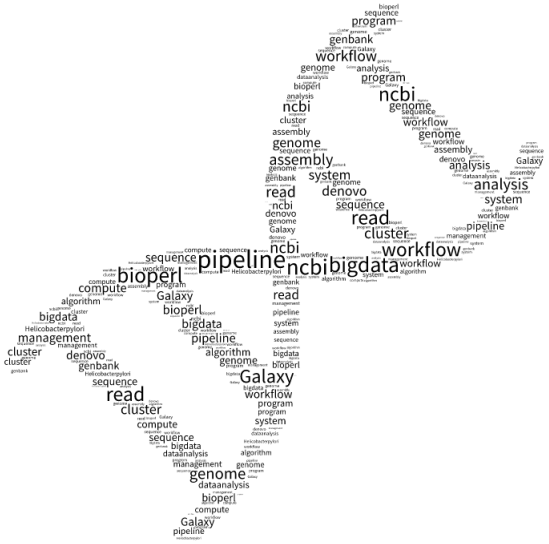
CHALMERS
UNIVERSITY OF TECHNOLOGY

UNIVERSITY OF GOTHENBURG



Bioinformatics pipeline development to support *Helicobacter pylori* genome analysis

Master's thesis in Computer Science

SEYEDEH SHAGHAYEGH HOSSEINI

Department of Computer Science and Engineering
CHALMERS UNIVERSITY OF TECHNOLOGY
UNIVERSITY OF GOTHENBURG
Gothenburg, Sweden 2016

Master's thesis 2016

Bioinformatics pipeline development to support *Helicobacter pylori* genome analysis

SEYEDEH SHAGHAYEGH HOSSEINI

Department of Computer Science and Engineering
UNIVERSITY OF GOTHENBURG
Gothenburg, Sweden 2016

Bioinformatics pipeline development to support *Helicobacter pylori* genome analysis
SEYEDEH SHAGHAYEGH HOSSEINI

Supervisor: Dr. Intawat Nookaew and Dr. Kaisa Thorell, Biology and Biological Engineering, Chalmers University
Professor Graham Kemp, Chalmers University
Examiner: Professor Peter Damaschke, Computer Science and Engineering, Chalmers University

Master's Thesis in Computer Science
Department of Computer Science and Engineering
University of Gothenburg

Cover:
A word cloud illustration of DNA structure. The picture is generated using https://tagul.com.
Gothenburg, Sweden 2016

Bioinformatics pipeline development to support *Helicobacter pylori* genome
Seyedeh Shaghayegh Hosseini
Department of Computer Science and Engineering University of Gothenburg

Abstract

Helicobacter pylori is a bacterium related to a variety of diseases and is a major risk factor for gastric cancer [1]. There can be differences in the genomes of *H. pylori* bacteria that are isolated from different patient groups and this project is motivated by the desire of biologists to investigate how these differences correlate with differences in disease. The development of high-throughput technologies in biological science has lead to big data phenomena. Next generation sequencing (NGS) is an example of such techniques which generate large text-based files, to store short fragments of the whole genome sequence data of an organism quickly and at relatively low-cost [2]. **Achieving methods for analysis and management of such complex datasets has emerged as a challenge which is the subject of this thesis.**

Using bioinformatics approaches, we proposed a pipeline for data analysis and management on two different platforms: High-performance computing and online workflow management system. For High-performance computing, we used a computer cluster from C3SE, a centre for scientific and technical computing at Chalmers University of Technology. On this platform, we developed a pipeline by scripting using perl programming language. The first step in the pipeline is error removal and quality control. Next is to find overlaps between the short fragments of genome sequence and then merging them into continuous, longer sequences. This called *de novo* genome assembly. The final step is genome annotation, the process of transferring biological information from experimentally characterized datasets or reference genomes to newly sequenced genomes. For each step in the pipeline, we used benchmarking techniques to find the best programs that are developed in the bioinformatics community and the case of a missing application we implemented it. The result of the pipeline is well characterised, biologically annotated datasets that are ready for analysis by biologists. For workflow management system, we chose a widely used bioinformatics workflow management system, the Galaxy project. Galaxy provides infrastructure for creating workflows and uploading datasets via a user interface. With Galaxy, we managed to implement a pipeline including quality control and *de novo* genome assembly.

Using the first method, we succeeded to analyse 52 datasets, and this project is the first study that on a significant scale, explores *H. pylori* and its association with gastric cancer.

Keywords: Workflow management system, Pipeline development, Bioinformatics, High-Performance Computing.

Contents

List of Figures

List of Tables

1
Introduction

Helicobacter pylori is a bacterium associated with a variety of diseases and is a major risk factor for gastric cancer [1]. There can be differences in the genomes of *H. pylori* bacteria that are isolated from different patient groups and this project is motivated by the desire of biologists to investigate how these differences correlate with differences in disease. Such analysis has only recently become possible, thanks to the development of "next generation sequencing" (NGS) technologies that now enable generating genome sequence data quickly at relatively low-cost [2]. NGS technologies generate large text-based files, which store short fragments of the whole genome sequence data of an organism. The development of high-throughput technologies in biological science has lead to big data phenomena. The complexity of biological systems makes each study a novel problem of its own which needs a unique combination of tools incorporated into various biological databases and repositories. **Achieving methods for analysis and management of such complex datasets has emerged as a challenge which is the subject of this thesis.**

The analysis workflow is dynamic by its nature. There could be changes in the properties of existing samples (datasets); extra samples might become available, etc. [3]. Therefore, a streamlined bioinformatics pipeline is crucial to facilitate and automate re-analyzing and reproducing results. Although there are some pipelines developed in the bioinformatics community, in most cases it is not possible to adopt the existing pipelines. The reason is the complicated set of dependencies, operating systems incompatibility, hardware incompatibility, etc. [4].

The genome of *H. pylori* is 1.6 megabases long. Using bioinformatics approaches, **we proposed a software pipeline for data analysis and management on two different platforms: High-performance computing and online workflow management system.**

For High-performance computing, we used a computer cluster from C3SE, a centre for scientific and technical computing at Chalmers University of Technology. On this platform, we developed a pipeline by scripting using perl programming language. The first part of the pipeline is error removal and quality control. Next is to find overlaps between the short fragments of genome sequence and then merging them into continuous, longer sequences. This is called *de novo* genome assembly. The final step is genome annotation, the process of transferring biological information from experimentally characterized datasets or reference genomes —that are available in public databases— to newly sequenced genomes. For each step in the pipeline, we used benchmarking techniques to find the best programs that are developed in the bioinformatics community and in the case of a missing application we implemented it. The result of the pipeline is well characterised, biologically annotated datasets

that are ready for biological analysis by the biologist.

For workflow management system, we chose Galaxy, a widely used bioinformatics workflow management system. Galaxy provides infrastructure for creating workflows. It is possible to state the starting point of analysis, introduce/choose the tool(s) and actions to be performed on datasets via a user interface. With Galaxy, we managed to implement a pipeline including quality control and *de novo* genome assembly.

Using the first method, we succeeded to analyse 52 datasets and this project is the first study that on a significant scale, explores *H. pylori* and its association with gastric cancer.

2
Background

2.1 Biological datasets

Understanding magnitude and complexity of the data are a necessity for a software development project. The following are three important consideration factors for analyzing biological datasets.

2.1.1 Complexity of biological process and methods and their effect on generated datasets

There are different biological processes and methods from which data are generated. Variations in methods are reflected on the produced datasets. Genome sequencing is the process of obtaining the order of nucleotides in a DNA fragment. Next Generation Sequencing (NGS), a new sequencing technology, provides the means for sequencing large numbers of genomes in parallel [2]. NGS technologies generate datasets in FASTQ [5] format which consist of millions of reads or small short sequences and. Each FASTQ file stores a read and its corresponding quality scores. Figure 2.1 shows a record of a FASTQ file.

```
@DHYGT8Q1:68:C05CJABXX:2:1101:2226:2186 1:N:0:ACTTGA
TGATAGGATTAGAGATCAAGCGAGAATTGTTGTTTGGGGAATTATCCAGT
+
@@@DDDDDFDFFBFD??EFCD=B)@@C?DGEECF>BBAFFGBEEIIIF<8
```

Figure 2.1. Example of a record in FASTQ format. The first line contains an identifier of the record and it always starts with @ character. The second line contains the DNA sequence, in this example, the length is 50. The third line begins with a + character and is optionally followed by the same record identifier. The fourth line contains the quality values in ASCII. This line is always the same length as the sequence line.

2.1.2 Reference databases and data formats

There are large numbers of biological databases that are publicly available. Such databases are classified into different categories. Relevant to this study is Entrez a biological data search tool maintained by the National Center for Biotechnology Information (NCBI), which provides an interface to more than 20 interconnected biological data sources of different kinds [6]. The Entrez Programming Utilities (E-utilities) are a collection of back-end programs that provide an interface into the

```
>gi|15644634:c1184409-1182706 Helicobacter pylori 26695 chromosome, complete genome
ATGAGACGGAGTTTTTTGAAAACGATTGGCTTGGGTGTGATAGCACTCTTTTTGGGTTTGTTAAACCCTT
TGAGTGCGGCGAGTTACCCCCCCCATTAAAAACACTAAAGTAGGCTTAGCCCTTTCTAGCCACCCGCTAGC
TAGTGAGATCGGGCAAAAGGTTTTAGAAGAGGGAGGTAATGCGATTGATGCGGCTGTAGCGATAGGTTTT
```

(a) Gene sequence.

```
>gi|695198868|ref|YP_009075736.1| replication protein B (plasmid) [Helicobacter
pylori]
MPRVRSELNPKRPKKFKRPSVKKSQKYLITQDNRLIYAKYGDTTANELKFFYYIISKLNPLNDKDFELCE
MPISEILGEALDHENLDANHTYIKNLCRSLSKRILEDESLVIDPETNKEDELFEVMAIFKRIQYLKKKAV
```

(b) Protein sequence.

Figure 2.2. Example of a sequence in FASTA format. The first line contains an identifier of the record and it always starts with > character. The second line contains the sequence.

Entrez query and database system NCBI [7]. Formerly sequenced genomics data of *Helicobacter pylori* are available in these online databases. GenBank [8] is a sequence database that stores gene and protein sequence of all publicly available species, including *H. pylori*, in a flat file. Appendix B shows a sample Genbank file. FASTA is another form of a text-based file in bioinformatics, which is used for storing either gene or protein sequences of a species. Figure 2.2 shows a gene and protein sequence from a FASTA file from NCBI. The first line of FASTA file contains an identifier of the record and it always starts with > character and the second line contains the gene or protein sequence.

2.2 Data analysis workflow

NGS applications generate large files that must go through multiple rounds of processing and be managed throughout. There are comprehensive sets of stand-alone software packages and tools developed in bioinformatics community [4], which are used through different steps of the analysis workflow. Pre-process, *de novo* assembly, and downstream analysis are fundamental parts of the workflow.

2.2.1 Pre-process

Biological experiments and instruments can generate incorrect data in their output files. NGS technologies are no exception in that sense. Depending on the sequencing machine they can introduce false information into a dataset [9]. This sequencing error can affect the downstream analysis. Therefore, error detection and quality control are crucial steps.

2.2.2 *De novo* Genome assembly

De novo genome assembly is the process of aligning short reads of DNA from genome sequences to find overlaps and then merging overlapping reads into continuous sequence or contigs. Assembly algorithms rely on graph structure built upon the sequencing reads, and traversal algorithms to reduce overlapping sequences [10]. There are different methods of genome assembly:

2.2.2.1 Overlap-consensus-layout method

In this method, an assembler program distinguishes overlaps between short sequences. For each identified overlaps, it merges the short sequences into longer sequences. Figure 2.3 is a simple illustration of a genome. Green segments are almost identical and overlap-consensus-layout assembler may make the mistake of connecting the blue and orange segments, and skip the red segment in between.

Figure 2.3. Error in generating assembly by Overlap-consensus-layout method because of repetition in genome.

2.2.2.2 De Bruijn graph

De Bruijn graph-based assemblers are used to solve the issue of repetitive segments in a genome which Overlap-consensus-layout method was not able to recognize. In de Bruijn graph-based method, the short reads are split into even shorter units of uniform size (k-mers) and a de Bruijn graph is constructed from those short piece. Next, the de Bruijn graph is traversed to determine its underlying genome sequence. The vertices in the graph denote k-mers, and the edges show that the neighboring k-mers overlap by exactly k-1 letters (for example, the 4-mers ACTG and CTGC share exactly three letters). Figure 2.4 is an illustration of De Bruijn graph for a short sequence.

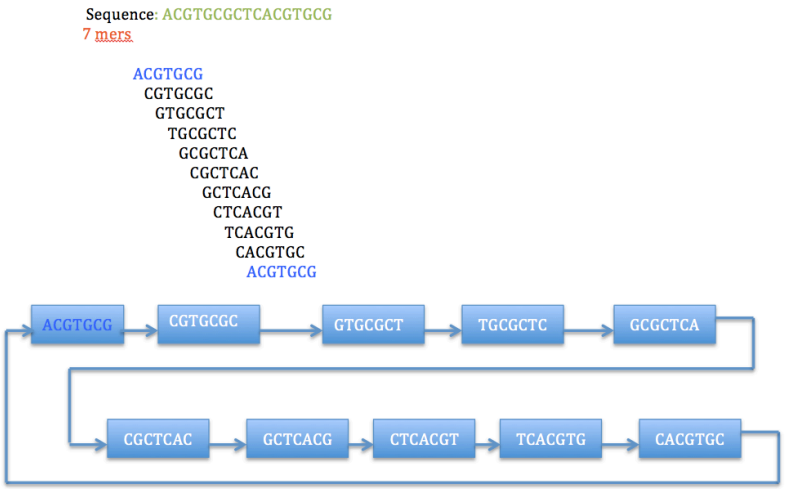


Figure 2.4. De Bruijn graph structure for a short sequence.

2.2.3 Downstream analysis

The main purpose of this step is to annotate datasets. **Annotation** is the process of attaching biological information to a gene or protein sequence. The idea is to transfer biological information from experimentally characterized datasets that are available in public databases to new, unknown datasets. It is only after this step that biological interpretation and data analysis are achievable. One form of data analysis is **Comparative genomics** which is a filed in biological research that employs computational algorithms and compares the structure of different variants of the same species to find similarities and differences. The goal of comparative genomics is to find conserved regions among the species or variations of the same species and to find which components make that species different from the others [11]. Mathematical modeling has shown probability of discovery of new genes even after sequencing hundreds of genomes of the same species [12]. To find the conserved regions, **Pan genome,** which is the union of all the genes of all different genetic variant of the species must be constructed with the aid of algorithms. The next step is to find **core genome** which are the genes that are present in all the genetic variants of the species. Pan and core genome provide the means for the biologist to study the evolution of the species.

2.3 Platform for pipeline by scripting

2.3.1 Cluster Computing and biological data management and analysis

Genome sequencing data analysis demands intensive computational resources. Each tool in different steps of analysis workflow requires data to be in particular formats or require diverse types of data to be linked together [13]. Therefore, a considerable amount of time and space is dedicated to the data reformatting and reanalyzing. Also, rigorous downstream analysis of datasets results in an incredible amount of data. Computer clusters and clustered storage systems can address this needs. Cluster computing is a form of HPC or high-performance computing. In cluster computing, multiple computers (nodes) connected, through a fast local area network and function as a single computer. Each node is configured to perform the same task, controlled by the scheduler software. Distributing tasks across these nodes is a solution for the analysis of data-intensive biological datasets.

Glenn cluster We used glenn cluster from C3SE, a center for scientific and technical computing, at the Chalmers University for this project. The operating system of this cluster is CentOS. Glenn has 379 compute nodes (total of 6080 cores) in total and 18.1 TB of RAM. Users do not have root access to the cluster. However, all the compilers and libraries are installed.

Environment module package This package provides dynamic modifications of user's environment variables. With this module system, it is possible to install

different tools needed for the pipeline and also different versions of the same tool and just switch between versions by specifying the tool/version.

2.3.2 Choice of programming language, Perl

We used perl programming language for the implementation of the pipeline. Perl has automatic memory management and extensive library collection, named CPAN, which is freely accessible. Also BioPerl [14], from Open Bioinformatics Foundation, provides libraries for biological objects and sequences. Given a very considerable part of this project was string manipulation; Perl regular expressions were a great advantage.

2.4 Pipeline by workflow management system

A workflow is a sequence of activities and tasks. Workflow management systems provide infrastructure to define and automate the execution of a workflow. A bioinformatics workflow management system is a subset of a scientific workflow management system that is designed to facilitate execution of multi-step computational data analysis in the form of tools, programs, and algorithms on biological data via a graphical user interface. Such systems are very important because they provide the ability to interconnect tools from multiple disciplines and platforms [15]. Also, workflows create detailed logs of every activity carried by the users. The central goal of creating workflows is to keep the data analysis consistent. Workflows can be used as a template for future analysis for the same types of data. Galaxy [16], Taverna [17], Chipster [18] and Yabi [19] are examples of such systems.

3

Methods

3.1 Datasets used in this project

Datasets of this project were genome sequence of *Helicobacter pylori*, generated by NGS technologies. In total, there were 52 datasets in FASTQ format with sizes varying from 278 MB to 1,6 GB. As listed in Table 3.1. From 52 datasets, we used 12 datasets as pilot data for designing the pipeline. Among the 12 patients median age was 46.5 year, range 18-60 years, and all but one of the patients were female. All the datasets were generated from patients who were recruited from Nicaragua.

3.2 Collection of 54 reference datasets from public repositories

With our reference genome collector program, we managed to download 36 completed and 18 incomplete genomes of *H. pylori* from Genbank. See section 3.4.2.1 for the details of implementation. See Table A.1 in appendix A for the list of collected genomes. We used the collected genomes for the comparative genomic analysis.

Table 3.1. Pilot *H. pylori* datasets used for designing the pipeline.

ID	File Size	Age of the patient	Sex of the patient
A	531 MB	26	F
B	778 MB	55	F
C	1003 MB	56	F
D	484 MB	58	M
E	968 MB	29	F
F	828 MB	58	F
G	588 MB	40	F
H	841 MB	18	F
2A	1.1 GB	60	F
2B	1.1 GB	53	F
2C	1.2 GB	30	F
2D	793 MB	24	F

3.3 Steps needed to realize the pipeline

Biologists of this project required studying the correlation between *H.pylori* genomes –isolated from Nicaraguan patients– and gastric cancer. This was only possible by comparing the genomics structure and characterization of each *H. pylori*. First we needed to build up the **draft genome** of each *H. pylori* from the "raw data" stored in FASTQ files. There were two main methods to achieve this. First, to align the reads to a pre-existing **reference genome**. Second, to perform *de novo* assembly on the reads. Since *H. pylori* has a very variable genome arrangement, alignment against any reference genome could lead to incorrect biased structure, and, therefore, we decided to proceed with *de novo* assembly. Based on previous studies on *de novo* assembly evaluation to get an accurate draft genome, we needed to provide good quality reads for assembly programs. Therefore, quality control of the reads prior to the assembly was the first required step of the pipeline. After assembly, we needed to evaluate the quality of the generated draft genome with the help of reference genomes. The draft genome generated by assembly programs are a set of contiguous sequences that do not have biologically sensible name. Therefore, the last main step was to annotate the draft genomes. As described earlier, because of the structure of *H. pylori* genome one reference genome is not sufficient to be used as a template for transferring gene name. So we needed to use all available reference genomes for annotation.

3.4 Pipeline by scripting

To implement the pipeline, first, we had to select the programs that were suitable to perform the tasks according to the data analysis workflow explained in the previous section and background chapter. In the case of a missing program, we had to implement it. Once we had all the programs, we developed a pipeline using perl programming language which is adopted to work on HPC. The pipeline acts as a wrapper around the programs, both the existing ones and the newly implemented ones. It coordinates the connection of all the programs and tools and also controls the flow of the information. Unlike piping in Linux, we needed to keep the files that are generated by each program in each intermediate step.
Figure 3.1 illustrates an overall view of the pipeline. More details of each step are explained in the following sections.

3.4.1 Selecting existing programs to perform steps in the pipeline

Quality control The quality scores in FASTQ format datasets indicates the probability of having an error in reads. Score 10 means that the probability of having incorrect data is 1/10 and score 20 corresponds to 1/100 and score 30 to 1 /1000. Therefore, we needed a program to 1)trim the reads with quality score cut-off of 30 and 2) filter out and remove reads shorter than 25 base length. Very short length reads can lead to an unspecific result in downstream analysis. SolexaQA v2.1 Dy-

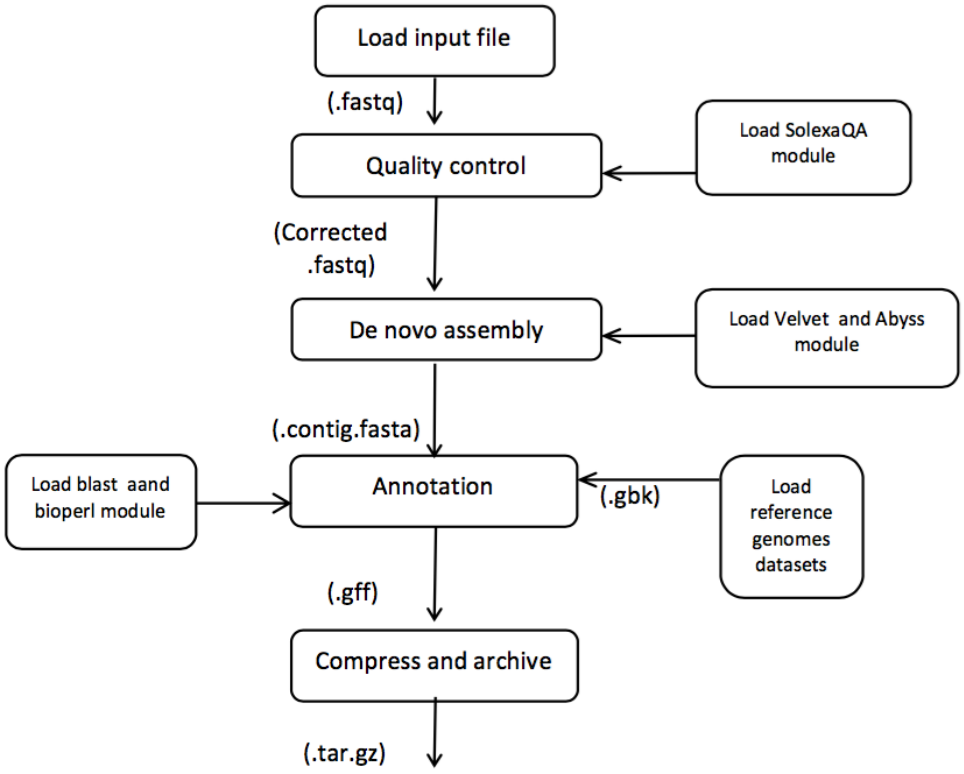


Figure 3.1. Overview of the pipeline.

namic Trim [20] is a program that is designed specifically for quality control of datasets generated by Illumina Sequencing machines, the very same one that was used to create our datasets. Therefore, for biological point of view, it was a reasonable candidate. We used time command to analyze the CPU time and elapsed time spent for pre-processing and quality control of 12 pilot datasets using SolexaQA software. We performed the tests on glenn cluster. The output of time command indicates 3 factors: Real is the total elapsed time including I/O; User is CPU time spent in the user-mode code, and Sys is CPU time spent in the kernel. Figure 3.2 shows realtime usage is approximately 30 minutes for our biggest test dataset (2C). This was a reasonable elapsed time. So we chose SolexaQA for the final pipeline.

Assembly benchmarking We chose VelvetOptimiser [21], Ray [22] and ABYSS [23] as candidate program for genome assembly since they all use de Bruijn graphs and also suitable for bacterium genome assembly. To characterize the best assembly program, we studied the draft genomes generated by all of these programs and looked into the generated contigs and also the distribution of contigs for 12 pilot datasets. We also compared the contigs with all available reference genomes to find which contigs were present in all reference genomes and which set of contigs belonged to that specific draft genome.

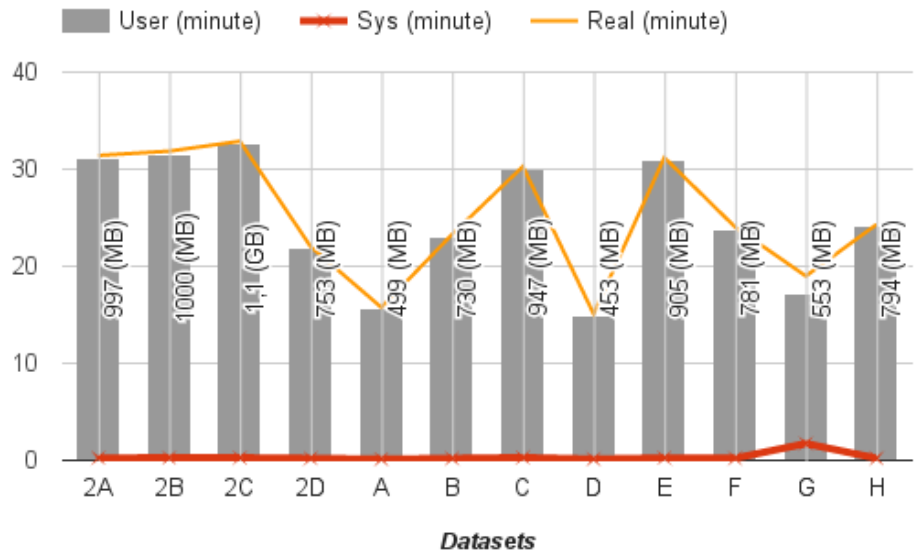


Figure 3.2. CPU time and elapsed time spent for pre-processing and quality control of dataset using SolexaQA software. Real is total elapsed time including I/O; User is CPU time spent in the user-mode code and Sys is CPU time spent in the kernel. Label on the bars shows the size of datasets.

3.4.2 Implementation of missing programs

3.4.2.1 Program for data collection from reference database

To perform an evaluation of the results of selected programs, we needed a set of reference datasets of *H. pylori*. We developed a program with perl, for automating the process of downloading all available datasets (genomes) of *H. pylori*. The program first parses the genome annotation report web page of *H. pylori* from NCBI database and looks for an identifier that is called RefSeq accession. RefSeq structure includes distinct prefix format of 2 characters (NC) followed by an underscore character ('_'). Therefore the regular expression (NC_\d{6}\. \d) can match the RefSeq identifiers. Next, the Genbank flat file for each RefSeq is downloaded and processed further by its primary tags to extract all the genes and all the proteins sequences. These sequences are then saved in DNA.fasta and Protein.fasta file respectively. (Algorithm 1)

3.4.2.2 Program for annotation

Annotation of the datasets and how to name the genes by their biologically logical name was a difficult task. *H. pylori* has around 1500 genes but only around 900 are available in all reference genomes. We implemented the annotation program by a holistic approach. First we made a global *H. pylori* reference dataset by concatenating all the reference genomes that were previously collected. Then for each dataset, we identified the genes. Then each gene was queried against the global dataset using blastp [24] software to find cross reference database identifiers of the top 20 significant matches. Then the associated biological information from these identifiers were downloaded. Furthermore, an algorithm was developed to cluster

Algorithm 1 Reference genome collection
RefSeq= Accession number: distinct prefix format of 2 characters (NC) followed by an underscore character ('_').
GB= Genbank Flat File Format,
GN=Genome Name,
OD=ObjectDescription,
CDS=Coding Region Sequence,
SF=SeqFeature

Input: Annotation report web page of *H. pylori*. **Output:** All reference datasets of *H. pylori* and a Log file that contains download date and time.

```
1: procedure REFERENCE DATASET COLLECTION
2:     Pars annotation report web page of H. pylori from NCBI.
3:     Find match with regx (NC_\d{6}\. \d)
4:     for each RefSeq do
5:         Download associated GB
6:         for each GB do
7:             Extract OD
8:             if OD eq "chromosome" then
9:                 Extract genome name, use regx s/chromosome(.*?)genome.//;
10:                Create Directory GN1.
11:                get SF
12:                if (SF.primary_tag eq "CDS") then
13:                    Extract protein_id and protein_sequence.
14:                    Create file GN1.Protein.fasta(">protein_id \n protein_sequence\n") in GN1 directory;
15:                    Extract gene_ id and sequence_string.
16:                    Create file GN1.DNA.fasta(">gene_id \n gene_sequence\n" in GN1 directory);
```

and rank this biological information. The ranking is done using result repetition. Finally, the top rated biological information were transferred to the genes. (Algorithm 2) Each dataset D is a collection of hundreds of thousands of reads R=(X, Y) where X is an identifier and Y is associated gene. The goal here is to add gene metadata to X.

3.4.3 Pan and core genome

OrthoMCL is a software for clustering proteins based on sequence similarity information [25] using the Markov cluster (MCL) algorithm. We used OrthoMCL to find protein families across all *H. pylori* reference genomes. Then we implemented a program to analyze the result from OrthoMCL output. The clusters that were present in all genomes were considered as the core genome while the total number of clusters represented the pan genome. Then we performed the same analyses for our Nicaraguan annotated sequences.

Algorithm 2 Algorithm for genome annotation

```
1: procedure ANNOTATION
2:     for each R in D do
3:         Query Y against reference genome database
4:         From result set, chose top 20 significant candidates (each candidate is a
   reference to an external database)
5:         for each Reference ID in result set do
6:             Download associated flat file format from external database
7:             Extract gene name + gene meta data
8:             Record number of observations of each gene
9:             Rank each gene. Gene with higher number of observation gets higher
   rank
10:            Choose the gene with top rank
11:            Transfer gene meta data of associated selected gene to X
```

3.4.4 Two proteins and their association with gastric cancer

cagA and babA are two proteins which are associated with increased gastric cancer risk. We downloaded the sequences of these two proteins from NCBI database and searched each Nicaraguan genome for the presence of them using blastp software.

3.4.5 Computer cluster configuration

For enabling easy software environment management on glenn cluster, environment modules package was used. First, we installed each program and then we created a modulefile that contained the information needed to configure the shell for the program (Listing 3.1).
This way we could also define if a program was dependent on any other program. Modules were useful in managing different versions of a program. Figure 3.3 shows a directory structure of sample installed programs.

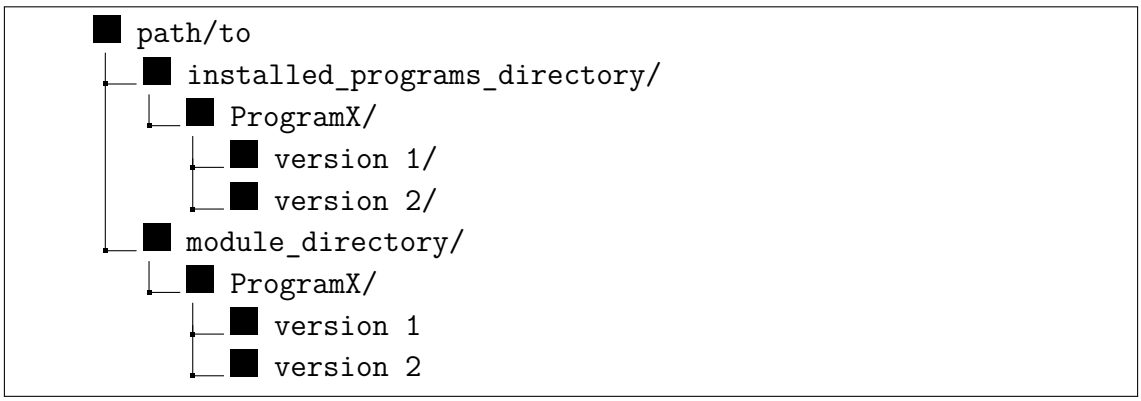


Figure 3.3. Directory listing for installed program and modulefiles

Listing 3.1. Modulefile for ProgramX

```
#%Module1.0
proc ModulesHelp { } {
puts stderr "Environment␣for␣ProgramX␣version␣1.0"
}

module-whatis "ProgramX␣version␣1.0"
prereq ProgramY CompilerZ
append-path PATH /path/to/installed_programs/ProgramX/
   version
```

To enable a program, the associated module file needs to be loaded. But first MODULEPATH environmental variable should be set to the path of directories containing modulefiles. Listing 3.2 and 3.3 shows how to Set Module path environmental variable and how to load and unload a module respectively.

Listing 3.2. Setting MODULEPATH

```
export MODULEPATH=$MODULEPATH:/path/to/module_directory
```

Listing 3.3. How to load and unload modules

```
$ module load ProgramY
$ module load CompilerZ
$ module load ProgramX/version1
$ module unload ProgramX/version1
$ module load ProgramX/version2
```

3.5 Pipeline by workflow management system, the Galaxy project

Given there are large number of bioinformatics workflow management systems, it was very hard to choose one. There has been little work performed on evaluating and comparing the features of each system, built-in functions, support for different data types, etc. Also to our knowledge, there was no workflow management system available for academic users in Sweden. Therefore to move forward, we looked at Galaxy, Taverna, Chipster and Yabi workflow usage worldwide. Figure 3.4 shows the number of scientific citations for each platform. The numbers are extracted from http://www.scopus.com.

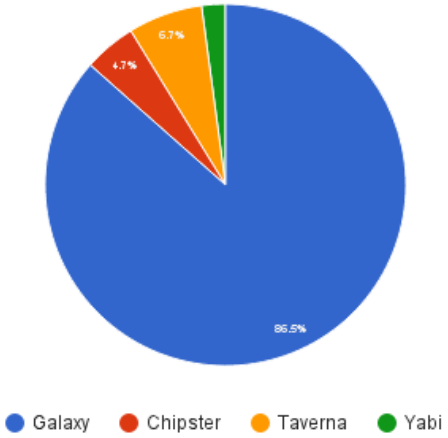


Figure 3.4. Bioinformatics workflow management system usage based on scientific citation. Galaxy is the most used framework.

3.5.1 Why Galaxy?

The Galaxy project is a free, web-based platform used for bioinformatics analysis [16] and is the most popular. Galaxy has 70 public servers from which 26 are located in Europe. (https://wiki.galaxyproject.org/PublicGalaxyServers). Galaxy has an active support system and very significant community of users and developers. Therefore, we chose Galaxy as our platform for workflow management system.

3.5.1.1 Galaxy distributions

Galaxy provides a graphical web interface so that users without advanced computational training can use this platform for data analysis. Galaxy is available in 3 primary distributions; the first form is publicly accessible servers that host Galaxy and users can freely run their analysis on these servers. However, there are some issues with these servers, such as data privacy, and also, these servers are not suitable for data-intensive analysis. The second form of Galaxy is a cloud-based privately hosted server. The cloud-based Galaxy is costly and still does not guarantee data privacy. In its third form, Galaxy offers local installation on UNIX-based computers.

3.5.1.2 Galaxy Architecture

The Galaxy framework is implemented in python programming language. A Galaxy instance includes a web server and an embedded SQLite database. Galaxy works well with the default database and web server, however, it is possible to use other types of web servers or relational databases to customize Galaxy to scale up. Galaxy comes with an install script and by running the script, first all the prerequisite Python libraries called eggs will be downloaded and installed, and then web server will be run, and the instance is accessible via a web-browser on localhost with port 8080.

3.5.1.3 A Galaxy Instance

We installed Galaxy with Python 2.6 on a Mac computer with 2 GHz processor and 8GB memory. Galaxy has an interface for authentication and authorization. An admin user can have access to these resources. Our fresh Galaxy had a default set of tools for text manipulation and visualization. Also a set of tools for converting to and from many common bioinformatics file formats. However, we needed to install more tools for the pipeline. There are two ways to add tools to Galaxy.

1. From Tool-Shed
   Galaxy has a repository of programs supported by bioinformatics community that is called Tool Shed. Installing the programs from Tool Shed is straightforward. Choosing a program will list the dependencies, and one can install all by one click. However, not all the versions of a program are available in the repository.
2. Adding custom tools
   Incorporating and installing a program that is not available in the Tool Shed is possible but needs more work. First, the tool should be copied to tools directory and then a **Tool Definition File**, which is an XML file, must be defined to specify all the command line parameters and input-output file of the tool. The final step is to **Make Galaxy aware of the new tool** by updating a config file with the tool name.

Installation of required tools to construct the workflow on local instance From Tool-Shed, we installed FastQC for quality control of the datasets and Trim Galore for trimming the reads. We installed VelvetOptimiser for *de novo* assembly. We did not manage to install Abyss the other tool for *de novo* assembly due to technical problems.

3.5.1.4 The workflow

To create a Galaxy workflow, the user can use a graphical interface that makes it possible to chain together a series of programs/tools. The user should specify how the programs should be connected and also defines the input/output file format of each program. Once the workflow is created, the user can assign the dataset(s) to be analyzed by the workflow. Uploading datasets to Galaxy instance is possible by the user interface and also via file transfer protocol (FTP).

Data analysis workflow on local instance We created a workflow of two main steps: quality control and *de novo assembly*. In the Quality control step, FastQC program is executed on datasets for quality control of the reads. Then using Trim Galore, each read with a low quality score(<30) or short reads (length<30) will be discarded. We run FastQC on datasets again to visualize the effect of read trimming. High-quality datasets from the previous step are sent to VelvetOptimiser *de novo* for assembly.

4

Results

4.1 Pipeline by scripting

4.1.1 Pre-process analysis

Figure 4.1 shows the effect of trimming on the number of reads. The difference in the number of reads before and after trimming is not significant which shows most of the reads were of good quality and had not been affected by the trimming program.

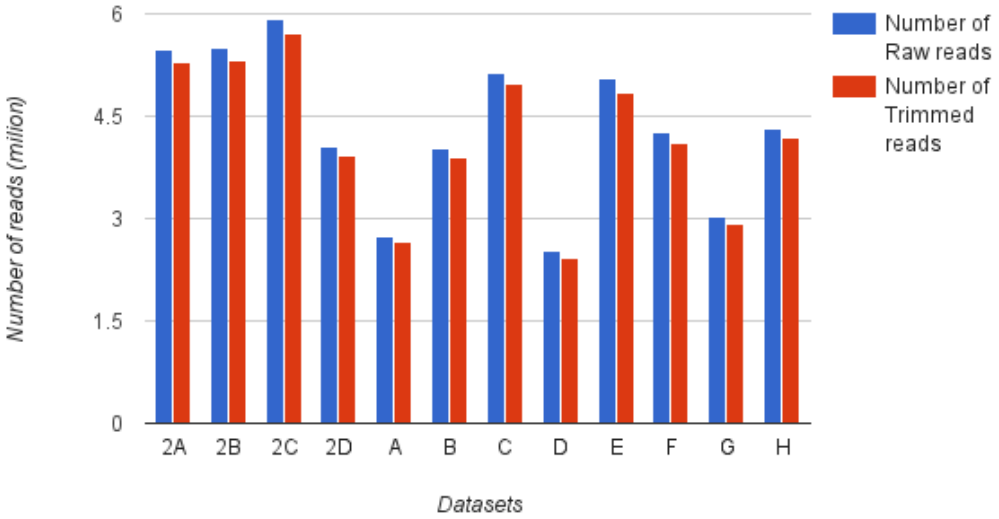


Figure 4.1. Number of reads (million) before and after trimming of low-quality reads.

4.1.2 Assembly benchmarking results

Figure 4.2 shows the difference between assembly programs regarding the number of contigs. A large number of contigs usually indicates a mis-assembled genome. The assembly program which generates the lowest number of contigs is Abyss. Ray creates too many contigs for C and G dataset.
Figure 4.3 illustrates distribution of contigs of different size for each dataset by different assembly programs. Most of contigs generated by Ray are small contigs of size 20 kilo base pair (KBP) and again Abyss shows the better result of generating not so many small size contigs and also generating most of the big contigs of size 80 to 100 (KBP). We eventually decided to use both VelvetOptimiser and Abyss for assembling the genomes.

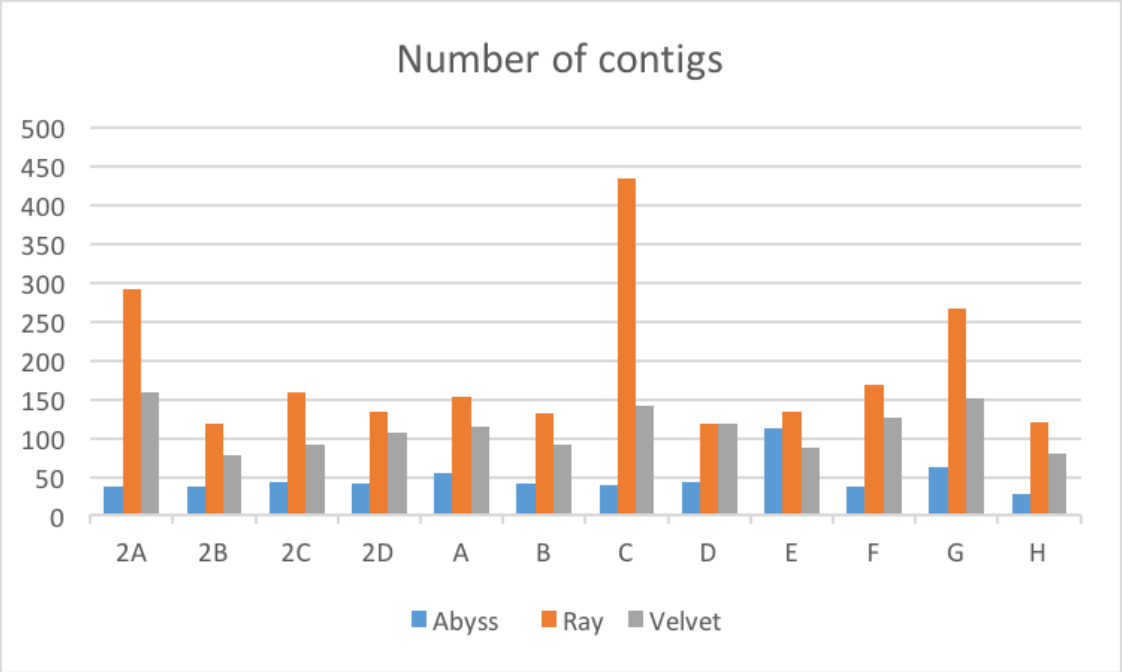


Figure 4.2. Number of contigs generated by each assembly program.

4.1.2.1 Characterizations of 52 *H. pylori* draft genomes from Nicaraguan patients

With the pipeline by scripting, we managed to characterize the 12 pilot datasets (*H. pylori* genomes of 12 Nicaraguan patients) successfully. Therefore, we used the pipeline for the rest of the datasets and in total, we succeeded to characterize 52 *H. pylori* genomes of Nicaraguan patients.

4.1.3 Pan genome and core genome analysis

We constructed the pan genome or the union of all the genes for our pilot datasets (the *H. pylori* genomes of 12 Nicaraguan patients). The pan genome consisted of 1787 clusters and out of these 1240 clusters were present in all 12 genomes, i.e. 69.4 percent of the pan genome (Figure 4.4). We also constructed the pan genome of all *H. pylori* genomes which we have collected from Genbank. See Table A.1 in Appendix A for the list. The core genome is the number of genes that are present in all the genomes, the shell genome is the number of genes that are shared among at least 75 percent of the genomes, and the pan genome is the total number of genes within all the analyzed genomes. The total pan genome of (36 completed, 18 incomplete and 12 Nicaraguan genomes) consisted of 3035 clusters. Within the pan genome, we identified a core of 346 clusters, 11.4 percent of the total clusters (Figure 4.5). Thirty-four of the clusters were unique for the Nicaraguan collection, while 1249 clusters that were present in other genomes were absent in the 12 Nicaraguans.

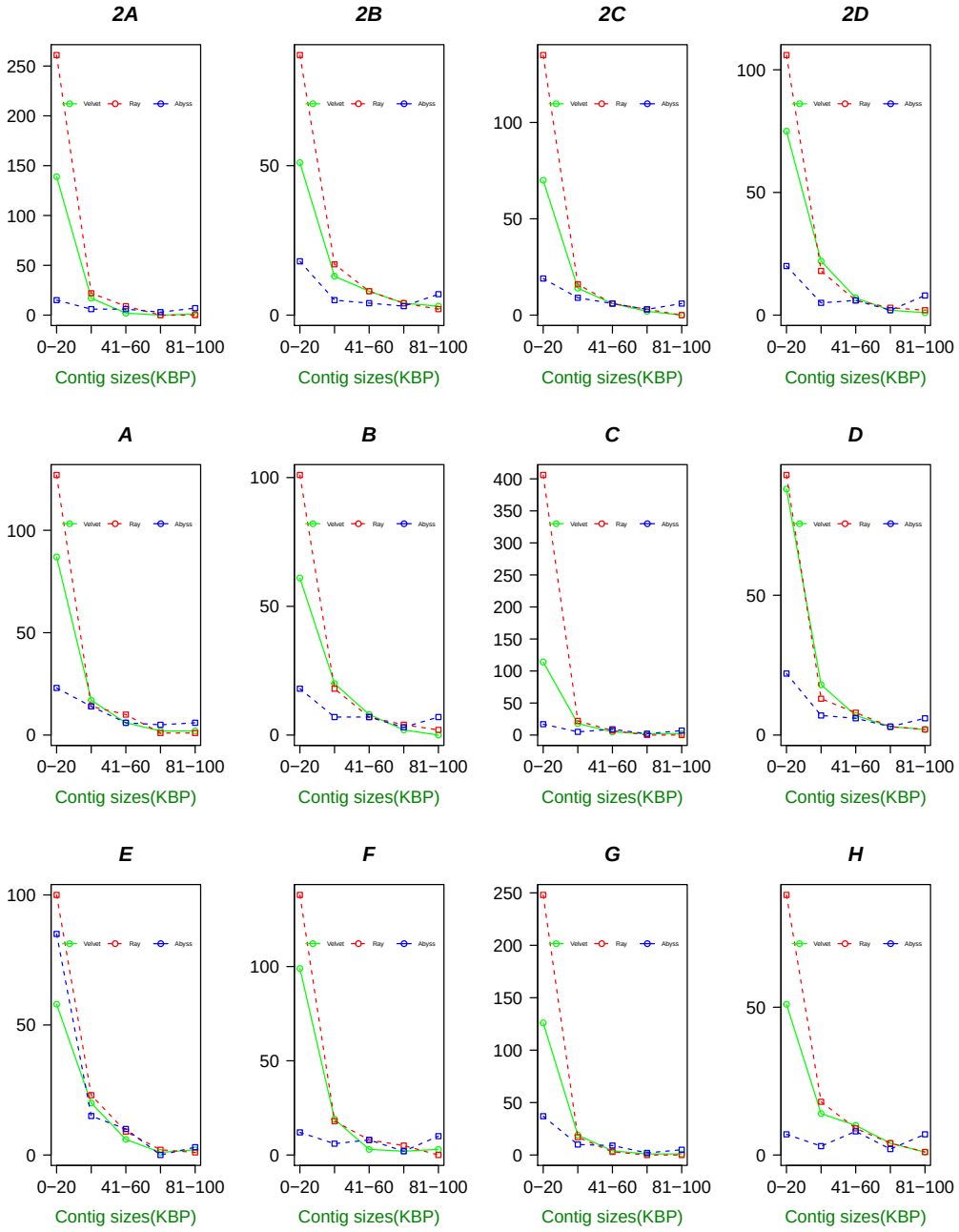


Figure 4.3. Contig length distribution by each assembly program. The x-axis shows contig size in KBP and the y-axis shows the number of contigs.

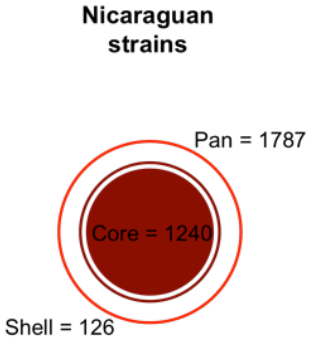


Figure 4.4. Pan genome and core genome comparisons within the 12 Nicaraguan *H. pylori* genomes. The pan genome consisted of 1787 clusters and out of these 1240 clusters were present in all 12 genomes, i.e. 69.4 percent of the pan genome.

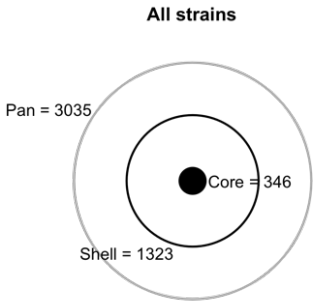


Figure 4.5. Pan genome and core genome comparisons within the 12 Nicaraguan genomes and within the publicly available *H. pylori* sequences from Genbank. Within the pan genome, we identified a core of 346 clusters, 11.4 percent of the total clusters.

4.1.4 Two proteins and their association with gastric cancer

Proteins cagA and babA are associated with increased gastric cancer risk. Our analysis showed that 37 patients out of the 52 in the study carry both cagA and babA, 3 carry only cagA, 10 carry only babA, and two carry neither.

4.2 Pipeline by workflow management system

Figure 4.6 shows the workflow which we constructed on our Galaxy instance.

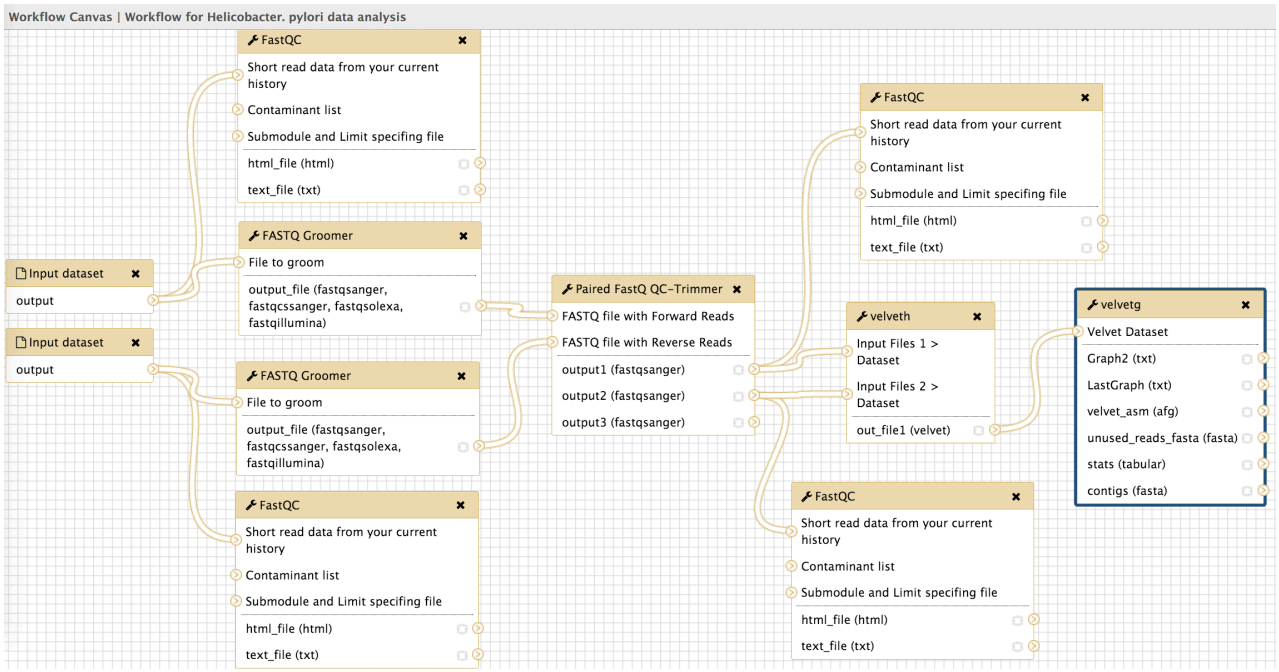


Figure 4.6. A Galaxy workflow for data analysis

5

Conclusion

Computer science has influenced biological science and cutting-edge research. Big biological data produced by high-throughput technologies have volume and variety from the three V's of big data, volume, velocity and variety. Analysis of such datasets requires robust algorithms. Piecing together and managing these steps is a prominent challenge. Choosing a platform to withhold the data and is in harmony with the analysis pipeline is another challenge.

To construct the fundamental elements of the pipeline we relied on the tools that are already developed in the bioinformatics community. We did not want to invent the wheel in each and every step. Bioinformatics has a very active and dynamic community. There could exist a handful of programs that are designed to perform the very same task. Based on what we experienced through this project, the most challenging task is to find the best program that firstly supports the data of your interest and secondly uses reasonable computational resources and time to perform and finally produces biologically meaningful result. Since each program was developed independently of the language we used for developing the pipeline, configuring each program and making it work with the pipeline was the other issue. This included contacting authors for reporting issues and waiting for the new versions of the tools and in case of no reply, changing the source code of the tool if there was enough documentation.

Scripting on computer cluster vs. workflow management system Since we did not have access to an instance of Galaxy on a computer cluster, any comparison in performance with the pipeline on computer cluster was not possible.

Pipeline on computer cluster In developer's point of view, the advantage of scripting method is the flexibility in design. With scripting, it is possible to perform sophisticated analysis, use APIs of databases to fetch data and to incorporate them into the pipeline, dynamically add tools and to implement the missing elements. In end user perspective, once the pipeline is implemented, performing the analysis is possible by providing the input files and cluster's scheduler for job submission. However, clusters often do not have a graphical user interface (GUI) but rather use a text-based command line interface (CLI). The majority of the life scientists who work with high-throughput data have limited or no knowledge of computer science. Therefore, working with command line interface could be a disadvantage from their point of view.

Workflow on workflow management system, Galaxy Administrator of such system is dependent on Galaxy team when it comes to incompatibility issues, updates problems, new releases, etc. Installing tools from Toolshed can be problematic as well. There are many versions of the same tool and is not trivial which version to use. For the tools that cannot be installed easily, there are Galaxy wrappers. However not every tool has a wrapper. From end user perspective, once all the desired tools are installed, it is possible to create workflows without prior knowledge in programming. Once the workflow is created, it is rather easy to upload files and run the analysis. Galaxy as a web-based platform provides the means for reproducibility of the research. Scientists can share the whole analysis workflow and datasets.

Research outcome driven by pipeline With the aim of the implemented pipeline by scripting and on the computer cluster, we managed to present and characterize the first whole-genome sequence of *H. pylori* from Nicaragua. By analyzing the draft genomes, we found most of the patients in this study carry two proteins which are known to have a role in gastric cancer. These results are a rather small part of the scientific research result driven from our pipeline. This proves how such pipelines can help biological scientist manage and analysis big biological datasets.

Potential ethical consequences of the performed project Approximately half of the world's population have an infection with *H. pylori*. Most people remain without any symptoms and within those 10-15 percent of the carriers develop gastric ulcers, and a few percent develops gastric cancer [1]. The automated pipelines in this project are used by scientists for analysing patients who carry *H. pylori* and is hopefully a contribution to human health by accelerating analysis cycle. We assured the security of data and analysis process by not using the public servers and performing the analysis on the computer cluster with restricted access. Also, the pipeline facilitates reproducibility of the result which addresses the reliability issues on research outcome.

5.1 Future work

Virtualization can be a way forward for this projects. For instance, the Docker software is a platform that can be used to run, execute, and distribute virtualized environments in the form of system images and containers [26]. Docker is available for different platforms. Windows and OSX operating systems. When Docker is installed, the user can download a pre-configured image or container. This container can be a galaxy server or any other server with a bioinformatics pipeline. Docker can be used as a safe way to allow multiple users access to a shared resource. While it is fair to say users can access the same computer cluster, it can also be a problem from an administrative point of view, to manage user permissions or give access to users.

Bibliography

[1] Polk DB, Peek RM. *Helicobacter pylori*: gastric cancer and beyond. Nat Rev Cancer. 2010;10:403–414. doi:10.1038/nrc2857.

[2] Wandelt S, Rheinländer A, Bux M, Thalheim L, Haldemann B, Leser U. Data Management Challenges in Next Generation Sequencing. Datenbank-Spektrum. 2012;12:161–171. doi:10.1007/s13222-012-0098-2.

[3] Li J, Doyle MA, Saeed I, Wong SQ, Mar V, Goode DL, et al. Bioinformatics pipelines for targeted resequencing and whole-exome sequencing of human and mouse genomes: a virtual appliance approach for instant deployment. PLoS One. 2014;9:95217. doi:10.1371/journal.pone.0095217.

[4] Pabinger S, Dander A, Fischer M, Snajder R, Sperk M, Efremova M, et al. A survey of tools for variant analysis of next-generation genome sequencing data. Brief Bioinform. 2014;15:256–278. doi:10.1093/bib/bbs086.

[5] Cock PJA, Fields CJ, Goto N, Heuer ML, Rice PM. The Sanger FASTQ file format for sequences with quality scores, and the Solexa/Illumina FASTQ variants. Nucleic Acids Res. 2010;38:1767–1771. doi:10.1093/nar/gkp1137.

[6] Ram S, Wei W. Understanding Semantic Links among Heterogeneous Biological Data Sources. In 15th Workshop on Information Technology and Systems, WITS 2005 University of Arizona. 2005;p. 123–128.

[7] Entrez Programming Utilities Help, Bethesda (MD): National Center for Biotechnology Information;. Available from: `http://www.ncbi.nlm.nih.gov/books/NBK25501/`.

[8] Benson DA, Mizrachi IK, Clark K, Lipman DJ, Ostell J, Sayers EW. GenBank. Nucleic Acids Res. 2013;40:48–53. doi:10.1093/nar/gkr1202.

[9] Fujimoto MS, Bodily PM, Okuda N, Clement MJ, Snell Q. Effects of error-correction of heterozygous next-generation sequencing data. BMC Bioinformatics. 2014;15:Suppl 7. doi:10.1186/1471-2105-15-S7-S3.

[10] Jünemann S, Prior K, Albersmeier A, Albaum S, Kalinowski J, Goesmann A, et al. GABenchToB: a genome assembly benchmark tuned on bacteria and benchtop sequencers. PLoS One. 2014;9:e107014. doi:10.1371/journal.pone.0107014.

[11] Touchman J. Comparative Genomics. Nature Education. 2010;3:13.

[12] Medini M, Donati C, Tettelin H, Masignani V, Rappuoli R. The microbial pan-genome. Curr Opin Genet Dev. 2005;6:589–594. doi:10.1016/j.gde.2005.09.006.

[13] Schadt EE, Linderman MD, Sorenson J, Lee L, Nolan GP. Computational solutions to large-scale data management and analysis. Nat Rev Genet. 2010;11:647–657. doi:10.1038/nrg2857.

[14] Stajich JE, Block D, Boulez K, Brenner SE, Chervitz SA, Dagdigian C, et al. The Bioperl toolkit: Perl modules for the life sciences. Genome Res. 2002;12:1611–1618. doi:10.1101/gr.361602.

[15] Hwee J. Evaluating virtualization and workflow management systems for biomedical applications to increase computational scientific reproducibility. Scholar Archive. 2015;11:1181–1185. doi:10.6083/M49P30MQ.

[16] Goecks J, Nekrutenko A, Taylor J, The Galaxy Team. Galaxy: a comprehensive approach for supporting accessible, reproducible, and transparent computational research in the life sciences. Genome Biol. 2010;8:86. doi:10.1186/gb-2010-11-8-r86.

[17] Wolstencroft K, Haines R, Fellows D, Williams A, Withers D, Owen S, et al. The Taverna workflow suite: designing and executing workflows of Web Services on the desktop, web or in the cloud. Nucleic Acids Res. 2013;41:557–561. doi:10.1093/nar/gkt328.

[18] Kallio MA, Tuimala JT, Hupponen T, Klemelä P, Gentile M, Scheinin I, et al. Chipster: user-friendly analysis software for microarray and other high-throughput Datenbank-Spektrum. BMC Genomics. 2011;12:507. doi:10.1186/1471-2164-12-507.

[19] Hunter AA, Macgregor AB, Szabo TO, Wellington CA, Bellgard MI. Yabi: An online research environment for grid, high performance and cloud computing . Biology and Medicine. 2011;7:1. doi:10.1186/1751-0473-7-1.

[20] Cox MP, Peterson DA, Biggs PJ. At-a-glance quality assessment of Illumina second-generation sequencing data. BMC Bioinformatics. 2010;11:485. doi:10.1186/1471-2105-11-485.

[21] Zerbino DR, Birney E. Velvet: Algorithms for de novo short read assembly using de bruijn graphs. Genome Res. 2008;18:821–829. doi:10.1101/gr.074492.107.

[22] Boisvert S, Laviolette F, Corbeil J. Ray: simultaneous assembly of reads from a mix of high-throughput sequencing technologies. Comput Biol. 2010;11:1519–1533. doi:10.1089/cmb.2009.0238.

[23] Simpson JT, Wong K, Jackman SD, Schein JE, Jones SJM, Birol İ. Abyss: A parallel assembler for short read sequence data. Genome Res. 2009;6:1117–1123. doi:10.1101/gr.089532.108.

[24] Altschul SF, Gish W, Miller W, Myers EW, Lipman DJ. Basic local alignment search tool . J Mol Biol. 1990;3. doi:10.1016/S0022-2836(05)80360-2.

[25] Fischer S, Brunk BP, Chen F, Gao X, Harb OS, Iodice JB, et al. Using OrthoMCL to assign proteins to OrthoMCL-DB groups or to cluster proteomes into new ortholog groups . Curr Protoc Bioinformatics. 2011;Chapter 6:Unit 6.12:1–19. doi:10.1002/0471250953.bi0612s35.

[26] Docker: the container engine;. Available from: `https://github.com/docker/docker`.

A

Appendix 1 List of reference genomes of *H. pylori*

Table A.1. Table 2 *H. pylori*, 36 complete 18 incomplete genomes.

Organism	Accession number	Number of contigs[c]
H. pylori Cuz20	NC_017358.1	c
H. pylori 908	NC_017357.1	c
H. pylori 83	NC_017375.1	c
H. pylori Shi169	NC_017740.1	c
H. pylori Shi417	NC_017739.1	c
H. pylori XZ274	NC_017926.1	c
H. pylori G27	NC_011333.1	c
H. pylori HPAG1	NC_008086.1	c
H. pylori B38	NC_012973.1	c
H. pylori F30	NC_017365.1	c
H. pylori Shi112	NC_017741.1	c
H. pylori Shi470	NC_010698.2	c
H. pylori PeCan18	NC_017742.1	c
H. pylori Sat464	NC_017359.1	c
H. pylori v225d	NC_017355.1	c
H. pylori HUP B14	NC_017733.1	c
H. pylori India7	NC_017372.1	c
H. pylori J99	NC_000921.1	c
H. pylori P12	NC_011498.1	c
H. pylori Puno135	NC_017379.1	c
H. pylori 26695	NC_000915.1	c
H. pylori PeCan4	NC_014555.1	c
H. pylori SJM180	NC_014560.1	c
H. pylori SouthAfrica7	NC_017361.1	c
H. pylori Puno120	NC_017378.1	c
H. pylori ELS37	NC_017063.1	c
H. pylori SNT49	NC_017376.1	c
H. pylori 2017	NC_017374.1	c
H. pylori F32	NC_017366.1	c
H. pylori Gambia94 24	NC_017371.1	c
H. pylori F16	NC_017368.1	c
H. pylori 2018	NC_017381.1	c
H. pylori Lithuania75	NC_017362.1	c
H. pylori F57	NC_017367.1	c
H. pylori 51	NC_017382.1	c
H. pylori B8	NC_014256.1	c
H. pylori 9810	ABSX00000000.1	51
H. pylori B128	ABSY00000000.1	73
H. pylori B45	AFAO00000000.1	63
H. pylori NAB47	AJFA00000000.2	385
H. pylori NAD1	AJGJ00000000.2	400
H. pylori NCTC 11637 = CCUG 17874	AIHX00000000.1	163
H. pylori NQ4044 NQ4044	AKNW00000000.1	17
H. pylori NQ4053 NQ4053	AKNV00000000.1	6
H. pylori NQ4076 NQ4076	AKNX00000000.1	4
H. pylori NQ4099 NQ4099	AKNU00000000.1	6
H. pylori NQ4110 NQ4110	AKNZ00000000.1	3
H. pylori NQ4161 NQ4161	AKNY00000000.1	9
H. pylori NQ4200 NQ4200	AKNS00000000.1	14
H. pylori NQ4216 NQ4216	AKNR00000000.1	13
H. pylori NQ4228 NQ4228	AKNT00000000.1	6
H. pylori P79	AIHW00000000.1	215
H. pylori HPKX_438_AG0C1	ABJO00000000.1	2602
H. pylori HPKX_438_CA4C1	ABJP00000000.1	3766

[c] indicates complete genome.

B

Appendix 2 Genbank flat file

```
LOCUS       SCU49845     5028 bp    DNA             PLN       21-JUN-1999
DEFINITION  Saccharomyces cerevisiae TCP1-beta gene, partial cds, and Axl2p
            (AXL2) and Rev7p (REV7) genes, complete cds.
ACCESSION   U49845
VERSION     U49845.1  GI:1293613
KEYWORDS    .
SOURCE      Saccharomyces cerevisiae (baker's yeast)
  ORGANISM  Saccharomyces cerevisiae
            Eukaryota; Fungi; Ascomycota; Saccharomycotina; Saccharomycetes;
            Saccharomycetales; Saccharomycetaceae; Saccharomyces.
REFERENCE   1  (bases 1 to 5028)
  AUTHORS   Torpey,L.E., Gibbs,P.E., Nelson,J. and Lawrence,C.W.
  TITLE     Cloning and sequence of REV7, a gene whose function is required for
            DNA damage-induced mutagenesis in Saccharomyces cerevisiae
  JOURNAL   Yeast 10 (11), 1503-1509 (1994)
  PUBMED    7871890
REFERENCE   2  (bases 1 to 5028)
  AUTHORS   Roemer,T., Madden,K., Chang,J. and Snyder,M.
  TITLE     Selection of axial growth sites in yeast requires Axl2p, a novel
            plasma membrane glycoprotein
  JOURNAL   Genes Dev. 10 (7), 777-793 (1996)
  PUBMED    8846915
REFERENCE   3  (bases 1 to 5028)
  AUTHORS   Roemer,T.
  TITLE     Direct Submission
  JOURNAL   Submitted (22-FEB-1996) Terry Roemer, Biology, Yale University, New
            Haven, CT, USA

FEATURES             Location/Qualifiers
     source          1..5028
                     /organism="Saccharomyces cerevisiae"
                     /db_xref="taxon:4932"
                     /chromosome="IX"
                     /map="9"
     CDS             <1..206
                     /codon_start=3
                     /product="TCP1-beta"
                     /protein_id="AAA98665.1"
                     /db_xref="GI:1293614"
                     /translation="SSIYNGISTSGLDLNNGTIADMRQLGIVESYKLKRAVVSSASEA
                     AEVLLRVDNIIRARPRTANRQHM"
     gene            687..3158
                     /gene="AXL2"
     CDS             687..3158
                     /gene="AXL2"
                     /note="plasma membrane glycoprotein"
                     /codon_start=1
                     /function="required for axial budding pattern of S.
                     cerevisiae"
                     /product="Axl2p"
                     /protein_id="AAA98666.1"
                     /db_xref="GI:1293615"
                     /translation="MTQLQISLLLTATISLLHLVVATPYEAYPIGKQYPPVARVNESF
                     TFQISNDTYKSSVDKTAQITYNCFDLPSWLSFDSSSRTFSGEPSSDLLSDANTTLYFN
                     VILEGTDSADSTSLNNTYQFVVTNRPSISLSSDFNLLALLKNYGYTNGKNALKLDPNE
                     VFNVTFDRSMFTNEESIVSYYGRSQLYNAPLPNWLFFDSGELKFTGTAPVINSAIAPE
                     TSYSFVIIATDIEGFSAVEVEFELVIGAHQLTTSIQNSLIINVTDTGNVSYDLPLNYV
                     YLDDDPISSDKLGSINLLDAPDWVALDNATISGSVPDELLGKNSNPANFSVSIYDTYG
                     DVIYFNFEVVSTTDLFAISSLPNINATRGEWFSYYFLPSQFTDYVNTNVSLEFTNSSQ
                     DHDWVKFQSSNLTLAGEVPKNFDKLSLGLKANQGSQSQELYFNIIGMDSKITHSNHSA
```

```
                     NATSTRSSHHSTSTSSYTSSTYTAKISSTSAAATSSAPAALPAANKTSSHNKKAVAIA
                     CGVAIPLGVILVALICFLIFWRRRRENPDDENLPHAISGPDLNNPANKPNQENATPLN
                     NPFDDDASSYDDTSIARRLAALNTLKLDNHSATESDISSVDEKRDSLSGMNTYNDQFQ
                     SQSKEELLAKPPVQPPESPFFDPQNRSSSVYMDSEPAVNKSWRYTGNLSPVSDIVRDS
                     YGSQKTVDTEKLFDLEAPEKEKRTSRDVTMSSLDPWNSNISPSPVRKSVTPSPYNVTK
                     HRNRHLQNIQDSQSGKNGITPTTMSTSSSDDFVPVKDGENFCWVHSMEPDRRPSKKRL
                     VDFSNKSNVNVGQVKDIHGRIPEML"
     gene            complement(3300..4037)
                     /gene="REV7"
     CDS             complement(3300..4037)
                     /gene="REV7"
                     /codon_start=1
                     /product="Rev7p"
                     /protein_id="AAA98667.1"
                     /db_xref="GI:1293616"
                     /translation="MNRWVEKWLRVYLKCYINLILFYRNVYPPQSFDYTTYQSFNLPQ
                     FVPINRHPALIDYIEELILDVLSKLTHVYRFSICIINKKNDLCIEKYVLDFSELQHVD
                     KDDQIITETEVFDEFRSSLNSLIMHLEKLPKVNDDTITFEAVINAIELELGHKLDRNR
                     RVDSLEEKAEIERDSNWVKCQEDENLPDNNGFQPPKIKLTSLVGSDVGPLIIHQFSEK
                     LISGDDKILNGVYSQYEEGESIFGSLF"

ORIGIN
        1 gatcctccat atacaacggt atctccacct caggtttaga tctcaacaac ggaaccattg
       61 ccgacatgag acagttaggt atcgtcgaga gttacaagct aaaacgagca gtagtcagct
      121 ctgcatctga agccgctgaa gttctactaa gggtggataa catcatccgt gcaagaccaa
      181 gaaccgccaa tagacaacat atgtaacata tttaggatat acctcgaaaa taataaaccg
      241 ccacactgtc attattataa ttagaaacag aacgcaaaaa ttatccacta tataattcaa
      301 agacgcgaaa aaaaaagaac aacgcgtcat agaacttttg gcaattcgcg tcacaaataa
      361 attttggcaa cttatgtttc ctcttcgagc agtactcgag ccctgtctca agaatgtaat
      421 aatacccatc gtaggtatgg ttaaagatag catctccaca acctcaaagc tccttgccga
      481 gagtcgccct cctttgtcga gtaattttca cttttcatat gagaacttat tttcttattc
      541 tttactctca catcctgtag tgattgacac tgcaacagcc accatcacta gaagaacaga
      601 acaattactt aatagaaaaa ttatatcttc ctcgaaacga tttcctgctt ccaacatcta
      661 cgtatatcaa gaagcattca cttaccatga cacagcttca gatttcatta ttgctgacag
      721 ctactatatc actactccat ctagtagtgg ccacgcccta tgaggcatat cctatcggaa
      781 aacaataccc cccagtggca agagtcaatg aatcgtttac atttcaaatt tccaatgata
      841 cctataaatc gtctgtagac aagacagctc aaataacata caattgcttc gacttaccga
      901 gctggctttc gtttgactct agttctagaa cgttctcagg tgaaccttct tctgacttac
      961 tatctgatgc gaacaccacg ttgtatttca atgtaatact cgagggtacg gactctgccg
     1021 acagcacgtc tttgaacaat acataccaat ttgttgttac aaaccgtcca tccatctcgc
     1081 tatcgtcaga tttcaatcta ttggcgttgt taaaaaacta tggttatact aacggcaaaa
     1141 acgctctgaa actagatcct aatgaagtct tcaacgtgac ttttgaccgt tcaatgttca
     1201 ctaacgaaga atccattgtg tcgtattacg gacgttctca gttgtataat gcgccgttac
     1261 ccaattggct gttcttcgat tctggcgagt tgaagtttac tgggacggca ccggtgataa
     1321 actcggcgat tgctccagaa acaagctaca gttttgtcat catcgctaca gacattgaag
     1381 gattttctgc cgttgaggta gaattcgaat tagtcatcgg ggctcaccag ttaactacct
     1441 ctattcaaaa tagtttgata atcaacgtta ctgacacagg taacgtttca tatgacttac
     1501 ctctaaacta tgtttatctc gatgacgatc ctatttcttc tgataaattg ggttctataa
     1561 acttattgga tgctccagac tgggtggcat tagataatgc taccatttcc gggtctgtcc
     1621 cagatgaatt actcggtaag aactccaatc ctgccaattt ttctgtgtcc atttatgata
     1681 cttatggtga tgtgatttat ttcaacttcg aagttgtctc cacaacggat ttgtttgcca
     1741 ttagttctct tcccaatatt aacgctacaa ggggtgaatg gttctcctac tatttttgc
     1801 cttctcagtt tacagactac gtgaatacaa acgtttcatt agagtttact aattcaagcc
     1861 aagaccatga ctgggtgaaa ttccaatcat ctaatttaac attagctgga gaagtgccca
     1921 agaatttcga caagctttca ttaggtttga aagcgaacca aggttcacaa tctcaagagc
     1981 tatattttaa catcattggc atggattcaa agataactca ctcaaaccac agtgcgaatg
     2041 caacgtccac aagaagttct caccactcca cctcaacaag ttcttacaca tcttctactt
```