et al. A Very Large Scale, High Throughput and Low CostDNA Sequencing Method based on a New2-Dimensional DNA Auto-Patterning Process. 1998. 17

P. Mayer, L. Farinelli, G. Matton, and E. Kawashima. Method Of Nucleic Acid Amplification, 2007. 17

K. J. McKernan, H. E. Peckham, G. L. Costa, et al. Sequence and structural variation in a human genome uncovered by short-read, massively parallel ligation sequencing using two-base encoding. *Genome research*, 19(9):1527–41, October 2009. 18

M. Melville, T. Charlebois, W. Mounts, et al. Oligonucleotide Arrays To Monitor Gene Expression And Methods For Making And Using Same, 2005. 5

M. Melville, T. Charlebois, W. Mounts, et al. Novel Polynucleotides Related To Oligonucleotide Arrays To Monitor Gene Expression, 2006. 6

J. R. Miller, A. L. Delcher, S. Koren, et al. Aggressive assembly of pyrosequencing reads with mates. *Bioinformatics*, 24(24):2818–24, December 2008. 38

J. R. Miller, S. Koren, and G. Sutton. Assembly algorithms for next-generation sequencing data. *Genomics*, 95(6):315–327, March 2010. 40

E. W. Myers, G. G. Sutton, A. L. Delcher, et al. A whole-genome assembly of Drosophila. *Science (New York, N.Y.)*, 287(5461):2196–2204, March 2000. 34, 38

T. Namiki, T. Hachiya, H. Tanaka, and Y. Sakakibara. MetaVelvet: an extension of Velvet assembler to de novo metagenome assembly from short sequence reads. *Nucleic acids research*, 40(20):e155, November 2012. 47

L. Narlikar and I. Ovcharenko. Identifying regulatory elements in eukaryotic genomes. *Briefings in functional genomics & proteomics*, 8(4):215–30, July 2009. 29

U. Ohler, G. Stemmer, S. Harbeck, and H. Niemann. Stochastic segment models of eukaryotic promoter regions. *Pacific Symposium on Biocomputing. Pacific Symposium on Biocomputing*, 388:380–91, January 2000. 29

U. Ohler. Identification of core promoter modules in Drosophila and their application in accurate transcription start site prediction. *Nucleic acids research*, 34 (20):5943–50, January 2006. 29

U. Ohler, G. Liao, H. Niemann, and G. Rubin. Computational analysis of core promoters in the Drosophila genome. *Genome Biology*, pages 1–12, 2002. 99

a. O'Shea-Greenfield and S. T. Smale. Roles of TATA and initiator elements in determining the start site location and direction of RNA polymerase II transcription. *The Journal of biological chemistry*, 267(9):6450, March 1992. 26

J. Pellicer, M. F. Fay, and I. J. Leitch. The largest eukaryotic genome of them all? *Botanical Journal of the Linnean Society*, 164(1):10–15, September 2010. 13

Y. Peng, H. C. M. Leung, S.-M. Yiu, and F. Y. L. Chin. IDBA–a practical iterative de Bruijn graph de novo assembler. In *Research in Computational Molecular Biology*, pages 426–440. Springer, 2010. 47

Y. Peng, H. C. M. Leung, S. M. Yiu, and F. Y. L. Chin. Meta-IDBA: a de Novo assembler for metagenomic data. *Bioinformatics*, 27(13):i94–101, July 2011. 47

P. Peterlongo and R. Chikhi. Mapsembler, targeted and micro assembly of large NGS datasets on a desktop computer. *BMC Bioinformatics*, 13(1):48, March 2012. 43, 48, 50, 52, 77, 98

P. A. Pevzner. l-Tuple DNA sequencing: computer analysis. *Journal of biomolecular structure & dynamics*, 7(1):63–73, 1989. 38

P. A. Pevzner, H. Tang, and M. S. Waterman. An Eulerian path approach to DNA fragment assembly. *Proceedings of the National Academy of Sciences of the United States of America*, 98(17):9748–9753, August 2001. 40

P. J. Plauger, M. Lee, D. Musser, and A. A. Stepanov. *C++ Standard Template Library*. Prentice Hall PTR, Upper Saddle River, NJ, USA, 1st edition, 2000. 53, 55

C. Plessy, N. Bertin, H. Takahashi, et al. Linking promoters to functional transcripts in small samples with nanoCAGE and CAGEscan. *Nature Methods*, 7(7): 528–34, July 2010. 99

M. Pop, S. Salzberg, and M. Shumway. Genome sequence assembly: Algorithms and issues. *Computer*, (July):47–54, 2002. 32, 37

D. S. Prestridge. Predicting Pol II promoter sequences using transcription factor binding sites. *Journal of Molecular Biology*, 249(5):923–32, June 1995. 28, 29

D. Pribnow. Nucleotide sequence of an RNA polymerase binding site at an early T7 promoter. *Proceedings of the National Academy of Sciences of the United States of America*, 72(3):784–8, March 1975. 26

K. D. Pruitt, T. Tatusova, and D. R. Maglott. NCBI reference sequences (RefSeq): a curated non-redundant sequence database of genomes, transcripts and proteins. *Nucleic acids research*, 35(Database issue):D61–5, January 2007. 108

T. Puck. Development of the Chinese Hamster Ovary (CHO) Cell. *Molecular Cell Genetics*, 1:37–64, 1985. 2

J. Y. Qin, L. Zhang, K. L. Clift, et al. Systematic comparison of constitutive promoters and the doxycycline-inducible promoter. *PloS one*, 5(5):e10611, January 2010. 24

E. A. Rach, H.-y. Yuan, W. H. Majoros, P. Tomancak, and U. Ohler. Motif composition, conservation and condition-specificity of single and alternative transcription start sites in the Drosophila genome. *Genome Biology*, 10(7):R73, January 2009. 124

M. Ronaghi, S. Karamohamed, B. Pettersson, M. Uhlén, and P. Nyrén. Real-time DNA sequencing using detection of pyrophosphate release. *Analytical biochemistry*, 242(1):84–9, November 1996. 16

M. Ronaghi, M. Uhlén, and P. Nyrén. A sequencing method based on real-time pyrophosphate. *Science (New York, N.Y.)*, 281(5375):363, 365, July 1998. 16

J. G. Ruby, P. Bellare, and J. L. Derisi. PRICE: software for the targeted assembly of components of (Meta) genomic sequence data. *G3 (Bethesda, Md.)*, 3(5): 865–80, May 2013. 47, 52

O. Rupp, J. Becker, K. Brinkrolf, et al. Construction of a Public CHO Cell Line Transcript Database Using Versatile Bioinformatics Analysis Pipelines. *PloS one*, 9(1):e85568, January 2014. 120

A. Sandelin, P. Carninci, B. Lenhard, et al. Mammalian RNA polymerase II core promoters: insights from genome-wide studies. *Nature reviews. Genetics*, 8(6): 424–36, June 2007. 99, 124, 127

F. Sanger, G. M. Air, B. G. Barrell, et al. Nucleotide sequence of bacteriophage phi X174 DNA. *Nature*, 265(5596):687–695, February 1977a. 13

F. Sanger, S. Nicklen, and A. R. Coulson. DNA sequencing with chain-terminating inhibitors. *Proceedings of the National Academy of Sciences of the United States of America*, 74(12):5463–5467, December 1977b. 13

F. Sanger, A. Coulson, and G. Hong. Nucleotide sequence of bacteriophage λ DNA. *Journal of molecular biology*, pages 729–773, 1982. 32

S. Saxonov, P. Berg, and D. L. Brutlag. A genome-wide analysis of CpG dinucleotides in the human genome distinguishes two distinct classes of promoters. *Proceedings of the National Academy of Sciences of the United States of America*, 103(5):1412–7, January 2006. 111, 126, 128

B. Schaeling. *The Boost C++ libraries*. XML Press, Laguna Hills, California, 2011. 53, 55

W. F. Scherer, J. T. Syverton, and G. O. Gey. Studies on the propagation in vitro of poliomyelitis viruses IV. Viral multiplication in a stable strain of human malignant epithelial cells (strain HeLa) derived from an epidermoid carcinoma of the cervix. *The Journal of experimental medicine*, 97(5):695–710, 1953. 26

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*, 155(1):68–77, August 2011. 17

T. Shiraki, S. Kondo, S. Katayama, et al. Cap analysis gene expression for high-throughput analysis of transcriptional starting point and identification of promoter usage. *Proceedings of the National Academy of Sciences of the United States of America*, 100(26):15776–81, December 2003. 32

J. Siek. *The boost graph library: user guide and reference manual*. Addison-Wesley, Boston, 2002. 53

A. Simon and M. Taylor. Model involving gene inactivation in the generation of autosomal recessive mutants in mammalian cells in culture. *Molecular and cellular biology*, 1982. 2

J. T. Simpson and R. Durbin. Efficient de novo assembly of large genomes using compressed data structures. *Genome research*, 22(3):549–56, March 2012. 38

J. T. Simpson, K. Wong, S. D. Jackman, et al. ABySS: a parallel assembler for short read sequence data. *Genome research*, 19(6):1117–23, June 2009. 40

S. T. Smale and D. Baltimore. The "initiator" as a transcription control element. *Cell*, 57(1):103–13, April 1989. 126

S. T. Smale. Transcription initiation from TATA-less promoters within eukaryotic protein-coding genes. *Biochimica et Biophysica Acta (BBA)-Gene Structure and Expression*, 1351(1):73–88, 1997. 27

A. Smith. *Oxford dictionary of biochemistry and molecular biology*. Oxford University Press, Oxford New York, 1997. 12

D. R. Smith, A. R. Quinlan, H. E. Peckham, et al. Rapid whole-genome mutational profiling using next-generation sequencing technologies. *Genome research*, 18 (10):1638–42, October 2008. 19

E. Solomon. *Biology*. Brooks/Cole Thomson Learning, Belmont, CA, 2005. 12

V. Solovyev and a. Salamov. The Gene-Finder computer tools for analysis of human and model organisms genome sequences. *Proceedings / ... International Conference on Intelligent Systems for Molecular Biology ; ISMB. International Conference on Intelligent Systems for Molecular Biology*, 5:294–302, January 1997. 29

S. Sonnenburg, A. Zien, and G. Rätsch. ARTS: accurate recognition of transcription starts in human. *Bioinformatics*, 22(14):e472–80, July 2006. 28, 29

E. M. Southern. United Kingdom patent application GB8810400. *International patent application PCT GB*, 89:460, 1988. 38

R. Staden. The Staden sequence analysis package. *Molecular biotechnology*, 5(3): 233–41, June 1996. 112

J. E. Stajich, D. Block, K. Boulez, et al. The Bioperl toolkit: Perl modules for the life sciences. *Genome research*, 12(10):1611–8, October 2002. 104

G. Stormo and T. Schneider. Use of the 'Perceptron'algorithm to distinguish translational initiation sites in E. coli. *Nucleic Acids Research*, 10:2997–3012, 1982. 109

G. G. Sutton, O. White, M. D. Adams, and A. R. Kerlavage. TIGR Assembler: A New Tool for Assembling Large Shotgun Sequencing Projects. *Genome Science and Technology*, 1(1):9–19, January 1995. 36

Y. Suzuki, H. Taira, T. Tsunoda, et al. Diverse transcriptional initiation revealed by fine, large-scale mapping of mRNA start sites. *EMBO reports*, 2(5):388–93, May 2001. 26, 126, 129

K. a. Swan, D. E. Curtis, K. B. McKusick, et al. High-throughput gene mapping in Caenorhabditis elegans. *Genome research*, 12(7):1100–5, July 2002. 38

J. Tarhio and E. Ukkonen. A greedy approximation algorithm for constructing shortest common superstrings. *Theoretical Computer Science*, 57, 1988. 34

B. Tirosh, N. Iwakoshi, and B. Lilley. Human cytomegalovirus protein US11 provokes an unfolded protein response that may facilitate the degradation of class I major histocompatibility complex products. *Journal of virology*, 79(5):2768–2779, 2005. 25

J. Tjio and T. Puck. Genetics of somatic mammalian cells: II. Chromosomal constitution of cells in tissue culture. *The Journal of experimental medicine*, 1958. 2

W. Trobin. The Boost Graph Library. *ist.tugraz.at*, 2002. 75

E. Valen, G. Pascarella, A. Chalk, et al. Genome-wide detection and analysis of hippocampus core promoters using DeepCAGE. *Genome research*, 19(2):255–65, February 2009. 99

L. Wang and T. Jiang. On the complexity of multiple sequence alignment. *Journal of computational biology : a journal of computational molecular cell biology*, 1 (4):337–48, January 1994. 38

X. Wang, Z. Xuan, X. Zhao, Y. Li, and M. Q. Zhang. High-resolution human core-promoter prediction with CoreBoostHM. *Genome research*, 19(2):266–75, February 2009. 29

R. L. Warren and R. a. Holt. Targeted assembly of short sequence reads. *PloS one*, 6(5):e19816, January 2011. 46, 48, 49, 52

R. L. Warren, G. G. Sutton, S. J. M. Jones, and R. a. Holt. Assembling millions of short DNA sequences using SSAKE. *Bioinformatics*, 23(4):500–1, March 2007. 37, 46

J. D. Watson and F. H. Crick. Molecular structure of nucleic acids; a structure for deoxyribose nucleic acid. *Nature*, 171(4356):737–738, April 1953a. 11

J. D. Watson and F. H. C. Crick. Genetical Implications of the Structure of Deoxyribonucleic Acid. *Nature*, 171:964–967, 1953b. 11

K. Wetterstrand. DNA Sequencing Costs: Data from the NHGRI Genome Sequencing Program (GSP), 2014. 15

M. E. Wiebe, F. Becker, R. Lazar, et al. A Multifaceted Approach to Assure That Recombinant tPA is Free of Adventitious Virus. *Advances in animal cell biology and technology*, pages 66–71, 1989. 4

A. Wierzbicki, T. Ream, J. Haag, and C. Pikaard. RNA polymerase V transcription guides ARGONAUTE4 to chromatin. *Nature Genetics*, 41(5):630–634, 2009. 24

Wikipedia. Trie — Wikipedia The Free Encyclopedia, 2014. 46

I. Willis. RNA polymerase III. *EJB Reviews 1993*, 11:1–11, 1994. 24

S. Wu, X. Xie, A. Liew, and H. Yan. Eukaryotic promoter prediction based on relative entropy and positional information. *Physical Review E*, 75(4):041908, April 2007. 28

F. M. Wurm. The industry's workhorse - Mammalian expression systems. *Modern Biopharmaceuticals*, pages 723–759, 2005. 4

X. Xie, S. Wu, K.-M. Lam, and H. Yan. PromoterExplorer: an effective promoter identification method based on the AdaBoost algorithm. *Bioinformatics*, 22(22): 2722–8, November 2006. 28

X. Xu, H. Nagarajan, N. E. Lewis, et al. The genomic sequence of the Chinese hamster ovary (CHO)-K1 cell line. *Nature Biotechnology*, (July), July 2011. 6, 89, 115

G. Yerganian. Pathology of Hamsters. *Prog. Exp. Tumor Res. Res*, pages 2–41, 1972. 2

G. Yerganian. The biology and genetics of Chinese hamster. *Molecular Cell Genetics*, pages 3–36, 1985. 2

D. R. Zerbino and E. Birney. Velvet: algorithms for de novo short read assembly using de Bruijn graphs. *Genome research*, 18(5):821–829, May 2008. 47, 49

D. R. Zerbino, G. K. McEwen, E. H. Margulies, and E. Birney. Pebble and rock band: heuristic resolution of repeats and scaffolding in the velvet short-read de novo assembler. *PLoS One*, 4(12):e8407, 2009. 40

Z. Zhang, S. Schwartz, L. Wagner, and W. Miller. A greedy algorithm for aligning DNA sequences. *Journal of computational biology : a journal of computational molecular cell biology*, 7(1-2):203–14, 2000. 79

S. Zhao, W.-P. Fung-Leung, A. Bittner, K. Ngo, and X. Liu. Comparison of RNA-Seq and Microarray in Transcriptome Profiling of Activated T Cells. *PLoS ONE*, 9(1):e78644, January 2014. 121

X. Zhao, E. Valen, B. J. Parker, and A. Sandelin. Systematic clustering of transcription start site landscapes. *PloS one*, 6(8):e23409, January 2011. 32

Acknowledgments

First of all, I wish to thank my supervisors, namely Prof. Dr. Alexander Goesmann, Prof. Dr. Alfred Pühler and Prof. Dr. Jens Stoye, who provided me with the opportunity to carry out this thesis under their supervision. I am grateful for the helpful advices and constructive criticism throughout the whole Ph.D. project.

Thanks also go to Prof. Dr. Sven Rahmann for his willingness to examine this work as external reviewer.

I would like to thank Dr. Karina Brinkrolf and Prof. Dr. Andreas Tauch for their expertise and contributions to the Chinese hamster related publications and Dr. Jochen Blom and Sebastian Jänicke for many valuable discussions and for proof-reading the manuscript. Not unmentioned should be all other members of the Bioinformatics Resource Facility (BRF) and the CeBiTec Support Team for their technical expertise.

I want to acknowledge the funding granted by the "CLIB Graduate Cluster Industrial Biotechnology" and the additional courses and opportunities provided by the cluster. I also would like to acknowledge the additional funding by the German Federal Ministry of Education and Research within the "ENHANCE" and "SulfoSYS$_{Biotec}$" programs.

Special thanks go to my parents and my friends, who supported me through all these years in all imaginable aspects. Finally, I wish to thank my girlfriend Tina for her support, her patience, and emotional and motivational support throughout this Ph.D. project.

Bielefeld, June 2014
Tobias Jakobi

ERKLÄRUNG

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

Bielefeld, Juni 2014

Tobias Jakobi