UIO forside her

aspects of reproducability and reusability of genome assembly evaluation with the gage-b as a case study

Sabba Ifzal

May 1, 2014

Tom side

Acknowledgements

Takke folk som har hjulpet til

Veiledere – Torbjørn, Lex, Geir Kjetil og Ksenia

Andre folk i 10 – Sveinung, Kai, Jon

Ruth for sensur

Ghada for samtaler

etc

Abstract

Kort oppsummert hele oppgaven

Why you do this

How you do this

The results

Conclusion

Preface

Hvem er oppgaven ment for? Andre bioinformatikkstudenter

Contents

[Acknowledgements 5](#_Toc383649176)

[Abstract 6](#_Toc383649177)

[Preface 8](#_Toc383649178)

[Contents 10](#_Toc383649179)

[List of figures 12](#_Toc383649180)

[List of tables 13](#_Toc383649181)

[Chapter 1 Introduction 14](#_Toc383649182)

[1.1 Background 14](#_Toc383649183)

[1.2 PROBLEM STATEMENT / AIMs 14](#_Toc383649184)

[1.3 Literature review 15](#_Toc383649185)

[1.4 Problem solution 15](#_Toc383649186)

[Chapter 2 Background 17](#_Toc383649187)

[2.1 Genome Assembly 17](#_Toc383649188)

[2.1.1 Sequencing 17](#_Toc383649189)

[2.1.2 Assembly 18](#_Toc383649190)

[2.1.3 Quality measures 19](#_Toc383649191)

[2.2 Bioinformatics **Error! Bookmark not defined.**](#_Toc383649192)

[2.2.1 Benchmarking **Error! Bookmark not defined.**](#_Toc383649193)

[2.3 Galaxy 21](#_Toc383649194)

[Chapter 3 Material 22](#_Toc383649195)

[3.1 Datasets 22](#_Toc383649196)

[3.1.1 GAGE-B 22](#_Toc383649197)

[3.1.2 Galaxy 23](#_Toc383649198)

[3.2 Reference genomes 23](#_Toc383649199)

[3.3 Software 24](#_Toc383649200)

[Chapter 4 Methods 26](#_Toc383649201)

[4.1 Reproducing the GAGE-B results 26](#_Toc383649202)

[4.2 Galaxy 26](#_Toc383649203)

[Chapter 5 Results 27](#_Toc383649204)

[Chapter 6 Discussion 28](#_Toc383649205)

[6.1 Interpreting the GAGE-B results 28](#_Toc383649206)

[6.2 Analysis of galaxy tool 28](#_Toc383649207)

[6.2.1 Performance 28](#_Toc383649208)

[6.2.2 Potential use 28](#_Toc383649209)

[6.2.3 Weakness 28](#_Toc383649210)

[6.3 Further work 28](#_Toc383649211)

[6.4 Conclusion 28](#_Toc383649212)

[Glossary 29](#_Toc383649213)

[Appendix a (Plot/Figure) 31](#_Toc383649214)

[Appendix B (Tables) 32](#_Toc383649215)

List of figures

[figure 1 31](#_Toc377132394)

[Table 7-1 Bla bla bla 32](#_Toc377132395)

List of tables

[figure 1 31](#_Toc377132379)

[Table 7-1 Bla bla bla 32](#_Toc377132380)

# Introduction

## Background

DNA sequencing technology such as Next Generation Sequencing (NGS) is developing and revolutionizing the field of sequencing, allowing scientists to determine the sequence of nucleotides with an extreme speed. The task of puzzling together small pieces of a sequence from a new genome into larger continuous parts, better known as an assembly, is performed by assemblers such as CELERA/CABOG [[1](#_ENREF_1)], Velvet [[2](#_ENREF_2)] and ABySS [[3](#_ENREF_3)] among others. This is a difficult task and is performed with many adjustable parameters and varying speed and results, making the assessment of the algorithms used by the software tools important. Even though some tools, such as QUAST [[4](#_ENREF_4)], that measure the quality of a certain method exist, the bioinformatics field lacks enough information about which method would perform best under certain conditions. There have been some attempts on assessment resulting in benchmarks such as GAGE[[5](#_ENREF_5)] and Assemblathon 1 and 2 [[6](#_ENREF_6), [7](#_ENREF_7)], but in general, the development of benchmarks is slower than the development of assembly methods in itself, making the needs for a new system even more urgent.

Visualize things .. edit first line.

Some biologists have pointed out that one of the disadvantages of current benchmarks is the lack of visualization, such as bar charts and scatterplots. Others want a system that is technically advanced, but user-friendly so that less experienced computer-users can easily adapt to the use of the system. This can be performed by reducing the number of required installation, creating makefiles or by reducing the number of steps required to get an assessment of an assembly. With the rising numbers of new assemblers, each proclaiming to be better than the previous version or the competitor, the need for a system which can give a comparison of old against new with minimal effort is desirable. The desired outcome for a new system is something that will reduce the installation requirement and increase the assessment statistics with more visual parameters such as custom designed plots depending on the users need.

**Why decide to have this project…**

**Why should the project be done, why use time, define reproduce/reuse here**

## Problem statement / Aims

This Galaxy tool will be used for the assessment of how well various tools for genome assembly, short sequence mapping or variant calling perform.

Reproduceability gage-b… and reuseability. Can we have the same approach for our (same) data

Ikke nevn Galaxy her! Målet med oppgaven er å lage et system som kan gjøre dette og dette. Ikke noe annet, ikke noe spesifikt. Eksakt hva du skal gjøre. Kobling til gage-b her, hvis det finnes en kobling.

What you actually doing in the project.

Visualize quastresult and add information quast-like, that can be useful for determining which assembly is the best.

Same principle can be added.

Organize a automatized principle. Make something to compare.

## Literature review

Quast, gage assemblathon

Hvordan andre har funnet den beste løsningen

## Problem solution

**Briefly about the solution. How you solved it.**  
The solution to the increasing need of a new benchmark was to develop a system that combined the good statistical output from QUAST with the flexibility of custom code and visualization in the Galaxy framework. Since QUAST has a rather good output structure, the Galaxy tool reuses this structure with some modifications, using python and javascript, to give the users a more tailored view of the output that can be viewed, modified and rerun as input for the next assessment.

**Benefit of system. Quast need install etc,**   
Using the Galaxy framework to create a tool to compare assemblies benefits future user because they ***do not need to install anything*** as long as it is running on the University of Oslo’s server: *invitro.titan.uio.no:24688*. This is good news for those who get frustrated for having to download, compile, install and run everything separately. All the users need to do is create a user-account (if they want to store their results), upload their assemblies, or copy datasets that other users have published, and run the tool. If anyone wants the tool on their own server, then all they need to do is copy the tool folder from github-repository to their Galaxy instance and add proper link to the tool as described in the attached readme file.

One of the advantages of this tool compared to for instance QUAST is that if a new dataset or assembler is available, then the user can effortlessly compare an old Galaxy-result with the output from the new assembly. This can be done since Galaxy stores each run with its parameters as an element in current history (Figure 1-1). The user save time because they only need to add the old result as one parameter and the new assembly as the second parameter instead of manually adding all the old datasets, the new assembly and other parameters.

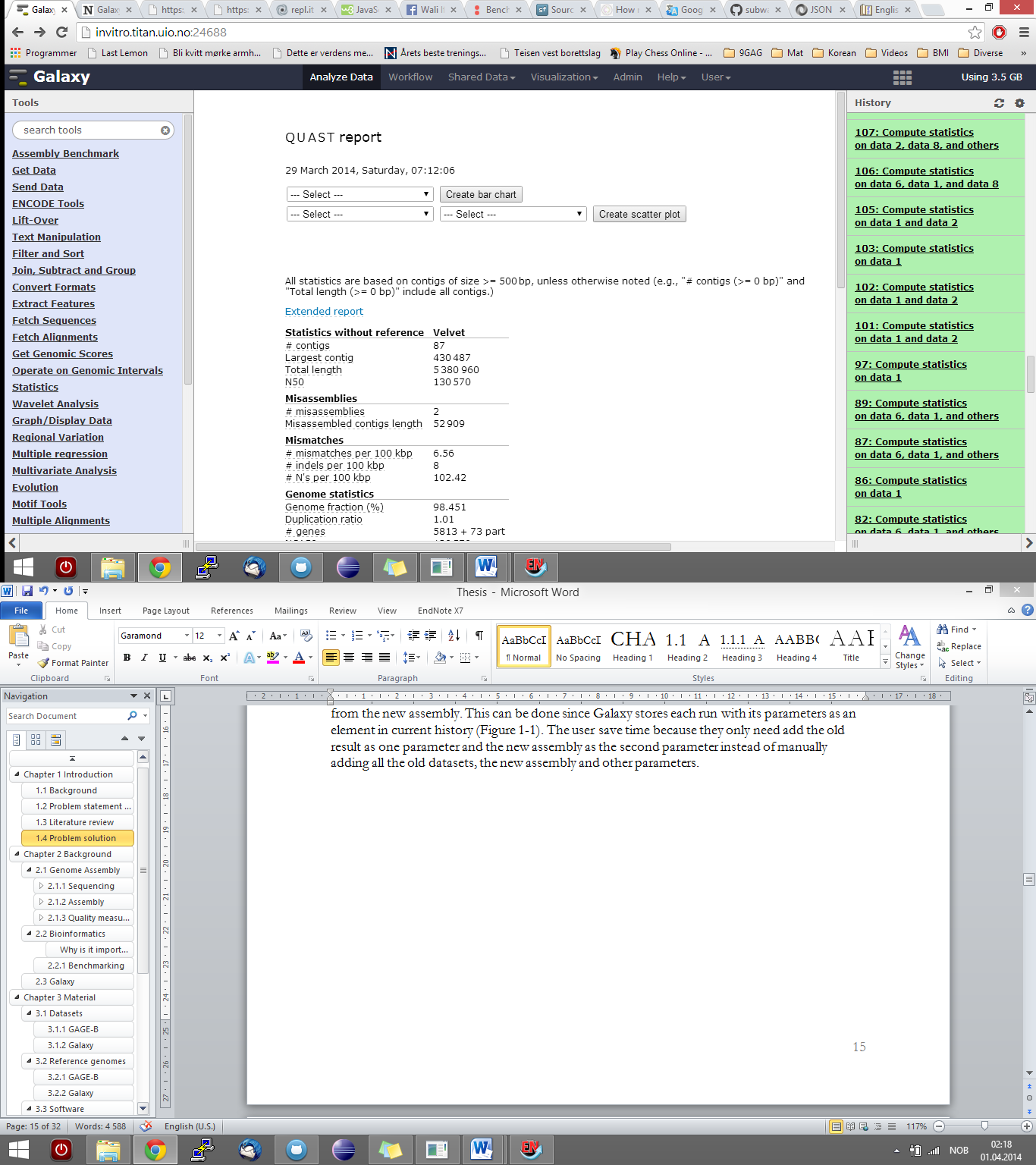


Figure 1‑ Screenshot of the Galaxy tool with an example of the history panel

# Background

Since the thesis will involve discussion of problem areas in bioinformatics that require some biological knowledge, this chapter will provide the basics of genome assembly, reproducibility, reusability and the Galaxy Project.

## Genome sequencing and assembly

What more powerful form of study of mankind could there be   
than to read our own instruction book?  
Francis Collins

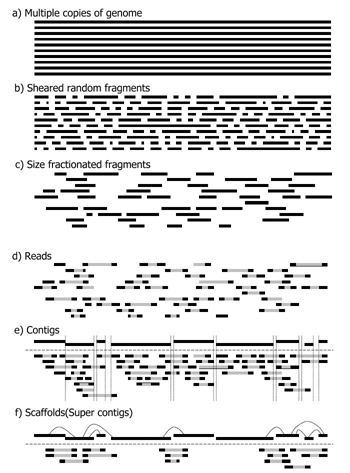


Figure ‑ Basic steps of genome sequencing and assembly [[8](#_ENREF_8)]

### Sequencing

When looking at sequencing in a biological context, it is usually referred to as a process (a method or technology) that is used to get a set of **reads** from one or multiple copies of a genome as illustrated in Figure 2-1(a-d). How this process works in practice depend on the sequencing technology used which can be read more about in the next section.

#### Sequencing technologies

This section will cover three well-known sequencing technologies, including how they work, how many reads and bases they create, and the cost for each per million bases. Information used in this section is gathered from the book *Algorithms in Bioinformatics* [[9](#_ENREF_9)].

**454 PYROSEQUENCING TECHNOLOGY**

This technology was developed in 2005 and was the first NGS technology licensed by 454 Life Science. In this technology, a set of single-stranded **template DNA** is loaded on a set of arranged beads that are amplified using emulsion-based PCR (emPCR) and read in parallel using pyrosequencing (as shown in Figure 2-2). 454 Pyrosequencing technology generates 1 million reads (length 400 bases) per 10 hours and cost about US$60 per million bases.

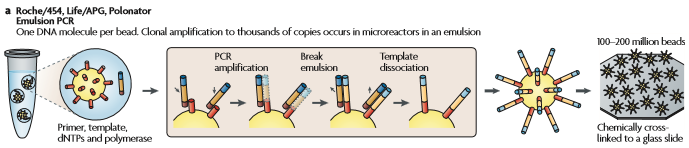


Figure 2‑ Template preparation in 454 pyrosequencing technology [[10](#_ENREF_10)](p.33)

**ILLUMINA SOLEXA TECHNOLOGY**

This technology was acquired and commercialized by Illumina in 2006. The difference from 454 was that Illumina generated shorter reads. After preparing a set of single stranded template DNA and randomly fix the two ends of the template on the surface of a flow cell, it amplifies the template using bridge PCR and read the flow cell in parallel using four-color fluorescent dye and a polymerase-mediated primer extension reaction (as shown in Figure 2-3). In this way it generates about 80 million reads (length 50 bases) in 3 days and cost about US$2 per million bases, which is a major reduction from the cost of 454 technology.

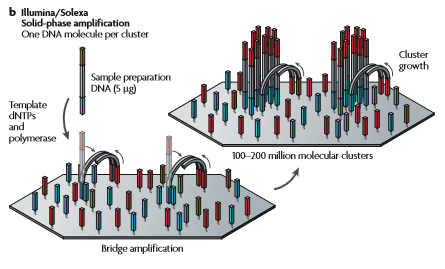


Figure 2‑ Template preparation in illumina [[10](#_ENREF_10)](p.33)

The datasets used in this thesis are only MiSeq and Hiseq Illumina **paired-end reads**. Table 2-1 lists the differences between the instruments and also read type beeing paired-end reads or **single-end reads**. In general, Illumina MiSeq focus on speed and simplicity for targeted and small genome sequencing, with small genome, amplicon, and targeted gene panel sequencing as key applications. Illumina HiSeq on the other hand, focus on power and efficiency for large-scale genomics. exome, transcriptome sequencing, and more.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Instrument \* | Run type | Run time | Bases/read | Maximum number of reads/lane | Total number of bases/lane |
| MiSeq (50 cycles v2 kit) | SE50 | 5 h | 50 | 15-17 million | 750-850 Mb |
| MiSeq (300 cycles v2 kit) | PE150 | 24 h | 150 + 150 | 30-40 million | 4.5-5.1 Gb |
| MiSeq (500 cycles v2 kit) | PE250 | 39 h | 250 + 250 | 30-40 million | 7.5-8.5Gb |
| MiSeq (150 cycles v3 kit) | SE150 | 24 h | 150 | 22-25 million | 3.3-3.8 Gb |
| MiSeq (150 cycles v2 kit) | PE75 | 24 h | 75 + 75 | 44-50 million | 3.3-3.8 Gb |
| MiSeq (600 cycles v3 kit) | PE300 | 65 h | 300 + 300 | 44-50 million | 13.2-15 Gb |
| HiSeq 2000 v3 | SE50 | 2-3 days | 50 | 187 million | 9.4 Gb |
| HiSeq 2000 v3 | SE100 | 5-6 days | 100 | 187 million | 18.7 Gb |
| HiSeq 2000 v3 | PE50 | 5-6 days | 50 + 50 | 374 million | 18.7 Gb |
| HiSeq 2000 v3 | PE100 | 9-11 days | 100 + 100 | 374 million | 37.4 Gb |

Table ‑ Comparison of illumina sequencing instruments and capasity  
 \* The error rate for Illumina MiSeq and HiSeq sequencer is ≥0.1%  
 Source: <http://dna.uga.edu/services/hiseq-miseq-sequencing/>

**ABI SOLID TECHNOLOGY**

Supported Oligonucleotide Ligation and Detection Platform (SOLiD) was introduces in 2007 by Life technologies. This technology loads a set of single-stranded template DNA on beads and amplifies it using emPCR and then distributes the beads randomly on a glass plate so that it can be read in parallel via sequencing by ligation. It can generate about 115 million reads (length 35 bases) in 5 days and cost about US$2 per million bases.

#### De novo sequencing vs resequencing

De novo sequencing (from latin as “from the beginning”, “afresh” or “anew”) is a collective term used for methods that sequence a DNA sequence for the first time (meaning that it has no previously known sequence), known genomes where significant structural variation is expected, or for microbial sequencing that includes experimental strains and genomes with high plasticity. This kind of sequencing use starting materials such as bacterial, viral, phage, fungus, eukaryote genomic DNA, fragmented DNA or BACs and fosmids. It can use human genome, but substantial challenges exist for its application to this.

Resequencing on the other hand…

#### Sanger sequencing vs next generation sequencing

Sanger sequencing…

Next-Generation sequencing (NGS) technologies are usually applied as a general term for sequencing platforms that use post-Sanger technology to sequence DNA fragments in parallel. They were developed to reduce weaknesses in de novo sequencing. As a result, this technology had a major increase in efficiency as well as a decrease in cost per base and could produce a lot of data cheaply compared to de novo sequencing. The “down side” was that the read length was shorter, thus creating an assembly challenge [[11](#_ENREF_11)] as it made for example sequencing repeated elements difficult and could result in mis-assembled genomes as Figure 2-4 shows below.

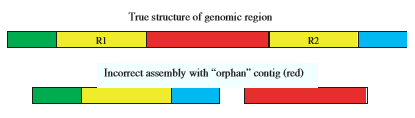


Figure 2‑ The challenge of sequencing repeated element [[12](#_ENREF_12)](Fig 1)

### Assembly

Assembly can, roughly speaking, be described as a process where some reads, with a minimum of X **read depth** or **coverage**, are used to make **contigs** (Figure 2-1(e)), which are then used to make **scaffolds** (Figure 2-1(f)). The goal is to end up with one continuous sequence equal to the target DNA. The new sequence can then be mapped back to a reference, if one exists, to check the correctness of the assembly to a certain degree. This process can be used to determine the order of genes, full chromosomes or entire genomes. This is important because the sequence in which the nucleotides appears in gives scientists valuable information about that part of the DNA which can, for instance, be used to look for disease-causing mutations in genes

#### Assembly algorithms

Still nothing here

#### Assembly challenges

As we have seen in the section above about assembly algorithms, quite a number of papers describe and discuss upon assembly methods, algorithms and how the assembly process works in general. But unfortunately, when it comes to the number of papers describing how (or discussing why) assembly is a difficult process, and challenges regarding it, the number decreases a lot. This contributes to making the understanding of genome assembly even more difficult for someone entering the field for the first time. Nonetheless, one paper that tries to discuss this is *Genome assembly reborn: recent computational challenges* by Mihai Pop,[[13](#_ENREF_13)] where Pop use solving a jigsaw puzzle as a metaphor to an assembly process. Another complementary paper used for this section is *Genome assembly forensics: finding the elusive mis-assembly* by Phillippy, Scatz and Pop[[14](#_ENREF_14)].

One of the problems regarding assembly process is about genome repeats which can be described as large stretches of sky in a jigsaw puzzle. Repeats tend to confuse the assembly process, because they seem identical to the assembler. They also make it difficult to distinguish between sequencing error and polymorphism among near-identical repeats. There are also the tandem repeats making it even more difficult. Many of the problems concerning repeats can be generalized in the following two categories:

**1. Repeat collapse and expansion**

The assembly can incorrectly gauge the number of repeats, thus including too few or too many copies, which can result in phenotypic differences, such as Huntigton’s disease.

The assembler can also mis-join reads originating from distinct repeat copies into one unit or include extra copies of repeat, both which can be noted as a higher or lower density of reads.

**2. Sequence rearrangement and inversion**

The assembler can shuffle the order of multiple repeat copies, which could be misinterpreted as a biological rearrangement event, meaning that one could draw wrong conclusions depending on the rearranged sequence.

During both repeat collapse and rearrangement, reads may get placed in a wrong copy of a repeat, therefore SNP could be a useful indicator of such a mis-assembly.

Another problem is looking at the complexity of an assembly, which depends on the number of reads being assembled. An assembly becomes more complex the more reads it assembles, making it quite complex, considering the development of shorter reads and sequencing tools that generates several million reads in a reasonable time and at a low cost.

One of the most time consuming task is probably the computation of overlaps. This task can have assembly errors which can occur due to limitations of the assembly algorithm, or by providing incorrect or incomplete assembly-parameters. It can be difficult to see where there are indels (an insertion or deletion of bases), mis-join, or find the exact placement of reads, and the detection of these errors are what scientists try to improve.

### Quality measures

There are many traps to avoid when it comes to assembly, and how well they are avoided can be measured and used to determine how well the results are. Some quality measures are easier to assess than others, especially with a reference genome. Of course, with a reference, the solution is already there, and the interesting part might be to spot the differences, compared to "normally" when the correctness of an assembly is undefined. It is therefore many criterias that can be used to assess the quality and correctness of an assembly such as the coverage and length of contigs or scaffolds, the length of the gaps between scaffolds, Nx (usuallyN50), how accurate or correct the sequence is compared to its reference, the error rate or how fast and cost-efficient it is, to mention some of the criterias. Other metrics such as the number of unaligned contigs, relocations, translocations and inversions can also be used by comparing to a reference genome. It is also possible to measure by metrics such as; the total number of contigs in the assembly, how long the assembly is (in number of bases), how long the misassembled contigs are or by looking at the (average) number of indels after x number of aligned bases.

Different measures can weight different depending on what the purpose of the assembly is. For instance, the size of scaffolds might be less important than the error rate in one case whereas the number of genes might be crucial in another case. A couple of commonly used quality measures are listed in the subsections below, followed by a brief overview of a tool that assesses an assembly using these measures among others.

#### Nx

Nx of an assembly is a metric defined as a weighted median of the lengths of the sequences it contains, equal to the length of the longest sequence s, such that the sum of the lengths of sequences greater than or equal in length to s is greater than or equal to x% of the genome being assembled [[7](#_ENREF_7)].

#### NGX

NGx of an assembly is a metric similar to Nx except that it accounts for x% of the length of the reference genome, instead of x% of the assembly length [[4](#_ENREF_4)].

#### Coverage

The read coverage can also say something about the assembly. Usually, the higher the coverage is, the more politely is the nucleotide base position. The most ideal situation is when all the contigs have as high read coverage as possible.

#### Contig size

This is defined as the total number of contigs in an assembly. In general, the fewer and longer the contigs are, the better it is. That is of course while assuming that the contigs are assembled correctly, which unfortunately is not true in all cases. This is where the coverage might clarify the correctness of the contigs.

#### Representation

The representation can be used as a quality measure, assuming that a reference genome is available. It is then defined as the percent of the reference genome which is covered by assembled contigs. This is a measure that is desired to be as high as possible.

#### Number of gaps

This measure calculates the number of gaps present between assembled contigs which align to the reference. Since misassemblies can force a contig to be aligned in more than one location, the number of gaps could be higher than the total number of contigs [[15](#_ENREF_15)].

#### Mismatch rate

The mismatch rate is the percent of mismatches in the assembly. The percentage is calculated by dividing the total number of mismatches by the total number of bases in the specified reference or reference region[[15](#_ENREF_15)].

#### Indel rate

This **indel** rate is defined as the percent of indels in the assembly. The percentage is obtained by dividing the total number of indels by the total number of bases in the specified reference or reference region [[15](#_ENREF_15)].

#### Misassembly rate

The misassembly rate is defined as the average number of misassemblies found per Mb of sequence [[15](#_ENREF_15)]. A position is considered a misassembly if it is a **relocation**, a **translocation** or an **inversion**.

#### Error rate

The error rate is defined as the percent of errors in an assembly. The rate is obtained by dividing the total number of base errors (indel and mismatch) by the total number of bases in the specified reference or reference region [[15](#_ENREF_15)].

#### Quast

Quality Assessment Tool (QUAST) is a tool that evaluates and compares genome assemblies both with and without a reference genome. It is designed to improve existing assembly comparison software (such as GAGE) and produces results as reports, summary tables and plots that support SVG, PNG and PDF formats. An example of a metric that QUAST use is the NGx, which is like the Nx, but instead of comparing to the assembly length, the contigs are compared to the reference genome length [[4](#_ENREF_4)] As you can see in the figure below, quast gives a rather numerical report without giving the overall “best assembly” in the comparison. QUAST is rather mathematical, thus giving the user a table with numeric data and a minimum of dynamic and static plots based on the table-values.

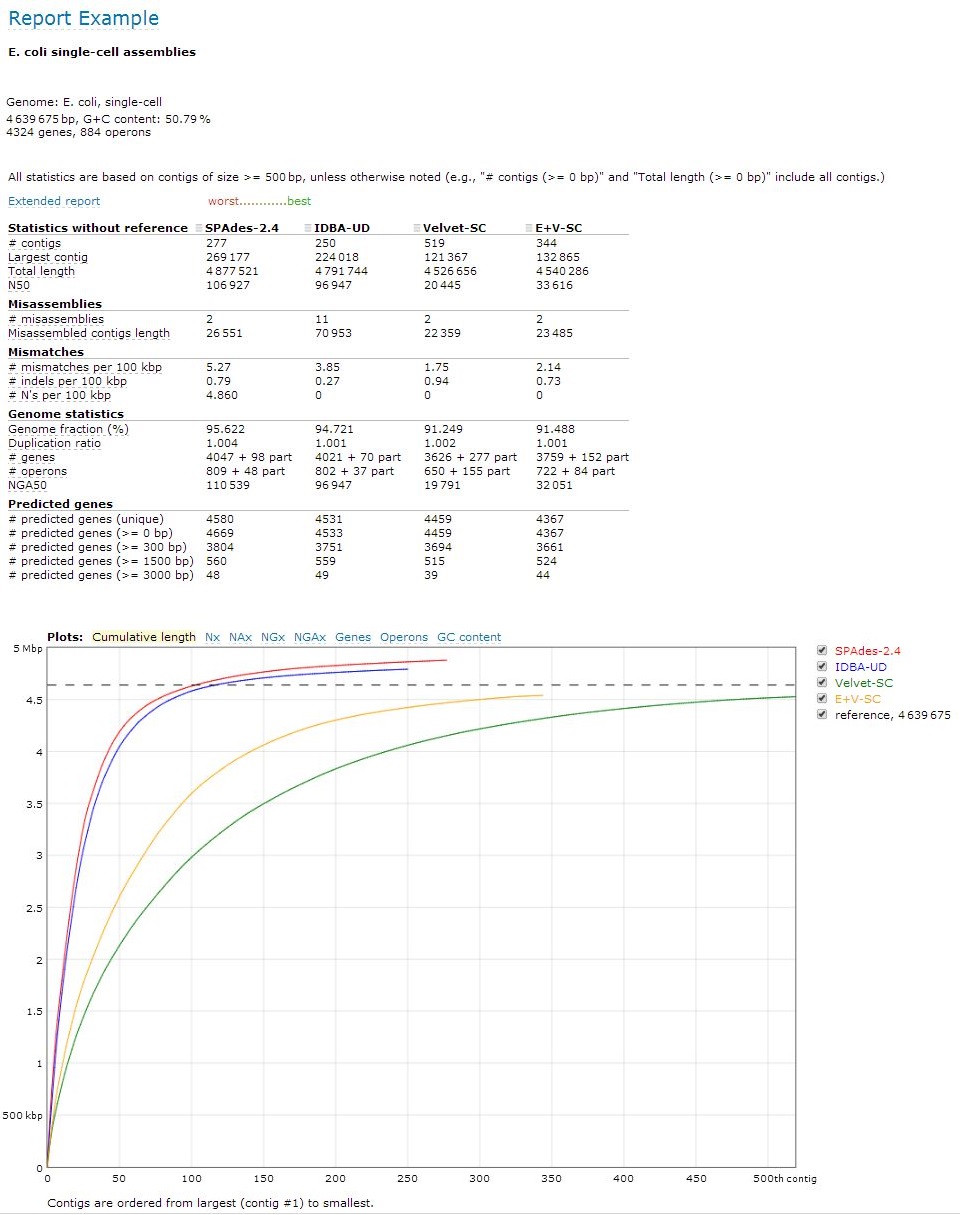


Figure 2‑ Quast report example (<http://quast.bioinf.spbau.ru/>)

## Reproducability

An experiment is reproducible until another laboratory tries to repeat it.   
Alexander Kohn [[16](#_ENREF_16)]

One of the main principles of **the scientific method** is the ability to reproduce an entire experiment or study. Reproducibility is said to be a fundamental part of science because it enables people to develop work further by applying new data or methodology, build on the work of others or to verify published results. It is expected, in a biological context, that findings can be replicated by independent data, analytical methods, laboratories and instruments. [[17](#_ENREF_17)]

Unfortunately, in the field of bioinformatics, the amount and complexity of data collections with the increasingly sophisticated analyses can sometimes make it difficult to reproduce the results fully. In some cases, studies cannot be replicated at all due to the lack of time, money or resources while in other cases, even if there exits somewhat reproducible research, the documentation is poorly written, making a correct reproduction quite difficult. The documentations might be written poorly because the researchers feel that they need to sustain their reputation by getting results fast so that they can win the race of publishing new findings first. Unfortunately, this often implies that the end justifies the means, making reproducibility quite difficult. Lately, to avoid those kinds of trouble for other, maybe independent researchers, it has been common to provide the datasets and software used for the findings so that other scientist can verify the published findings or conduct alternative analysis.

Many papers have been written over the years about reproducibility and one paper written by Sandve et al. [[18](#_ENREF_18)] have a good 10-rules description for reproducible computational research as follows:

1. For every result, keep track of how it was produced
2. Avoid manual data manipulation steps
3. Archive the exact versions of all external programs used
4. Version control all custom scripts
5. Record all intermediate results, when possible in standardized formats
6. For analysis that includes randomness, note underlying random seeds
7. Always store raw data behind plots
8. Generate hierarchical analysis output, allowing layers of increasing details to be inspected
9. Connect textual statements to underlying results
10. Provide public access to scripts, runs, and results

The replication of findings and studies by multiple independent scientists will in the future be important to the accumulation of scientific evidence. Hopefully, more researchers will adapt to this description in upcoming publications, thus making reproducibility simpler.

## Reusability

Good programmers know what to write. Great ones know what to rewrite (and reuse).  
Eric S. Raymond [[19](#_ENREF_19)]

The notion of reusability is, as stated by Prieto-Diaz in *Status Report: Software reusability* [[20](#_ENREF_20)], an old idea were solutions to current problems are modified, combined, and adapted to solve similar new problems. In computer science and software engineering, reusability is described as the reuse of source code segments, product generated during software development (such as system specification and requirements documents) and any information needed for developing new software.

Writing reusable code is hard. Not only do developers have to deal with local services, permissions, dependencies and license issues, they also have to provide decent comments explaining exactly *what* their code does and all sorts of documentation that another developer might need to reuse the code properly. The problem with reusability can be that sometimes only the biological results that matter for a given publication comes first, resulting in non-reusable software afterwards that few takes time and effort to make reusable again [15]. It is also, on the other hand, difficult to reuse code because some developers think that it is easier to build something from scratch. In this way, they know exactly *what* is happening, *how* it is happening and *when* it is happening. For some developers, it’s faster to write something again in their own style than to read and understand someone else’s code segment and figure out where to modify changes for the new purpose.

Something more here…

## The Galaxy project

Galaxy is an open, web-based platform for data intensive biomedical research. Whether on the free public server or your own instance, you can perform, reproduce, and share complete analyses.  
galaxyproject.org

# Material

## Datasets

### Read data

|  |  |  |
| --- | --- | --- |
|  | Sequencing technology | Size |
| Aeromonas hydrophila SSU | Hiseq | 7 GB |
| Bacillus cereus VD 118 | Hiseq | 7 GB |
| Bacillus cereus ATCC 10987 | Miseq | 2 GB |
| Bacteroides fragilis HMW 615 | Hiseq | 7 GB |
| Mycobacterium abscessus 6G-0125-R | Hiseq | 2.5 GB |
| Mycobacterium abscessus 6G-0125-R | Miseq | 2 GB |
| Rhodobacter sphaeroides 2.4.1 | Hiseq | 4.5 GB |
| Rhodobacter sphaeroides 2.4.1 | Miseq | 1.5 GB |
| Staphylococcus aureus M0927 | Hiseq | 4.5 GB |
| Vibrio cholerae CO 1032(5) | Hiseq | 2 GB |
| Vibrio cholerae CO 1032(5) | Miseq | 1.5 GB |
| Xanthomonas axonopodis pv. Manihotis UA 323 | Hiseq | 8 GB |

Table 3‑1 – Species, their sequencing technology and size  
Downloadlink: <http://ccb.jhu.edu/gage_b/datasets/index.html>

All the read datasets used in this thesis are shown above in table 3-1. The reproducing of the GAGE-B papers results was initially started with all datasets in table 3-1. The datasets were first assembled according to the recipe on their site with an assembler named Velvet. Out of the 8 assemblers used by the GAGE-B researchers, Velvet was chosen first because it was the most easy-to-install assembler available in regards to dependencies and student permission on the faculty-computers. After a while, realizing the amount of time and effort needed to reproduce the results fully, the focus shifted from all datasets and assemblers, to one set of MiSeq and HiSeq data assembled using all the 8 assemblers instead. The dataset that was chosen for this was the specie *Vibrio cholera.* The reason behind this choice was that Vibrio cholerae consisted of the smallest set of MiSeq and Hiseq data, with a total of 3.5 GB compared to a total of 4.5-8 GB for the other species. The choice was based on the fact that assembling the data with multiple assemblers, interpreting the results and comparing them to the original GAGE-B results, as described in the paper, were all quite time-consuming tasks.

Sometimes, the raw reads produced by the sequencer are not correct in their whole length because of contaminants, adapter sequences or low-quality sequences. Using the entire read then may introduce artifacts in the genome assembly, and to avoid that, the reads are trimmed or cleaned using various software tools such as for example Trimmomatic [[21](#_ENREF_21)] or, as the GAGE-B researchers have done, by removing adapter sequences and performing q10 quality trimming using the ea-utils package.

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | ABySS | CABOG | MIRA | MaSuRCA | SGA | SOAPdenovo | SPAdes | Velvet |
| V.cholearea  – Hiseq | clean | clean | raw | Raw | clean | clean | clean | clean |
| V.cholearea – Miseq | clean | raw | clean | Clean | clean | raw | clean | clean |

Table 3‑ – Read type used for each assembler on V.cholerae

Originally, the thought behind doing the assemblies when precompiled results were available was that the results were going to be used later on in the Galaxy tool. The tool could assess the results and see if they correlated with the GAGE-B conclusion. But unfortunately, it did not go as planned and the task of partially reproducing the GAGE-B paper had to be discarded. Chapter 4 or 5 elaborate more on the reason behind this undesired choice of termination.

As for the Galaxy tool, the testing was performed using all listed in table 3-1 with both contigs and scaffolds files.

### Reference genomes

While partially reproducing the GAGE-B results, the fastest way was to skip the reference genomes initially to check if the basic statistic was somewhat similar. The idea was to use reference genomes afterwards, but this task never advanced enough to include all the reference files so in the end, only the reference for Vibrio cholera were used.

The reference genome and the genefiles used for the assessment of the Galaxy tool where all downloaded from the GAGE-B’s website <http://ccb.jhu.edu/gage_b/datasets/index.html>. Each species had quite a list of files available, but only the sequence files (fna) and their corresponding gene files (gff) were used. The name of the reference genomes and link to their download site are shown in table 3-4.

|  |  |  |
| --- | --- | --- |
| Aeromonas hydrophila ATCC 7966 | | |
| Chromosome 1 | 4.6 MB | NC\_008570 |

|  |  |  |
| --- | --- | --- |
| Bacillus cereus ATCC 10987 | | |
| Chromosome 1  Plasmid pBc10987 | 5.1 MB  206 kB | NC\_003909  NC\_005707 |

|  |  |  |
| --- | --- | --- |
| Bacteroides fragilis 638R | | |
| Chromosome 1 | 5.2 MB | NC\_016776 |

|  |  |  |
| --- | --- | --- |
| Mycobacterium abscessus | | |
| Chromosome 1  Plasmid 1 | 4.9 MB 23.2 kB | NC\_010397 NC\_010394 |

|  |  |  |
| --- | --- | --- |
| Rhodobacter sphaeroides 2.4.1 | | |
| Chromosome 1 Chromosome 2  Plasmid A Plasmid B Plasmid C Plasmid D Plasmid E | 3.1 MB 934 kB 113 kB 113 kB 104 kB 100 kB 36.8 kB | NC\_007493 NC\_007494 NC\_009007 NC\_007488 NC\_007489 NC\_007490 NC\_009008 |

|  |  |  |
| --- | --- | --- |
| Staphylococcus aureus SA300\_TCH1516 | | |
| Chromosome 1  Plasmid pUSA300HOUMR Plasmid pUSA01-HOU | 2.8 MB 26.9 kB 3.2 kB | NC\_010079 NC\_010063 NC\_012417 |

|  |  |  |
| --- | --- | --- |
| Vibrio cholerae O1 biovar eltor str. 16961 | | |
| Chromosome 1  Chromosome 2 | 2.9 MB 1.0 MB | NC\_002505 NC\_002506 |

|  |  |  |
| --- | --- | --- |
| Xanthomonas axonopodis pv. Citrumelo | | |
| Chromosome 1 | 4.8 MB | NC\_016010 |

Table 3‑ Reference genome for each dataset  
Source: <http://ccb.jhu.edu/gage_b/datasets/index.html>

## Software

The various pieces of software used while reproducing the GAGE-B papers results were:

* ABySS v1.3.4
  + <http://www.bcgsc.ca/platform/bioinfo/software/abyss>
* CABOG v7.0
  + <http://sourceforge.net/apps/mediawiki/wgs-assembler>
* MIRA v3.4.0
  + <http://www.chevreux.org/project_mira.html>
* MSRCA v1.8.3
  + <http://www.genome.umd.edu/>
* SGA v0.9.34
  + <https://github.com/jts/sga/wiki>
* SOAPdenovo2 v2.04 + GapCloser v1.12
  + <http://sourceforge.net/projects/soapdenovo2>
  + <http://soap.genomics.org.cn/soapdenovo.html>
* SPAdes v2.3.0
  + <http://bioinf.spbau.ru/en/spades>
* Velvet v1.2.08
  + <http://www.ebi.ac.uk/~zerbino/velvet/>

Software used in combination with the Galaxy framework was:

* Python v2.7
* Quast v2.2
* Google charts

# Methods

This chapter covers a description of the tool and implementation.

## Reproducing the GAGE-B results

After a while, realizing the amount of time and effort needed to reproduce the results fully, I decided that I would focus on just one species.

Why vibrio cholera? Because of this and that.

## Galaxy

# Results

Describe why you could not use your own assembly based on the GAGE-B datasets.

# Discussion

## Interpreting the GAGE-B results

## Analysis of galaxy tool

### Performance

### Potential use

### Weakness

## Further work

## Conclusion

Referanseliste!

Glossary =)

Glossary

|  |  |
| --- | --- |
|  |  |
|  |  |
| BAC | Abbreviation of *bacterial artificial chromosome* |
| bacterial artificial chromosome (BAC) | An artificially constructed segment of nucleic acid used for transforming and cloning in bacteria, usually E. coli. |
| bacteriophage | A virus that parasitizes a bacterial cell |
| chromosome | A threadlike, gene-carrying structure found in the nucleus. Each chromosome consists of one very long DNA molecule and associated proteins |
| contig | A continuous sequence of DNA that have been assembled from overlapping reads |
| coverage | The average number of reads representing a given nucleotide in a reconstructed sequence; also known as read depth or depth |
| de novo sequencing |  |
| deoxyribonucleic acid (DNA) | A double-stranded, helical nucleic acid molecule capable of replicating and determining the inherited structure of a cell's proteins |
| DNA | Abbreviation of *deoxyribonucleic acid* |
| eukaryote | An organism whose cells contain membraine-bound organelles and whose DNA is enclosed in a cell nucleus and is associated with proteins |
| fungus | About 80,000 known species of organisms of the kingdom Fungi, which includes the yeasts, rusts, smuts, mildews, molds, mushrooms, and toadstools |
| genome | The complete complement of an organism's genes; an organism's genetic material |
| indel | An insertion or the deletion of bases in the DNA |
| inversion | A misjoin of a scaffold/contig where the two pieces map to the opposite strands on the same chromosome |
| nucleus | The chromosome-containing organelle of a eukaryotic cell |
| paired end reads | Reads that are sequenced from both ends and referred to as R1 and R2. Usually there is a "gap" in between them and although we don't know the sequence of DNA in between R1 and R2, we still have gained useful information from the knowledge that R1 and R2 are next to each other with a known orientation and distance apart |
| phage | A virus that infects bacteria; also called a bacteriophage |
| polymorphism |  |
| protein | A three-dimensional biological polymer constructed from a set of 20 different monomers called amino acids |
| read | Pieces of a sequence acquired under sequencing used for mapping/assembly that vary in length from less than 100 base pairs up to several thousand base pairs. Usually, with a double stranded chain, the reads contains the direction as well |
| read depth | See *coverage* |
| relocation | A misjoin of a scaffold/contig where the two pieces map to different locations on the reference genome |
| scaffold | A series of contigs that are in the right order but not necessarily connected in one continuous stretch of sequence. The remaining gaps between contigs in a scaffold can usually be sequenced because the placement of contigs are often known |
| single end reads | As opposed to *paired end reads,* single end reads are only sequenced from one end of the fragment |
| template DNA | a nucleotide sequence that directs the synthesis of a sequence complementary to it by the rules of *Watson crick base pairing*. A molecule that provides the structural mould to create similar molecules |
| the scientific method | A series of steps used for investigating observable events, acquiring new knowledge or correcting/integrating previous knowledge. |
| translocation | A misjoin of a scaffold/contig where the two pieces map to different chromosomes or plasmids |
| viral | A biological virus |
| virus | A submicroscopic, noncellular particle composed of a nucleic acid core and a protein coat (capsid); parasitic; reproduces only within a host cell. |
| Watson crick base pairing | guanine-cytosine (G-C) and adenine-thymine (A-T) |

Appendix a  
(Plot/Figure)



figure ‑

Appendix B  
(Tables)

|  |  |
| --- | --- |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

Table ‑ Bla bla bla

1. Miller, J.R., A.L. Delcher, S. Koren, et al.,

*Aggressive assembly of pyrosequencing reads with mates.*

Bioinformatics, 2008. **24**(24): p. 2818-2824.

2. Zerbino, D.R. and E. Birney,

*Velvet: Algorithms for de novo short read assembly using de Bruijn graphs.*

Genome Research, 2008. **18**(5): p. 821-829.

3. Simpson, J.T., K. Wong, S.D. Jackman, et al.,

*ABySS: a parallel assembler for short read sequence data.*

Genome Research, 2009. **19**(6): p. 1117-23.

4. Gurevich, A., V. Saveliev, N. Vyahhi, et al.,

*QUAST: quality assessment tool for genome assemblies.*

Bioinformatics, 2013. **29**(8): p. 1072-5.

5. Salzberg, S.L., A.M. Phillippy, A. Zimin, et al.,

*GAGE: A critical evaluation of genome assemblies and assembly algorithms.*

Genome Research, 2012. **22**(3): p. 557-67.

6. Bradnam, K., J. Fass, A. Alexandrov, et al.,

*Assemblathon 2: evaluating de novo methods of genome assembly in three vertebrate species.*

Gigascience, 2013. **2**(10).

7. Earl, D., K. Bradnam, J. St John, et al.,

*Assemblathon 1: a competitive assessment of de novo short read assembly methods.*

Genome Research, 2011. **21**(12): p. 2224-41.

8. *Genome assembly*.

9. Sung, W.-K., *Algorithms in Bioinformatics: A Practical Introduction*. 2009: CRC Press (Taylor & Francis Group). 24-25.

10. Metzker, M.L.,

*Sequencing technologies - the next generation.*

Nature Reviews Genetics, 2010. **11**(1): p. 31-46.

11. Schlebusch, S. and N. Illing,

*Next generation shotgun sequencing and the challenges of de novo genome assembly.*

South African Journal of Science, 2012. **108**(11-12): p. 8.

12. Salzberg, S.L. and J.A. Yorke,

*Beware of mis-assembled genomes.*

Bioinformatics, 2005. **21**(24): p. 4320–21.

13. Pop, M.,

*Genome assembly reborn: recent computational challenges.*

Briefings in bioinformatics, 2009. **10**(4): p. 354-66.

14. Phillippy, A.M., M.C. Schatz, and M. Pop,

*Genome assembly forensics: finding the elusive mis-assembly.*

Genome Biology, 2008. **9**(3).

15. Barthelson, R., A.J. McFarlin, S.D. Rounsley, et al.,

*Plantagora: Modeling Whole Genome Sequencing and Assembly of Plant Genomes.*

PLoS ONE, 2011. **6**(12): p. Table S1.

16. Kohn, A. *Reproducibility*. 08.04.14]; Available from: <http://todayinsci.com/K/Kohn_Alexander/KohnAlexander-Quotations.htm>.

17. Peng, R.D., F. Dominici, and S.L. Zeger,

*Reproducible Epidemiologic Research.*

American Journal of Epidemiology, 2006. **163**(9): p. 783-789.

18. Sandve, G.K., A. Nekrutenko, J. Taylor, et al.,

*Ten Simple Rules for Reproducible Computational Research.*

PLOS Computational Biology, 2013. **9**(10): p. e1003285.

19. Raymond, E.S., *The cathedral and the bazaar*. 1998. 1998: O'Reilly Media.

20. Prieto-Diaz, R.,

*Status report: software reusability.*

Software, IEEE, 1993. **10**(3): p. 61-66.

21. Bolger, A.M., M. Lohse, and B. Usadel,

*Trimmomatic: A flexible trimmer for Illumina Sequence Data.*

Bioinformatics, 2014.