



Universidad
de La Laguna

Escuela Superior de
Ingeniería y Tecnología
Sección de Ingeniería Informática

Trabajo de Fin de Máster

NanoDJ: Una herramienta
bioinformática para el análisis de
datos del secuenciador de ADN
MinION

*NanoDJ: A bioinformatics tool for the analysis of
MinION DNA sequencer data*

Héctor Rodríguez Pérez

La Laguna, 2 de julio de 2019

D. **Marcos Colebrook Santamaría**, con N.I.F. 43.787.808-V, Profesor Titular de Universidad adscrito al Departamento de Ingeniería Informática y de Sistemas de la Universidad de La Laguna, como tutor

D. **José L. Roda García**, con N.I.F. 43.356.123-L, Profesor Titular de Universidad adscrito al Departamento Ingeniería Informática y de Sistemas de la Universidad de La Laguna, como cotutor

C E R T I F I C A (N)

Que la presente memoria titulada:

“Herramienta bioinformática usando Jupyter para el secuenciador de ADN MinION”

ha sido realizada bajo su dirección por D. **Héctor Rodríguez Pérez**, con N.I.F. 51.151.492-K.

Y para que así conste, en cumplimiento de la legislación vigente y a los efectos oportunos firman la presente en La Laguna a 2 de julio de 2019.

Agradecimientos

A mi familia, novia y amigos.

A mis tutores, Dr. Marcos Colebrook y Dr. José L. Roda por su ayuda y confianza durante todo este tiempo.

Y en especial al Dr. Carlos Flores, por hacer posible este innovador proyecto, y que todos esperamos que tenga futuro y largo recorrido.

Licencia



© Esta obra está bajo una licencia de Creative Commons
Reconocimiento-NoComercial-SinObraDerivada 4.0
Internacional.

Resumen

Este proyecto nace de la colaboración entre la Ingeniería Informática y la Biología dentro de la ULL. Continúa con la línea del Trabajo de Fin de Grado realizado en 2016 y tiene como objetivo el desarrollo de una herramienta de integración de programas y pipelines destinados al análisis de datos extraídos con el secuenciador de ADN MinION. Los datos originales que se utilizan para evaluar y trabajar con la solución desarrollada han sido aportados por el Dr. Carlos Flores.

NanoDJ tiene como finalidad ser de utilidad en el entorno docente y científico, y pretende facilitar el uso de las últimas herramientas de software libre destinadas a trabajar con este secuenciador dentro de un entorno aislado y de fácil instalación.

Palabras clave: Bioinformática, Jupyter Notebook, MinION, ADN.

Abstract

This project is born from the collaboration between Computer Engineering and Biology within the ULL. It continues with the line of the Final Degree Project carried out in 2016 and aims to develop a tool for integrating programs and pipelines for the analysis of data extracted with the DNA sequencer MinION. The original data used to evaluate and work with the solution developed have been provided by Dr. Carlos Flores.

NanoDJ aims to be useful in the teaching and scientific environment, and aims to facilitate the use of the latest free software tools designed to work with this sequencer in an isolated and easy to install environment.

Keywords: *Bioinformatics, Jupyter Notebook, MinION, DNA.*

Índice General

Capítulo 1. Introducción	1
1.1 Bioinformática.....	1
1.2 Minion.....	3
1.3 Jupyter.....	4
1.4 Docker.....	6
1.5 Objetivos y requisitos	7
Capítulo 2. Estado del arte	9
2.1 Secuenciación de ADN.....	9
2.2 Herramientas bioinformáticas actuales	10
Capítulo 3. Diseño y desarrollo de la solución	11
3.1 Definiendo el alcance y las características a incluir.....	11
3.2 Basecalling con Albacore	13
3.3 Porechop	14
3.4 Control de calidad con BioPython.....	15
3.5 Canu pipeline (Canu + Racon + Pilon).....	16
3.6 Canu y Nanopolish.....	17
3.7 Flye.....	17
3.8 Ensambladores híbridos.....	18
3.9 QUAST	19
3.10 NanoSim.....	20
3.11 Rebaler.....	20
3.12 Bandage	21
3.13 Cuaderno educativo.....	22
3.14 Construcción de la imagen Docker.....	24
3.15 Problemas y retos a la hora de empaquetar correctamente las aplicaciones necesarias.....	26
3.16 Análisis de la solución terminada	28

Capítulo 4. Resultados	29
4.1 Resultados numéricos	29
4.2 Resultados gráficos	32
Capítulo 5. Presupuesto	38
5.1 Recursos humanos	38
5.1.1 Ingeniero/a informático/a.....	38
5.1.2 Biólogo/a.....	39
5.2 Costes materiales.....	39
5.3 Costes totales	40
Capítulo 6. Resumen y Conclusiones	41
Capítulo 7. Summary and Conclusions	43
Bibliografía	45

Índice de figuras

Figura 1.1. Colocación de la muestra en el flowcell del MinION	3
Figura 1.2. El proyecto Jupyter	5
Figura 1.3. Esquema del funcionamiento de Docker	6
Figura 3.1. Vista de la interfaz de Jupyter Lab y el cuaderno de control de calidad.....	15
Figura 3.2. Captura de un fragmento del cuaderno de ensamblado con Canu+Racon+Pilon	16
Figura 3.3. Captura de un fragmento del cuaderno de ensamblado con Flye.	18
Figura 3.4. Captura de un fragmento del cuaderno de ensamblados híbridos	19
Figura 3.5. Informe de comparación de ensamblados generado con QUAST .	19
Figura 3.6. Captura de un fragmento del cuaderno de simulación de lecturas	20
Figura 3.7. Captura de un fragmento del cuaderno de Rebaler	21
Figura 3.8. Captura de la interfaz gráfica de Bandage	22
Figura 3.9. Captura de un fragmento del cuaderno educativo.....	23
Figura 3.10. Esquema de funcionalidades y herramientas de NanoDJ	24
Figura 3.3. Parte del fichero Dockerfile con las instrucciones para construir la imagen de NanoDJ	26
Figura 3.4. Vista general del repositorio de NanoDJ en GitHub	27
Figura 4.1. Distribución de longitud de lecturas.....	32
Figura 4.2. Distribución del porcentaje de contenido guanina-citosina de las lecturas.....	33
Figura 4.3. Distribución de las puntuaciones de calidad de las lecturas.....	34
Figura 4.4. Comparación entre las calidades de dos subconjuntos separados en dos rangos de contenido guanina-citosina	35
Figura 4.5. Visualización con Bandage del ensamblado híbrido con Unicycler	35

Figura 4.6. Visualización de la longitud y número de bases en las distintas muestras demultiplexadas con Porechop.....36

Figura 4.7. Proporción de lecturas clasificadas por BLAST para cada especie (no se incluyen la proporción de las no identificadas con ninguna referencia)37

Índice de tablas

Tabla 1.2. Conjuntos de datos utilizados para la evaluación y prueba de NanoDJ.....	13
Tabla 1.2. Resumen del ensamblado de las lecturas de <i>E.coli</i> utilizando el cuaderno de Canu + Racon + Pilon.....	29
Tabla 1.2. Comparación de tres ensambladores de novo para los datos en FASTA de <i>E. coli</i> K-12-MG1655.....	30
Tabla 1.2. Comparación de resultados de los ensamblados híbridos de los datos de <i>S.Agalactiae</i>	30
Tabla 1.2. Resultados de la simulación y clasificación de las lecturas de ADN mitocondrial en el ejercicio de metagenómica.	31
Tabla 5.1. Resumen del presupuesto para el Ingeniero/a Informático/a.....	38
Tabla 5.2. Resumen del presupuesto para el/la Biólogo/a.....	39
Tabla 5.3. Resumen del presupuesto para los materiales.....	39
Tabla 5.4. Resumen del presupuesto para los costes totales.....	40

Capítulo 1.

Introducción

1.1 Bioinformática

La bioinformática consiste en la aplicación de técnicas computacionales en entornos de análisis y gestión de datos de tipo biológico. Se trata de un campo de estudio multidisciplinar en el que se solucionan y se investigan problemas a escalas donde el uso intensivo de recursos computacionales se hace necesario. La gran mayoría de campos de estudio relacionados con la biología en los que se aplican estas técnicas son de índole molecular, siendo destacable el campo de la genómica, donde los avances tecnológicos tanto en computación como en instrumentación han permitido obtener resultados de forma menos costosa y con mayor cantidad de datos.

La **genómica** es la disciplina que estudia el conjunto completo del ADN contenido en el interior de cada célula, al que llamamos genoma. Los avances en genómica permiten un mayor conocimiento de la estructura, función y evolución del genoma de los seres vivos, y concretamente en el ámbito del genoma humano, ayuda a comprender las enfermedades con componente genético y permite el desarrollo de terapias génicas.

El material genético está formado por una cadena de doble hélice formada por unas moléculas denominadas nucleótidos o bases. En el caso del ADN, hay cuatro nucleótidos diferentes que podemos encontrar en cada posición. Estos son adenina, guanina, citosina y timina, representados usualmente con las letras A, G, C y T respectivamente. Por tanto, podemos representar una cadena de ADN como una cadena de caracteres que solo pueden contener las letras que representan a cada nucleótido. El tamaño del genoma es también bastante importante a la hora de trabajar. Se mide en pares de bases y sus múltiplos como pueden ser kilobases (Kbp) de 1.000 pares de bases, megabases (Mbp) de 1.000.000 pares de bases y así sucesivamente. Mientras

que una bacteria puede presentar un genoma de unas 3-4 Mbp, el genoma humano tiene un tamaño aproximado de 3,2 Gbp (unos 3200 millones de pares de bases).

Para obtener el genoma de un individuo se lleva a cabo el proceso de secuenciación del ADN. Los instrumentos de secuenciación leen el fragmento de ADN base por base, con un índice de error y longitud de la lectura que depende del instrumento utilizado. Como resultado de la secuenciación, se obtiene el orden de las bases de los fragmentos que se han leído, a los que llamamos **lecturas**. A continuación, estas lecturas deben pasar por una etapa de ensamblado para así obtener finalmente el genoma. Se puede realizar un ensamblado *de novo*, denominado de esta forma ya que se realiza desde cero y sin una referencia del organismo a ensamblar, y en el que las lecturas se van superponiendo unas con otras para formar fragmentos cada vez más largos. El resultado se denomina **secuencia de consenso** y puede estar formada por una única cadena o por varios fragmentos (**contigs**) que no se lograron unir.

Por otro lado, se pueden mapear las lecturas contra un genoma de referencia con el que se cuenta previamente. El mapeado de lecturas contra una referencia es un proceso que en el ámbito del estudio del genoma humano resulta interesante, ya que permite identificar las distintas variantes en el genoma de un individuo, así como la posición y gen donde se produce. La búsqueda e interpretación de estas variantes aportan información muy valiosa de cara por ejemplo a diagnosticar ciertas patologías o determinar el riesgo que tiene un individuo de padecer ciertas enfermedades.

La **metagenómica** estudia el material genético obtenido directamente de muestras ambientales. Este campo de estudios tiene aplicaciones también en el ámbito sanitario, donde la secuenciación de muestras extraídas del tracto respiratorio o digestivo aportan información valiosa que ayudan en el diagnóstico y pronóstico de ciertos tipos de patologías.

Para obtener toda esta información es necesario el desarrollo de algoritmos y herramientas que traten estos problemas de forma eficiente y sobre todo escalable. La **bioinformática** dentro del campo de la genómica supone una oportunidad para mejorar los servicios sanitarios y calidad de vida de las

personas ya que la convergencia de los avances en computación y secuenciación de ADN están poniendo al alcance de la comunidad científica la posibilidad de llevar a cabo nuevos descubrimientos y diseñar experimentos que hace varios años eran impensables dados los costes económicos y de tiempo y dinero.

1.2 Minion

El secuenciador de ADN MinION [1] fue lanzado en el año 2014 por la empresa *Oxford Nanopore Technologies (ONT)*. Se trata de un secuenciador de bajo coste con respecto a las opciones disponibles antes de su lanzamiento y con un tamaño menor que el de un *smartphone*, lo que lo convierte en la herramienta ideal para realizar trabajos fuera del laboratorio.



Figura 1.1. Colocación de la muestra en el flowcell del MinION

El funcionamiento del MinION difiere de cualquier otra tecnología de secuenciación anterior. Utiliza una pieza desechable denominada *flowcell*, en la que se coloca la muestra de ADN previamente preparada para ser secuenciada. Se conecta vía USB a un ordenador y se inicia la secuenciación, asistida por un software (Metrichor) que provee el fabricante. La interfaz de Metrichor muestra el estado de la secuenciación en tiempo real y aporta visualizaciones y métricas para medir el rendimiento durante el proceso.

La lectura de las bases de la cadena de ADN se realiza gracias una membrana con poros que se extiende por la *flowcell* y por la que circula una pequeña

corriente eléctrica. Conforme una cadena va atravesando uno de esos poros, se registra una diferencia en el potencial de corriente en este. El MinION registra estos cambios de potencial, a los que denomina eventos. Estos eventos junto con información adicional son almacenados en unos ficheros con formato FAST5 (la cual es una variante del estándar HDF5) utilizado para el almacenamiento de colecciones complejas de metadatos.

A partir de estos ficheros, se pueden obtener los ficheros de secuencia finales por medio de un proceso de nombrado de bases o *basecalling*. Actualmente *Oxford Nanopore* distribuye un software basado en redes neuronales profundas para llevar a cabo la conversión de eventos a bases.

El posterior análisis de los ficheros de secuencias requiere la utilización de software ajeno a la empresa. Estas herramientas son desarrolladas y distribuidas por la comunidad y llevan a cabo distintas tareas de análisis. Por tanto, todo el trabajo posterior, es decir, el análisis de los datos desde que se extraen los ficheros de secuencia necesita del uso de software bioinformático y distintas técnicas de análisis para la obtención de cualquier tipo de resultado.

Además, dada la diferencia entre las características de las lecturas obtenidas con MinION y las obtenidas con otras tecnologías, parte de las herramientas utilizadas para procesos de ensamblado y mapeado han tenido que ser desarrolladas específicamente para este tipo de lecturas.

1.3 Jupyter

Jupyter [2] es el proyecto sucesor de IPython notebooks, una aplicación web que permite la creación de documentos interactivos estructurados en celdas que pueden contener tanto código en Python, R u otros lenguajes, así como texto en Markdown, fórmulas en LaTeX imágenes y gráficas. La principal ventaja es que con la aplicación de Jupyter, podemos ejecutar las celdas de código que hemos escrito desde el mismo documento, que se visualiza en un navegador web.

Esto permite que las anotaciones o explicaciones que se suelen plasmar en documentos de protocolo dentro del ámbito de trabajo científico puedan

hacerse en la misma ventana desde la que ejecutamos código y visualizamos resultados, en lugar de elaborar un documento por separado para tal fin. Como resultado, integramos código, visualizaciones de datos y apuntes que no solo mejoran el entorno de trabajo, sino que tienen un gran valor en la comunicación científica y docente.

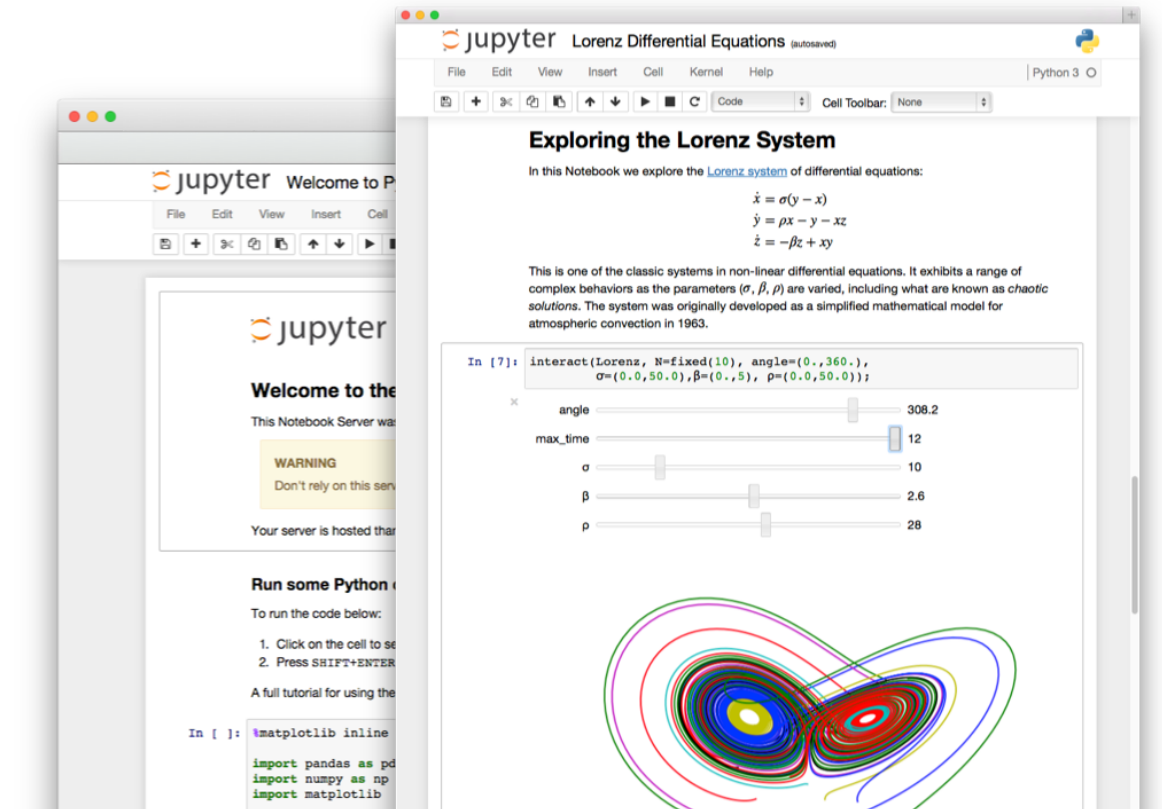


Figura 1.2. El proyecto Jupyter

Los cuadernos de Jupyter ya han tomado protagonismo en el ámbito científico sirviendo como plataforma para presentar resultados. Este es el caso del descubrimiento de las ondas gravitacionales y proyectos relacionados con el virus del Zika [3].

Actualmente, el proyecto ha evolucionado hasta el nacimiento de **Jupyter Lab**, que presenta una interfaz mejorada y permite trabajar con varios cuadernos abiertos a la vez o el desarrollo de plugins.

En resumen, el avance tecnológico ha ido provocando una convergencia cada vez mayor entre el campo de la biología y la informática. Esto se puede comprobar observando la evolución en el desarrollo y distribución del software

biológico y las necesidades en cuanto a recursos que este necesita en muchos casos.

1.4 Docker

Docker[3] es una aplicación para la automatización del despliegue de aplicaciones dentro de contenedores software proporcionando una capa de abstracción y virtualización a nivel de sistema operativo. La sencillez a la hora de construir imágenes y desplegar contenedores de forma rápida posibilita la creación de entornos distribuidos donde varias aplicaciones se despliegan en una o varias máquinas físicas o virtuales. Además, permite el despliegue de aplicaciones o servicios a medida que se dispone de los recursos necesarios o realizar despliegues bajo demanda.

Docker permite subir y descargar imágenes de su repositorio oficial, *DockerHub*. Los usuarios pueden administrar imágenes almacenadas en la nube y poner a disposición de la comunidad las suyas propias. Todo esto se realiza desde la línea de comandos y facilita enormemente el desarrollo y reutilización de imágenes ya estén en local o no.

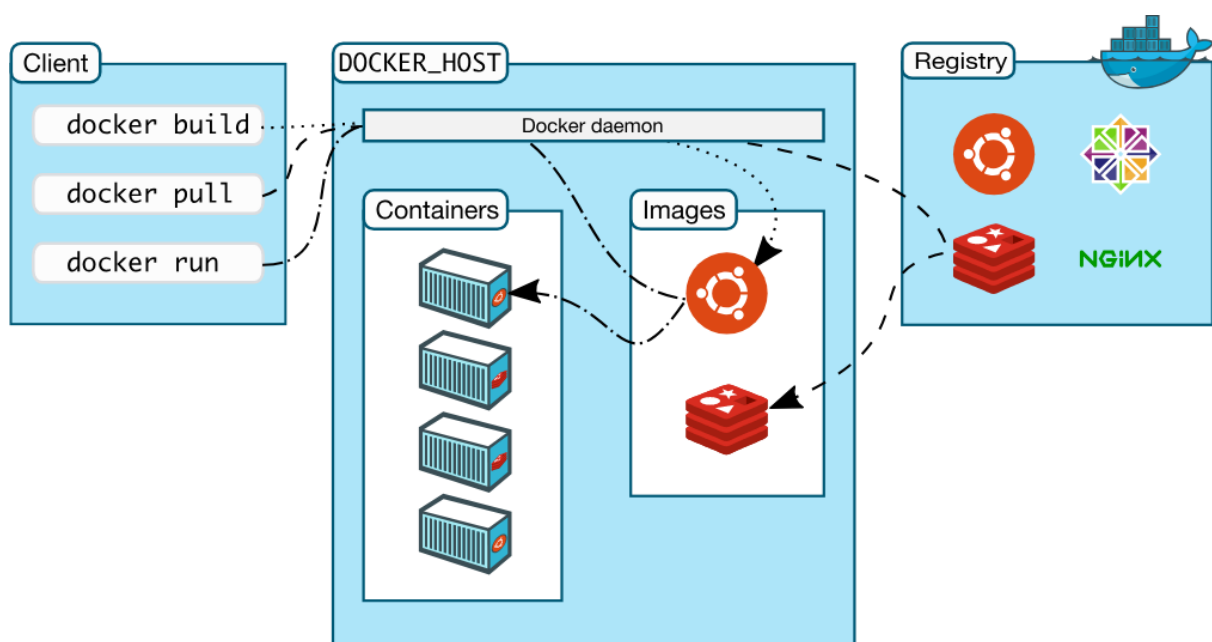


Figura 1.3. Esquema del funcionamiento de Docker

A día de hoy, la reproducibilidad de los resultados publicados, un obstáculo a la hora de comprobar la rigurosidad de un estudio o experimento se ha facilitado gracias a la popularización de estos entornos de virtualización como Docker, que permiten desplegar de forma sencilla contenedores con pipelines ya contruidos y sin merma de rendimiento frente a la ejecución directamente en nuestra máquina. De esta manera, el colectivo científico se beneficia y participa de la comunidad *open source* haciendo uso de herramientas inicialmente desarrolladas con la finalidad de ser muy útiles en otros ecosistemas como el empresarial, sobre todo en el ámbito de la infraestructura.

1.5 Objetivos y requisitos

El objetivo de este trabajo es, siguiendo la misma línea que en el TFG análogo [5], crear un entorno fácilmente desplegable dotado de las aplicaciones y herramientas necesarias para trabajar con datos extraídos del secuenciador MinION, sobre todo en el ámbito del ensamblado de bacterias y en el de la metagenómica.

Junto con este entorno, también se incluirán distintos cuadernos de Jupyter, que realizarán paso por paso y de forma guiada, algunos de los procesos más comunes dentro del estado del arte de las áreas a las que se destina este entorno. Los cuadernos estarán separados según su uso y se podrán usar de forma secuencial, partiendo de datos en crudo hasta lograr por ejemplo un ensamblado, o llevar a cabo una tarea concreta a partir de datos intermedios con los que se cuente previamente.

Se construirán cuadernos para el pre-procesado de datos en crudo, controles de calidad, varias estrategias o pipelines de ensamblado, simulación de lecturas y visualización de resultados.

Finalmente, se distribuirá este entorno en forma de imagen de Docker, con los materiales adicionales como los cuadernos, datos de prueba y documentación incluidos en un repositorio de GitHub. La naturaleza del proyecto y las herramientas utilizadas para su desarrollo harán posible también que a lo

largo del tiempo se vayan actualizando contenidos conforme al estado del arte de esta disciplina.

NanoDJ tiene como objetivo facilitar el acceso a un entorno para el análisis y construcción de pipelines para datos de MinION. Una vez descargada la imagen Docker, el entorno de NanoDJ se despliega en segundos en cualquier máquina con la aplicación Docker instalada. De esta forma, el usuario, sobre todo el profesional de la biología sin experiencia en la instalación y preparación de entornos de trabajo bioinformático, puede beneficiarse también de las últimas herramientas y *pipelines* disponibles sin invertir tiempo en la instalación y preparación de rutinas.

Capítulo 2.

Estado del arte

2.1 Secuenciación de ADN

La secuenciación de ADN ha ido disminuyendo en costes con el paso del tiempo. De los 100 millones de dólares que costaba en 2002, se ha llegado al punto en el que secuenciar el genoma de un individuo cuesta menos de 1000 dólares. Desde ese año hasta hoy, las tecnologías de secuenciación han ido cambiando, pasando por varias etapas o generaciones. Cada generación y tecnología, tiene sus características, como por ejemplo el índice de error, la longitud de las cadenas que es capaz de secuenciar o incluso el tiempo que se invierte desde la extracción del material genético hasta la obtención de los ficheros de lecturas.

NGS (*Next Generation Sequencing*) es el término designado para el conjunto de tecnologías de secuenciación que han hecho posible la disminución en el coste y la posibilidad de llevar a cabo descubrimientos y estudios en el contexto del genoma humano. En este contexto se encuentra el secuenciador MinION que además aporta una característica novedosa frente a instrumentos de generaciones pasadas; la capacidad de secuenciar lecturas largas. Aunque el índice de error de este aparato es alto en comparación con el de otros secuenciadores y, por ello dificulta su uso para experimentos en el ámbito de la variación genética y enfermedad en humanos, al menos ha abierto la puerta a nuevas técnicas de ensamblado o experimentos de metagenómica. El hecho de que las lecturas sean largas facilita la identificación de las especies con mayor acierto y detalle, lo que unido al bajo coste y la sencillez a la hora de preparar las muestras, hará posible que la identificación de patógenos en humanos por medio de la secuenciación pueda ser una realidad dentro del ámbito sanitario en unos años.

2.2 Herramientas bioinformáticas actuales

La mayoría de los estudios en los que se hace uso de técnicas y herramientas bioinformáticas se realizan en entornos UNIX. Las tareas que se realizan consisten en el tratamiento y análisis de datos, ya sea en bases de datos o más comúnmente en ficheros de texto con formatos ya estandarizados. Esto hace posible el uso de *scripts* que se apoyan en herramientas del ecosistema UNIX para llevar a cabo ciertas tareas aparte del software que se desarrolla para llevar a cabo procesos más complejos. Estos paquetes software no suelen recibir como entrada ficheros con datos directamente extraídos de instrumentos y tampoco tienen como salida los resultados finales del experimento, sino que a menudo sirven como paso intermedio en procesos de análisis facilitando la reutilización de un mismo software en varios contextos.

Llevar a cabo un estudio con este tipo de datos conlleva el diseño de *pipelines*, donde en cada etapa, se define de forma concreta qué se recibe como entrada y cuál será la salida. En un principio, se realizan controles de calidad destinados a mejorar y hacer más preciso el desempeño de las herramientas en las siguientes etapas.

Una vez se cuenta con los datos ya filtrados, el pipeline sigue su ejecución, pasando por las distintas herramientas software, que no tienen por qué estar construidas de la misma forma o con el mismo lenguaje.

En este sentido, se puede observar que un estudio que necesita de la aplicación de técnicas bioinformáticas precisa del diseño de *pipelines* basados en herramientas desarrolladas por la comunidad científica, y que en la mayoría de los casos no dan como salida resultados finales o reciben como entrada datos en crudo.

Capítulo 3.

Diseño y desarrollo de la solución

3.1 Definiendo el alcance y las características a incluir

En el ámbito de este trabajo, se trabajará fundamentalmente con ficheros de secuencia. Los formatos estándar para almacenar esta información son **FASTA** y **FASTQ**, y se detallan a continuación.

Un fichero **FASTA** es un fichero de texto que contiene una o varias secuencias. Cada secuencia se incluye en una nueva línea y viene precedida de otra línea de cabecera que comienza con el carácter “>” y que aporta información sobre la lectura que va a continuación.

```
>SEQ_ID  
GATTTGGGGTTCAAAGCAGTATCGATCAAATAGTAAATCCATTTGTTCAACTCACAG  
TTT
```

Los ficheros **FASTQ** tienen una estructura parecida. Por cada secuencia incluida en el fichero hay una línea de cabecera que comienza con el carácter “@” y que identifica la lectura. A continuación, se incluye la línea con la secuencia y además se incorporan dos nuevas líneas con información adicional. La primera comienza por “+” y opcionalmente va acompañada de nuevo por el identificador de la lectura e información adicional. La línea restante presenta las calidades codificadas de cada base de la secuencia, por lo que debe contener el mismo número de caracteres que la línea de la secuencia.

```
@SEQ_ID  
GATTTGGGGTTCAAAGCAGTATCGATCAAATAGTAAATCCATTTGTTCAACTCACAG  
TTT
```

```
+  
! ' ' * ( ( ( ( * * * + ) ) % % % + + ) ( % % % ) . 1 * * * -  
+ * ' ' ) ) * * 55CCF > > > > > > CCCCCCCC 65
```

Para llevar a cabo la búsqueda y evaluación de las herramientas que se incluirán en NanoDJ, primero se han definido los **casos de uso** que se van a abarcar inicialmente:

- Obtención de ficheros FASTA y FASTQ a partir de datos en crudo del MinION (en formato FAST5).
- Control de calidad de las lecturas obtenidas, con posibilidad de realizar filtros, y visualizar resultados.
- Ensamblado *de novo* y mapeado de lecturas contra una referencia.
- Ensamblado híbrido haciendo uso de lecturas obtenidas a partir de distintos instrumentos.
- Simulación de lecturas.

Una vez se han definido los casos de uso se pasa a la fase de evaluación de herramientas. Por cada caso que se ha propuesto, se prueban las herramientas y pipelines disponibles para ese fin. Estas herramientas con sus ejemplos de uso pueden encontrarse referenciadas en artículos científicos, publicaciones y foros.

Se creará un cuaderno por cada *pipeline* independiente o herramienta con un uso determinado, es decir, no solo contaremos por ejemplo con un cuaderno para la parte de ensamblado *de novo* sino que por cada *pipeline* y conjunto de herramientas destinadas a ese fin, se construirá un cuaderno que tenga como resultado final el ensamblado.

El proceso de evaluación de herramientas y creación de los cuadernos finales que se incluyen en NanoDJ se realiza en una máquina Linux, en la que se instalarán directamente estas herramientas, así como Jupyter Lab. Para la elección final de herramientas y el enfoque que se les dará en los cuadernos, se tendrán en cuenta los siguientes criterios:

- Facilidad de uso.
- Calidad de los resultados.

- Eficiencia en tiempo/espacio.

Para la elección de las herramientas se ha llevado a cabo una evaluación con los siguientes conjuntos de datos pertenecientes a lecturas extraídas con MinION.

Organismo	Tamaño (Mbp)	Tipo de ficheros	Número de lecturas (X1000)	Fuente
E. coli	4.6	FAST5/FASTA	164.47	Web[6]
S. agalactiae	2.2	FAST5/FASTA	3.72/2.975	Este estudio
H. Sapiens (mitocondrial)	0.016	FAST5	59.36	Web[7]
Vertebrados (mitocondrial)	Variable	FASTA	0.21	Web[8]

Tabla 3.1. Conjuntos de datos utilizados para la evaluación y prueba de NanoDJ

A continuación, se detalla el contenido de cada cuaderno así como una descripción de las herramientas utilizadas en estos. Los cuadernos no solo incluyen las celdas de código para ejecutar los programas sino que también van acompañados de explicaciones, anotaciones y referencias.

3.2 Basecalling con Albacore

Para este primer cuaderno, el usuario podrá traer los datos en crudo directamente extraídos del secuenciador. Estos datos vienen en formato FAST5. Dichos ficheros contienen metadatos e información sobre los eventos registrados en los poros del secuenciador, que serán transformados mediante el proceso de *basecalling*. Cabe destacar que el tamaño de estos ficheros es bastante grande en comparación con el tamaño que después tendrán nuestros ficheros de lecturas en formato FASTA y FASTQ.

El software destinado al *basecalling* es **Albacore**[9]. Esta herramienta la provee de forma oficial *Oxford Nanopore* y actualmente es capaz de ejecutarse desde el software *Metrichor* en tiempo real mientras el MinION realiza la secuenciación. Al tiempo de la realización de este trabajo, el proceso de *basecalling* no se podía hacer de forma simultánea a la secuenciación, sino que era el usuario el que manualmente ejecutaba Albacore para extraer las lecturas una vez terminaba la secuenciación.

Cabe destacar que, aunque solo se necesita un comando para llevar a cabo este proceso con todos los ficheros FAST5, es la tarea más exigente de las que se presentan en este trabajo, recomendando el fabricante el uso de GPUs cuando el número de ficheros de entrada es muy grande.

3.3 Porechop

Existe la posibilidad de secuenciar en una misma carrera varias muestras distintas. Esto es algo común en otras tecnologías y resulta bastante útil para obtener resultados de manera más rápida y disminuir los costes. Esto se realiza en el laboratorio, añadiendo antes del proceso de secuenciación unos códigos de barras distintos para cada muestra. Estos códigos de barras son pequeñas cadenas cuya composición es conocida y que se añaden al inicio y al final de cada fragmento (lectura). De esta forma, una vez se han obtenido los ficheros de lecturas, se pueden separar las secuencias que pertenecen a cada muestra sabiendo cuál fue el código de barras que se asignó a esta.

Para realizar esta tarea con lecturas de tipo Nanopore, se ha construido un cuaderno que hace uso de Porechop [10], una herramienta destinada a detectar códigos de barras y adaptadores en las secuencias, cortarlos y finalmente separar las lecturas en distintos ficheros de secuencias según la muestra a la que pertenecen.

En este paso no se obtienen resultados finales, sino que se separan las lecturas que fueron secuenciadas en una misma carrera para su posterior análisis en otros cuadernos. Por último, se ha incluido una visualización hecha con el paquete *matplotlib* [11] para observar la proporción de lecturas de cada muestra detectada y su calidad.

3.4 Control de calidad con BioPython

Una vez se han obtenido los ficheros de lecturas FASTA/FASTQ, es necesario llevar a cabo el **control de calidad** previo a otras tareas como la de ensamblado. En esta etapa, los datos son filtrados para evitar que lecturas de baja calidad empeoren los resultados futuros. Estos filtros eliminan por ejemplo lecturas muy cortas o con una calidad media por debajo de un umbral. Para este proceso, existen herramientas con interfaz por línea de comandos que realizan los filtros más comunes de forma directa. Sin embargo, y aprovechando que los cuadernos de Jupyter pueden ser una forma interactiva de hacer análisis de datos, se ha escogido BioPython [12], una librería del lenguaje Python destinada a trabajar con datos biológicos, incluido ficheros FASTA/FASTQ.

Haciendo uso de esta librería, podemos cargar nuestros ficheros de lecturas y su información básica en forma de *dataframe*, lo que nos permite realizar posteriormente filtros por calidad o longitud de lectura, además de visualizaciones con *matplotlib*. Una vez hecho el control de calidad, se puede volcar la salida filtrada en un fichero FASTA/FASTQ.

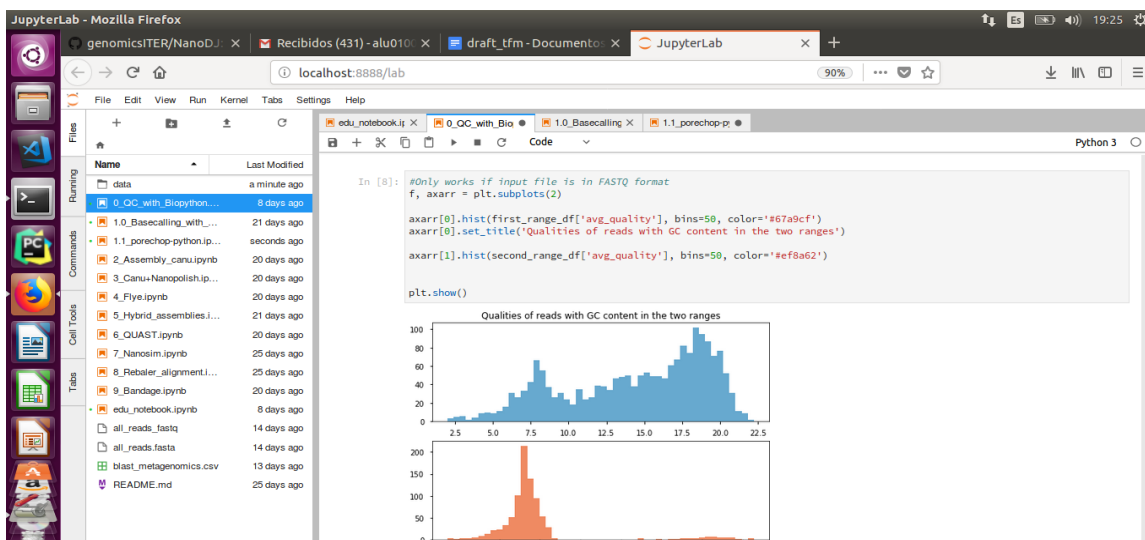


Figura 3.1. Vista de la interfaz de Jupyter Lab y el cuaderno de control de calidad

3.5 Canu pipeline (Canu + Racon + Pilon)

El primero de los cuadernos destinados al ensamblado *de novo* utiliza el ensamblador Canu [13], una herramienta que construye una secuencia de consenso pero que necesita del uso de herramientas de pulido para lograr buenos resultados.

En este caso, una vez se ha obtenido el ensamblado con Canu, se realiza un primer pulido con la herramienta Racon [14], que toma los *contigs* obtenidos por herramientas de ensamblado básicas y realiza mejoras apoyándose en las lecturas utilizadas para construir estos *contigs*, es decir, los ficheros de lecturas de los que partimos.

A continuación, se utiliza otra herramienta de corrección o pulido de ensamblados, Pilon [15]. Tanto para la ejecución de este programa como para el pulido con Racon, se hace uso de herramientas para mapear las lecturas contra una referencia (minimap [16]) y utilidades para el procesamiento intermedio de ficheros (samtools [17]).

Assembly with the Canu pipeline

Canu is a popular assembler based on Celera Assembler and built specifically to work with ONT reads. It consists on a 4-step pipeline that generates a 'draft assembly' without reference. In order to get better results, Canu is often used with tools that improves its result. In this notebook, we will build a popular Canu pipeline using Canu + Racon + Pilon.

Canu works with either FASTA or FASTQ files (compressed and uncompressed), but FASTQ format is needed to run the next steps and complete the full pipeline. Help page can be shown with "canu -h" command. These are the parameters needed for running Canu with our data:

- -p and -d: Assembly files prefix and output directory. Both parameters can be the same and output directory doesn't have to exist before execution.
- genomeSize: The estimated genome size. In our case, 2.1 mbp so we write '2.1m'. We can put letter g for gbp or k for kbp as well.
- -nanopore-raw: The path to our reads in FASTQ.

```
In [1]: canu -p agalactiae \
        -d data/agalactiae/canu_output \
        genomeSize=4.6m \
        useGrid=false \
        minReadLength=50 \
        minOverlapLength=50 \
        -nanopore-raw data/agalactiae/merged-output.fasta

-- Canu snapshot v1.7 +137 changes (r8829 73d5caa1b1087b65f7853ecbebc1bb1dcbd1bc14)
--
-- CITATIONS
```

Figura 3.2. Captura de un fragmento del cuaderno de ensamblado con Canu+Racon+Pilon

3.6 Canu y Nanopolish

Utilizando la misma herramienta para obtener un ensamblado inicial, en este cuaderno se opta por otra estrategia para el pulido de *contigs* obtenidos con Canu. En este caso se utiliza Nanopolish [18], que utiliza los ficheros en crudo de lecturas (FAST5) para realizar la corrección del ensamblado. De nuevo, Nanopolish no solo necesita de un ensamblado base y de las lecturas en crudo, sino que precisa también de algunos ficheros intermedios como un mapeado contra referencia que obtenemos con mapeadores como pueden ser minimap o BWA [19] (el utilizado en esta ocasión).

3.7 Flye

El último cuaderno destinado a realizar un ensamblado *de novo* utiliza la herramienta Flye [20] como ensamblador. Este software sigue una estrategia basada en la construcción de un grafo de ABruijn [21] para generar el ensamblado. Esta técnica ya era utilizada con éxito con lecturas cortas provenientes de otros instrumentos de secuenciación. Una vez realizadas las pruebas, se comprobó que es una buena opción para obtener ensamblados rápidos de una calidad aceptable en comparación con otras herramientas para ensamblar.

En este caso no se incluyó ninguna fase de corrección o pulido aunque dado que se obtiene un fichero de secuencias (*contigs*) como salida, también se pueden llevar a cabo este tipo de tareas utilizando las mismas herramientas y procedimientos que en los dos cuadernos anteriores.

De novo assembly with Flye

[Flye](#) is a de novo assembler for long and noisy reads such as ONT reads. Flye builds the ABrujin graph and performs an extra repeat classification and analysis step. It also has a polished module that produces the final assembly.

Flye basic arguments are:

- nanoraw FASTA reads file of ONT reads.
- out-dir Output directory.
- genome-size Size of the assembled genome.
- threads Number of available threads to perform the assembly.

```
In [ ]: flye --nano-raw data/agalactiae/merged-output.fastq --out-dir data/agalactiae/flye_output --genome-size 4.7m --threads 2
```

Related publications:

[Mikhail Kolmogorov, Jeffrey Yuan, Yu Lin and Pavel Pevzner, "Assembly of Long Error-Prone Reads Using Repeat Graphs", bioRxiv, 2018](#)

[Yu Lin, Jeffrey Yuan, Mikhail Kolmogorov, Max W Shen, Mark Chaisson and Pavel Pevzner, "Assembly of Long Error-Prone Reads Using de Bruijn Graphs", PNAS, 2016](#)

```
In [ ]:
```

Figura 3.3. Captura de un fragmento del cuaderno de ensamblado con Flye

3.8 Ensambladores híbridos

Se ha incluido un cuaderno con el que generar ensamblados haciendo uso de lecturas provenientes de distintas tecnologías. En este caso, las lecturas extraídas del MinION pueden complementarse con lecturas de máquinas Illumina o IonTorrent. Estas estrategias de ensamblado híbrido basan su efectividad en que se pueden construir contigs de gran tamaño utilizando lecturas largas para luego realizar correcciones y uniones entre contigs con ayuda de lecturas más cortas y precisas como las que genera, por ejemplo, la tecnología Illumina.

Se han incluido dos herramientas para llevar a cabo este tipo de ensamblados: Unicycler [22], con la que se obtienen muy buenos resultados ensamblando bacterias y MaSuRCA [23], destinada sobre todo al ensamblado de genomas más grandes y con una mayor cantidad de lecturas.

Hybrid assemblies with Unicycler

We can get an hybrid assembly based on ONT + Illumina reads with Unicycler. **Unicycler** is an assembly pipeline for bacterial genomes. IonTorrent reads can also be used to build an hybrid assembly with ONT reads. The pipeline has assembly and improvement steps and need the following inputs:

- 1 and -2: Illumina reads.
- l: Long reads (ONT in our case).
- s: IonTorrent reads.

REMARK: Data should not be included in /data folder of this project (see "Additional data" section in README file)

```
In [ ]: #Illumina agalactiae data to be released
./unicycler-runner.py \
-1 data/Agalactiae/Data_Illumina/Raw/WGS_bacterialIsolates_MiSeq_training-33608589/Sagalactiae_HRC-41
-2 data/Agalactiae/Data_Illumina/Raw/WGS_bacterialIsolates_MiSeq_training-33608589/Sagalactiae_HRC-41
-l data/agalactiae/merged-output.fastq \
-o trihibrido_output_albacore
```

Hybrid assembly with MaSuRCA

MaSuRCA is a de novo assembler which has the capacity to assemble both only short reads or a mixture of short and long reads.

MaSuRCA has a configuration file where arguments and input data paths are placed. Once the configuration file is ready, the "masurca" binary script will

Figura 3.4. Captura de un fragmento del cuaderno de ensamblados híbridos

3.9 QUAST

Una vez hemos generado ensamblados con las distintas estrategias propuestas, se puede realizar una **comparativa** entre las distintas salidas. QUAST [24] es una herramienta que recibe uno o más ensamblados distintos de una misma muestra y genera informes con comparativas y tablas de datos útiles.

Estos informes son creados en HTML por lo que pueden consultarse en el navegador de forma cómoda. Aparte, los datos y visualizaciones en los que se apoyan estos reportes, pueden ser consultados en ficheros aparte.

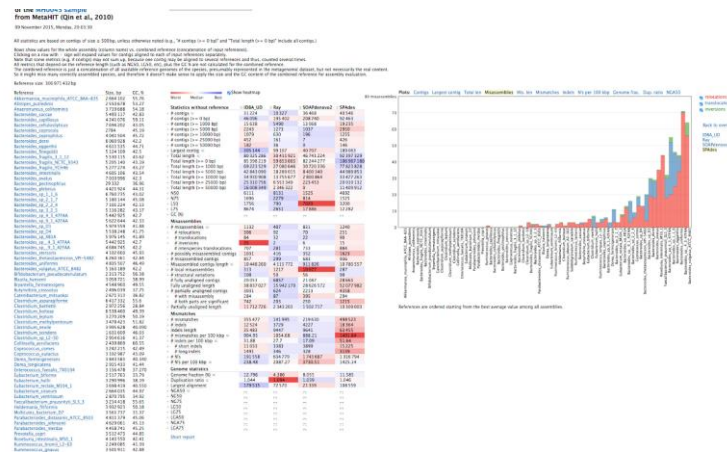
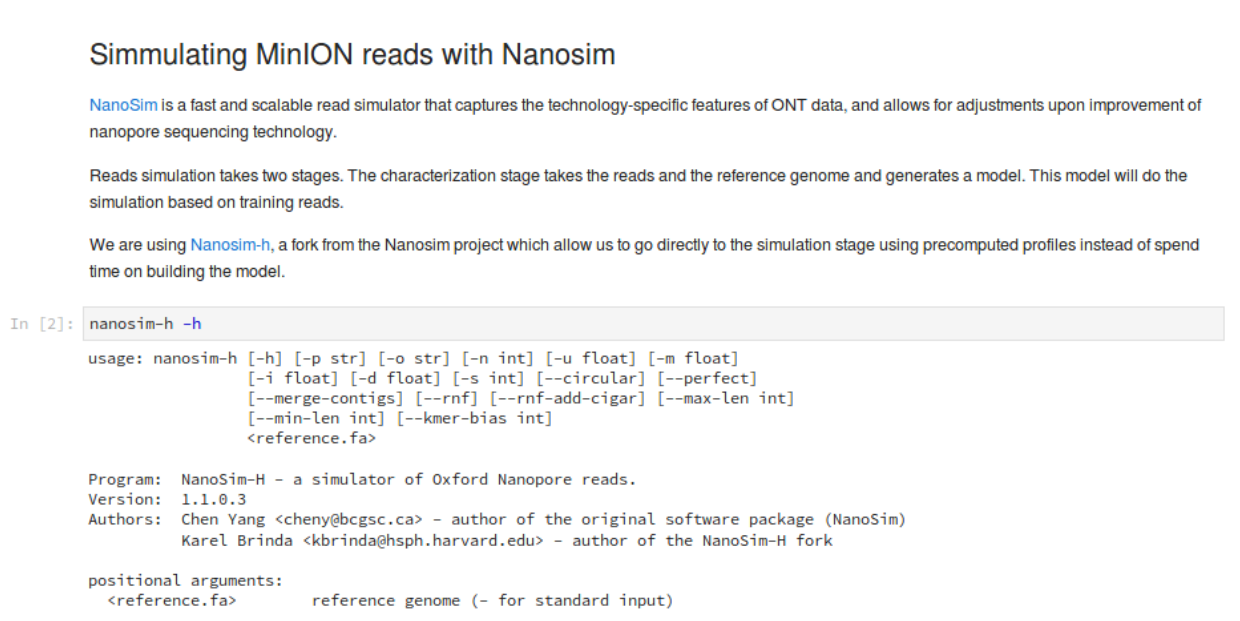


Figura 3.5. Informe de comparación de ensamblados generado con QUAST

3.10 NanoSim

NanoSim [25] es una herramienta para la simulación de lecturas de *Oxford Nanopore*. Esta herramienta crea un modelo estadístico a partir de un conjunto de lecturas extraídas directamente del instrumento (sin control de calidad previo). A partir de este modelo y una referencia, NanoSim crea un fichero de lecturas simuladas, donde la distribución de longitud de las lecturas o el índice de error dependen de los parámetros del modelo.

En este caso, para facilitar y agilizar la simulación de lecturas, se ha optado por NanoSim-h, un fork del proyecto original que incluye modelos predefinidos sobre los que empezar a simular lecturas. Estas lecturas simuladas son utilizadas posteriormente en el cuaderno educativo que se ha incluido con el proyecto.



```
In [2]: nanosim-h -h

usage: nanosim-h [-h] [-p str] [-o str] [-n int] [-u float] [-m float]
                [-i float] [-d float] [-s int] [--circular] [--perfect]
                [--merge-contigs] [--rnf] [--rnf-add-cigar] [--max-len int]
                [--min-len int] [--kmer-bias int]
                <reference.fa>

Program: NanoSim-H - a simulator of Oxford Nanopore reads.
Version: 1.1.0.3
Authors: Chen Yang <cheny@bcgsc.ca> - author of the original software package (NanoSim)
        Karel Brinda <kbrinda@hsph.harvard.edu> - author of the NanoSim-H fork

positional arguments:
  <reference.fa>      reference genome (- for standard input)
```

Figura 3.6. Captura de un fragmento del cuaderno de simulación de lecturas

3.11 Rebaler

Rebaler [26] es una herramienta que permite el ensamblado de genomas a partir de lecturas largas utilizando una referencia. Una vez se completa este

ensamblado, la herramienta ejecuta automáticamente varias etapas de pulido utilizando Racon, herramienta y usada en uno de los cuadernos de ensamblado *de novo* con Canu. La principal ventaja de esta herramienta es la posibilidad de obtener buenos resultados haciendo uso únicamente de este software.

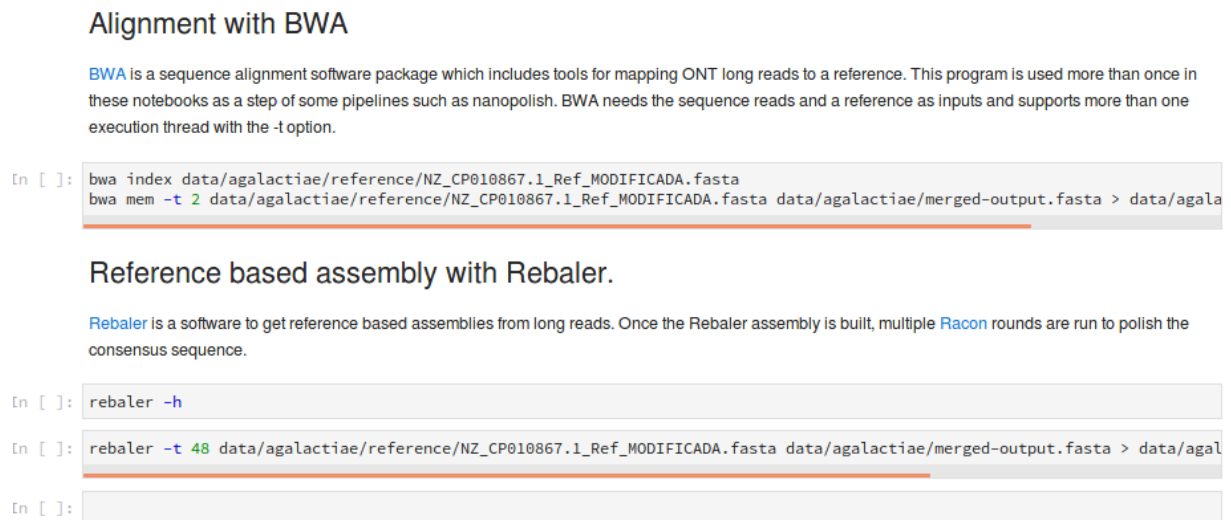


Figura 3.7. Captura de un fragmento del cuaderno de Rebaler

3.12 Bandage

Algunas herramientas de ensamblado, proveen como salida no solo el fichero de lecturas con los *contigs* resultantes sino que además añaden un fichero *.gfa* con el grafo del ensamblado. Estos ficheros no solo se contienen los *contigs* sino también incluyen uniones con otros *contigs* y caminos alternativos (un *contig* puede presentar dos o más uniones con otros *contigs* por el mismo lado).

Se ha incluido un cuaderno con Bandage [27], una aplicación para la visualización de este tipo de ficheros. Al tratarse de una aplicación con interfaz gráfica, la ejecución del comando que la inicia genera una ventana con esta interfaz y desde la que se cargan los ficheros de entrada y se visualizan.

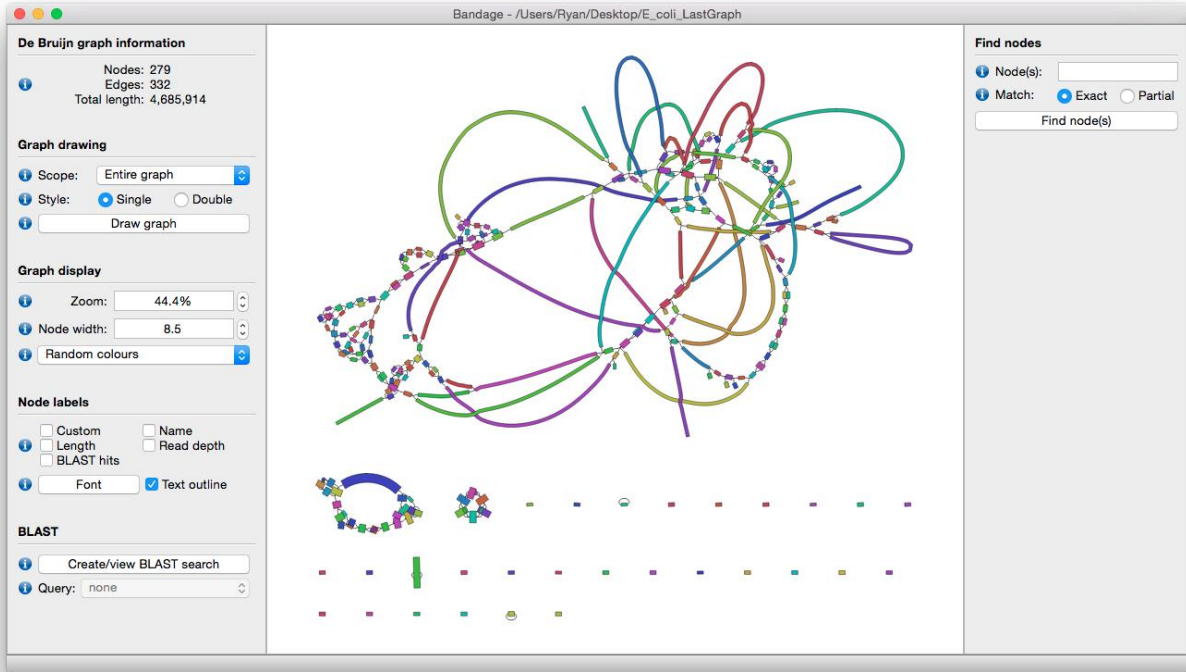


Figura 3.8. Captura de la interfaz gráfica de Bandage

3.13 Cuaderno educativo

Aparte de los cuadernos dirigidos a llevar a cabo tareas concretas, se ha construido un cuaderno en el que se explica un ejemplo de datos con MinION en el contexto de un ejercicio de metagenómica y con la finalidad de ser utilizado en el entorno docente.

Se comienza realizando controles de calidad y visualizaciones de los datos de partida con BioPython. Estos datos (incluidos en el proyecto) son obtenidos a partir de NanoSim-h, que en este caso simula un mismo número de lecturas para 7 genomas de referencia pertenecientes a distintas especies animales. Las lecturas simuladas son mezcladas posteriormente en un mismo fichero, simulando que pertenecen a una muestra ambiental, como ocurre en la metagenómica.

Tras el control de calidad, se utiliza el software de alineamiento BLAST [28] para crear una base de datos con las referencias de nuestras lecturas. Una vez

tenemos la base de datos creada, realizamos una consulta con el fichero que contiene las lecturas. BLAST devuelve un fichero *.csv* que contiene la especie que identifica en cada lectura. Por último, se utiliza este fichero csv para generar una visualización de la proporción de lecturas de cada especie.

En el siguiente esquema se muestran las distintas funcionalidades y herramientas de NanoDJ separadas según su finalidad. También se presenta una vista general del pipeline completo desde la extracción de lecturas hasta la comparación y visualización de ensamblados.

```
arguments:

In [8]: !makeblastdb -in data/metagenomics/references/database/metagenomics_references_withnames.fasta -parse_seqids -dbtype nucl

Building a new DB, current time: 06/30/2018 14:52:01
New DB name: /home/jovyan/notebooks/data/metagenomics/references/database/metagenomics_references_withnames.fasta
New DB title: data/metagenomics/references/database/metagenomics_references_withnames.fasta
Sequence type: Nucleotide
Deleted existing Nucleotide BLAST database named /home/jovyan/notebooks/data/metagenomics/references/database/metagenomics_references_withnames.fasta
Keep MBits: T
Maximum file size: 1000000000B
Adding sequences from FASTA; added 7 sequences in 0.00670505 seconds.

Once our database is created, we are ready to query it. The input is a FASTA file with 210 unidentified sequences. The query command has the following arguments:

In [10]: !blastn -query data/metagenomics/metagenomics_exercise_sample.fa -db /home/jovyan/notebooks/data/metagenomics/references/d

We get the output on csv format as it is easier to be processed by the pandas package to analyze and plot the results.

Data must be filtered in order to know which sequences aligned with any reference and which sequences did not. The data is grouped by subject so we have the count of sequences that aligned with each reference.

Finally, the number of sequences aligned to a reference is transformed into the proportion of reads that are aligned to that reference.
```

Figura 3.9. Captura de un fragmento del cuaderno educativo

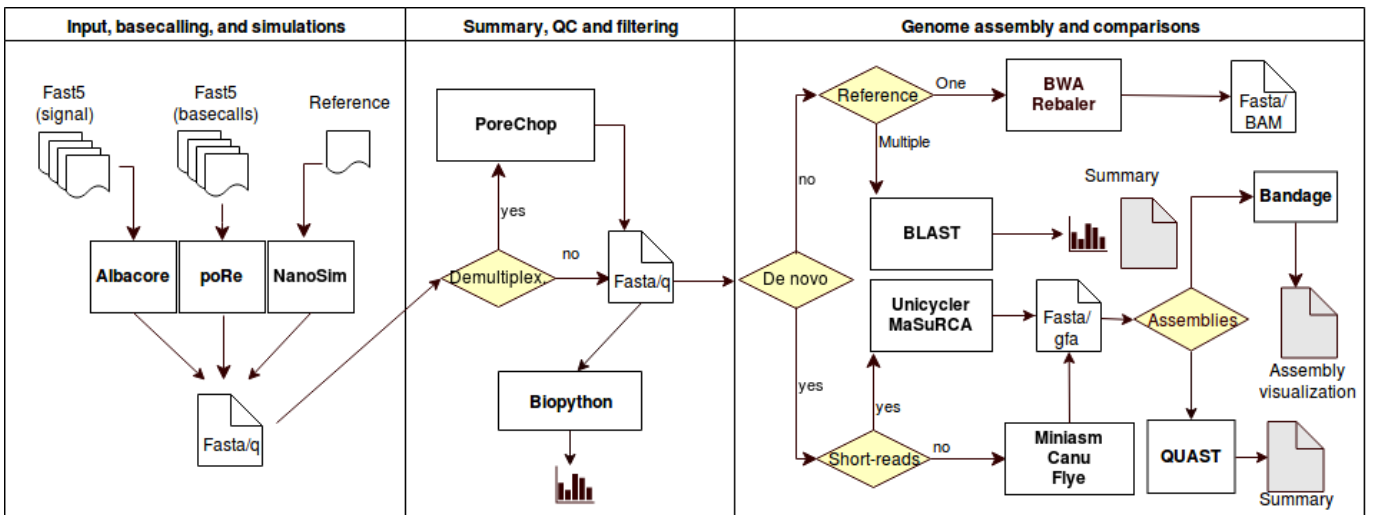


Figura 3.10. Esquema de funcionalidades y herramientas de NanoDJ

3.14 Construcción de la imagen Docker

Con los cuadernos y su contenido creados y evaluados, se pasa a la fase de creación de una imagen Docker que reúna todas las herramientas necesarias para la correcta ejecución de todos los cuadernos.

Aunque en un principio la filosofía de Docker se basa en una imagen para una aplicación, en nuestro caso, no es factible crear una imagen por cada herramienta incluida. Por un lado, las herramientas deben ser ejecutadas de forma natural, mediante línea de comandos sobre un entorno en el que ya estén instaladas, puesto que lanzar un contenedor por cada aplicación que se desee arrancar desde un cuaderno puede resultar en una merma de la experiencia de uso, además de resultar en un entorno mucho más complejo.

Por tanto, se pretende construir una sola imagen que contenga de base la aplicación Jupyter Lab, y en la que además se instalen las herramientas presentes en los cuadernos, de forma que lanzando el contenedor y abriendo un cuaderno de Jupyter, se pueda ejecutar cualquiera de las herramientas desde este.

La primera decisión a la hora de construir la imagen es sobre cuál se va a trabajar de base. Es posible crear una imagen Docker a partir de una imagen de un sistema operativo o utilizando otra imagen con aplicaciones ya instaladas.

Con la idea de construir nuestra imagen a partir de una que ya incluya algunas de las herramientas básicas como pueden ser lenguajes como Python, R, o la aplicación de Jupyter Lab, se ha escogido una de las imágenes que ofrece el proyecto Docker Stacks [29]. Concretamente la imagen *datascience-notebook*, basada en Ubuntu, ya incluye lenguajes y herramientas utilizadas en el ámbito de la ciencia de datos.

Una vez escogida la imagen base, se comienza a construir la nuestra utilizando un fichero *Dockerfile*. En este fichero se detalla, desde qué imagen base se quiere partir, y cuáles son los comandos y acciones que deben ir ejecutándose paso a paso para la construcción de la imagen final.

En primer lugar, se ejecutan comandos para descargar paquetes y librerías desde el repositorio oficial *apt*. Esto se realiza en primer lugar ya que la instalación de las herramientas principales requiere de dependencias que no se incluyen en la imagen base.

A continuación, se ejecutan los comandos para descargar e instalar las herramientas en el directorio */home/jovyan/software* dentro del contenedor. Una vez se completa la instalación de todas las herramientas, se añaden al *PATH* del sistema para que puedan ser ejecutadas desde cualquier ubicación.

Cabe destacar que las imágenes base que ofrece Docker, así como otras imágenes disponibles no siempre contienen las librerías y paquetes de software que por ejemplo se incluyen por defecto en las imágenes ISO de distribuciones Linux, por lo que a medida que se instalan o se prueban las aplicaciones que utilizamos en los cuadernos, se deben ir incluyendo en el fichero *Dockerfile* las dependencias de las que hace uso.

```

RUN pip install bash_kernel biopython nanosim-h jupyterlab
RUN python -m bash_kernel.install

RUN git clone https://github.com/lh3/minimap2 && \
  cd minimap2 && \
  make && \
  cd ..

RUN git clone --recursive https://github.com/isovic/racon.git racon && cd racon && mkdir build && cd build && \
  cmake -DCMAKE_BUILD_TYPE=Release .. && \
  make

RUN git clone https://github.com/rrwick/Unicycler.git && cd Unicycler && make && cd ..

RUN git clone https://github.com/lh3/miniasm && (cd miniasm && make)

RUN git clone https://github.com/marbl/canu.git && \
  cd canu/src && \
  make -j 1

RUN git clone https://github.com/ablab/quast.git && \
  cd quast && \
  ./setup.py install \
  && cd ..

RUN git clone https://github.com/rrwick/Porechop.git && \
  cd Porechop && \
  python3 setup.py install

RUN git clone https://github.com/rrwick/Rebaler.git && \
  cd Rebaler && \
  . . . . .

```

Figura 3.11. Parte del fichero Dockerfile con las instrucciones para construir la imagen de NanoDJ

3.15 Problemas y retos a la hora de empaquetar correctamente las aplicaciones necesarias

La primera observación que debe hacerse sobre la construcción de la imagen es que los ficheros de los cuadernos no están incluidos en la imagen. En el caso de que se quisieran incluir y además poder aprovechar DockerHub como repositorio de la imagen, deberían ser descargadas por ejemplo de un servidor *ftp* externo. En el fichero Dockerfile se incluiría el comando para descargar estos ficheros y ya estarían disponibles para su uso. Sin embargo, se ha optado por incluir estos cuadernos junto a la documentación en un repositorio de GitHub.

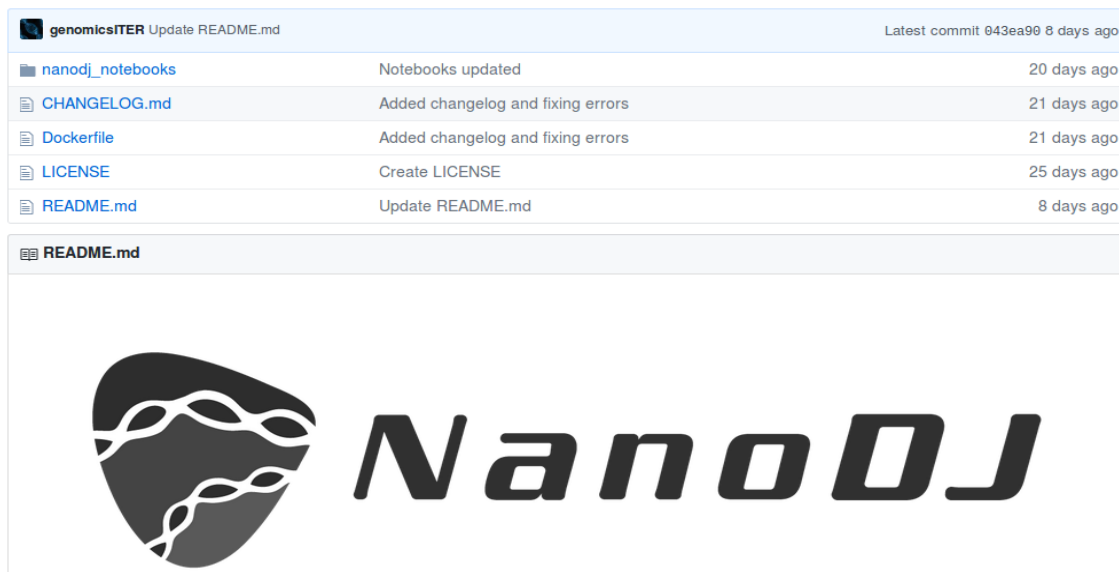


Figura 3.12. Vista general del repositorio de NanoDJ en GitHub

Esto soluciona el problema de que cualquier cambio que se quisiera realizar en alguno de los cuadernos ya incluidos no se podría guardar, dada la naturaleza volátil de los contenedores. La única forma de solucionar este problema de forma sencilla es incluir los cuadernos a modo de volumen cada vez que se lance el contenedor.

La creación de volúmenes es el mecanismo que implementa Docker para hacer posible el uso e intercambio de ficheros entre la máquina que hace de *host* y el propio contenedor. Montando como volumen el directorio donde están los cuadernos y datos de prueba, podemos acceder a este desde el contenedor, así como tener nuestros ficheros de salida también en el *host*.

Jupyter Lab funciona como una aplicación web que por defecto se publica en nuestra máquina, por lo que, si va a ejecutarse dentro de un contenedor aislado, en principio no va a poder accederse desde la máquina *host*. Docker permite mapear puertos del contenedor a puertos de la máquina que ejerce de *host*. Esto se realiza en el momento en el que lanzamos el contenedor, y nuestro caso, se hace un mapeado del puerto 8888 del contenedor al mismo puerto, pero de nuestra máquina. Como resultado, accediendo a <http://localhost/8888> en nuestra máquina, abrimos la interfaz de Jupyter Lab aunque no se esté ejecutando directamente en la máquina *host*.

La ejecución de aplicaciones con interfaz gráfica a través de un contenedor es posible en máquinas Linux. Para esto, se monta como volumen las X de nuestra máquina *host* (*/tmp/.X11-unix*) y colocamos el valor de la variable de entorno *DISPLAY* de nuestro *host* en el contenedor. Este proceso se hace en el comando *docker run* que despliega nuestro contenedor y puede hacerse independientemente de que tengamos montado también el directorio con los cuadernos y datos en otro volumen distinto.

3.16 Análisis de la solución terminada

La imagen de NanoDJ se distribuye en DockerHub y tiene un peso de 12 Gb. La imagen base desde la que se parte pesa unos 6 Gb por lo que las aplicaciones que se ha integrado tienen un peso de 6 Gb.

Por otro lado, los cuadernos y la documentación del proyecto están publicados en GitHub. Se incluye un directorio con los cuadernos y algunos datos de prueba, así como un fichero con la licencia (GNU v3.0 en este caso), un *CHANGELOG* y el fichero *README.md* con la documentación e instrucciones necesarias para la puesta en funcionamiento de **NanoDJ**.

github.com/genomicsITER/NanoDJ

La solución se inicia en segundos una vez se cuenta con la imagen descargada. Es posible importar y exportar datos de entrada y salida, así como modificar los cuadernos propuestos o crear otros nuevos. Una vez se termina la sesión de trabajo, se para y elimina el contenedor activo. NanoDJ posibilita el despliegue del entorno de trabajo completo de forma sencilla y cumple con el objetivo de facilitar la realización de análisis y estudios utilizando las últimas herramientas disponibles, sobre todo para alumnos y profesionales sin la base de informática necesaria para recrear un entorno similar.

Capítulo 4.

Resultados

4.1 Resultados numéricos

Comparación de ensamblados *de novo* de la muestra de *E. coli* K12-MG1655 utilizando Canu y la herramienta de pulido Racon. Se compara el ensamblado a partir de los FASTA disponibles y a partir de la extracción con Albacore de los FAST5.

Parámetros	FASTA	FAST5	FAST5-Racon
Longitud ensamblada (bp)	4,602,643	4,662,047	4,706,877
Contigs (≥ 500 bp)	1	1	1
Fracción del genoma (%)	99,88	99,98	9,99
Mismatches	14.674	8.446	10.533
Inserciones/deleciones	55.889	24.187	21.805
N50	4.602.643	4,662,047	4.706.877
%GC	51,07	50,95	50,75
Tiempo empleado (sec)	138.221,92	15.553,49	15.646,13

Tabla 4.1. Resumen del ensamblado de las lecturas de *E. coli* utilizando el cuaderno de Canu + Racon + Pilon

Parámetros	Canu	Flye	Miniasm
Longitud ensamblada (bp)	4,602,643	4,678,264	4.404.394
Contigs (≥ 500 bp)	1	1	1
Alineamiento más largo (bp)	3.336.128	2.310.685	77
Mismatches	14.674	12.678	0
Inserciones/deleciones	55.889	40.090	2
N50	4.602.643	4.678.264	4.404.394
%GC	51,07	50,72	52,48
Tiempo empleado (sec)	138.221,92	1.038	104

Tabla 4.2. Comparación de tres ensambladores de novo para los datos en FASTA de *E. coli* K-12-MG1655

Parámetros	Unicycler	MaSuRCA
Longitud ensamblada (bp)	2.159.288	2.099.176
Contigs (≥ 500 bp)	8	16
Contig más largo (bp)	1.019.216	627.716
Fracción del genoma (%)	98,57	95,29
Alineamiento más largo (bp)	1.018.789	627.116
<i>Mismatches</i>	165	230
Inserciones/deleciones	38	53
N50	714.293	434.380
%GC	35,39	35,47
Tiempo empleado (sec)	9.836,52	3.009,03

Tabla 4.3. Comparación de resultados de los ensamblados híbridos de los datos de *S. Agalactiae*.

Especie	Lecturas simuladas	Lecturas asignadas	Proporción
<i>Gallus gallus</i>	30	27	14,5
<i>Alligator mississippiensis</i>	30	25	13,4
<i>Bos taurus</i>	30	27	14,5
<i>Equus caballus</i>	30	29	15,6
<i>Oreochromis niloticus</i>	30	30	16,1
<i>Rattus norvegicus</i>	30	24	12,9
<i>Ovis aries</i>	30	24	12,9

Tabla 4.4. Resultados de la simulación y clasificación de las lecturas de ADN mitocondrial en el ejercicio de metagenómica.

4.2 Resultados gráficos

Visualizaciones de datos pertenecientes a los ficheros de lecturas obtenidos a partir de una muestra de la bacteria *S.agalactiae*

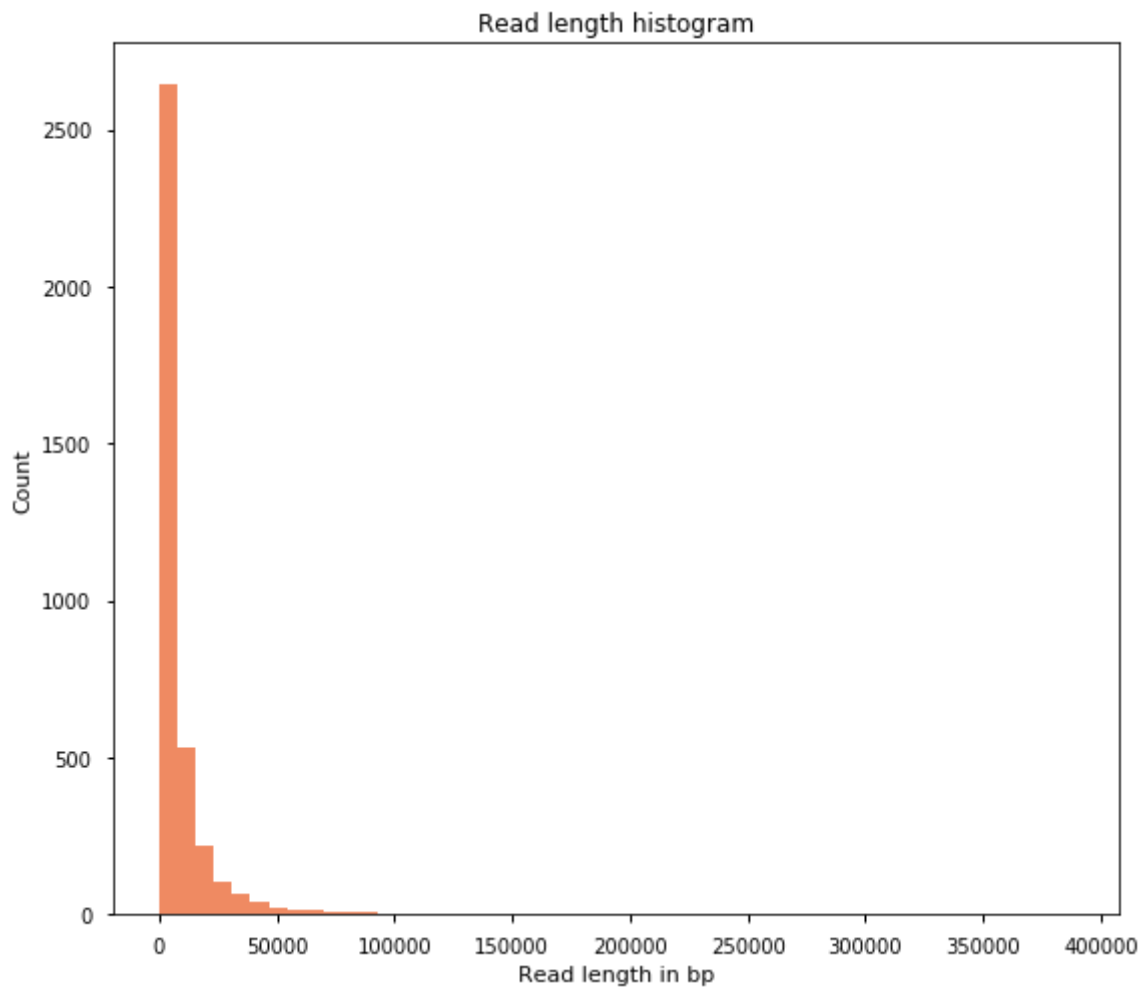


Figura 4.1. Distribución de longitud de lecturas

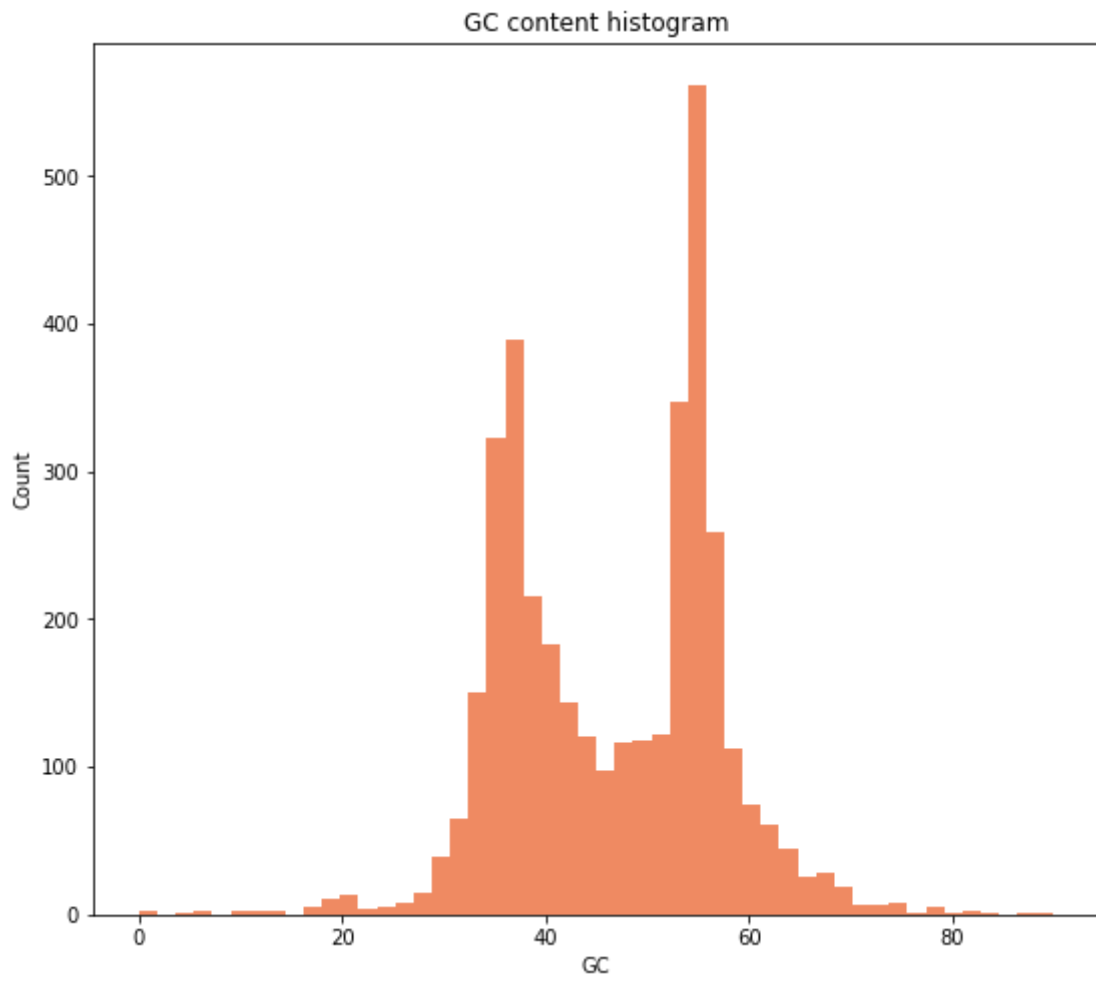


Figura 4.2. Distribución del porcentaje de contenido guanina-citosina de las lecturas

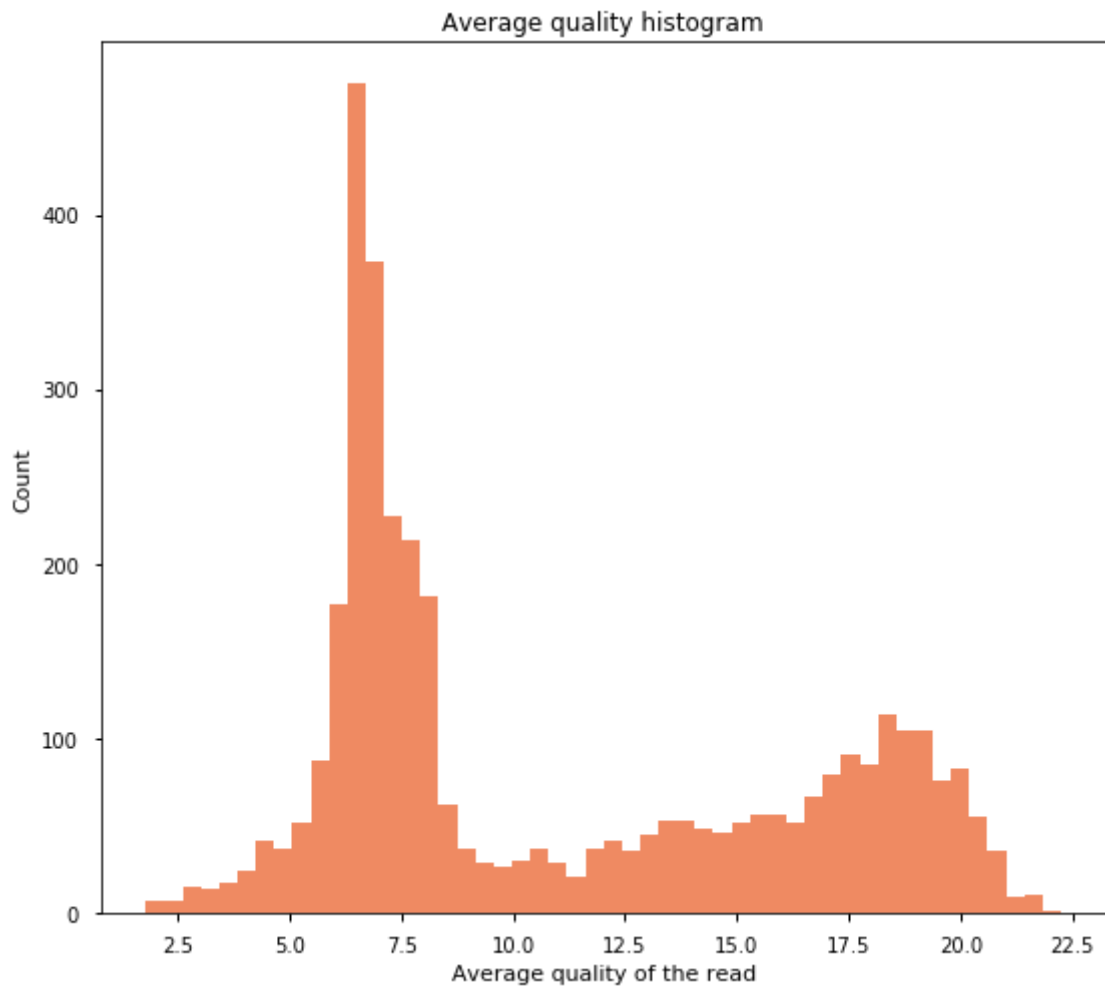


Figura 4.3. Distribución de las puntuaciones de calidad de las lecturas

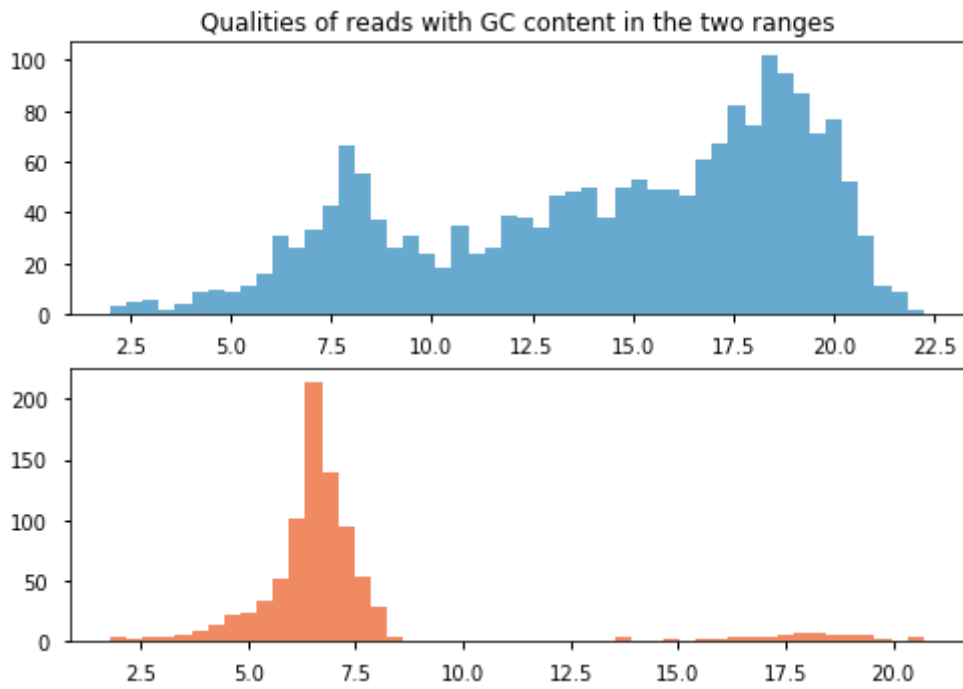


Figura 4.4. Comparación entre las calidades de dos subconjuntos separados en dos rangos de contenido guanina-citosina

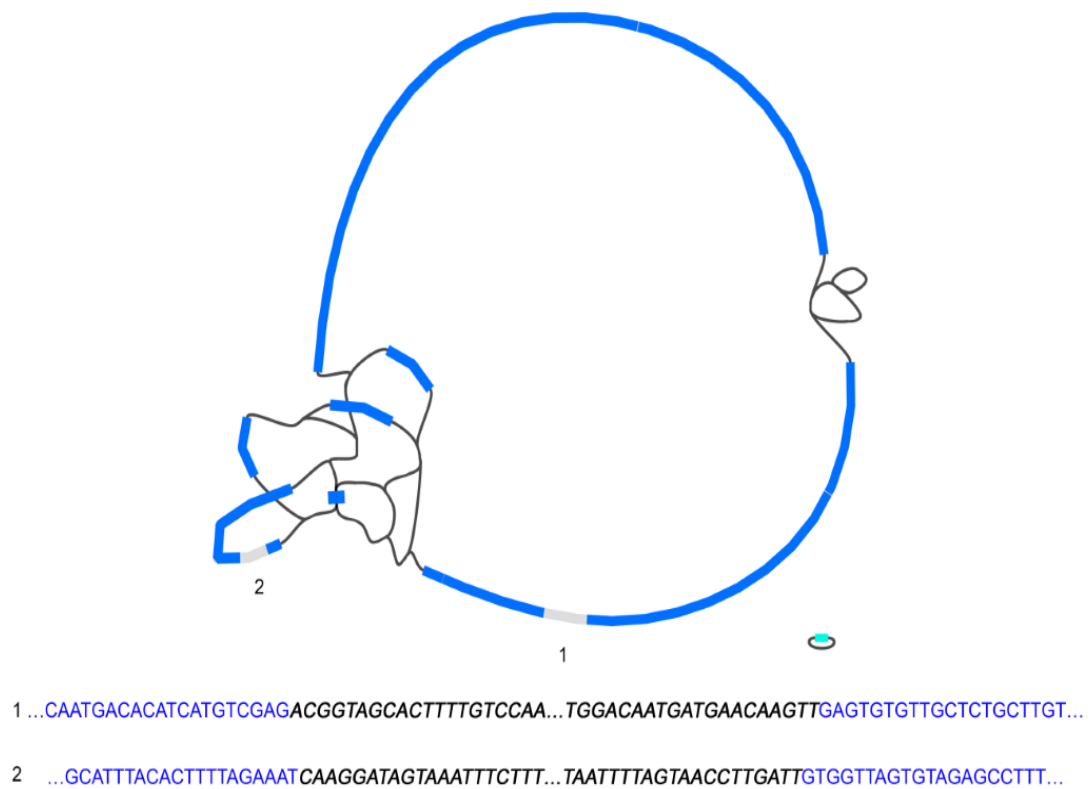


Figura 4.5. Visualización con Bandage del ensamblado híbrido con Unicycler

Visualizaciones incluidas en los cuadernos de Porechop y metagenómica:

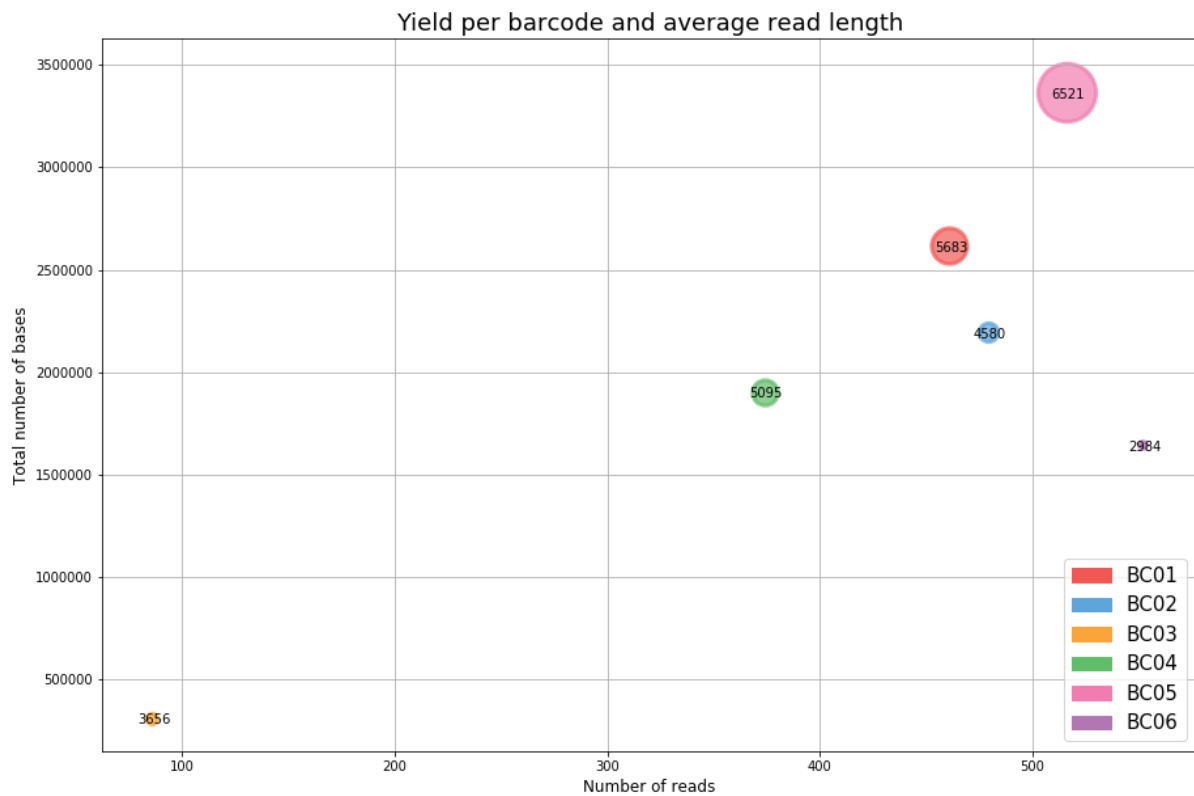


Figura 4.6. Visualización de la longitud y número de bases en las distintas muestras demultiplexadas con Porechop



Figura 4.7. Proporción de lecturas clasificadas por BLAST para cada especie (no se incluyen la proporción de las no identificadas con ninguna referencia)

Capítulo 5.

Presupuesto

Para elaborar el presupuesto tendremos en cuenta tanto recursos humanos (ingeniero/a informático/a y biólogo/a con especialidad en genómica) como el material necesario para llevar a cabo la secuenciación y posterior análisis. Se incluyen también horas de documentación y aprendizaje ya que la extracción de resultados numéricos o gráficos conlleva el conocimiento de los procesos y herramientas necesarias para tal fin, aparte de las horas dedicadas a la secuenciación y el análisis.

5.1 Recursos humanos

5.1.1 Ingeniero/a informático/a

Tarea	Horas	Precio	Total
Fase de documentación	80	15 €/h	1.200 €
Implementación y resultados	40	15 €/h	600 €
Redacción de la memoria	20	8 €/h	160 €

Tabla 5.1. Resumen del presupuesto para el Ingeniero/a Informático/a

5.1.2 Biólogo/a

Tarea	Horas	Precio	Total
Fase de documentación	80	15 €/h	1.200 €
Preparación de muestras	10	15 €/h	150 €
Puesta en marcha del MinION	10	15 €/h	150 €

Tabla 5.2. Resumen del presupuesto para el/la Biólogo/a

5.2 Costes materiales

Material	Precio
Portátil Dell XPS 13	1.179 €
MinION (User Starter Pack)*	1.000 €
Material de laboratorio necesario (plásticos)	300 €

Tabla 5.3. Resumen del presupuesto para los materiales

El pack de inicio del MinION y su respectivo kit de secuenciación incluyen una *flowcell* que puede ser utilizada unas 3-4 veces si se lleva a cabo una limpieza del consumible. Para posteriores pruebas o experimentos se deben adquirir de nuevo kits de secuenciación y *flowcells* que no vienen incluidas en el pack inicial

5.3 Costes totales

Elemento	Coste
Tareas (Ingeniero/a Informático/a)	1.960 €
Tareas (Biólogo/a)	1.500 €
Material	2.479 €
Total:	5.939 €

Tabla 5.4. Resumen del presupuesto para los costes totales

Capítulo 6.

Resumen y Conclusiones

Realizar este trabajo ha supuesto continuar con la línea seguida en el Trabajo de Fin de Grado. A lo largo del tiempo entre ambos trabajos, se ha observado la evolución en el estado del arte tanto de las herramientas para trabajar con el MinION, así como de las tecnologías utilizadas para construir NanoDJ.

El trabajo dentro de este campo se realiza en un entorno multidisciplinar, donde el rol de el/la ingeniero/a informático/a es necesario y bastante valorado. Muchos retos de la genómica y la bioinformática necesitan del trabajo de profesionales de nuestro sector para realizar nuevos estudios y descubrimientos. Llevar a cabo trabajos de este tipo requiere a los/as biólogos/as contar con conocimientos de informática que a menudo no son cubiertos en su formación, y al/la ingeniero/a informático/a con conocimientos de genómica que son importantes para obtener resultados rigurosos y de calidad.

La aparición de tecnologías de secuenciación novedosas y prometedoras como el MinION, ha motivado desde su salida, al desarrollo de algoritmos y herramientas destinados a trabajar con esta tecnología en concreto. Convertir la secuenciación en un proceso asequible y con posibilidades de ser realizado fuera del laboratorio, permite imaginar un futuro donde las ventajas de la secuenciación del ADN mejorarán la calidad sistema sanitario.

El proyecto Jupyter continúa afianzándose cada vez en más ámbitos y con la salida de Jupyter Lab, se presenta como una manera sencilla y efectiva de integrar el entorno de trabajo con la documentación y presentación de resultados. Es sin duda, un proyecto a seguir durante los próximos años

Docker por su parte, es ya una pieza fundamental en muchos ámbitos del sector tecnológico. El despliegue de las infraestructuras enteras utilizando

contenedores y la posibilidad de empaquetar aplicaciones y entornos de forma sencilla, ha mejorado la reproducibilidad en el sector científico y viene a solucionar problemas derivados de la instalación de muchas herramientas o portabilidad de entornos de trabajo.

En resumen, seguir la línea del Trabajo de Fin de Grado ha sido una experiencia muy valiosa, ya que permite observar la evolución de las tecnologías de secuenciación y análisis de datos a lo largo del tiempo. La bioinformática es una de las múltiples salidas de nuestra profesión, y posiblemente es una de las que más futuro tiene a medio y largo plazo vista la evolución hasta ahora. Es un sector que requiere de un aprendizaje continuo y de una capacidad de adaptación constante. Estos requisitos pueden suponer a veces un sacrificio, pero tiene como recompensa conocer el funcionamiento de los mecanismos más complejos de la vida y ayudar a afrontar los retos del mundo actual.

Capítulo 7.

Summary and Conclusions

Carrying out this work has meant continuing with the line followed in the Final Degree Project. Throughout the time between both works, it has been observed the evolution in the state of the art of both the tools to work with the MinION, as well as the technologies used to build NanoDJ.

The work within this field is carried out in a multidisciplinary environment, where the role of the IT engineer is necessary and highly valued. Many challenges of genomics and bioinformatics require the work of professionals in our sector to carry out new studies and discoveries. Carrying out jobs of this type requires biologists to have computer skills that are often not covered in their training, and the computer engineer with knowledge of genomics that is important to obtain results rigorous and of quality.

The emergence of innovative and promising sequencing technologies such as the MinION, has motivated since its release, the development of algorithms and tools to work with this technology in particular. Converting sequencing into an affordable process with possibilities to be performed outside the laboratory, allows us to imagine a future where the advantages of DNA sequencing will improve the quality of the healthcare system.

The Jupyter project continues to consolidate itself in more and more areas and with the release of Jupyter Lab, it is presented as a simple and effective way to integrate the work environment with documentation and presentation of results. It is without a doubt, a project to follow during the next years

Docker, for his part, is already a fundamental part in many areas of the technology sector. The deployment of entire infrastructures using containers and the possibility of packaging applications and environments in a simple way, has improved the reproducibility in the scientific sector and comes to solve problems arising from the installation of many tools or portability of work environments.

In summary, following the line of the Final Degree Project has been a very valuable experience, since it allows observing the evolution of sequencing technologies and data analysis over time. Bioinformatics is one of the multiple outputs of our profession, and it is possibly one of the most promising medium and long-term perspectives of evolution up to now. It is a sector that requires continuous learning and a capacity for constant adaptation. These requirements can sometimes involve a sacrifice, but it has as a reward to know the functioning of the most complex mechanisms of life and help to face the challenges of the current world.

Bibliografía

- [1] MinION www.nanoporetech.com
- [2] The Jupyter Project www.jupyter.org
- [3] Docker <https://www.docker.com/>
- [4] Di Tommaso P, Palumbo E, Chatzou M, Prieto P, Heuer ML, Notredame C. (2015) The impact of Docker containers on the performance of genomic pipelines. PeerJ 3:e1273 doi.org/10.7717/peerj.1273
- [5] Herramienta bioinformática usando Jupyter para el secuenciador de ADN MinION riull.ull.es/xmlui/handle/915/3089?show=full
- [6] Datos accesibles de una muestra de E.coli lab.loman.net/2016/07/30/nanopore-r9-data-release
- [7] Lecturas Nanopore de humanos (23 cromosomas + mitochondrial) github.com/nanopore-wgs-consortium/NA12878
- [8] Referencias de genomas de animales hgdownload.soe.ucsc.edu/downloads.html
- [9] Albacore nanoporetech.com/about-us/news/new-basecaller-now-performs-raw-basecalling-improved-sequencing-accuracy
- [10] Porechop github.com/rrwick/Porechop
- [11] Matplotlib matplotlib.org/
- [12] BioPython biopython.org
- [13] Canu github.com/marbl/canu
- [14] Racon github.com/isovic/racon
- [15] Pilon github.com/broadinstitute/pilon
- [16] minimap github.com/lh3/minimap2
- [17] samtools samtools.sourceforge.net/
- [18] Nanopolish github.com/jts/nanopolish
- [19] BWA github.com/lh3/bwa

- [20] Flye github.com/fenderglass/Flye
- [21] Yu Lin, Jeffrey Yuan, Mikhail Kolmogorov, Max W Shen, Mark Chaisson and Pavel Pevzner, "Assembly of Long Error-Prone Reads Using de Bruijn Graphs", PNAS, 2016
- [22] Unicycler github.com/rrwick/Unicycler
- [23] MaSuRCA www.genome.umd.edu/masurca.html
- [24] QUAST bioinf.spbau.ru/quast
- [25] Nanosim github.com/bcgsc/NanoSim
- [26] Rebaler github.com/rrwick/Rebaler
- [27] Bandage rrwick.github.io/Bandage/
- [28] BLAST www.ncbi.nlm.nih.gov/BLAST/
- [29] Docker Stacks github.com/jupyter/docker-stacks