RAPPORT DE STAGE

Intern Python Developer

Name: Eric Palanques Tost
Tutor at the university: Aurélien Serandour
Tutor at the company: Julián Atienza
Company: Oxford Nanopore Technologies
Address of the company: Gosling Building, Oxford Science Park,
Edmund Halley Rd,
OX4 4DQ, Oxford, United Kingdom
Dates: from the 09/04/2018 to the 31/08/2018

1. Index

This report summarizes the work developed during an internship at the company Oxford Nanopore Technologies, in Oxford, United Kingdom. The internship had a duration of 21 weeks beginning on Monday the 9th of April 2018 and finishing on Friday the 31st August 2018.

2. Acknowledgements

I would like to thank my tutor at the company Julián Atienza for trusting me and teaching the workflow of a computer scientist.

I would also like to thank Amber Wright for giving me the opportunity to work in a Bioinformatics project, a field for which I have found a passion.

I would also like to thank Steven Pool, my senior coworker, for helping me out numerous times and teaching me proper coding abilities

Finally, I would like to thank the teachers of the BioSTIC option at the Ecole Centrale de Nantes: Sophie Limou, Aurelien Serandour and Olivier Roux for making me discover the world of genetics and bioinformatics, that has become a passion a probably the field of my future jobs.

3. Glossary

- **Nanopore**: a small pore (hole) with a size of just some nanometers (10^{-9} m). It is formed by nanopore-forming proteins.

- **Flow cell**: it is an of electric system composed by two liquids with different ionic charge and separated by a membrane. The difference of ionic charge between the two liquids generates a voltage that can be measured.

- **MinION**: the main device commercialized by Oxford Nanopore Technologies. It is a small portable sequencer (105x23x33mm of size and 90g of weight) that uses a flow cell traversed by nanopores. When a DNA or RNA strand passes through the pore, the ionic equilibrium between the two layers of the membrane is disrupted, creating a change in the voltage and therefore a change in the electric signal. This change is characteristic of every few bases so the nucleotidic sequence can be reconstructed from the electric signal.

- **MinKnow:** It's the software used by the MinION. It allows the connection with the computer at a software level. Every interaction with the MinION is done through MinKnow.

- **Bream**: it is a package allowing a set of instructions for the device plus a performance checking. It interacts with the device using MinKnow. It is needed when using Mk Core Connector.

- **Mk Manager**: it is manager for MinKnow. It streams information of the devices connected to the computer and can expose this information for remote use through a port.

- **Port:** it is an endpoint for wireless or physical communication with a device. For example, a USB port is a physical port to connect to other devices, and the port 80 is the port through which HTTP connections are made. In the case of the MinION, MinKnow exposes its Mk Manager through the port 9500, and the gUI is exposed through the port 8080.

- **Mk Core Connector**: It's the name of the back-end project of my internship. It is used to control several MinIONs attached to different computers from a single administrator computer.

- **gUI (for Mk Core Connector)**: gUI stands for Graphical User Interface. It is a package that gives graphical support for a software, in this case Mk Core Connector. Instead of running Mk Core Connector from the terminal, the user has a web page with buttons, dropdowns and other HTML elements that facilitate the use of Mk Core Connector.

- **gRPC:** stands for Google Remote Procedure Call. It is a technology to connect two devices together. One device will act as a server, that will execute commands, and the other will act as a client, that will tell the servers to run commands and send back the result. It uses files called Protocol Buffers (abbreviated as Protobuff) to write the messages between the two computers. It is focused on speed and efficiency.

- **WebSocket**: like gRPC, it is a way to connect devices together wirelessly. One device will act as server and the others will act as clients. Unlike gRPC, the server will be able to send messages to the clients whenever there is an update to the information they request. This makes it efficient, because the client doesn't have to constantly ask the server for the current state of the information. This is great for specific uses, like the connection to the Mk Manager, but it is slower for connections that won't be updated, like the regular connections with the MinION.

- **Metagenomic sample:** A sample containing a mix of genomes from different species.

- **Host depletion**: Consists in identifying the genome of a particular specie inside a metagenomic sample. Normally this is followed by the separation of the host genome from the sample.

- **Hash:** a function that rewrites a character chain into a fixed length identifier. No matter the length of the original character chain, the hash will always have the same length.

- **MinHash technique**: A method used to detect similarities between two entities. It uses the probability that the two entities share the same hashes to determine its similarity rate.

- **HDF5:** it is a technology that allows to store very large data objects hierarchically. Its hierarchical structure allows easy access to the data. This is the format used to store the output data of the MinION.

- **SQLlite:** A technology for data storage. It allows complex requests and storage of the databases on a remote server.

- **BULK file:** a file created on the database performance project for a better accessibility to the data generated by the MinION.

4. Résumé en Français

4.1. L'entreprise et le MinION

J'ai effectué un stage dans la société Oxford Nanopore Technologies, basée à Oxford, au Royaume-Uni. Oxford Nanopore Technologies produit des séquenceurs génétiques et fait désormais partie des leaders de la nouvelle génération de séquençage. Son bureau principal est situé à Oxford (Royaume-Uni), mais elle a aussi des bureaux aux États-Unis, en Chine et au Japon, ainsi qu'une forte présence commerciale en Inde, en Allemagne et en France. Actuellement, elle compte environ 400 travailleurs.

Le principal produit de cette société est un séquenceur portable en temps réel appelé MinION formé par une cellule de flux (Flow Cell) recouverte de milliers de petits pores (nanopores, d'óu le nom de la société). L'équilibre ionique entre les deux parties de la Flow Cell produit un signal électrique enregistré par un capteur. Lorsqu'une chaîne ADN / ARN est introduite dans le séquenceur, elle est guidée par une enzyme à travers le pore, perturbant l'équilibre ionique et donc le signal électrique. Chaque nucléotide a une façon caractéristique de perturber le signal, de sorte qu'en analysant le signal de sortie, la séquence nucléotidique complète peut être obtenue.

Le principal avantage du MinION est sa portabilité (sa taille est à peu près celle d'une clé USB), ce qui lui permet de séquencer directement sur le terrain. Grâce à sa technique de séquençage unique utilisant des nanopores, il fournit des informations en temps réel pour obtenir rapidement les résultats. Il génère également de longues lectures dont les tailles ne sont limitées que par la qualité de la préparation de l'échantillon, fournissant des informations générales sur les zones de grande complexité. Son faible coût et sa facilité d'utilisation en font une option encore plus intéressante.Son principale inconvénient est qu'il n'est toujours pas aussi précis que ses principaux concurrents, mais sa précision augmente considérablement chaque année.

4.2. Le stage

Au cours de ce stage, j'ai travaillé sur trois projets différents en tant que développeur Python. Le projet principal a consisté à développer un outil pour gérer plusieurs MinION simultanément. Le second projet s'est centré sur l'étude de la performance d'un logiciel appelé Mash en tant qu'outil d'identification d'un génome particulier dans un échantillon qui contient plusieurs génomes. Le troisième projet a eut pour

but améliorer la méthode actuelle de stockage des données générées par le MinION. Enfin, j'ai également écrit un blog de stage pour l'entreprise mais ce travail n'a pas été inclus dans ce rapport.

4.3. Conception et implémentation de Mk Core Connector et de son interface utilisateur

Le principal projet de ce stage a consisté à développer un outil de gestion d'une flotte de MinION. Cet outil peut être utilisé, par exemple, dans un laboratoire qui souhaite exécuter une expérience sur plusieurs MinIONs simultanément. La gestion centralisée des MinIONs permettrait un contrôle supérieur du travail de laboratoire, une réduction du temps de gestion et des erreurs, ce qui se traduirait en la génération de plus de données et améliorerait ainsi la précision des résultats. Au lieu d'exécuter l'expérience sur chaque MinION un par un, ce projet est installé sur un ordinateur administrateur qui se connecte à tous les ordinateurs auxquels un MinION est associé. Le programme centralisera le contrôle de tous les MinIONs afin qu'une seule personne puisse exécuter des protocoles sur tous les MinIONs et recevoir des informations en temps réel sur leur statut. Le programme permettra également de regrouper les ordinateurs par zones et d'exécuter différentes expériences dans chaque zone. De cette façon, chaque ordinateur d'une zone exécute la même expérience, indépendamment des ordinateurs des autres zones.

Le projet (appelé Mk Core Connector) a été écrit en Python en utilisant l'appel de procédure à distance (gRPC) de Google et WebSocket pour interagir avec les ordinateurs distants. Une interface utilisateur graphique (GUI) a été développée pour faciliter l'utilisation du projet. La GUI a également été écrite en Python avec un module qui traduit le code Python en JavaScript, HTML et CSS.

Le logiciel intégré du MinION s'appelle MinKnow et connecte l'appareil à l'ordinateur. MinKnow est le logiciel par lequel toute interaction avec le périphérique est faite. Ce projet nécessite l'installation de MinKnow sur n'importe quel ordinateur avec un MinION attaché et l'utilisation de l'add-on pour Minknow appelé «Bream», qui ajoute des protocoles supplémentaires pour le MinION et des fonctionnalités pour mieux suivre son statut. L'administrateur doit disposer de Mk Core Connector et de son interface utilisateur, ainsi que d'un accès réseau aux ports des autres ordinateurs.

Le code a été validé par plusieurs tests automatiques, écrits dans Pytest et Unittest. L'utilisation du module Selenium a été nécessaire pour tester l'interface utilisateur. Ces tests ont été exécutés à chaque changement de code et la modification n'a été acceptée que lorsque tous les tests ont été exécutés avec succès.

Enfin, le code a été intégré dans un environnement virtuel, ce qui garantit son fonctionnement sur tous les environnements, quels que soient le système d'exploitation, les dépendances installées, etc. Un programme d'installation a été généré et publié dans le référentiel de l'entreprise. Le résultat final a été un package qui fonctionne correctement et qui peut être facilement installé en téléchargeant son programme d'installation sur n'importe quel ordinateur, quel que soit son logiciel.

4.4. Evaluation de la performance de Mash

Ce projet visait à évaluer la performance d'un logiciel appelé «Mash» pour appliquer la technique nommée Host Depletion (épuisement de l'hôte)..

Grâce à sa portabilité, le MinION a mis au point une nouvelle méthode de travail sur le terrain. Il a été utilisé dans plusieurs projets pour séquencer dans des endroits éloignés, comme la forêt tropicale équatorienne ou la savane ouest-africaine. Habituellement, le but de ces projets est d'identifier des spécimens ou des virus particuliers affectant une population. Les échantillons extraits sur le champ ne sont jamais purs, ils contiennent beaucoup de bactéries mixtes ou d'autres cellules d'échantillon. Il est alors important d'identifier le génome d'un hôte particulier à l'intérieur de l'échantillon. Bien qu'il existe des outils bien connus qui permettent de faire cette identification, ces outils reposent sur des méthodes exactes qui prennent beaucoup de temps. «Mash» utilise la technique MinHash pour déterminer de manière probabiliste le pourcentage de confinement de l'hôte à l'intérieur de l'échantillon, réduisant ainsi le temps de calcul nécessaire. L'objectif de ce projet est de comparer Mash avec des algorithmes similaires et de déterminer si le temps et la précision des résultats sont meilleurs que la compétence.

Pour mettre en place cette évaluation, j'ai obtenu un échantillon rempli de lectures de rats et un échantillon ne contenant aucune lecture de rat. En comparant ces deux fichiers avec un génome de rat de référence téléchargé depuis Internet, j'ai pu déterminer: le taux de faux négatifs (lectures de rat non identifiées comme rat) et le taux de faux positifs (lectures d'autres espèces identifiées comme des lectures de rat).

Ces taux ont été calculés en utilisant différentes valeurs pour les paramètres d'entrée acceptés par Mash: la longueur kmer (k) et la taille de l'esquisse (s). La longueur kmer se réfère à la longueur des multiples fragments dans lesquels le génome sera brisé avant les calculs. La taille de l'esquisse fait référence à la quantité de fois que l'algorithme s'exécutera sur le même fragment pour se rapprocher des résultats.

Les résultats correspondaient à ce qui était attendu: plus la longueur du kmer était longue, plus le taux de faux négatifs était élevé et plus le taux de faux positif était faible. Le contraire était attendu à des longueurs plus faibles. Dans le même temps, plus la taille de l'esquisse était élevée, plus la précision était élevée mais plus le temps de calcul était élevé.

En particulier, sur Mash et pour les échantillons prélevés, le choix pour lequel le temps de calcul était plus bas mais les résultats étaient suffisamment précis (faible Faux Négatifs et Faux Positifs faibles) a été trouvé pour s = 100 et k >29, ce qui correspond a un temps d'exécution trop élevé. Pour des résultats inférieurs, l'économie de temps n'a pas été considérée suffisante, donc l'algorithme n'a pas été accepté. Cette étude a donc permit de prendre la décision d'ajourner l'utilisation de Mash en attendant que d'autre études, sur d'autres ordinateurs soient mis en place pour confirmer ces résultats.

4.5. Performance de la base de données

L'objectif de ce projet était de créer un outil permettant de mieux stocker les résultats générés par le MinION.

Le MinION interprète la perturbation du signal électrique qui génère les brins d'ADN / ARN lors du passage à travers les pores de la cellule d'écoulement afin de déterminer la séquence nucléotidique. La sortie de ce processus est un ensemble de données contenant le signal électrique brut et la séquence nucléotidique.

L'une des caractéristiques du MinION est qu'il concatène toutes les lectures avant le début du séquençage. Cela permet un processus de séquençage plus rapide et moins compliqué, car les brins d'ADN / ARN doivent être guidés jusqu'à une seule fois par cycle. Cependant, les résultats générés sont également concaténés, il faut donc une troisième séquence indiquant le début et la fin de chaque lecture du signal électrique brut et de la séquence nucléotidique.

Le traitement et le filtrage de ces fichiers ne sont pas simples, car le signal électrique est stocké en utilisant l'heure comme référence et la séquence nucléotidique est stockée en utilisant le signal électrique comme référence. Cela rend difficile la séparation des résultats en fonction des lectures de la durée, ce qui peut être intéressant pour les chercheurs.

Dans ce projet, nous avons étendu le fichier HDF5 utilisé pour stocker ces données dans un nouveau fichier appelé «fichier BULK». Ce fichier est créé à partir d'un fichier HDF5 en important toutes ses données. Il peut générer de nouveaux fichiers

contenant uniquement des lectures spécifiques ou la séquence générée pendant une période donnée. De plus, il peut être automatiquement téléchargé dans une base de données SQLite, ce qui permet des requêtes plus complexes et peut être téléchargé sur une base de données distante afin de stocker beaucoup moins de mémoire localement sur l'ordinateur.

Ce projet était également présenté sous le nom de Mk Core Connector. Des tests ont été écrits pour garantir le comportement correct du software développé. Comme de la cas du premier projet, un environnement virtuel a été créé pour rendre le programme exécutable sur tout environnement logiciel et une installation a été créée pour faciliter l'installation sur des ordinateurs.

Le logiciel développé est prêt pour être validé par les utilisateurs, ce qui sera effectué dans les prochains mois. Si, comme on l'espère, la validation d'usager est positive, le logiciel pourrait être intégré au logiciel intégré MinION MinKnow.

5. Introduction

5.1. The Generations of Sequencing

Since the first genome ever sequenced in the early 70's, genome sequencing techniques have rapidly evolved. The discoveries on the DNA molecule and the improvement of the computation potence has led to different sequencing methods and several companies offering their uses.

The so called first generation sequencing appeared back in 1977 with the Sanger and Maxam-Gilbert method. For years these were the only sequencing methods but they were very time consuming and expensive. The improvement of computational power allowed the arrival of the Second Generation of Sequencing around 2005, that broke the genome in millions of short reads, sequenced them in parallel, and realigned the genome after. It offered a much higher speed, lower princes and easier methods to obtain results. Companies like Roche and Illumina belong to this period. However, as new zones of the genome were sequenced, high complexity regions were found impossible to sequence or reconstruct from short reads. This has led to the appearance of the Third Generation of Sequencing, that uses long reads that store more information allowing to sequence the more complex regions. It also offers lower prices, higher speed and easier sample preparation. Oxford Nanopore Technologies´ MinION is, along with Pacific Bioscience, the most used method of this new generation.

5.2. Oxford Nanopore Technologies

Oxford Nanopore is one of the reference companies of the Third Generation of Sequencing. It was founded in 2005 to explore the usage of nanopores for sequencing and their first product, the MinION, the first and unique portable real-time sequencer, was commercially available on 2015. From then, they have developed the MinION´s scaled-up brothers: the GridION (launched on 2017) and the PromethION (launched on 2018). They are also developing a mobile-compatible device called SmidgeION expected to be launched on 2019.

The company has near 400 workers between its headquarter in Oxford (the office where I was based) and its satellite offices in Cambridge (UK), New York, Cambridge (US), Shanghai and Tokyo. It also has broader commercial presence in

India, Germany and France and strong collaborations with the universities of Oxford, Harvard and California Santa Cruz.

Up to today, the company the company has raised more than 800 millions of dollars from investors and it is expected to be the first trillionaire company of the United Kingdom.

5.3. Nanopore Sequencing and the MinION

Oxford Nanopore´s main product is the MinION, the first portable and real-time sequencer for DNA and RNA[1]. With a weight of 90g and a size of 105x23x33mm, it is easily transported to be used in field or in the laboratory.

The MinION´s main component is a flow cell: a battery formed two by to electrochemical liquids separated by a membrane. The ion exchange between the layers generates a voltage between the two layers of the membrane. This membrane is traversed by up to 2048 nanopores[2] all of them with an unique identifier.
When a sample is introduced on the device, an adapter enzyme is attached to the DNA/RNA strands to drag them to the nanopores. There, the two strands are separated and attached covalently one behind the other. Another enzyme pushes the strands through the pore at a given rhythm starting by the 5' end.

As the DNA passes through the pore, the voltage between the layers of the membrane changes, which creates a particular electric signal. The signal generated by groups of 3 to 6 bases is compared with a reference and interpreted to get the identity of the bases (see figure 1). The output is a high quality read created from both strands of the DNA.

The process can also be done with single stranded cDNA, producing faster but less accurate results.

Figure 1*: Process of sequencing inside a nanopore. Font:* [3]

With this method, the MinION can provide real time information about the strands, providing a much faster method for researchers to detect specific patterns without waiting for the whole sequence to complete. This can be done manually or automatically by the MinION: the MinION can look for particular patterns in specific locations of the genome by aligning the sequencing data with a reference. If the pattern isn't found on the location, the sequence is ejected from the nanopore and quickly substituted by another one.

Its small dimensions, low weight and real-time capacities make it a perfect device to use on field. In multiple occasions it has been used to sequence in remote places like the Amazon rainforest for Zika virus surveillance[4], West Africa to monitor the Ebola virus[5], the Antarctic to sequence microbial communities[6], Tanzania to sequence frog DNA[7], Snowdonia National Park to study closely related plant species[8] and even the International Space Station to evaluate its performance off-Earth[9].

5.4. Summary of the internship

During this 21-weeks internship I have worked in different projects coding with Python. The Gantt diagram of the internship projects can be found on the figure 2.

The main project in which I have work consisted in developing a program called “Mk Core Connector” (MCC) and a user interface (UI) to run scripts on a fleet of MinIONs. The inform about this project is written on the chapter “Design and implementation of MK Core Connector and its UI”

I have also worked in evaluating a software called Mash as an efficient tool for identifying a host inside a metagenomic sample. It can be found on the chapter: "Host Depletion with Mash".

Finally, I have also worked in a small project about storing and sorting sequencing data with HDF5 and SQL. It can be found on the chapter "Database performance"

Apart from these projects, I have written a blog for the company about my internship using a static web page generator called Hugo. I have not added this project on this inform as it didn't match the purpose of this inform.

Importantly, the code written during this internship has not been added to this inform in order to preserve its confidentiality.

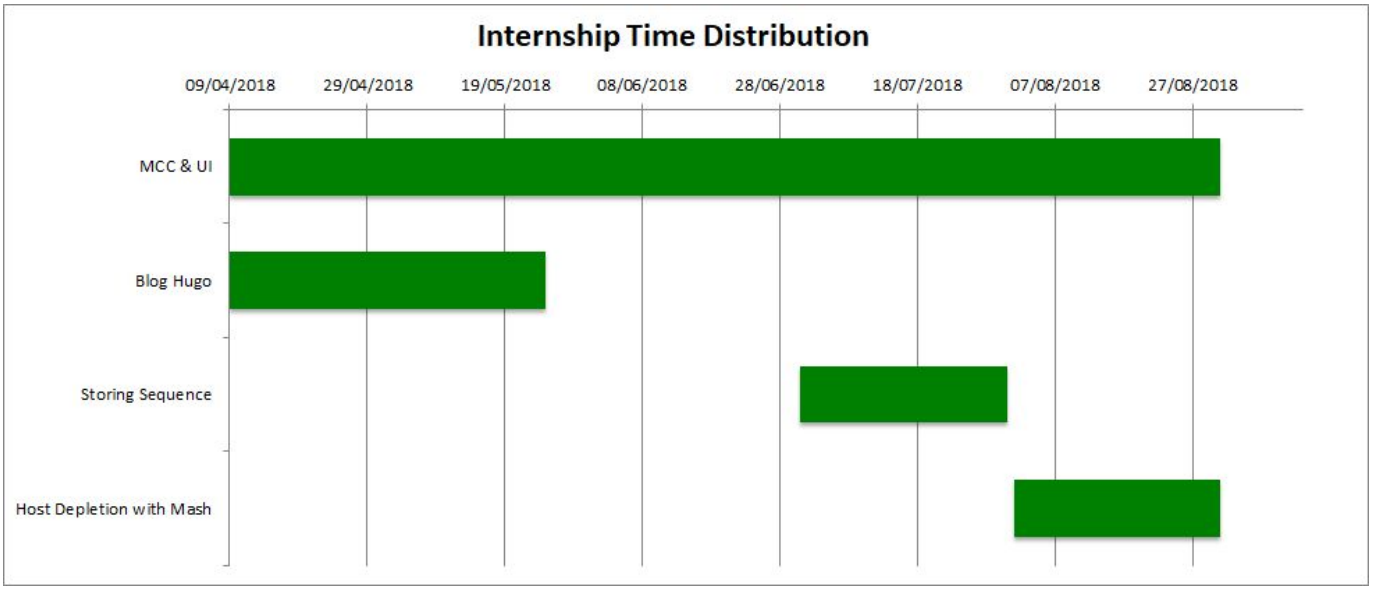


Figure 2*: Gantt Diagram of the work done during the internship*

6. Design and implementation of MK Core Connector and its UI

6.1. Background

Motivation

After the launch of the MinION, the Oxford Nanopore has worked to expand the possibilities of the device. Examples of that are the recurrent new launches of sample preparation kits, each of them with detailed instructions that allow sample preparation even for someone that doesn't know about the biology behind. Some of that samples are specially prepared to be kept without a fridge or without PCR amplification so that sample preparation and sequencing can be done on the field without having to carry special equipment to preserve the biological liquids.

Among these various applications, Oxford Nanopore is also adapting the MinION to be used in the laboratory. The software department of Oxford Nanopore has been working on the possibility to run multiple MinIONs on multiple computers hosts from a single administrator computer.

A possible application for this case is when different parts of the laboratory are running different experiments in parallel. Each part of the laboratory can be considered an area where multiple computers run the same protocol on the MinIONs that they have attached (*Figure 3*). With all the MinIONs running at the same time they become a much better tool as they generate more results without taking extra time. With more results there will be more statistical population, which means more accuracy and statistical power of the sequencing results. With all the MinION's being synchronized with a single computer, one person can run the experiment on all of them with a single click, and can control the status of each MinION and the notifications it sends at glance.

It is important to note that the laboratory is just one of the multiple environments where this project could be used. Other environments could be, as an example, a production center where MinIONs are tested in big quantities at the same time before they are commercially available.

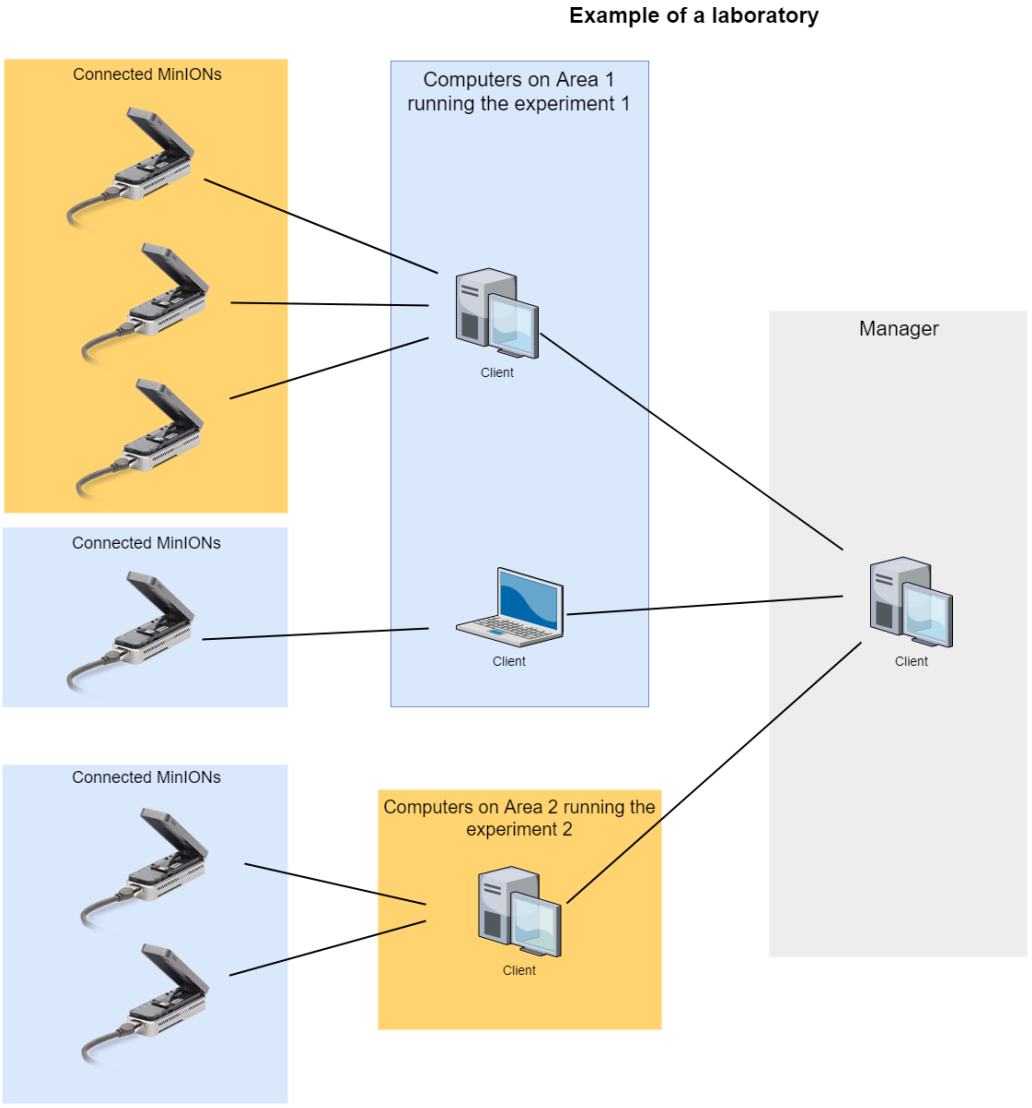


Figure 3: Example of how a Laboratory could be organized with this project

As the MinION is a low cost device, it is economically sustainable to have multiple MinIONs running the same experiment. In fact, comparing it with illumina's MiSeq, it has a cost of $1000 against $125000 of the MiSeq and a cost per run of $99 [10] against $750 for the MiSeq[11], so you could buy 125 MinIONs for the price of one Illumina sequencer.

Software of the MinION

Every MinION is controlled with a software called **Minknow**. Minknow is the communicator between the computer and the MinION. Any instruction that the user wants to perform on the MinION (or receive from it) will be sent/received from the MinION through this software.

MinKnow has a package called **Mk Manager** that acts as a notifier of the actions that the MinIONs are doing. Mk Manager can stream basic information like the port where the MinIONs are connected or the ID of the MinIONs. It can expose this stream of information through a computer port.

MinKnow can also be used by other softwares to control the MinION. Examples of this software's applications include more available protocols for the MinION (in form of scripts), a better data visualization or better tools to track the MinION. One of these softwares is called **Bream**, and it is the one used in this project. Bream includes more scripts for the MinION and a performance check, improving the trackability of the MinION's real-time performance.

A diagram of this layers working together can be found on Figure 4.

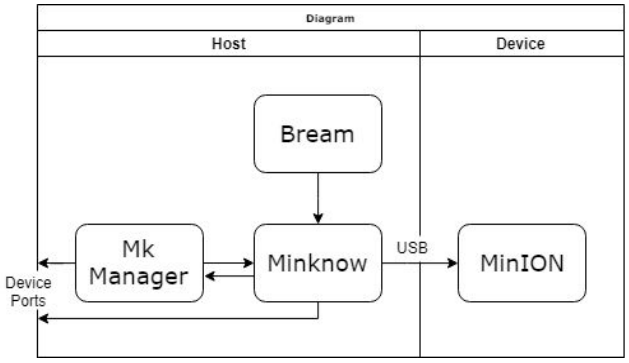


Figure 4: Diagram of the layers of a computer running Minknow

Objective of the project

The project aims to build a tool (called **Mk Core Connector**) that will connect an administrator to all the computers that have a MinION attached and control them remotely. It will also include a **graphical user interface (gUI)** for an easier use of Mk Connector. A gUI is a web page that will display the information of Mk Core Connector and use its tools graphically instead of on the bash terminal.

6.2. Requirement Models

Roles

There is only one role for the project: the administrator who runs the program and controls all the MinIONs attached to all the hosts.

General requirements

The project has to be installed on the administrator computer. This computer can run any operating system and doesn't need any extra software installed.
It has to be able to access all the computers on the areas by the port 9500, which is the port where the Hosts expose their Mk Manager.
The Hosts don't need to have this project installed, they just need a MinION connected with Minknow, Bream installed and the Mk Manager running on the port 9500.
Finally, the computers belonging to the same area must have the same version of Bream and Minknow installed so that they can run exactly the same protocols.

Functional requirements

The functional requirements of the application were listed on the basis of several discussions with potential users of the software and mostly the managing team of the projects. Table 1 lists these functional requirements.

Name	Availability	MCC/ UI	Description
List the areas	always	both	List all the areas to which you can be connected.
Select areas	always	both	Select specific areas from the whole list.
Connect to a list of areas	always	both	Establish connection with the computers of the areas and their connected devices.
Get MinION status	After area selection	both	Get the current status of the MinION (ready, running, user-stopped, error or waiting for flow cell)
Get MinION notifications	After area selection	both	Get real time notifications about what the MinION is doing or what it needs to perform an action.

List scripts	After area selection	both	List of the available scripts to run on one area.
Run a script	After area selection	both	Start a script/protocol of the MinIONs of an area
Advance stage	When a script stage ends	both	On a script running with multiple stages, proceed to the next stage.
Stop a script	If a script is running	both	Interrupt the running script/protocol running on the area
Get run results	At the end of the script	both	Give the results of the currently running script.
View/hide an area	After area selection	UI	Hide the information about specific selected areas.
Refresh connections	After area selection	UI	Refresh the connection to the selected areas.

Table 1: Functional requirements for the UI. From left to right: Name of the required action, availability of the action, if the action is found on the Mk Core Connector (MCC) and/or just its gUI, and description of the action.

6.3. Design of Mk Core Connector

Data Model

The project has 4 entities depicted in the class diagram of Figure 5:

- **Area Manager,** the Administrator, that can choose and connect to specific areas between all of them. It has access to all the areas.

- **Area**: a group of computers (hosts) that work together.

- **Host**: a computer belonging to an area and with at least one MinION attached.

- **Device**: a MinION attached to a computer.

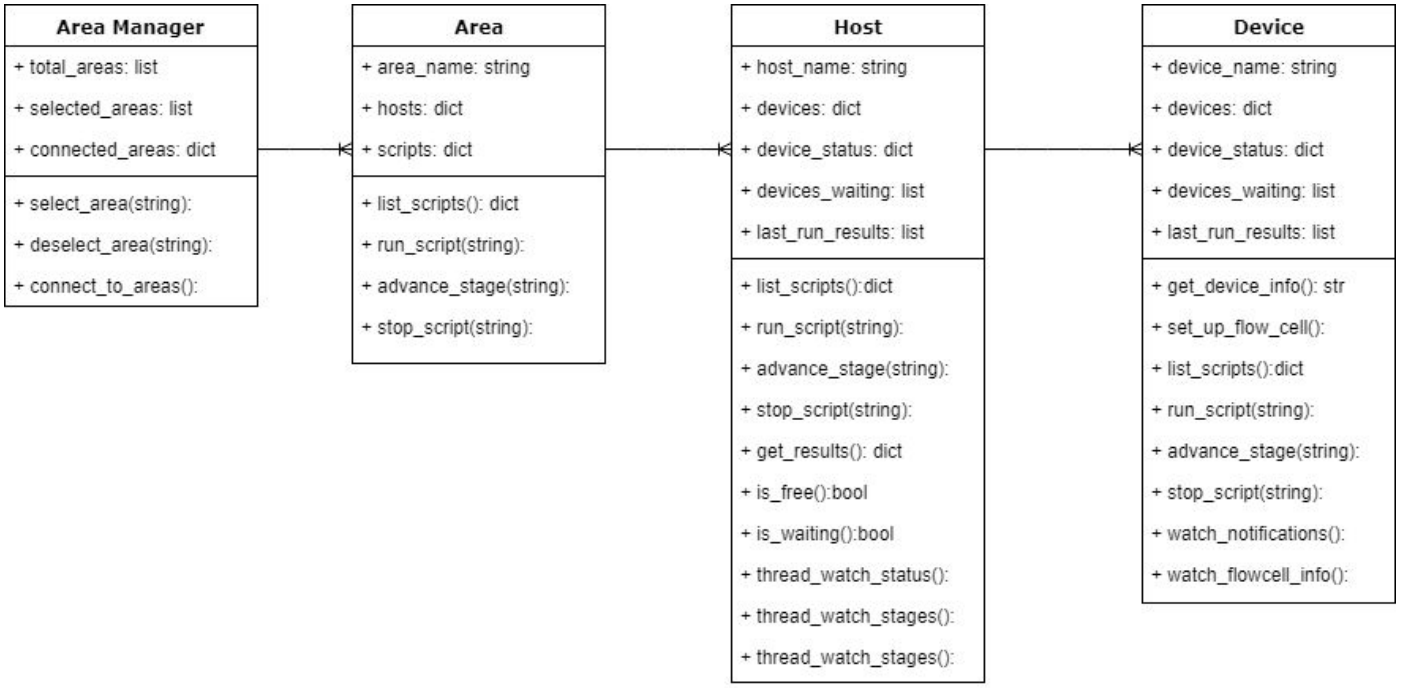


Figure 5: Class Diagram of the UI.

Connection technology

To connect the Administrator with the Hosts, two technologies have been used: gRPC and Web Socket.

gRPC (Google Remote Procedure Call) is a technology that allows a client-server communication. It is the best option to connect to the MinION remotely because of its workflow: it sends a request and receives the response asynchronously in a very fast and efficient way. Furthermore, requests are written in a neutral language called Protobuf that is automatically translated to any code that the client or server requests. Annex A develops on *gRPC.*

WebSocket has been used punctually to connect remotely with the Mk Manager. WebSocket is designed for communication between web servers, so it is focused on timing of the data exchange rather than its speed. This makes it unsuitable for the fast and irregular communications that the MinION needs, but it is the best option to connect with the Mk Manager. The Mk Manager streams information continually on the same channel and doesn't need very high speeds. In this specific case scenario, timing becomes an advantage, and so WebSocket becomes a better option than gRPC.

Methods workflow

Mk Core Connector is a complex program, that uses several connections to manipulate the MinION. To start with, it reads an internal TOML file that contains a list of all the areas and the IP of every host on each area.

When specific areas are selected, Mk Core Connector instantiates an Area class for each area and a Host class for each IP on the Area. At that moment, the connection is not established yet so the Area and Host instance are just class instances living on the administrator's computer.

The host class will then instantiate a class called Manager that will use the hosts' IP to connect to the host computers through the port 9500, where the host is exposing the Mk Manager. This is needed because at this point the administrators' computer doesn't know neither the number of MinIONs attached to the host nor the port where they are running, so it can't connect with them directly. Instead, the Mk Manager always runs on the port 9500 and can give information about the connected devices.

Once the connection is established, the Administrator computer will request information to the host's Mk Manager about the number of MinIONs connected to the host and the port where each of them is running. When it has the port number of every MinION running on the host, it will close the connection with the Mk Manager on the port 9500 and it will start a connection with each MinION individually on their port. Mk Core Connector has its own protobuf files for the communication needed with the MinION.

Every time that the user runs one action, Mk Core Connector will use gRPC and the functions defined on the protobufs to send the right instructions to the MinION.

A UML diagram for this workflow can be found on Figure 6.

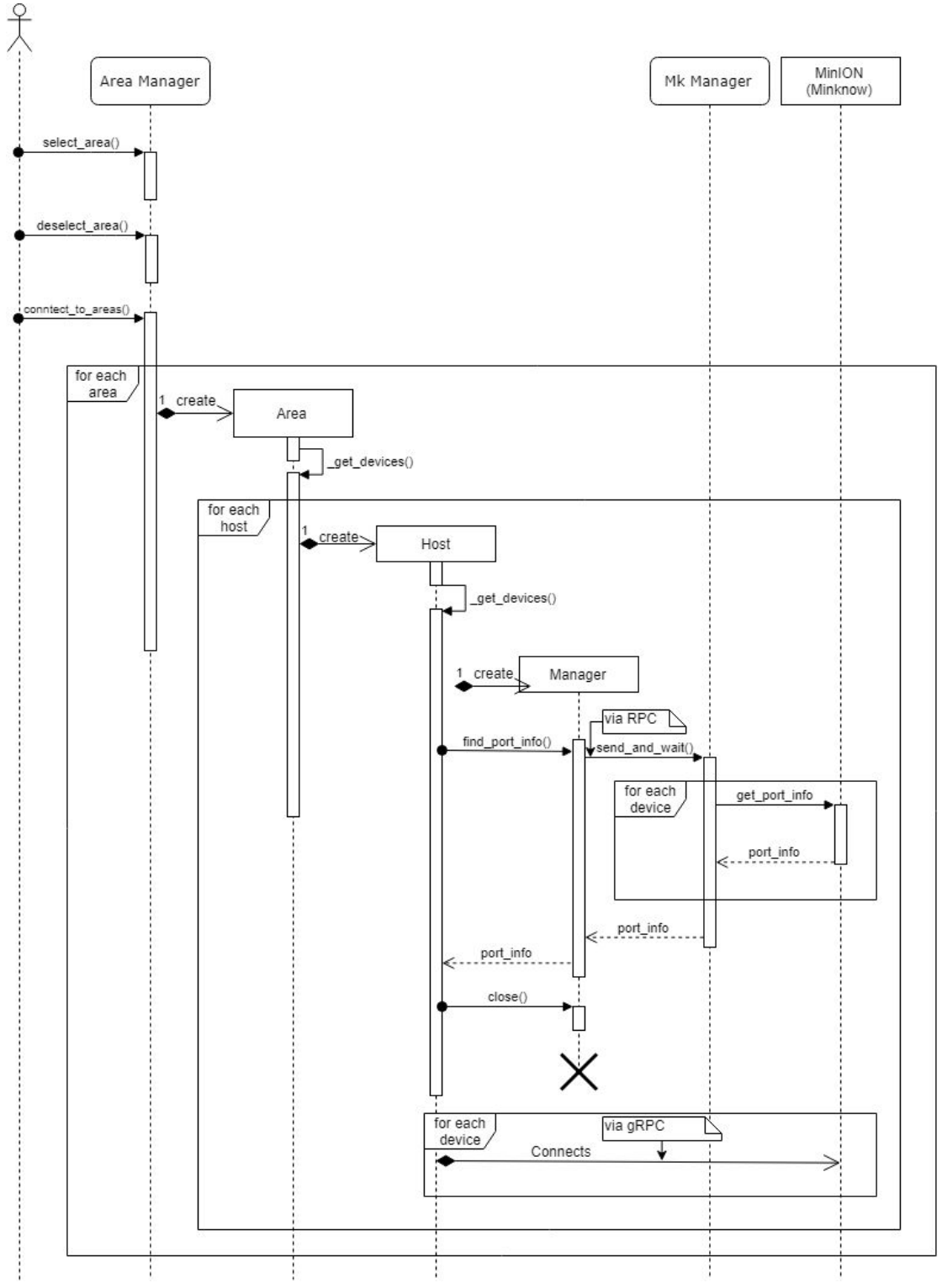


Figure 6: UML Diagram of the connection between the Administrator and the MinIONs

6.4. Design of the UI

Architecture

The UI is composed of two pages: the Area Selection page, where the administrator can select a group of areas between all the available ones and start a connection with them; and the Script page, where the administrator can manage the MinIONs of the selected areas.

The user will first access to the Area Selection page and from there, once at least one area has been selected, to the Script Page. In the script page the user can re-establish the connections with the MinIONs any time by clicking “Reconnect” and go back to the Area Selection Page anytime as well (see Figure 7).

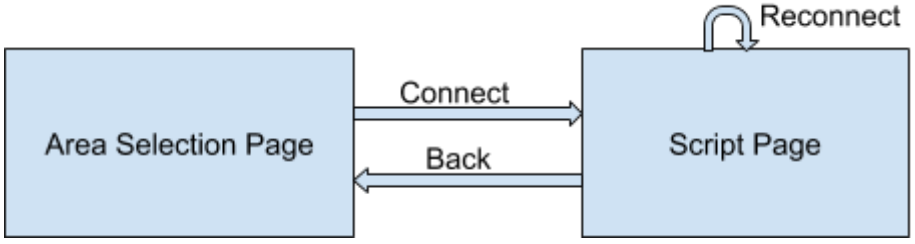


Figure 7: Flow diagram of the UI pages.

6.5. Implementation

Tools used

To facilitate the execution of the project from the bash script, the language **Make** has been used. A file **Makefile**[12], written with the Make syntax, which is very similar to Bash scripting, is a file on which you can define functions and call them from a terminal. The Makefile is useful to quickly run tests, setup dependencies, executing a Python script or for any task that would involve several lines of code on a bash shell. The Makefile saves each function with a name and the user can run the function by just typing *make + name of the function*.

As Mk Core Connector has been developed by a team of 3 persons (including me), we have used the tool **GitLab**[13] to manage the code development. Gitlab is a platform that allows to programmers to work separately on a project and then push the code to the internet. By pulling the code from the internet, pushing new versions and merging it with the newer versions, the programmers can develop without erasing the other one’s work.

The edition framework is PyCharm that provides a user-friendly environment with debugging facilities.

Documentation was written in Sphynx, a package that generates static web-sites on the basis of comments inserted in the code as docstrings and other files written in rst language.

Libraries used

To develop the UI, the **Sofi** module for Python[14] has been used. Sofi provides useful classes to write in CLS and HTML from Python. On execution, the CLS, HTML and JavaScript files are generated, the gUI is created and exposed with WebSocket on the port 8080. This allowed me to start writing the gUI from Python without having to learn from zero JavaScript.

Mk Core Connector has been tested with **pytest** and **unittest**, the common tools used to write tests. The gUI has been tested with the Python module **Selenium[15]** , specifically devoted to test gUIs. Selenium offers a variety of classes and functions to simulate clicks on given parts of the screen. It starts by opening a browser on a given URL (in this case, the port 8080) and then it automates the actions by detecting the HTML code shown on the browser. For example, it can detect an HTML objects like a button and then click on them, or open a dropdown and select an item from it. At the same time it provides information about what is happening on the screen, so that the user can test with pytest that the gUI is working correctly.

Packaging the project

The package provides the following characteristics:

- Make the software independent of the environment, i.e. usable on any computer regardless of its installed dependencies. This is achieved with a tool called **Pipenv**[16]. Pipenv creates a virtual environment inside the project with Python and all the dependencies installed. This virtual environment will be packaged with the project. When the project is run on another computer, the Python executable from that virtual environment will be used instead of the Python installed on the computer. In other words, the project will carry with itself all the tools that it needs to work properly. It also facilitates the installation of compatible versions for all the dependencies used in the project.

- Create an executable and an installer to easily distribute the project. This has been done with a tool called **setup.py.** Setup.py is a file containing information about the project such as the project name and version, the developer name and email and other similar data. An installer has been generated from the setup.py with a simple line of code. It has then been released to the local repository of the company.

- Docker[17] is a tool that wraps an application with all the parts it needs into a container. The containers can then be run in any other machine and it will be ensured that they work. In this case, Host and Administrator containers have been created separately to ensure that the project works correctly in both roles. After that, a whole laboratory has been simulated by running the Host container multiple times and connecting them to the Administrator container. More information on Docker can be found in the Annex B.

6.6. Validation Methods

The software has been validated thought tests during its development in order to ensure its correctness and robustness. User validation to evaluate the usability of the interface fall outside the scope of the project.

Tests are essential to ensure that the project works well, specially if multiple developers are working simultaneously on the same project. Any changes on the code will be followed by running the tests to see if all the functionalities of the program are still working or are now affected by the changes. The Continuous Development /Continuous Integration (CD/CI) paradigm (see Figure 8) has been followed. In a CI pipeline, each time that a developper pushes new code, it generates a pipeline through which the code passes a series of jobs split in different stages. Each job consists of some instructions that will test if the code is correctly written and compatible. As soon as a job fails, the pipeline fails, so the only way to push code is to successfully pass all the tests.

This procedure allows to make sure that multiple developers working at the same time, can work on different parts of the same project without creating integration problems between their parts. Thanks to that, the master branch of the code is always functional (it passes all the tests).

We have used the CI pipeline platform integrated on GitLab. More details on CD/CI can be found in Annex C.

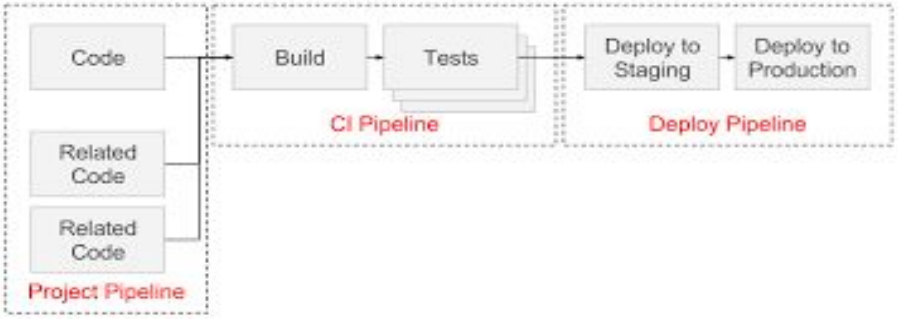


Figure 8: Diagram flow for the CD/CI pipeline.

6.7. Results

As a result of the analysis and implementation described above, the software was developed. No screenshots could be made of the real interface, but *Figure 9* illustrates its aspect.

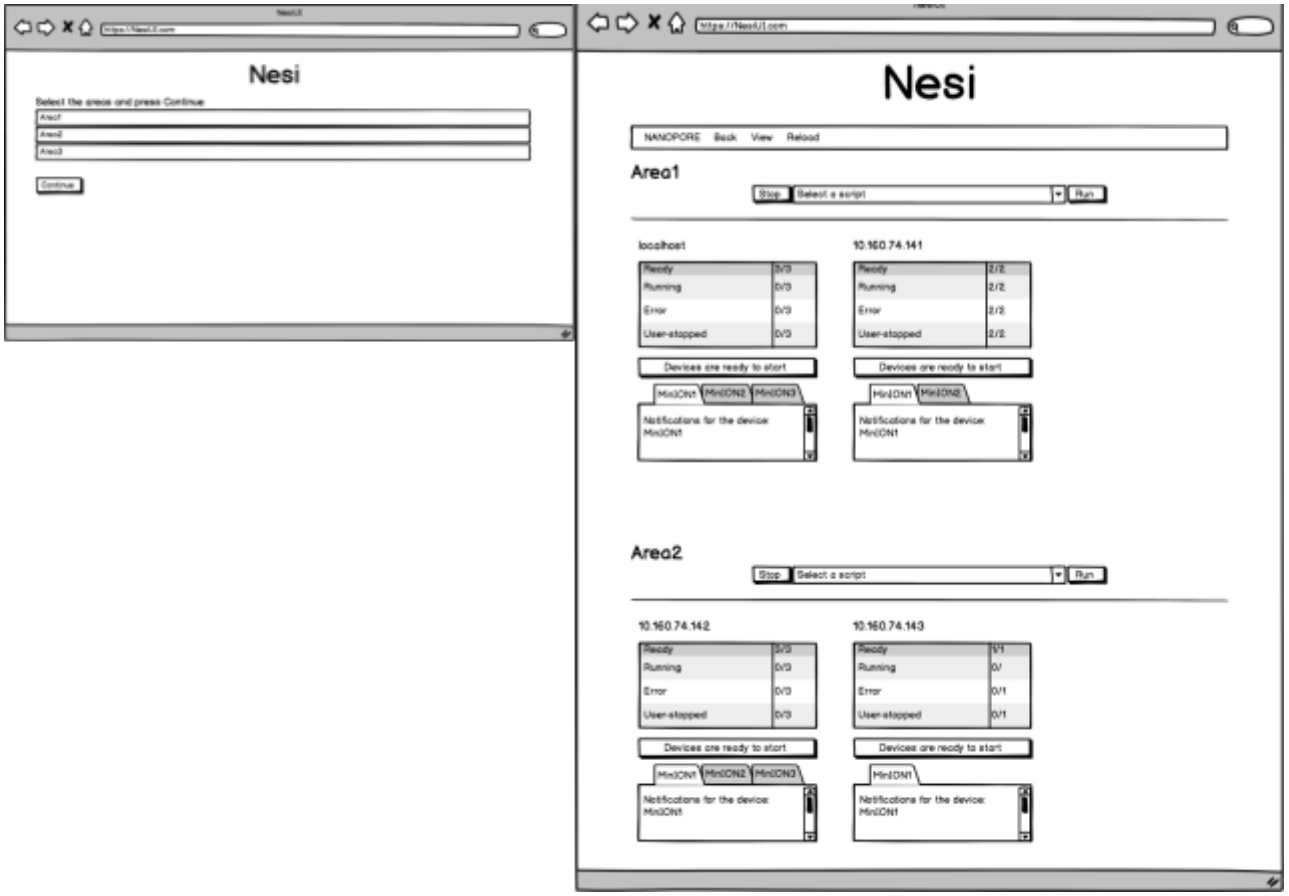


Figure 9: Visual aspect of the UI. At left: the Area Selection page; at right: the main page to control the MinIONs.

7. Host Depletion with Mash

7.1. Background

As Whole Genome Sequencing is each time more accurate, sample quality and quantity is becoming a major technical bottleneck in clinical sequencing. A large proportion of field-derived specimens are heavily contaminated with other specimen´s material.

“For example, when trying to study a particular pathogen that has affected a particular host, attempts to sequence a sample without adequate removal of the host DNA negate some of the cost benefit realized by the current advances in sequencing technology. Whereas problems associated with the small amounts of starting material have been addressed extensively through the development of alternative library preparation methods as well as the discovery of novel amplification technologies, host contamination has remained a big challenge, especially for pathogens that are very difficult to culture in vitro. Moreover, due to the small size of pathogen genomes in comparison to that of the genome of their host, the presence of just a few nucleated host cells in a pathogen clinical specimen can completely inundate that sample with host DNA contamination”.[18]

Other approaches to host depletion are focused in finding the appearance of a particular genome in the metagenomic sample rather than on the isolation of it. The only fact of knowing if a specific genome is in the sample can give information about the host, like its diet or its illnesses. It is also a proof of the quality of the sample itself. With the appearance of the MinION, the portable sequencer developed by Oxford Nanopore Technologies, this approach is being increasingly used. Between its multiple applications on field, Oxford Nanopore has been sequencing animal remains at the remote Ecuadorian Choco rainforest, finding species that were thought to be extinct[19]. In Africa, it has been used to identify virus-affected wheat plants before their consumption[20] and its applications seem to be increasing over time.

However, as this is a work to be done on field, efforts have to be made to make this process faster. While many tools use exact methods to determine the reads that belong to a specimen, like Blast, we are looking for a faster tool that can be used on field, even at the price of a lower accuracy.

On this project I have been using the software Mash, a toolkit based on the MinHash technique, to estimate the containance of a particular genome inside a sample at the fastest possible time without sacrificing much accuracy.

7.2. The MinHash Technique

Mathematical Context

Consider two sets A and B. Mathematically, the containment of A inside B as well as the resemblance of A and B, can be calculated with the expressions on Figure 10:

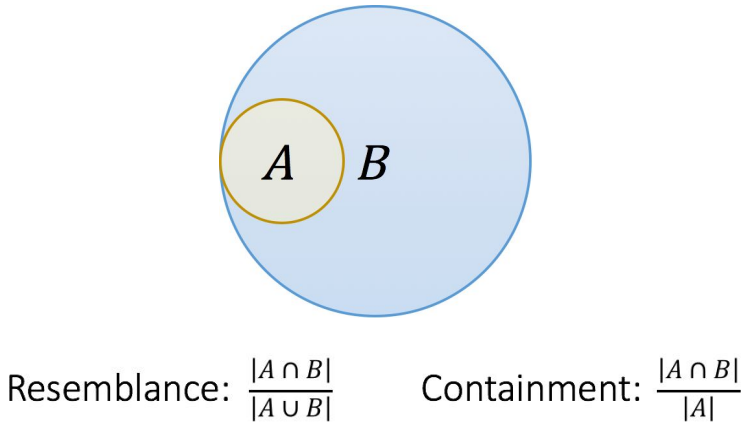


Figure 10: Scheme and formula for the resemblance and the containment.[21]

In the case that we want to study, B would be the whole genome of a the specimen that we want to find on the metagenomic sample, and A is any read of the metagenomic sample. If the index for the containment equals one, then A is completely contained in B and thus, A is a read belonging to the specimen B. Any value between 0 and 1 of the containment index represents a partial containment and can mean either a genome from the specimen B that can´t be fully contained because of sequencing errors or a genome from another specimen that is similar to the genome of B. Differentiating these two cases can be very hard especially for low accuracy sequencing methods or highly similar specimens on the metagenomic sample.

Calculating the containment index for every read of the metagenomic sample takes a lot of computational resources. The MinHash gives a probabilistic estimate of the containment index that takes much less memory and less time to compute.[22]

Method

The MinHash technique was created to detect duplicates of web pages. When applied to genomics, its algorithm is slightly different from the original one, but it remains very similar.

When a genetic sequence is inputed, it will be divided into all its possible subsequences of length k, called kmers.Figure 11 illustrates this process with a k value of 7.

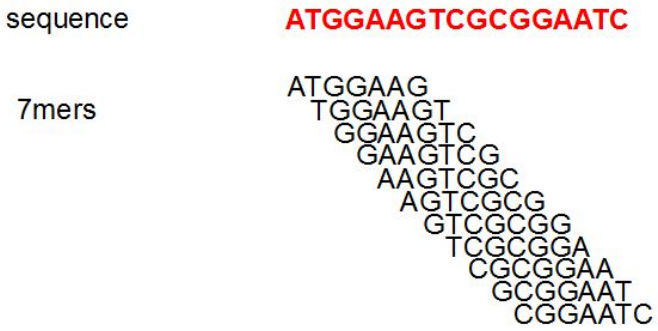


Figure 11: The division of one sequence in kmer's

A hash function will be applied on each kmer, creating a unique identifier of fixed length. These identifiers (or hashes) will be identical if they come from identic kmers. Of all the hashes, only a few of them will be stored: the ***s*** identifiers with the lower hash value, where ***s*** is a given parameter. We call these identifiers "minhash values" (h_{min}).

When comparing two sequences (S1 and S2). The probability of both sharing an h_{min} value is equal to the the containment index, as shown on the equation 1.

$$pr[h_{min}(S1) == h_{min}(S2)] = J(S1, S2)$$

Equation 1: probability of two sequences having the same h_{min} value

Therefore, the containment index can be estimated by evaluating the expression $h_{min}(S1) == h_{min}(S2)$ for all the h_{min} values calculated and then dividing the amount of times where the expression was True by the total amount of calculations.
The more different h_{min} values were calculated, the more exact will be the Containment Index.

In other words, instead of comparing the whole read with the whole genome, Mash will compare some random fragments of the read and will try to find them on the genome. The number of fragments it compares (s) and the length of each fragment (k) will determine the performance of the method.

7.3. Mash

Mash is an open-source software that uses MinHash to answer the resemblance and the containment problems. It has been developed to work with the current database sizes in an acceptable time, as an alternative to the time consuming method BLAST.

Mash works in two steps: Sketching and Screening. First, sketch files are created from a sequence stored in fasta or fastq format. When creating a sketch file, the method described above (see Method) is followed to obtain a list of the *s* minhash values for the given sequence. The values of s and k are inputted by the user. As minhashes are strings of just a few characters, their size can be thousands of times smaller than the original file. Once the sketch file is generated, it can be used to calculate the resemblance/containment on another file, that should also be another sketch created with the same *s* and *k* values. Sketches can be reused continuously, greatly reducing the operations needed to determine the containment index.

Sketch size

The sketch size (s) is the number of minhashes that will be calculated on the sketch file. The containment index is calculated using these hashes, so the more hashes used, the the more accurate the containment Index estimation will be, as shown in the Equation 2.

$$\varepsilon = \sqrt{\frac{1}{s}}$$

Equation 2: error derived of the MinHash Method calculations. [21]

Thus, the accuracy can be controlled changing the sketch size at the price of calculating more hashes (which will lead to higher computational times).

Kmer length

The kmer length (k) will affect the sensibility and the specificity of the calculations. When using a small kmer size, the sequences that are being compared will be fragmented into a lot of small fragments. The smaller the fragments are, the less information they carry about their original sequence, thus the more general they become. Small fragments have a much higher probability of being shared by chance (e.g. two sequences have a much higher chance of sharing a fragment ATAA than a fragment ATAAGGTAGCCC). The probability of matching kmers just by chance can be calculated with the Equation 3.

$$p = \frac{1}{\frac{4^k}{g}+1}$$

Equation 3: random match probability between kmers [21]

where g is the genome size. The largest the genome is, the largest kmer length is needed to reduce the probability.

On the other hand, higher kmer values are much more vulnerable to sequencing rates. When two theoretically identical fragments are compared, if one of them contains a sequencing error that doesn´t match the other sequence, the whole fragment loses identity score. Therefore, the larger the kmer is, the more bases are affected by the sequencing error, raising the False Negative rate (less specificity). At the same time, the matching kmers have more matching bases, so True Positive rate is higher (more sensibility).

Finally, the kmer length having to be hashed, it is restricted to $4^k \leq 2^{32}$ for a 32-bit hash and $4^k \leq 2^{64}$ for a 64-bit hash.
The key of the analysis resides in a good choice of these two values.

7.4. Objectives

Analyse how Mash applies the MinHash technique and evaluate if it is a good technique.

7.5. Experiment

Obtaining the reference

To perform the analysis we took two samples from the Oxford Nanopore Technologies laboratory. One had been extracted from a rat defecation and

contained several rat sequences mixed with other bacterias. The other one came from an unrelated experiment and contained no rat sequences at all. They both had been sequenced with the MinION and the sequencing data was stored on a fastq format file for each of them.

The first objective was to obtain a file containing only rat reads. This file would be used as reference to detect the false negative rate. To do so, I downloaded from NCBI the reference rat genome and I used BLASTN to align this reference rat genome with the sample that contained rat reads. BlastN is a time consuming but exact method that returned all the reads that had been well aligned with the rat genome. The result was a selection of 1314 out of 4000 reads from the sample with an identity score superior that 96% with the rat genome. The reads were separated into another fastq file using a python script written with the library *BioPython*.

The sample that contained no rat reads would be used to get the false positive rate.

Idea

As Mash uses the Jacquard Index to solve the containment problem, it doesn't output the reads of the query contained on the reference, but rather all the reads of the query with a high alignment score with the reference. To determine if it was a match or not, I determined the arbitrary limit of 90% of identity or more for a match. This limit was later on changed and studied (see Results).

The idea of the experiment was to use Mash to calculate the containment of the rat reads with the rat genome. Theoretically, as all the file generated on the previous step contained only rat reads, they should all be contained on the rat genome with a high identity score. Any read that Mash would see contained on the rat genome (identity <90%) would be counted as a false negative.
At the same time, the same idea was used with the file containing no rat reads. All the reads with an identity score superior to 90% were considered as matches and therefore, false positives.

As explained before (*see Mash*), Mash uses the parameters sketch size and kmer length to estimate the Jacquard Index so, for different values of these parameters, different False Positive and False Negative rates would be gotten. The best values for these parameters would be the ones that would make the lowest False Positive and False Negative rates.

Operations

The alignment had been done with the sketch sizes: 100, 1000 and 10000, and for the kmer lengths 16 to 32. The lower limit for the kmer length (16) was determined with the Equation 3, since it was the lowest kmer length for which the probability of matching kmers by chance was lower than 0,01.

7.6. Results

For a match rate of 90%, the results on the Figure 12 were obtained.

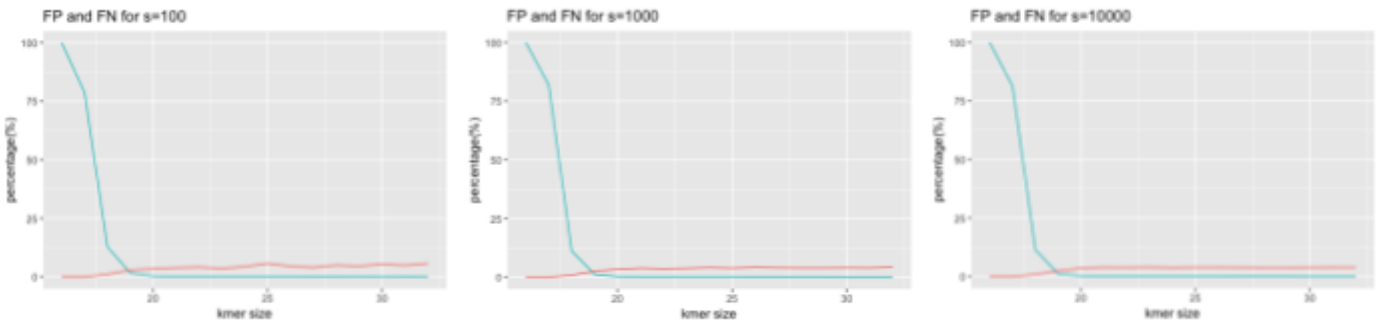


Figure 12: False Positive (blue) and False Negative (red) rate for different kmer length (X axis) and sketch sizes (Y axis)

As expected by theory, for small kmer lengths the probability of matching kmers by chance is too significative and dramatically raises the False Positive rate. Also, higher kmer lengths mean lower False Positive rate, while the opposite applies for the False Negative rate.

We can see from here that reasonable results are gotten with kmer lengths superior than 19. The increment on the sketch sizes slightly diminishes the False Negative rate at higher kmer lengths from 5.6% on s=100 to 5.1% on s=1000 and 5.05% on s=10.000. The time taken to calculate the rates is shown in Figure 13.

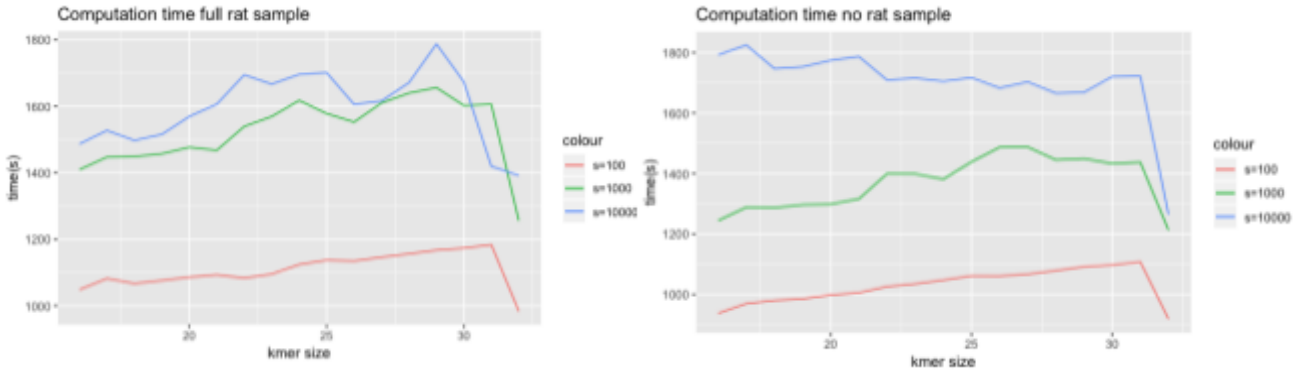


Figure 13: Time (Y axis) taken to calculate the False Negative rate (left) and the False Positive rate (right) for different kmer sizes (X axis) and for different sketch sizes (s=100 red, s=1000 green, s=10000 blue)

As previously thought, for higher sketch sizes the computational time gets higher, getting up to 1800 seconds (30 minutes) for sketch sizes of 10000, or 1000 seconds for a sketch size of 100.

The decrease on the False Negative rate is too small to compensate the increase on the computational time. However, experiments with sketch sizes of 10, 50 and 70 got significantly got worse results, so the sketch size of 100 was selected as the best option.

Now, with the selected sketch size of 100, the identity percentage for a match was changed from 90% to 85% and 95% getting the results shown in Figure 14:

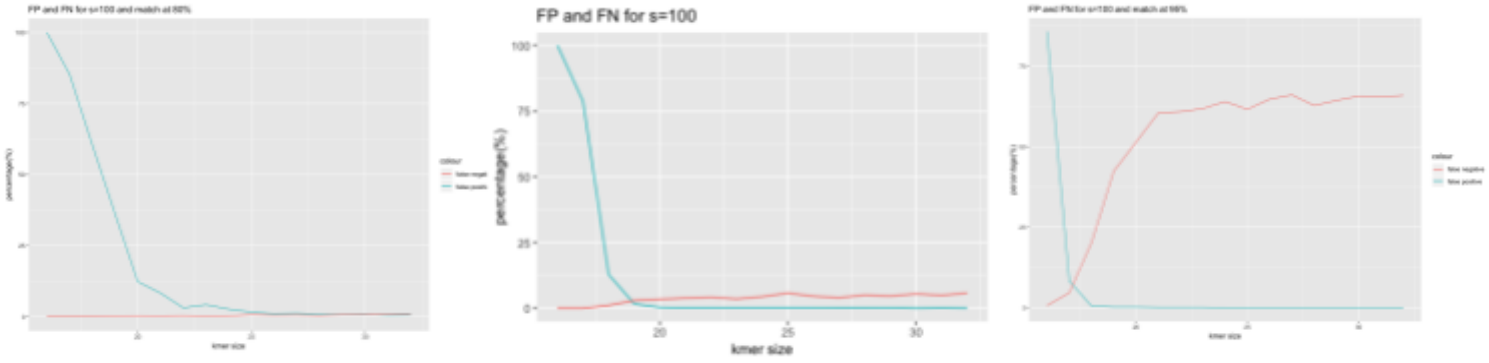


Figure 14: False Positive (blue) and False Negative (red) rate for different kmer lengths (X axis) using an identity rate of 85% (left), 90% (center) and 95% (right) to consider a match.

For a small identity limit, all the reads from the rat file are matched, getting a False Negative rate of zero for all kmer lengths. Reads from bacterias with a genome similar to the rat one are confused with rat sample and therefore the False Positive rate is higher. However, this is compensated with high kmer lengths, getting the best results at k>29.

For a higher identity limit, most of the rat reads don't reach the limit and are not considered matches. The False Negative rate is therefore too elevated. Sequencing errors are probably responsible for not having a match superior to 95% on most reads.

7.7. Discussion

Mash has given fair results with different parameters. However, as with all methods today based on an identity score between query and reference, its ability to give good results greatly depends on the similarity of the metagenomic data. For example, specimens with 90% of similarity with the rat would force to push the limit of the matching limit higher than 90% but for these scores, a very accurate sequencing method would be needed. Some knowledge of the composition of the sample and of the accuracy of the sequencing method is then needed to fix a good identity limit for the match. While the accuracy of the method can be easily known, the composition of the sample would require a pre study which would mean more time, failing the objective of being a fast method. A possible method to solve this would be to study the specimens with a very similar genome and treat the sample in order to avoid this specific organisms.

However, all these considerations aren't specific to Mash, since all the modern methods rely on similarity between sequences to solve the containment problem. Mash achieved good results lowering the False Positive and False Negative rate for large kmer lengths. It also obtained results much faster than Blast, with a sketch size of 100 (17 minutes). Nevertheless, it 17 minutes is a too large value compared with other softwares like Minimap (10 minutes), with which the company had previously experimented. Although having good FP and FN rates, the higher computational times were a major obstacle for using Mash on the field. This worse performance could still be explained by a problem on the computer generating the results (a MacBook from 2015) like having other operations running at the same time that reduced the computational power of the computer. Repeating the experiment in another computer would be a good way of being sure of the results.

7.8. Conclusions

Mash is a software that uses the MinHash method to calculate alignments between sequences. In this case, it has been used as a fast probabilistic method to determine if a particular host was found inside a metagenomic sample. While it has proven good results with a good choice for kmer lengths and sketch sizes, its long computational time remains too high against other softwares.

8. Database performance

8.1. Background

Motivation

Over the last few years there has been a revolution in DNA sequencing technology. From the First Generation of Sequencing that produced 300-1000 base pairs per run, now a MinION can produce up to 10^6 base pairs, and this number is just limited by the sample preparation so longer reads are to be expected with better sample preparation methods. Furthermore, the cost of sequencing has decreased from almost $10.000 on 2001 to less than $0.1 on 2018[23]. This massive cost decrease along with the much bigger data output has created a big problem for bioinformaticians to store the sequencing data. A huge amount of data is being produced every day and without a good method store and access this data, no further progress could be made because no modern computer could handle this large output.

As informaticians try to solve the problem, new ways of storing data arrive to the market, like HDF or SQLite or the file formats FASTA, FASTQ, BAM or SAM.

Current methods involve filtering the sequencing data several times to leave just the necessary information for the user. As an example, a typical sequencing raw data can take hundreds of Gigabytes or even some Terabytes. After treatment, a text file is generated containing just the genetic sequence, which can take some between 0.5 and 4 Gigabytes. Finally, a new text file can be generated containing just the variations of the sequence from its reference genome, reducing the space it takes to a few Megabytes or even some hundreds of Kilobytes[24] (see Figure 15).

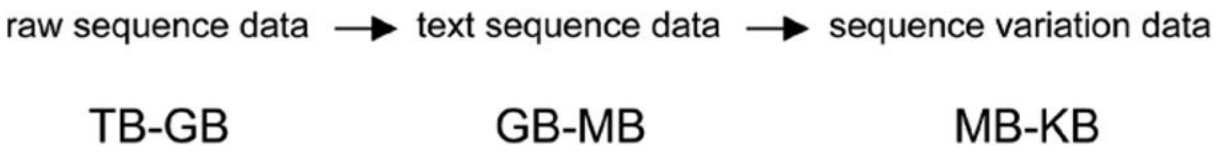


Figure 15: size reduction with consecutive data filtering.

However, with each step a lot of information is lost, so it is important to have a good methods to store and access the data on its raw form. Good methods for storing and accessing this data provide much lower times for filtering and gives the user more

choice for filtering specific regions, which translates to less total size for the text sequence file.

The MinION uses one of latest releases of HDF, called HDF5[25], to store its raw data. While this method has proven to work well for its purposes it has its limitations. In this project I have worked to improve way in which MinION's raw data is stored, to facilitate its access. I have also worked to compensate its limitations using other technologies, like SQLite.

HDF5 and SQL

HDF5 (Hierarchical Data Format version 5) is a data format used to store large amounts of data. Its main characteristic is the hierarchical organisation.

It is composed of two objects: Groups and Datasets. Datasets are multidimensional arrays where the data is stored. Groups are containers that can hold Datasets and/or other Groups. Groups can support attributes to store the metadata.

Its hierarchical structure allows very rapid data access, as it can access the requested Datasets navigating through the groups instead of loading the whole Datasets. Other technologies like SQL would need to load all the data to select a specific group from the whole, which would increase the response time. This makes HDF5 the reference format to store data in many applications.

However, HDF5 has its downsides too compared to SQL. To start with, it has to be run locally, what means that it needs enough storage on the computer to store all the datasets. This is avoided using SQL, that can store the data on a server remotely. Also, despite being much faster because of its hierarchical structure, it also loses the ability to process complex requests, like taking specific parts of different datasets.

MinION's Raw Data

As explained in the section Nanopore Sequencing and the MinION, the MinION is formed by multiple pores that generate an electric signal when a DNA strain passes through them. This electric signal is received by the computer along with the unique identifier of the pore, and is interpreted to determine the nucleotide bases. As the reads are concatenated one behind the other during the sample preparation, the electric signal is continuous and a secondary signal is needed to indicate the beginning and the end of each read.

The raw data produced by the MinION is an HDF file containing some comon metadata about the sequencing and, for each pore, some metadata of the pore, the electric signal generated by the port, the nucleotidic sequence extracted from the signal and a sequence specifying the beginning and the end of each read, as presented on Figure 16.

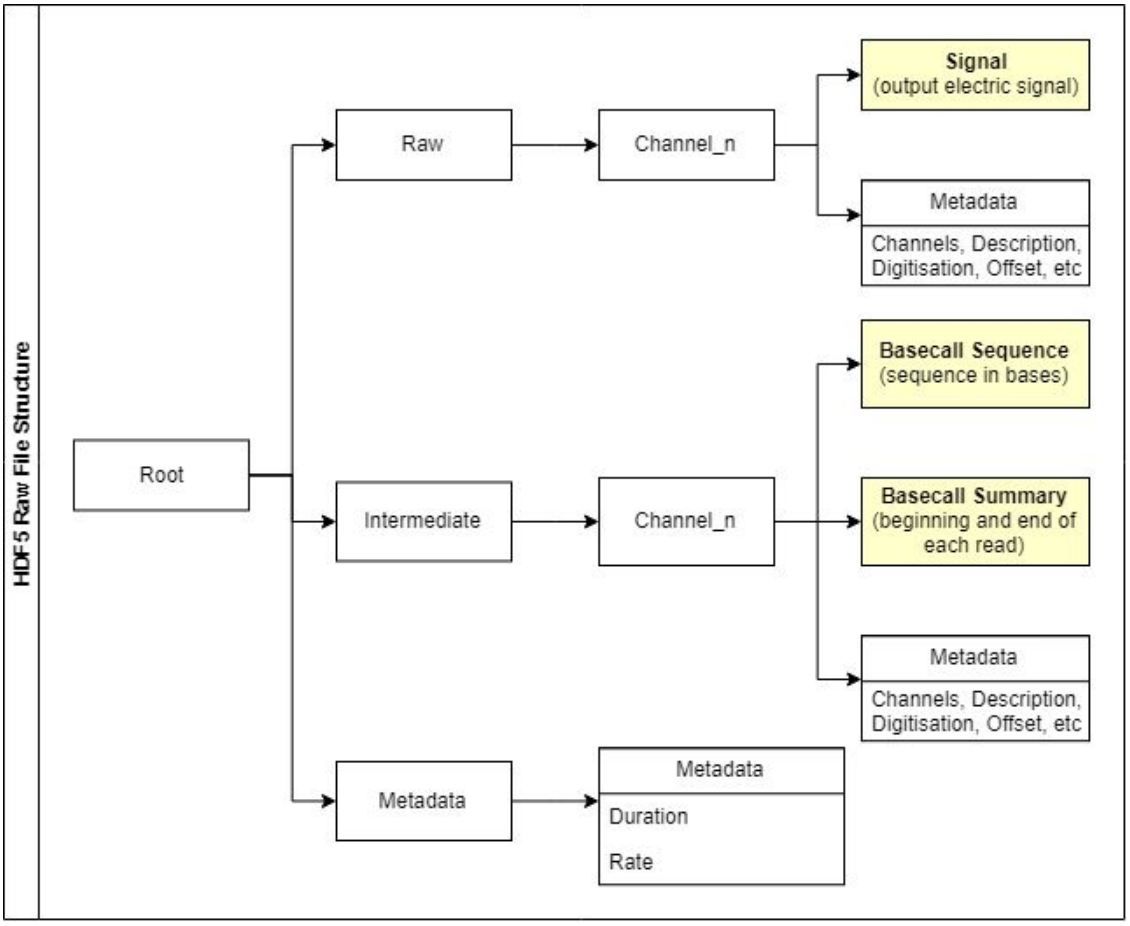


Figure 16: Structure of an HDF5 file generated by the MinION

8.2. Methods

The purpose of this project is to treat the disadvantages that the HDF5 format involves, in order to improve the performance of the database. To do that, we have developed a project coded in Python that introduces several improvements to the database.

Allowing more specific requests

Most of the time, researchers are just interested in a small part of the sequencing data: some specific reads or the data generated in a particular time interval. With the HDF5 generated by the MinION, it is difficult to filter out this specific information because the reads are not clearly separated and the time is remotely stored. All the data it is splitted over the three different Datasets: the raw electric signal is stored over time, the sequencing bases are stored over the electric signal and the read lengths are stored in a third dataset.

Normally this information is merged during the transformation to the text sequence file, that stores just the genetic sequence separated by reads. However, it might be interesting to manipulate this data from its raw format because a lot of information is lost during the transformation to text sequence file, like the electric signal that could help to detect and explain sequencing errors.
That's why we have developed an extension of the HDF5 format that we have called BULK file. It is essentially an HDF5 file but it has some additional features.

The BULK file is created from a MinION-generated HDF5 file, importing all the information. At the moment of creation, the BULK file allows to modify the existing Metadata or to add some, like the description of the sample, the offset or notes about the experiment.

The main use of the BULK file is that it has the option to generate new BULK files containing just specific reads or just the information extracted over a specific period of time. For instance, it can filter out the unimportant reads but keep all the information on the important ones.

Allowing storage on the internet

Another problem that can be solved with this project is the storage of the data. Some users may prefer lower speed in exchange for more complex queries. This is an advantage that only SQL can give, that's why we have added a functionality that converts HDF5 files and BULK files to SQLite.

The algorithm creates a new SQLite database already organized to store all the most important data and it imports the data from the HDF5/BULK file. The user is also given the option to choose where to store the databases, including the possibility to store it locally on a .sqlite file.

A conceptual Entity-Relationship diagram of the base can be found on the Figure 17.

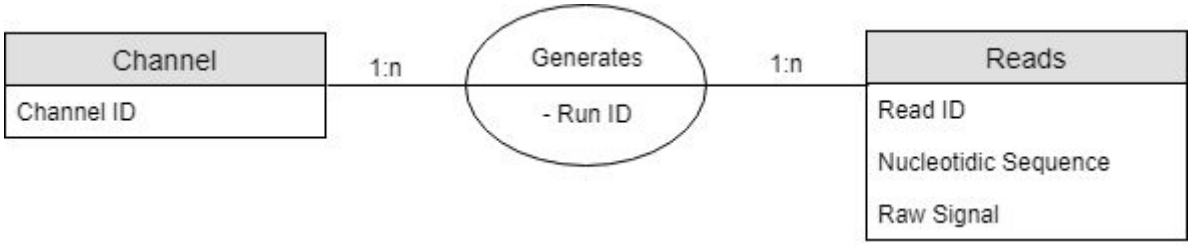


Figure 17. Entity-Relationship diagram with missing data.

Finally, to incorporate more data, the Channel entity has been replaced by the Metadata entity, resulting the Relationship Model on Figure 18.

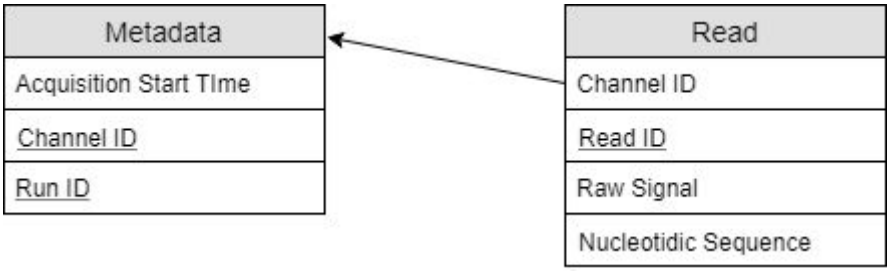


Figure 18: Relationship model to store sequencing results on the database.

8.3. Validation

Multiple tests have been written to ensure the correct performance of the algorithm. However, it will still pass through more development and performance analysis before it is released internally on the company. Unfortunately, my internship has finished before passing this checks so I won’t witness this process.

9. Bibliography

9.1. Cited

1. "MinION - Oxford Nanopore Technologies." https://nanoporetech.com/products/minion.

2. "The Oxford Nanopore MinION - Genome Biology - BioMed Central." 25 nov.. 2016, https://genomebiology.biomedcentral.com/articles/10.1186/s13059-016-1103-0.

3. "Sequencing DNA (or RNA) | Real-time, Ultra Long-Reads ... - YouTube." 19 jun.. 2017, https://www.youtube.com/watch?v=GUb1TZvMWsw.

4. "Mobile real-time surveillance of Zika virus in Brazil. - NCBI." 29 sept.. 2016, https://www.ncbi.nlm.nih.gov/pubmed/27683027.

5. "Real-time, portable genome sequencing for Ebola surveillance. - NCBI." 3 feb.. 2016, https://www.ncbi.nlm.nih.gov/pubmed/26840485.

6. "In-field metagenome and 16S rRNA gene amplicon ... - bioRxiv." 10 ago.. 2018, https://www.biorxiv.org/content/early/2018/08/10/073965.

7. "On site DNA barcoding by nanopore sequencing - PLOS." 4 oct.. 2017, https://journals.plos.org/plosone/article?id=10.1371/journal.pone.0184741.

8. "Field-based species identification of closely-related plants ... - NCBI." 21 ago.. 2017, https://www.ncbi.nlm.nih.gov/pubmed/28827531.

9. "Nanopore DNA Sequencing and Genome Assembly on ... - NCBI - NIH." 21 dic.. 2017, https://www.ncbi.nlm.nih.gov/pubmed/29269933.

10. "Product comparison - Oxford Nanopore Technologies." https://nanoporetech.com/products/comparison.

11. "Comparing Price and Tech. Specs. of Illumina MiSeq, Ion Torrent PGM ...." http://nextgenseek.com/2012/08/comparing-price-and-tech-specs-of-illumina-miseq-ion-torrent-pgm-454-gs-junior-and-pacbio-rs/.

12. "What is a Makefile and how does it work? | Opensource.com." 22 ago.. 2018, https://opensource.com/article/18/8/what-how-makefile.

13. "The only single product for the complete DevOps lifecycle - GitLab ...." http://gitlab.com/.

14. "GitHub - tryexceptpass/sofi: an OS agnostic UI module for Python." https://github.com/tryexceptpass/sofi.

15. "Selenium - Web Browser Automation." https://www.seleniumhq.org/.

16. "Pipenv: Python Dev Workflow for Humans — pipenv 2018.7.1.dev0 ...." https://docs.pipenv.org/.

17. "What is Docker? | Opensource.com." https://opensource.com/resources/what-docker.

18. "Efficient Depletion of Host DNA Contamination in ... - Semantic Scholar." https://pdfs.semanticscholar.org/386b/5c37cdb38bbe9aad3e4528e198522f031ace.pdf.

19. "Real-time DNA barcoding in a rainforest using nanopore sequencing ...." https://academic.oup.com/gigascience/article/7/4/giy033/4958980.

20. "Real time portable genome sequencing for global food security." 4 may.. 2018, https://nanoporetech.com/resource-centre/real-time-portable-genome-sequencing-global-food-security.

21. *"Mash Screen: what's in my sequencing run? – Genome Informatics ...." 25 sept.. 2017, https://genomeinformatics.github.io/mash-screen/.*

22. "Mash: fast genome and metagenome distance ... - Genome Biology." 20 jun.. 2016, https://genomebiology.biomedcentral.com/articles/10.1186/s13059-016-0997-x.

23. "The Cost of Sequencing a Human Genome - National Human ...." 6 jul.. 2016, https://www.genome.gov/27565109/the-cost-of-sequencing-a-human-genome/.

24. "Genome sequence data: management, storage, and ... - Future Science." https://www.future-science.com/doi/10.2144/000113134.

25. "HDF Group - HDF5 - The HDF Group." 29 nov.. 2017, https://www.hdfgroup.org/HDF5/.

9.2. Other used

1. "Generations of Sequencing Technologies: From First to Next Generation." https://www.omicsonline.org/open-access/generations-of-sequencing-technologies-from-first-to-next-generation-0974-8369-1000395.php?aid=87862.

2. "Oxford Nanopore Technologies." http://nanoporetech.com/.

3. "Real-time DNA barcoding in a rainforest using nanopore sequencing ...." https://academic.oup.com/gigascience/article/7/4/giy033/4958980.

4. "Protocol Buffers | Google Developers." https://developers.google.com/protocol-buffers/.

5. "grpc / grpc.io." https://grpc.io/.

Annexes

I. Annex A: gRPC

To understand what is gRPC, it is first needed to understand what is RPC

A.1. What is rpc?

RPC (Remote Procedure Call) is a protocol used by a computer to run code in another remote machine without having to worry for the communication between them.

It is mostly used on the client-server scheme, where a client tells the server to execute a process and send back the results, like if it had been executed on the client.

Traditionally, it is a synchronous process, this means that the program won't advance until it has received response (or multiple responses) or until stop signal is fired. However, multiple RPC can be calls can happen at the same time using threads. See Figure A.1 to see the workflow.

Most RPCs use an IDL to write the messages. An IDL (Interface Definition Language) is a neutral language that can be understood in any programming language. Thanks to that, the client and the server can be written in different programming languages without affecting the communication

A.2. Examples of use

There are many implementations of RPCs, like the ONC RPC from Sun, the DCOM from Microsoft or the gRPC from Google.

An easy example its use is on the Windows updates, where the client (the PC) will connect to the Windows server to check for updates, and if there is one, the server will answer with the updates.

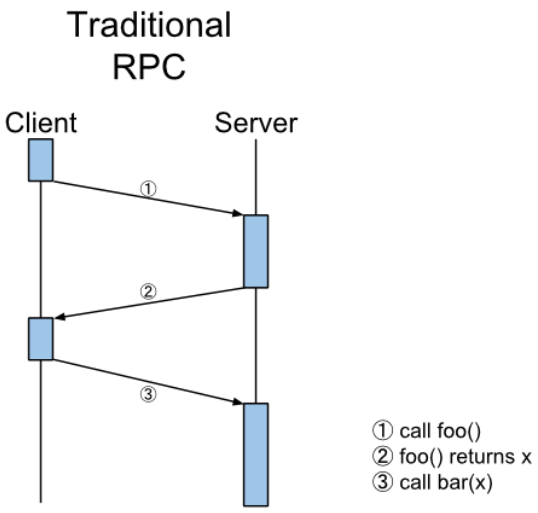


Figure A.1: Traditional RPC workflow.

A.3. What is grpc?

gRPC is a type of RPC developed by Google. It uses Protocol Buffers as IDL and HTTP/2 for data transport.

A.4. Advantages of grpc

gRPC is a modern framework of the RPC. Without explaining the technical details, it is more efficient and has more functions that other methods, like the asynchronous connectivity and the bidirectional connectivity

The messages are written in a neutral language called Protobuff in files called Protocol Buffers (abbreviated as Protobuff). Protobuffs support multiple programing languages as they can be automat automatically translated to Java, C++, Python, Objective-C, C#, Ruby, JavaScript, Go and lite-runtime

A.5. How is it related with this internship?

The gRPC is used by the project Mk Core Connector to communicate with the MinION. In this situation, Mk Core Connector acts as a Client and the MinION as a server. The Protobuff files are stored on Mk Core Connector and their automatic translation to Python is sent to the Host computers so that they can act a servers.

When using Mk Core Connector (or its UI), the Administrator will ask the Host computer to run the translated Protobuff code with their attached MinION's and return the results.

II. Annex B: Docker

B.1. What is Docker?

Dockerize is an open-source project that automatizes the *containerization* of applications. This means that it packs an application and its dependencies inside a virtual container that can be executed in any Linux server.

B.2. Why is it useful for?

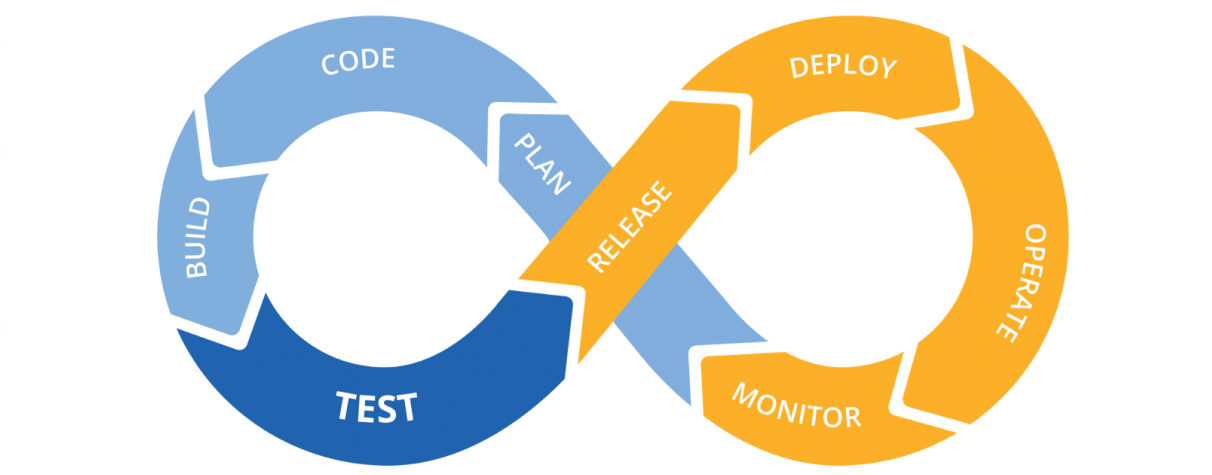
Docker allows you to use an application inside any Linux machine without having compatibility problems. Every container will pack all the necessary tools to run an application and the application itself.
For example, let's say that someone has created an application that needs Python 3 to run. If someone wants to run the application but has Python 2 installed instead of Python 3, he/she will have compatibility problems. Let's say that now the first person has packed the application in a Docker container with Java 8. Now there will be no problem running the application on any other machine, since the container having Java 8 inside, it will be independent from its environment and it use the Java 8 dependency that it packs.
This solves all compatibility problems between machines. The container will be executable by itself and any person that runs it will be able to run the application without problems.

B.3. How does it work?

Docker uses the most basic resources of a machine, the ones that are common to all operative systems. That way it doesn't need to install an operative system in order to execute the applications.

In that sense, it is different from a Virtual Machine, since the Virtual Machine needs to have an operative system installed in order to work. That's why we say that the Docker containers are lighter than the Virtual Machines.

III. Annex C: CD and CI pipeline



C.1. What is a Pipeline?

The concept “Pipeline” is a quite visual one. It consists of a chain of elements (processes, threads, coroutines, functions, etc) arranged so that the output of each one is the input of the next.

A pipeline automatizes the execution of jobs that have to be done sequentially.

C.2. What is a CI Pipeline

In a code written by multiple developers, every time that a developer adds new code to the existing one, it has to make sure that it doesn’t create issues.

In a CI pipeline, each time that a developper pushes new code, it generates a pipeline through which the code passes a serie of jobs split in different stages. Each job consists of some instructions that will test if the code is correctly written and compatible. As soon as a job fails, the pipeline fails, so the only way to push code is to successfully pass all the tests.

This procedure allows to make sure that multiple developers working can work on different parts of the same project without creating integration problems between these parts. Thanks to that, the code is solidly improved.

The fact that all the testing is automatized through a pipeline, ensures that all the tests are run and passed before merging, and makes the overall process easier and faster for the developer, that doesn't have to run it manually.

C.3. What is the CD (Continuous Delivery) Pipeline?

The CD (Continuous Delivery) pipeline is an evolution of the CI pipeline. On a CD pipeline, the code, a part from creating a build and passing all tests, has to be ready for release. This means that the project has to be always available for release at any point.

C.4. What is a CD (Continuous Deployment) Pipeline?

The CD (Continuous Deployment) pipeline is even more automatized than the Continuous Delivery. On the Continuous Deployment, the code is automatically released to the customers without human intervention so any new changes have to be able to do so before they can merge with the current code.