UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO

Maestría y Doctorado en Ciencias Bioquímicas

Uso de tecnologías para análisis bioinformático de datos de secuenciación masiva

Tesis

QUE PARA OPTAR EL GRADO DE:
Maestro en Ciencias

PRESENTA:
Fernando Riveros Mckay Aguilera

TUTOR PRINCIPAL
Dr. Juan Enrique Morett Sanchez
Instituto de Biotecnología, UNAM

MIEMBROS DEL COMITÉ TUTOR:
Dr. Fidel Alejandro Sanchez Flores
Instituto de Biotecnología, UNAM
Dr. Víctor Manuel González Zúñiga
Centro de Ciencias Genómicas, UNAM

Cuernavaca, Morelos, Julio 2014

Integrantes de jurado de examen

- **Presidente:** Dra. María Esperanza Martínez Romero *Centro de Ciencias Genómicas, UNAM*
- **Secretario:** Dr. Alejandro Garcíarrubio Granados *Instituto de Biotecnología, UNAM*
- **Vocal:** Dr. Osbaldo Resendis Antonio *Instituto Nacional de Medicina Genómica*
- **Vocal:** Dr. Ernesto Pérez Rueda *Instituto de Biotecnología, UNAM*
- **Vocal:** Dr. José Adelfo Escalante Lozada *Instituto de Biotecnología, UNAM*

Agradecimientos

Creo que a las primeras personas que tengo que agradacer es a mis padres y hermanos por todo el apoyo que me han dado en mi formación personal y académica. Definitivamente todos mis logros son fruto de esta formación.
También muchas gracias a mi motor: Jhoanna.
Gracias a toda mi familia y en especial a mis abuelos que han sido excelentes conmigo.
A Xime, Leo y Martin por apoyarme tanto y soportarme tanto en mi búsqueda de PhD. Soy muy afortunado de haberlos tenido como amigos.
Al Dr. Enrique Morett por ser el mejor tutor que alguien como yo podría tener.
Perro, Mayar, Superman, Chiapas, Daniela, Mitz, Jaz y demás personas que me caen bien de Cuerna que hicieron mi estancia un poco más amena.
Winter Genomics que ha sido clave en mi formación académica, muchas gracias. Definitivamente es una empresa que debería ser grande. (Diana gracias por las correcciones extras)
A los miembros de mi comite tutoral por guiarme estos dos años y darle forma a este trabajo.
A los miembros del jurado por sus correcciones y comentarios para hacer de esta tesis algo de lo que me pueda sentir orgulloso.
A Jerome por ser tan excelente administrador del cluster del IBT. Increíble recurso humano.
A Toño y Gloria en docencia que han sido súper buenos conmigo.
Si consideras que debiste estar aquí y no lo estás escríbeme correo para incluirte en la próxima.
Y ya... si estás leyendo esta tesis muchas gracias a tí por leerla.

Resumen

Los avances en tecnologías de secuenciación han beneficiado enormemente el campo de la biología e impulsado nuevos enfoques de estudio como son las diversas "ómicas". Una de las aplicaciones más importantes de la secuenciación es el conocer el genoma completo de un organismo. El conocer el genoma completo de los organismos ayuda a responder varias preguntas sobre evolución, funcionalidad, predisposición a ciertas enfermedades o enfermedades genéticas entre otras cosas.

Ya que ninguna tecnología de secuenciación hasta la fecha permite obtener la secuencia completa de un organismo, se han desarrollado distintas metodologías de ensamblado *de novo* para armar el rompecabezas del genoma. Cada proyecto de ensamblado *de novo* se puede separar en tres etapas: preprocesamiento de lecturas, ensamblado y postprocesamiento de ensamble. Existen algoritmos para cada una de estas etapas que intentan optimizar los resultados que se pueden producir en uno de estos proyectos.

El trabajo actual analiza distintos algoritmos utilizados en las etapas de pre y post procesamiento para evaluar su efecto sobre los resultados de ensambles *de novo* en genomas pequeños (menores a 10 Mb). Para este fin se simularon computacionalmente cuatro genomas de distintas complejidades genómicas y también se utilizaron cuatro genomas de datos reales del aparato de secuenciación Illumina GAIIX. El análisis en datos simulados y reales permitió observar diferencias significativas en distintas métricas (NG50, número de contigs, tasa de error) que se utilizan para evaluar ensambles *de novo* lo cual llevó a la conclusión que distintas metodologías como la corrección de lecturas por algoritmos como ECHO o la eliminación de redundancia con el algoritmo Khmer pueden optimizar distintas métricas sin que haya una sola metodología que optimice todas. Esto lleva a que se tiene que hacer un análisis costo-beneficio antes de cualquier proyecto de ensamble para decidir qué es lo más importante siendo consciente de que se pueden sacrificar ciertas métricas para optimizar otras.

Índice

Índice de Figuras

Índice de Tablas

1 Introducción

1.1 Las ”ómicas”

Los avances tecnológicos de nuestra era que nos permiten generar y analizar datos a gran escala han tenido un impacto gigante sobre nuestra forma de hacer ciencia en los últimos años[1]. Lo que antes eran diferentes disciplinas han tenido que convergir para manejar la cantidad de información que se ha estado generando haciendo la línea que separa a los biólogos, químicos, matemáticos e informáticos a veces un poco difusa. No sólo tenemos mayor capacidad de producción y procesamiento de información que antes, también ha crecido de forma casi exponencial la capacidad de compartir esta información. Las consecuencias de esto se pueden ver observando el número de publicaciones por año en las últimas décadas, como se muestra en la figura 1.

Figura 1: Publicaciones indexadas en MEDLINE por año. Adaptado de [2]. El eje X muestra los años mientras que el eje Y muestra el número de publicaciones indexadas.

La biología se ha beneficiado considerablemente de los avances tecnológicos,

cambiando nuestra forma de analizar datos con la llegada de las "ómicas" donde analizamos los diferentes componentes de la biología molecular a un nivel más global. El sufijo de "ómicas" en el contexto de biología molecular se puede remontar a la aparición del término "genómicas" que se dio en una conferencia internacional en Bethesda discutiendo la viabilidad del mapeo del genoma humano completo. Durante una reunión pequeña en esta conferencia con el motivo de discutir el nombre de una nueva revista, el Dr. Thomas H. Roderick propuso la palabra "genómicas"[3]. Posteriormente surgió el término "proteómica" para describir el "complemento proteíco al genoma" por parte de Mark Wilkins en 1995 [4]. En los últimos años derivado de esto han surgido varias otras "ómicas" como metabolómica, transcriptómica, interactómica y lipidómica por mencionar algunas, sin embargo todas comparten la esencia de las originales que es el análisis a gran escala de datos biológicos. El manejo de tanta información sólo es posible gracias al uso de computadoras y bases de datos que nos permiten organizar de forma sistemática los datos biológicos. La bioinformática entonces es una disciplina que busca fusionar los conocimientos biológicos con los computacionales con este fin de análisis masivo de información.

1.2 ¿Por qué secuenciar un organismo?

La estructura del ADN, la molécula clave de la herencia biológica en los seres vivos, fue determinada en 1953 por los científicos James Watson y Francis Crick como una molécula de doble-hélice en la revista *Nature* [5]. El descubrimiento de esta estructura fue un hito en la biología ya que esta molécula es posiblemente una de las piezas más importantes en la evolución, pero fue hasta principios de los 70s donde empezaron a surgir metodologías para secuenciar fragmentos de ADN [6, 7]. El interés principal cuando surgieron estas metodologías era en genes codificantes, es decir, que producían proteínas. Se pensaba que secuenciando el genoma completo y por ende todos los genes de un organismo, se podía entender su biología. En el caso de algunas especies bacterianas, la secuenciación de su genoma completo ayudaría a entender su patogenicidad. Este interés fue dejado muy en claro en los primeros esfuerzos de secuenciación de genoma completo [8, 9]. El secuenciar

genomas completos bacterianos también permitió entender mejor el impacto de ciertos procesos biológicos en la evolución, como es el caso de la transferencia horizontal entre genomas bacterianos [10] o la organelización de mitocondrias [11]. En el caso del proyecto del genoma humano, el cual se concibió en 1984 y fue lanzado en 1990, se tenía como principal motivación para secuenciar el genoma completo del humano el entender y eventualmente tratar más de 4000 enfermedades genéticas que afectan a la humanidad así como enfermedades multifactoriales en las cuales la predisposición genética juega un papel muy importante [12]. Este gran interés por descifrar el rompecabezas que es el genoma del ser humano tuvo un momento clave en el 2001 con la secuenciación del primer draft del genoma humano [13, 14].

1.3 Tecnologías de secuenciación

Como se mencionó previamente, las tecnologías de secuenciación se han venido desarrollando desde inicios de los 70s con el afán de conocer la información genética de los distintos organismos que conforman la vida de este planeta [6, 7]. Es en 1977 cuando Frederick Sanger desarrolló el método de secuenciación de ADN que predominaría por las siguientes dos décadas denominado "DNA sequencing with chain-terminating inhibitors" [15]. Con esta tecnología se podían generar fragmentos (o lecturas) de aproximadamente 700 pares de bases, sin embargo este tamaño no es suficiente para secuenciar un genoma completo por lo tanto se fueron desarrollando formas de subdividir los genomas en fragmentos pequeños para luego ser ensamblados. Dos metodologías principales fueron usadas para estos fines: "chromosome walking" [16] y "shotgun sequencing"[17]. En el caso del genoma humano, fue la segunda técnica la que fue usada debido a la practicidad sobre secuencias más grandes de esta técnica. La técnica en general se basa en fragmentar ADN genómico completo, clonar los distintos fragmentos en un vector apropiado, secuenciar los fragmentos en estas clonas y por último ensamblar utilizando algoritmos computacionales los cuales serán descritos posteriormente. Un complemento a esta metodología era la secuenciación de lecturas pareadas donde se leen ambos extremos de un fragmento grande del cual se conoce el tamaño. De esa forma,

aunque no se cuenta con la secuencia intermedia, se puede conocer la distancia promedio de dos lecturas en el genoma lo cual ayuda cuando se quiere lidiar con secuencias repetitivas [18]. Dos variantes de la técnica de "shotgun sequencing" fueron utilizadas por los dos grupos distintos que secuenciaron el genoma humano. El consorcio del proyecto del genoma humano [13] utilizó la técnica de "hierarchical shotgun sequencing" en la cual después de fragmentar el ADN genómico con enzimas de restricción, estos fragmentos son clonados en BACs que permiten grande tamaños de inserto. Con estas BACs después se hace un mapa físico del cromosoma para saber el orden de estos fragmentos grandes. Estos fragmentos posteriormente son fragmentados nuevamente en pedazos más pequeños los cuales son secuenciados y ensamblados. Ya que se tienen los ensambles de cada clona se unen con la información del mapa físico del cromosoma para generar el ensamble final. Este método tiene la ventaja de ser menos propenso a errores debido a que es menos probable que sea afectado por secuencias repetitivas ya que se va secuenciando con base en el mapa físico y entonces se conoce el orden *a priori*. Las principales limitantes de esta metodología son el tiempo y el costo.

El otro grupo que era la compañía Celera [14] utilizó la técnica de "whole genome shotgun sequencing (WGS)" para secuenciar el genoma humano. En esta técnica no se usan BACs ya que directamente se fragmenta el ADN completo en fragmentos pequeños los cuales son clonados en plásmidos y después son secuenciados y ensamblados. Esta técnica puede llegar a ser más propensa a errores pero resulta ser más rápida y barata que el otro enfoque [19].

Al final ambos genomas fueron publicados al mismo tiempo pero esto fue resultado de mucho tiempo de trabajo y de sobre todo una gran inversión financiera ya que los costos de este proyecto de secuenciación se elevaron a los 100 millones de dólares [20]. Parte de los objetivos iniciales del proyecto del genoma humano, era la reducción en costos de secuenciación, bajando de los $2 a $5 dolares (rango de precios inicial) a $0.50 [12]. La demanda por reducción de costos y tiempos tuvo como resultado el surgimiento de nuevas tecnologías de secuenciación denominadas "next-generation sequencing platforms" utilizando distintas metodologías como pirosecuenciación, secuenciación por síntesis y secuenciación por ligación

para abordar esta problemática [21, 22, 23]. Estas nuevas tecnologías además de reducir costos y tiempos aumentaron el rendimiento produciendo millones de secuencias en una sola corrida lo cual sería crucial en estas nuevas tecnologías para compensar su principal limitante que era el tamaño de las lecturas ya que en su mayoría eran lecturas muy pequeñas (siendo las más pequeñas de 35 pares de bases y las más grandes de 108 pares de bases dependiendo de la tecnología empleada). Estas constricciones de tamaño de lectura que inicialmente se tenían, gradualmente han sido superadas y pueden llegar ya hasta 30,000 pares de bases con el secuenciador de tercera generación SMRT de Pacific Biosciences [24]. Otras características importantes de estas nuevas tecnologías de secuenciación es la omisión del paso de clonación ya que cada una estas metodologías utiliza técnicas para amplificación distintas como PCR de emulsión [25] o PCR en puente [26]. En el caso las tecnologías de tercera generación, el paso de amplificación por PCR es completamente desechado [27]. Como se puede observar en la figura 2, las tecnologías de secuenciación masiva han disminuido significativamente el costo de secuenciación de ADN y ARN aumentando al mismo tiempo la cantidad de información obtenida. Gracias a esto se han podido hacer estudios a gran escala en estos campos como el proyecto de los 1000 genomas o la detección de errores de transcripción *in vivo* [28, 29].

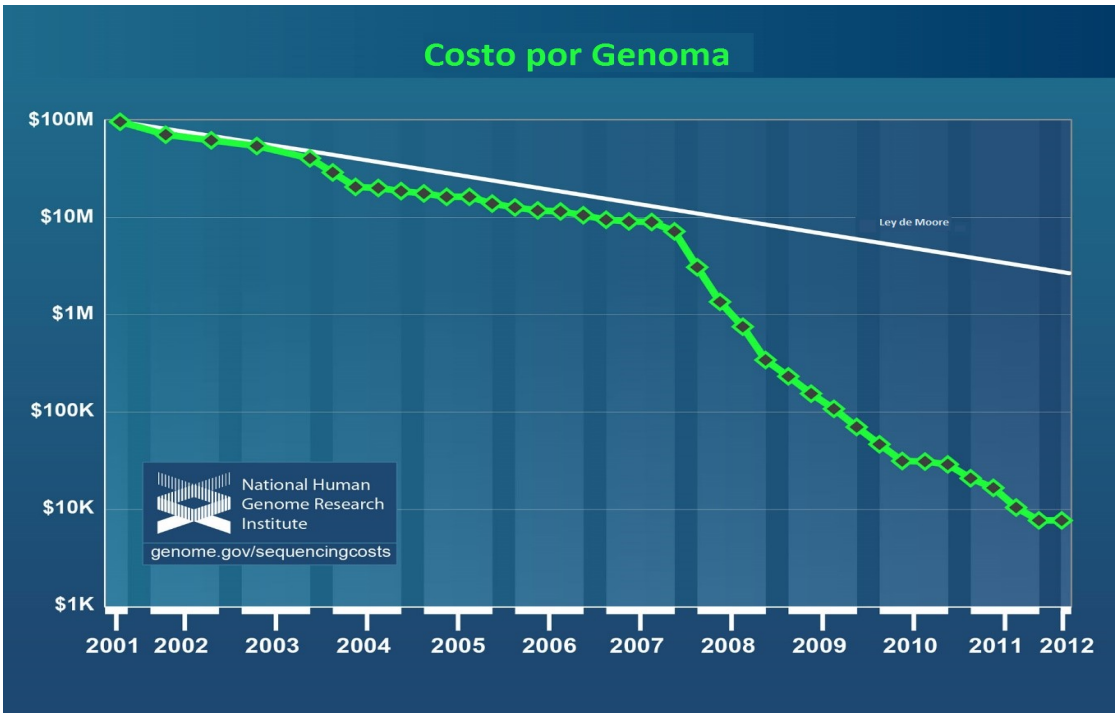

Figura 2: Costo por secuenciación completa de genoma humano. Adaptado de [20]. En el eje X se grafican los años mientras que en el eje Y se grafica el precio por genoma humano. Para mostrar el éxito de las tecnologías de secuenciación, se muestra una línea blanca que grafica datos hipotéticos reflejando la Ley de Moore como punto de comparación. Esta ley describe una tendencia en la industria del hardware computacional que involucra la duplicación del poder computacional cada dos años. Una tecnología es considerada exitosa cuando se mantiene a la par de la Ley de Moore.

1.4 Algoritmos de ensamblado *de novo*

Así como las tecnologías de secuenciación han ido evolucionando, también las metodologías para ensamblado *de novo*. Como se mencionó previamente, hasta la fecha ninguna tecnología de secuenciación produce lecturas suficientemente grandes para secuenciar un organismo completo por lo tanto se necesitan de algoritmos computacionales para armar el rompecabezas que es el genoma a partir de los fragmentos pequeños. Al principio los algoritmos más comunes para ensamblado de lecturas eran basados en el enfoque de ”overlap-layout-consensus” [30]. En este enfoque se le proveen todas las lecturas al algoritmo el cual luego calcula los sobrelapes entre bordes de las lecturas. Después cada lectura es graficada como un nodo y cada sobrelape como una arista uniendo dos nodos que presentan este so-

brelape. El algoritmo después determina el mejor camino que recorre cada nodo exactamente una vez (camino Hamiltoniano). La información redundante (vértices y aristas no usados) es después desechada. El proceso es repetido muchas veces y después las secuencias resultantes son combinadas para sacar la secuencia consenso. Algoritmos como Atlas, ARACHNE, Celera, PCAP y Phusion se basan en este método [31, 32, 33, 34, 35] La metodología es ilustrada en la figura 3.

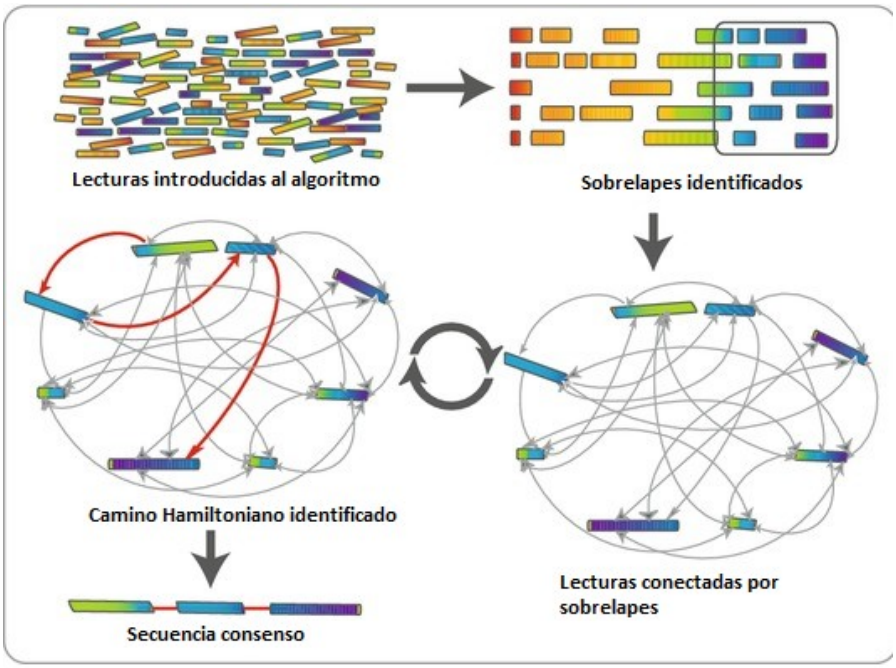

Figura 3: Algoritmo overlap-layout-consensus. Adaptado de [36]. En el primer paso, las lecturas obtenidas del secuenciador son introducidas al algoritmo. Después se procesan sobrelapes entre los bordes de estas lecturas para después hacer un grafo conectando estas lecturas por medio de estos sobrelapes. Se identifica un camino Hamiltoniano y se repite el proceso de construcción de grafo y resolución de camino Hamiltoniano. Al final se llega a una secuencia consenso basada en los caminos Hamiltonianos encontrados.

Este método aunque ideal para lecturas Sanger, con la llegada de las tecnologías de secuenciación nuevas resultó muy costoso computacionalmente para los millones de sobrelapes que necesitaban ser procesados de las lecturas producidas por estas tecnologías.

El enfoque más predominante en los ensambladores de nueva generación fue el basado en grafos *de Bruijn*. Bajo este método, todas las lecturas son divididas en palabras de tamaño k (*k-meros*) y se crean grafos a partir de estos *k-meros*

donde cada arista representa una de estas secuencias y los nodos representan sobrelapes con otros *k-meros* (aunque en algunas implementaciones los nodos son los *k-meros* y las aristas los sobrelapes [37]). El primer ensamblador en implementar este enfoque fue EULER [38]. Esto reduce el número de nodos considerablemente ya que en lugar de volverse el grafo centrado en lecturas se vuelve centrado en *k-meros*. De las desventajas que pueden venir con este enfoque es que la resolución de secuencias repetidas puede llegar a ser más compleja ya que información de conectividad puede ser perdida al separar las lecturas en *k-meros*, especialmente en lecturas más grandes. Otra desventaja es que la construcción del grafo al separar las lecturas en *k-meros* puede requerir bastante memoria RAM aunque algunos algoritmos como ABySS [39] y Ray [40] lidian con esta limitante al paralelizar la construcción del grafo. La resolución de la secuencia vuelve a ser un problema de encontrar el mejor camino que recorra todos los nodos (o las aristas dependiendo de la implementación) exactamente una vez. La metodología es ilustrada en la figura 4. Con el aumento del tamaño de lecturas y mejoras en algoritmos computacionales se han vuelto a usar algoritmos basados en overlaps para tecnologías de nueva generación como por ejemplo SGA [41] que reduce el tiempo computacional requerido para procesar los sobrelapes representando las lecturas como un arreglo de sufijos y encontrando los sobrelapes usando el "FerraginaManzini index" (FMindex).

Algo clave sin importar el algoritmo utilizado, es que los mayores tamaños de lectura siempre producen mejores ensambles, por lo cual son muy importantes los avances en tamaños de lectura en la secuenciación masiva [42].

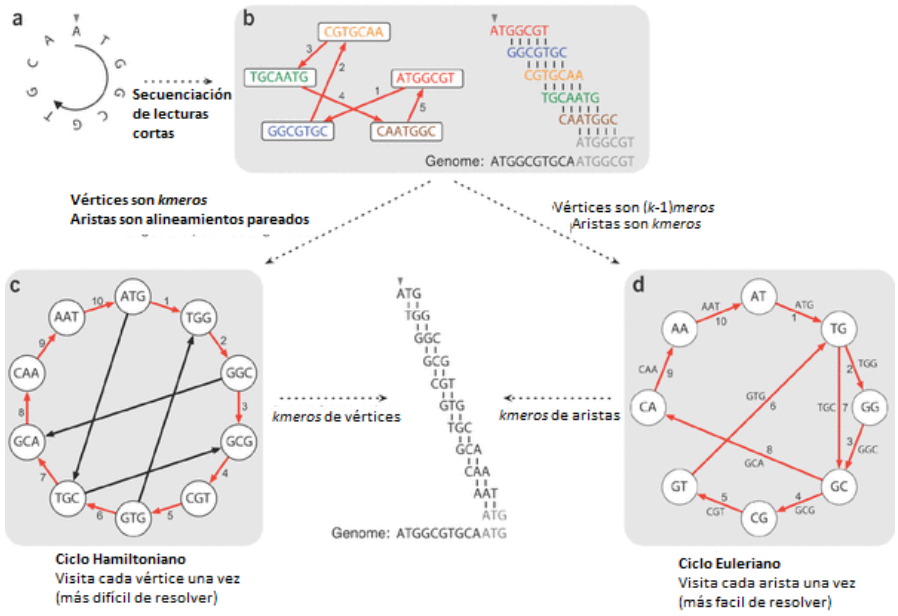

Figura 4: Algoritmos basados en grafos *de Bruijn*. Adaptado de [43]. a) Muestra un ejemplo de un genoma circular. b) Muestra el acercamiento utilizado anteriormente para lecturas de tecnologías Sanger basado en sobrelapes. c) y d) Muestran alternativas donde se parten primero las lecturas en secuencias de tamaño k (*k-meros*). En c) los *k-meros* son los vértices mientras que las aristas son alineaminetos pareados. La resolución del genoma es por medio de un ciclo Hamiltoniano. En d) Los aristas son *k-meros* y lo vértices son (k-1)meros. La resolución del genoma esp or medio de un ciclo Euleriano. La mayoría de los algoritmos modernos basados en grados de Bruijn se basan en d) por la mayor facilidad de resolución.

1.5 Tecnología Illumina

Los aparatos de secuenciación masiva de Illumina son los más populares por el momento en el mundo [44]. Esta popularidad se debe en gran parte a la calidad de los datos y costo accesible de sus aparatos de secuenciación. Aunque actualmente existen distintos aparatos de secuenciación de Illumina como el MiSeq, el HiSeq y el Illumina Genome Analyzer(GA) IIx, los cuales varían en tiempo, costo y número de secuencias producidas, todos comparten el principio de secuenciación por síntesis usando un terminador reversible [45]. Ya que este trabajo fue realizado con secuencias del Illumina GAIIx, se describirá el proceso bajo el cual se realizan lecturas pareadas con esta tecnología utilizando el kit Nextera [46] para preparación de la muestra:

1. Todo comienza por la preparación de la muestra, en este paso se extrae y

purifica el ADN.

2. Utilizando transposomas simultáneamente se fragmenta el ADN y se marca con adaptadores.

3. En un paso de amplificación se añaden otros marcadores como sitios de unión del "*primer*" de secuenciación, índices y regiones complementarios a los oligos en las celdas donde se realiza la secuenciación. Estas celdas son portabobjetos con carriles dentro de los cuales se encuentran fijados oligos de dos tipos.

4. La hibridación sucede con el primer tipo de oligos que es complementario al marcador que previamente se le añadió al fragmento de ADN a secuenciar.

5. Una polimerasa genera un complemento al fragmento hibridado. El ADN de doble cadena es desnaturalizado y el templado original es lavado.

6. Las hebras son clonalmente amplificadas por medio de PCR en puente mediante el doblado de la molécula de ADN e hibridación del segundo tipo de oligo al otro extremo de dicha molécula. Este proceso es repetido muchas veces en millones de clusters logrando la amplificación clonal de todos los fragmentos.

7. Las hebras en cadena complementaria son cortadas y lavadas. Los extremos 3' son bloqueados para prevenir la unión de "*primers*" no deseada.

8. Se comienza a secuenciar con la extensión del primer "*primer*" de secuenciación. Durante cada ciclo, 4 nucleótidos marcados fluorescentemente compiten para ser incorporados en la hebra de ADN naciente, siendo solo uno de ellos incorporados dependiendo de la base complementaria en la cadena de ADN original. Después de la adición de cada uno de estos nucleótidos, los clusters son excitados con luz produciendo la emisión de una señal fluorescente característica de cada nucleótido marcado.

9. Para cada cluster todas las lecturas idénticas son leídas simultáneamente y millones de estos clusters son leídos paralelamente.

10. Al completar la primera lectura, la lectura es lavada. El primer índice es leído, se desecha y se desprotege el extremo 3' lo cual permite el redoblado de la secuencia donde se vuelve a pegar el extremo 3' con el segundo tipo de oligo. Se lee el segundo índice, se desecha y después se crea un ADN de doble cadena. Se vuelve a proteger el extremo 3' y ahora se lava el ADN original dejando la cadena complementaria.

11. Se lee esta cadena de la misma forma que se hizo para la primera lectura.

12. Utilizando los índices al final se pueden unir parejas de lecturas.

Todos estos pasos pueden ser observados de una forma más didáctica en [47].

Un algoritmo después es utilizado para el llamado de bases leyendo los archivos de intensidades generando un archivo FASTQ [48], el cual incluye la secuencia de la lectura y un valor de calidad para cada ciclo de lectura [49].

1.6 Control de calidad en tecnologías de secuenciación

Una de las prácticas más comunes es la evaluación de calidad de la biblioteca secuenciada. Para esto las herramientas más utilizadas son FastQC [50] y la librería de ShortRead del paquete de Bioconductor [51] para R [52]. FastQC prácticamente genera todas las gráficas suficientes para una evaluación de calidad de los datos y de una forma más rápida. Todo esto en un formato muy amigable para usuarios poco familiarizados con la bioinformática (figura 5).

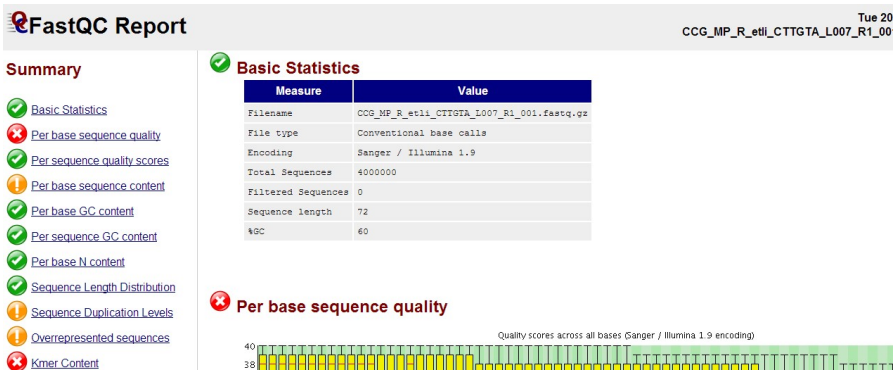

Figura 5: Output de FastQC. Muestra gráficas de evaluación de calidad y en cada sección te muestra con diferentes símbolos si se comporta de manera positiva, sospechosa o negativa.

ShortRead es preferible cuando se quieren personalizar un poco más las gráficas o se quiere obtener otro tipo de información con respecto a las calidades. También ShortRead es útil ya que el mismo programa puede servir para filtrado de lecturas en caso de requerirlo.

Las gráficas más informativas y para las cuales es más importante para el usuario común saber interpretarlas son las gráficas de secuencia por ciclo y calidad por ciclo. La gráfica de secuencia por ciclo muestra la frecuencia de cada una de las 4 bases en cada posición de las lecturas. Idealmente se espera que la frecuencia correlacione con el contenido de GC estimado del genoma estudiado. Aunque FastQC separa las gráficas de secuencia por ciclo y contenido de Ns (bases que el secuenciador no pudo determinar), podría ser más práctico saber cómo generar una sola gráfica que contenga toda esta información con ShortRead. Una gráfica normal esperada para secuenciación de un genoma completo con aproximadamente 40% GC se ve de la siguiente forma:

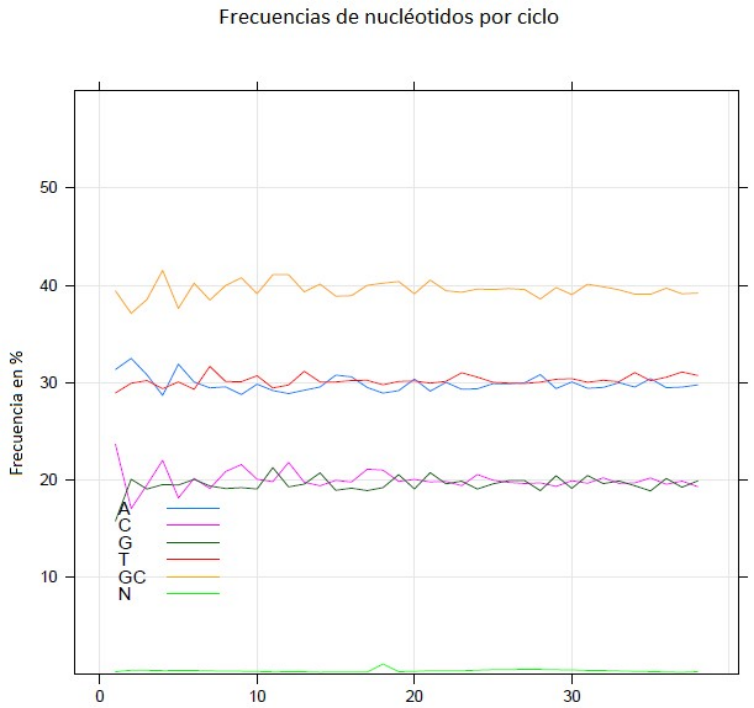

Figura 6: Gráfica de secuencia por ciclo esperada para un genoma con porcentaje de GC estimado del 40%. El eje X grafica la posición de la base en la lectura mientras que el eje Y muestra la frecuencia de cada base en porcentaje.

La presencia de una alta frecuencia de Ns en una gráfica indica problemas en la secuenciación, principalmente debido a fallas ópticas del aparato de secuenciación. Los sesgos hacia ciertas bases pueden ser o no considerados como aberrantes dependiendo del contexto de la secuenciación. En una secuenciación de genoma completo no se esperan sesgos ni en lecturas completas ni en segmentos de la lectura ya que en teoría se está cubriendo el genoma de forma azarosa y homogénea (aunque esto último frecuentemente no se cumple estrictamente [53]). Con abundante número de lecturas entonces se espera que la frecuencia sea casi homogénea. Un sesgo parcial de la lectura es normalmente asociado con la secuencia del adaptador utilizado para la construcción de las bibliotecas y que es ligado a los extremos de los fragmentos a secuenciar, lo cual puede hablar de una mala fragmentación del ADN, dimerización y amplificación accidental del dímero de adaptadores. Cuando la secuencia sobrerrepresentada no es adaptador entonces puede ser un problema inherente a la muestra. En algunos proyectos como análisis de microRNAs es esperado tener sesgos parciales en las lecturas ya que se están buscando específicamente ciertos fragmentos pequeños de ADN. La calidad por ciclo es un estimado de la probabilidad de que esté errónea una base en la lectura [54, 55]. El instrumento de secuenciación normalmente da un valor de calidad (valor Q) a cada base que lee. Este valor está en el intervalo de 0 a 40, donde los valores más bajos indican una probabilidad (p) más alta de ser errónea la base asignada. La relación está dada por la fórmula $Q = -10 \log 10p$ y se muestra en la figura 7.

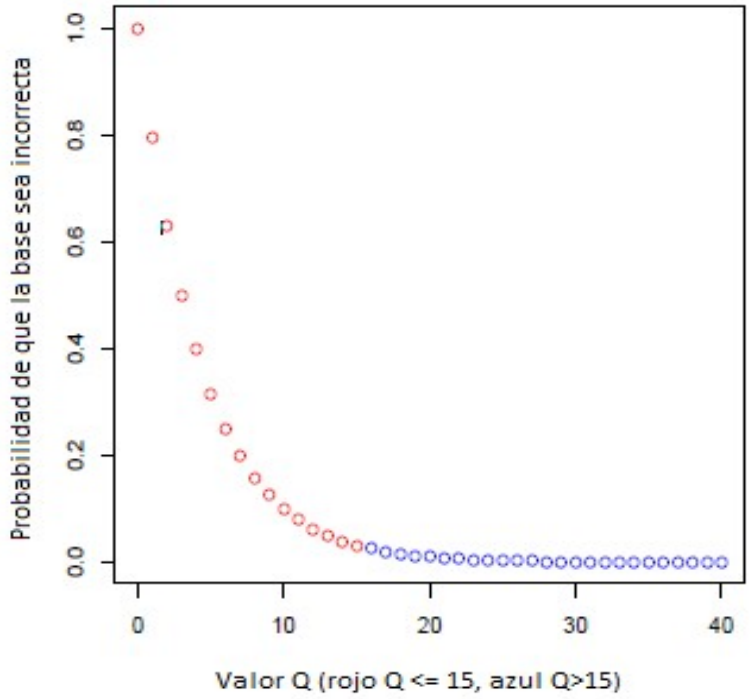

Figura 7: Relación del valor Q (eje X) con la probabilidad de que el llamado de base sea erróneo (eje Y).

Normalmente se espera que todas las bases tengan calidades mayores a 20 (probabilidad de error de 1 en cada 100) y óptimamente arriba de 30 (probabilidad de error de 1 en cada 1000). Normalmente en todas las tecnologías de nueva generación se espera un declive en la calidad hacia las últimas bases y muchas veces por eso se ”rasuran” las lecturas en el extremo 3’, lo cual puede hacer diferencias muy grandes en mapeos sin embargo en ensambles *de novo* en estudios recientes no se encontró gran aportación en precisión y se encontró una disminución en la continuidad del ensamble [56, 57]. FastQC presenta las calidades en *boxplots* las cuales son representaciones gráficas muy adecuadas para representar cuartiles en una muestra, como se puede ver en la figura 8.

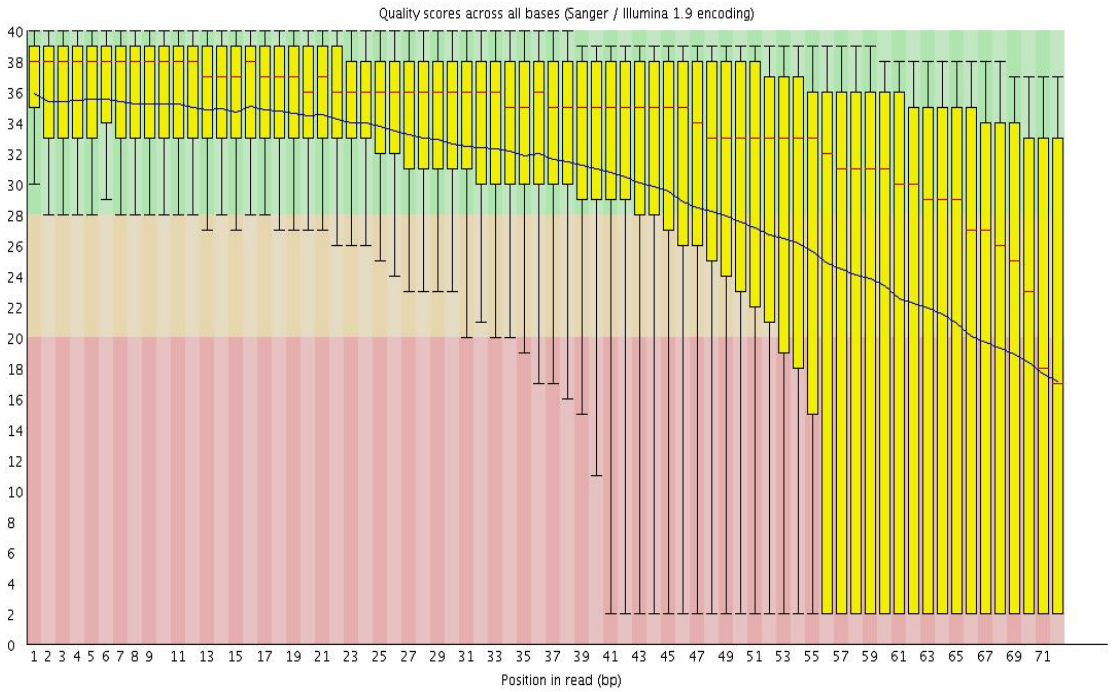

Figura 8: *Boxplots* de calidades por ciclo. El eje X grafica posición de la base en la lectura mientras que el eje Y grafica el valor de calidad. Las calidades en la zona roja son consideradas malas, las que están en la región amarilla regulares y las que están en la región verde buenas.

1.7 Prácticas comunes en preprocesamiento

Antes de la secuenciación masiva, era una práctica común filtrar secuencias de vector de las lecturas [58] debido a que el uso de vectores en la secuenciación por Sanger llegaba a producir contaminación de muestra. Ahora con la secuenciación masiva aunque se ahorra ese paso, existen otros filtros comunes que se utilizan. Reportes iniciales para ensamblado *de novo* recomendaban ciertos pasos de filtrado para producir mejores resultados [59]. Hasta la fecha, varios grupos de trabajo siguen utilizando algunos o todos de estos pasos de filtrado a la hora de ensamblar *de novo* un genoma, especialmente el rasurado de secuencias 3' [60, 61, 62, 63, 64, 65, 66]. En general los filtros principales en todo tipo de proyecto son: limpiado de secuencias adaptadoras, limpiado de secuencias de baja complejidad, eliminación de secuencias con bases ambiguas (Ns), filtro de secuencias con alta probabilidad de estar incorrectas y rasurado de extremos 3'. Por comodidad se puede hacer

un solo script de R usando el paquete de ShortRead, que puede hacer todos los diferentes tipos de filtrado, aunque existen algunos programas como los incluidos en el toolkit fastx [67] que pueden hacer todo más rápido aunque de una forma menos personalizable.

En el filtrado de adaptadores la función de trimLRPatterns de ShortRead y fastx_clipper de fastx son las más útiles ya que permiten un tipo de "ventana deslizante" del adaptador a lo largo de la lectura, lo que puede eliminar todos los casos donde haya secuencia de adaptador sin importar si su cobertura es total o parcial. La función de trimLRPatterns tiene una de las ventajas de que se le puede dar un vector con el número de discrepancias permitidas por "ventana" lo cual permite una más fina eliminación de adaptadores, aunque su desventaja con respecto al fastx_clipper es la velocidad. Para la eliminación de secuencias de baja complejidad la función polynFilter de ShortRead y fastx_artifacts_filter de fastx son las que se pueden usar. Fastx elimina secuencias donde se tiene la misma base en todas las posiciones de la lectura exceptuando 3 a lo máximo. El filtro polynFilter se le puede especificar cuantas veces tiene que estar la misma base en la lectura para eliminarla. Para el filtrado de bases ambiguas una muy buena opción es el nFilter de ShortRead ya que se le puede especificar el número máximo de Ns presentes en la lectura. En proyectos de ensamble *de novo* la tolerancia es baja y en general no se aceptan lecturas con Ns; en otros tipos de proyecto donde se alinea contra una referencia se pudiera ser más permisivo con el número de bases ambiguas toleradas. En ensambles *de novo* el fastx_clipper también elimina toda secuencia con Ns.

El filtro de calidad puede hacerse usando el fastq_quality_filter de fastx al cual se le puede dar un mínimo de calidad y qué porcentaje de bases en la lectura pueden tenerlo. Usando el paquete de ShortRead se pueden hacer filtros usando estadísticas más adecuadas para datos de secuenciación masiva como eliminar lecturas cuya mediana de calidad está debajo de un límite. Este paso si se usa normalmente es mejor utilizarlo después del rasurado ya que las últimas bases pueden afectar la mediana de una lectura que puede ser útil.

El rasurado de extremos 3' se puede hacer ya sea cortando todas las lecturas

en cierta posición basado en las gráficas de calidad por ciclo o usando algunos programas como el fastq_quality_trimmer que cortan a partir de cierto valor dado por el usuario. Los valores debajo de 20 son normalmente los puntos donde se empieza a cortar, ya sea evaluando lectura por lectura o evaluando en la gráfica de calidad por ciclo el punto donde la mediana es menor a 20.

Al final estos filtrados iniciales permiten tener un tamaño reducido de lecturas con el cual se puede empezar a trabajar. Este set además de abarcar menos espacio en disco duro puede también reducir tiempos en pasos posteriores de un proyecto bioinformático donde se requieran estas lecturas.

1.8 Estadísticos comunes en evaluación de ensamblado y su relevancia

Normalmente cuando se ensambla *de novo* un genoma, se reportan una serie de métricas de continuidad para evaluar lo bien ensamblado que se encuentra este genoma [60, 61, 62, 63, 64, 65, 66]. Estas métricas normalmente son:

- Número de contigs: Casi siempre cuando se da este valor, también se da el tamaño mínimo de contig utilizado como valor de corte ya que muchas veces se eliminan contigs demasiado pequeños que probablemente son errores de secuencia (por ejemplo contigs menores al tamaño de lectura inicial). Este número es clave ya que nos dice que tan fragmentado quedó el genoma.

- N50: Este es otro valor muy utilizado ya que habla también de la continuidad del ensamble. Este valor representa el tamaño de contig en el cual toda la colección de contigs de este tamaño o mayor abarcan la mitad o más del total de bases ensambladas. Valores grandes de N50 en teoría nos hablan de una buena continuidad en el genoma ensamblado y se busca que sea mayor al tamaño promedio de genes. Este valor es el más utilizado para comparar ensambles.

- Tamaño promedio de contig: Este valor indica cual es el tamaño promedio de los contigs lo cual es otro indicador menos pesado de continuidad y que

nos sigue dando una idea de la cantidad de información completa génica que se puede extraer de este ensamble.

- Contig más grande: Otro valor que es un indicador de continuidad y que se espera sea lo más cercano al total de bases ensambladas posible.

Así como estas métricas, cuando se cuenta con información de una referencia (especialmente cuando se prueba la eficiencia de algoritmos de ensamblado *de novo*) se pueden incluir métricas extras de precisión y continuidad. Las siguientes son proporcionadas por algoritmos como QUAST [68] para evaluar ensambles:

- Misassemblies: Son rearreglos cromosomales grandes, ya sea una relocación de más de 1Kb, una inversión o una traslocación.
- Local misassemblies: Son rearreglos crosomomales pequeños que están relacionados a secuencias repetidas de no mucha extensión.
- Mismatches: Discrepancias de una sola base con la referencia.
- Indels: Indeles pequeños con respecto a la referencia.
- Número de genes (en caso de contar con anotación): Esta métrica nos dice el número de genes completos y parciales que se encontraron

2 Antecedentes

2.1 Estudios previos de comparación de algoritmos

El análisis de la enorme cantidad de información que producen las tecnologías de secuenciación masiva necesita de habilidades computacionales así como conocimiento biológico suficiente para poder interpretar adecuadamente los resultados. Con una amplia variedad de distintos algoritmos para análisis de estos datos que a veces puede ser abrumadora, existen estudios que demuestran que el tiempo de obtención y la confianza de los resultados pueden variar considerablemente dependiendo de la metodología que se esté utilizando [56, 57, 69, 70, 71, 72]. Tres de estos

estudios son de especial interés para ensamblado *de novo*. El primero de ellos es el *Assemblathon 1* [70]. En este estudio se simuló un genoma diploide nuevo y de este se simularon lecturas de Illumina Hi-Seq pareadas con distinto tamaño de inserto. Estas lecturas fueron hechas públicas para que distintos laboratorios pudieran acceder a ellas y someter sus ensambles. Los grupos podían someter más de un ensamble. Al final 17 grupos distintos sometieron 41 ensambles. Los grupos no solo variaban en ensambladores utilizados sino en metodología empleada para cada ensamble lo cual podía generar diferencias entre dos grupos utilizando el mismo ensamblador. El *Assemblathon 1* evaluó los ensambles con distintas métricas de continuidad y precisión, posteriormente ordenó al mejor ensamble de cada uno con respecto a los demás en cada una de estas métricas. La primera conclusión más importante de este estudio es que como fue descrito previamente, las diferentes metodologías pueden producir resultados muy diferentes. La segunda conclusión importante es que ningún grupo fue consistentemente mejor en todas las categorías lo cual puede ser un indicio de que siempre se tendrá que tener un sacrificio de unas métricas por otras. Las limitaciones del *Assemblathon 1* fueron la naturaleza simulada de los datos ya que con datos reales es posible que algunas metodologías hubieran sido mejores que otras en algunas métricas cambiando su clasificación. La otra posible limitante dependiendo del enfoque, es que no se compararon los distintos pasos en un ensamble por individual sino en metodologías completas lo cual hace difícil comparar grupos modularmente.

Las principales limitaciones con respecto a los datos simulados fueron eliminadas en el estudio del *Assemblathon 2* [72] donde se hizo el mismo formato de competencia pero ahora utilizando 3 genomas reales de vertebrados. Ya que a diferencia del primer estudio no se contaba con el genoma correcto, se asistieron de sets de datos experimentales como mapas ópticos o secuencias de fósmidos para evaluar algunas métricas de precisión. Otra adición importante es la inclusión de otras tecnologías de secuenciación además de Illumina. Las conclusiones importantes de este estudio son las mismas del pasado relacionado a la consistencia de los ensambladores en cuestión a las distintas métricas donde otra vez, ninguno fue mejor en todas las categorías. Con la adición de dos especies extra también se vio que la

consistencia tampoco se presentó entre especies. Ya que muchos de los ensambles fueron hechos por los desarrolladores de los algoritmos de ensamble, es muy probable que los resultados sean los mejores posibles para cada ensamblador por lo cual esto se podría convertir en una limitante en la práctica ya que el conocimiento del algoritmo, la calidad de la documentación y la facilidad de instalación y uso pueden ser factores importantes a la hora de tener un proyecto de ensamble *de novo* en un laboratorio ajeno a los desarrolladores. Por último el estudio de GAGE (Genome Assembly Gold-standard Evaluations) [71] realizó pruebas comparando distintos ensambladores *de novo*. En este caso fue un solo grupo el que realizó los ensambles variando los parámetros lo cual era un reflejo más adecuado de la realidad de muchos grupos de trabajo. Para cada ensamblador eligieron como mejor ensamble el que tuviera las mejores métricas de N50 ya que como fue mencionado previamente, es la métrica más utilizada para comparar ensambles. También se utilizaron dos algoritmos de preprocesamiento en todos los ensambles. Sus conclusiones principales son que el grado de continuidad varía enormemente entre ensambladores y genomas, que la calidad de los datos más que los ensambladores tiene un efecto dramático en las métricas de ensamblado y que las métricas de precisión no correlacionan bien con las de continuidad. Una limitante que tienen estudios como este y los anteriores es que los algoritmos de ensamblado se encuentran bajo constante actualización, por lo cual los resultados pueden variar de un mes a otro dependiendo de las versiones utilizadas. En el proyecto de esta tesis se decide probar ciertos algoritmos de preprocesamiento y postprocesamiento en lugar de comparar ensambladores. Los algoritmos de preprocesamiento como mencionan en GAGE pueden llegar a ser claves en los resultados de un ensamble más que el propio ensamblador. El postprocesamiento en teoría debería ayudar a mejorar los estadísticos de continuidad y/o precisión.

2.2 Algoritmos preprocesamiento

Además de los filtrados comunes existen algoritmos especiales para eliminar errores en las lecturas y reducir en algunos casos todavía más el set de datos con el cual se trabajará. Existen diversos algoritmos para este preprocesamiento y a

continuación se describirán tres de ellos: Khmer [73], Quake [74] y ECHO [75]. Como se mencionó previamente, un concepto básico en los ensambles *de novo* es el "*k-mero*" que son palabras de tamaño k. Todos los ensambladores basados en grafos de bruijn utilizan este concepto como parte clave de su algoritmo. Los programas de preprocesamiento Khmer y Quake también se basan en ellos para su funcionamiento. En principio las lecturas pueden ser divididas en subsecuencias con un tamaño k cuyo límite superior es el tamaño de la lectura. Estas subsecuencias sobrelapan en k-1 bases. Como parámetro en ensambladores *de novo* basados en grafos de bruijn es el parámetro más importante. La elección del tamaño de k involucra un balance entre sensibilidad y especificidad siendo los tamaños más pequeños los que aumentan conectividad del grafo siendo más permisivos. Una propiedad muy importante que es intrínseca de la subdivisión en *k-meros* es que después de crear el "diccionario" de *k-meros* a partir de las lecturas, podemos cuantificar su aparición lo cual nos genera un espectro de *k-meros* como se puede observar en la figura 9.

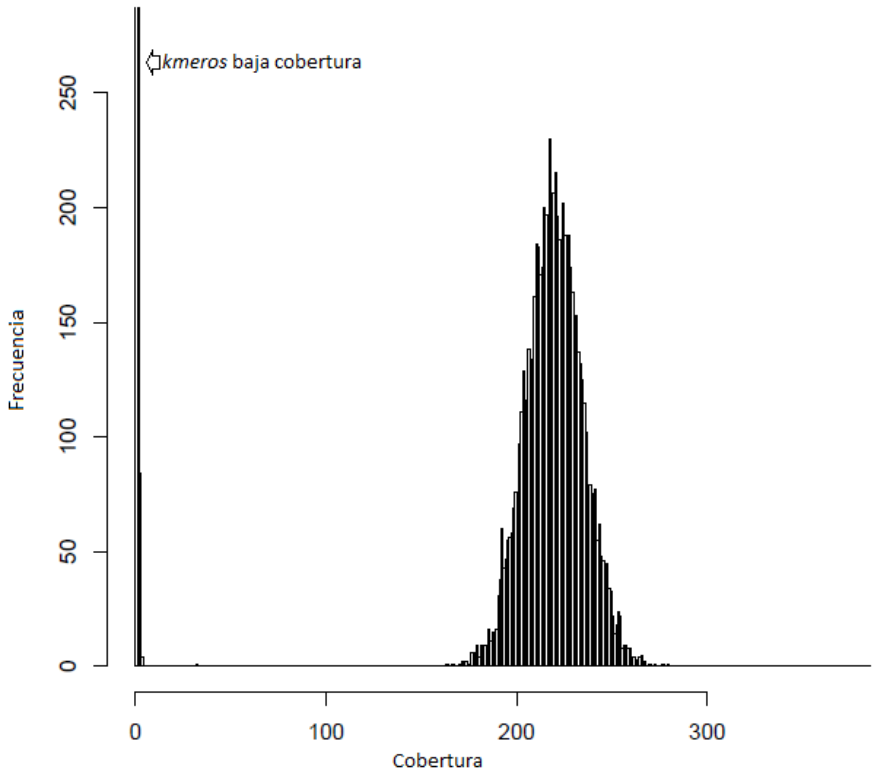

Figura 9: Espectro de *k-meros*. El primer pico que se señala son los *k-meros* de baja cobertura que en su mayoría son errores de secuenciación. El segundo pico muestra el espectro de *k-meros* reales del genoma secuenciado con una cobertura variable debido a la naturaleza azarosa del proceso de secuenciación.

Este espectro dependiendo del tamaño escogido de k nos muestra generalmente entre 2 y 3 picos en una biblioteca normal. El primer pico que se observa son *k-meros* que ocurren con una frecuencia bajísima lo cual significa un error de secuenciación la mayoría del tiempo o posible contaminación mínima. El segundo pico son en su mayoría secuencias únicas del genoma que por la naturaleza de alto rendimiento de la secuenciación han sido representadas con alta cobertura (este valor depende del tamaño del genoma y de la cantidad y tamaño de lecturas). Por último en algunos casos tenemos un tercer pico que normalmente son secuencias repetidas o a veces presencia de replicones extra con multicopia. Los algoritmos de preprocesamiento como Khmer y Quake utilizan estos espectros para procesar las

lecturas antes de meterlas a un ensamblador. Sin embargo, casi todos los ensambladores también tienen un paso donde toman en cuenta histogramas de *k-meros* para el ensamble. Khmer se basa en la premisa de que se puede reducir la cantidad de datos sin reducir la cantidad de información en las lecturas. Esto es debido a que al secuenciar un genoma el objetivo es generar suficientes datos para tener el genoma completo cubierto y para esto se trata de conseguir el mayor número de lecturas, ya que como se había mencionado antes, la secuenciación es azarosa y por lo tanto algunas posiciones serán cubiertas más que otras. Sin embargo, los instrumentos de secuenciación tienen una tasa de error y la acumulación de lecturas se puede también convertir en la acumulación de errores. Khmer separa las lecturas en palabras de tamaño k y usando la misma información de las lecturas, almacena progresivamente lecturas cuya mediana de cobertura de estos *k-meros* es menor a un valor asignado. Posteriormente se eliminan secuencias con *k-meros* de baja abundancia (en su mayoría errores de secuenciación) y por último se puede hacer otro corte opcional para bajar todavía más la cobertura. Quake es un algoritmo que utiliza el espectro de *k-meros* y las calidades de las bases para recortar y/o corregir lecturas antes de ser metidas a un ensamblador. Este corrector fue utilizado en el proyecto de GAGE donde en algunos casos mencionaban mejorías de ensambles después de ser preprocesados con este algoritmo.

Por último ECHO es el único algoritmo que no se basa en *k-meros* sino en sobrelapes de lecturas y un modelo estadístico para corregir dichas lecturas sin necesitar un genoma de referencia. Una gran ventaja que los creadores mencionan de su algoritmo es que todos los parámetros clave son automáticamente calculados por el programa lo cual hace más fácil el trabajo para el usuario quien no se tiene que preocupar por escoger los mejores parámetros en el programa. Una de las desventajas de este programa es que requiere mucho más RAM que los otros algoritmos además de que tarda más tiempo en terminar. También es importante mencionar que varios algoritmos cuentan con su propio algoritmo interno para tratar de corregir errores en las lecturas antes de ensamblar como ABySS [39], Velvet [38] y SGA [41].

2.3 Algoritmos postprocesamiento

Como se puede observar en los estudios de Assemblathon y GAGE, los ensambles realizados con tecnologías de secuenciación masiva tienden a presentar errores en la secuencia, ya sea por rearreglos estructurales o por variantes de un solo nucleótido. Cuando se finaliza un ensamble, se pueden alinear las lecturas de vuelta al ensamble para verificar congruencia con lo esperado dado las características de la biblioteca (como puede ser los tamaños de inserto correctos en lecturas pareadas). Existen diversos algoritmos que usan el principio de alineamiento de vuelta para intentar corregir ensambles, a continuación se describirán dos que usan esta información para dos fines distintos: IMAGE y ICORN [76, 77]. Estos algoritmos necesitan de lecturas pareadas. IMAGE tiene como objetivo principal mejorar la continuidad de un ensamble. Su metodología se base en primero alinear las lecturas de vuelta a la referencia. Las lecturas que alinean en extremos de contigs y sus parejas que no alinean son ensambladas en nuevos contigs. Estos nuevos contigs luego son alineados de vuelta al ensamble original. Después se vuelven a alinear las lecturas al nuevo ensamble y este proceso se itera hasta que los gaps son cerrados. ICORN también utiliza un algoritmo para alinear las lecturas de vuelta al ensamble, pero a diferencia de IMAGE el enfoque es corregir errores de ensamble. Después de alinear conservando solo las parejas de lecturas con tamaños de inserto correcto se hace un llamado de SNPs y microindeles y utilizando esta información se trata de corregir el ensamble asumiendo que estos son errores de ensamble. Evaluando cobertura antes y después de la corrección el algoritmo decide si quedarse o no con la corrección. Este proceso es repetido iterativamente.

3 Hipótesis

El uso de herramientas bioinformáticas de pre y post procesamiento afectará significativamente las métricas de calidad de ensamble para las distintas muestras.

4 Objetivo General

Buscar los mejores algoritmos de pre y postprocesamiento para ensambles de novo.

4.1 Objetivos Específicos

- Evaluar y comparar distintos algoritmos de preprocesamiento y postprocesamiento sobre datos simulados y reales
- Basado en los resultados generar análisis de costo-beneficio para distintos pipelines

5 Metodología

5.1 Diseño de flujo de trabajo

Se trabajó sobre dos sets de datos distintos: datos simulados y datos reales. Los simulados tienen la ventaja de permitirnos trabajar con datos ideales de la complejidad genómica deseada. Así podemos variar algunos factores (replicones multicopia, contenido de GC) y evaluar si tienen algún impacto en la consistencia de los algoritmos probados. En los datos simulados se probaron dos ensambladores diferentes: ABySS [39] y SGA [41]. Se decidió usar estos dos ensambladores para evaluar si se comparte una metodología de preprocesamiento óptima entre ellos. Estos dos ensambladores se basan en principios diferentes siendo el primero basado en grafos *de bruijn* y el segundo basándose en "string graphs". En datos simulados no se consideró que los filtros comunes tuvieran relevancia debido a la idealidad de los datos, por lo tanto no se aplicaron estos filtros aquí.

En el caso de datos reales se trabajaron con dos sets de genomas públicamente disponibles y con dos sets secuenciados en la Unidad Universitaria de Secuenciación Masiva de DNA (UUSMD) de la UNAM [78] con un genoma cercano de referencia disponible. En estos casos se trabajó con un solo ensamblador sin embargo se probaron más permutaciones de filtros. En todos los casos se descargó la anotación del genoma que se usaría como referencia para la búsqueda de genes. Todos los

ensambles fueron hechos en un solo procesador aunque existiera la opción de paralelización. Se separó en distintos módulos la evaluación de algoritmos empezando primero por algoritmos de preprocesamiento y después el módulo de postprocesamiento. Cada módulo cuenta con un grupo de programas candidatos de acceso libre para el análisis basado en recopilación de información reciente. Para evaluar los ensambles se utilizó la herramienta de QUAST [68] y se hicieron tres tipos de evaluación: precisión, integridad (continuidad), y global.

Las tablas de precisión son utilizadas para determinar los errores que se tuvieron en el ensamble. Ya que en seis casos se tiene una referencia disponible, estas tablas pueden proporcionar una tasa de error auténtica. En el caso de los dos genomas donde solo está disponible una especie cercana, muchos de los rearreglos y variantes son considerados variantes verdaderas por lo cual aunque claramente no se puede conocer con certeza qué es real y qué es error, nos podemos guiar bajo un principio de parsimonia donde de dos ensambles de similar tamaño, el que tenga menos diferencias con la especie más cercana es el más correcto. Los tipos de error se separan por niveles de gravedad: en el nivel uno se encuentran los errores de rearreglos grandes (inversiones o relocaciones de más de 1kb), en el nivel dos se encuentran los errores en secuencias pequeñas repetitivas (en mayoría colapso de repetidas) y por último en nivel tres se encuentran los errores de discrepancias y microindeles.

Las tablas de integridad o continuidad son basadas en cuatro métricas que se consideraron importantes para evaluar continuidad. La primera es el tamaño total del ensamble, la segunda es el número de genes completos encontrados, la tercera es el número de contigs y la cuarta es el NG50, el cual recientemente se ha considerado como un valor mejor que el N50 para evaluar un genoma debido a que es un valor normalizado tomando en cuenta el tamaño conocido o estimado del genoma objetivo [79].

Finalmente en las gráficas globales se combinan los tipos de errores previamente mencionados, genes completos, genes totales (incluyendo parciales), número de contigs, NG50 y tiempo de análisis. Estás gráficas de estrella son una manera fácil de evaluar generalmente un ensamble donde la mayor área representa un

mejor ensamble sin que se optimice una sola métrica en particular. La estrella también está segmentada de tal forma que la mitad superior sean estadísticos de integridad y continuidad, tres cuartas partes inferiores de errores y una cuarta parte de tiempo. Para hacer esta gráfica los valores fueron normalizados entre 0.2 y 1 para una mejor visualización de acuerdo a la siguiente fórmula:

$$0.2 + \frac{x - \min x}{\max x - \min x} \times 0.8$$

La única excepción fue cuando se graficaban genes completos los cuales se normalizaban con respecto al máximo de genes totales en vez del máximo genes completos. Para valores donde lo óptimo son valores pequeños (como número de contigs y errores) se usó la siguiente variación de la fórmula:

$$1.2 - (0.2 + \frac{x - \min x}{\max x - \min x} \times 0.8)$$

De esta forma, los valores más grandes son los más pequeños originalmente y viceversa. Una gráfica ideal se ve de la siguiente manera:

Figura 10: Ejemplo gráfica estrella. La mitad superior son métricas de continuidad, tres cuartos inferiores derechos son errores y el cuarto inferior izquierdo es tiempo.

Una limitante al normalizar de esta forma al producir estas gráficas es que puede llegar a ser muy sensible cuando en todos los ensambles hay muy poca variabilidad en una métrica, por ejemplo, si solo hubiera dos genes de diferencia entre el mejor ensamble y el peor ensamble, en el peor ensamble se vería diminuto el pedazo de esa métrica comparado con el otro ensamble aunque no es tan grave la diferencia. El valor mínimo de 0.2 se escogió para mejorar la visibilidad en secciones donde se encuentran los valores más pobres.

5.2 Simulación de datos

Se simularon lecturas pareadas de Illumina con tamaño de 100 bases cada una y cobertura promedio de 250x para los siguientes organismos:

- *Escherichia coli* str. K-12 substr. MG1655
- *Plasmodium falciparum* 3D7 (cromosomas 11 y 13)
- *Rhizobium etli* CFN 42 y su plásmido p42a
- *Streptomyces coelicolor* A3

Estos organismos fueron escogidos porque tienen diversas características que permitirán evaluar la metodología abarcando la mayor variedad de casos posibles. *E. coli* fue escogido ya que es un organismo modelo muy bien ensamblado y anotado. *P. falciparum* y *S. coelicolor* fueron escogidos debido a que son dos extremos en contenido de GC siendo el primero muy pobre y el segundo con muy alto contenido de GC. Por último *R. etli* fue escogido debido a que se quería probar la metodología con un organismo con plásmidos. En este caso la simulación de cobertura del plásmido fue el doble a la del genoma para tratar de simular un plásmido multicopia. Todas las lecturas fueron simuladas con el programa Sherman [80] con una tasa de error de .15%.

5.3 Bases de datos públicas de lecturas

Como parte de los esfuerzos mundiales de colaboración existen bases de datos donde muchos grupos de trabajo suben las lecturas crudas de sus corridas de se-

cuenciación masiva para su acceso público. Entre estas las más populares son el SRA de NCBI [81], el ENA del EBI [82] y el DRA de DDBJ [83]. Del ENA se descargaron dos corridas de Illumina GAIIX. La primera fue de *Escherichia coli* str. K-12 substr. MG1655 al igual que en los datos simulados, en este caso fueron 22, 720,100 lecturas pareadas de 100 bases lo que representa una cobertura estimada promedio de 979X. La segunda fue de *Bacillus subtilis* str. QB928 donde fueron 2, 477,257 lecturas pareadas de 75 bases lo que representa una cobertura estimada promedio de 89X. Estos sets están disponibles en `http://www.ebi.ac.uk/ena/data/view/ERR022075` y `http://www.ebi.ac.uk/ena/data/view/SRR535841` respectivamente.

5.4 Datos reales de *Klebsiella*

Para las pruebas a realizar con datos reales contamos con lecturas de *Klebsiella pneumonia* ChroAq1 aislada de un ambiente altamente contaminado con cromo y que resiste a altas concentraciones de este metal. Esto es de mucho interés ya que esta resistencia se da gracias a que puede reducir Cromo VI y Fierro III, un fenotipo que no se presenta en su pariente más cercano *Klebsiella pneumoniae* subsp. *pneumoniae* MGH 78578. Esta Klebsiella no tiene ninguna referencia publicada así que la calidad de los resultados depende mucho de utilizar las herramientas adecuadas tanto para el ensamble como para la anotación. Para esto partimos de lecturas ”pair-end” y ”mate-pair”, obtenidas con el instrumento Illumina GAIIX, que nos servirán para tener un genoma lo más contiguo posible tratando de resolver repetidas de hasta 2000 bases. Para mantener su comparabilidad con los otros sets de datos, no se utilizan las lecturas ”mate-pair” debido a que estas lecturas son incorporadas en los algoritmos de forma diferente a las ”pair-end”. Esta diferencia podría significar que ambos tipos de lecturas no compartan un mismo óptimo preprocesamiento. Aunque no se usaron en este escrito, estas lecturas ”mate-pair” posteriormente fueron utilizadas para el ensamble final y anotación (datos no mostrados). Las lecturas ”pair-end” fueron 11,999,772 lecturas pareadas de 72 bases lo que representa una cobertura estimada promedio de 325X tomando como referencia la especie más cercana disponible.

5.5 Datos reales de *Leuconostoc*

También se utilizaron lecturas de una cepa de *Leuconostoc* de la cual su pariente más cercano es *Leuconostoc mesenteroides* subsp. *mesenteroides* ATCC 8293. Esta cepa es de interés por su papel en la fermentación de la bebida mexicana del pulque. Esta cepa también fue secuenciada tanto por la metodología ”pair-end” como ”mate-pair” sin embargo al igual que en el caso anterior solo se utilizaron lecturas ”pair-end” para su comparabilidad aunque en el ensamble final y anotación fueron utilizados también los ”mate-pair”. Al final se utilizaron 35,986,030 lecturas pareadas de 38 bases lo que representa una cobertura estimada promedio de 189X tomando como referencia la especie más cercana disponible.

6 Resultados y discusión

6.1 Módulo preprocesamiento: Datos simulados

El primer análisis de preprocesamiento se hizo sobre datos simulados. Como se mencionó previamente, trabajar con datos simulados nos da la ventaja de poder trabajar con genomas de distintas complejidades. Para estos datos simulados, cada uno de los sets fue tratado de la siguiente forma, en paréntesis viene el nombre al cual se hace referencia en el resto del escrito:

- Lecturas crudas (Raw)
- Quake sobre lecturas crudas (Quake)
- Khmer normalizando a cobertura 30x (Khmer)
- ECHO sobre Khmer 30x. (Khmer ECHO)
- Muestreo azaroso de lecturas crudas para tener una cobertura aproximada de 30x (Sub)
- ECHO sobre muestreo azaroso (Sub ECHO)

En el caso de las lecturas simuladas, ECHO no pudo correr bien sobre ninguno de los sets de datos crudos por constricciones de memoria RAM y espacio en disco duro (puede llegar a generar archivos temporales de enorme tamaño en lo que está corriendo). Por esta razón se decidió probar ECHO solo sobre las submuestras de las lecturas. Quake solo es mostrado sobre las lecturas crudas en estas comparaciones ya que tuvo prácticamente el mismo efecto que ECHO en las submuestras. Para todas estas librerías se corrió tanto ABySS como SGA modificando el parámetro de K en ABySS y sus equivalentes en 2 pasos de SGA. El ensamble de mejor K por ensamblador se seleccionó en base a errores encontrados y si eran iguales de acuerdo a número de contigs. El mayor peso se le dio a ensambles con el menor número de errores de nivel 1 y 2, seguido por microindeles y finalmente seguido por mismatches. Antes de presentar las gráficas, se presenta una tabla donde se describe cada abreviación utilizada para los ensambles simulados:

Ensamble	Abreviación
Raw Abyss	R.A
Quake Abyss	Q.A
Khmer Abyss	K.A
Khmer ECHO Abyss	K.E.A
Sub Abyss	Sub.A
Sub ECHO Abyss	Sub.E.A
Raw SGA	R.S
Quake SGA	Q.S
Khmer SGA	K.S
Khmer ECHO SGA	K.E.S
Sub SGA	Sub.S
Sub ECHO SGA	Sub.E.S

Tabla 1: Abreviaciones ensambles datos simulados

Los resultados de la evaluación de precisión se muestran en la siguiente tabla:

E. coli	R.A	Q.A	K.A	K.E.A	Sub.A	Sub.E.A	R.S	Q.S	K.S	K.E.S	Sub.S	Sub.E.S
Nivel 1	0	0	0	0	0	0.02	0	0	0	0	0	0
Nivel 2	0	0	0.02	0.02	0.02	0.02	0	0	0	0	0	0
Nivel 3	1.83	1.81	3.40	3.26	1.74	2.31	0.50	0.18	0.48	1.34	0.20	0.46
***R. etli* (C/P)**												
Nivel 1	0/0	0/0	0/0	0/0	0/0	0/0.51	0/0	0/0	0/0	0/0	0/0	0/0
Nivel 2	0/0	0/1.02	0/0	0/0	0/0	0/0	0/0	0/0	0/0	0/0	0/0	0/0
Nivel 3	2.35/26.4	3.4/26.88	1.71/30.5	1.05/29.04	1.8/54.46	1.76/64.84	0.51/15.54	0.64/12.33	0.53/19.25	0.3/18.4	1.07/2.68	1.39/5.9
P. falciparum												
Nivel 1	0	0	0.12	0.12	0.06	0.06	0	0	0	0	0	0
Nivel 2	1.20	1.28	1.33	1.37	0.70	0.68	0	0	0	0	0	0
Nivel 3	14.77	12.93	18.05	21.69	12.51	11.37	2.15	3.02	2.44	3.36	4.32	4.65
S. coelicolor												
Nivel 1	0	0.01	0.01	0.03	0.02	0.01	0	0	0	0.03	0	0
Nivel 2	0.03	0.03	0.03	0.03	0.05	0.03	0	0	0	0.05	0	0
Nivel 3	4.31	5.63	5.34	4.81	4.41	4.71	0.14	0.18	0.14	2.47	0.09	0.22

Tabla 2: Tasas de error de ensambles de datos simulados. Cada número representa tasa por 100 kbs. En el caso de *R. etli* se separan los errores en cromosoma (C) y plásmido (P).

En esta tabla se puede observar que el ensamble más correcto la mayoría de las veces fue con los datos crudos o corregidos con Quake, salvo dos excepciones donde el muestreo azaroso producía los ensambles más correctos utlizando SGA (*R. etli* y *S. coelicolor*. En general no existe consistencia en los algoritmos de corrección. De particular interés es el algoritmo de ECHO que en la mayoría de los casos produce el efecto contrario al esperado generando peores ensambles. Otra cosa a notar es que Khmer casi siempre produce ensambles más erróneos. Los muestreos azarosos en general producen mejores resultados que los muestreos por Khmer en cuestión a precisión. En algunos casos, los resultados no fueron consistentes entre ensambladores, como es el caso de los datos de Khmer en *E .coli* donde empeoró la precisión usando ABySS mientras que al ensamblarse con SGA mejoró.

A continuación se muestran los resultados de la evaluación de integridad o continuidad en una tabla:

E. coli	R.A	Q.A	K.A	K.E.A	Sub.A	Sub.E.A	R.S	Q.S	K.S	K.E.S	Sub.S	Sub.E.S
Tamaño	4625040	4624930	4616235	4602576	4606031	4607328	4556600	4559494	4558427	4554237	4555313	4555908
Genes	4422	4422	4398	4395	4342	4352	4356	4370	4366	4344	4349	4342
Contigs	82	82	99	99	147	137	126	120	119	134	124	130
NG50	133751	133751	96117	96117	66030	70415	82866	97584	97591	73772	67415	78683
R. etli												
Tamaño	4630713	4635849	4628351	4607866	4524554	4515512	4529669	4528879	4532281	4524702	4489668	4498402
Genes	4291	4293	4285	4288	4131	4134	4239	4238	4235	4211	3377	3584
Contigs	44	51	56	45	258	276	63	64	71	89	1088	853
NG50	297761	286319	286319	297761	32336	29013	201586	195862	175553	130191	6154	8067
P. falciparum												
Tamaño	4906568	4907032	4879465	4878190	4853347	4845975	4865520	4854505	4844820	4823456	4859311	4834110
Genes	1129	1140	1109	1100	1021	1026	1089	1051	1045	957	963	965
Contigs	132	121	163	184	377	373	264	367	344	630	534	540
NG50	159681	269915	85035	73388	26272	26337	59754	33316	33334	15281	18062	18137
S. coelicolor												
Tamaño	8712413	8716004	8840244	9000400	8684267	8679608	8571947	8568559	8569575	8605391	8567948	8559360
Genes	7857	7853	7840	7823	7746	7753	7769	7765	7764	6507	7748	7729
Contigs	99	112	124	143	228	218	166	179	172	1737	194	227
NG50	231777	200469	155211	157658	86235	93757	138364	128886	133704	7774	114384	98174

Tabla 3: Métricas de integridad (continuidad) en ensambles con datos simulados

En estos estadísticos otra vez los ensambles de Quake y las lecturas crudas consistentemente son los mejores. Los ensambles de Khmer a pesar de tener un poco más de errores, consistentemente presentan contenido génico y NG50s altos comparados con los muestreos azarosos. Algo muy interesante es que la reducción de datos generó una pérdida casi nula de información donde muchas veces como en R.etli y S.coelicolor se recuperan prácticamente todos los genes que se encontraron en las crudas. Por último, cabe notar que varios de los ensambles que en la evaluación de precisión tuvieron muy pocos errores al ser examinados en métricas de continuidad resultaron ser ensambles altamente fragmentados.

Por útlimo mostramos los resultados del análisis global en las figuras 11, 12, 13, 14:

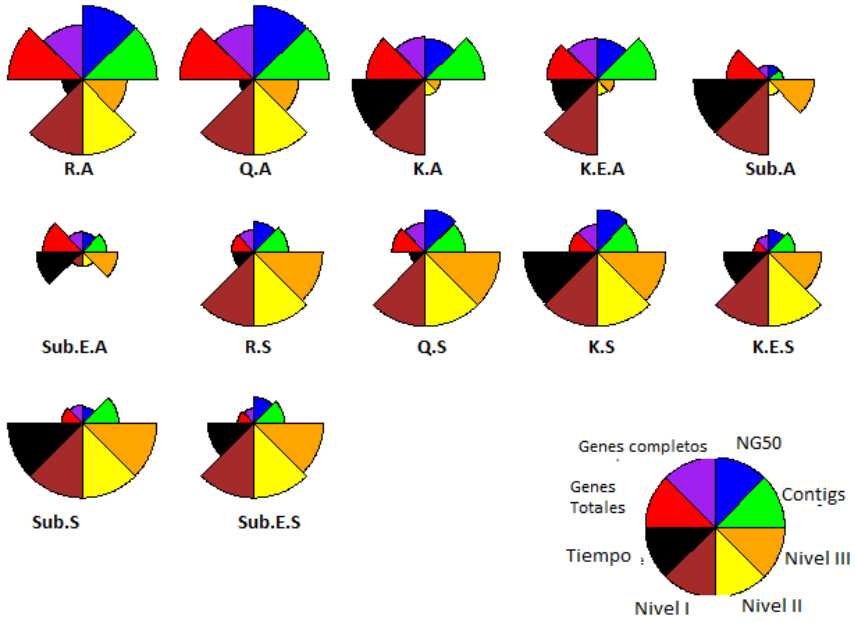

Figura 11: Gráfica estrella *E. coli* simulado. La métrica medida por cada sección se encuentra mostrada en la leyenda. Los valores estan normalizados de tal forma que una mayor área indica un mejor valor.

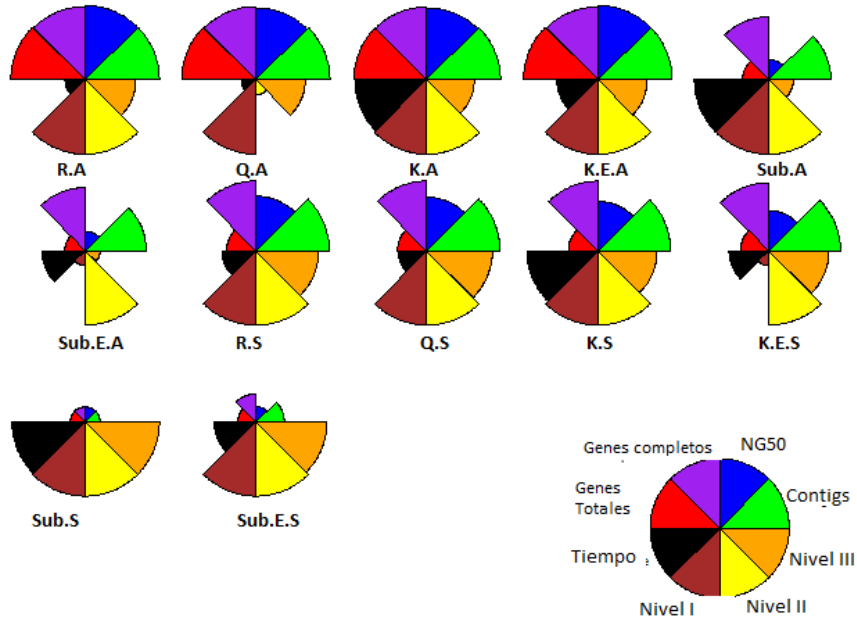

Figura 12: Gráfica estrella *R. etli* simulado. La métrica medida por cada sección se encuentra mostrada en la leyenda. Los valores estan normalizados de tal forma que una mayor área indica un mejor valor.

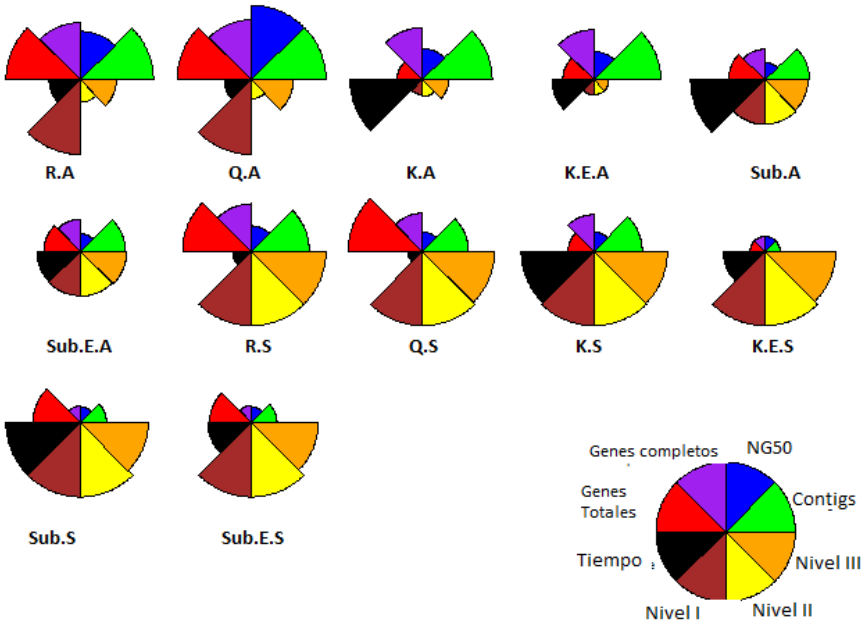

Figura 13: Gráfica estrella *P. falciparum* simulado. La métrica medida por cada sección se encuentra mostrada en la leyenda. Los valores estan normalizados de tal forma que una mayor área indica un mejor valor.

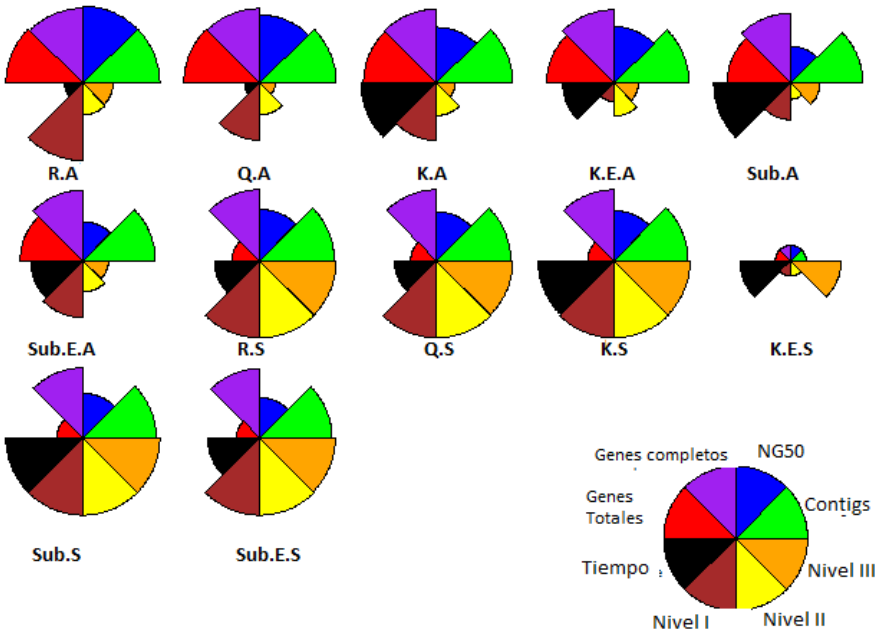

Figura 14: Gráfica estrella *S. coelicolor* simulado. La métrica medida por cada sección se encuentra mostrada en la leyenda. Los valores estan normalizados de tal forma que una mayor área indica un mejor valor.

En las gráficas globales sigue el mismo panorama donde en general las lecturas crudas suelen ser los mejores ensambles aunque metiendo la variable de tiempo en algunos casos la mejor opción parece ser usar Khmer ya que es un ahorro muy grande tiempo al probar varios parámetros de un ensamblador. Las diferencias pueden variar de 70 minutos probando 10 parámetros diferentes en el ensamblador a 8 horas probando esos mismos 10 parámetros. En la tabla 4 se muestra un estimado de tiempos por paso.

	ECHO submuestreo 30x	Khmer cobertura 30x	Quake lecturas crudas	ABySS submuestreo 30x*	ABySS lecturas crudas*	SGA submuestreo 30x*	SGA lecturas crudas*
E. coli	3.5 hrs	16 mins	40 mins	70 mins	8 hrs	70 mins	7.9 hrs
R. etli	4 hrs	17 mins	30 mins	50 mins	7.8 hrs	60 mins	6 hrs
P. falciparum	5 hrs	17 mins	40 mins	60 mins	8 hrs	60 mins	10 hrs
S. coelicolor	5.5 hrs	17 mins	30 mins	120 mins	15 hrs	120 mins	9 hrs

Tabla 4: Tiempos aproximados de pasos importantes. Los pasos marcados con * son pasos donde el tiempo toma en cuenta correr el ensamblador probando 10 diferentes valores del parámetro K o su equivalente

Cuando se trabaja con simulaciones, es esperado sobreestimar la calidad de los resultados ya que los problemas de los datos reales pueden ser difíciles de reproducir computacionalmente. Podría ser por esta razón que casi siempre los mejores resultados fueron obtenidos con secuencias crudas. Los algoritmos de corrección en muchos casos parecen meter errores. Esto es un poco sorprendente ya que en teoría no tendría por qué ser así, esto es posiblemente un artificio de los datos simulados pero no se puede descartar la posibilidad de que la complejidad del genoma sí pueda afectar estos programas. Algo que se puede ver es el efecto del muestreo de lecturas sobre la calidad del ensamble. Cuando agarramos solo un subset vimos que aun así se podían tener ensambles sin perder casi nada de información. Esto es muy importante ya que la reducción en tiempos puede ser muy significante especialmente cuando se están probando muchos parámetros en los ensambladores y una diferencia de hasta 8 veces en el tiempo de ejecución representa mucha ventaja a la hora de hacer investigación. Un ejemplo claro es en *S. coelicolor* donde en el mejor ensamble con datos crudos sólo se encontraron 5 genes más en comparación con el que se hizo con el muestreo con Khmer.

Otra observación muy importante es que en datos simulados que en teoría son casi perfectos no se pudo rescatar ningún ensamble cerrado, todos estaban

fragmentados en contigs. Esto reafirma la necesidad de combinar la secuenciación masiva con otras tecnologías ya que no es posible cerrar un genoma actualmente con solo una corrida de una tecnología de secuenciación masiva de segunda generación. Es también una pregunta importante para el investigador la necesidad de cerrar el genoma ya que aunque no estén cerrados se rescató más del 99% de los genes en todos los casos (con *R. etli* CFN42 se rescató el 100%) Uno de los casos más interesantes fue el de *R. etli*. Se observó mucha variación en el ensamble del plásmido dependiendo de la metodología utilizada. De especial interés es que cuando el cromosoma fue mejor ensamblado, el plásmido tuvo un muy alto número de discrepancias (95% de las discrepancias fueron en plásmido). Debido a que las diferencias más significantes fueron en número de discrepancias y no indeles o rearreglos, se podría pensar que usando las mismas lecturas se podría corregir el ensamble.

El espectro de *k-meros* que se puede observar en las figuras 15 y 16 puede dar un indicio del porqué de estos resultados. Por ejemplo, se puede observar que en principio, el muestreo "azaroso" que hicimos de los sets crudos no rescató el espectro de *k-meros* original y ya que ECHO trabajo sobre este subset pudo presentar los mismos sesgos y consecuentemente introducir errores.

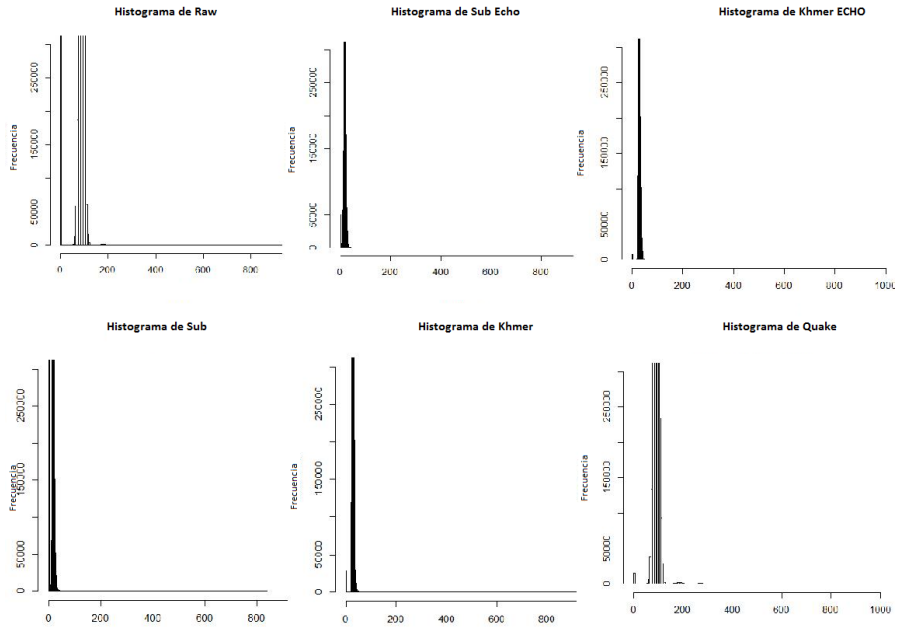

Figura 15: Espectro de *k-meros* de *E. coli* para diferentes sets de datos

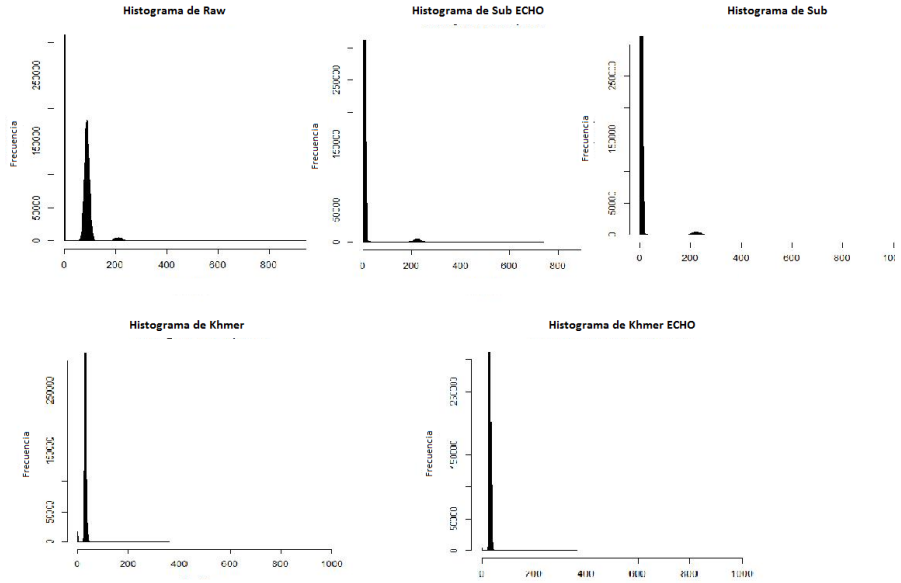

Figura 16: Espectro de *k-meros* de *R. etli* para diferentes sets de datos

Por último, se tiene que tener cuidado con las métricas comúnmente utilizadas para evaluar un ensamble (número de contigs, n50 y contig más largo) ya que se

pudo observar especialmente en ensambles como el de *P. falciparum* como esos ensambles más contiguos pueden tener un número muy grande de errores difíciles de detectar y corregir (rearreglos). Sin embargo los ensambles más correctos en estos casos no solo tenían el doble de contigs sino que les faltaba 6% más que a estos ensambles con errores, aunque esto casi no se refleja en la búsqueda de genes por parte de Quast.

6.2 Módulo preprocesamiento: Datos reales

Los sets de datos reales nos permiten poder evaluar los algoritmos en un contexto más realista. Para los sets de datos reales solo se utilizó ABySS sin embargo se consideraron más filtros. ECHO se pudo correr en todos los datos crudos excepto en los de *E. coli* reales. También se probó reducir aun más la cobertura con Khmer a 5X. Además de los algoritmos previamente utilizados, se usaron los filtros comunes siguientes:

- Filtro de Ns (no se permitieron lecturas con N)
- Filtro de adaptador
- Filtro de secuencia de baja complejidad
- Filtro de mediana de calidad debajo de 20

No se probaron filtros de trimming principalmente porque ya existe un artículo reciente viendo sus efectos sobre ensambles [56]. De todas maneras ABySS tiene un paso previo al ensamble donde rasura bases de calidad 3 lo cual se consideró suficiente para asegurar el mejor ensamble. Otra cosa que es importante mencionar es que aunque bien estos sets de datos varían en tamaño de lectura, esto no debería ser factor debido a los algoritmos de preprocesamiento elegidos. Al ser céntricos en *k-meros* Quake y Khmer, todas las lecturas son reducidas a *k-meros* por lo tanto la información del tamaño de lectura se vuelve irrelevante a la hora de corregir. ECHO por su parte en su escrito fue probado en distintas bibliotecas con tamaños de lectura variables sin observar discrepancias [69]. Los sets de datos a probar junto con sus abreviaciones son mostrados en la tabla siguiente:

Ensamble	Abreviación
Raw	Raw
Quake Raw	Q.R
ECHO Raw	E.R
Khmer 30x Raw	K30.R
Khmer 5x Raw	K5.R
Khmer 30x ECHO	K.E.R
Filtros comunes	Filt
Quake sobre Filt	Q.F
ECHO sobre Filt	E.F
Khmer 30x sobre Filt	K30.F
Khmer 5x sobre Filt	K5.S
Sub	SUB

Tabla 5: Abreviaciones ensambles datos reales

En todos los sets de datos para el ensamblado el único parámetro que se movió en el ensamblador es la K a excepción de los que se normalizó a 5x donde se redujo el número de conexiones necesarias para unir dos contigs (n).

Se empieza nuevamente por la evaluación de precisión:

E. coli	Raw	Q.R	E.R	K30.R	K5.R	K.E.R	Filt	Q.F	E.F	K30.F	K5.F	Sub
Nivel 1	0.04	0.04	N/A	0.06	0.04	0.08	0.04	0.06	N/A	0.06	0.06	0.11
Nivel 2	0.09	0.13	N/A	0.09	0.11	0.09	0.09	0.11	N/A	0.13	0.09	0.57
Nivel 3	4.87	3.75	N/A	3.18	4.30	3.27	4.87	4.16	N/A	4.29	4.90	10.13
B. subtilis												
Nivel 1	0.02	0.04	0.02	0.02	0	0.02	0.02	0.05	0.05	0.09	0.04	0.02
Nivel 2	0.02	0.02	0.02	0	0	0.07	0	0	0.02	0.05	0.04	0.05
Nivel 3	7.45	6.93	11.82	8.24	2.55	7.92	78.92	77.73	77.39	108.89	38.01	11.81
Klebsiella												
Nivel 1	0.94	0.99	0.77	1.02	1.04	0.98	0.95	1.02	0.99	0.92	0.99	0.79
Nivel 2	0.58	0.64	0.68	0.65	0.67	0.55	0.60	0.65	0.57	0.70	2.07	0.42
Nivel 3	627.72	630.26	641.12	642.96	642.23	638.1	628.54	621.74	629.59	617.1	617.26	547.53
Leuconostoc												
Nivel 1	2.10	2.10	2.04	2.10	2.11	2.15	2.10	2.10	2.23	2.10	2.12	1.91
Nivel 2	1.74	1.78	1.72	1.74	1.74	1.78	1.74	1.79	1.75	1.73	1.75	1.65
Nivel 3	1765.43	1766.88	1780.6	1765.27	1769.36	1769.17	1767.28	1768.87	1779.74	1767.19	1771.41	1759.29

Tabla 6: Tasas de error de ensambles de datos simulados. Cada número representa tasa por 100 kbs.

A diferencia de los datos simulados, las lecturas crudas en muchos casos no fueron las de menos errores. ECHO cuando se pudo utilizar sobre lecturas crudas

produjo mejoras en errores graves y/o discrepancias, sin embargo sus efectos sobre el subset de Khmer generalmente fue negativo. Khmer a diferencia de los datos simulados muchas veces presentó mejora en tasa de errores mientras que Quake sigo siendo inconsistente. Los ensambles con filtrados comunes nunca presentaron mejoras y en algunos casos como en *Bacilus* aumentaron considerablemente la tasa de errores. Cabe volver a recalcar que en los casos de *Klebsiella* y *Leuconostoc* las tasas son tan altas debido a que mucha de esa variación es real con respecto a su referencia más cercana, sin embargo en estos ensambles que eran de tamaños muy similares generalmente lo más parsimonioso es pensar que el que tiene menos de estos es el más correcto. Para explorar este punto, los ensambles de *Klebsiella* se alinearon entre ellos usando BLAT [84]. Los contigs más grandes que presentaban incongruencias fueron seleccionados para realizar un BLAST y encontrar regiones discrepantes en ensambles con homología a genes anotados para posteriormente analizar el alineamiento del gen contra cada uno de los contigs. El contig candidato fue una región que contiene dos genes de fimbria, *pilB* y *pilC* con el cual ningún contig alinea perfectamente excepto uno. Se volvió a hacer un alineamiento con BLAT ahora usando como blanco esta región donde se dieron los resultados mostrados en la figura 17.

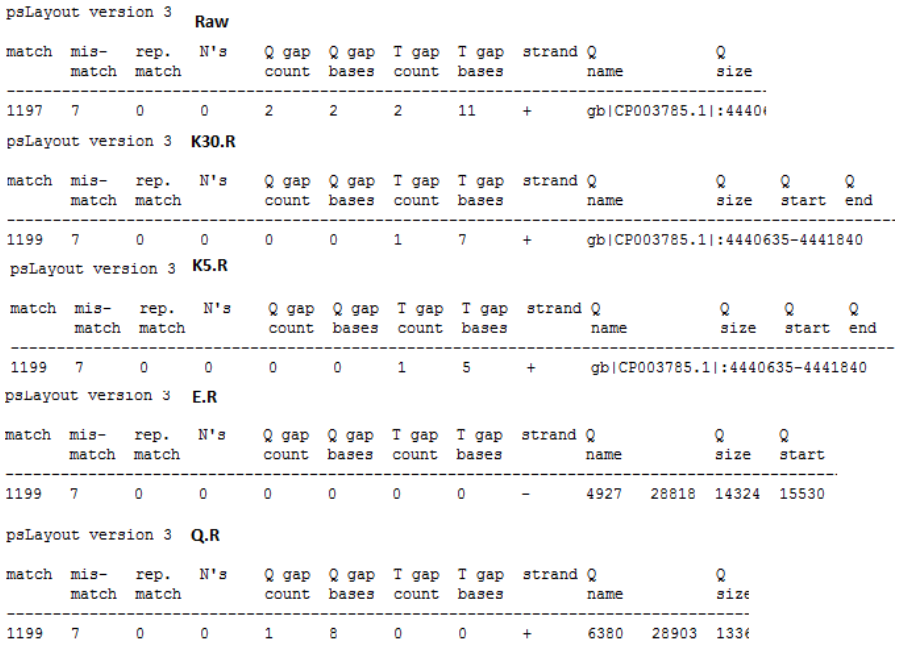

Figura 17: BLAT región *Klebsiella* vs ensambles. Los ensambles observados son de arriba hacia abajo: Raw (lecturas crudas), K30.R (Khmer normalizando a cobertura 30x), K5.R (Khmer normalizando a cobertura 5X), E.R (ECHO sobre lecturas crudas), Q.R (Quake sobre lecturas crudas).

Se puede observar que solo uno alineo perfectamente que fue el que utilizó ECHO, después de ese el contig de 5x fue el que menos bases tuvo cubiertas con el gap el cual, cuando se inspeccionó más a fondo, era producido por Ns que estaban en el ensamble. También se puede observar que todos presentan 7 mismatches los cuales son candidatos de variantes verdaderas.

Los resultados de la evaluación de integridad o continuidad se muestran en la siguiente tabla:

E. coli	Raw	Q.R	E.R	K30.R	K5.R	K.E.R	Filt.	Q.F	E.F	K30.F	K5.F	Sub
Tamaño	4645365	4631426	N/A	4629049	4634288	4629657	4645213	4636511	N/A	4627757	4627873	4578246
Genes	4429	4417	N/A	4416	4417	4417	4429	4425	N/A	4408	4410	4277
Contigs	77	87	N/A	77	89	78	77	79	N/A	79	93	183
NG50	127414	110457	N/A	133711	118999	131744	127414	115397	N/A	125945	95832	50769
B. subtilis												
Tamaño	4079598	4082849	4079885	4080107	4015880	4077667	3978726	4000616	3937437	3348315	2451846	4078095
Genes	4025	4023	4022	4025	2980	4019	3121	3210	2915	1706	762	3984
Contigs	33	36	35	33	1234	37	1024	944	1258	2231	2859	83
NG50	487963	487953	487961	487961	4580	415330	5845	6401	4481	1437	600	77845
Klebsiella												
Tamaño	5599502	5622187	5602117	5647072	5664426	5622924	5590646	5479427	5563724	5526404	5542826	4035912
Genes	4101	4234	4277	4276	4305	4146	4050	3858	4009	3860	3801	1735
Contigs	459	313	348	268	236	413	522	665	559	696	687	2327
NG50	28950	47892	42197	53946	67706	32909	25011	18753	23725	19603	17387	1594
Leuconostoc												
Tamaño	1900130	1900857	1861147	1900630	1891457	1900298	1898456	1898785	1876195	1900016	1885108	1876793
Genes	1499	1500	1485	1500	1499	1499	1499	1499	1493	1499	1492	1488
Contigs	19	17	27	18	19	15	18	17	21	18	29	47
NG50	139884	184604	121023	300084	231165	300084	184604	184604	139891	300084	81668	133009

Tabla 7: Métricas de integridad (continuidad) en ensambles con datos reales

Khmer otra vez tuvo un desempeño muy bueno en estas métricas. Aquí se puede observar cómo incluso reduciendo la cobertura a 5x se pudo recuperar mucha información y en algunos casos más información que con las puras lecturas crudas. ECHO solo en uno de los casos tiene un desempeño muy malo que es en *Leuconostoc*. También se puede observar que los sets donde hubo filtros comunes son consistentemente inferiores en estas métricas a los basados en lecturas crudas. El muestreo azaroso siempre es considerablemente peor que casi todas las alternativas. Quake sigue siendo un poco inconsistente en estas métricas aunque siempre produce buenos resultados.

Por último se presetan las gráficas globales:

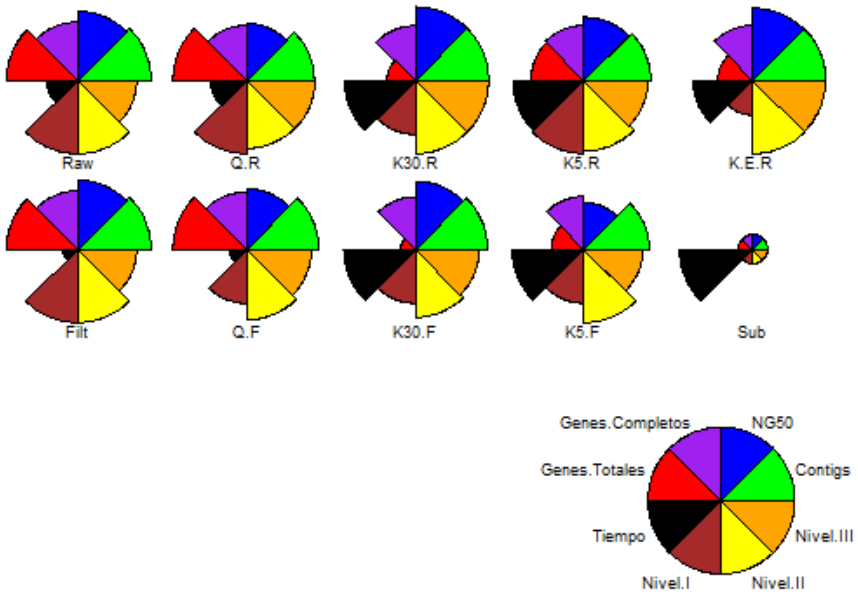

Figura 18: Gráfica estrella *E. coli* real. La métrica medida por cada sección se encuentra mostrada en la leyenda. Los valores estan normalizados de tal forma que una mayor área indica un mejor valor.

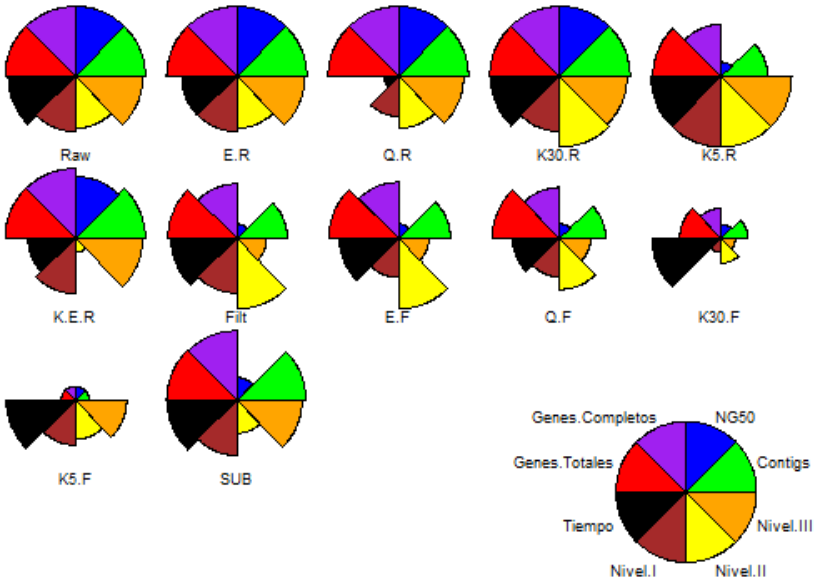

Figura 19: Gráfica estrella *B. subtilis*. La métrica medida por cada sección se encuentra mostrada en la leyenda. Los valores estan normalizados de tal forma que una mayor área indica un mejor valor.

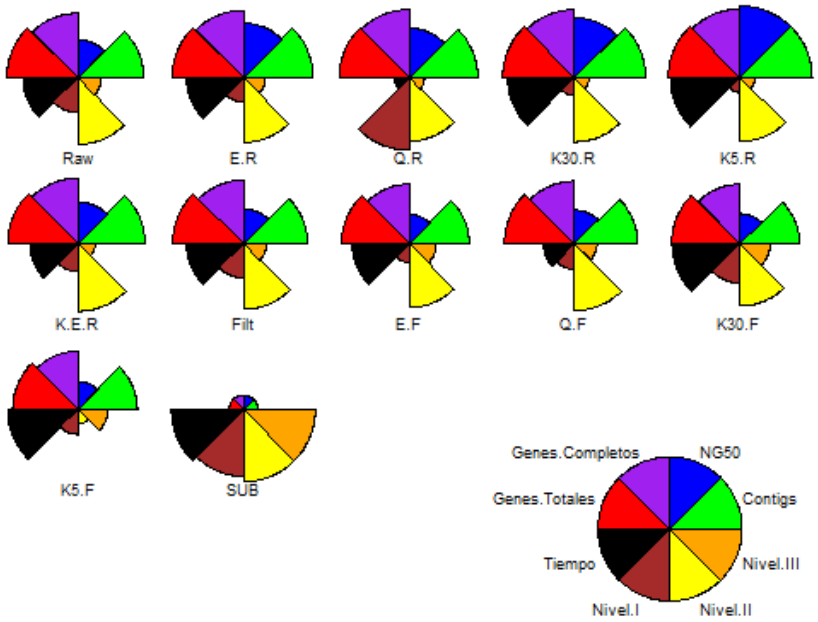

Figura 20: Gráfica estrella *Klebsiella* real. La métrica medida por cada sección se encuentra mostrada en la leyenda. Los valores estan normalizados de tal forma que una mayor área indica un mejor valor.

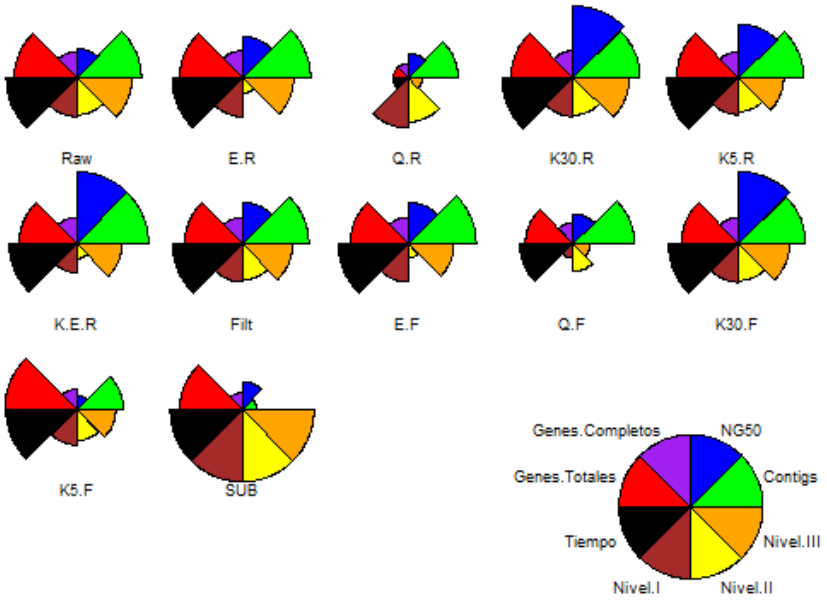

Figura 21: Gráfica estrella *Leuconostoc*. La métrica medida por cada sección se encuentra mostrada en la leyenda. Los valores estan normalizados de tal forma que una mayor área indica un mejor valor.

En estas gráficas donde se toma en cuenta el tiempo se pude observar otra vez el valor de Khmer. En casos donde no había muchas lecturas iniciales de entrada como en *B. subtilis* aun cuando la diferencia de tiempo no fue mucha, se produjeron mejores ensambles globalmente. El muestreo azaroso produce en general resultados pobres y los filtrados comunes también. Los resultados de Quake son consistentes con lo previo reportado en GAGE [71] donde no siempre era óptimo filtrar con Quake. En el artículo de Quake con el único set de datos que trabaja reales es con datos de *E. coli* para los cuales fue los únicos donde se encontró una diferencia considerable con respecto a usar las puras lecturas crudas. La gran diferencia en observaciones que se puede observar comparando datos reales con datos simulados enfatiza la necesidad de probar algoritmos en datos reales en vez de simulaciones.

6.3 Módulo postprocesamiento

Se probaron dos algoritmos en este módulo: ICORN [77] e IMAGE [76]. El primero tiene como énfasis corregir errores mientras que el segundo extender los contigs y cerrar gaps. Ambos se basan en mapeo iterativo de las lecturas de vuelta al ensamble. Para este módulo solo se hicieron pruebas con un ensamble por genoma. Se evaluaron el ensamble crudo, ICORN sobre el ensamble crudo, IMAGE sobre el ensamble crudo y ICORN sobre el resultado de IMAGE. En ICORN se hicieron 5 iteraciones mientras que en IMAGE se hicieron 15. Ya que las diferencias en continuidad en muchos casos eran nulas solo se hizo una tabla de precisión para ver mejoras con ICORN e IMAGE además de una tabla de diferencia de genes encontrados para verificar la efectividad de IMAGE. En cuestión a tiempos fueron muy variables ya que aunque por ejemplo IMAGE podía tardar entre 2 y 24 horas en correr mientras que ICORN variaba entre 5 y 12 horas. Al final ambos programas agregan un tiempo considerable al proyecto.

En la siguiente tabla se muestran los resultados de precisión para datos simulados y reales:

Simulados				
***E. coli* simulado**	Original	ICORN	IMAGE	ICORN_IMAGE
Nivel 1	0	0	0.17	0.17
Nivel 2	0.02	0.02	0.17	0.17
Nivel 3	2.84	1.99	9.06	7.88
P. falciparum				
Nivel 1	0.12	0.12	0.28	0.28
Nivel 2	1.33	1.35	1.48	1.48
Nivel 3	16.70	17.07	25.68	22.96
R. etli				
Nivel 1	0	0	0.17	0.17
Nivel 2	0	0	0.02	0.02
Nivel 3	26.70	13.30	52.89	34.80
S. coelicolor				
Nivel 1	0.01	0.01	0.10	0.10
Nivel 2	0.03	0.03	0.03	0.03
Nivel 3	5.34	3.61	8.41	6.57
Reales				
***E. coli* real**				
Nivel 1	0.04	0.04	0.97	0.97
Nivel 2	0.86	0.86	0.86	0.86
Nivel 3	4.390	4.390	12.93	11.11
B. subtilis				
Nivel 1	0.02	0.02	0.12	0.12
Nivel 2	0.02	0.02	0.02	0.02
Nivel 3	5.16	5.16	8.130	6.46
Klebsiella				
Nivel 1	0.97	0.97	1.02	1.02
Nivel 2	0.58	0.58	0.65	0.65
Nivel 3	634.19	632.81	634.82	633.2
Leuconostoc				
Nivel 1	2.26	2.26	2.16	2.16
Nivel 2	1.79	1.79	1.74	1.74
Nivel 3	1765.35	1765.81	1765.4	1765.92

Tabla 8: Tasas de error de ensambles antes y después de postprocesamiento. Cada número representa tasa por 100 kbs. La tabla está separada entre datos simulados y datos reales.

Aunque en los datos simulados ICORN consistentemente presentaba mejoras (excepto en *P. falciparum*), en los datos reales las mejoras son marginales. En ningún caso ICORN pudo corregir los errores graves los cuales eran introducidos con frecuencia al utilizar IMAGE. Esto se verificó en todas las iteraciones de IMAGE donde el número de errores iba aumentando con cada iteración (datos no mostrados). A continuación se presenta una tabla de genes por ensamble, donde

la columna de diferencia de genes representa la ganancia o pérdida de genes de IMAGE con respecto al ensamble original.

Organismo	# Genes Original	# Genes IMAGE	Diferencial completos	Diferencial total
E. coli (sim)	4394+44 part	4495 + 29 part	52	37
R. etli	4288 + 20 part	4291 + 17 part	3	0
P. falciparum	1108 + 80 part	1096 + 83 part	-12	-9
S. coelicolor	7837 + 60 part	7854 + 56 part	17	13
E. coli (real)	4429 + 52 part	4445 + 36 part	16	0
B. subtilis	4025 +7 part	4004 + 28 part	-21	0
Klebsiella	4043+389 part	3858 + 571 part	-185	-3
Leuconostoc	1504 + 50 part	1503 + 51 part	-1	0

Tabla 9: Diferencia génica entre ensambles antes y después de IMAGE. La primera columna es el organismo, la segunda y tercera columna es el número de genes completos más el número de genes parciales. La cuarta columna es el diferencial de genes completos de IMAGE con respecto al ensamble original y la quinta columna es el diferencial de genes totales.

En casi todos los sets de datos reales IMAGE además de introducir errores, redujo el número de genes encontrados. Estos resultados parecen indicar que IMAGE no vale la pena en general correrlo mientras que ICORN se recomienda usar si se desea el ensamble más correcto posible (aunque las diferencias sean posiblemente marginales).

Algo importante de estos dos programas es que la documentación es un poco pobre y los parámetros modificables también pueden ser problemáticos ya que en algunos casos como en IMAGE hay valores de tamaño de inserto que el script tiene predeterminados sin que puedas modificarlos sin meterte al script.

6.4 *BacterialAnnotAid*: Paquete de Bioconductor en desarrollo

Por último se comenzó el desarrollo de un paquete de Bioconductor con el título de BacterialAnnotAid el cual trata de ayudar en la anotación de organismos procariontes de una manera amigable al usuario. El paquete toma como entrada un resultado de BLAST en formato XML o realiza un BLAST dentro del paquete utilizando como entrada un archivo FASTA o un multi-fasta generado a partir de búsqueda de ORFs en los organismos por medio de un algoritmo como Glimmer

[85]. Los resultados de los mejores hits se muestran en forma de alineamientos con coloreados especiales para hits, mismatches y gaps enfatizando dominios. Esto puede ayudar cuando se está anotando un genoma y se tienen errores en el ensamble. Al mostrar cada gen en su contexto de homología, se puede buscar como alinean las lecturas en genes cuyo alineamiento con top hits muestra dominios relativamente importantes truncados. Este truncamiento en la búsqueda de ORFs puede ser ya sea por un error de ensamble que introdujo un frameshift o una variación real (si se observa un buen soporte por las lecturas originales en esa área). Así se puede compensar la disminución en precisión que puede haber al elegir un ensamble contiguo contra uno más fidedigno concentrandose solo sobre los genes de interés y verificar cuales tienen posibles errores. Las figuras 22 y 23 muestran ejemplos del output con ORFs en Leuconostoc:

En el caso del ORF en la figura 23, este parece ser confiable así que esta región no requeriría de inspección manual para posibles errores de ensamble. Este paquete también podría en un futuro ampliarse para usos en otro tipo de proyectos de bioinformática orientados a procariontes. Hasta ahora también se ha utilizado para análisis de TSSs (Transcription Start Sites) donde personalizando el output de BLAST solo se regresan hits cuyo comienzo de alineamiento sea en las primeras bases lo cual es útil cuando se buscan sitios de inicio de transcripción alternos intragénicos.

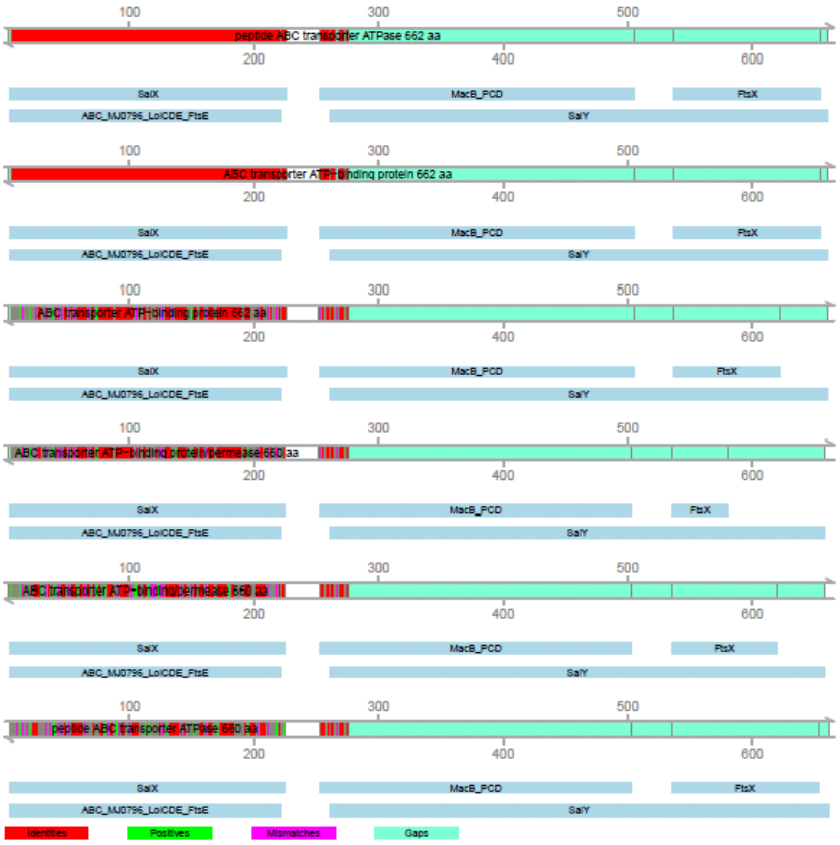

Figura 22: ORF potencialmente erróneo en *BacterialAnnotAid*. Los tractos rojos marcan identidades en el alineamiento, los verdes cambios de aminoácido "positivos", los morados mismatches y los azul claro gaps. Abajo de cada alineamiento se encuentran resaltados dominios en los homólogos cercanos.

En la figura 22 se muestra un ORF potencialmente erróneo. El hecho de que los mejores hits tengan ciertos dominios que este ORF no contiene parece indicar que este ORF es un candidato a error de ensamble. En un caso como éste sería bueno revisar cómo alinean las lecturas a esta región del ensamble.

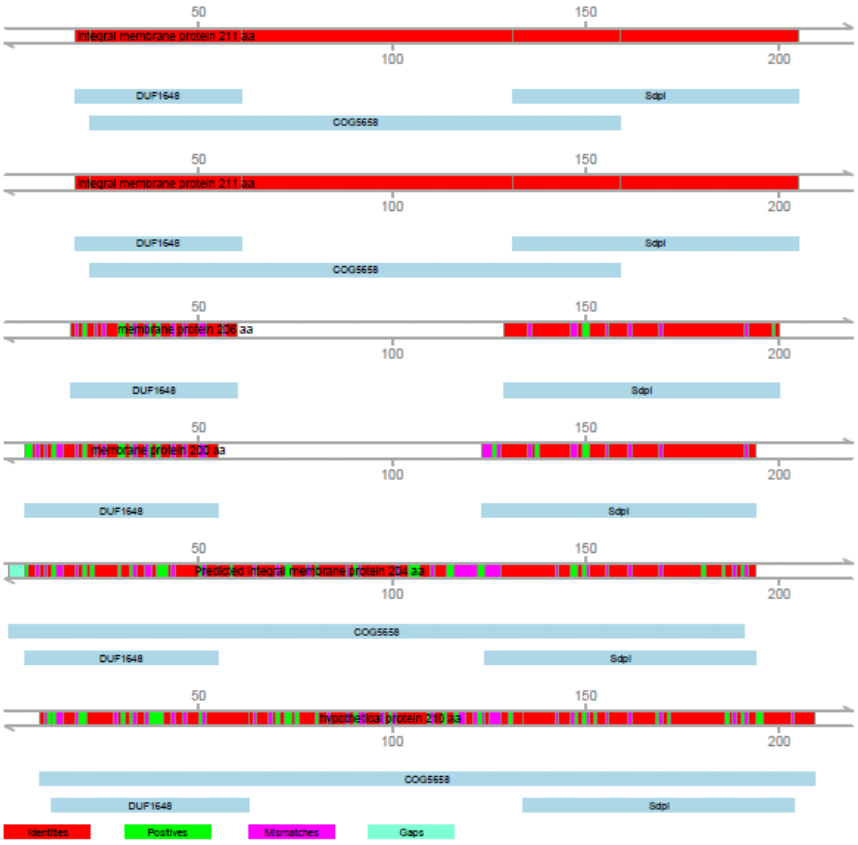

Figura 23: ORF probable en *BacterialAnnotAid*. Los tractos rojos marcan identidades en el alineamiento, los verdes cambios de aminoácido "positivos", los morados mismatches y los azul claro gaps. Abajo de cada alineamiento se encuentran resaltados dominios en los homólogos cercanos.

7 Conclusiones

Los algoritmos de preprocesamiento y postprocesamiento tienen un impacto significativo en las métricas comúnmente utilizadas para evaluar ensambles *de novo* además de otros factores no siempre medidos pero muy importantes como el tiempo. En muchos casos el impacto de los algoritmos de preprocesamiento y postprocesamiento puede ser negativo sobre la calidad del ensamble final. Como han concluido estudios anteriores comparando ensambladores, no existe una metodología óptima para garantizar el mejor ensamble posible con todas las métricas maxi-

mizadas sino que casi siempre se puede esperar sacrificar la continuidad por la precisión o viceversa. Depende del investigador las métricas que prefiere optimizar. Se podría dar un ejemplo donde al investigador le interesa explorar todos los genes de una especie y por lo tanto lo que más le importa es la continuidad sin importar si existen errores de precisión, en casos como este se podría usar Khmer y podría valer la pena el riesgo de invertir tiempo en IMAGE y ver si contribuye a la extensión del ensamble. O en el otro extremo, si un investigador está interesado en variantes o tasas de mutación de especies no antes descritas entonces es posible que le interese el ensamble lo más correcto posible entonces es posible que invierta el tiempo corriendo ICORN y ECHO si cuenta con los recursos computacionales. Por otro lado, una empresa que ofrece servicios bioinformáticos podría valorar la consistencia que puede ofrecer una metodología en tiempos y que constantemente entrega muy buenos resultados aunque no sean los mejores en una sola métrica, enfocándose en correr Khmer sobre los datos y ensamblar sin ningún otro filtro. Los resultados de los filtros comunes son un poco sorprendentes pero hay dos cosas que cabe mencionar al respecto: la primera es que los ensambladores constantemente están mejorando y por lo tanto es posible que ya hoy en día estén tan avanzados que los pasos de filtrado comunes sean un extra no necesario y que puede en muchos casos reducir no solo datos sino también información que puede ser útil. La segunda es que por su naturaleza, programas como Khmer o Quake lidian con muchos de los errores que se tratan de corregir con filtros comunes implícitamente. Por ejemplo, una lectura con muchos errores presentará *k-meros* con muy baja frecuencia por lo tanto, es eliminado naturalmente en estos programas sin necesidad de analizar las calidades de las bases. Se reconocen varias limitaciones con respecto a este proyecto de las cuales la principal es el tiempo que no permite que se realicen todas las permutaciones posibles de filtros y parámetros de ensambladores por lo cual no puede ser considerado un proyecto exhaustivo y sería posible (aunque raro) que exista un set de parámetros del ensamblador que hubiera cambiado los roles de ciertas metodologías. Otra limitación fue el cómputo ya que ECHO hubiera sido interesante probarlo en más sets de datos pero constricciones de RAM y disco duro no permitieron probarlo bien. También se sabe que existen muchos

algoritmos los cuales no pudieron ser incluidos debido a falta de tiempo o porque son muy recientes como REAPr [86], Nesoni [87] o Pilon [88]. REAPr y Pilon son otras herramientas de postprocesamiento que también se basan en mapeo de lecturas de vuelta al ensamble. REAPr busca sitios de posibles errores y rompe los contigs en esas posiciones aumentando la fragmentación pero reduciendo la tasa de error mientras que Pilon trata de corregir áreas de discrepancia entre las lecturas y el ensamble similar a ICORN. Nesoni es un set de herramientas que incluye unas que trabajan con el espectro de *k-meros* y que podría presentar una alternativa a Quake. Por último, otra limitante reconocida es que solo se usó un genoma eucariote en el análisis. Aun con estas limitaciones los resultados del trabajo son relevantes al campo ya que hasta la fecha no se conoce ningún trabajo donde se comparen algoritmos de preprocesamiento y postprocesamiento sobre datos simulados y reales. Para finalizar, existen varios algoritmos de dominio público que pueden ser utilizados para distintos tipos de proyecto basados en secuenciación masiva, desde los pasos primeros de filtrado de datos hasta los últimos de análisis de resultado. Sin embargo, es importante notar que los programas no son suficientes sin una persona que pueda darle significado biológico a los resultados y, al mismo tiempo, entender el programa y el porqué de los resultados. Es muy importante destacar que cualquier algoritmo computacional siempre arrojará resultados que pueden no tener sentido biológico, entonces hay que tener cuidado con los datos que entran ya que su calidad se refleja en la calidad de los resultados (*rubbish in rubbish out*).

8 Perspectivas

Debido a que estos algoritmos de preprocesamiento y postprocesamiento tienden a ser estáticos es posible que estos resultados a diferencia de los de un estudio como GAGE sean vigentes más tiempo y se puedan complementar con algoritmos que no fueron probados en este proyecto. Se espera que el mejoramiento en rendimiento, precisión y tamaño de lectura de los propios aparatos de secuenciación puede hacer cada vez menos necesario el filtrado y mejoramiento de ensambles aunque no siem-

pre todos los laboratorios contarán con las últimas tecnologías de secuenciación lo cual sigue haciendo prácticas este tipo de guías. Independientemente de la metodología bioinformática a utilizar, se recomienda ampliamente en un proyecto de ensamblado *de novo* incluir más de un tipo de librería y/o tecnología de secuenciación. Por ejemplo, con las librerías "mate-pair" en los datos de *Klebsiella* y *Leuconostoc* se logró reducir el número de contigs de forma considerable llegando a los 55 en *Klebsiella* y a los 6 en *Leuconostoc*. Por último se espera poder seguir desarrollando el paquete de *BacterialAnnotAid* como una herramienta auxiliar en estos casos.

9 Referencias

[1] National Research Council (US) Committee on Frontiers at the Interface of Computing and editors. Biology; Wooley JC, Lin HS. Catalyzing inquiry at the interface of computing and biology. washington (dc): National academies press (us). `http://www.ncbi.nlm.nih.gov/books/NBK25448/`, 2005.

[2] J Priem. Medline literature growth chart. `http://jasonpriem.org/2010/10/medline-literature-growth-chart/` [Extraído el 30 de Abril 2014]., 2010.

[3] Bob Kuska. Beer, bethesda, and biology: How "genomics" came into being. *J. Natl. Cancer. Inst.*, 90(2):93, 1998.

[4] McCray AT Lederberg J. 'ome sweet 'omics - a genealogical treasury of words genealogical treasury of words. *Scientist*, 15(7):8, 2001.

[5] James D. Watson and Francis H. Crick. Molecular structure of nucleic acids: A structure for deoxyribose nucleic acid. *Nature*, 171:737–738, 1953.

[6] W. Min Jou, G. Haegeman, M. Ysebaert, and W. Fiers. Nucleotide sequence of the gene coding for the bacteriophage MS2 coat protein. *Nature*, 237(5350):82–88, May 1972.

[7] N. M. Maizels. The nucleotide sequence of the lactose messenger ribonucleic acid transcribed from the UV5 promoter mutant of *Escherichia coli*. *Proc. Natl. Acad. Sci. U.S.A*, 70(12):3585–3589, December 1973.

[8] F. Sanger, G. M. Air, B. G. Barrell, N. L. Brown, A. R. Coulson, C. A. Fiddes, C. A. Hutchison, P. M. Slocombe, and M. Smith. Nucleotide sequence of bacteriophage phi X174 DNA. *Nature*, 265(5596):687–695, February 1977.

[9] R. D. Fleischmann, M. D. Adams, O. White, R. A. Clayton, E. F. Kirkness, A. R. Kerlavage, C. J. Bult, J. F. Tomb, B. A. Dougherty, and J. M. Merrick. Whole-genome random sequencing and assembly of *Haemophilus influenzae* Rd. *Science*, 269(5223):496–512, July 1995.

[10] J G Lawrence and H Ochman. Molecular archaeology of the *Escherichia coli* genome. *Proc. Natl. Acad. Sci. U.S.A*, 95(16):9413–7, 1998.

[11] G Gadaleta, G Pepe, G De Candia, C Quagliariello, E Sbisagrave;, and C Saccone. The complete nucleotide sequence of the *Rattus norvegicus* mitochondrial genome: cryptic signals revealed by comparative analysis between vertebrates. *J. Mol. Evol.*, 28(6):497–516, 1989.

[12] The Human Genome Project. Understanding out genetic inheritance. `http://www.genome.gov/10001477`[Extraído el 30 de Abirl 2014], 2012.

[13] International Human Genome Sequencing Consortium. Initial sequencing and analysis of the human genome. *Nature*, 409:860–921, 2001.

[14] J. C. Venter and Others. The sequence of the human genome. *Science*, 291(5507):1304–1351, 2001.

[15] F. Sanger, S. Nicklen, and A.R. Coulson. Dna sequencing with chain-terminating inhibitors. *Proc. Natl. Acad. Sci. U.S.A*, 74:5463–5467, 1977.

[16] A. Cann. *DNA viruses: a practical approach*. The practical approach series. New York, 1999.

[17] S. Anderson. Shotgun dna sequencing using cloned dnase i-generated fragments. *Nucleic Acids Res.*, 9(13):3015–3027, 1981.

[18] Al Edwards and C. Thomas Caskey. Closure strategies for random dna sequencing. *Methods*, 3(1):41 – 47, 1991.

[19] Robert H Waterston, Eric S Lander, and John E Sulston. On the sequencing of the human genome. *Proc. Natl. Acad. Sci. U.S.A*, 99(6):3712–6, 2002.

[20] Kris Wetterstrand. Dna sequencing costs: Data from the nhgri genome sequencing program(gsp). `www.genome.gov/sequencingcosts`[Extraído el 30 de Abril 2014].

[21] M Margulies, M Egholm, W E Altman, S Attiya, J S Bader, L A Bemben, J Berka, M S Braverman, Y J Chen, Z Chen, S B Dewell, L Du, J M Fierro, X V Gomes, B C Godwin, W He, S Helgesen, C H Ho, C H Ho, G P Irzyk, S C Jando, M L Alenquer, T P Jarvie, K B Jirage, J B Kim, J R Knight, J R Lanza, J H Leamon, S M Lefkowitz, M Lei, J Li, K L Lohman, H Lu, V B Makhijani, K E McDade, M P McKenna, E W Myers, E Nickerson, J R Nobile, R Plant, B P Puc, M T Ronan, G T Roth, G J Sarkis, J F Simons, J W Simpson, M Srinivasan, K R Tartaro, A Tomasz, K A Vogt, G A Volkmer, S H Wang, Y Wang, M P Weiner, P Yu, R F Begley, and J M Rothberg. Genome sequencing in microfabricated high-density picolitre reactors. *Nature*, 437(7057):376–380, September 2005.

[22] D R Bentley, S Balasubramanian, H P Swerdlow, G P Smith, J Milton, C G Brown, K P Hall, D J Evers, C L Barnes, H R Bignell, et al. Accurate whole human genome sequencing using reversible terminator chemistry. *Nature*, 456:53–59, 2008.

[23] Anton Valouev, Jeffrey Ichikawa, Thaisan Tonthat, Jeremy Stuart, Swati Ranade, Heather Peckham, Kathy Zeng, Joel A. Malek, Gina Costa, Kevin McKernan, Arend Sidow, Andrew Fire, and Steven M. Johnson. A high-resolution, nucleosome position map of *C. elegans* reveals a lack of universal sequence-dictated positioning. *Genome res.*, 18(7):1051–1063, July 2008.

[24] Pacific Biosciences. http://www.pacificbiosciences.com/products/smrt-technology/smrt-sequencing-advantage/.

[25] Richard Williams, Sergio G. Peisajovich, Oliver J. Miller, Shlomo Magdassi, Dan S. Tawfik, and Andrew D. Griffiths. Amplification of complex gene libraries by emulsion PCR. *Nat Meth*, 3(7):545–550, July 2006.

[26] P. Mayer, L. Farinelli, and E.H. Kawashima. Method of nucleic acid amplification, 2013. US Patent 8,476,044.

[27] Eric E. Schadt, Steve Turner, and Andrew Kasarskis. A window into third-generation sequencing. *Hum. Mol. Genet.*, 19(R2):R227–R240, 2010.

[28] Richard M. Durbin, Goncalo R. Abecasis, David L. Altshuler, Adam Auton, Lisa D. Brooks, Richard M. Durbin, Richard A. Gibbs, Matt E. Hurles, and Gil A. McVean. A map of human genome variation from population-scale sequencing. *Nature*, 467(7319):1061–1063, 2010.

[29] JF Gout, WK Thomas, Z Smith, K Okamoto, and M Lynch. Large-scale detection of in vivo transcription errors. *Proc. Natl. Acad. Sci. U.S.A.*, 110(46):18584–18589, November 2013.

[30] Zhenyu Li, Yanxiang Chen, Desheng Mu, Jianying Yuan, Yujian Shi, Hao Zhang, Jun Gan, Nan Li, Xuesong Hu, Binghang Liu, et al. Comparison of the two major classes of assembly algorithms: overlap–layout–consensus and de-bruijn-graph. *Brief. Funct. Genomics*, 11(1):25–37, 2012.

[31] Paul Havlak, Rui Chen, K James Durbin, Amy Egan, Yanru Ren, Xing-Zhi Song, George M Weinstock, and Richard A Gibbs. The atlas genome assembly system. *Genome Res*, 14(4):721–32, April 2004.

[32] S Batzoglou, D Jaffe, K Stanley, J Butler, S Gnerre, E Mauceli, B Berger, J Mesirov, and E Lander. Arachne: a whole-genome shotgun assembler. *Genome Res*, 12:177, 2002.

[33] E W Myers, G G Sutton, A L Delcher, I M Dew, D P Fasulo, M J Flanigan, S A Kravitz, C M Mobarry, K H Reinert, K A Remington, E L Anson, R A Bolanos, H H Chou, C M Jordan, A L Halpern, S Lonardi, E M Beasley, R C Brandon, L Chen, P J Dunn, Z Lai, Y Liang, D R Nusskern, M Zhan, Q Zhang, X Zheng, G M Rubin, M D Adams, and J C Venter. A whole-genome assembly of drosophila. *Science*, 287(5461):2196–204, March 2000.

[34] Xiaoqiu Huang, Jianmin Wang, Srinivas Aluru, Shiaw-Pyng Yang, and LaDeana Hillier. Pcap: a whole-genome assembly program. *Genome Res.*, 13(9):2164–70, September 2003.

[35] James C Mullikin and Zemin Ning. The phusion assembler. *Genome Res.*, 13(1):81–90, January 2003.

[36] Jennifer Commins, Christina Toft, and Mario A. Fares. Computational biology methods and their application to the comparative genomics of endocellular symbiotic bacteria of insects. *Biol. Proced. Online*, 11, December 2009.

[37] D Zerbino and E Birney. Velvet: algorithms for de novo short read assembly using de bruijn graphs. *Genome Res*, 18:821, 2008.

[38] P Pevzner, H Tang, and M Waterman. An eulerian path approach to dna fragment assembly. *Proc Natl Acad Sci USA*, 98:9748, 2001.

[39] J Simpson, K Wong, S Jackman, J Schein, S Jones, and I Birol. Abyss: a parallel assembler for short read sequence data. *Genome Res*, 19:1117, 2009.

[40] Sébastien Boisvert, François Laviolette, and Jacques Corbeil. Ray: Simultaneous assembly of reads from a mix of high-throughput sequencing technologies. *J. Comp. Biol.*, 17(11):1519–1533, 2010.

[41] Jared T. Simpson and Richard Durbin. Efficient construction of an assembly string graph using the fm-index. *Bioinformatics*, 26(12):367–373, 2010.

[42] Sante Gnerre, Iain Maccallum, Dariusz Przybylski, Filipe J Ribeiro, Joshua N Burton, Bruce J Walker, Ted Sharpe, Giles Hall, Terrance P Shea, Sean Sykes,

Aaron M Berlin, Daniel Aird, Maura Costello, Riza Daza, Louise Williams, Robert Nicol, Andreas Gnirke, Chad Nusbaum, Eric S Lander, and David B Jaffe. High-quality draft assemblies of mammalian genomes from massively parallel sequence data. *Proc Natl Acad Sci U S A*, 108(4):1513–1518, 2011.

[43] P.E.C. Compeau, P.A. Pevzner, and G. Tesler. How to apply de bruijn graphs to genome assembly. *Nat. Biotechnol.*, 29(11):987–991, 2011.

[44] OmicsMaps. `http://omicsmaps.com/`.

[45] D R Bentley, S Balasubramanian, H P Swerdlow, G P Smith, J Milton, C G Brown, K P Hall, D J Evers, C L Barnes, H R Bignell, et al. Accurate whole human genome sequencing using reversible terminator chemistry. *Nature*, 456:53–59, 2008.

[46] Illumina Inc. Nextera dna sample preparation kit. `http://www.illumina.com/products/nextera_dna_sample_prep_kit.ilmn` [Extraído el 04 de Mayo del 2014], 2013.

[47] Illumina Inc. Illumina sequencing technology [vídeo]. `https://www.youtube.com/watch?v=womKfikWlxM` [Extraído el 03 de Mayo del 2014], 2013.

[48] Peter J A Cock, Christopher J Fields, Naohisa Goto, Michael L Heuer, and Peter M Rice. The Sanger FASTQ file format for sequences with quality scores, and the Solexa/Illumina FASTQ variants. *Nucleic Acids Res*, 38(6):1767–1771, 2010.

[49] M Kircher, U Stenzel, and J Kelso. Improved base calling for the illumina genome analyzer using machine learning strategies. *Genome Biol*, 10:R83, 2009.

[50] Simon Andrews. Fastqc. `http://www.bioinformatics.babraham.ac.uk/projects/fastqc/`.

[51] Robert C Gentleman, Vincent J. Carey, Douglas M. Bates, Ben Bolstad, Marcel Dettling, Sandrine Dudoit, Byron Ellis, Laurent Gautier, Yongchao

Ge, Jeff Gentry, Kurt Hornik, Torsten Hothorn, Wolfgang Huber, Stefano Iacus, Rafael Irizarry, Friedrich Leisch Cheng Li, Martin Maechler, Anthony J. Rossini, Gunther Sawitzki, Colin Smith, Gordon Smyth, Luke Tierney, Jean Y. H. Yang, and Jianhua Zhang. Bioconductor: Open software development for computational biology and bioinformatics. *Genome Biol*, 5:R80, 2004.

[52] R Development Core Team. *R: A language and environment for statistical computing*. R Foundation for Statistical Computing, Vienna, Austria, 2005. ISBN 3-900051-07-0.

[53] JC Dohm, C Lottaz, T Borodina, and H Himmelbauer. Substantial biases in ultra-short read data sets from high-throughput dna sequencing. *Nucleic Acids Res*, 36:e105+, 2008.

[54] B Ewing, L Hillier, MC Wendl, and P Green. Base-calling of automated sequencer traces using Phred. I. Accuracy assessment. *Genome Res.*, 8(3):175–185, March 1998.

[55] Brent Ewing and Phil Green. Base-Calling of Automated Sequencer Traces Using Phred. II. Error Probabilities. *Genome Res*, 8(3):186–194, March 1998.

[56] Cristian Del Fabbro, Simone Scalabrin, Michele Morgante, and Federico M. Giorgi. An Extensive Evaluation of Read Trimming Effects on Illumina NGS Data Analysis. *PLoS ONE*, 8(12):e85024+, December 2013.

[57] Xiaoqing Yu, Kishore Guda, Joseph Willis, Martina Veigl, Zhenghe Wang, Sanford Markowitz, Mark D. Adams, and Shuying Sun. How do alignment programs perform on sequencing data with varying qualities and from repetitive regions? *BioData Mining*, 5:6, 2012.

[58] James Robert White, Michael Roberts, James A. Yorke, and Mihai Pop. Figaro: a novel statistical method for vector sequence removal. *Bioinformatics*, 24(4):462–467, 2008.

[59] Nick Loman. Tips for de novo bacterial genome assembly. `http://pathogenomics.bham.ac.uk/blog/2009/09/`

tips-for-de-novo-bacterial-genome-assembly/ [Extraído el 03 de Mayo del 2014], 2009.

[60] Camila Valenzuela, Juan A Ugalde, Guido C Mora, and Carlos A Santiviago. Draft Genome Sequence of *Salmonella enterica* Serovar Typhi Strain. *Genome Announc.*, 2(1):2–3, 2014.

[61] Dali Liu, Yanbin Zhou, Mariko Naito, Hiromichi Yumoto, Qiang Li, Yoichiro Miyake, Jingping Liang, and Rong Shu. Draft Genome Sequence of *Porphyromonas gingivalis* Strain SJD2 , Isolated from the Periodontal Pocket of a Patient with Periodontitis in. *Genome Announc.*, 2(30901692):1–2, 2014.

[62] Samuel S Hunter, Hirokazu Yano, Wesley Loftie-eaton, Julie Hughes, Leen De Gelder, Pieter Stragier, and Paul De Vos. Draft Genome Sequence of *Pseudomonas moraviensis* R28-S. *Genome Announc.*, 2(1):153–154, 2014.

[63] Donald M Gardiner, Jiri Stiller, and Kemal Kazan. Genome Sequence of *Fusarium graminearum* Isolate CS3005. *Genome Announc.*, 2(2):4–5, 2014.

[64] Alexis Criscuolo and De Blanchardière. Draft Genome Sequence of *Campylobacter coli* Strain IPSID-1 Isolated from a Patient with Immunoproliferative Small Intestinal Disease. *Genome Announc.*, 2(2):3–4, 2014.

[65] Jayanta D Choudhury, Arnab Pramanik, Nicole S Webster, Lyndon E Llewellyn, and Ratan Gachhui. Draft Genome Sequence of *Pseudoalteromonas sp* . Strain NW 4327 (MTCC 11073 , DSM 25418), a Pathogen of the Great Barrier Reef Sponge Rhopaloeides odorabile. *Genome Announc.*, 2(1):2007–2008, 2014.

[66] Livio Antonielli, Joseph Strauss, Angela Sessitsch, and Harald Berger. Draft Genome Sequence of *Phaeomoniella chlamydospora* Strain RR- HG1 , a Grapevine Trunk Disease (Esca) -Related Member of the. *Genome Announc.*, 2(2):2013–2014, 2014.

[67] Hannon Lab. Fastx. http://hannonlab.cshl.edu/fastx_toolkit/index.html.

[68] Alexey A. Gurevich, Vladislav Saveliev, Nikolay Vyahhi, and Glenn Tesler. Quast: quality assessment tool for genome assemblies. *Bioinformatics*, 29(8):1072–1075, 2013.

[69] Wenyu Zhang, Jiajia Chen, Yang Yang, Yifei Tang, Jing Shang, and Bairong Shen. A Practical Comparison of De Novo Genome Assembly Software Tools for Next-Generation Sequencing Technologies. *PLoS ONE*, 6(3):e17915+, March 2011.

[70] Dent A. Earl, Keith Bradnam, John St. John, Aaron Darling, Dawei Lin, Joseph Faas, Hung On Ken Yu, Buffalo Vince, Daniel R. Zerbino, Mark Diekhans, Ngan Nguyen, Pramila Nuwantha, Ariyaratne Wing-Kin Sung, Zemin Ning, Matthias Haimel, Jared T. Simpson, Nuno A. Fronseca, Inanç Birol, T. Roderick Docking, Isaac Y. Ho, Daniel S Rokhsar, Rayan Chikhi, Dominique Lavenier, Guillaume Chapuis, Delphine Naquin, Nicolas Maillet, Michael C. Schatz, David R. Kelly, Adam M. Phillippy, Sergey Koren, Shiaw-Pyng Yang, Wei Wu, Wen-Chi Chou, Anuj Srivastava, Timothy I. Shaw, J. Graham Ruby, Peter Skewes-Cox, Miguel Betegon, Michelle T. Dimon, Victor Solovyev, Petr Kosarev, Denis Vorobyev, Ricardo Ramirez-Gonzalez, Richard Leggett, Dan Maclean, Fangfang Xia, Ruibang Luo, Zhenyu Li, Yinlong Xie, Binghang Liu, Sante Gnerre, Iain Maccallum, Dariusz Przybylski, Filipe J. Ribeiro, Shuangye Yin, Ted Sharpe, Giles Hall, Paul J. Kersey, Richard Durbin, Shaun D. Jackman, Jarrod A. Chapman, Xiaoqiu Huang, Joseph L. Derisi, Mario Caccamo, Yingrui Li, David B. Jaffe, M. Green, Richard, David Haussler, Ian Korf, and Benedict Paten. Assemblathon 1: A competitive assessment of de novo short read assembly methods. *Genome Res.*, September 2011. International competition of de novo genome assembly. The Symbiose team (IRISA/CNRS/ENS Cachan Brittany) participated to this competition.

[71] S. L. Salzberg, A. M. Phillippy, A. Zimin, D. Puiu, T. Magoc, S. Koren, T. J. Treangen, M. C. Schatz, A. L. Delcher, M. Roberts, et al. Gage: A critical

evaluation of genome assemblies and assembly algorithms. *Genome Res.*, 22, 2012.

[72] Keith R. Bradnam, Joseph N. Fass, Anton Alexandrov, Paul Baranay, Michael Bechner, Inanc Birol, Sébastien Boisvert, Jarrod A. Chapman, Guillaume Chapuis, Rayan Chikhi, Hamidreza Chitsaz, Wen-Chi Chou, Jacques Corbeil, Cristian Del Fabbro, T. Roderick Docking, Richard Durbin, Dent Earl, Scott Emrich, Pavel Fedotov, Nuno A. Fonseca, Ganeshkumar Ganapathy, Richard A. Gibbs, Sante Gnerre, Elénie Godzaridis, Steve Goldstein, Matthias Haimel, Giles Hall, David Haussler, Joseph B. Hiatt, Isaac Y. Ho, Jason Howard, Martin Hunt, Shaun D Jackman, David B. Jaffe, Erich Jarvis, Huaiyang Jiang, and et al. Assemblathon 2: evaluating de novo methods of genome assembly in three vertebrate species. *GigaScience*, 2(10), July 2013.

[73] C Titus Brown, Adina Howe, Qingpeng Zhang, Alexis B Pyrkosz, and Timothy H Brom. A Reference-Free Algorithm for Computational Normalization of Shotgun Sequencing Data. 2012.

[74] David Kelley, Michael Schatz, and Steven Salzberg. Quake: quality-aware detection and correction of sequencing errors. *Genome Biol.*, 11(11):R116, 2010.

[75] Wei-chun Kao. *Algorithms for Next-Generation High-Throughput Sequencing Technologies*. PhD thesis, UC, Berkeley, 2011.

[76] Isheng Tsai, Thomas Otto, and Matthew Berriman. Improving draft assemblies by iterative mapping and assembly of short reads to eliminate gaps. *Genome Biol*, 11(4):R41, 2010.

[77] Thomas D. Otto, Mandy Sanders, Matthew Berriman, and Chris Newbold. Iterative correction of reference nucleotides (icorn) using second generation sequencing technology. *Bioinformatics*, 26(14):1704–1707, 2010.

[78] Unidad Universitaria de Secuenciación Masiva de DNA. `http://www.uusmd.unam.mx/`.

[79] Keith Bradnam. Why is n50 used as an assembly metric? http://www.acgt.me/blog/2013/7/8/why-is-n50-used-as-an-assembly-metric.html [Extraído el 06 de Mayo del 2014], 2013.

[80] Babraham Bioinformatics. http://www.bioinformatics.babraham.ac.uk/projects/sherman/.

[81] SRA. http://www.ncbi.nlm.nih.gov/sra.

[82] ENA. http://www.ebi.ac.uk/ena/.

[83] DDBJ. http://trace.ddbj.nig.ac.jp/dra/index.html.

[84] W J Kent. BLAT–the BLAST-like alignment tool. *Genome Res*, 12(4):656–664, 2002.

[85] Steven L. Salzberg, Arthur L. Delcher, Simon Kasif, and Owen White. Microbial gene identification using interpolated markov models. *Nucleic Acids Res*, 26:544–548, 1998.

[86] Martin Hunt, Taisei Kikuchi, Mandy Sanders, Chris Newbold, Matthew Berriman, and Thomas Otto. Reapr: a universal tool for genome assembly evaluation. *Genome Biol*, 14(5):R47, 2013.

[87] Broad Institute. http://www.broadinstitute.org/software/pilon/.

[88] Victorian Bioinformatics Consortium. http://www.vicbioinformatics.com/software.nesoni.shtml.