universität
wien

Diplomarbeit

A Novel Compression Approach for Mapped High-Throughput Sequencing Data Sets

Verfasser

Niko Popitsch

angestrebter akademischer Grad

Magister der Naturwissenschaften (Mag.rer.nat.)

Wien, im September 2012

Studienkennzahl lt. Studienblatt:	A 490
Studienrichtung lt. Studienblatt:	Molekulare Biologie
Betreuer:	Univ.-Prof. Dr. Arndt von Haeseler

Abstract

A major challenge of current high-throughput sequencing (HTS) experiments is not only the generation of the sequencing data itself but also their processing, storage and transmission. The enormous size of these data motivates the development of data compression algorithms usable for the implementation of the various storage policies that are applied to the produced intermediate and final result files.

This thesis gives a brief introduction into the field of high-throughput nucleic acid sequencing and into current approaches for the compression of the data resulting from such experiments. In the main part of the thesis, NGC, a tool for the compression of mapped *read* data stored in the SAM format (one kind of HTS data), is presented. NGC enables lossless and lossy compression and introduces two novel ideas: First, it contains a way to reduce the number of required code words by exploiting common features of the sequenced *reads* mapped to the same genomic positions; second, it contains a highly configurable way for the quantization of per-base quality values which takes their influence on downstream analyses into account.

NGC, evaluated with several real-world data sets, saves 33-66% of disc space using lossless and up to 98% disc space using lossy compression. By applying two popular variant and genotype prediction tools to the decompressed data, we show that the lossy compression modes preserve over 99% of all called variants while outperforming comparable methods in some configurations.

Zusammenfassung

Eine der größten aktuellen Herausforderungen im Zusammenhang mit Hochdurchsatz-Sequenzierungsexperimenten (High-Throughput Sequencing, HTS) liegt nicht im Erzeugen der Daten selbst, sondern in deren Prozessierung, Speicherung und Übertragung. Die enorme Größe dieser Daten motiviert die Entwicklung von Datenkompressionsalgorithmen für die Realisierung der verschiedenen Datenspeicherkonzepte die auf die produzierten (Zwischen-)Ergebnisse von HTS Experimenten angewandt werden.

Die vorliegende Arbeit gibt einen Überblick über das Feld der Hochdurchsatz-Nukleinsäure-Sequenzierung und in aktuelle Ansätze für die Kompression solcher Daten. Im Hauptteil der Arbeit wird NGC vorgestellt, ein Werkzeug für die Kompression von gemappten *reads* die im weitverbreiteten SAM Format gespeichert sind (eine Art von HTS Daten). NGC ermöglicht sowohl verlustfreie als auch verlustbehaftete Kompression und beinhaltet zwei neuartige Ideen: Erstens enthält es eine Methode zur Reduktion der erforderlichen Code-Wörter, welche gemeinsame Merkmale der *reads* die an dieselbe genomische Position gemappt wurden ausnützt. Zweitens beinhaltet NGC eine konfigurierbare Methode für die Quantisierung der Qualitätswerte welche deren Einfluss auf nach-gelagerte Anwendungen berücksichtigt.

NGC, mit mehreren echten Datensätzen evaluiert, spart 33-66% des benötigten Speicherplatzes bei verlustfreier und bis zu 98% des benötigten Speicherplatzes bei verlustbehafteter Kompression ein. Durch die Anwendung zweier gängiger Varianten- und Genotyp-Vorhersagewerkzeuge auf die dekomprimierten Daten wird gezeigt, dass die verlustbehaftete Kompression, besser als vergleichbare Werkzeuge in manchen Konfigurationen, über 99% der gefundenen Varianten präserviert.

Table of Contents

1 Introduction **1**

2 Nucleic Acid Sequencing **3**

2.1 Nucleic Acid Preparations . . . 3
2.2 Sanger Sequencing . . . 6
2.3 High-throughput Sequencing . . . 8
2.4 Error Sources in Current Sequencing Methods . . . 13
2.5 Summary . . . 15

3 High-throughput Sequencing Data **17**

3.1 Representation of Raw Reads: FASTQ . . . 17
3.2 Mapping and Alignment . . . 17
3.3 Representation of Mapped Reads: SAM/BAM . . . 18
3.4 Compression of HTS Data . . . 19

4 NGC: Lossless and Lossy Compression of Aligned High-throughput Sequencing Data **21**

4.1 Introduction . . . 21
4.2 Materials and Methods . . . 22
4.3 Results . . . 29
4.4 Discussion . . . 33

5 Conclusions **35**

5.1 Future Work . . . 35

I Addenda **37**

A Supplementary Data **39**

A.1 Supplementary Description of Variant Calling Pipelines . . . 39
A.2 Computing Environment . . . 39
A.3 GATK Pipeline . . . 40
A.4 SAMTOOLS Pipeline . . . 40

A.5 Definitions 41
A.6 Counting Horizontal and Vertical Run-lengths 41
A.7 Compression Parameters 42
A.8 Supplementary Data Tables 42

Bibliography **49**

Curriculum Vitae **57**

Chapter 1

Introduction

High-throughput sequencing (HTS) refers to a set of novel technologies that enable the accurate, fast and affordable sequencing of long stretches of nucleic acids (DNA, RNA). The last decade brought an amazing boom of these technologies which changed the biology research landscape considerably. Even small biology laboratories can today (and increasingly in the near future) resort to such sequencing data which adds genome scientific approaches, such as RNA-sequencing (RNA-seq) or whole genome sequencing (WGS) to the set of standard laboratory methods. By this, much expensive (in terms of time and money) work in the wet-lab can be saved or planned more accurately and it is expected that many future life science advances will be founded in the availability of accurate sequencing data [Col10, HHR10, KK10, LMD+12].

The ongoing advent of HTS technologies will lead to a several-fold increase of the produced sequencing data in the near future. The enormous size of the produced data, however, introduces new challenges. Today, a major challenge of HTS experiments is not only the generation of the sequencing data itself but also their processing, storage and transmission [Kah11, WRB+12, LBB12]. Many sequencing data sets have to be stored for a long time (e.g., experimental data for reasons of scientific reproducibility, medical records for legal reasons, etc.). Further, such data sets have to be transferred over networks (e.g., for reasons of data exchange, in cloud computing environments, etc.) which introduces additional costs and project delays due to networking bandwidth limits. This consequently motivates the development of data compression algorithms usable for the implementation of the various storage policies that are applied to the produced intermediate and final result files.

This thesis is concerned with the compression of one important type of such data files: mapped read data. Current HTS technologies produce files containing millions of short (100-400bp) reads. An early step of most HTS data analysis pipelines is to map these reads against a reference genome. Such mapped read files, often several Gigabytes in size, are then usually subject to various filtering and analysis steps that produce many intermediate and final result files that need to be analyzed, transferred and archived. The main part (Chapter 4) of this thesis presents a novel approach for the lossless and lossy compression of such mapped read data.

This thesis is organized as follows: In Chapter 2 we give a brief introduction in the field of nucleic acid sequencing in general and into current high-throughput methods in particular. Chapter 3 describes prevalent data formats for the representation of HTS data and highlights current trends in the compression of these. Chapter 4 presents NGC, a tool for the lossless and lossy compression of mapped read data and Appendix A contains associated supplementary data. Finally, Chapter 5 concludes with an outlook on the development of HTS technologies and possible future work.

Chapter 2

Nucleic Acid Sequencing

Nucleic acid sequences (e.g., RNA, DNA) are unbranched polymers of nucleotides joined together by phosphodiester linkages. They are considered as the primary carriers of genetic information and play central roles in all living organisms. The primary structure of nucleic acid sequences can be represented by a string of base characters (A,C,T/U,G) derived by reading the sequence in 5' to 3' direction [AJL+02].

The determination of the digital representation of nucleic acid sequences is valuable for numerous theoretical and applied fields of biology. A generalized pathway for their determination with today's HTS technologies is depicted in Figure 2.1. After some general preparations, so-called sequencing libraries are created. These are then sequenced in a massively parallel, iterative fashion as described in Section 2.3. Sequencing results in large numbers of short sequence fragments ("reads") that are assembled by special software. Finally, researchers can analyze, annotate, exchange and archive the resulting sequence data sets. In the following, we exemplary describe the main steps of DNA extraction from cells that were subject to some biological experiment (cf. [DD87, MDP88, LZL+91, SR01]).

2.1 Nucleic Acid Preparations

Before DNA can be sequenced, it has to be extracted and purified. DNA extraction from tissues, cells or cell compartments are standard laboratory routines and researchers may resort to a large number of protocols for this purpose. Here, we exemplary describe the main steps of DNA extraction from cells that were subject to some biological experiment.

Cell breakage and extraction. First, cells and tissues have to be disrupted in a controlled fashion to expose the contained DNA. Methods for cell breakage include mechanical (e.g., sonication, bead milling, mechanical homogenization, etc.) and/or enzymatic methods (e.g., lysozyme or zymolyase digestion).

Next, the DNA has to be separated as good as possible from bound proteins, salts and cell debris as these would interfere with the sequencing reactions. Bound proteins may be degraded using protease or peptidase enzymes, subsequent protein precipitation can be done for example by salting out with ammonium or sodium acetate. DNA is then usually separated from this mixture by phenol-chloroform extraction: the cell debris/DNA solution is mixed with a water-saturated 1:1 mixture of phenol and chloroform. After centrifugation, the DNA can be found in the interphase between the lower organic (chloroform) phase and the aqueous phase. The DNA is then precipitated from the solution using ice-cold ethanol or isopropanol as it is insoluble in these alcohols. The alcohol further removes the salts added in the protein precipitation stage.

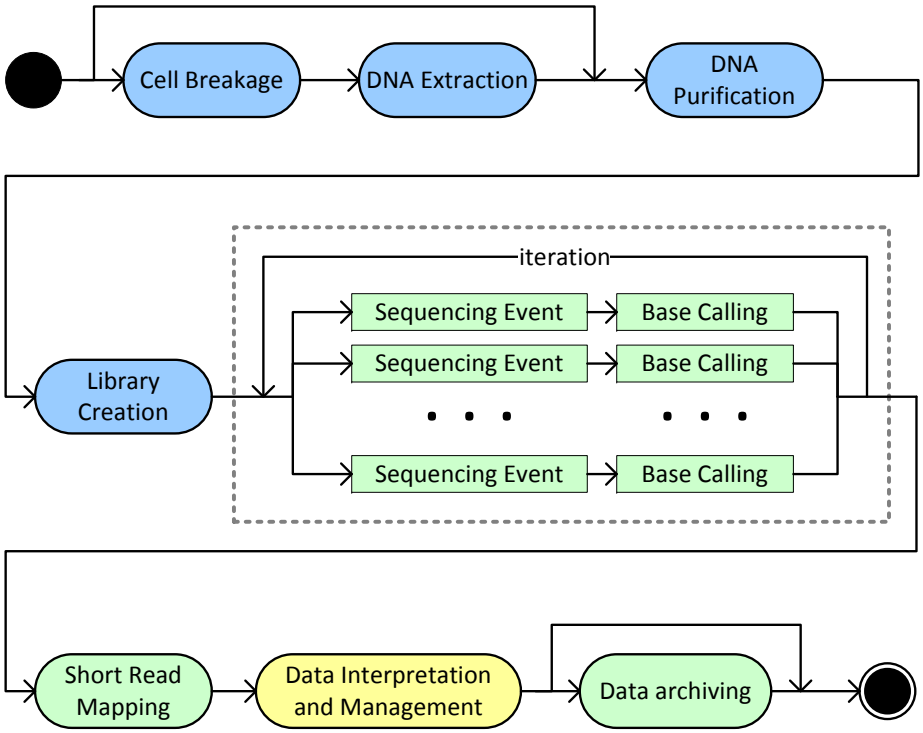


Figure 2.1: Generalized pathway of nucleic acid sequencing experiments. Blue activities take place in the wet-lab, green ones on special-purpose hardware (e.g., sequencing machinery, GPUs, Hardware RAIDs). Data interpretation and management needs, in many cases, to take place on regular lab PCs. The boxed center region of the figure shows the massively parallel sequencing steps of todays high-throughput sequencing technologies.

Purification. DNA samples for HTS sequencing have to be very pure to avoid background noise in the sequencing reactions. Therefore, samples are usually purified (sometimes also in intermediate steps of the above-mentioned extraction procedure) using column or gel purification or membrane filtration before sequencing libraries are created. PCR products are usually prepared for DNA sequencing by ultrafiltration or by cutting the product out of an electrophoresis gel in order to get rid of the PCR by-products (primers, nucleotides, etc.). The required amounts of purified DNA depend on the sequencing method and the length of the fragment of interest. Usual ranges are between several *ng* up to a few μg.

2.1.1 Sequencing Library Creation

Purified DNA is a precondition for the creation of so-called *sequencing libraries*. In general, these preparation steps include DNA fragmentation and end-modifications (e.g., creation of blunt or overhanging fragment ends). Usually, the DNA fragments are ligated to some platform-specific adapter sequences that are in turn required for DNA immobilization. The exact protocols for library creation are available from the respective vendor Web-sites and are not reproduced here. All current technologies are, however, based on the principle of “shotgun sequencing” which requires respective library preparation steps.

Shotgun sequencing. In general terms, the idea of shotgun sequencing is to (i) break the sample DNA into random fragments, (ii) amplify and sequence these fragments and (iii) concatenate and merge them based on overlapping sequence regions. By this, the original DNA sequence can be assembled from a large number of short sequence fragments. These fragments are called *reads* and overlapping reads are assembled into so-called sequence *contigs* (contiguous regions). Adjacent contigs are then concatenated into so-called *scaffolds* (aka *supercontigs*), see Figure 2.2 [GWL+05].

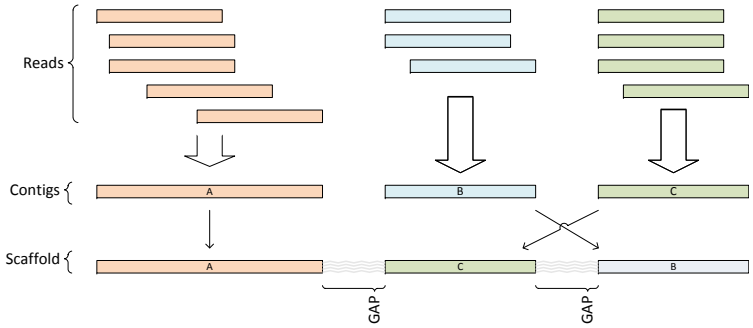


Figure 2.2: Overlapping reads are merged to contigs. Gaps between contigs can be bridged by paired-end reads from different contigs to determine the contig sequence on a scaffold. The figure is a modified reprint from [GWL+05], p. 396.

Modern sequencing technologies use the same principle for the reconstruction of DNA sequences: First, the DNA is split up randomly into smaller subsequences that act as templates for the sequencing process. Historically, this was done by treating the DNA either with restriction enzymes or by using physical methods (e.g., sonication or shearing). Today, enzymatic methods, such as *in vitro* transposition, are emerging. Here, a transposase enzyme catalyzes the fragmentation (and at the same time also the insertion of adaptor sequences) of the DNA template in a single step while older methods require many steps (such as end-modifications, A-tailing and adaptor ligation) to achieve the same results [CGS09, AMA+10].

In the original shotgun approach, these DNA fragments are then cloned into an expression vector (e.g., a high-copy-number plasmid) and these vectors are transferred into some host cells (viruses, bacteria). There, they are copied (cloned) by making use of the cell's replication machinery which results in a large number of short DNA fragments that are subsequently sequenced. Modern sequencing technologies are, however, usually based on PCR amplification and clones are thus created by selecting appropriate primer sequences. The templates are then "read" (sequenced) by the used sequencing technology which results in a (usually very large) number of short *reads*.

Sequence assembly. Finally, the actual DNA sequence is reconstructed from these reads using special sequence assembly software. There are two different ways to do this: mapping and *de-novo* assembly.

Mapping means that the reads are mapped against some known reference sequence. This mapping process aims at finding the position in the reference sequence where a read fits "best", which does not mean that there has to be a 100% accordance. The original sequence can then be determined by considering the (overlapping) mapped reads. Such a reconstructed sequence may then, for example, be compared with the reference sequence which reveals genetic differences between them, e.g., single nucleotide polymorphisms (SNPs) or larger structural variations (SVs).

When no reference sequence is available, however, reads have to be assembled in the way described above for shotgun sequencing: *de-novo* assembly software first exploits overlaps of the read sequences to merge them into longer sequences called *contigs*. Contigs can then be merged into so-called *scaffolds* by

exploiting knowledge about the distance of paired ends (*mate pairs*). Multiple such assemblies may then, for example, be used to build a reference sequence for a certain species.

Paired ends. Paired ends refer to the two ends of one DNA fragment. They are separated by usually short (100-500bp) sequences of unknown DNA. Using sequence information from both ends of a DNA fragment of which the length is known greatly helps to reason upon structural rearrangements like insertions or deletions between these reads or to map them across repetitive regions. For example, when two paired ends are known to be separated by 500bp of DNA, but are mapped to positions that are 1000bp apart, it is likely that an insertion took place between them. Paired end information is usually an output of sequencer technology, i.e., this information is available in the resulting raw data sets. *Mate pairs* are paired ends that are separated by longer DNA sequences (due to different creation methods). As they "cover" longer distances when compared with paired reads, they may help in detecting more structural rearrangements (cf. [SPR+05]).

Coverage. One important parameter of sequence assembly is *coverage*. The coverage of a particular nucleotide of the reconstructed sequence is the average number of reads that were actually mapped to overlap with this nucleotide's position. The coverage of an assembly can thus be calculated with the simple formula $cov = n \times \frac{l}{g}$ where n is the number of reads, l is the average length of these reads and g is the length of the reference sequence. Obviously, high coverage is desirable as it may be used to compensate for errors in individual read sequences. Modern sequencing approaches allow high coverages which, however, comes at the cost of larger data sets resulting in high computational and storage demands.

After discussing how sequencing libraries are prepared and how sequenced reads are used for sequence assembly, we now continue by describing how these reads are actually created by the various sequencing technologies.

2.2 Sanger Sequencing

The first modern approach to DNA sequencing was published by Frederick Sanger et al. in 1977 [SNC77] and "Sanger sequencing" is still the biochemical foundation of the majority of today's HTS approaches [SJ08]. It is based on the inhibitory activity of dideoxynucleotide triphosphates (ddNTPs) on DNA polymerase I.

ddNTPs lack a 3'-OH group on their deoxyribose sugar which is where the polymerase would attach a subsequent nucleotide in a replication process (Figure 2.3). Usually a phosphodiester bond would be created in a condensation reaction of the 5' phosphate of an dNTP and the 3'-OH of the previous nucleotide. However, the lack of this 3'-OH group in ddNTPs inhibits this chain elongation event which is the basis of this so-called dideoxy chain-termination DNA sequencing method. In the following, we describe the main steps of this method in more detail.

Double-strand separation and primer annealing. In a first step, the strands of purified double-stranded DNA (dsDNA) are separated by heating them. This heating process leads to the breakage of hydrogen bonds and Van-der-Waal (VdW) interactions between the two complementary strands which results in their separation.

Primer attachment. In the following, short oligonucleotide primers (about 20nt long) are added to this heated mixture of single-stranded DNA (ssDNA). The mixture is cooled and complimentary DNA strands reattach. As the short primer sequences are more agile when compared with the two complimentary DNA strands, it is much more frequent that primers rather than complimentary DNA strands anneal with a DNA template strand. Thus, this step results in a large number of DNA strands with an attached, short DNA primer.

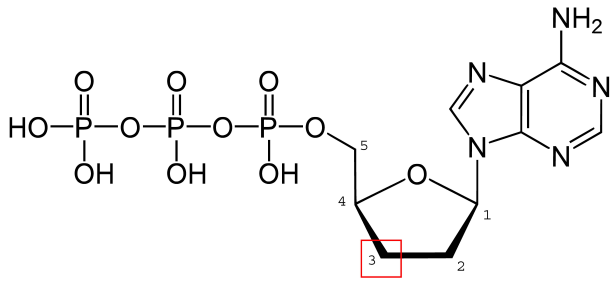


Figure 2.3: 2',3'-dideoxyadenosine triphosphate (ddATP). The red box shows the position of the missing -OH group. The figure is a modified reprint of http://upload.wikimedia.org/wikipedia/commons/b/b2/Desoxyadenosintriphosphat_protoniert.svg, in Wikipedia. Retrieved June 24, 2012.

Strand elongation by DNA polymerase. After this, DNA polymerase I as well as dNTPs (dATP, dGTP, dCTP, dTTP) are added to this mixture. The polymerase sequentially elongates the primed DNA fragments by incorporating the dNTPs that are complimentary to the respective nucleotide on the template strand as done during cell replication.

Termination by dideoxynucleotide triphosphates. Now, however, dye-labeled ddNTPs (ddATP, ddGTP, ddCTP, ddTTP, see Figure 2.3) that lack the above-mentioned 3'-OH groups are added in small concentrations to this mixture. These ddNTP are randomly incorporated into the elongated DNA sequences by the polymerase enzyme. As explained above, this ultimately stops the elongation of the respective growing DNA strand. As the concentration of the ddNTPs is low, this incorporation event is rather rare. However, all possible lengths of the replicated DNA fragments will occur in this reaction mixture for statistical reasons, although in possibly differing concentrations. To increase the yield of this reaction one may repeat the above-listed phases multiple times in a thermal cycler as known from regular PCR experiments. Note that an increased reaction yield means also to be able to successfully apply this method with less original DNA template.

Size separation. Actual sequence determination is then done by sorting the replicated fragments by size (length) and then "reading" them sequentially. For this, the complimentary strands are again separated by heating. Size sorting of the replicated strands is done by capillary electrophoresis. In this process, a thin capillary is filled with gel and the reaction mixture from the previous step is loaded at one end. An electric field is applied and as DNA molecules are negatively charged due to their phosphate backbone, the DNA fragments are pulled through the gel. The speed at which the fragments travel through the gel is, however, determined by their length: the shorter a sequence, the faster it travels as there is less steric hindrance with the molecules in the gel.

Measurement. When the size-sorted DNA fragments finally leave the capillary, a laser excites the dye of the terminating ddNTP on each sequence. The excited dye then emits photons of a specific wavelength due to fluorescence that are recorded by a special sensor (a photocell). Each ddNTP is labeled with a different dye that results in different wavelength of the actually emitted radiation[1].

[1] $\lambda_G = 540nm, \lambda_A = 570nm, \lambda_T = 595nm, \lambda_C = 620nm$

Digital electropherogram. The sensor data is then transmitted to a computer that stores this data in the form of an electropherogram (Figure 2.4). This electropherogram is the basis of the subsequent base-calling process that assigns an actual sequence of base characters (A,C,T,G) to the peaks of the measured radiation.

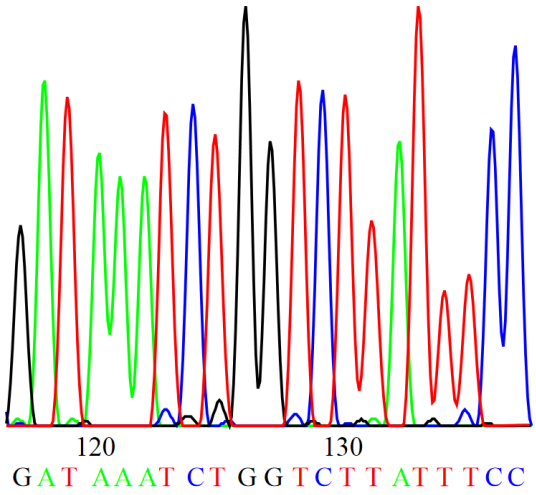


Figure 2.4: Electropherogram. The figure is a modified reprint of http://commons.wikimedia.org/wiki/File:DNA_sequence.svg, in *Wikipedia.* Retrieved August 17, 2012.

2.3 High-throughput Sequencing

Current Sanger sequencing techniques were reported to sequence about 6 Mb of DNA per day at costs of about $500 per Mb [KK10]. However, in the recent past several alternative techniques have been developed that reach much higher throughput at lower costs. These so-called *next generation sequencing* techniques produce much shorter reads (currently approximately 100-500 nt) when compared with the original Sanger technique which makes the assembly process more difficult. However, they produce a much larger number of these reads (hundreds of millions) in much shorter time (in less than a day). The higher coverage of these data sets is then exploited to reconstruct the DNA with appropriate accuracy. Such high-throughput techniques are generally cheaper and faster than the traditional shotgun sequencing approach. Several methods based on different biochemical reactions have been developed in the recent past and will be discussed in the following. All of these methods, however, are based on a spatial separation (compartmentalization) of DNA templates which enables massive parallelization of the sequencing reactions.

2.3.1 Compartmentalized DNA Amplification

One factor for the high throughput rates of modern sequencing approaches is massive parallelization. This is basically achieved by compartmentalization of DNA templates into micro compartments (wells, droplets in a water/oil emulsion, etc.). The sequencing reactions as well as the read-out take place in all these separated compartments in parallel. The basis for most parallelization approaches is the possibility to create small polymerase colonies (*polonies*).

Polonies. Polony-based sequencing technologies perform massive, parallel DNA amplification while keeping identical fragments spatially separated [SPR+05]. Various polony techniques including bridge PCR and emulsion PCR were developed.

In *bridge PCR*, small DNA fragments are amplified using primers that are covalently linked to a solid substrate (e.g., beads). The name is derived from the fact that the DNA elongation products actually form bridges between the bound primers. The primers are covalently bound to a solid substrate and thus immobilized. Consequently, the location of the corresponding amplified DNA templates is determined by the spatial arrangement of the bound primers [FRW+06, SJ08].

In *emulsion PCR (ePCR)*, the amplified DNA templates are compartmentalized in aqueous droplets in a water-in-oil emulsion. This compartmentalization has the nice side-effect that it reduces unwanted recombination events that lead to chimeric DNA [WPM+06][2]. The copied templates can be bound to magnetic beads in processes similar to the BEAMing method described by Dressman et al. [DYT+03]. In BEAMing, magnetic beads that are covalently coated with streptavidin are bound to biotinylated PCR primers. Beads and PCR reagents are mixed in a water-in-oil emulsion in such a proportion that an aqueous compartment contains at most one bead and DNA template on average. This microemulsion is then temperature-cycled like in any normal PCR resulting in a huge number of DNA copies that are bound to the respective beads. The beads can then be separated easily from the emulsion by using a magnet. Other polony techniques not discussed in detail here include *in situ* polonies and picotiter PCR.

2.3.2 Reversible Terminator Technologies

Reversible terminator technologies work very similar to the original Sanger method as they also rely on interrupted polymerase activity due to terminator nucleotide analogs. However, in this method, the nucleotide derivates are equipped with a *cleavable* chemical group. This group terminates the chain elongation process and contains the fluorescent label used for detecting the respective base incorporation. This group is removed chemically in a subsequent step and chain elongation by DNA polymerase continues [BKJ06, TRFT08] which is not possible in the Sanger method.

Illumina/Solexa Sequencing

The popular Illumina sequencing technologies[3] makes use of this reversible terminator technology. The sequence amplification is done by bridge PCR and the resulting amplified DNA is arranged immobilized in an array. During the sequencing step, fluorescently labeled reversible terminators for all four possible bases are used in parallel. These terminators compete for binding to the template. After the proper terminators are bound, their fluorescent labels are read out and the terminator groups are removed chemically. Then, the process begins again with a now elongated template [FRW+06, MBG+10]. This is the iterative step depicted in Figure 2.1.

2.3.3 Pyrosequencing

Another alternative to the classical Sanger method is pyrosequencing. Pyrosequencing differs from the Sanger method as it does not rely on a chain termination step but rather on the detection of pyrophosphate release events that occur when nucleotides are incorporated in the growing DNA fragment by polymerase. In this technique, the template DNA is effectively immobilized and the dNTPs (dATPαS, dGTP, dCTP, dTTP) are added and removed (washed out) sequentially. The incorporation of a nucleotide into the growing DNA chain (effectively a condensation reaction) releases pyrophosphate (PPi). This pyrophosphate reacts with a bioluminescent enzyme (e.g., luciferase) which emits a detectable electromagnetic radiation (light) [Ron01, KK10].

[2] Recombinant DNA created from multiple species is called "chimeric DNA".
[3] http://www.illumina.com/

Luciferase reaction. The light emitted at nucleotide incorporation time is produced by a simple biochemical pathway involving the released pyrophosphate (PPi), adenosine 5' phosphosulfate (APS), Oxygen and Luciferin as well as the enzymes ATP sulfurylase and luciferase (see pathway 2.3.1).

Pathway 2.3.1 (Pyrosequencing principle, slightly adapted reprint from [Ron01])

$$(DNA)_n + dNTP \xrightarrow{DNA polymerase} (DNA)_{n+1} + PPi \tag{2.1}$$

$$PPi + APS \xrightarrow{ATP sulfurylase} ATP + SO_4^{2-} \tag{2.2}$$

$$ATP + Luciferin + O_2 \xrightarrow[Mg2+]{Luciferase} AMP + PPi + Oxyluciferin + CO_2 + Light \tag{2.3}$$

The oxidation of luciferin effectively leads to the emission of detectable amounts of light. After each step the respective dNTPs have to be removed (washed) from the reaction compartments (e.g., microtiter wells). As incomplete removal of the nucleotides leads to false sequencing signals (in particular to phasing problems as described below), strategies for improving this step were developed. An enzymatic method is the addition of apyrase, a nucleotide-degrading enzyme from potato. Apyrase quickly degrades unincorporated dNTPs to the respective dNDPs. However, there is still enough time for the above described luciferase reaction to take place. Thus, shortly after a nucleotide incorporation event all unincorporated dNTPs are degraded and the next type of dNTP can be added.

Deoxy-adenosine-5'-(α-thio)-triphosphate (dATPαS) is used instead of the usual dATP as the latter is a substrate of luciferase and therefore leads to false signals. A further improvement of the method was introduced by the addition of single strand binding proteins (SSB) to the reaction system that effectively cover and stabilize the template DNA [Ron01].

454 Sequencing

The pyrosequencing technique was used by 454 Life Sciences[4] for parallelized sequence determination [KK10]. In 454 sequencing[5], a large plate containing millions of wells (reaction compartments) is used for sequencing. Each well contains exactly one single bead (due to size constraints) and each bead is associated with single stranded template DNA. This is done by fusing the template with a short sequence that is complementary to a short oligonucleotide that is bound to the bead. The DNA is covered by single strand binding proteins and the beads are then incubated with a mixture of DNA polymerase, ATP sulfurylase and luciferase. dNTPs are added in each sequencing iteration and are washed out/degraded by apyrase shortly after. Before this, however, matching dNTPs are incorporated in the growing nucleotide chain resulting in light production by luciferin. The light produced by each well is then recorded by a charge-coupled device (CCD) detector.

2.3.4 Sequencing by Ligation

The sequencing approaches discussed so far are all based on the extension of a DNA template and the accurate detection of nucleotide incorporation events, i.e., on the function of the DNA polymerase enzyme. Such sequencing approaches are therefore referred to as *sequencing by synthesis* [TRFT08]. In contrast to this, *sequencing by ligation* refers to sequencing approaches that are based on DNA ligase events. DNA ligase is an enzyme capable of building a covalent phosphodiester bridge between a free 5' phosphoryl group and a free 3' hydroxyl group of two nucleotides in an ATP dependent manner.

[4] 454 Life Sciences was later bought by Roche Diagnostics.
[5] http://my454.com/

SOLiD

The SOLiD sequencing platform[6] is a *sequencing by ligation* approach that makes use of ePCR for DNA compartmentalization and parallelization. First, a sequencing library is constructed and immobilized using rolling circle amplification[7]. Then, a sequencing primer is ligated to the DNA molecules from this library and a mixture of 8-mer[8] probes that carry four distinct fluorescent labels is added. These labels code for the two 3'-most nucleotides of a 8-mer and the various 8-mers then compete for ligation with the primer. The one that binds with the greatest affinity is ligated by DNA ligase and all others are washed out. Then the bound fluorescent label is read out and the bound octamer is enzymatically cleaved by a restriction enzyme, removing the trailing three bases and the label itself (i.e., a five-nucleotide long fragment remains attached to the template). This process determines what nucleotides are bound to the positions 1 and 2 of the DNA template[9]. Note, that the SOLiD method actually uses only four distinct labels for encoding the two 3'-most nucleotides. These labels may distinguish only between four different sets of dinucleotides. This means that the dinucleotides AA and TT, for example, would be encoded by the same label. This is, however, not a problem as later in the procedure the surrounding pairs are also read, as explained below, and a unique function for transforming the sequence of label signals into an actual DNA sequence exists. After this first step another round is started and the positions 6 and 7 are read (the prior positions are "blocked" by the mentioned 5-mer), etc. After several rounds, the strands are separated again and a new round with a new primer is started. This new primer is shifted by one nucleotide and thus allows different positions (2 and 3, 7 and 8, etc.) to be read by the method. This procedure is repeated until finally all positions are read out.

2.3.5 Single Molecule Sequencing

All above-mentioned HTS technologies are based on DNA polonies and thus on PCR-replicated ensembles of DNA templates. The large number of copied templates has benefits: the loss of single molecules or the wrong incorporation of a nucleotide in one of these copies is compensated by the others. However, on the downside, the template copying process itself introduces errors too that can be avoided when sequencing is done using a single DNA molecule as a template.

HeliScope

Helicos' Heliscope technology[10] is such a single molecule sequencing approach. It is based on iterative detection of the fluorescence signals emitted from single DNA templates that are elongated in a *sequencing by synthesis* approach. The difference is that by using a highly sensitive detection system, no clonal amplification of the sequence template is required. By this, errors introduced in library building or amplification are avoided. However, current error rates of this technique are still higher than for PCR-based systems, mainly due to weak fluorescence signals. Currently, HeliScope reads are only around 35 nucleotides long [MBG+10].

[6] http://www.appliedbiosystems.com/

[7] Rolling circle amplification (RCA) is a replication system found in bacteriophages, e.g., in the phage $\phi 29$. A polymerase (e.g., $\phi 29$ DNA polymerase) replicates a circular DNA template molecule with a short attached primer. This primer is displaced by the polymerase after the first replication "round" and replication continues. Finally, RCA results in a long concatemer of template copies. RCA is a simple and robust way to amplify DNA sequences without thermal cycling as required, e.g., for PCR [FX95, HSMS99, DNGL01].

[8] The original sequence by ligation approach was published in [SPR+05] using 9-mers instead of 8-mers.

[9] Note, that different designs of the 8-mers would report a different position in this first step, cf. [RK08] where the method is described with positions 4 and 5.

[10] http://www.helicosbio.com/

2.3.6 Other Emerging Approaches

Ion torrent

Ion torrent's[11] sequencing approach (aka *pH sequencing*) is quite different from the previously discussed ones. It is in principle based on the release of a hydrogen ion (H+) that occurs when DNA polymerase includes a nucleotide into the growing DNA sequence (cf. Pathway 2.3.2). In Ion torrent's technology, DNA templates are first compartmentalized into a large number of microtiter wells. A proper ratio between DNA molecules, polymerases and wells ensures that each well contains one or zero DNA templates and DNA polymerases on average. These wells are backed by a ion-sensitive layer and a proprietary H+ sensor which is basically a very small pH-meter. Whenever a nucleotide is included into the growing sequence, the pH in the respective wells decreases which is recognized by the H+ sensor. Now the four different bases are added one after the other in an iterative fashion and the respective base character is added to the called DNA sequences of the individual wells if their H+ sensor detected a base incorporation. Note that the pH-meter converts chemical into digital information and is thus the actual base-caller of this technology [Rus11].

Pathway 2.3.2 (Nucleotide incorporation)

$$(DNA)_n + dNTP \xrightarrow{polymerase} (DNA)_{n+1} + P_2O_7^{4-} + H^+ \tag{2.4}$$

SMRT

Pacific Bioscience[12] developed a single molecule real-time sequencing approach (SMRT) that is based on a chip with thousands of nanoscale pores that contain an immobilized DNA polymerase at their bottom. Labeled nucleotides are added that diffuse in and out of the pore very fast. A laser is used to excite the labels of the nucleotides when they are at the bottom of the pore (near the polymerase). Eventually, the proper nucleotide is incorporated into the growing sequence by the polymerase which retains this nucleotide for milliseconds in this particular position. The difference in how long the particular nucleotides reside in a particular region of the nanopore results in different signal intensities for the various nucleotides and in high signal-to-noise ratios. This technique uses special, proprietary nucleotides that are labeled at the phosphate chain rather than at the base to avoid polymerase stalling. The labels are actually removed by polymerase at incorporation time when the phosphodiester bond is created [EFG+09]. One particular interesting application of SMRT is that the raw fluorescence signals created by this technique allow it to reason upon the methylation states of the nucleotides in the used DNA template (as these result in different polymerase kinetics) which opens new possibilities in epigenome research [FWL+10, DHH10, ZJ10]

Nanopore Sequencing

The Oxford nanopore technology[13] is based on protein nanopores inserted into an artificial lipid bilayer. Molecules passing through a pore result in characteristic, detectable changes of an electric current that flows through these pores. These current fluctuations are detected by a patch-clamp amplifier.

Two currently developed sequencing approaches are based on this effect. In *exonuclease sequencing*, a DNA exonuclease is attached to the pore. This enzyme cleaves nucleotides from a given DNA sequence one at a time. The cleaved nucleotides then pass through the pore and their sequence can be detected from the changes in the current. In *strand sequencing*, DNA polymerase is attached to the pore. The template strand is pulled base-by-base through the nanopore by the polymerase and thus produces the sequence signal. Nanopore sequencing can also be used to identify epigenetic DNA modifications [CWJ+09, WSH+10].

[11] http://www.iontorrent.com/
[12] http://pacificbiosciences.com/
[13] http://www.nanoporetech.com/

Graphene-based Approaches

Recently, graphene based DNA sequencing techniques were proposed: In principle, DNA is sequentially passed through a voltage-biased tunnel gap inside a solid-state nanopore and the differing electronic properties of the nucleotides would affect an electric current flowing through that pore (quite similar to the Oxford nanopore technique). It was proposed to use graphene, a single-atom thick carbon material, for building electrodes and membranes for such a technology due to the special characteristics of this material (conducting properties, thickness, robustness, etc.) [Pos10].

2.4 Error Sources in Current Sequencing Methods

None of the described sequencing approaches is error-free, i.e., the resulting read sequences do not always correspond perfectly with the chemical reality. As the discussed HTS techniques show higher error rates than the classical Sanger method, this latter method still acts as a "gold standard" for sequencing experiments. The rate of errors is an important issue if the detection of rare variations between an analyzed genome and some reference is the goal of a sequencing experiment [DHH10].

One measure for errors done in sequencing is the average error rate, that is usually further split up into the error rate for substitutions and the rate for insertion or deletion (InDel) events. The various sequencing platforms vary not only in their error probabilities but also in the percentages of what kind of errors (substitution, insertion, deletion) occur. 454 sequencing shows, e.g., mostly InDel error while substitutions are rather rare. *Vice versa* does the Illumnia technology show more substitution errors than InDels errors in the produced reads [MBG+10].

In general, however, the error probability for a particular base in a read increases with its position in this read. In other words, the first bases of a read are more reliable than the later ones. Reasons for this are manifold (e.g., decreasing enzyme efficiency, phasing, etc.) and vary for the particular technologies [KK10]. In the following we discuss some general error sources resulting in imperfect reads.

2.4.1 Long Homopolymer Subsequences

One "hotspot" of errors in all *sequencing by synthesis* techniques are long homopolymer regions, i.e., genomic stretches of the same nucleotide. The main problem for methods based on DNA polymerase is the known polymerase slippage at such regions. This problem occurs also on short VNTR (variable number tandem repeat) regions [KK10]. A problem of the 454 technology, for example, is that multiple incorporations of the same nucleotide may occur per "round", resulting in artificially elongated homopolymer regions [DHH10, LMD+12].

2.4.2 Phasing – Loss of Synchronization

Phasing describes the problem that some individuals of a particular template ensemble get ahead or fall behind the other ones during the sequencing reactions and are thus "out of phase" with the others ("loss of synchronization").

This may happen, for example, when nucleotides are not completely washed out between two pyrosequencing iterations (cycles). When this occurs, some templates may already bind one of the "old" nucleotides before the next cycle begins and would thus be one base "ahead" of the others. It may, however, also occur that some templates do not incorporate a particular base and thus fall one base "behind" the others. In reversible terminator technologies, there is also the problem that bases with non-functional terminators are incorporated in some templates which results in the incorporation of another base in the same cycle for these templates [KK10].

The major problem of unsynchronized, out-of-phase strands is that phasing is a cumulative effect. This means that such out-of-sync polonies contribute increasing levels of noise to the overall signal over time. Phasing therefore contributes to the decreasing reliability of bases occurring "later" in a read. The errors stemming from loss of synchronization can be partially compensated computationally when the introduced error rates are known and reproducible. In [MEA+05], the authors report that they derived error rates by sequencing synthetically created test sequences and used these data to automatically correct the raw signals.

2.4.3 Base Calling

Base calling is the actual process that converts the raw signals of a sequencing instrument into strings of base characters (digital reads). Usually, this is done by special algorithms that analyze the raw read-outs (e.g., images taken with a CCD camera). In other terms, the central base calling step is the act of mapping a sequence of signals (e.g., fluorescence intensities in an electropherogram derived from an image) to a sequence of base characters. This mapping is imperfect and introduces further errors. A base-calling process usually calculates two values per sequenced base:

1. A IUPAC[14] base character (which may include the N character which stands for "any base").
2. A quality value that expresses how confident the algorithm is about the called base.

The most wide-spread base-calling algorithm is the Phred algorithm [EHWG98] that was employed extensively in the Human Genome Project. Phred reads an electropherogram (Figure 2.4) and calls the bases by analyzing its peaks. Additionally, it assigns a quality score (the "Phred-score") to each base which is logarithmically related to the probability p of a wrong base-call: $q = -10 \times \log_{10} p$. This means, for example, that a Phred score of 20 corresponds to a base-calling accuracy of 99% respectively to the probability that 1 out of 100 base calls at this position would be wrong[15]. In real data sets, Phred scores range up to a value of about 60 which stands for nearly 100% accurate base call. A wide-spread measure for a read sequence called with Phred, however, is its *Phred20* (aka *Q20*) score, which corresponds to the number of base calls with a Phred-score ≥ 20 [G+04].

Phred scores are calculated based on various signal features, such as peak shape and resolution. The extracted feature values are then used to retrieve corresponding error probabilities from technology-dependent lookup tables. These tables were created by analyzing traces of known DNA sequences and counting the base-calling errors. Although Phred is quite accurate at predicting error rates (see [Ric98]), other approaches for base-calling were developed over time. The base-calling problem was for example tackled using Hidden Markov Models (HMMs) and results were comparable with Phred [BEDE04]. In [LWA07], the authors report a significant performance improvement when compared with this and to the Phred algorithms. Yet another approach, ABI's KB base-caller, reports longer Q20 stable reads (i.e., reads where each base was called with a Phred-score ≥ 20) when compared with Phred [G+04]. A general problem for all base calling approaches is obviously the above mentioned SNR decrease. Consequently, all modern sequencing technologies include specialized software algorithms to increase the SNR based on technology-specific characteristics (e.g., to compensate for the estimated "crosstalk" between wells that incorporate/do not incorporate the currently deployed nucleotide in the 454 method).

2.4.4 Decreasing Signal to Noise Ratios

Several other technology dependent reasons lead to decay of the read-out signal. A loss in *signal fluorescence intensity*, for example, results in increased base-calling errors in the respective technologies. This is, e.g., the main source of errors in the Helios single molecule sequencing approach[16]. Further, as explained above, the

[14] http://www.iupac.org/

[15] $q = -10 \times \log_{10}(0.01) = 20$.

[16] Note that the fluorescence signal intensity is linearly dependent on the number of incorporated bases.

increasing loss of synchronization in polony-based sequencing methods leads to increasingly polluted read-out signals which makes the base-calling process more error prone. Signal decay and phasing problems are the main reasons for decreasing signal to noise ratios (SNR) in the created sequencing signals over time.

2.5 Summary

Although differing in their biochemistry, all described HTS approaches are similar in their workflows. Here, we briefly repeat the general steps as already reported analogously by Shendure and Ji in [SJ08]:

1. Sequencing library preparation by random fragmentation followed by *in vitro* ligation with common adaptor sequences.
2. Parallel fragment cloning via *in situ* polonies, ePCR or bridge PCR which results in a spatial separation of "amplicons".
3. Sequencing by altering cycles of enzyme-driven (polymerase, ligase) biochemistry.
4. Read-out (e.g., by imaging technologies or ion-sensitive layers) that acquires signals from a whole array of amplicons in parallel.

Current HTS technologies, however, still suffer from higher error rates when compared with Sanger sequencing. This means that the generated (short) reads contain certain amounts of bases that differ from the chemical reality which leads to subsequent issues, e.g., when reconstructing the whole nucleic acid sequence from them or when determining genomic variants. Quality values, associated with each read base by base-calling software, provide some measure for how "reliable" a base might be. These quality values can be used for the computational compensation of certain errors (e.g., phasing problems) while others (such as polymerase slippage) cannot be easily detected. It is one goal of bioinformatics research to compensate for such errors by exploiting, e.g., the large number of available reads produced by HTS technologies.

In a summary, HTS technologies have the great advantage of massive parallelism at reduced reaction volumes when compared with the classical Sanger technique. This results in faster sequencing at reduced costs but comes currently at the costs of shorter read length and less accurate base calls [SJ08, KK10, LMD+12]. It is thus common for HTS data that the determined reads differ from the actual nucleic acid sequence they originate from (due to sequencing errors) and from some reference sequence (such as the human genome) they are compared with (due to natural variation occurring in any individual[17]).

[17] It is, for example, estimated that the DNA sequences of two human individuals vary in 0.1% of the positions on average[Int05].

Chapter 3

High-throughput Sequencing Data

3.1 Representation of Raw Reads: FASTQ

There are several data formats that store read data (and its associated meta data) created by HTS technologies. The currently most wide-spread data format for such "raw" reads is arguably FASTQ [CFG+10], an extension of the well-known FASTA format (see Figure 3.1)[1]. FASTQ files are often huge files (tens of Gigabytes in size) storing sets of short digital reads and their associated per-base quality values.

```
@HWI-SN815_0060:5:1101:10000:102873/1
CCTTGTTAAAAACCAGATCACAAATCTGGGGCTCTTGGTCCCATTGGAGAAGGAAGGAAGAGCCTCAAAATAAGTGTGCACCCATGCACATATTCAGGAA
+
CCCFFFFFHHHHHJJJJJJJJJJJJJJJJJJJJJJJJJHHJJIJGIJJGJJJJHGIJIJJJJJJJHHHHHFFFFCBDEEEEEDDDDDDDDDDEEFEDDDD
@HWI-SN815_0060:5:1101:10000:114680/1
TCCTGTCACCGACGCTTCCTCAGGGCCCACACGTCACGCCACCCGTCGGCCTGTCACCGACGCTTCCTCAGTCCCCACAGGCCATGCCACCCCTCTTCCG
+
>;=;AA7@C<+00)27<?BB=B?#################################################################################
```

Figure 3.1: A FASTQ file storing two reads (one drawn in black color, the other one in blue color). Each read is represented by four lines: (i) a sequence identifier, (ii) the raw sequence characters, (iii) a plus character (and an optional sequence name), (iv) a string encoding the quality scores of each base character. Quality values are represented by ASCII characters whose character code was calculated by adding a certain offset to the respective quality value Phred score (cf. Section 2.4.3). This is done to map the quality values to the set of visible ASCII characters (which start at character code 32). Note that there exist multiple different FASTQ variants using different offsets (e.g., 33 or 64) [CFG+10, Ill11].

3.2 Mapping and Alignment

Assembly of raw reads in order to reconstruct the original nucleotide sequences is done by special software algorithms as briefly described in Section 2.1.1. In the following, we assume that a reference sequence for mapping the reads against is always available and do not discuss *de-novo* assembly any further.

Mapping algorithms try to find the position(s) on the reference sequence where a particular read "originated" from. Usually, such algorithms start by determining a number of candidate regions (e.g., by performing

[1] An alternative data format is, for example, Illumina's QSEQ format [Ill11]. Today, however, unmapped reads are increasingly stored also in SAM/BAM files as described below.

a *k-mer search*) and then align the reads to the respective subsequence of the reference sequence which leads to some alignment score. Here, some algorithms support gaps in these alignments (i.e., they support InDels in the reads) while others do not [SMZ+12]. The genomic position(s) of the highest scoring alignment(s) is/are then usually reported as a read's mapping position(s). The output of a mapping process is thus basically a set of genomic positions along with some additional meta data, such as a measure about how confident an algorithm is about a mapping position (a mapping quality value).

Mapping reads against a reference sequence constitutes another possible source of errors. The main reason for this is that alignments are usually not unambiguous, i.e., there are often multiple positions on the reference a particular read can be mapped to. For example, reads stemming from a homopolymer or from a repeated or copied subsequence may be mapped to several positions in the resulting sequence. Knowledge about what reads constitute paired ends (and should thus be mapped at a particular, known distance) helps in the disambiguation of possible mapping sites. Some mapping algorithms additionally exploit knowledge about certain characteristics of a respective sequencing technology the read data originates from. Particularly helpful in this regard are known average error rates and error types (substitutions vs. InDels) for each read position.

Mapping a very large number of short reads is a complex and time-consuming task and several specialized algorithms/platforms were and are constantly developed to solve this problem efficiently an accurately. Refer to [SMZ+12, LH10, MBG+10] for recent surveys of current algorithmic approaches and tools for sequence alignment.

3.3 Representation of Mapped Reads: SAM/BAM

The result of a mapping process is a set of mapped reads (i.e., sequences and quality values, their mapping positions and additional meta data) that is stored in a particular file format for further analysis. The current *de-facto* standard for storing mapped read alignments is the Sequence Alignment/Map format (SAM) [LHW+09][2]. SAM is a text file format where each entry (a SAM *record*) stores eleven mandatory fields (storing, e.g., a read's names, its quality-values, its mapping quality, etc.), see Figure 3.2.

```
HWI-ST815:40:81KKTABXX:4:2102:1398:146819
0   3   57913613    37   57M1I26M1D9M1I7M  *   0   0
TGTTATGAAACCACTGAGCTATTGGGAACAAGACTTAGAGACAACTATTTGCGTGGATTTTTTTTTTTTTAA
GGAAAAATACGTTGGAAAATAAAACTGT
CCCFFFFFHHHHDIIJIJJJIHHIJJIIJIIIGIIIJJIJEGHEHIGIJJHIDDFG=GEGCEEFDBDBD####
##############################
XT:A:U  NM:i:3  X0:i:1  X1:i:0  XM:i:4  XO:i:1  XG:i:1  MD:Z:83^T16
```

Figure 3.2: A SAM record representing a read that maps to position 57913613 on chromosome 3 of the respective reference sequence. The 11 mandatory SAM fields are drawn in different colors to increase the readability of the figure. The optional tags are drawn in grey color with italic letters, for details refer to [The11].

Each SAM record contains a CIGAR string (*Compact Idiosyncratic Gapped Alignment Report*, as known from pairwise sequence alignment) that describes its differences from the respective subsequence on the mapping reference sequence. The CIGAR operations supported by SAM include *match/mismatch*, *insert* and

[2]It has to be noted that several alternative formats, such as CALF [Gre08] or the Illumina Export format, were proposed for the representation of mapped read data. However, most bioinformatics tools concerned with mapped or unmapped reads nowadays support the SAM/BAM format.

delete operations. In Figure 3.2, the CIGAR string "57M1I26M1D9M1I7M" represents that the respective read mapped against the reference genome with two non-consecutive 1bp insertions and one 1bp deletion between them[3].

SAM CIGAR strings furthermore support two kinds of so-called *clipping* operations: soft- and hard-clipping. Clipping means that bases at the beginning or at the end of a read are removed, usually due to their low quality-values (which is, e.g., common for bases at the ends of reads as discussed in Chapter 2). This also excludes them from the actual alignment process where such "unreliable" stretches of base characters could introduce major errors. The actual base characters of *soft-clipped* bases are not really removed from the read sequences but are still contained in the SAM file and can thus be used by subsequent algorithms. A respective CIGAR entry describes which part of the read was used in the alignment and which parts were "clipped off". *Hard-clipped* bases, however, are neither included in the alignment process nor in the SAM file itself. The respective CIGAR entry tells subsequent algorithms basically that there were some bases in the raw read data that were clipped-off (e.g., before or during the alignment process). Their sequence, however, is not reproducible.

SAM additionally supports optional tags (see the lowermost line in Figure 3.2) that may store custom read-specific meta data. Such tags frequently store meta data calculated during the mapping process (such as the number of perfect hits of a read) or by other downstream operations (such as the per-base qualities before some recalibration process). Refer to the SAM specification for more details and CIGAR examples that also include padded and clipped bases [The11]. SAM is easily parsable and software libraries in various programming languages exist for reading/writing this format. One problem of this text-based format is, however, that the resulting file sizes are usually very large (tens of GBytes). For this reason, a binary, compressed version of SAM was developed. The binary SAM (BAM) format consists basically of concatenated BGZF (Blocked GNU Zip Format) blocks of SAM records. These blocks, each one basically a standard gzip archive, are limited to 64 kBytes and are indexed by a hierarchical index structure that allows faster random access to individual reads/sections in the alignment.

BAM files are, however, still quite large as discussed in the following because the used compression approach does not pay respect to special characteristics of the data but merely applies a general-purpose compression method (gzip) to them. This motivated the development of specialized compression algorithms for these data as discussed in the following and in the main Chapter 4 of this thesis.

3.4 Compression of HTS Data

The enormous size of the data associated with HTS experiments motivates the development of specialized compression algorithms for these data. These algorithms can broadly be classified into three groups:

1. Algorithms that compress any biological sequence data (e.g., standalone DNA sequences), such as GenCompress [CKL00], DNACompress [CLMT02], DNAPack [BLF05] or XM [CDAM07], see [GSU09] for a comprehensive review.

2. Algorithms specialized for compressing unmapped HTS data (mainly stored in FASTQ format), such as SOLiDzipper [JPAH11], G-SQZ [TLS10], DSRC [DG11] or the algorithms proposed in [BBN+11, WAA11].

3. Algorithms for compressing whole sets of mapped HTS sequence data as described in [DRC+10, HYFLCB11, KSK+11, STZH11].

This thesis is concerned with the latter group and in the following we briefly highlight two current trends in mapped HTS data compression: reference-based compression of the actual sequence data and value transformation for the compression of per-base quality values.

[3] *57M* means that there were 57 bases that matched or mismatched the reference, *1I* means a 1bp insertion, *26M* means again 26 matches/mismatches, *1D* means a 1bp deletion, etc.

3.4.1 Reference-based Compression

The idea of reference-based compression is to exploit that reads usually differ only slightly from the respective subsequence on the reference sequence they were mapped to. In such a case it is more efficient to store the (few) differences between read and reference than to store the actual read sequence itself and several tools have exploited this [BWB09, CLLX09, DRC+10, WZ11, HYFLCB11]. Drawbacks of reference-based compression are that the reference sequence has to be available at decompression time (however, this is usually the case) and that it is not applicable to unmapped reads (cf. [BBN+11]).

3.4.2 Quality-value Transformations

While the actual read sequences can be effectively compressed with the mentioned approaches, this is much harder for the per-base quality values (q-values) that accompany them. A main reason for this is that q-values have a much wider range of possible values when compared with sequence data. There are, e.g., 94 possible q-values in Sanger format (0-93, see [CFG+10]) while sequence data is usually composed from only five IUPAC characters (A,C,T,G,N). Another problem is that q-values show quasi-random distributions in real-world data and the resulting high entropies of these data makes them hard to compress. The most popular strategy to overcome these issues is to transform q-values (either losslessly or lossy) in order to improve subsequent compression with a general-purpose compression algorithm, such as gzip[4], lzma[5] or bzip2[6].

Lossless transformations. A lossless q-value transformation strategy is, for example, to store not the actual q-values themselves but rather their difference to the respective preceding q-value [KSK+11]. While this does not reduce the range of possible values (in fact, it even increases it by one), it often leads to longer consecutive stretches of close or even equal values which is subsequently exploited by the mentioned compression algorithms. Several alternative transformation methods were proposed, e.g., in [WAA11].

Lossy transformations. A lossy value transformation strategy is to really reduce the range of possible q-values by quantization. Quantization reduces the number of possible q-values by mapping subsets of values to one single value. Again, this is exploited by subsequent data compression algorithms [HYFLCB11, KSK+11]. Quantization is an irreversible step which makes this strategy lossy.

In the following main part of this thesis, we present a novel compression approach for mapped HTS data sets that makes use of both of these general strategies.

[4] http://www.gzip.org/
[5] http://www.7-zip.org/sdk.html
[6] http://bzip.org/

Chapter 4

NGC: Lossless and Lossy Compression of Aligned High-throughput Sequencing Data

In this chapter we present NGC, a tool for the compression of mapped short read data stored in the widespread SAM/BAM format. NGC enables lossless and lossy compression and introduces two novel ideas: First, we present a way to reduce the number of required code words by exploiting common features of reads mapped to the same genomic positions; second, we present a highly configurable way for the quantization of per-base quality values which takes their influence on downstream analyses into account. NGC, evaluated with several real-world data sets, saves 33-66% of disc space using lossless and up to 98% disc space using lossy compression. By applying two popular variant and genotype prediction tools to the decompressed data, we could show that the lossy compression modes preserve over 99% of all called variants while outperforming comparable methods.

Note that this chapter is a slightly extended version of [PvH12].

4.1 Introduction

Current high-throughput sequencing (HTS) technologies enable the fast, accurate and affordable sequencing of long stretches of DNA, which adds genome scientific approaches, such as RNA-sequencing (RNA-seq) or whole genome sequencing (WGS) to the set of standard laboratory methods. These technologies result in huge amounts of digital data that have to be processed, transferred, stored and archived, which includes "raw" sequencing data and an even larger number of intermediate and final result files that are produced by pipelines of data analysis and manipulation tools. Such files store HTS data in different (pre-) processing states and associated metadata describing these data, such as read names or mapping quality values, using various file formats, e.g., unmapped reads stored in FASTQ format, mapped reads stored in SAM/BAM or called variations stored in the VCF format.

In general, all such files are subject to differing data handling and storage policies that define, e.g., where and how long these files are stored, how fast they have to be accessible or how secure this access has to be. The emerging field of personal genomic sequencing, for example, will result in large amounts of data with high security demands, but not necessarily fast access times. Note that such policies are influenced not only by practical considerations (such as available storage space) but also, e.g., by legal constraints and privacy issues. It is the costs associated with processing, storage and transmission of these data, rather than the generation of sequencing data itself, that constitute a major challenge to HTS experiments today (cf.

[PPG11, Kah11, KSLI12]). This motivates the development of data compression algorithms specialized for the discussed file formats, and recent research in this field may be divided into three major, but overlapping, categories: (i) compression of genomic sequences as generally produced by re-sequencing experiments [CDAM07, WZ11, PPG11], (ii) compression of unmapped short reads [TLS10, DG11, BBN+11] and (iii) compression of aligned read data [DRC+10, HYFLCB11, KSK+11, STZH11]. This work falls in the latter category, namely by compression of aligned short reads stored in the popular SAM text file format [LHW+09]. This data format stores not only short sequences of DNA characters (read bases) but also a lot of associated metadata such as per-base quality values (q-values), read names or mapping positions. Along with this easily processable text format goes a compressed binary variant (BAM) that basically comprises a blocked, gzipped version of SAM. Our tool, NGC, allows a more efficient compression of the data stored in SAM/BAM files by handling each contained data stream individually, using value transformations and compression algorithms that pay attention to the respective value distributions. An overview of our solution is depicted in Figure 4.1. In this chapter, we mainly discuss the used compression approaches for two of these data streams, namely read bases and q-values and briefly sketch our strategies for encoding read names and alignment positions. Our proposed method for the (lossless) compression of read bases builds on the wide-spread idea to store such data relative to some reference sequence [BWB09, CLLX09, DRC+10, WZ11, HYFLCB11]. However, we propose to traverse the bases in an alignment of reads in a per-column way that exploits common features of multiple mapped reads rather than handling each read individually as done in previous research. This leads ultimately to a reduction of required code words and, in consequence, to a more efficient data compression. We measured the achieved compression rates (the required bits per sample) and overall compression ratios (the ratio between compressed and uncompressed size. The smaller this ratio, the better) and compared them with related tools. Regarding the compression of q-values, we contribute a detailed discussion of several possibilities for their lossy compression and analyse the impact of the associated information-loss on subsequent data analysis pipelines. We propose a novel way for lossy q-value compression that distinguishes between different categories of q-values and is able to preserve the original qualities of bases in selected columns, that are the main targets of variant-calling and genotype prediction algorithms. We have evaluated our lossy per-base quality value compression using variant calling pipelines composed of state-of-the-art analysis tools and found that our proposed methods may preserve 99-100% of all called variants on average while outperforming comparable methods. Our evaluation included *Homo sapiens*, *Mus musculus*, *Escherichia coli* and *Arabidopsis thaliana* data from exome, whole genome, ChIP and RNA sequencing experiments.

4.2 Materials and Methods

4.2.1 Datasets and software availability

The used evaluation datasets are deposited in the Sequence Read Archive [KSLI12] under study numbers/run accession numbers: ChIP-Seq (mouse): SRX014899 / SRR032209, Reseq/hm (human): SRX000376 / SRR001471, RNA-Seq (E. coli): ERX007969 / ERR019653, Reseq (E. coli, paired end): ERX008638 / ERR022075, Reseq (E. coli): SRX118029 / SRR402891, Reseq (A. thaliana): SRX011868 / SRR029316. The human exome-sequencing data set was kindly given to us by B. Streubel. The Reseq/chr20 (human) dataset is a resequencing data set of human chromosome 20, available from the GATK resource bundle (see Appendix A). The data sets were mapped using BWA v0.6.1 [LD10] with standard parameters for single respectively paired end data. Unmapped reads were pruned from the data sets; variants were called with GATK v1.4 [MHB+10] and SAMTOOLS v0.1.18 [LHW+09] using the parameter settings given in Appendix A. NGC was implemented in Java 1.7 and is available for non-commercial use at
`http://purl.org/lsdv/ngc`.

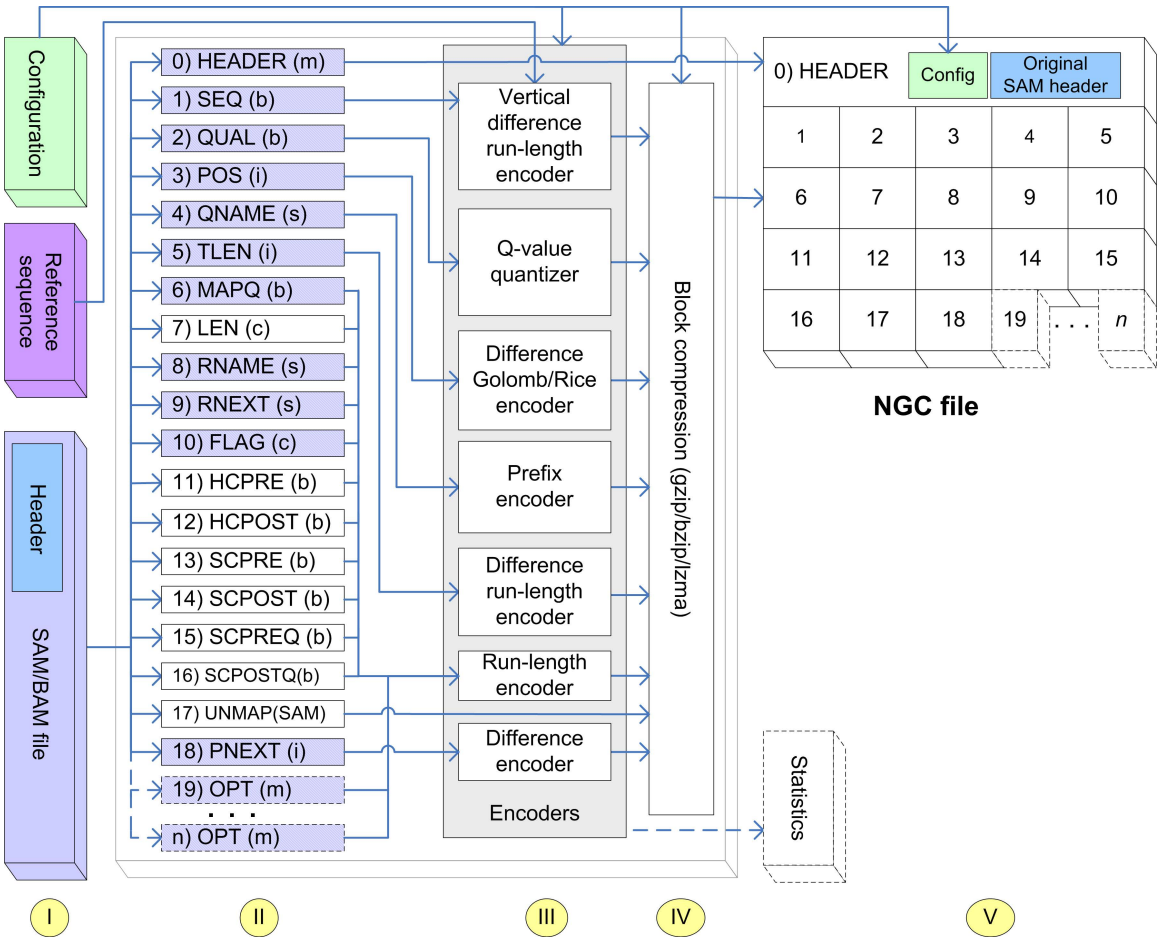


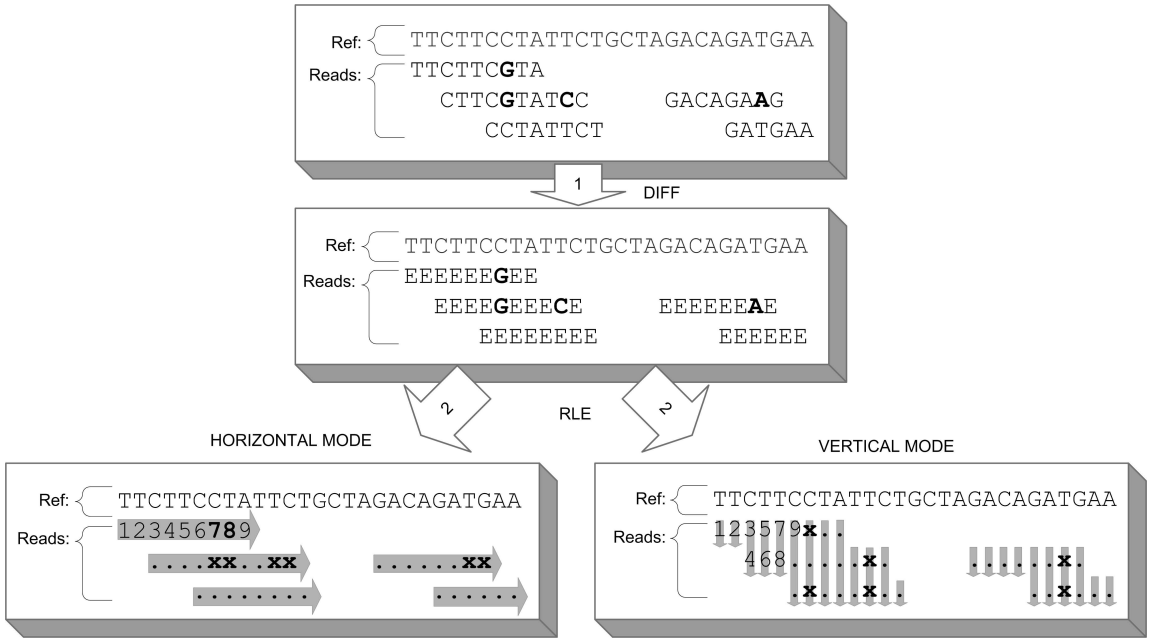
Figure 4.1: Schematic overview of the NGC compression approach. NGC takes a SAM/BAM file, a reference sequence and a set of configuration parameters as input (step I) and generates an NGC file and an optional statistics file (optional components drawn with dashed lines). First, NGC de-multiplexes the various data streams and adds some additional streams (step II). These streams are then passed to our various encoders that transform their values and prepare them for the subsequent block-compression (step III). In this step IV, the data is compressed using a general-purpose compression algorithm (gzip, bzip2 or lzma). Finally, the compressed data blocks, the original SAM file header and the required configuration information are combined to one single output file (step V). Hatched streams (step II) encode the information described in [The11] although with very different encoding schemas. Unhatched streams encode: 7: read lengths, 11-16: data required for reconstructing clipped bases/q-values, 17: unmapped reads. The basic data types of the streams are written in parentheses: (s)tring, (i)nteger, (c)har, (b)yte, (m)ixed or SAM format.

4.2.2 NGC

We have developed a tool (NGC) that enables the complete lossless and lossy compression of alignment data stored in SAM/BAM files. The NGC compressor takes an alignment file, a reference sequence and several configuration parameters as input and outputs a compressed file (Figure 4.1). On the other hand, the NGC decompressor reverses this operation. When the lossless mode of NGC is used, the resulting file is semantically equal to the original file in the sense that it contains the exact same information, although it might not be byte-equal as (i) the order of optional SAM fields is not preserved which results in different byte streams in the resulting BAM file and (ii) the SAM fields MD ("mismatching positions") and NM ("number of mismatches") are dropped at compression time and automatically recalculated at decompression time which may result in slightly different values because of ambiguities in the SAM specification. When one of the multiple lossy modes is chosen, the quality values in the resulting file additionally differ (partly) from the original ones. NGC treats each read in the original file as an n-tuple of values of differing data type. Read names, for example, are of type string, alignment coordinates are of type integer, mapping quality values are of type byte, etc. Such an n-tuple consists in principle of the 11 mandatory and all optional SAM fields (cf. [The11]), however, for algorithmic reasons, there are some deviations, for example, we do not store the CIGAR information, but reconstruct it from the other data and store some additional metadata such as read lengths (cf. Figure 4.1). NGC treats each element in these n-tuples individually, that is, each data field may undergo independent steps of value transformations and compression. Individual fields may even be dropped during the compression phase (e.g., NGC enables users to prune the read names from the data to save space. New read names will then be auto-generated at decompression). Figure 4.1 gives an overall picture of our software and shows the default encoding of the different data streams. In the following, we describe the four streams that take up most of the space in compressed data files in more detail: read base sequence information, per-base quality values, read positions and read names.

4.2.3 Read base compression

We call our idea for the compression of read bases "vertical difference run-length encoding" (VDRLE). Generally, run-length encoding (RLE) is a simple encoding scheme for sequences of characters from some alphabet A (in our case the IUPAC nucleotide single-letter codes). A run-length (RL) is basically a pair of a single character from this alphabet and a positive integer number indicating the RL of this character in the sequence, an RLE is then an ordered sequence of such RLs. For example, a DNA sequence S="AACTTT" is encoded by the RLE $\{(A,2),(C,1),(T,3)\}$. The general compression idea of RLE is that long stretches of identical characters can be represented by one single RL that requires storing only the two RL symbols. Obviously, this simple encoding strategy is not very effective for DNA sequences, as most run lengths would be very short, and a large number of such pairs would be needed to encode the sequence. The situation changes, however, when the encoded sequence contains long stretches of identical characters, as in this case, much less RL pairs than characters would be required. Our proposed compression algorithm for read bases is based on reducing the number of required RLs that may then be effectively encoded using well-known coding schemas respectively compression algorithms. The first step of VDRLE is similar to other reference-based compression approaches: we do not encode a read sequence S itself but rather its differences to some reference sequence R. For this, let us formally introduce a new character "E" to a now extended alphabet $A_e = A + \{\text{"}E\text{"}\}$ that represents that a character in S is unchanged in comparison to R. Now, we calculate a sequence "diff" using the function $\Delta : A^n \times A^m \rightarrow A_e^n$ that replaces all characters in the first sequence that map to the same characters in the second sequence with this special character. Thus, $S' = \Delta(S,R)$ is constructed by replacing all characters in S that map to equal characters in the reference sequence R with "E" characters (cf. Figure 4.2, step 1). It is easy to see that the more similar S and R, the more and the longer the expected runs of "E" characters in S'. Our improvement to existing RLE-based approaches is to encode such diff-ed read bases in an alignment "vertically" (i.e., position after position or "columnwise", cf. [GSU09]) instead of "horizontally" (i.e., read after read). This means that the stream of base characters that is encoded is not



(a) Vertical and horizontal difference run-length encoding. The figure shows the difference for a very simple alignment (top of figure). In step 1, all bases that match the reference are replaced by the "E" character. In step 2, the resulting read bases are run-length encoded. The left box shows a "horizontal" way to do this: the bases of each read (sorted by alignment position) are enumerated (we show the indices of the first nine bases in the figure) and a new RL is started whenever a base differing from the previous one is encountered (we have marked such positions boldface). The lower-right box shows our vertical approach: the read bases are enumerated column-after-column. Again, the start positions of RLs are marked boldface. Vertical encoding saves two RLs in this toy-example.

	Reseq/chr20 *(human)*	Reseq/hm *(human)*	Reseq *(A. thaliana)*	Reseq *(E. coli)*	Reseq *(E. coli, PE)*	Exome-seq *(human)*	RNA-seq *(E. coli)*	ChiP-seq *(mouse)*
Horiz. DRLE counts	43,532,619	900,546	13,341,776	4,303,863	191,384,090	3,386,907	56,295,485	8,127,608
Vert. DRLE counts	28,855,452	824,683	8,912,294	3,587,923	48,162,138	2,123,039	1,348,494	6,364,152
Ratio [%]	0.66	0.92	0.67	0.83	0.25	0.63	0.02	0.78

(b) This table shows the counts for horizontal and vertical difference run-length encoding (DRLE) in our test data sets. Figure 4.3 shows an excerpt of one of these data sets.

Figure 4.2: Vertical difference run-length encoding results in less required run-lengths.

simply a concatenation of read bases but first lists all base characters that are mapped to position 1, then the ones mapped to position 2, etc. (similar to the SAMTOOLS mpileup command). Figure 4.2 summarizes our approach. The figure shows how $\Delta(S, R)$ is calculated in the first step by replacing all bases in the individual reads that match the respective reference bases. Our algorithm then generates a stream of base characters by traversing such a diff-ed alignment columnwise and encodes this stream by RLE. A new RL is generated whenever a different base than the last encoded one is encountered. For actually storing these RLs, we simply map all characters in our alphabet to byte values and write them, followed by one byte representing the RL length, to a standard byte stream. This stream is finally compressed using a general-purpose compression algorithm (users may choose between bzip2, gzip, lzma and no block compression). Note that the described VDRLE method exploits common features between reads mapped to the same genomic region to decrease the number of required code words to be encoded. It exploits the fact that short-read mapping software tries to minimize differences between reference sequence and mapped reads which often results in the same bases

differing from the reference sequence being stacked above each other rather than next to each other, regardless whether these bases are real variants from the reference sequence or just sequencing artefacts.

4.2.4 Quality value compression

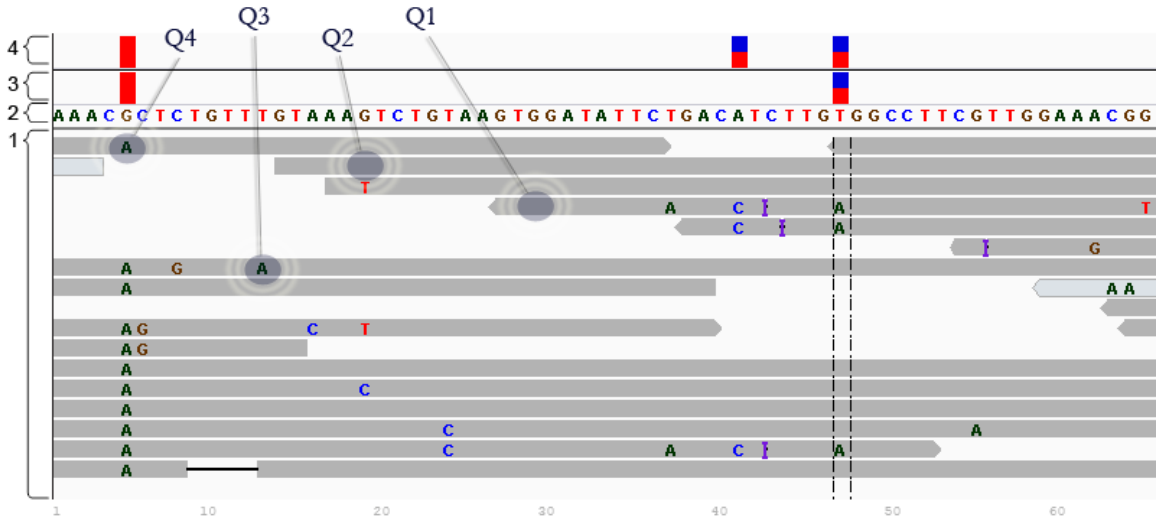
As others have reported before, we also found per-base quality values (q-values) very hard to compress because of their quasi-random distributions, respectively the high entropy of these data that results in rather high (undesirable) data compression ratios [DG11, KSK+11, STZH11, WAA11]. Although we have tried several approaches to improve lossless q-value compression, including an adaptive arithmetic coding approach [WNC87] that builds a statistical model per read position, we did not reach significantly better compression rates over all data sets in comparison with the general-purpose BZIP2 algorithm (data not shown). We, therefore, currently do not apply any value transformation (not even diff-encoding or GapTranslating as advocated in [WAA11]) in our lossless mode, but we encode the raw data using bzip2 with parameter settings that result in its best compression (i.e., using the "-9" switch). With our data sets, bzip2 achieved compression rates of 1.8-3.7 bits/q-value which compares well with other reported numbers (cf. [KSK+11, WAA11]).

This rather bad compressibility of q-values is, however, the current main challenge for our compression method which becomes clear when considering their large fractions (82-97%) of the overall file sizes of losslessly compressed data sets as shown in Table 4.1. It was, however, proposed by several authors in the recent past that it might be feasible to store q-values in a so-called lossy manner, i.e., to discard some information in favour of better compression ratios [KSK+11, HYFLCB11, WAA11].

4.2.5 Lossy q-value compression

A central step in most lossy data compression methods is quantization. Quantization of q-values maps the set of possible Phred quality values (e.g., 0-93 in Sanger format [CFG+10]) to a smaller set of values, usually by binning. Although this does not reduce the number of code words that have to be encoded, it greatly enhances the effect of subsequent probabilistic or dictionary-based compression methods as there are much smaller ranges of possible code words. However, this loss of information also affects the results of downstream applications that make use of q-values, such as software for the removal of polymerase chain reaction (PCR) duplicates, for variant calling or for genotype prediction. It is a common goal of lossy compression approaches to find a good trade-off in this regard. There are many possible q-value quantization schemas that differ in complexity and in their influence on compression ratios and downstream effects. NGC uses a simple binning strategy that maps all q-values that lie within an interval to some single value within this interval (e.g., its upper or lower border). Such possibly non-uniform quantization intervals should be disjoint and should cover the whole range of input values. Note that extreme cases of this approach are to use a single interval that spreads the whole input value range and thus assigns a single value to all q-values or to use as many intervals as there are possible q-values which basically results in no quantization at all. Our quantization method does not treat all q-values in an alignment equally. It rather distinguishes between q-values of bases that (i) match or mismatch the reference sequences and (ii) reside in single- or multiallelic alignment columns (i.e., columns that contain one respectively multiple different base characters in the read sequences). The resulting four possible q-value categories (annotated in Figure 4.3) are as follows:

- Q1: q-values of bases that match the reference and occur in columns where all bases match.
- Q2: q-values of bases that match the reference and occur in multiallelic columns.
- Q3: q-values of bases that mismatch the reference and occur in multiallelic columns.
- Q4: q-values of bases that mismatch the reference and occur in columns where only this allele occurs.



(a) Excerpt of an alignment to illustrate our VDRLE approach for read base compression. The figure is a modified screenshot of region chr1:121484997-121485088 in the Reseq/hm (human) data set made with IGV 2.0.17 [TRM12]. It shows (from bottom to top): the reads with only the bases that differ the mapping reference shown, INDELs are drawn as I-characters or horizontal lines respectively [1], the mapping reference [2], variants called by SAMTOOLS [3] and GATK [4] respectively. Our read sequence compression creates RLs by traversing this alignment columnwise (see text). The present alignment could be encoded with 75 horizontal but only 55 vertical RLs. Note that in NGC, however, the reads would be sorted strictly by coordinates and thus the results would be slightly different. The marked bases/q-values are: Q1 - bases that match the reference in columns that contain no alternate allele; Q2/Q3 - bases that match/mismatch the reference in multiallelic columns; Q4 - bases that mismatch the reference in columns that contain only one kind of alternate allele.

	Reseq/chr20 *(human)*	Reseq/hm *(human)*	Reseq *(A. thaliana)*	Reseq *(E. coli)*	Reseq *(E. coli, PE)*	Exome-seq *(human)*	RNA-seq *(E. coli)*	ChiP-seq *(mouse)*
Q1 counts	3,873,164,564	61,831,496	303,452,695	491,077,181	42,556,735	256,925,783	31,453,173	472,315,173
Q2 counts	807,325,823	119,513	109,224,800	204,412,364	**4,388,981,896**	65,240,735	**188,284,057**	21,246,748
Q3 counts	22,271,926	155,855	5,702,544	2,158,208	102,563,275	1,641,199	29,657,344	2,348,243
Q4 counts	1,582,569	324,201	1,089,650	6,578	294	113,933	3,625	1,769,643

(b) This table lists the counts of the respective q-values in our evaluation data sets. It is notable that in two data sets the number of Q2 values exceeds the number of Q1 values significantly (printed boldface) which is ascribable to the respective experimental settings.

Figure 4.3: Q-value categories and their distributions.

Our general idea is that q-values of distinct categories have also differing impacts on downstream analyses. For this reason, our approach allows the application of different quantization schemas to q-values of differing categories. We propose, for example, to use more fine-grained sets of quantization intervals for q-values of substituted bases (Q3, Q4) rather than for the ones that match the mapping reference as the latter ones have much less impact on the results of downstream applications as described below. Our tool further provides the possibility to "lock" an arbitrary number of columns in the alignment, so that the original q-values will be retained in these positions. This is useful, e.g., in re-sequencing experiments where users already know the positions of some (expected) variations beforehand. The counts of the four q-value categories in our evaluation data sets are shown in the table in Figure 4.3. These numbers reveal that Q1 and Q2 values of the categories are naturally much more frequent than Q3 and Q4 values, which makes them the primary targets of our quantization strategies. NGC supports two modes for storing quantized q-values. The first mode considers q-values in a horizontal way and stores them to a simple byte stream. The second mode traverses q-values vertically and stores them run-length encoded (comparable with the described VDRLE approach, however, without the "diff-ing" step). This latter mode is more efficient when the majority of q-

values are quantized, which leads to long stretches of equal q-values. Note that the resulting byte streams are subsequently compressed by some general-purpose block-compression method in both cases. The final compression ratio of our lossy compression approach consequently depends on the present q-value distribution, the chosen interval-sets for the four q-value categories, the partitioning of q-values into these categories, an optional list of "locked" alignment columns and the chosen block compression mode.

4.2.6 Evaluation

To find reasonable trade-offs between quantization strategies, resulting compression ratios and possible downstream effects, we conducted a comprehensive experiment. In this experiment, we compressed the data sets listed in Table 4.1 using different combinations of q-value quantization intervals. Then we decompressed the data and used a pipeline of state-of-the-art tools to call variants (SNPs, INDELS, etc.) and their predicted genotypes in the obtained BAM files. This pipeline contained steps for INDEL realignment, PCR duplicate removal and various variant filtering steps (for details, see Appendix A). We further used two wide-spread variant calling tools, namely GATK [MHB+10] and SAMTOOLS [LHW+09] for variant and genotype prediction. The resulting variant sets were then compared with the ones found in the respective unquantized data sets that served as our "gold standard". We counted the number of recovered (true positive), lost (false negative) and additional (false positive) variants as well as the number of variants that differed in their predicted genotype. We further measured the compression ratio of the particular approach which is defined as the ratio between compressed and uncompressed file size. Additionally, we compared our tool with cram[1] and goby[2], two tools that also compress SAM/BAM files. Goby (we used its latest version v2.1) supports only lossless compression. Cram is based on the method described in [HYFLCB11] but was further developed in the meantime, and we used its latest available version (v0.85) in our evaluation, treating it as an alternative lossy compression method. Cram enables users to preserve q-values of various categories selectively. It allows, for example, preserving only q-values of substituted bases or of insertion regions. This leads to a number of quantization strategies that were evaluated by us as summarized in Table 4.2. We basically configured our tool to either quantize q-values by simply assigning a single value to them (maximum compression) or to use a simple "standard" binning scheme:

Definition 4.1 (Standard quantization scheme)

$$q' = f_{quant}(q) = \begin{cases} q, & q < 2 \\ 2, & q < 10 \\ 15, & q < 30 \\ 30, & otherwise \end{cases}$$

This schema was derived by studying the descriptions of various SNP calling software that incorporate per-base quality values in their calculations. Note, however, that multiple other schemas might be useful and that an extensive study of this regard is beyond the scope of this work. For a direct comparison, we configured a mode (m4, see Table 4.2) that quantizes q-values as similar as possible to a lossy configuration of cram (parameters given in Appendix A.7), called *cram-lossy* in the remainder of this thesis. We further configured a mode "recovery" where NGC was provided the VCF file obtained by calling variants with GATK in the unquantized data sets. NGC then "locks" the respective columns and preserves its original q-values, and we were curious whether this could significantly improve variant recovery. We additionally configured one mode (m5) that maximized compression by using the most restrictive quantization scheme and additionally dropping all q-values of reads with a low mapping quality.

[1] http://www.ebi.ac.uk/ena/about/cram_toolkit
[2] http://campagnelab.org/software/goby/

4.2.7 Other streams

For read positions (stream POS, Figure 4.1), we store the differences to the previous position Golomb/Rice-encoded as also proposed by other authors (cf. [DRC+10]). We estimate the Golomb-parameter by first calculating the mean difference between two neighbouring read alignment positions by sampling and then applying the method discussed in [Kie04]. It is noticeable that Golomb-codes may result in very long code words when encoding numbers that are much larger than this parameter because of the unary coding scheme used for encoding the quotient. Such large "gaps" between reads occur, for example, between adjacent covered regions in exome sequencing data. Long unary prefixes are, however, effectively compressed by the subsequent block compression methods (e.g., bzip2). Read names (stream QNAME,Figure 4.1) are usually systematic names that contain information, such as instrument and flowcell identifiers, run numbers or x/y coordinates. For these names, we have developed a simple encoder that looks for the longest common, stable prefix of the last n reads and encodes this prefix only once and the variable rest for each read. Our simple method requires approximately 3-5 bytes per read name (see Appendix A), however, this could be further optimized (cf. [STZH11] for an alternative approach). We believe, however, the information stored in these strings is rarely used after the mapping phase, and consequently our tool enables users to drop them. Decompressed files will then contain automatically generated read names. When not dropped, read names take up 8-19% of the file size in the losslessly compressed evaluation data sets that increases up to 30-85% when using lossy compression.

4.3 Results

To verify that it results in less RLs when traversing an alignment "vertically" rather than "horizontally", we have counted the RLs in our evaluation data (see Figure 4.2). We found considerably less code words when compared with a "horizontal" approach (up to 50X less). Further, we compressed all data sets with NGC and calculated how many bits per encoded read base were used by its VDRLE method. The results varied between 0.12 and 0.03 bits/base (see Appendix A), and we believe that there is some potential left when optimizing the RLE encoder. As the read stream accounts for only about 1-2% of our total compressed file size; however, we focused on a different stream that accounts for about 35-71% in the original BAM files and for about 82-97% in our losslessly compressed files: the per-base quality values.

4.3.1 Data compression, decompression and variant recovery

We evaluated how well the different q-value quantization modes listed in Table 4.2 perform at preserving the variants called in the nonquantized datasets with our pipelines. For this, we compressed the data using the respective quantization schemas, decompressed them and called their variants in comparison with their mapping reference sequence. The resulting variant sets were then compared with the ones called in the respective unquantized data sets. The results of our evaluation are summarized in Figure 4.4 that contrasts the achieved compression ratios with the precision and the sensitivity of the chosen quantization strategy. It can be seen that our tool provides lossless compression ratios between approx. 0.3-0.6 which corresponds to space savings of 40-70%. We compared these measures with the lossless compression ratios of cram and goby, absolute numbers and a comparison of the compression ratios for lossless compression are given in Table 4.1.

The compression ratios of the lossy modes depend mainly on the used quantization schemas for Q1-Q4 values. A comparison of m4 and cram-lossy reveals that we provide a more efficient lossy compression than the cram tool (Table 4.1) when we configure both tools to basically quantize the same q-values (although it has to be noted that cram-lossy preserves qualities of Q2 bases if there are two or more reads that differ from the reference at a particular column which is a slightly different behaviour than in NGC's m4 mode where all Q2 q-values are quantized). The precision and sensitivity analysis also reveals to what extent

q-value quantization has to be paid in terms of false positives and negative calls. Precision (aka positive predictive value) measures the ratio between true positives and positive calls, sensitivity (aka recall rate or true positive rate) measures the ratio between true positives and the sum of true positive and false negatives (see Appendix A). One extreme is the "drop all" mode that simply replaces all q-values by a fixed value (data given in Appendix A). This mode primarily leads not only to a large number of false positive variant calls but also to much higher false positive rates than all other modes and consequently to low precision/sensitivity measures. Our proposed mode m1 (quantizing only Q1 values) results in high and stable (over all data sets) precision and sensitivity rates while considerably lowering the achieved compression ratios in comparison with lossless compression. The outliners for the m1 compression ratios can be explained when considering that the number of Q2 bases exceeds the number of Q1 bases for the RNA-seq *(E. coli)* and the Reseq *(E. coli, PE)* data sets (cf. Table 4.1). This means that there are few columns that completely match the reference sequence, which is unfavourable for RLE that was used for this mode.

		Reseq/chr20 *(human)*	Reseq/hm *(human)*	Reseq *(A. thaliana)*	Reseq *(E. coli)*	Reseq *(E. coli, PE)*	Exome-seq *(human)*	RNA-seq *(E. coli)*	ChiP-seq *(mouse)*
Mapped reads		50,663,069	487,201	11,651,942	19,379,287	44,938,891	3,239,217	6,927,728	13,824,441
BAM size [MB]		5,868	53	504	655	2,945	199	177	637
Base/q-value count		5.21×10^9	6.24×10^7	4.19×10^8	6.98×10^8	4.54×10^9	3.24×10^8	2.49×10^8	4.98×10^8
Avg. read length		101	128	36	36	101	100	36	36
Coverage		81.19	0.02	3.51	150.37	978.26	0.10	53.75	0.18
q-value % of total file size (BAM)		0.41	0.35	0.49	0.64	0.66	0.64	0.71	0.44
q-value % of total file size (NGC lossless)		0.53	0.82	0.88	0.97	0.89	0.95	0.97	0.86
Lossless compression ratios	NGC	0.60	0.32	0.44	0.50	0.57	0.52	0.55	0.40
	Cram	0.69	0.45	0.47	0.52	0.78	0.65	0.59	0.44
	Goby	-[3]	-[4]	0.72	0.75	0.85	0.75	0.83	0.66
Comp./decomp times (lossless) [min]	NGC	150/95	8/7	26/8	41/10	89/55	12/4	17/72	38/14
	Cram	26/65	1/1	3/6	4/9	20/50	1/3	1/3	3/8
	Goby	-	-	8/13	11/20	135/170[5]	3/5	4/7	9/16
Lossy compression ratios	NGC (m4)	0.29	0.07	0.07	0.02	0.08	0.03	0.02	0.06
	Cram-lossy	0.35	0.08	0.10	0.03	0.22	0.05	0.03	0.09
Comp./decomp times (lossy) [min]	NGC (m4)	94/71	8/7	21/6	32/6	43/37	9/2	14/137	32/11
	Cram-lossy	27/56	1/1	2/5	3/6	16/34	1/2	1/2	3/6

Table 4.1: Evaluation data sets. The table provides some data describing our evaluation data sets and compares the compression ratios (i.e., compressed divided by uncompressed size) and compression/decompression times of NGC, cram and goby for lossless and lossy compression (commandline parameters for the various modes are given in Appendix A.7). Read names were not included in these data sets, all BAM files contained tags that were preserved during the de/compression. The bottom section of the table compares compression ratios and times for NGC's m4 mode and cram's lossy configuration (see text). Absolute numbers for all other lossy modes are given in Appendix A.

Better compression with slightly higher counts of false positives/negatives is achieved with m2, which uses our proposed quantization scheme f_{quant} (see Definition 4.1) for Q1 and Q2 bases. M4 basically quantizes

[3] Goby could not compress this data set within two days after which we stopped the job.

[4] Goby could compress/decompress this data set, however, the resulting BAM file contained less reads than the original.

[5] We had to increase the maximum heap size for compressing this data set with goby, see Appendix A

the same q-values as cram-lossy but results in better compression, m3 lies between m2 and m4 in most measures. M5 provides maximum compression which is, however, near the ratios achieved by m4/cram. The recovery mode was configured equal to m2 but here we additionally passed the set of variants called from the respective lossless data set as parameter and did not quantize q-values at these positions. This resulted in only slightly worse compression but could increase precision and sensitivity for most data sets. For some, however, precision slightly dropped (e.g., Reseq *(E. coli)*).

Evaluation mode	Q1	Q2	Q3	Q4	RLE?	Low MAPQ filter	Known variants?	AVG % of recovered variants
lossless	-	-	-	-	no	-	-	100.0%
m1	std	-	-	-	yes	-	-	99.7%
m2	std	std	-	-	yes	-	-	98.9%
m3	30	std	-	-	yes	-	-	98.7%
m4	30	30	-	-	yes	-	-	96.1%
cram-lossy	30	30[6]	-	-	-	-	-	96.1%
m5	30	30	std	-	yes	20	-	95.8%
recovery	std	std	-	-	yes	-	yes	99.7%
drop all	30	30	30	30	yes	-	-	92.7%

Table 4.2: Evaluation modes. The columns Q1-Q4 list which quantization strategy was used for which kind of q-values. An entry "std" refers to the proposed standard binning scheme that is explained in the text. The value "30" means that the respective q-values were dropped and assigned the constant value 30. The column RLE shows whether run-length encoding was used for q-value compression or not, the "low MAPQ filter" column shows whether q-values of reads with a mapping quality lower than the given value were dropped or not and the "known variants" column shows whether a set of known variants (the ones called in the lossless data set) was used to lock q-values in the respective columns. The last column lists the achieved overall percentage of recovered variants. Note that this measure neglects the false positive rates of the respective methods and refer to the recovery precision and sensitivity measures in Figure 4.4 and to the data tables in Appendix A for details in this regard. The "cram-lossy" mode in row 6 refers to the lossy configuration of cram we evaluated against. All other modes were realized by configuring our NGC tool. Note that the configuration of the recovery mode in row 8 was derived from the m2 mode.

Finally, we were interested in how well the individual quantization modes "preserved" the predicted genotype of called variants. For this, we used GATK's method to classify each found variant as homozygous, heterozygous or unknown. Then we counted how many of the variants that were re-detected after compression/decompression with the respective quantization method (i.e., the true positives) changed their classification from homozygous to heterozygous or vice versa. In Figure 4.4, we plotted the percentage of these "preserved" variants in the true positives. Not surprisingly, methods with the lowest compression ratios lead not only to more false positives and negatives but also to more variants with wrongly predicted genotypes which might be an issue for use cases where these data is considered (e.g., when searching for loss-of-heterozygosity regions). It can also be observed that the recovery mode performed well in preserving these classifications, and it outperforms it's nearest comparable mode (m2).

4.3.2 Computing times

We compared the compression and decompression times of NGC with cram and goby and list the times in Table 4.1. NGC is considerably slower than cram in both categories, although the differences are smaller when considering decompression. We attribute this mainly to the following two reasons: first, the VDRLE

[6]Note that cram-lossy preserves qualities of Q2 bases if there are two or more reads that differ from the reference at a particular column which is a slightly different behaviour than in NGC's m4 mode where all Q2 q-values are quantized.

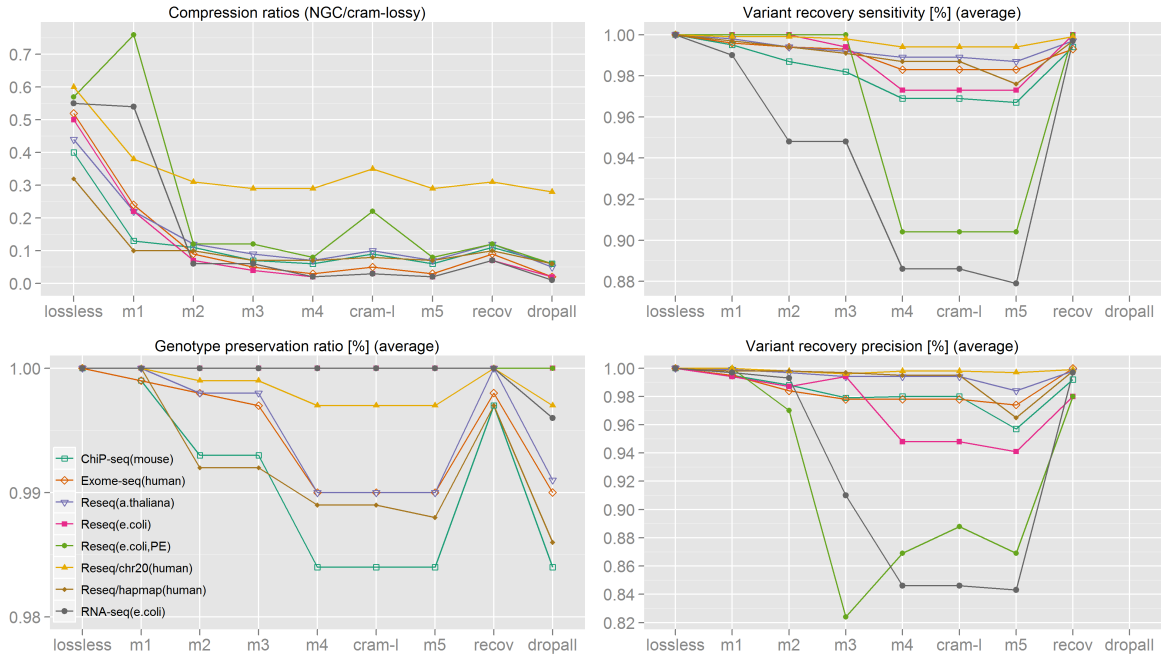


Figure 4.4: Evaluation data. The upper-left figure compares the compression ratios of the different modes for the used evaluation data sets (cram-l refers to the used cram-lossy mode). The absolute byte sizes and the compression ratios of the decompressed BAM files can be found in Appendix A. Note that the Reseq/chr20 (human) BAM files are less compressible as they contain large sections of optional BAM tags (see Appendix A). The figures on the right side show the variant recovery precision and sensitivity averaged over the data we obtained by applying two different variant-calling tools (GATK and SAMTOOLS). We omitted the data for the "drop all" mode in these two figures for readability as the corresponding values would require a much larger scale. The lower-left figure shows how many of the re-found variants (i.e., the true positives) did not change their predicted genotype classification, either from homozygous to heterozygous or vice versa. All absolute data values and the used command-line parameters are given in Appendix A. Please note the differing scales of the y-axes in the four figures.

approach is much more time consuming when compared with traversing an ordered alignment read-by-read. Second, we have not yet optimized our tool in this regard as we consider it as an archiving solution where computing times may not be the primary issue when they stay within practically useful limits. Although efficiently computing the compressed data may not be an issue, it is certainly helpful if the decompression of data is fast and here NGC is not substantially worse than cram and even faster than goby except for the RNA-Seq (E. coli) data set.

The lossy compression/decompression modes of NGC and cram are commonly faster when compared with the respective lossless modes (see Table 4.1. Again, RNA-Seq (E. coli) is a considerable exception where the decompression times nearly doubled for NGC). For some data sets, NGC's lossy modes compressed twice as fast and decompressed 66% faster (absolute times are given in Appendix A).

4.4 Discussion

Continuously dropping costs per sequenced genome and technology advances, such as longer read lengths and shorter sequencing durations, will result in enormous amounts of data directly associated with HTS experiments. This issue cannot be ignored by referring to the also dropping costs per byte of secondary/tertiary storage as (i) the rate at which sequencing data is produced already overtook the rate of storage production (2,3); (ii) it is not only the data storage but also their increasing transmission, e.g., between nodes in a cluster-/cloud-computing environment, between storage servers/long-term archives in a research facility or between research centres via the Internet which imposes a challenge; (iii) data transmission leads in many cases also to data redundancies (e.g., cached parts, data copies) that further enlarge this "sequence data heap". All this will constitute a major bottleneck for HTS experiments in the future that can partly be widened when reducing file sizes by data compression. Our proposed solution for the compression of SAM/BAM files contributes two major improvements to the state of the art. First, we propose a way to exploit common features of reads mapped to the same position in order to reduce the required number of code words. Second, we propose a model for the controlled lossy compression of per-base quality values that enables a comprehensive configuration of the trade-off between low compression ratios and high variant recovery rates. Lossless compression with NGC may save between one-third and two-thirds of the disc space required for such alignment files, depending on characteristics of the input data. With lossy compression, however, we reach space savings up to 98%. In this thesis, we contribute a novel method for the compression of read bases in mapped alignments that encodes read bases losslessly using 0.12-0.03 bits/base. We achieve this by representing read bases in a vertical (per-column) way as also done by previous data formats, such as CALF [Gre08] or the Ensembl Multi Format (EMF) [FAB+08]. However, different from those, we exploit common features between reads mapped to the same genomic position in order to reduce the number of required code words. We further contribute a novel approach for the lossy compression of per-base quality values. These quality values account for up to 97% of the used disc space in our losslessly compressed files which can be reduced to 2-33% with our proposed lossy modes. Note that our idea of vertical difference run-length encoding could also be used to compress whole sets of alignments that were mapped to the same genome (e.g., stemming from multiple resequencing experiments or from metagenomics analyses) and we plan to elaborate on this in future research. One possible improvement of our VDRLE approach is to further enhance the clustering of equal bases in (deep) alignment columns by applying a Burrows-Wheeler transform to them, cf. [CBJR12]. Lossy compression means a controlled loss of information that usually affects downstream data handling methods. Kozanitis et al. [KSK+11] have already shown that their model of lossy q-value representation shows only little impact on SNP calling with CASAVA, and somewhat comparable, yet in a more elaborated manner, we evaluated various q-value quantization strategies in order to find optimal trade-offs between compression rates and variant recovery. Different from this study, we used a more complex variant calling pipeline for evaluation that included INDEL realignment, duplicate removal and two widely-adopted variant calling tools, namely GATK and SAMTOOLS. The idea of our proposed lossy modes is to preserve maximum precision in those parts of the data that are of high relevance to downstream applications. Variant-calling or genotype prediction software, for example, considers an alignment and its q-values also positionwise and will benefit from the maintained q-value characteristics in such columns. Note, however, that variant recovery in our evaluation depends not only on the actual variant calling tool but also on upstream processes that are influenced by q-values, such as the PCR duplicate removal. Our evaluation showed that NGC outperforms the comparable cram and goby approaches with respect to lossless and lossy compression. Note that the BAM files resulting from decompression of our lossy compressed alignments are also considerably (up to 5X) smaller than the originals. This is because quality quantization will result also in improved compression of BAM's gzip blocks. These BAM files are, however, still considerably larger than the corresponding NGC files (see Appendix A). The configurability of our tool further allows for lossy q-value quantization approaches that outperform cram-lossy with respect to higher variant recovery and higher genotype preservation rates. Fur-

thermore, cram currently suffers from not being able to reconstruct original read names[7] whereas our tool can reconstruct BAM alignments completely, including the original file headers and all optional tags, and is independent of additional data, such as BAM index files. The current version of cram is, however, considerably faster than our tool when we compress data. Further, our data format does not allow efficient random access to portions of its data, as proper index structures are currently missing. As our tool was designed for archiving of BAM files, we do not consider this as a major obstacle for its application; we do, however, plan to improve NGC in this regard. Our q-value quantization methods also outperform the value transformation proposals in [WAA11], who report compression rates of 1 bit/q-value for a data set we also used in our evaluation. Other related work to NGC is the cSRA format that stores compressed-by-reference alignments in the sequence read archive [KSLI12], SAMZIP [STZH11], a tool that is specialized to compress whole SAM/BAM files but does not allow lossy compression and SlimGene [KSK+11] that compresses reads in the Illumina Export format.

4.4.1 Unmapped reads

As other reference based encoding methods, our proposed approach for sequence data compression cannot be applied to unmapped reads. We, therefore, currently store those in an own file block encoded in the original BAM format. We do, however, quantize their q-values in lossy compression modes (for this purposes, an own quantization scheme can be defined, i.e. an additional fifth q-value category was added). Fritz et al. [HYFLCB11] proposed the assembly of contiguous sequences in such unmapped reads to which these could then be mapped; however, their method resulted in additional mapping of only about 15% of those reads and consequently achieves only limited improvement of compression ratios.

[7] As of August 2012, the latest version of cram (v0.9) is also able to compress read names.

Chapter 5

Conclusions

High-throughput sequencing technologies considerably changed the landscape of available analysis methods in several biology-related fields recently. Apart from enabling whole new scientific approaches such as genome-wide association studies they have the potential to completely replace existing, wide-spread analysis methods such as Microarrays.

Altman et al. called HTS the „Swiss pocketknife of molecular biology" [AWB+12], explicitly referring to the flexibility of the method but not to the size of the produced data. On the contrary, the enormous amounts of data associated with HTS experiments developed into one of the biggest bottlenecks today. And the situation will worsen in the near future with emerging commercial and research fields such as Personal Genomics [TMF09] and constantly improving technologies that will result, e.g., in longer read lengths at less costs and possibly more produced data per run.

One possibility to widen the mentioned bottleneck is the development of specialized data compression algorithms that may reduce the effective size of the resulting data. In this regards, the current development in the area of HTS data compression might, to some extent, be comparable with the situation of multimedia research shortly before audio, video and image capturing devices became available to the mass market: the application of the whole arsenal of available data compression algorithms led to the development of novel, specialized, lossless and lossy (e.g., JPEG or MP3) compression algorithms that ultimately enabled further progress in the development of capturing devices. It is our belief that efficient HTS data compression algorithms will play a gable role in the field of nucleic acid sequencing.

In this thesis, we presented and evaluated NGC, one such solution for reducing the enormous data heap that is associated with today's high-throughput sequencing experiments.

5.1 Future Work

Our solution introduces some novel aspects to the field of HTS data compression but is still in its infancy and suffers from several major drawbacks that should be addressed in future work.

Most importantly should the compression and more importantly the decompression speed of NGC be optimized. This could be achieved, e.g., by the introduction of multi-threading or by optimization of the stream handling implementations.

Furthermore, efficient random access to the contained information is required. This might be achieved by implementing efficient indexing structures (e.g., based on interval trees, recursive tree data structures that allow efficient access to a set of intervals [Ede80, McC80, Sam05]). Such an index could store the byte offsets of all NGC data streams for selected genomic positions which would enable NGC to "jump" directly into the respective data sections. By this, users could, e.g., extract only subsets of the contained HTS data such as only the reads from a particular genomic region.

Another useful extension to NGC would be to make the data format completely streamable. This means that the NGC compressor should be able to read SAM records (or possibly also other input formats) from some input stream and immediately outputs the compressed data to some output stream. *Vice versa*, the decoder should be able to read from such a stream and directly write SAM record to an output stream. This would enable the transmission of compressed mapped read data over networks and the integration of NGC with stream-based UNIX tools. Such an extension to NGC should be quite straightforward as our implementation is internally already based on data streams. One would only have to multiplex these streams at runtime (currently, the various streams are written to individual files that are merged in the end) and provide respective buffering data structures to preserve data integrity.

Last but not least should the meta data handling capabilities of NGC be extended. It would be useful to be able to store meta data describing, e.g., the data provenance or the used reference sequence directly in an NGC archive which would provide the basis for automatic data integration and validation in the future.

Part I

Addenda

Appendix A

Supplementary Data

A.1 Supplementary Description of Variant Calling Pipelines

The Reseq/chr20 *(human)* data set corresponds to *NA12878.HiSeq.WGS.bwa.cleaned.recal.hg19.20.bam*, a BAM file that was downloaded from the GATK resource bundle v1.5 b37[1]. This file contains mapped reads of the NA12878 individual's chromosome 20. Note that this data set contains all original q-values of the reads (probably the ones before q-value recalibration) stored in OQ tags which makes this file less compressible in comparison to the others. All other data sets were downloaded from the SRA respectively given to us as FASTQ files and were mapped using bwa v 0.6.1-r104 [LD10] with standard parameters for single respectively paired end data. The following reference sequences were used for mapping:

1. Homo sapiens genome human g1k v37, available at ftp://ftp.sanger.ac.uk/pub/1000genomes/tk2/main_project_reference/human_g1k_v37.fasta.gz
2. Mus musculus genome mm9, NCBI Build 37, available at http://hgdownload.cse.ucsc.edu/goldenPath/mm9/bigZips/mm9.2bit
3. Escherichia coli str. K-12 substr. MG1655 genome, available at ftp://ftp.ncbi.nlm.nih.gov/genomes/Bacteria/Escherichia_coli_K_12_substr__MG1655_uid57779/NC_000913.fna
4. Arabidopsis thaliana TAIR10 genome, available at ftp://ftp.arabidopsis.org/home/tair/Sequences/whole_chromosomes/

Note that we pruned unmapped reads from the data sets using SAMTOOLS v0.1.18 [LHW+09] before our tests and that our algorithm expects the input SAM/BAM file to be sorted by mapping coordinates. Note also that our solution preserves also hard- and soft-clipped bases that actually break the order in such a coordinate-sorted file by storing those in some extra data streams (cf. Figure 4.1).

A.2 Computing Environment

The evaluation experiments were conducted on a server equipped with 2 Xeon E5520 processors and a total of 32GBytes of RAM. Both, cram and NGC, were executed using Java 1.7.0_02 64Bit with 4Gbytes of maximum heap size (-Xmx switch). Goby was executed using Java 1.6.0_27 64Bit as the latest version we used (v 2.1) did not run with Java 1.7. We also used 4Gbytes of maximum heap size, except for compressing the Reseq *(E. coli, PE)* data set where we used 16GByte as we ran into OutOfMemory errors.

[1] Available at ftp://gsapubftp-anonymous@ftp.broadinstitute.org/bundle/1.5/b37, be sure to login as user gsapubftp-anonymous/<blank>. The file can be found in the subfolder bundle/1.5/b37.

A.3 GATK Pipeline

Our pipeline for GATK [MHB+10] variant calling consisted of the following steps:

- Addition of read groups using Picard v1.56 (http://picard.sourceforge.net), AddOrReplaceReadGroups
- INDEL realignment with GATK v1.4 RealignerTargetCreator and IndelRealigner
- Duplicate removal with Picard MarkDuplicates
  - Duplicates were removed from the alignment
  - Note that this step was omitted for the RNA-seq (e.coli) data set!
- Variant calling with GATK UnifiedGenotyper
- Variant filtration with GATK VariantFiltration
  - For SNPs, we filtered for low coverage (<5), low quality (<50), variant confidence (<1.5), strand bias and mapping quality
  - For INDELs, we filtered for variant confidence, read position bias and strand bias
- Both variant sets were then combined using GATK CombineVariants
- Note that the first three pipeline stages were omitted for the Reseq/chr20 (human) data set as comparable preprocessing was already done for the respective BAM file

A.4 SAMTOOLS Pipeline

Our pipeline for variant calling with SAMTOOLS consisted of the following steps:

- Addition of read groups using Picard AddOrReplaceReadGroups
- INDEL realignment with GATK RealignerTargetCreator and IndelRealigner
- Duplicate removal with Picard MarkDuplicates
  - This step was omitted for the RNAseq data set.
- Variant calling with SAMTOOLS v0.1.18 mpileup/bcftools/vcfutils
  - Variants were filtered by minimum (5) and maximum (2000) read depth
- Note that the first three pipeline stages were omitted for the Reseq/chr20 (human) data set as comparable preprocessing was already done for the respective BAM file

Note that we did not do any quality score recalibration in this evaluation pipeline as such a step would affect all quality scores in the data sets which would blur the effects of the different quantization scenarios. Further note, that the Picard algorithm for the removal of redundant sequences (stemming mainly from PCR amplification) also incorporates read bases qualities and will therefore also lead to different read-sets depending on the used q-value quantization strategy.

A.5 Definitions

True positives (tp), false positives (fp) and false negatives (fn) were determined by comparing the sets of called variants from the BAM files created by compressing and uncompressing the original alignment using (1) the lossless mode (unquantized) and (2) the respective quantization mode.

Definition A.1 (Compression ratio) $cr = \frac{compressedsize}{uncompressedsize}$

Compression rates are, in contrast, usually measured in bits per sample.

Definition A.2 (Space savings) $1 - cr$

Definition A.3 (Precision) $prec = \frac{tp}{tp + fp}$

Definition A.4 (Sensitivity (aka recall rate)) $sens = \frac{tp}{tp + fn}$

Definition A.5 (Genotype preservation percentage) $gpp = 1 - \frac{cgt}{tp}$

where cgt is the number of variants from the set of true positives that changed their genotype classification from homozygous to heterozygous or vice versa.

Definition A.6 (Coverage) $cov = \frac{sb}{ref}$

where sb is the number of sequenced bases and ref is the length of the reference sequence.

A.6 Counting Horizontal and Vertical Run-lengths

For counting the run-lengths (RLs) presented in Figure 4.2 we considered two streams of read bases. The "horizontal" one was created by simply iterating over the reads in their given order in the alignment file. The "vertical" one was produced as described in the figure. Note that clipped bases are not counted and that RLs are also counted within insertions. In the vertical mode, we consider each inserted "column" as follows: for each read spanning the insertion, we consider either its inserted base character (A,C,T,G,N and all other IUPAC/FASTA characters) in this column or, if none, (e.g., in padded regions) a special character "X". These base characters are then "diffed" against the reference character "X" (so that a padded position finally results in an "E" character). Note that this method slightly penalizes the vertical mode, i.e., a more optimized method would count even less required RLs in the vertical mode. Further note that the given numbers are the "theoretical" numbers of required run-lengths as described in Chapter 4. In practice, however, not only these numbers but also the amount of bits needed to represent this information is relevant. For representing the length-values of the RLs one could, for example, use fixed-sized 8-bit code words. This would mean that RLs with a length value greater than 256 have to be split-up, which again increases the number of required code words. Reserving more bits would again reduce the number of code words but would require more disc space per code word. With our data we actually found that using one byte per RL length was the best tradeoff in this regard (data not shown).

A.7 Compression Parameters

We used the following commandline parameters for the evaluation experiments:

A.7.1 NGC v0.0.1

Lossless:

```
-best -truncateNames
```

m1:

```
-best -q1levels standard -qvalRleEncoding -truncateNames
```

m2:

```
 -best -q1levels standard -q2levels standard -qvalRleEncoding -truncateNames
```

m3:

```
-best -q1levels standard -qvalRleEncoding -truncateNames
```

m4:

```
 -best -q1levels 30 -q2levels 30 -qvalRleEncoding -truncateNames
```

m5:

```
-best -q1levels 30 -q2levels 30 -q4levels standard -qvalRleEncoding -mmq 20 -q5levels 30 -truncateNames
```

recovery:

```
-best -q1levels 30 -q2levels standard -qvalRleEncoding -truncateNames -variantList <variants.vcf>
```

drop all:

```
 -best -q1levels 30 -q2levels 30 -q3levels 30 -q4levels 30 -qvalRleEncoding -truncateNames
```

A.7.2 Cram v0.85

cram lossless compression:

```
--capture-all-quality-scores --capture-all-tags
```

cram lossy compression:

```
--capture-insertion-quality-scores --capture-piled-quality-scores
--capture-substitution-quality-scores --capture-unmapped-quality-scores --capture-all-tags
```

cram decompression:

```
--calculate-md-tag --calculate-nm-tag
```

A.7.3 Goby v2.1

goby lossless compression:

```
--sorted --preserve-all-tags --preserve-soft-clips --preserve-all-mapped-qualities
--ambiguity-threshold 1000000
```

goby decompression:

```
<no options>
```

NOTE: Goby could compress/decompress the Reseq/hm *(human)* data set, however, the resulting BAM file contained less reads than the original. Goby could not compress the Reseq/chr20 *(human)* with 16GByte of dedicated RAM within two days after which we stopped the job.

A.8 Supplementary Data Tables

This section contains the absolute data values from our evaluation experiments.

data set	Evaluation mode	mapped reads	Bases and qvalues	BAM [byte]	NGC [byte]	Comp. ratio	Space saving	NGC bases [byte]	bits/ base	NGC qvals [byte]	bits/ qvalue	% of total	decompressed BAM [byte]	comp ratio	comp [min]	decomp [min]	name stream [byte]	% of total	bytes per read
	noquant (lossless)	3,239,217	323,921,700	199,019,319	102,584,334	52%	48%	1,715,214	0.04	97,856,190	2.42	95%	171,711,628	0.86	12.3	3.7	18,078,007	15%	5.6
	dropall (drop all)	-	-	-	4,791,322	2%	98%	-	-	63,162	0.00	1%	44,041,660	0.22	10.6	2.3	-	79%	-
	quant-bestacc1 (m2)	-	-	-	17,126,057	9%	91%	-	-	12,397,899	0.31	72%	56,928,926	0.29	9.0	2.4	-	51%	-
Exome-seq	quant-bestacc2 (m1)	-	-	-	46,917,260	24%	76%	-	-	42,189,120	1.04	90%	111,001,230	0.56	10.2	2.3	-	28%	-
(human)	quant-bestacc3 (m3)	-	-	-	10,186,064	5%	95%	-	-	5,457,922	0.13	54%	53,182,239	0.27	8.9	2.3	-	64%	-
	quant-recover (recovery)	-	-	-	17,156,135	9%	91%	-	-	12,427,791	0.31	72%	57,045,240	0.29	9.5	2.6	-	51%	-
	quant-cramstyle (m4)	-	-	-	6,715,773	3%	97%	-	-	1,987,637	0.05	30%	48,425,207	0.24	9.2	2.4	-	73%	-
	bestcomp (m5)	-	-	-	6,650,525	3%	97%	-	-	1,922,359	0.05	29%	48,129,248	0.24	9.1	2.4	-	73%	-
	noquant (lossless)	13,824,441	497,679,876	637,462,912	253,351,159	40%	60%	4,871,794	0.08	218,041,390	3.50	86%	575,748,603	0.90	37.9	13.9	55,374,026	18%	4.0
	dropall (drop all)				35,480,052	6%	94%	-	-	170,300	0.00	0%	300,547,675	0.47	33.1	11.3	-	61%	-
	quant-bestacc1 (m2)	-	-	-	70,735,770	11%	89%	-	-	35,426,026	0.57	50%	352,336,803	0.55	34.0	14.3	-	44%	-
ChiP-seq	quant-bestacc2 (m1)	-	-	-	81,960,525	13%	87%	-	-	46,650,799	0.75	57%	370,004,971	0.58	34.3	14.6	-	40%	-
(mouse)	quant-bestacc3 (m3)	-	-	-	43,025,778	7%	93%	-	-	7,716,044	0.12	18%	313,736,041	0.49	32.5	11.9	-	56%	-
	quant-recover (recovery)	-	-	-	70,792,579	11%	89%	-	-	35,482,691	0.57	50%	352,502,637	0.55	33.0	11.7	-	44%	-
	quant-cramstyle (m4)	-	-	-	40,893,894	6%	94%	-	-	5,584,166	0.09	14%	309,874,382	0.49	32.3	11.4	-	58%	-
	bestcomp (m5)	-	-	-	39,819,255	6%	94%	-	-	4,509,497	0.07	11%	308,338,605	0.48	32.3	11.3	-	58%	-
	noquant (lossless)	487,201	62,435,260	53,036,863	16,768,124	32%	68%	699,000	0.09	13,700,379	1.76	82%	52,001,981	0.98	8.1	7.0	1,521,858	8%	3.1
	dropall (drop all)				3,346,561	6%	94%	-	-	281,054	0.04	8%	33,707,167	0.64	7.7	6.9	-	31%	-
	quant-bestacc1 (m2)	-	-	-	5,351,544	10%	90%	-	-	2,285,438	0.29	43%	36,420,556	0.69	7.8	7.0	-	22%	-
Reseq/hm	quant-bestacc2 (m1)	-	-	-	5,420,639	10%	90%	-	-	2,354,551	0.30	43%	36,541,621	0.69	8.1	7.0	-	22%	-
(human)	quant-bestacc3 (m3)	-	-	-	3,907,285	7%	93%	-	-	841,796	0.11	22%	34,414,107	0.65	7.7	6.8	-	28%	-
	quant-recover (recovery)	-	-	-	5,353,280	10%	90%	-	-	2,287,030	0.29	43%	36,425,209	0.69	8.9	6.9	-	22%	-
	quant-cramstyle (m4)	-	-	-	3,895,898	7%	93%	-	-	830,415	0.11	21%	34,397,816	0.65	7.7	6.8	-	28%	-
	bestcomp (m5)	-	-	-	3,572,059	7%	93%	-	-	506,546	0.06	14%	33,951,894	0.64	7.7	6.8	-	30%	-
	noquant (lossless)	6,927,728	249,398,208	177,499,665	97,572,476	55%	45%	1,069,347	0.03	94,967,939	3.05	97%	162,770,179	0.92	16.8	71.5	21,872,597	18%	3.2
	dropall (drop all)	-	-	-	2,635,471	1%	99%	-	-	30,892	0.00	1%	38,063,337	0.21	14.5	137.3	-	89%	-
	quant-bestacc1 (m2)	-	-	-	11,535,980	6%	94%	-	-	8,931,413	0.29	77%	56,841,423	0.32	14.0	137.4	-	65%	-
RNA-seq	quant-bestacc2 (m1)	-	-	-	96,097,149	54%	46%	-	-	93,492,600	3.00	97%	152,276,330	0.86	16.5	140.4	-	19%	-
(e.coli)	quant-bestacc3 (m3)	-	-	-	10,625,902	6%	94%	-	-	8,021,341	0.26	75%	55,244,237	0.31	13.9	137.7	-	67%	-
	quant-recover (recovery)	-	-	-	11,885,504	7%	93%	-	-	9,280,793	0.30	78%	57,550,581	0.32	13.9	137.1	-	65%	-
	quant-cramstyle (m4)	-	-	-	3,864,545	2%	98%	-	-	1,259,990	0.04	33%	40,101,575	0.23	13.5	136.7	-	85%	-
	bestcomp (m5)	-	-	-	3,861,949	2%	98%	-	-	1,257,364	0.04	33%	40,099,384	0.23	13.7	137.4	-	85%	-
	noquant (lossless)	44,938,891	4,538,790,538	2,945,453,583	1,668,314,197	57%	43%	37,487,695	0.07	1,491,019,976	2.63	89%	2,785,167,869	0.95	88.6	54.9	170,317,681	9%	3.8
	dropall (drop all)	-	-	-	177,821,365	6%	94%	-	-	1,090,498	0.00	1%	865,556,533	0.29	52.7	36.6	-	49%	-
	quant-bestacc1 (m2)	-	-	-	346,569,412	12%	88%	-	-	169,658,967	0.30	49%	1,050,959,891	0.36	47.2	41.8	-	33%	-
Reseq	quant-bestacc2 (m1)	-	-	-	2,228,518,987	76%	24%	-	-	2,051,608,560	3.62	92%	2,806,699,545	0.95	102.6	109.3	-	7%	-
(e.coli,PE)	quant-bestacc3 (m3)	-	-	-	345,407,618	12%	88%	-	-	168,676,769	0.30	49%	1,051,615,770	0.36	47.2	42.1	-	33%	-
	quant-recover (recovery)	-	-	-	346,614,897	12%	88%	-	-	169,704,304	0.30	49%	1,051,072,595	0.36	47.8	44.2	-	33%	-
	quant-cramstyle (m4)	-	-	-	221,446,088	8%	92%	-	-	44,715,245	0.08	20%	942,726,965	0.32	42.6	36.7	-	43%	-
	bestcomp (m5)	-	-	-	221,451,594	8%	92%	-	-	44,720,721	0.08	20%	942,726,968	0.32	42.3	36.2	-	43%	-
	noquant (lossless)	19,379,287	697,654,332	654,501,526	324,615,167	50%	50%	2,973,219	0.03	314,274,125	3.60	97%	563,155,799	0.86	40.5	9.7	77,362,742	19%	4.0
	dropall (drop all)	-	-	-	10,417,673	2%	98%	-	-	76,589	0.00	1%	152,097,133	0.23	34.4	6.0	-	88%	-
	quant-bestacc1 (m2)	-	-	-	44,230,055	7%	93%	-	-	33,888,983	0.39	77%	221,512,742	0.34	33.7	7.9	-	64%	-
Reseq (e.	quant-bestacc2 (m1)	-	-	-	142,243,365	22%	78%	-	-	131,902,311	1.51	93%	407,132,803	0.62	37.6	12.2	-	35%	-
coli)	quant-bestacc3 (m3)	-	-	-	25,244,189	4%	96%	-	-	14,903,123	0.17	59%	191,677,341	0.29	32.7	7.0	-	75%	-
	quant-recover (recovery)	-	-	-	44,229,978	7%	93%	-	-	33,888,762	0.39	77%	221,514,134	0.34	33.0	6.4	-	64%	-
	quant-cramstyle (m4)	-	-	-	13,734,400	2%	98%	-	-	3,393,340	0.04	25%	157,659,318	0.24	32.1	5.9	-	85%	-
	bestcomp (m5)	-	-	-	13,730,685	2%	98%	-	-	3,389,595	0.04	25%	157,642,215	0.24	32.1	5.8	-	85%	-
	noquant (lossless)	11,651,942	419,469,912	503,542,318	220,974,482	44%	56%	6,505,896	0.12	194,212,884	3.70	88%	455,320,689	0.90	25.6	7.6	45,409,933	17%	3.9
	dropall (drop all)	-	-	-	27,008,448	5%	95%	-	-	246,964	0.00	1%	213,729,259	0.42	21.2	5.1	-	63%	-
	quant-bestacc1 (m2)	-	-	-	60,370,238	12%	88%	-	-	33,608,747	0.64	56%	267,101,809	0.53	21.8	6.9	-	43%	-
Reseq (a.	quant-bestacc2 (m1)	-	-	-	112,676,889	22%	78%	-	-	85,915,416	1.64	76%	334,463,372	0.66	23.3	8.9	-	29%	-
thaliana)	quant-bestacc3 (m3)	-	-	-	43,078,926	9%	91%	-	-	16,317,460	0.31	38%	242,511,743	0.48	21.1	6.0	-	51%	-
	quant-recover (recovery)	-	-	-	60,540,777	12%	88%	-	-	33,779,142	0.64	56%	267,512,418	0.53	21.2	5.8	-	43%	-
	quant-cramstyle (m4)	-	-	-	34,976,357	7%	93%	-	-	8,214,897	0.16	23%	228,982,963	0.45	21.0	5.6	-	56%	-
	bestcomp (m5)	-	-	-	33,880,881	7%	93%	-	-	7,119,391	0.14	21%	226,934,332	0.45	20.8	5.5	-	57%	-
	noquant (lossless)	50,663,069	5,116,969,969	5,867,598,471	3,499,104,195	60%	40%	22,518,040	0.04	1,838,887,311	2.87	53%	5,389,447,424	0.92	149.9	94.9	288,669,905	8%	5.7
	dropall (drop all)				1,656,521,548	28%	72%	-	-	1,337,838	0.00	0%	3,078,877,612	0.52	112.0	69.8	-	15%	-
	quant-bestacc1 (m2)				1,838,534,489	31%	69%	-	-	181,684,357	0.28	10%	3,319,031,623	0.57	99.8	79.1	-	14%	-
Reseq/chr20	quant-bestacc2 (m1)				2,213,480,428	38%	62%	-	-	556,630,314	0.87	25%	4,096,347,307	0.70	111.0	93.5	-	12%	-
(human)	quant-bestacc3 (m3)				1,718,428,520	29%	71%	-	-	63,244,828	0.10	4%	3,217,765,094	0.55	95.3	72.5	-	14%	-
	quant-recover (recovery)				1,840,308,198	31%	69%	-	-	183,457,841	0.29	10%	3,328,321,967	0.57	96.7	73.7	-	14%	-
	quant-cramstyle (m4)				1,684,592,441	29%	71%	-	-	29,408,755	0.05	2%	3,156,695,367	0.54	94.4	70.9	-	15%	-
	bestcomp (m5)				1,683,782,581	29%	71%	-	-	28,598,865	0.04	2%	3,151,866,089	0.54	94.6	71.8	-	15%	-

Table A.2: NGC compression ratios/times

Data set	Q-value quantization strategy	tp	fp	fn	tn	acc	prec	sens	spec	HET/HOM	HET/HOM (%)	recovered variants(%)
Exome-seq (human)	noquant (lossless)	3,994	-	-	3,101,800,745	1.000	1.000	1.000	1.000	-	100.0%	100.0%
	dropall (drop all)	3,897	6,013	97	3,101,794,732	1.000	0.393	0.976	1.000	52	98.7%	97.6%
	quant-bestacc1 (m2)	3,982	85	12	3,101,800,660	1.000	0.979	0.997	1.000	13	99.7%	99.7%
	quant-bestacc2 (m1)	3,982	28	12	3,101,800,717	1.000	0.993	0.997	1.000	4	99.9%	99.7%
	quant-bestacc3 (m3)	3,975	98	19	3,101,800,647	1.000	0.976	0.995	1.000	17	99.6%	99.5%
	quant-recover (recovery)	3,966	76	28	3,101,800,669	1.000	0.981	0.993	1.000	9	99.8%	99.3%
	quant-cramstyle (m4)	3,928	88	66	3,101,800,657	1.000	0.978	0.983	1.000	46	98.8%	98.3%
	bestcomp (m5)	3,924	91	70	3,101,800,654	1.000	0.977	0.982	1.000	46	98.8%	98.2%
	cram	3,928	88	66	3,101,800,657	1.000	0.978	0.983	1.000	46	98.8%	98.3%
ChiP-seq (mouse)	noquant (lossless)	16,929	-	-	2,725,748,552	1.000	1.000	1.000	1.000	-	100.0%	100.0%
	dropall (drop all)	15,379	5,700	1,550	2,725,742,852	1.000	0.730	0.908	1.000	208	98.6%	90.8%
	quant-bestacc1 (m2)	16,872	226	57	2,725,748,326	1.000	0.987	0.997	1.000	112	99.3%	99.7%
	quant-bestacc2 (m1)	16,869	68	60	2,725,748,484	1.000	0.996	0.996	1.000	19	99.9%	99.6%
	quant-bestacc3 (m3)	16,852	338	77	2,725,748,214	1.000	0.980	0.995	1.000	120	99.3%	99.5%
	quant-recover (recovery)	16,841	133	88	2,725,748,419	1.000	0.992	0.995	1.000	27	99.8%	99.5%
	quant-cramstyle (m4)	16,665	287	264	2,725,748,265	1.000	0.983	0.984	1.000	268	98.4%	98.4%
	bestcomp (m5)	16,622	1,116	307	2,725,747,436	1.000	0.937	0.982	1.000	269	98.4%	98.2%
	cram	16,665	287	264	2,725,748,265	1.000	0.983	0.984	1.000	268	98.4%	98.4%
Reseq/hm (human)	noquant (lossless)	861	-	-	3,101,803,878	1.000	1.000	1.000	1.000	-	100.0%	100.0%
	dropall (drop all)	816	259	45	3,101,803,619	1.000	0.759	0.948	1.000	8	99.0%	94.8%
	quant-bestacc1 (m2)	859	1	2	3,101,803,877	1.000	0.999	0.998	1.000	5	99.4%	99.8%
	quant-bestacc2 (m1)	859	-	2	3,101,803,878	1.000	1.000	0.998	1.000	-	100.0%	99.8%
	quant-bestacc3 (m3)	859	2	2	3,101,803,876	1.000	0.998	0.998	1.000	5	99.4%	99.8%
	quant-recover (recovery)	859	-	2	3,101,803,878	1.000	1.000	0.998	1.000	2	99.8%	99.8%
	quant-cramstyle (m4)	852	1	9	3,101,803,877	1.000	0.999	0.990	1.000	9	98.9%	99.0%
	bestcomp (m5)	832	39	29	3,101,803,839	1.000	0.955	0.966	1.000	9	98.9%	96.6%
	cram	852	1	9	3,101,803,877	1.000	0.999	0.990	1.000	9	98.9%	99.0%
RNA-seq (e.coli)	noquant (lossless)	151	-	-	4,639,524	1.000	1.000	1.000	1.000	-	100.0%	100.0%
	dropall (drop all)	127	148	24	4,639,376	1.000	0.462	0.841	1.000	1	99.2%	84.1%
	quant-bestacc1 (m2)	149	2	2	4,639,522	1.000	0.987	0.987	1.000	-	100.0%	98.7%
	quant-bestacc2 (m1)	148	1	3	4,639,523	1.000	0.993	0.980	1.000	-	100.0%	98.0%
	quant-bestacc3 (m3)	149	2	2	4,639,522	1.000	0.987	0.987	1.000	-	100.0%	98.7%
	quant-recover (recovery)	150	1	1	4,639,523	1.000	0.993	0.993	1.000	-	100.0%	99.3%
	quant-cramstyle (m4)	144	-	7	4,639,524	1.000	1.000	0.954	1.000	-	100.0%	95.4%
	bestcomp (m5)	142	1	9	4,639,523	1.000	0.993	0.940	1.000	-	100.0%	94.0%
	cram	144	-	7	4,639,524	1.000	1.000	0.954	1.000	-	100.0%	95.4%
Reseq (e.coli,PE)	noquant (lossless)	94	-	-	4,639,581	1.000	1.000	1.000	1.000	-	100.0%	100.0%
	dropall (drop all)	91	4,229	3	4,635,352	0.999	0.021	0.968	0.999	-	100.0%	96.8%
	quant-bestacc1 (m2)	94	6	-	4,639,575	1.000	0.940	1.000	1.000	-	100.0%	100.0%
	quant-bestacc2 (m1)	94	-	-	4,639,581	1.000	1.000	1.000	1.000	-	100.0%	100.0%
	quant-bestacc3 (m3)	94	51	-	4,639,530	1.000	0.648	1.000	1.000	-	100.0%	100.0%
	quant-recover (recovery)	94	4	-	4,639,577	1.000	0.959	1.000	1.000	-	100.0%	100.0%
	quant-cramstyle (m4)	76	27	18	4,639,554	1.000	0.738	0.809	1.000	-	100.0%	80.9%
	bestcomp (m5)	76	27	18	4,639,554	1.000	0.738	0.809	1.000	-	100.0%	80.9%
	cram	76	22	18	4,639,559	1.000	0.776	0.809	1.000	-	100.0%	80.9%
Reseq (e. coli)	noquant (lossless)	77	-	-	4,639,598	1.000	1.000	1.000	1.000	-	100.0%	100.0%
	dropall (drop all)	77	1,029	-	4,638,569	1.000	0.070	1.000	1.000	-	100.0%	100.0%
	quant-bestacc1 (m2)	77	1	-	4,639,597	1.000	0.987	1.000	1.000	-	100.0%	100.0%
	quant-bestacc2 (m1)	77	1	-	4,639,597	1.000	0.987	1.000	1.000	-	100.0%	100.0%
	quant-bestacc3 (m3)	76	4	1	4,639,594	1.000	0.950	0.987	1.000	-	100.0%	98.7%
	quant-recover (recovery)	77	2	-	4,639,596	1.000	0.975	1.000	1.000	-	100.0%	100.0%
	quant-cramstyle (m4)	75	4	2	4,639,594	1.000	0.949	0.974	1.000	-	100.0%	97.4%
	bestcomp (m5)	75	4	2	4,639,594	1.000	0.949	0.974	1.000	-	100.0%	97.4%
	cram	75	4	2	4,639,594	1.000	0.949	0.974	1.000	-	100.0%	97.4%
Reseq (a. thaliana)	noquant (lossless)	195,510	-	-	119,472,240	1.000	1.000	1.000	1.000	-	100.0%	100.0%
	dropall (drop all)	193,677	14,414	1,833	119,457,826	1.000	0.931	0.991	1.000	1,015	99.5%	99.1%
	quant-bestacc1 (m2)	195,362	449	148	119,471,791	1.000	0.998	0.999	1.000	128	99.9%	99.9%
	quant-bestacc2 (m1)	195,379	169	131	119,472,071	1.000	0.999	0.999	1.000	18	100.0%	99.9%
	quant-bestacc3 (m3)	195,327	712	183	119,471,528	1.000	0.996	0.999	1.000	148	99.9%	99.9%
	quant-recover (recovery)	195,293	324	217	119,471,916	1.000	0.998	0.999	1.000	35	100.0%	99.9%
	quant-cramstyle (m4)	195,044	723	466	119,471,517	1.000	0.996	0.998	1.000	1,141	99.4%	99.8%
	bestcomp (m5)	194,189	4,444	1,321	119,467,796	1.000	0.978	0.993	1.000	1,140	99.4%	99.3%
	cram	195,044	723	466	119,471,517	1.000	0.996	0.998	1.000	1,141	99.4%	99.8%
Reseq/chr20 (human)	noquant (lossless)	112,753	33	15	3,101,691,938	1.000	1.000	1.000	1.000	2	100.0%	100.0%
	dropall (drop all)	112,252	19,473	516	3,101,672,498	1.000	0.852	0.995	1.000	349	99.7%	100.0%
	quant-bestacc1 (m2)	112,740	206	28	3,101,691,765	1.000	0.998	1.000	1.000	50	100.0%	100.0%
	quant-bestacc2 (m1)	112,744	38	24	3,101,691,933	1.000	1.000	1.000	1.000	12	100.0%	100.0%
	quant-bestacc3 (m3)	112,695	322	73	3,101,691,649	1.000	0.997	0.999	1.000	211	99.8%	99.9%
	quant-recover (recovery)	112,738	95	30	3,101,691,876	1.000	0.999	1.000	1.000	27	100.0%	100.0%
	quant-cramstyle (m4)	112,106	99	662	3,101,691,872	1.000	0.999	0.994	1.000	407	99.6%	99.4%
	bestcomp (m5)	112,106	104	662	3,101,691,867	1.000	0.999	0.994	1.000	419	99.6%	99.4%
	cram	112,082	30	686	3,101,691,941	1.000	1.000	0.994	1.000	403	99.6%	99.4%

Table A.3: Variant recovery (GATK)

Data set	Q-value quantization strategy	tp	fp	fn	tn	acc	prec	sens	spec	HET/HOM	HET/HOM (%)	recovered variants(%)
Exome-seq (human)	noquant (lossless)	3,007	-	-	3,101,801,732	1.000	1.000	1.000	1.000	0	100.0%	100.0%
	dropall (drop all)	2,874	1,323	133	3,101,800,409	1.000	0.685	0.956	1.000	18	99.4%	95.6%
	quant-bestacc1 (m2)	2,982	35	25	3,101,801,697	1.000	0.988	0.992	1.000	5	99.8%	99.2%
	quant-bestacc2 (m1)	2,991	10	16	3,101,801,722	1.000	0.997	0.995	1.000	2	99.9%	99.5%
	quant-bestacc3 (m3)	2,981	58	26	3,101,801,674	1.000	0.981	0.991	1.000	5	99.8%	99.1%
	quant-recover (recovery)	2,983	16	24	3,101,801,716	1.000	0.995	0.992	1.000	3	99.9%	99.2%
	quant-cramstyle (m4)	2,953	65	54	3,101,801,667	1.000	0.978	0.982	1.000	24	99.2%	98.2%
	bestcomp (m5)	2,957	90	50	3,101,801,642	1.000	0.970	0.983	1.000	25	99.2%	98.3%
	cram	2,953	65	54	3,101,801,667	1.000	0.978	0.982	1.000	24	99.2%	98.2%
ChiP-seq (mouse)	noquant (lossless)	6,293	-	-	2,725,759,188	1.000	1.000	1.000	1.000	0	100.0%	100.0%
	dropall (drop all)	5,082	1,136	1,211	2,725,758,052	1.000	0.817	0.808	1.000	93	98.2%	80.8%
	quant-bestacc1 (m2)	6,148	62	145	2,725,759,126	1.000	0.990	0.977	1.000	40	99.3%	97.7%
	quant-bestacc2 (m1)	6,247	35	46	2,725,759,153	1.000	0.994	0.993	1.000	11	99.8%	99.3%
	quant-bestacc3 (m3)	6,095	137	198	2,725,759,051	1.000	0.978	0.969	1.000	47	99.2%	96.9%
	quant-recover (recovery)	6,244	52	49	2,725,759,136	1.000	0.992	0.992	1.000	22	99.6%	99.2%
	quant-cramstyle (m4)	5,999	144	294	2,725,759,044	1.000	0.977	0.953	1.000	96	98.4%	95.3%
	bestcomp (m5)	5,997	147	296	2,725,759,041	1.000	0.976	0.953	1.000	97	98.4%	95.3%
	cram	5,999	144	294	2,725,759,044	1.000	0.977	0.953	1.000	96	98.4%	95.3%
Reseq/hm (human)	noquant (lossless)	327	-	-	3,101,804,412	1.000	1.000	1.000	1.000	0	100.0%	100.0%
	dropall (drop all)	284	26	43	3,101,804,386	1.000	0.916	0.869	1.000	5	98.2%	86.9%
	quant-bestacc1 (m2)	324	1	3	3,101,804,411	1.000	0.997	0.991	1.000	3	99.1%	99.1%
	quant-bestacc2 (m1)	326	1	1	3,101,804,411	1.000	0.997	0.997	1.000	0	100.0%	99.7%
	quant-bestacc3 (m3)	322	1	5	3,101,804,411	1.000	0.997	0.985	1.000	3	99.1%	98.5%
	quant-recover (recovery)	326	1	1	3,101,804,411	1.000	0.997	0.997	1.000	1	99.7%	99.7%
	quant-cramstyle (m4)	322	3	5	3,101,804,409	1.000	0.991	0.985	1.000	4	98.8%	98.5%
	bestcomp (m5)	322	8	5	3,101,804,404	1.000	0.976	0.985	1.000	4	98.8%	98.5%
	cram	322	3	5	3,101,804,409	1.000	0.991	0.985	1.000	4	98.8%	98.5%
RNA-seq (e.coli)	noquant (lossless)	11	-	-	4,639,664	1.000	1.000	1.000	1.000	0	100.0%	100.0%
	dropall (drop all)	8	34	3	4,639,630	1.000	0.190	0.727	1.000	0	100.0%	72.7%
	quant-bestacc1 (m2)	10	-	1	4,639,664	1.000	1.000	0.909	1.000	0	100.0%	90.9%
	quant-bestacc2 (m1)	11	-	-	4,639,664	1.000	1.000	1.000	1.000	0	100.0%	100.0%
	quant-bestacc3 (m3)	10	2	1	4,639,662	1.000	0.833	0.909	1.000	0	100.0%	90.9%
	quant-recover (recovery)	11	-	-	4,639,664	1.000	1.000	1.000	1.000	0	100.0%	100.0%
	quant-cramstyle (m4)	9	4	2	4,639,660	1.000	0.692	0.818	1.000	0	100.0%	81.8%
	bestcomp (m5)	9	4	2	4,639,660	1.000	0.692	0.818	1.000	0	100.0%	81.8%
	cram	9	4	2	4,639,660	1.000	0.692	0.818	1.000	0	100.0%	81.8%
Reseq (e.coli,PE)	noquant (lossless)	2	-	-	4,639,673	1.000	1.000	1.000	1.000	0	100.0%	100.0%
	dropall (drop all)	2	5	-	4,639,668	1.000	0.286	1.000	1.000	0	100.0%	100.0%
	quant-bestacc1 (m2)	2	-	-	4,639,673	1.000	1.000	1.000	1.000	0	100.0%	100.0%
	quant-bestacc2 (m1)	2	-	-	4,639,673	1.000	1.000	1.000	1.000	0	100.0%	100.0%
	quant-bestacc3 (m3)	2	-	-	4,639,673	1.000	1.000	1.000	1.000	0	100.0%	100.0%
	quant-recover (recovery)	2	-	-	4,639,673	1.000	1.000	1.000	1.000	0	100.0%	100.0%
	quant-cramstyle (m4)	2	-	-	4,639,673	1.000	1.000	1.000	1.000	0	100.0%	100.0%
	bestcomp (m5)	2	-	-	4,639,673	1.000	1.000	1.000	1.000	0	100.0%	100.0%
	cram	2	-	-	4,639,673	1.000	1.000	1.000	1.000	0	100.0%	100.0%
Reseq (e. coli)	noquant (lossless)	72	-	-	4,639,603	1.000	1.000	1.000	1.000	0	100.0%	100.0%
	dropall (drop all)	70	11	2	4,639,592	1.000	0.864	0.972	1.000	0	100.0%	97.2%
	quant-bestacc1 (m2)	72	1	-	4,639,602	1.000	0.986	1.000	1.000	0	100.0%	100.0%
	quant-bestacc2 (m1)	72	-	-	4,639,603	1.000	1.000	1.000	1.000	0	100.0%	100.0%
	quant-bestacc3 (m3)	72	3	-	4,639,600	1.000	0.960	1.000	1.000	0	100.0%	100.0%
	quant-recover (recovery)	72	1	-	4,639,602	1.000	0.986	1.000	1.000	0	100.0%	100.0%
	quant-cramstyle (m4)	70	4	2	4,639,599	1.000	0.946	0.972	1.000	0	100.0%	97.2%
	bestcomp (m5)	70	5	2	4,639,598	1.000	0.933	0.972	1.000	0	100.0%	97.2%
	cram	70	4	2	4,639,599	1.000	0.946	0.972	1.000	0	100.0%	97.2%
Reseq (a. thaliana)	noquant (lossless)	37,104	-	-	119,630,646	1.000	1.000	1.000	1.000	0	100.0%	100.0%
	dropall (drop all)	34,303	3,352	2,801	119,627,294	1.000	0.911	0.925	1.000	449	98.7%	92.5%
	quant-bestacc1 (m2)	36,650	159	454	119,630,487	1.000	0.996	0.988	1.000	93	99.7%	98.8%
	quant-bestacc2 (m1)	36,995	73	109	119,630,573	1.000	0.998	0.997	1.000	9	100.0%	99.7%
	quant-bestacc3 (m3)	36,567	309	537	119,630,337	1.000	0.992	0.986	1.000	107	99.7%	98.6%
	quant-recover (recovery)	36,957	122	147	119,630,524	1.000	0.997	0.996	1.000	26	99.9%	99.6%
	quant-cramstyle (m4)	36,371	323	733	119,630,323	1.000	0.991	0.980	1.000	483	98.7%	98.0%
	bestcomp (m5)	36,372	390	732	119,630,256	1.000	0.989	0.980	1.000	484	98.7%	98.0%
	cram	36,371	323	733	119,630,323	1.000	0.991	0.980	1.000	483	98.7%	98.0%
Reseq/chr20 (human)	noquant (lossless)	90,969	-	-	3,101,713,770	1.000	1.000	1.000	1.000	0	100.0%	100.0%
	dropall (drop all)	86,295	2,648	4,674	3,101,711,122	1.000	0.970	0.949	1.000	186	99.8%	94.9%
	quant-bestacc1 (m2)	90,726	273	243	3,101,713,497	1.000	0.997	0.997	1.000	57	99.9%	99.7%
	quant-bestacc2 (m1)	90,862	53	107	3,101,713,717	1.000	0.999	0.999	1.000	26	100.0%	99.9%
	quant-bestacc3 (m3)	90,645	381	324	3,101,713,389	1.000	0.996	0.996	1.000	89	99.9%	99.6%
	quant-recover (recovery)	90,831	70	138	3,101,713,700	1.000	0.999	0.998	1.000	38	100.0%	99.8%
	quant-cramstyle (m4)	90,370	368	599	3,101,713,402	1.000	0.996	0.993	1.000	175	99.8%	99.3%
	bestcomp (m5)	90,366	389	603	3,101,713,381	1.000	0.996	0.993	1.000	176	99.8%	99.3%
	cram	90,379	357	590	3,101,713,413	1.000	0.996	0.994	1.000	177	99.8%	99.4%

Table A.4: Variant recovery (SAMTOOLS)

Data set	Q-value quantization strategy	prec	sens	%	%recovered vars
Exome-seq (human)	noquant (lossless)	1.000	1.000	1.000	100%
	dropall (drop all)	0.539	0.966	0.990	97%
	quant-bestacc1 (m2)	0.984	0.994	0.998	99%
	quant-bestacc2 (m1)	0.995	0.996	0.999	100%
	quant-bestacc3 (m3)	0.978	0.993	0.997	99%
	quant-recover (recovery)	1.000	0.993	0.998	99%
	quant-cramstyle (m4)	0.978	0.983	0.990	98%
	bestcomp (m5)	0.974	0.983	0.990	98%
	cram	0.978	0.983	0.990	98%
ChiP-seq (mouse)	noquant (lossless)	1.000	1.000	1.000	100%
	dropall (drop all)	0.773	0.858	0.984	86%
	quant-bestacc1 (m2)	0.988	0.987	0.993	99%
	quant-bestacc2 (m1)	0.995	0.995	0.999	99%
	quant-bestacc3 (m3)	0.979	0.982	0.993	98%
	quant-recover (recovery)	0.992	0.994	0.997	99%
	quant-cramstyle (m4)	0.980	0.969	0.984	97%
	bestcomp (m5)	0.957	0.967	0.984	97%
	cram	0.980	0.969	0.984	97%
Reseq/hm (human)	noquant (lossless)	1.000	1.000	1.000	100%
	dropall (drop all)	0.838	0.908	0.986	91%
	quant-bestacc1 (m2)	0.998	0.994	0.992	99%
	quant-bestacc2 (m1)	0.998	0.997	1.000	100%
	quant-bestacc3 (m3)	0.997	0.991	0.992	99%
	quant-recover (recovery)	0.998	0.997	0.997	100%
	quant-cramstyle (m4)	0.995	0.987	0.989	99%
	bestcomp (m5)	0.965	0.976	0.988	98%
	cram	0.995	0.987	0.989	99%
RNA-seq (e.coli)	noquant (lossless)	1.000	1.000	1.000	100%
	dropall (drop all)	0.326	0.784	0.996	78%
	quant-bestacc1 (m2)	0.993	0.948	1.000	95%
	quant-bestacc2 (m1)	0.997	0.990	1.000	99%
	quant-bestacc3 (m3)	0.910	0.948	1.000	95%
	quant-recover (recovery)	0.997	0.997	1.000	100%
	quant-cramstyle (m4)	0.846	0.886	1.000	89%
	bestcomp (m5)	0.843	0.879	1.000	88%
	cram	0.846	0.886	1.000	89%
Reseq (e.coli,PE)	noquant (lossless)	1.000	1.000	1.000	100%
	dropall (drop all)	0.153	0.984	1.000	98%
	quant-bestacc1 (m2)	0.970	1.000	1.000	100%
	quant-bestacc2 (m1)	1.000	1.000	1.000	100%
	quant-bestacc3 (m3)	0.824	1.000	1.000	100%
	quant-recover (recovery)	0.980	1.000	1.000	100%
	quant-cramstyle (m4)	0.869	0.904	1.000	90%
	bestcomp (m5)	0.869	0.904	1.000	90%
	cram	0.888	0.904	1.000	90%
Reseq (e. coli)	noquant (lossless)	1.000	1.000	1.000	100%
	dropall (drop all)	0.467	0.986	1.000	99%
	quant-bestacc1 (m2)	0.987	1.000	1.000	100%
	quant-bestacc2 (m1)	0.994	1.000	1.000	100%
	quant-bestacc3 (m3)	0.955	0.994	1.000	99%
	quant-recover (recovery)	0.980	1.000	1.000	100%
	quant-cramstyle (m4)	0.948	0.973	1.000	97%
	bestcomp (m5)	0.941	0.973	1.000	97%
	cram	0.948	0.973	1.000	97%
Reseq (a. thaliana)	noquant (lossless)	1.000	1.000	1.000	100%
	dropall (drop all)	0.921	0.958	0.991	96%
	quant-bestacc1 (m2)	0.997	0.994	0.998	99%
	quant-bestacc2 (m1)	0.999	0.998	1.000	100%
	quant-bestacc3 (m3)	0.994	0.992	0.998	99%
	quant-recover (recovery)	0.998	0.997	1.000	100%
	quant-cramstyle (m4)	0.994	0.989	0.990	99%
	bestcomp (m5)	0.984	0.987	0.990	99%
	cram	0.994	0.989	0.990	99%
Reseq/chr20 (human)	noquant (lossless)	1.000	1.000	1.000	100%
	dropall (drop all)	0.911	0.972	0.997	97%
	quant-bestacc1 (m2)	0.998	0.999	0.999	100%
	quant-bestacc2 (m1)	1.000	0.999	1.000	100%
	quant-bestacc3 (m3)	0.996	0.998	0.999	100%
	quant-recover (recovery)	0.999	0.999	1.000	100%
	quant-cramstyle (m4)	0.998	0.994	0.997	99%
	bestcomp (m5)	0.997	0.994	0.997	99%
	cram	0.998	0.994	0.997	99%

Table A.5: Variant recovery (averaged)

Sample	Evaluation mode	mapped reads	Bases and qvalues	BAM [byte]	CRAM [byte]	Comp. ratio	Comp [s]	Decomp [s]	comp [min]	decomp [min]
Exome-seq (human)	lossless	3,239,217	323,921,700	199,019,319	129,302,897	0.65	67.52	166.60	1.1	2.8
	lossy	3,239,217	323,921,700	199,019,319	8,993,932	0.05	44.44	91.57	0.7	1.5
ChiP-seq (mouse)	lossless	13,824,441	497,679,876	637,462,912	282,756,168	0.44	200.71	468.84	3.3	7.8
	lossy	13,824,441	497,679,876	637,462,912	55,625,624	0.09	180.33	360.41	3.0	6.0
Reseq/hapma (human)	lossless	487,201	62,435,260	53,036,863	24,054,223	0.45	27.64	58.70	0.5	1.0
	lossy	487,201	62,435,260	53,036,863	4,342,572	0.08	27.45	50.81	0.5	0.8
RNA-seq (e.coli)	lossless	6,927,728	249,398,208	177,499,665	105,288,858	0.59	78.80	183.76	1.3	3.1
	lossy	6,927,728	249,398,208	177,499,665	4,979,867	0.03	61.80	124.19	1.0	2.1
Reseq (e.coli,PE)	lossless	44,938,891	4,538,790,538	2,945,453,583	2,308,990,877	0.78	1,212.96	2,976.40	20.2	49.6
	lossy	44,938,891	4,538,790,538	2,945,453,583	647,050,983	0.22	975.03	2,036.86	16.3	33.9
Reseq (e. coli)	lossless	19,379,287	697,654,332	654,501,526	339,334,229	0.52	218.61	521.37	3.6	8.7
	lossy	19,379,287	697,654,332	654,501,526	19,259,853	0.03	167.68	349.53	2.8	5.8
Reseq (a. thaliana)	lossless	11,651,942	419,469,912	503,542,318	235,324,311	0.47	159.18	376.83	2.7	6.3
	lossy	11,651,942	419,469,912	503,542,318	48,464,278	0.10	128.03	267.89	2.1	4.5
Reseq/chr20 (human)	lossless	50,663,069	5,116,969,969	5,867,598,471	4,055,011,808	0.69	1,572.47	3,894.58	26.2	64.9
	lossy	50,663,069	5,116,969,969	5,867,598,471	2,068,227,640	0.35	1,624.37	3,339.40	27.1	55.7

Table A.6: Cram evaluation data. The options for these modes were set as written above in Section A.7.

Sample	mapped reads	Bases and qvalues	BAM [byte]	GOBY [byte]	Comp. ratio	comp [min]	decomp [min]
Exome-seq (human)	3,239,217	323,921,700	199,019,319	149,878,386	0.75	3.13	5.43
ChiP-seq (mouse)	13,824,441	497,679,876	637,462,912	420,719,703	0.66	8.72	16.31
Reseq/hm (human)	487,201	62,435,260	53,036,863	-	-	-	-
RNA-seq (e.coli)	6,927,728	249,398,208	177,499,665	146,982,735	0.83	3.98	7.14
Reseq (e.coli,PE)	44,938,891	4,538,790,538	2,945,453,583	2,507,599,991	0.85	135.03	169.49
Reseq (e. coli)	19,379,287	697,654,332	654,501,526	491,915,033	0.75	11.21	20.10
Reseq (a. thaliana)	11,651,942	419,469,912	503,542,318	361,810,425	0.72	7.53	13.39
Reseq/chr20 (human)	50,663,069	5,116,969,969	5,867,598,471	-	-	-	-

Table A.7: Goby evaluation data. Note that the data sets Reseq/hm (human) and Reseq/chr20 (human) could not be properly compressed/decompressed with Goby as explained in above in Section A.7.3.

Bibliography

[AJL+02] Bruce Alberts, Alexander Johnson, Julian Lewis, Martin Raff, Keith Roberts, and Peter Walter. *Molecular biology of the cell*. Garland Science Taylor & Francis Group, 4th ed., 2002. (p. 3)

[AMA+10] Andrew Adey, Hilary G. Morrison, Asan, Xu Xun, Jacob O. Kitzman, Emily H. Turner, Bethany Stackhouse, Alexandra P. MacKenzie, Nicholas C. Caruccio, Xiuqing Zhang, and Jay Shendure. Rapid, low-input, low-bias construction of shotgun fragment libraries by high-density in vitro transposition. *Genome Biol*, 11(12):R119, 2010. (p. 5)

[AWB+12] André Altmann, Peter Weber, Daniel Bader, Michael Preuß, Elisabeth B. Binder, and Bertram Müller-Myhsok. A beginners guide to SNP calling from high-throughput DNA-sequencing data. *Hum Genet*, 131(10):1541–1554, 2012. (p. 35)

[BBN+11] Vishal Bhola, Ajit S. Bopardikar, Rangavittal Narayanan, Kyusang Lee, and TaeJin Ahn. No-reference compression of genomic data stored in FASTQ format. In *IEEE International Conference on Bioinformatics and Biomedicine*, pp. 147–150, Los Alamitos, CA, USA, 2011. IEEE Computer Society. (p. 19, 20, 22)

[BEDE04] Petros Boufounos, Sameh El-Difrawy, and Dan Ehrlich. Basecalling using hidden markov models. *Journal of the Franklin Institute*, 341(1-2):23 – 36, 2004. (p. 14)

[BKJ06] Lanrong Bi, Dae Hyun Kim, and Jingyue Ju. Design and synthesis of a chemically cleavable fluorescent nucleotide, 3'-O-allyl-dGTP-allyl-bodipy-FL-510, as a reversible terminator for DNA sequencing by synthesis. *Journal of the American Chemical Society*, 128(8):2542–2543, 2006. (p. 9)

[BLF05] Behshad Behzadi and Fabrice Le Fessant. DNA compression challenge revisited: A dynamic programming approach. In Alberto Apostolico, Maxime Crochemore, and Kunsoo Park, editors, *Combinatorial Pattern Matching*, volume 3537 of *Lecture Notes in Computer Science*, pp. 85–96. Springer Berlin / Heidelberg, 2005. (p. 19)

[BWB09] Marty C. Brandon, Douglas C. Wallace, and Pierre Baldi. Data structures and compression algorithms for genomic sequence data. *Bioinformatics*, 25(14):1731–1738, 2009. (p. 20, 22)

[CBJR12] Anthony J. Cox, Markus J. Bauer, Tobias Jakobi, and Giovanna Rosone. Large-scale compression of genomic sequence databases with the burrows-wheeler transform. *Bioinformatics*, 28(11):1415–1419, 2012. (p. 33)

[CDAM07] Minh Duc Cao, Trevor I. Dix, Lloyd Allison, and Chris Mears. A simple statistical algorithm for biological sequence compression. In *Data Compression Conference (DCC '07)*, pp. 43–52, 2007. (p. 19, 22)

[CFG+10] Peter J A. Cock, Christopher J. Fields, Naohisa Goto, Michael L. Heuer, and Peter M. Rice. The sanger FASTQ file format for sequences with quality scores, and the solexa/illumina FASTQ variants. *Nucleic Acids Res*, 38(6):1767–1771, 2010. (p. 17, 20, 26)

[CGS09] Nicholas Caruccio, Haiying Grunenwald, and Fraz Syed. Nextera™ technology for NGS DNA library preparation: simultaneous fragmentation and tagging by in vitro transposition. *Nature Methods*, 16(3), 2009. (p. 5)

[CKL00] Xin Chen, Sam Kwong, and Mingr Li. A compression algorithm for DNA sequences and its applications in genome comparison. In *Proceedings of the fourth annual international conference on computational molecular biology (RECOMB '00)*, p. 107, 2000. (p. 19)

[CLLX09] Scott Christley, Yiming Lu, Chen Li, and Xiaohui Xie. Human genomes as email attachments. *Bioinformatics*, 25(2):274–275, 2009. (p. 20, 22)

[CLMT02] Xin Chen, Ming Li, Bin Ma, and John Tromp. DNACompress: fast and effective DNA sequence compression. *Bioinformatics*, 18(12):1696–1698, 2002. (p. 19)

[Col10] Francis Collins. Has the revolution arrived? *Nature*, 464(7289):674–675, 2010. (p. 1)

[CWJ+09] James Clarke, Hai-Chen Wu, Lakmal Jayasinghe, Alpesh Patel, Stuart Reid, and Hagan Bayley. Continuous base identification for single-molecule nanopore DNA sequencing. *Nature Nanotechnology*, 4:265–270, 2009. (p. 12)

[DD87] J. J. Doyle and J. L. Doyle. A rapid DNA isolation procedure for small quantities of fresh leaf tissue. *Phytochemical Bulletin*, 19:11–15, 1987. (p. 3)

[DG11] Sebastian Deorowicz and Szymon Grabowski. Compression of genomic sequences in FASTQ format. *Bioinformatics*, 27:860–862, 2011. (p. 19, 22, 26)

[DHH10] Michel Delseny, Bin Han, and Yue Ie Hsing. High throughput DNA sequencing: The new sequencing revolution. *Plant Science*, 179(5):407 – 422, 2010. (p. 12, 13)

[DNGL01] Frank B. Dean, John R. Nelson, Theresa L. Giesler, and Roger S. Lasken. Rapid amplification of plasmid and phage DNA using Phi 29 DNA polymerase and multiply-primed rolling circle amplification. *Genome Res*, 11(6):1095–1099, 2001. (p. 11)

[DRC+10] Kenny Daily, Paul Rigor, Scott Christley, Xiaohui Xie, and Pierre Baldi. Data structures and compression algorithms for high-throughput sequencing technologies. *BMC Bioinformatics*, 11(1):514+, 2010. (p. 19, 20, 22, 29)

[DYT+03] Devin Dressman, Hai Yan, Giovanni Traverso, Kenneth W. Kinzler, and Bert Vogelstein. Transforming single DNA molecules into fluorescent magnetic particles for detection and enumeration of genetic variations. *Proc Natl Acad Sci U S A*, 100(15):8817–8822, 2003. (p. 9)

[Ede80] Herbert Edelsbrunner. Dynamic data structures for orthogonal intersection queries. Technical Report F59, Technical University Graz, 1980. (p. 35)

[EFG+09] John Eid, Adrian Fehr, Jeremy Gray, Khai Luong, John Lyle, Geoff Otto, Paul Peluso, David Rank, Primo Baybayan, Brad Bettman, Arkadiusz Bibillo, Keith Bjornson, Bidhan Chaudhuri, Frederick Christians, Ronald Cicero, Sonya Clark, Ravindra Dalal, Alex deWinter, John Dixon, Mathieu Foquet, Alfred Gaertner, Paul Hardenbol, Cheryl Heiner, Kevin Hester, David Holden, Gregory Kearns, Xiangxu Kong, Ronald Kuse, Yves Lacroix, Steven Lin, Paul

Lundquist, Congcong Ma, Patrick Marks, Mark Maxham, Devon Murphy, Insil Park, Thang Pham, Michael Phillips, Joy Roy, Robert Sebra, Gene Shen, Jon Sorenson, Austin Tomaney, Kevin Travers, Mark Trulson, John Vieceli, Jeffrey Wegener, Dawn Wu, Alicia Yang, Denis Zaccarin, Peter Zhao, Frank Zhong, Jonas Korlach, and Stephen Turner. Real-time DNA sequencing from single polymerase molecules. *Science*, 323(5910):133–138, 2009. (p. 12)

[EHWG98] Brent Ewing, LaDeana Hillier, Michael C. Wendl, and Phil Green. Base-calling of automated sequencer traces using Phred. phred. I. accuracy assessment. *Genome Res*, 8(3):175–185, 1998. (p. 14)

[FAB+08] Paul Flicek, Bronwen L. Aken, Kathryn Beal, Benoit Ballester, Mario Cáccamo, Yuan Chen, Laura Clarke, Guy Coates, Fiona Cunningham, Tim Cutts, Thomas A. Down, S. C. Dyer, T. Eyre, Stephen Fitzgerald, Julio Fernandez-Banet, Stefan Gräf, Syed Haider, Martin Hammond, Richard C. G. Holland, Kevin L. Howe, Kerstin Howe, Nathan Johnson, Andrew M. Jenkinson, Andreas Kähäri, Damian Keefe, Felix Kokocinski, Eugene Kulesha, Daniel Lawson, Ian Longden, Karine Megy, Patrick Meidl, Bert Overduin, Anne Parker, Bethan Pritchard, Andreas Prlic, S. Rice, Daniel Rios, Michael Schuster, I. Sealy, Guy Slater, Damian Smedley, Giulietta Spudich, S. Trevanion, Albert J. Vilella, Jan Vogel, Simon White, M. Wood, Ewan Birney, Tony Cox, Val Curwen, Richard Durbin, Xosé M. Fernández-Suarez, Javier Herrero, Tim J. P. Hubbard, Arek Kasprzyk, Glenn Proctor, James Smith, Abel Ureta-Vidal, and Stephen M. J. Searle. Ensembl 2008. *Nucleic Acids Res*, 36(Database issue):D707–D714, 2008. (p. 33)

[FRW+06] Milan Fedurco, Anthony Romieu, Scott Williams, Isabelle Lawrence, and Gerardo Turcatti. BTA, a novel reagent for DNA attachment on glass and efficient generation of solid-phase amplified DNA colonies. *Nucleic Acids Res*, 34(3):e22, 2006. (p. 9)

[FWL+10] Benjamin A. Flusberg, Dale R. Webster, Jessica H. Lee, Kevin J. Travers, Eric C. Olivares, Tyson A. Clark, Jonas Korlach, and Stephen W. Turner. Direct detection of DNA methylation during single-molecule, real-time sequencing. *Nature Methods*, 7(6):461–465, 2010. (p. 12)

[FX95] Andrew Fire and Si-Qun Xu. Rolling replication of short DNA circles. *Proc Natl Acad Sci U S A*, 92(10):4641–4645, May 1995. (p. 11)

[G+04] Curtis Gehman et al. Longer reads with the KB Basecaller. Technical report, Applied Biosystems, 2004. (p. 14)

[Gre08] Phil Green. Calf (compact alignment format), version 0.081113. Technical report, University of Washington, 2008. (p. 18, 33)

[GSU09] Raffaele Giancarlo, Davide Scaturro, and Filippo Utro. Textual data compression in computational biology: a synopsis. *Bioinformatics*, 25(13):1575–1586, 2009. (p. 19, 24)

[GWL+05] Anthony J. F. Griffiths, Susan R. Wessler, Richard C. Lewontin, William M. Gelbart, David T. Suzuki, and Jeffrey H. Miller. *Introduction to Genetic Analysis*. W. H. Freeman, 8th ed., 2005. (p. 5)

[HHR10] R. David Hawkins, Gary C. Hon, and Bing Ren. Next-generation genomics: an integrative approach. *Nat Rev Genet*, 11:476–486, 2010. (p. 1)

[HSMS99] Anson Hatch, Takeshi Sano, John Misasi, and Cassandra L. Smith. Rolling circle amplification of DNA immobilized on solid surfaces and its application to multiplex mutation detection. *Genet Anal*, 15(2):35–40, Apr 1999. (p. 11)

[HYFLCB11] Markus Hsi-Yang Fritz, Rasko Leinonen, Guy Cochrane, and Ewan Birney. Efficient storage of high throughput DNA sequencing data using reference-based compression. *Genome Res*, 21(5):734–740, 2011. (p. 19, 20, 22, 26, 28, 34)

[Ill11] Illumina. Casava v1.8.2 user guide. Technical report, Illumina, 2011. (p. 17)

[Int05] International HapMap Consortium. A haplotype map of the human genome. *Nature*, 437(7063):1299–1320, 2005. (p. 15)

[JPAH11] Young Jun Jeon, Sang Hyun Park, Sung Min Ahn, and Hee Joung Hwang. Solidzipper: A high speed encoding method for the next-generation sequencing data. *Evolutionary Bioinformatics*, 7:1–6, 2011. (p. 19)

[Kah11] Scott D. Kahn. On the future of genomic data. *Science*, 331(6018):728–729, 2011. (p. 1, 22)

[Kie04] Aaron Kiely. Selecting the golomb parameter in rice coding. Technical report, Jet Propulsion Laboratory, 2004. (p. 29)

[KK10] Martin Kircher and Janet Kelso. High-throughput DNA sequencing - concepts and limitations. *BioEssays*, 32(6):524–536, 2010. (p. 1, 8, 9, 10, 13, 15)

[KSK+11] Christos Kozanitis, Chris Saunders, Semyon Kruglyak, Vineet Bafna, and George Varghese. Compressing genomic sequence fragments using SlimGene. *J Comput Biol*, 18(3):401–413, 2011. (p. 19, 20, 22, 26, 33, 34)

[KSLI12] Yuichi Kodama, Martin Shumway, Rasko Leinonen, and International Nucleotide Sequence Database Collaboration. The Sequence Read Archive: explosive growth of sequencing data. *Nucleic Acids Res*, 40(Database issue):D54–D56, 2012. (p. 22, 34)

[LBB12] Po-Ru Loh, Michael Baym, and Bonnie Berger. Compressive genomics. *Nat Biotech*, 30:627–630, 2012. (p. 1)

[LD10] Heng Li and Richard Durbin. Fast and accurate long-read alignment with burrows-wheeler transform. *Bioinformatics*, 26(5):589–595, 2010. (p. 22, 39)

[LH10] Heng Li and Nils Homer. A survey of sequence alignment algorithms for next-generation sequencing. *Brief Bioinform*, 11(5):473–483, 2010. (p. 18)

[LHW+09] Heng Li, Bob Handsaker, Alec Wysoker, Tim Fennell, Jue Ruan, Nils Homer, Gabor Marth, Goncalo Abecasis, and Richard Durbin. The sequence alignment/map format and samtools. *Bioinformatics*, 25(16):2078–2079, 2009. (p. 18, 22, 28, 39)

[LMD+12] Nicholas J. Loman, Raju V. Misra, Timothy J. Dallman, Chrystala Constantinidou, Saheer E. Gharbia, John Wain, and Mark J. Pallen. Performance comparison of benchtop high-throughput sequencing platforms. *Nat Biotechnol*, 30(5):434–439, 2012. (p. 1, 13, 15)

[LWA07] Kuo-ching Liang, Xiaodong Wang, and Dimitris Anastassiou. Bayesian basecalling for DNA sequence analysis using hidden markov models. *IEEE/ACM Trans. Comput. Biol. Bioinformatics*, 4(3):430–440, 2007. (p. 14)

[LZL+91] Peter W. Laird, Alice Zijderveld, Koert Linders, Michael A. Rudnicki, Rudolf Jaenisch, and Anton Berns. Simplified mammalian DNA isolation procedure. *Nucleic Acids Res*, 19(15):4293, 1991. (p. 3)

[MBG+10] Alberto Magi, Matteo Benelli, Alessia Gozzini, Francesca Girolami, Francesca Torricelli, and Maria Luisa Brandi. Bioinformatics for next generation sequencing data. *Genes*, 1(2):294–307, 2010. (p. 9, 11, 13, 18)

[McC80] Edward M. McCreight. Efficient algorithms for enumerating intersecting intervals and rectangles. Technical Report CSL-80-9, Xerox PARC, 1980. (p. 35)

[MDP88] S. A. Miller, D. D. Dykes, and H. F. Polesky. A simple salting out procedure for extracting DNA from human nucleated cells. *Nucleic Acids Res*, 16(3):1215, 1988. (p. 3)

[MEA+05] Marcel Margulies, Michael Egholm, William E. Altman, Said Attiya, Joel S. Bader, Lisa A. Bemben, Jan Berka, Michael S. Braverman, Yi-Ju Chen, Zhoutao Chen, Scott B. Dewell, Lei Du, Joseph M. Fierro, Xavier V. Gomes, Brian C. Godwin, Wen He, Scott Helgesen, Chun H. Ho, Gerard P. Irzyk, Szilveszter C. Jando, Maria L. I. Alenquer, Thomas P. Jarvie, Kshama B. Jirage, Jong-Bum Kim, James R. Knight, Janna R. Lanza, John H. Leamon, Steven M. Lefkowitz, Ming Lei, Jing Li, Kenton L. Lohman, Hong Lu, Vinod B. Makhijani, Keith E. McDade, Michael P. McKenna, Eugene W. Myers, Elizabeth Nickerson, John R. Nobile, Ramona Plant, Bernard P. Puc, Michael T. Ronan, George T. Roth, Gary J. Sarkis, Jan F. Simons, John W. Simpson, Maithreyan Srinivasan, Karrie R. Tartaro, Alexander Tomasz, Kari A. Vogt, Greg A. Volkmer, Shally H. Wang, Yong Wang, Michael P. Weiner, Pengguang Yu, Richard F. Begley, and Jonathan M. Rothberg. Genome sequencing in microfabricated high-density picolitre reactors. *Nature*, 437(7057):376–380, 2005. (p. 14)

[MHB+10] Aaron McKenna, Matthew Hanna, Eric Banks, Andrey Sivachenko, Kristian Cibulskis, Andrew Kernytsky, Kiran Garimella, David Altshuler, Stacey Gabriel, Mark Daly, and Mark A. DePristo. The Genome Analysis Toolkit: a mapreduce framework for analyzing next-generation dna sequencing data. *Genome Res*, 20(9):1297–1303, 2010. (p. 22, 28, 40)

[Pos10] Henk W.Ch. Postma. Rapid sequencing of individual DNA molecules in Graphene nanogaps. *Nano Letters*, 10(2):420–425, 2010. (p. 13)

[PPG11] Armando J. Pinho, Diogo Pratas, and Sara P. Garcia. GReEn: a tool for efficient compression of genome resequencing data. *Nucleic Acids Res*, 40(4):e27, 2011. (p. 22)

[PvH12] Niko Popitsch and Arndt von Haeseler. NGC: Lossless and lossy compression of aligned high-throughput sequencing data. *Nucl Acids Res*, 2012. (p. 21)

[Ric98] Peter Richterich. Estimation of errors in "raw" DNA sequences: A validation study. *Genome Res*, 8(3):251–259, 1998. (p. 14)

[RK08] Nicole Rusk and Veronique Kiermer. Primer: Sequencing - the next generation. *Nat Meth*, 5(1):15+, 2008. (p. 11)

[Ron01] Mostafa Ronaghi. Pyrosequencing sheds light on DNA sequencing. *Genome Res*, 11(1):3–11, 2001. (p. 9, 10)

[Rus11] Nicole Rusk. Torrents of sequence. *Nat Meth*, 8(1):44–44, 2011. (p. 12)

[Sam05] Hanan Samet. *Foundations of Multidimensional and Metric Data Structures (The Morgan Kaufmann Series in Computer Graphics and Geometric Modeling)*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 2005. (p. 35)

[SJ08] Jay Shendure and Hanlee Ji. Next-generation DNA sequencing. *Nature Biotechnology*, 26(10):1135–1145, 2008. (p. 6, 9, 15)

[SMZ+12] Sophie Schbath, Véronique Martin, Matthias Zytnicki, Julien Fayolle, Valentin Loux, and Jean-François Gibrat. Mapping reads on a genomic sequence: an algorithmic overview and a practical comparative analysis. *J Comput Biol*, 19(6):796–813, Jun 2012. (p. 18)

[SNC77] Frederick Sanger, Steve Nicklen, and Alan R. Coulsen. DNA sequencing with chain-terminating inhibitors. *Proc. Natl. Acad. Sci. USA*, 74:5463–5467, 1977. (p. 6)

[SPR+05] Jay Shendure, Gregory J. Porreca, Nikos B. Reppas, Xiaoxia Lin, John P. McCutcheon, Abraham M. Rosenbaum, Michael D. Wang, Kun Zhang, Robi D. Mitra, and George M. Church. Accurate multiplex polony sequencing of an evolved bacterial genome. *Science*, 309(5741):1728–1732, 2005. (p. 6, 8, 11)

[SR01] Joseph Sambrook and David W. Russell. *Molecular Cloning: A Laboratory Manual*. Cold Spring Harbor Laboratory Press, 3rd ed., January 2001. (p. 3)

[STZH11] Muhammad Nazmus Sakib, Jijun Tang, W. Jim Zheng, and Chin-Tser Huang. Improving transmission efficiency of large sequence alignment/map (SAM) files. *PLoS ONE*, 6(12):e28251, 12 2011. (p. 19, 22, 26, 29, 34)

[The11] The SAM Format Specification Working Group. SAM Format Specification (v1.4-r962). Technical report, April 17 2011. (p. 18, 19, 23, 24)

[TLS10] Waibhav Tembe, James Lowey, and Edward Suh. G-SQZ: compact encoding of genomic sequence and quality data. *Bioinformatics*, 26(17):2192–2194, 2010. (p. 19, 22)

[TMF09] Tracy Tucker, Marco Marra, and Jan M. Friedman. Massively parallel sequencing: the next big thing in genetic medicine. *Am J Hum Genet*, 85(2):142–154, 2009. (p. 35)

[TRFT08] Gerardo Turcatti, Anthony Romieu, Milan Fedurco, and Ana-Paula Tairi. A new class of cleavable fluorescent nucleotides: synthesis and optimization as reversible terminators for DNA sequencing by synthesis. *Nucleic Acids Res*, 36(4):e25, 2008. (p. 9, 10)

[TRM12] Helga Thorvaldsdóttir, James T. Robinson, and Jill P. Mesirov. Integrative genomics viewer (IGV): high-performance genomics data visualization and exploration. *Brief Bioinform*, 2012. (p. 27)

[WAA11] Raymond Wan, Vo Ngoc Anh, and Kiyoshi Asai. Transformations for the compression of FASTQ quality scores of next generation sequencing data. *Bioinformatics*, 28(5):628–635, 2011. (p. 19, 20, 26, 34)

[WNC87] Ian H. Witten, Radford M. Neal, and John G. Cleary. Arithmetic coding for data compression. *Commun. ACM*, 30(6):520–540, 1987. (p. 26)

[WPM+06] Richard Williams, Sergio G. Peisajovich, Oliver J. Miller, Shlomo Magdassi, Dan S. Tawfik, and Andrew D. Griffiths. Amplification of complex gene libraries by emulsion PCR. *Nature Methods*, 3(7):545–550, 2006. (p. 9)

[WRB+12] Sebastian Wandelt, Astrid Rheinländer, Marc Bux, Lisa Thalheim, Berit Haldemann, and Ulf Leser. Data management challenges in next generation sequencing. *Datenbank-Spektrum*, pp. 1–11, 2012. (p. 1)

[WSH+10] Emma V. B. Wallace, David Stoddart, Andrew J. Heron, Ellina Mikhailova, Giovanni Maglia, Timothy J. Donohoe, and Hagan Bayley. Identification of epigenetic DNA modifications with a protein nanopore. *Chem. Commun.*, 46:8195–8197, 2010. (p. 12)

[WZ11] Congmao Wang and Dabing Zhang. A novel compression tool for efficient storage of genome resequencing data. *Nucleic Acids Res*, 39(7):e45, 2011. (p. 20, 22)

[ZJ10] Yingying Zhang and Albert Jeltsch. The application of next generation sequencing in DNA methylation analysis. *Genes*, 1(1):85–101, 2010. (p. 12)

Curriculum Vitae

Niko Popitsch

Center for Integrative Bioinformatics Vienna, Max F. Perutz Laboratories,
University of Vienna, Medical University of Vienna,
Dr. Bohr Gasse 9, Vienna, A-1030, Austria Phone: +43-1-4277-24028
E-Mail:: niko.popitsch@univie.ac.at
WWW: http://www.cibiv.at/~niko

Personal

Born:	February 2nd, 1976 in Graz, Austria
Citizenship:	Austria

Education

2005–2012	Diploma in molecular biology, University of Vienna
2007–2011	PhD in computer science, University of Vienna
1994–2000	Diploma in informatics, Vienna University of Technology

Research and professional experience

Since 2011	Postdoctoral researcher at the Medical University of Vienna/ Center for Integrative Bioinformatics Vienna (CIBIV)
2007–2011	Research assistant at the University of Vienna, research group *Multimedia Information Systems*
2003–2007	Senior researcher at ARC Seibersdorf research GmbH, Studio *Digital Memory Engineering*
2001–2003	Uma information systems
2000–2001	Blue-C Internet GmbH