Nicolás Gustavo Gaitán Gomez - 201727889
Sebastián Lemus Cadena - 201716495

Algoritmos en Biología Computacional: Tarea 2

Ensamblaje de Genomas

Introducción

El presente documento es un reporte de resultados relacionado con las diferentes salidas del programa ubicado en la carpeta “Reads Analyzer”, adjunta a este documento. El programa permite realizar **conteo de k-mers** de lecturas en archivos fastq, realizar un **ensamblaje de genomas** simple a partir de la metodología “Overlap Layout Consensus” y **simular lecturas** a partir de secuencias previamente ensambladas en archivos fasta. Por cada una de estas funcionalidades se responden las preguntas planteadas en la guía de trabajo

Reporte de Resultados

Distribución de K-mers

1. Probar con cada fastq de ejemplo tamaños de k-mer 5, 10, 20, 50 y 75 y registrar la cantidad de k-mers, la abundancia media y el tiempo de ejecución.

Nota: todas las distribuciones de k-mers se pueden visualizar en este link: https://drive.google.com/drive/folders/14Y1suTveeQOYu06GYcKuupImjqsuUiwK?usp=sharing
Cada archivo está rotulado igual que cada archivo de prueba, pero con un prefijo indicando el tamaño de k-mer. Por ejemplo el archivo “test20x-50” es la distribución de k-mers de las lecturas en la prueba “test20x” con un tamaño de kmer igual a 50.

En la siguiente tabla se registra la abundancia media, el tiempo de ejecución y la moda de repeticiones de k-mer por cada archivo de prueba y tamaño de k-mer. La abundancia media corresponde a la media aritmética de las frecuencias encontradas para los k-mers y la moda es el valor más alto de repetición de k-mers. La media es calculada como:

$$totalRepetitions = \sum_{i=1}^{maxFrequency} (i * numKmers_i)$$

$$totalKmers = \sum_{i=1}^{maxFrequency} numKmers_i$$

$$meanAbundance = totalRepetitions / totalKmers$$

Donde *maxFrequency* es el número de repetición máximo de kmers y $numKmers_i$ el número de k-mers que se repitieron *i* veces.

Archivo	Tamaño de k-mer	Abundancia media	Moda de k-mers	Tiempo de ejecución (ms)
test10x	5	13.3829	237	61
	10	10.0998	385	59
	20	8.8107	422	59
	50	5.5941	543	59
	75	3.0600	575	61
test20x	5	26.4584	163	74
	10	19.9197	259	68
	20	17.3819	297	71
	50	11.035	317	72
	75	5.6996	469	66
test50x	5	66.2069	126	88
	10	49.8357	192	90
	20	43.4860	199	86
	50	27.6073	228	80
	75	14.2025	294	78
test100x	5	132.17072	85	105
	10	99.5260	130	112
	20	86.8477	138	109
	50	55.1351	143	106
	75	28.3636	200	98

Como se puede observar, para cada archivo de prueba la moda es directamente proporcional al tamaño de los k-mers, mientras que la abundancia media es inversamente proporcional con respecto a los mismos. Por otro lado los valores de tiempo de ejecución no fluctúan mucho , aunque puede que si se aumenta más el tamaño de los k-mers, el tiempo de ejecución aumente notablemente. Las siguientes gráficas muestran estos dos comportamientos para la prueba “test50x”:

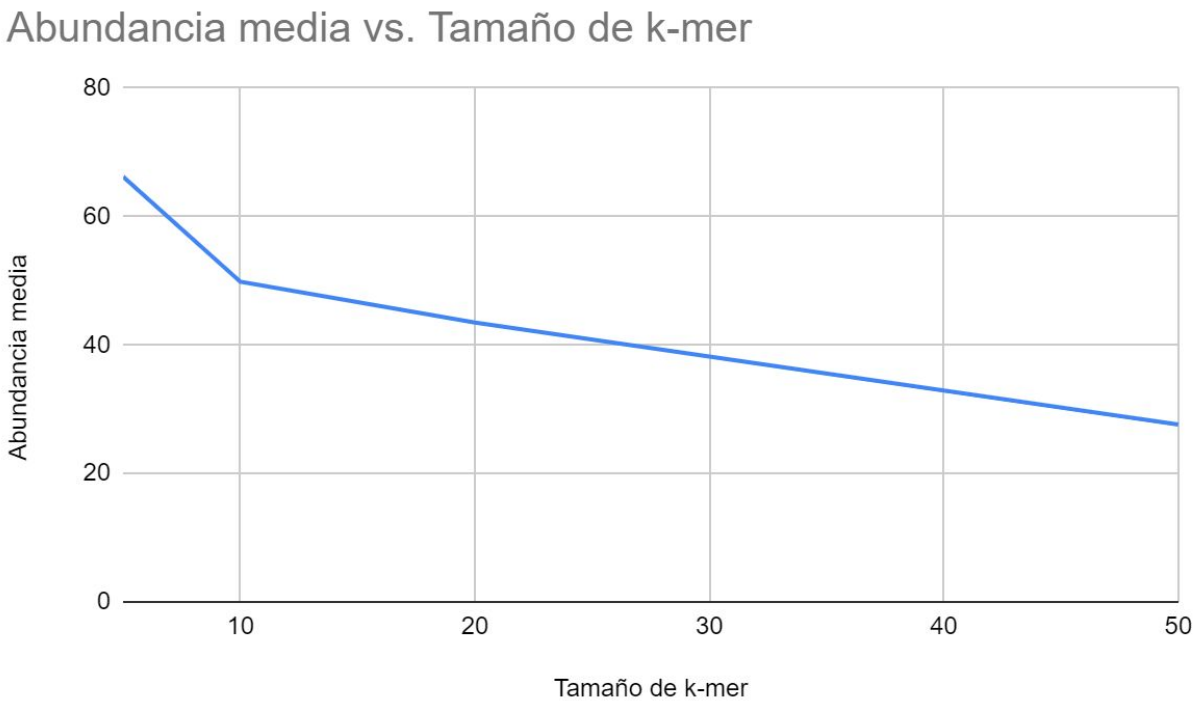

Figura 1: Abundancia media con respecto al tamaño de k-mer

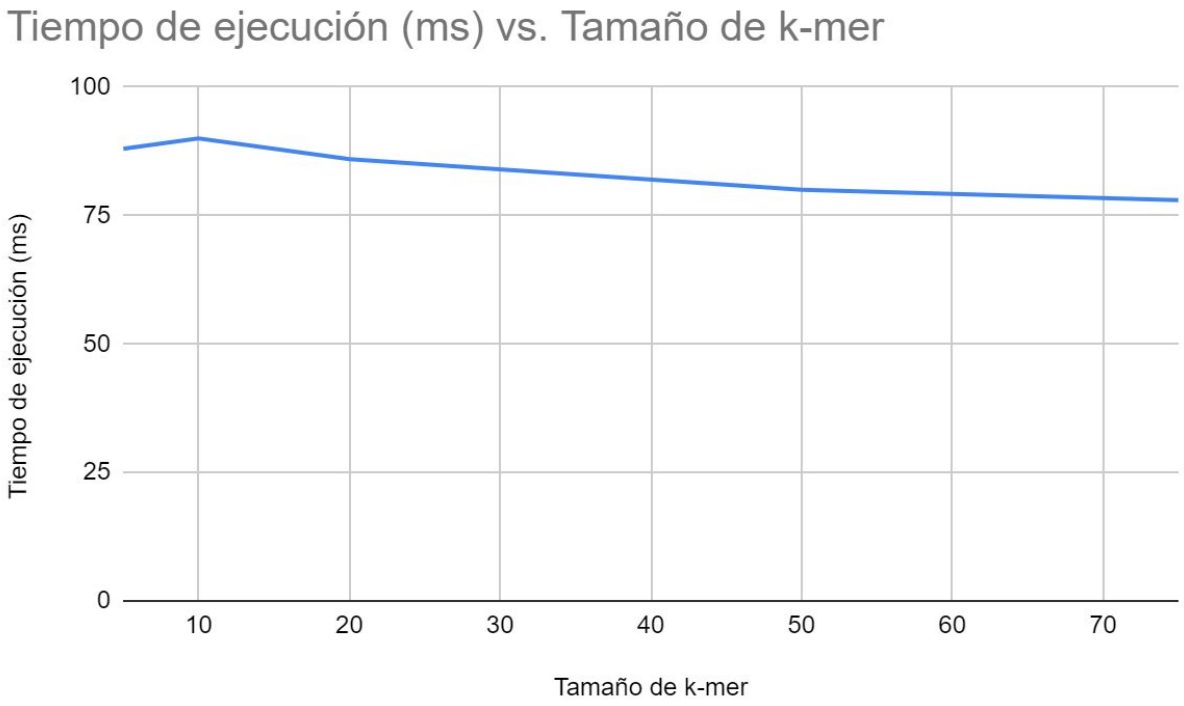

Figura 2: Tiempo de ejecución con respecto al tamaño de k-mer

Otro aspecto que se puede deducir es que el valor **máximo** de **abundancia media** aumenta conforme hay más cobertura entre las pruebas (denotada por el prefijo del nombre del archivo). Adicionalmente, graficando las frecuencias de k-mers, se pueden ver figuras similares a una distribución normal, cuya media ("pico" de la gráfica) corresponde a la **abundancia media**. También, como resultado de la tendencia descendente de la media para cada prueba, a medida que se aumenta el tamaño del k-mer, el centro de la gráfica se sesga hacia la izquierda. Como ejemplo, a continuación se muestran los gráficos de frecuencia para "test20x":

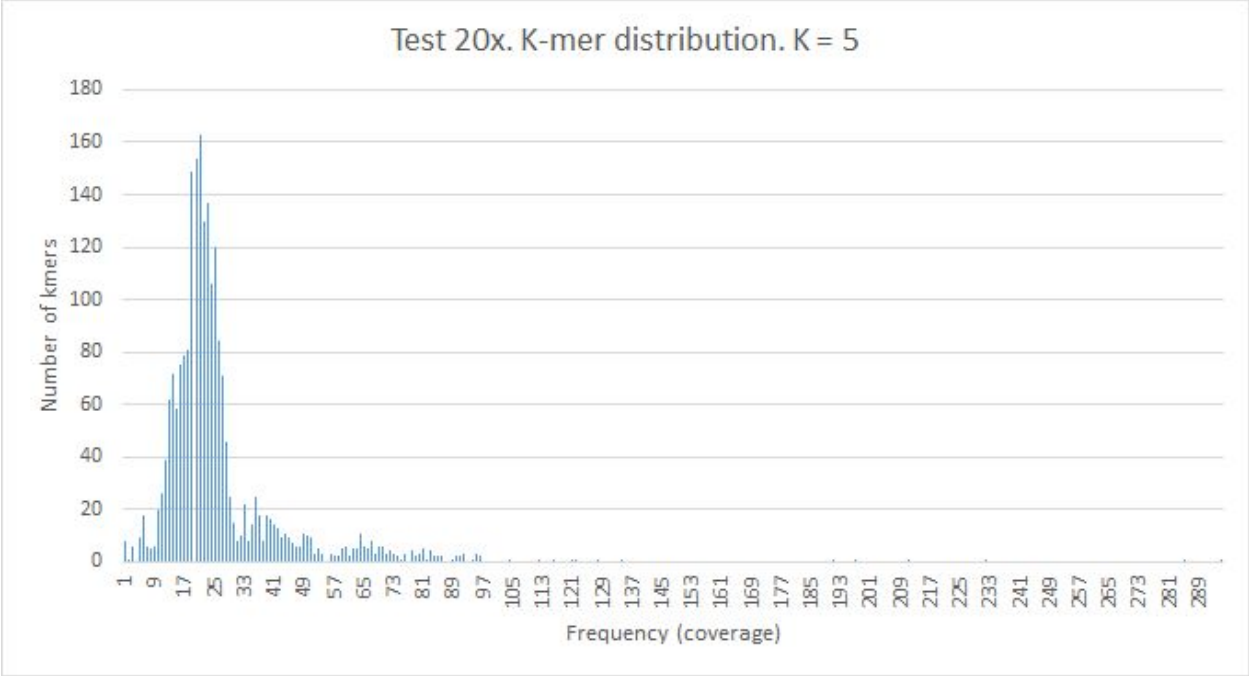

Figura 3: Distribución de k-mers para "test20x" y K = 5

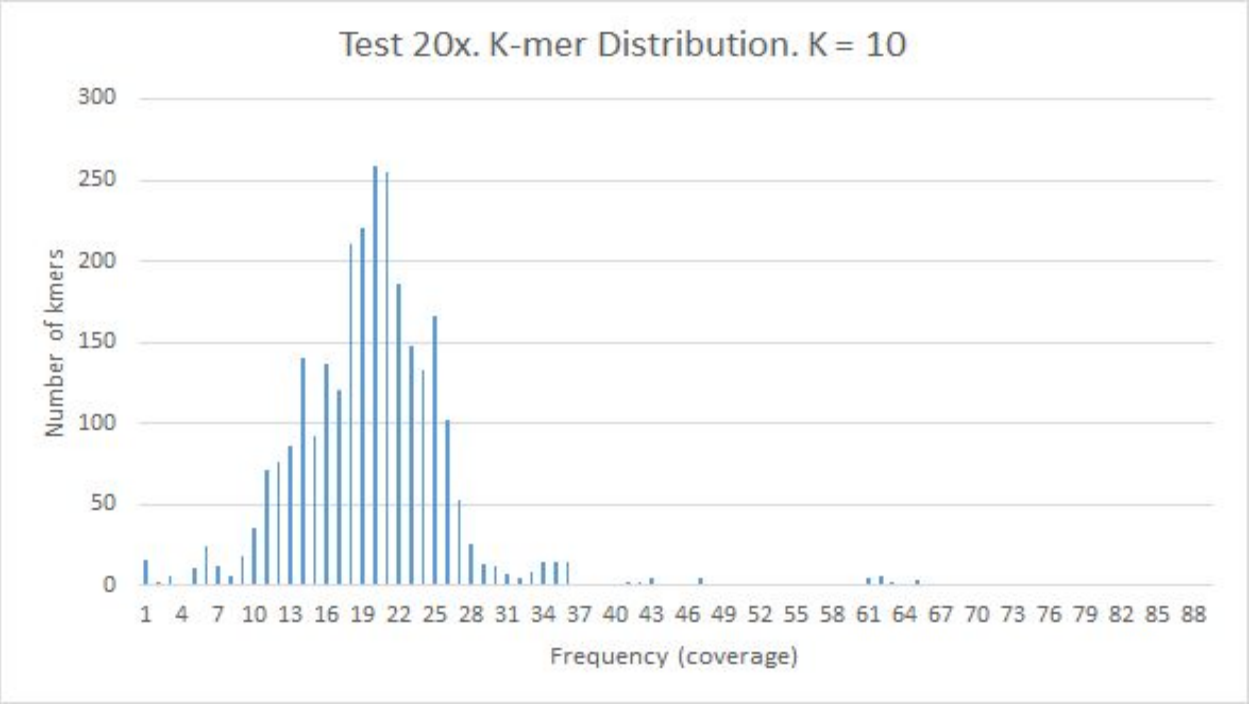

Figura 4: Distribución de k-mers para “test20x” y K = 10

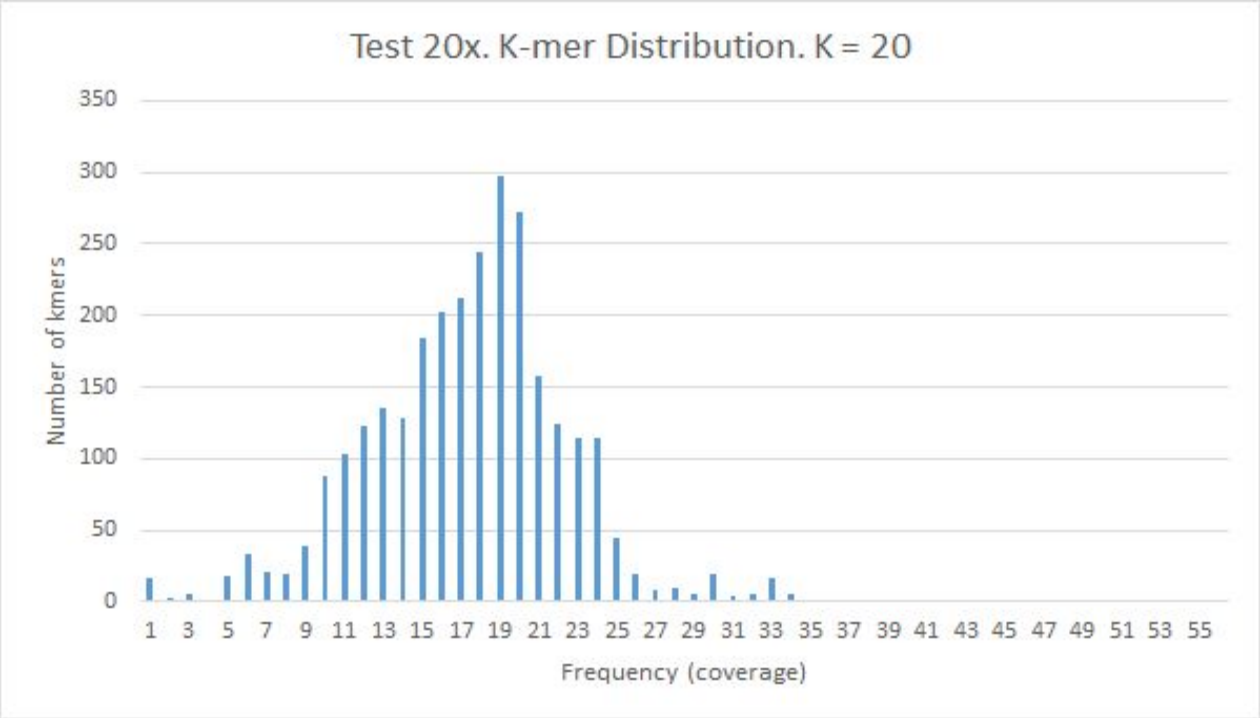

Figura 5: Distribución de k-mers para “test20x” y K = 20

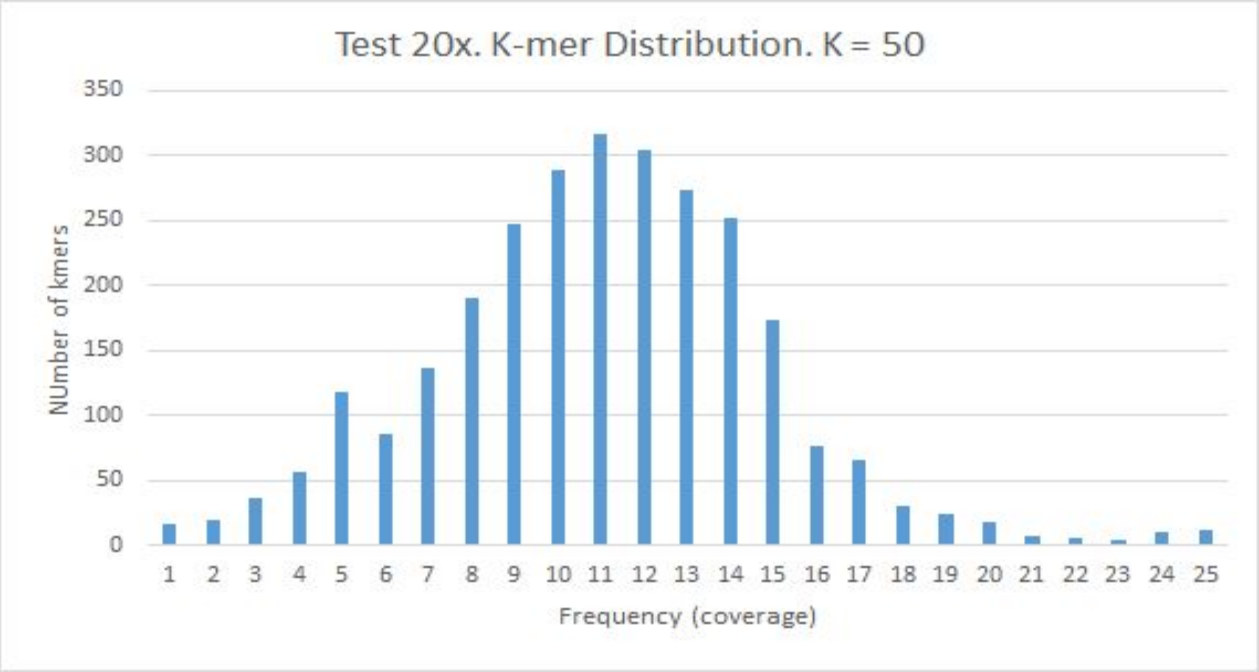

Figura 6: Distribución de k-mers para “test20x” y K = 50

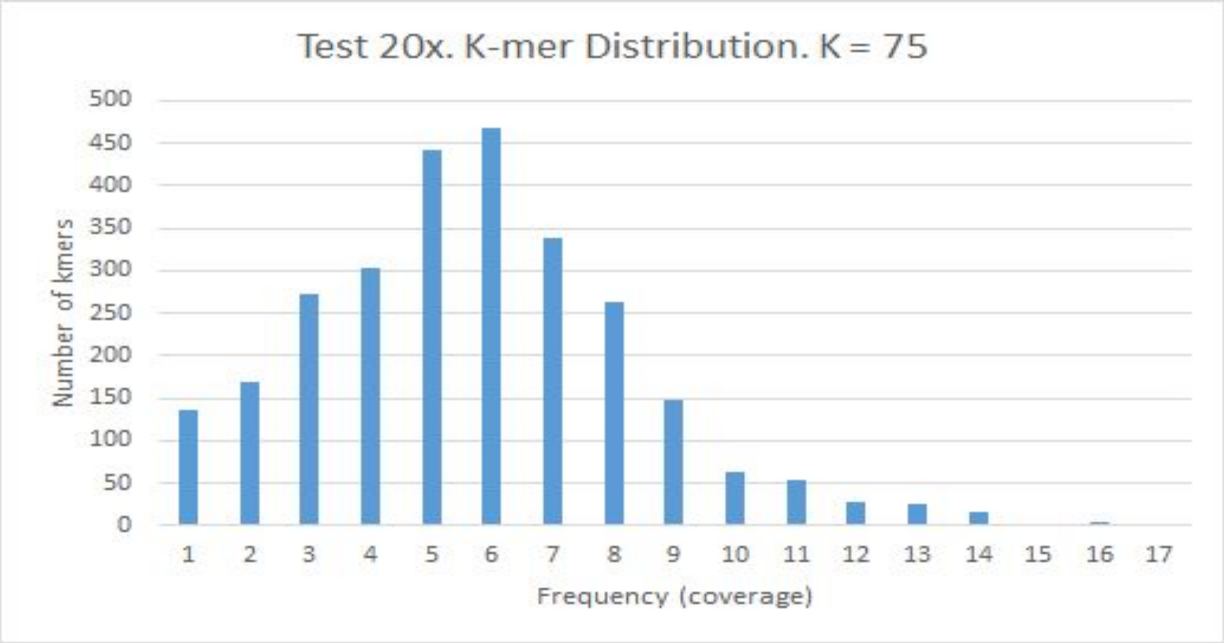

Figura 7: Distribución de k-mers para “test20x” y K = 75

En general, la distribución de k-mers se puede estimar mejor como una distribución Poisson que a una distribución normal (University of Connecticut, 2019).

Overlap Layout Consensus

2. Reportar para cada fastq de ejemplo disponible en la carpeta “data” la distribución de abundancias de secuencias, la distribución de sucesores por secuencia y el tiempo de ejecución.

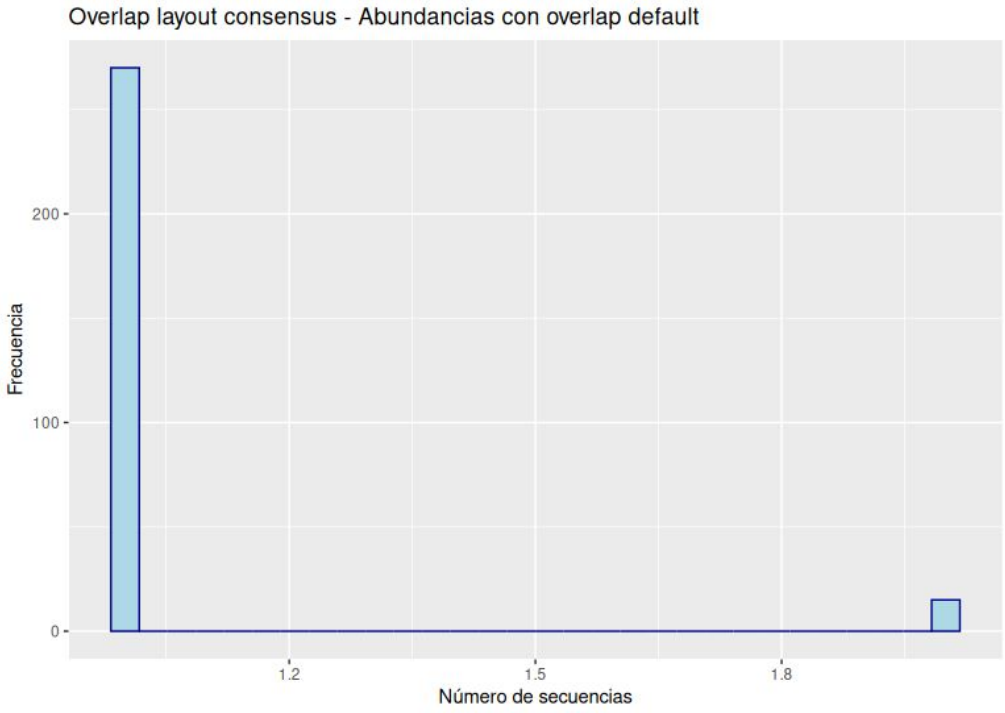

Figura 8: Distribución de abundancia de secuencias para “test10x.fastq”

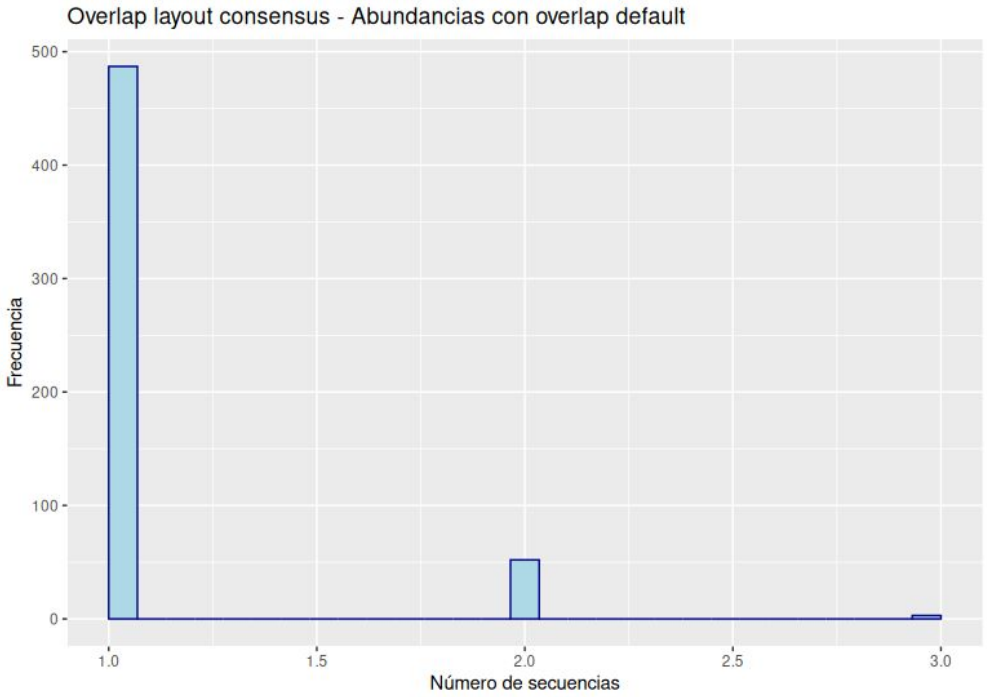

Figura 9: Distribución de abundancia de secuencias para “test20x.fastq”

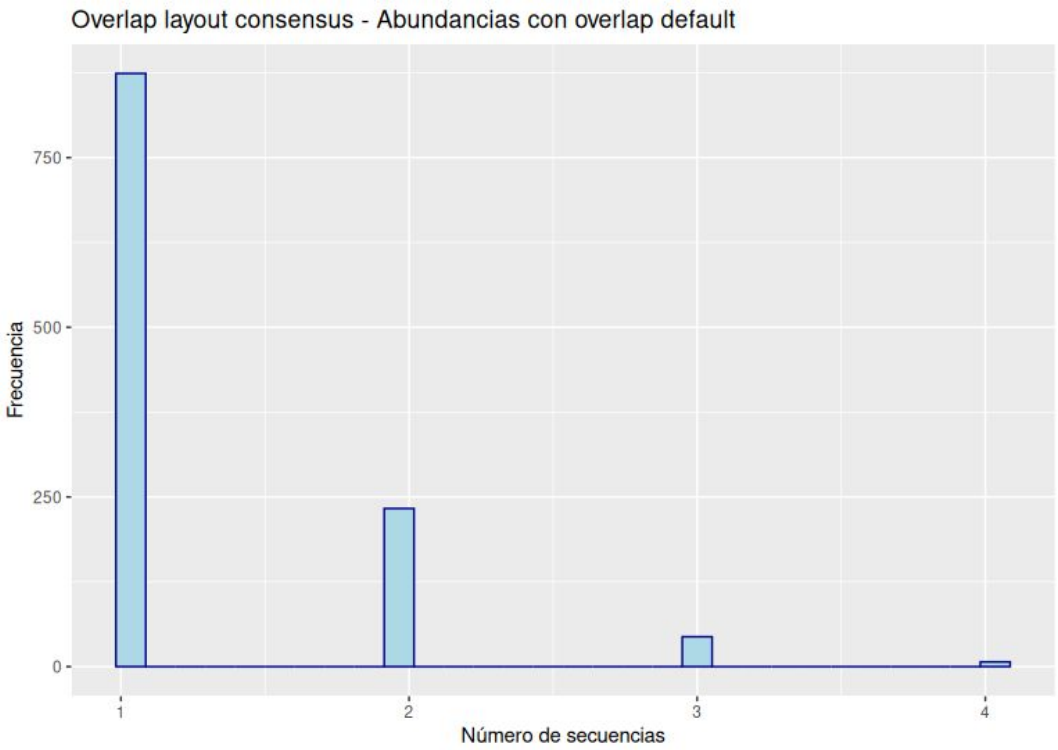

Figura 10: Distribución de abundancia de secuencias para “test50x.fastq”

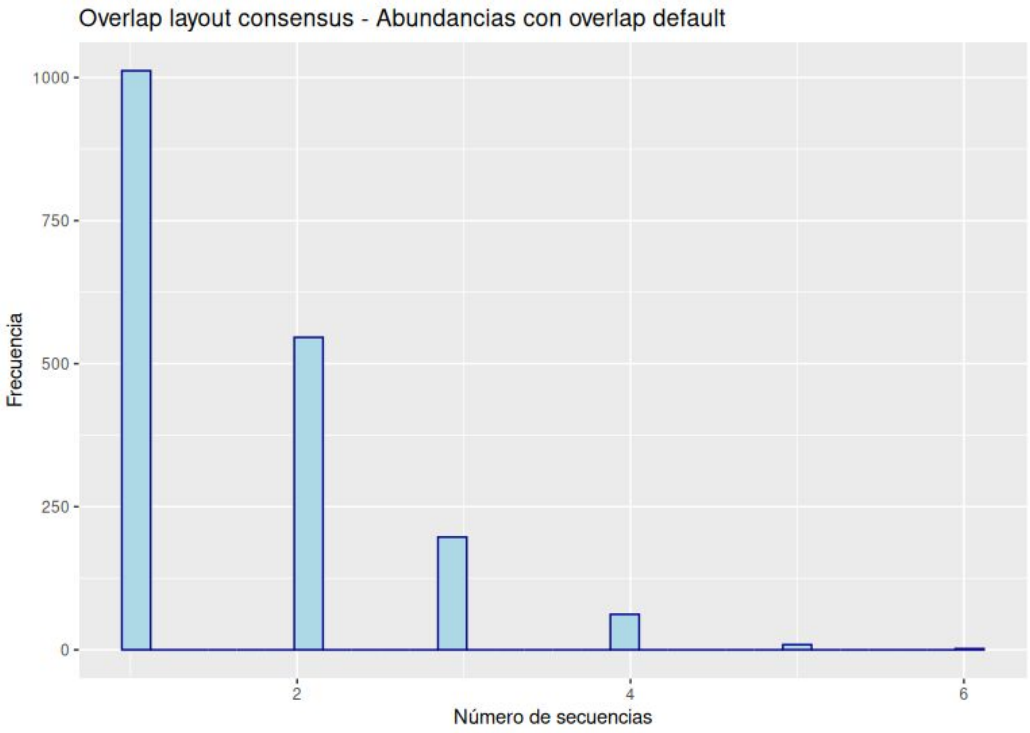

Figura 10: Distribución de abundancia de secuencias para “test100x.fastq”

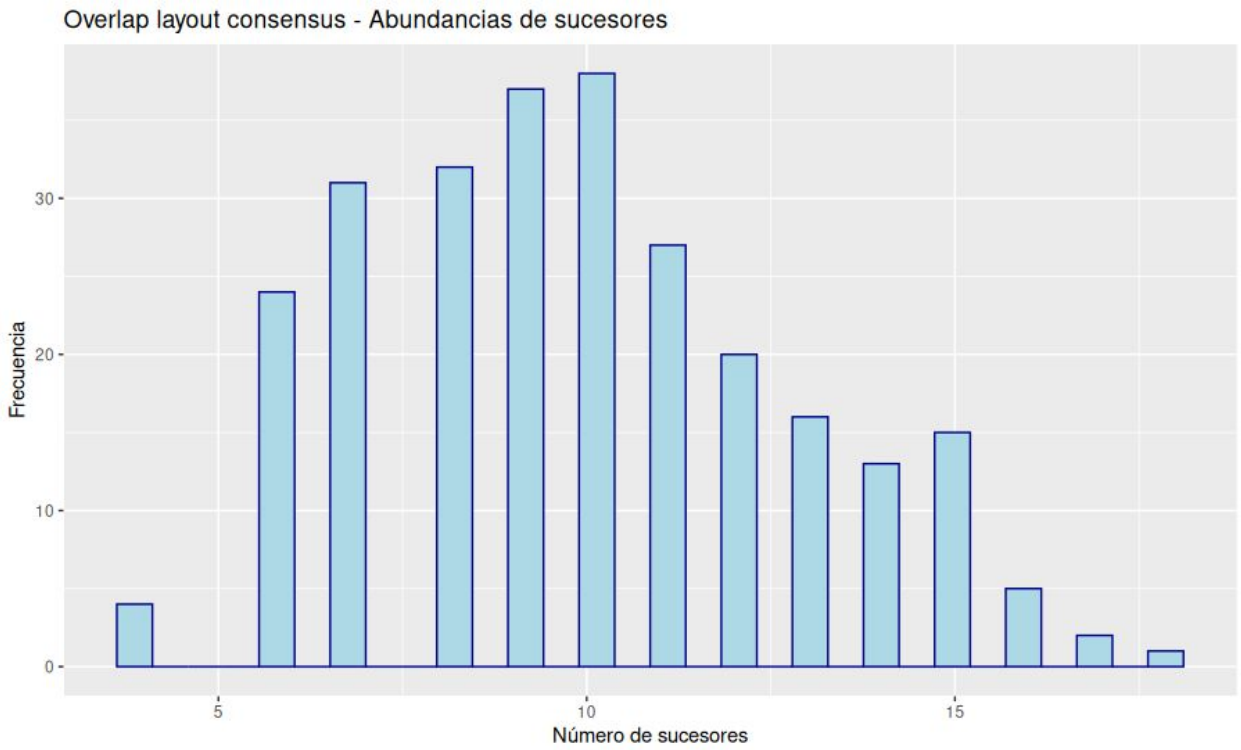

Figura 11: Distribución de abundancia de sucesores para “test10x.fastq”

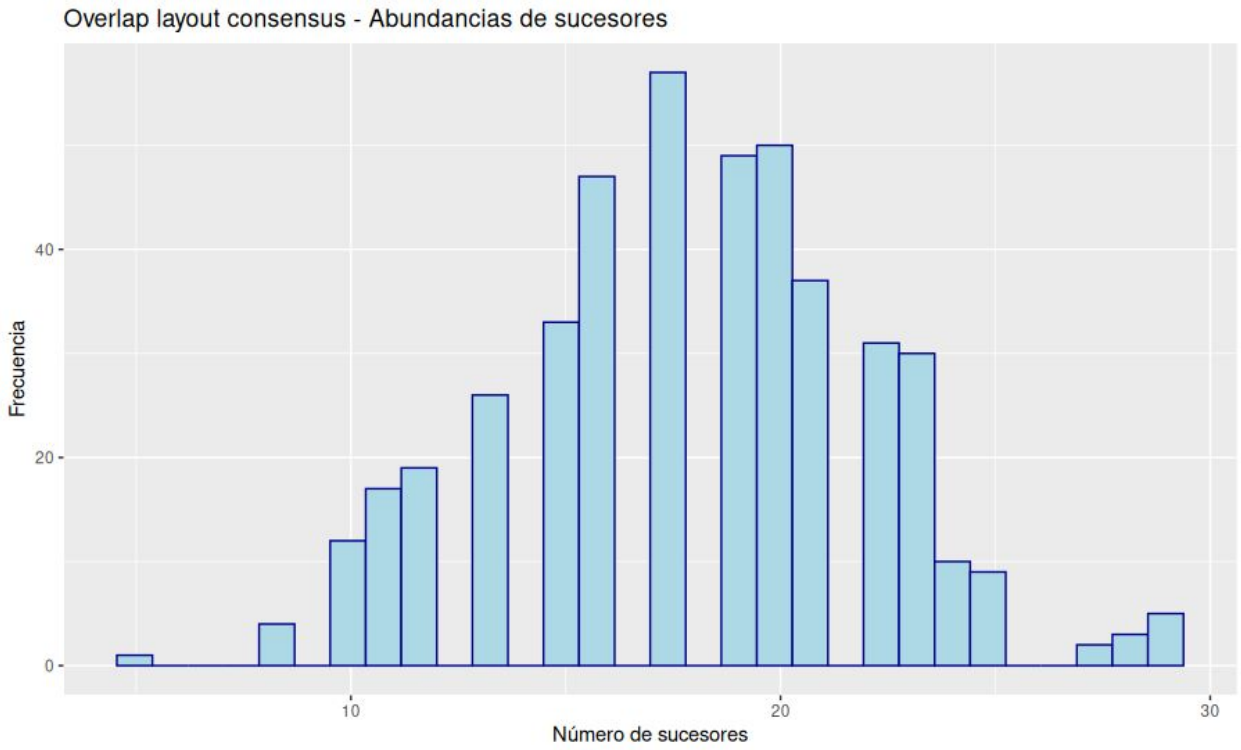

Figura 12: Distribución de abundancia de sucesores para “test20x.fastq”

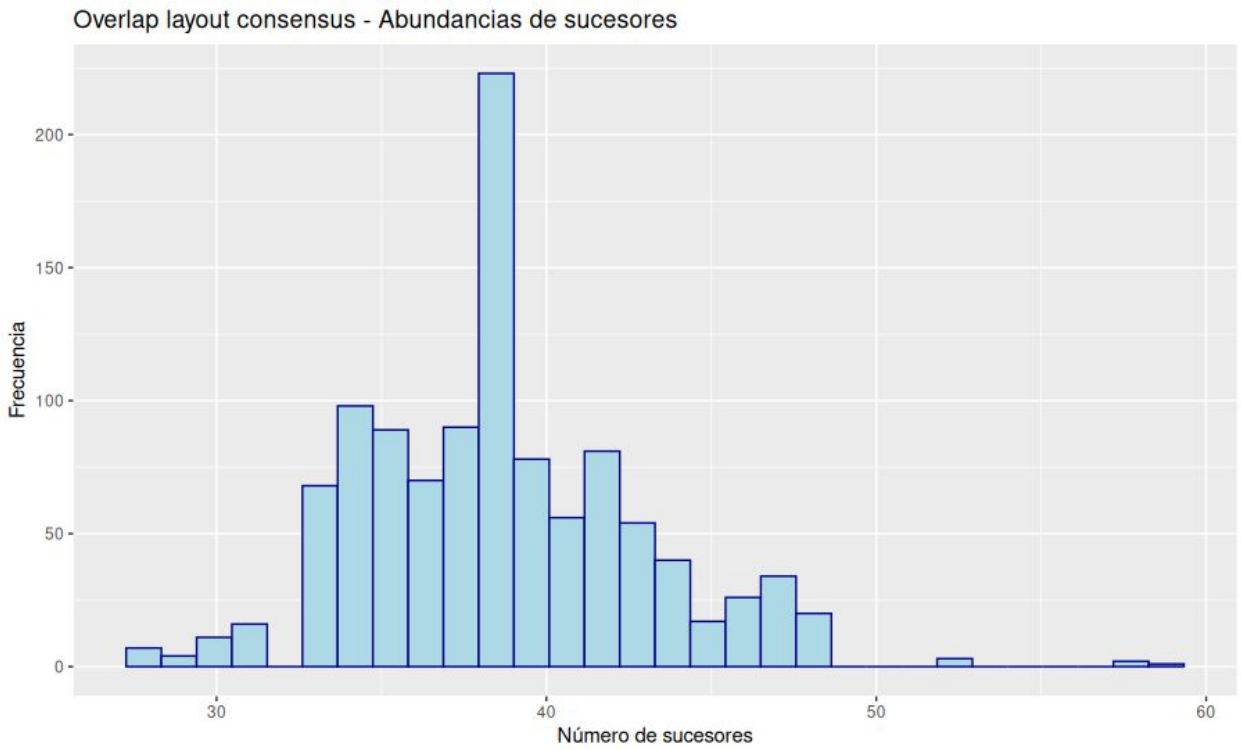

Figura 13: Distribución de abundancia de sucesores para “test50x.fastq”

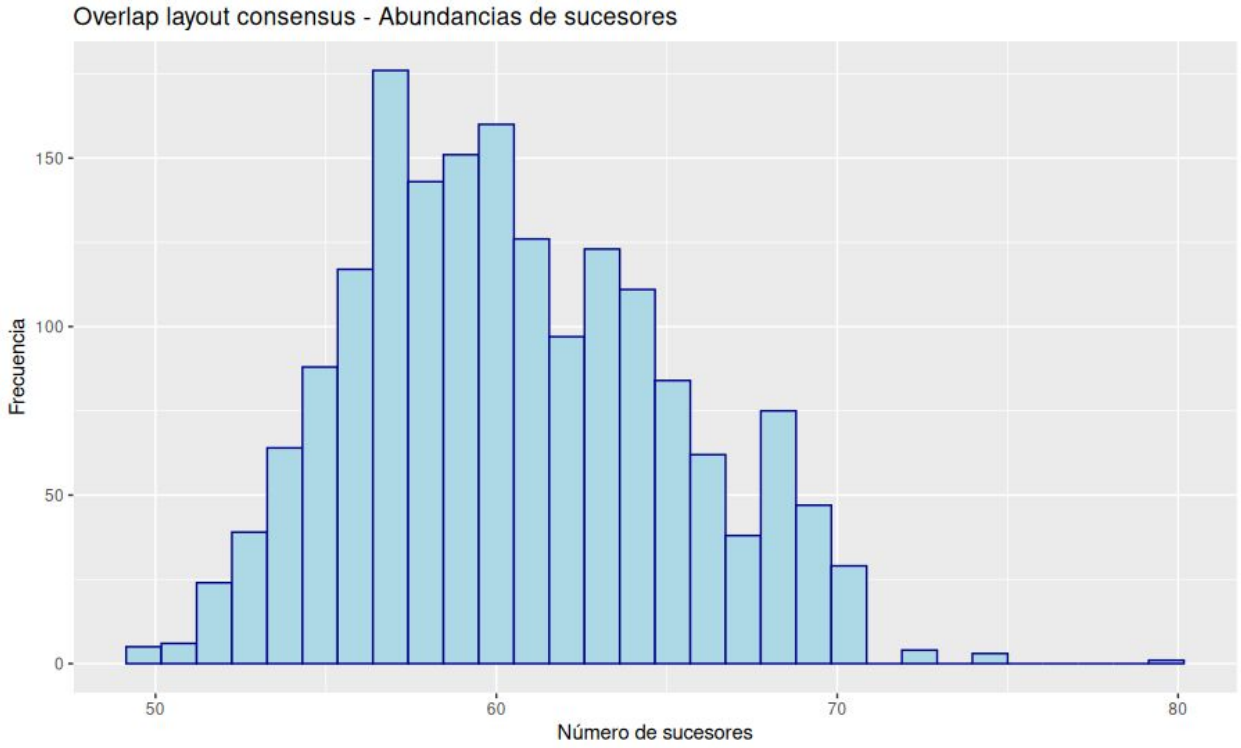

Figura 14: Distribución de abundancia de sucesores para “test100x.fastq”

A continuación se reportan los tiempos de ejecución para cada ensayo realizado:

Tiempo de ejecución construyendo el grafo de sobrelape (ms)	Tiempo de ejecución ensamblando la secuencia (ms)
100	8
235	11
934	22
2492	28

Se observa que, con el tamaño predeterminado de sobrelape (10 pares de bases), el tiempo que toma el programa para armar el grafo de sobrelapes y el que toma ensamblando la secuencia aumenta en tanto n (número de “reads” procesados) aumenta. Por otro lado, se puede concluir que a mayor profundidad aumenta la cantidad de secuencias que se repiten y la distribución de sucesores por secuencia se asemeja más a una distribución Poisson.

3. Reportar la secuencia que se genera y el tiempo de ejecución para cada fastq de ejemplo. Reportar para cada fastq entre qué tamaños de sobrelape se puede ensamblar la secuencia.

Nota: debido a la longitud de la secuencias generadas, se decidió omitir todo su contenido en este documento, pero se pueden observar en el siguiente link: https://drive.google.com/drive/folders/1gqpP8gUG3uxNsOOCsvjVCmxBypxzYQWj?usp=sharing

La siguiente tabla muestra, para cada archivo de prueba, la secuencia ensamblada, los tiempos de ejecución y los tamaños de sobrelape que se requieren para que se obtenga un ensamblaje

Archivo	Secuencia ensamblada	Tiempo de ejecución construyendo el grafo de sobrelape (ms)	Tiempo de ejecución ensamblando la secuencia (ms)	Tamaño de sobrelape requerido
test10x	l peloton de fusilamiento el coronel Aureliano Buendia habia de recordar aquella tarde remota en que su padre lo llevo a conocer el hielo.	285	22	3 a 86

	Macondo era entonces una aldea de veinte casas de barro y canhabrava construidas a la orilla de un rio de aguas diafanas...			
test20x	chos anhos despues frente al peloton de fusilamiento el coronel Aureliano Buendia habia de recordar aquella tarde remota en que su padre lo llevo a conocer el hielo. Macondo era entonces una aldea de veinte casas de barro y canhabrava construidas a la orilla	530	42	2 a 99
test50x	os anhos despues frente al peloton de fusilamiento el coronel Aureliano Buendia habia de recordar aquella tarde remota en que su padre lo llevo a conocer el hielo. Macondo era entonces una aldea de veinte casas de barro y canhabrava construidas a la orilla de un rio de aguas diafanas que se precipitaban por un lecho de piedras pulidas...	2515	53	10 a 89
test100x	Muchos anhos despues frente al peloton de fusilamiento el coronel Aureliano Buendia habia de recordar aquella tarde remota en que su padre lo llevo a conocer el hielo. Macondo era entonces una aldea de veinte casas de barro y canhabrava construidas a la orilla de un rio de aguas diafanas que se precipitaban...	9034	203	2 a 99

A medida que aumentaba la profundidad (dada en los nombres de los archivos de prueba) aumentaba considerablemente el número de secuencias distintas y la longitud de la cadena ensamblada. Como se esperaría, el tiempo de ejecución tanto para la construcción del grafo como para el ensamblaje de la cadena, aumentan conforme la profundidad aumenta. Concomitante a esto, se puede ver también que el tiempo de construcción del grafo es notablemente mayor al de la operación de ensamblado.

Las secuencias ensambladas mostradas anteriormente fueron generadas con un tamaño mínimo de sobrelape igual a 10. Entre más se acerca este valor a 100 (el tamaño de cada secuencia en los archivos fastq), la secuencia que se ensambla disminuye de tamaño. Para el archivo “test100x” y un tamaño mínimo de sobrelape igual a 10, se logra ensamblar la secuencia completa. Un aspecto a resaltar con respecto al tamaño mínimo de sobrelape, es que no necesariamente se van a poder ensamblar secuencias desde un tamaño igual a 1. Esto se debe a que la implementación actual busca la secuencia que no tenga predecesores en el grafo de sobrelape y, entre más pequeño sea el tamaño mínimo de sobrelape, es más probable que no exista una secuencia cuyo prefijo no sea sufijo de otra; así mismo, esto depende de las secuencias que estén presentes para cada archivo.

- **Completitud de las secuencias ensambladas**

Dado que se logró ensamblar la secuencia completa con “test100x”, se puede utilizar esta misma secuencia como referencia y, compararla con los ensamblajes obtenidos con los demás casos de prueba. A continuación se registra la proporción de caracteres que faltaron en cada caso de prueba con respecto a la secuencia de referencia (calculado como la división entre el número de caracteres faltantes y la totalidad de los caracteres en la secuencia de referencia)

Archivo	**Tasa de caracteres faltantes**
test10x	0.01381509
test20x	0.0007
test50x	0.001416932
test100x	0

Se puede notar que entre más profundidad, la tasa de caracteres faltantes disminuye, con la excepción de “test50x” que aumenta un poco con respecto a “test20x”. Esto, sin embargo, también depende de las secuencias que estén presentes en cada prueba: por ejemplo, en la prueba “test50x” no había ninguna secuencia que cubriera los primeros 4 caracteres del texto, mientras que en “test20x” no había ninguna secuencia que cubriera los primeros 2 caracteres (el resto de texto para los dos casos fue ensamblado de forma completa); esto hizo que

“test20x” tuviera menos tasa de error, a pesar de que “test50x” contuviera más redundancia. Un último detalle a notar es que, los caracteres faltantes de las pruebas con respecto a la referencia, siempre corresponden a un sufijo o un prefijo del texto completo.

Simulación de Lecturas

4. Completar el script “SimpleReadsSimulator” para generar lecturas aleatorias de una secuencia de por lo menos 10Kbp en formato fasta. Probar los comandos “Overlap” y “Kmers” de “ReadsAnalyzerExample” con tamaños de secuencia 50, 100, 200, 500 y con profundidad promedio de 5X, 10X, 20X, 50X y 100X. Reportar tiempos de ejecución de los algoritmos en cada caso. Determinar la cantidad de lecturas máxima que puede procesar cada comando con un máximo de memoria de 4Gb.

Nota: Los archivos simulados se pueden encontrar en la siguiente carpeta: https://drive.google.com/drive/folders/1F8FkokinKBRZ5mA3sLIPgUPAV_JqbdKg?usp=sharing

Para probar la simulación de lecturas del script “SimpleReadsSimulator” se utilizó una secuencia de Clostridium perfringens CP4 con un tamaño de 20945 bp; el archivo se puede descargar en el siguiente enlace:

https://drive.google.com/file/d/1Sav2yKUNJzeKyzdUQ-xwT7caVOsSjNZW/view?usp=sharing

La siguiente tabla muestra los tiempos de ejecución para los algoritmos de **conteo de k-mers** y **overlap-layout consensus**, para lecturas simuladas con distintas profundidades promedio y varios tamaños de secuencia:

Algoritmo	Tamaño de secuencia	Número de lecturas simuladas	Profundidad promedio esperada	Tiempo de ejecución (ms)
Overlap-layout consensus	50	2095	5X	1826
		4189	10X	6673
		8378	20X	25419
		20945	50X	140528
		41890	100X	352038
	100	1048	5X	908
		2095	10X	2831

		4189	20X	10818
		10473	50X	70056
		20945	100X	287674
	200	524	5X	422
		1048	10X	1474
		2095	20X	5047
		5237	50X	32738
		10473	100X	140864
	500	210	5X	228
		419	10X	1977
		838	20X	1929
		2095	50X	11723
		4189	100X	46335

Algoritmo	**Tamaño de secuencia**	**Número de lecturas simuladas**	**Profundidad promedio esperada**	**Tiempo de ejecución**
Conteo de K-mers	50	2095	5X	80
		4189	10X	118
		8378	20X	123
		20945	50X	183
		41890	100X	265
	100	1048	5X	81
		2095	10X	90
		4189	20X	130
		10473	50X	208

		20945	100X	409
	200	524	5X	83
		1048	10X	103
		2095	20X	133
		5237	50X	235
		10473	100X	409
	500	210	5X	82
		419	10X	136
		838	20X	129
		2095	50X	241
		4189	100X	417

Las siguientes figuras resumen de forma más clara los resultados de tiempo de ejecución para cada algoritmo (en el eje x, *L* significa la longitud de cada lectura y *P* la profundidad promedio):

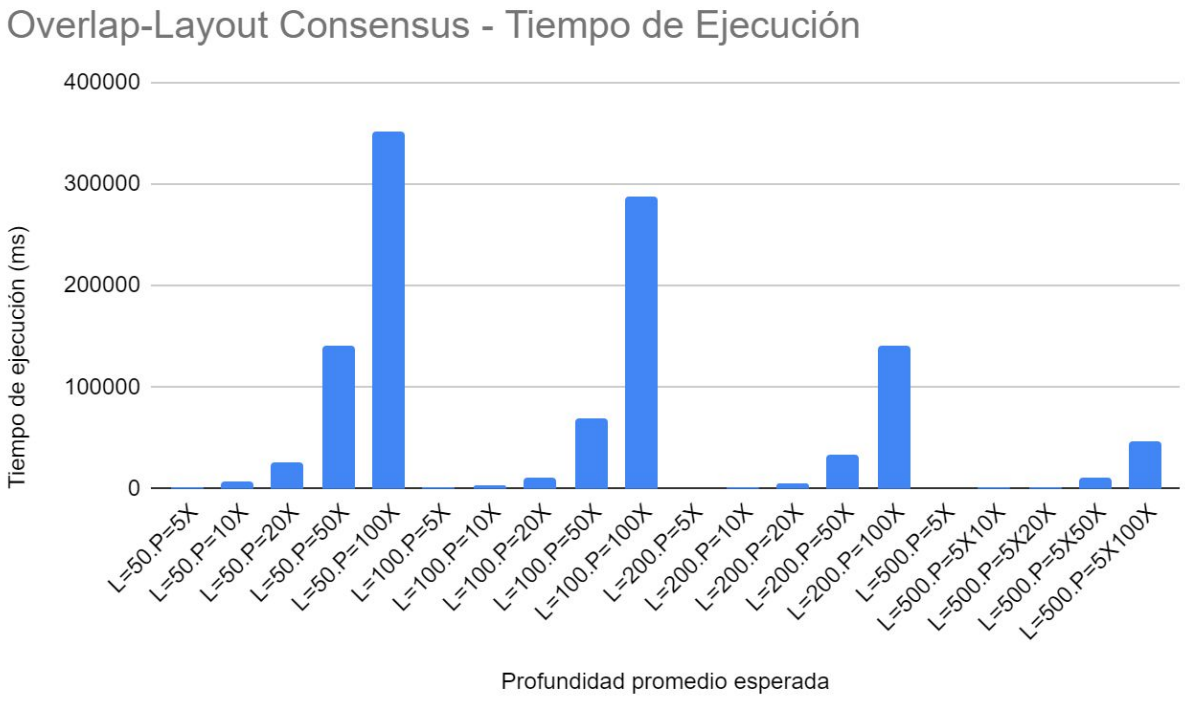

Figura 16: Tiempos de ejecución para Overlap-Layout Consensus

Figura 17: Tiempos de ejecución para Conteo de K-mers

Como se puede apreciar, el tiempo de ejecución siempre aumenta conforme la profundidad lo hace, sin embargo, en el caso del ejercicio con Overlap-Layout Consensus, se observa que a medida que aumenta L, el tiempo de ejecución disminuye. Esto se debe a que si se aumenta el tamaño de la lectura (L), el número de lecturas simuladas va a disminuir (para determinada profundidad *P*); este hecho se da al calcular el número de lecturas mediante la siguiente fórmula (recordar que 20945 es el tamaño de la secuencia utilizada para la simulación):

$$n\acute{u}meroLecturas = P * \frac{20945}{L}$$

Un último aspecto que se puede resaltar es que, podemos ver que los tiempos de ejecución crecen más rápido para Overlap-Layout Consensus que para el conteo de k-mers, con respecto al tamaño de la secuencia. Esto es evidencia del hecho de que, con la implementación actual el algoritmo para contar k-mers crece cuadráticamente en el tiempo sobre la longitud de la secuencia, mientras que el de Overlap-Layout Consensus crece de forma cúbica.

- **Límite en Memoria para los Algoritmos**

Se utilizó el simulador desarrollado en este punto para generar archivos más grandes y probar el límite de memoria de los algoritmos con la implementación actual. El programa arroja errores

de uso límite de memoria con 4GB disponibles, para un archivo con 1000000 secuencias que tienen tamaño 100. Podemos sin embargo ver que esto depende de la forma del grafo de sobrelape. Para *n* secuencias, el peor de los casos es la construcción de un grafo totalmente conectado, para lo cual se ocuparía n^2 en espacio; si se tiene espacio para guardar 4GB, *n* tendría que tener un tamaño máximo de 63Kbp aproximadamente (en el peor de los casos donde el archivo generado resulte en que todas las secuencias tengan sobrelape con todas las demás). Para el caso del conteo de k-mers, el espacio que se ocupa es $n * L * I$ donde L es el tamaño de cada secuencia e I es el tamaño de un número entero.

5. Mejorar el script reads simulator incluyendo un parámetro nuevo que permita simular una tasa de error aleatoria. Simular datos con un tamaño de lectura de 50bp y profundidad promedio de 20X. Visualizar y comparar la distribución de abundancia de k-mers en cada caso

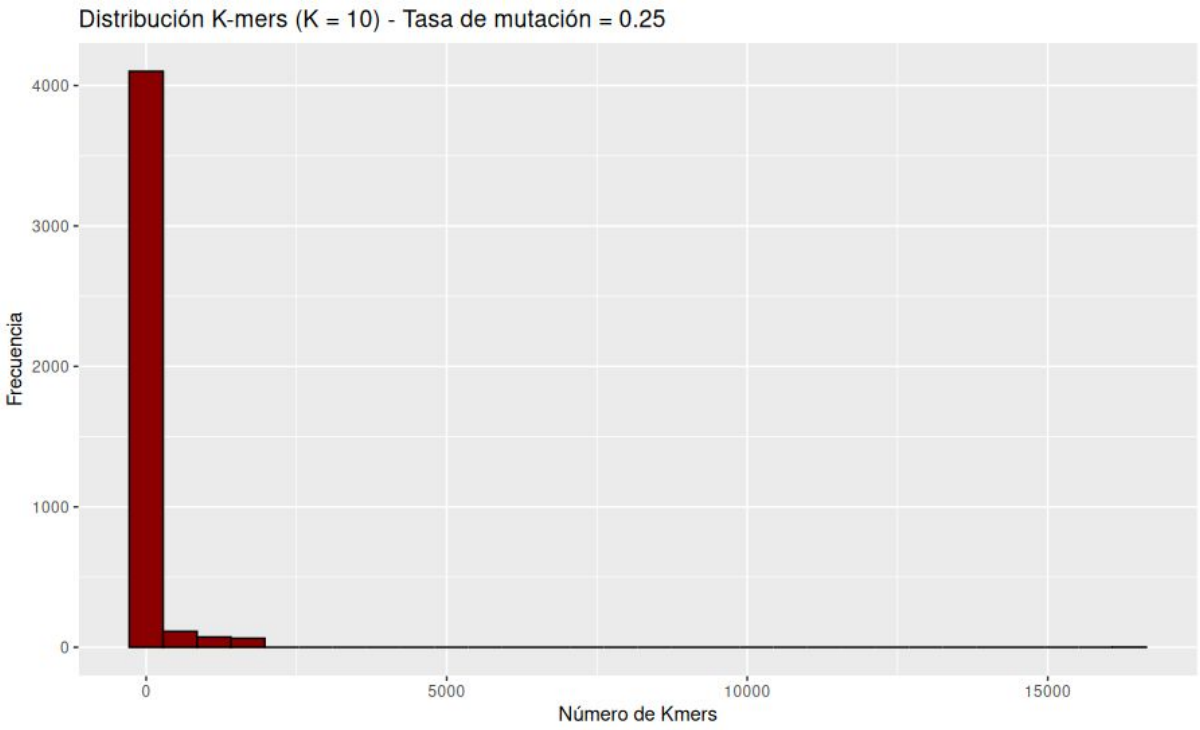

Figura 18: Distribución de Kmers de longitud 10 con tasa de mutación aleatorio = 0.25

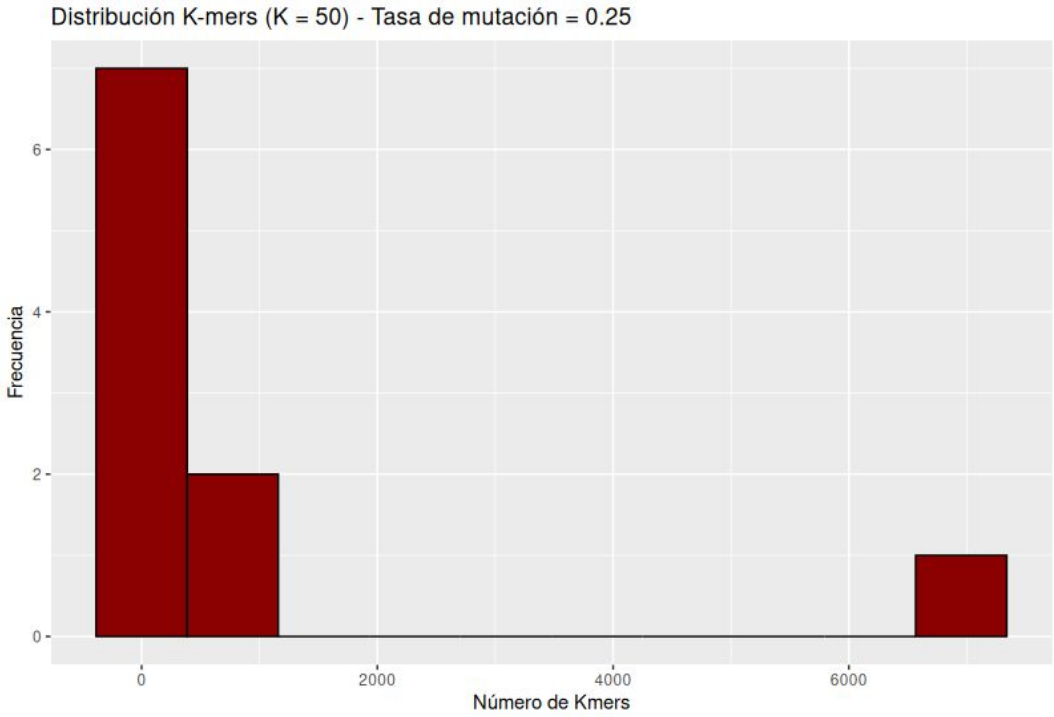

Figura 18: Distribución de Kmers de longitud 50 con tasa de mutación aleatorio = 0.25

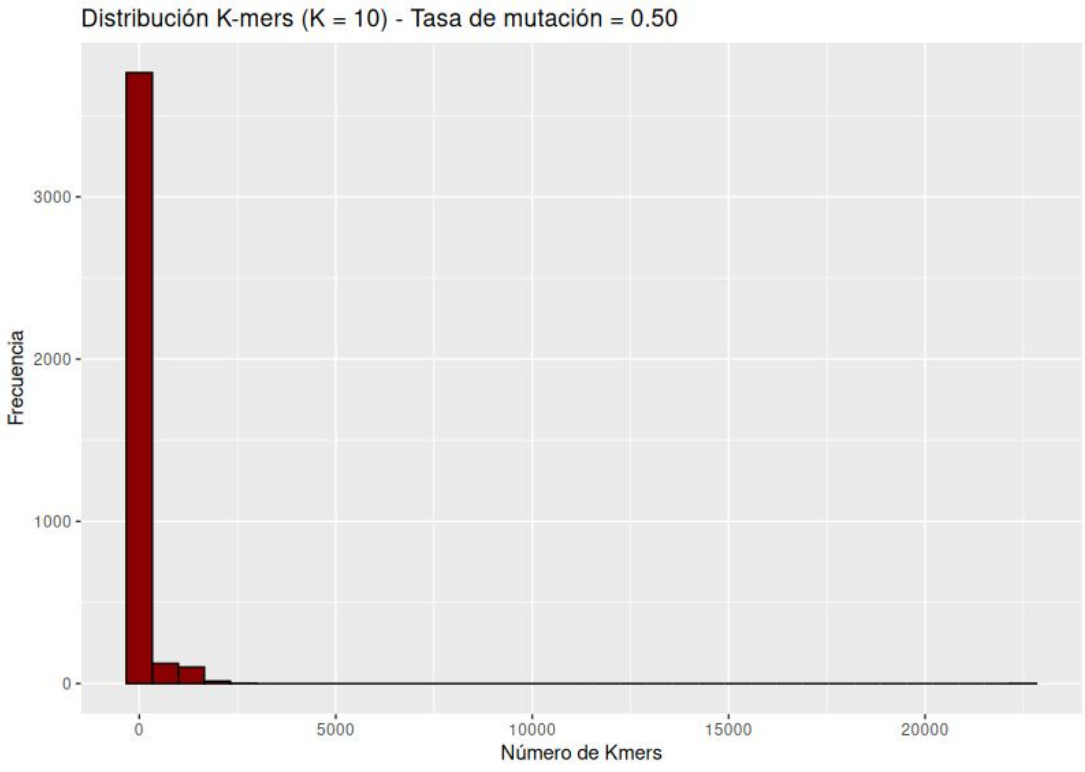

Figura 19: Distribución de Kmers de longitud 10 con tasa de mutación aleatorio = 0.50

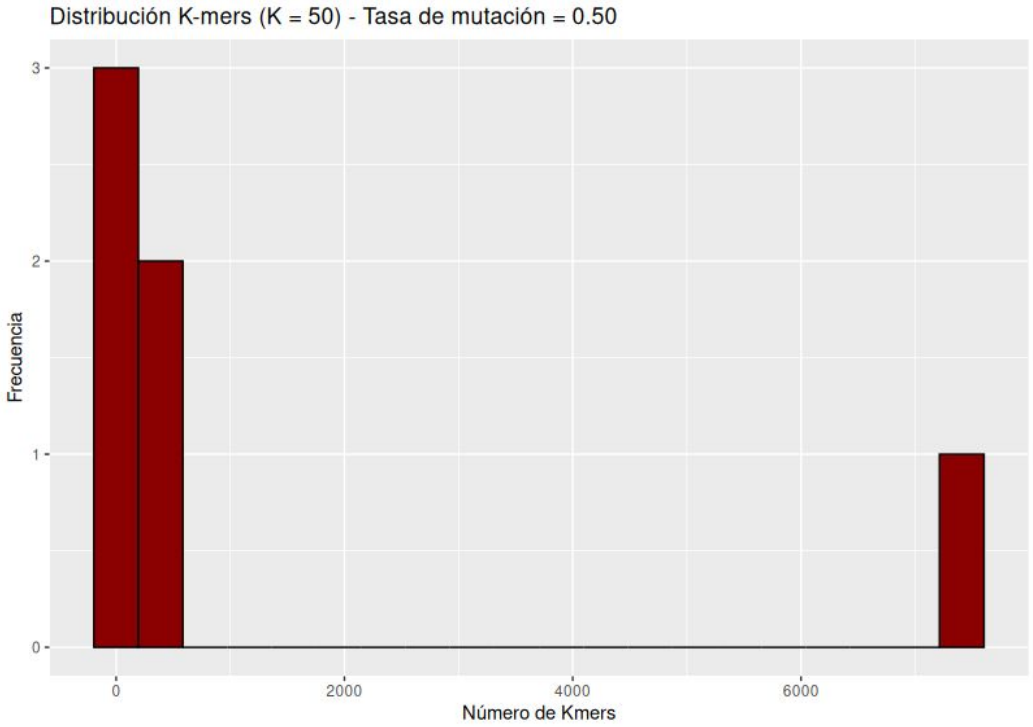

Figura 20: Distribución de Kmers de longitud 50 con tasa de mutación aleatorio = 0.50

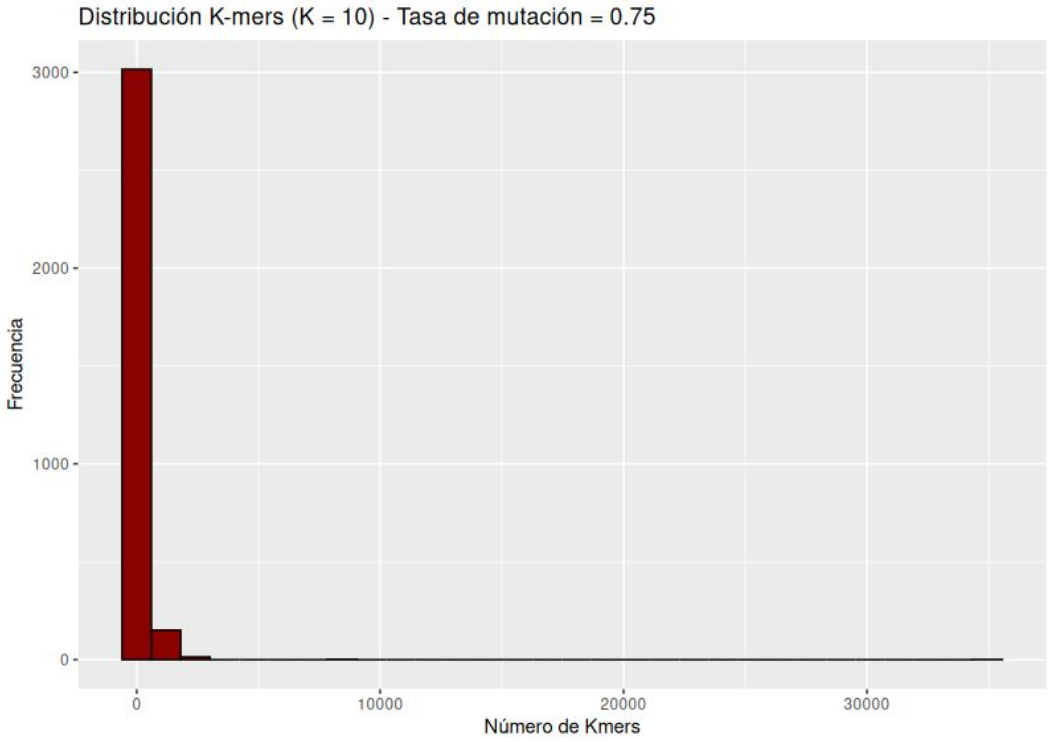

Figura 21: Distribución de Kmers de longitud 10 con tasa de mutación aleatorio = 0.75

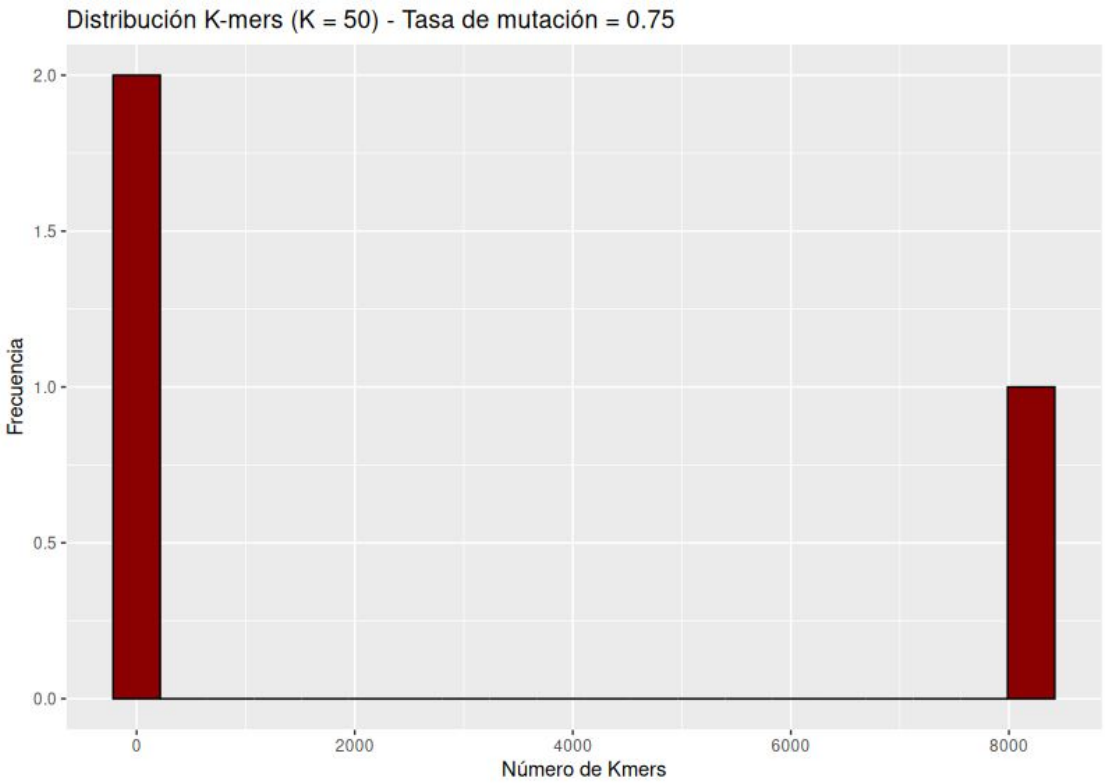

Figura 21: Distribución de Kmers de longitud 50 con tasa de mutación aleatorio = 0.75

A partir de las distribuciones de frecuencias de kmers, se observa que, incluso desde una tasa de mutación pequeña, de 0.25, la tendencia demuestra que se pierde la forma de distribución que se obtiene en kmers obtenidos de secuencias sin mutaciones. Aparte de esto, se observa que existe una gran cantidad de kmers que presentan repeticiones cuando su tamaño es menor, pues en los ensayos donde su longitud se determinó como 50pb se observa que una gran cantidad de estos se repiten muy pocas veces. Esto se puede explicar por el hecho que, dado que las mutaciones se presentan en los “reads”, si la longitud de los kmers es menor la probabilidad de que aun con estos cambios nucleotídicos estos no sean iguales es menor que con kmers de mayor longitud.

Bono: Buscar y descargar de la base de datos Sequence Read Archive (SRA) lecturas reales de secuenciación de genoma completo con Illumina de alguna bacteria y buscar también en NCBI un genoma ensamblado de la misma bacteria. Ejecutar el comando “Kmers” con la mayor cantidad de lecturas que sea posible y reportar la distribución de abundancias de k-mers. Comparar esta distribución con la distribución que se obtiene con lecturas simuladas sin errores con la misma profundidad promedio. Reportar si se ven diferencias importantes entre las distribuciones y las posibles causas de estas diferencias.

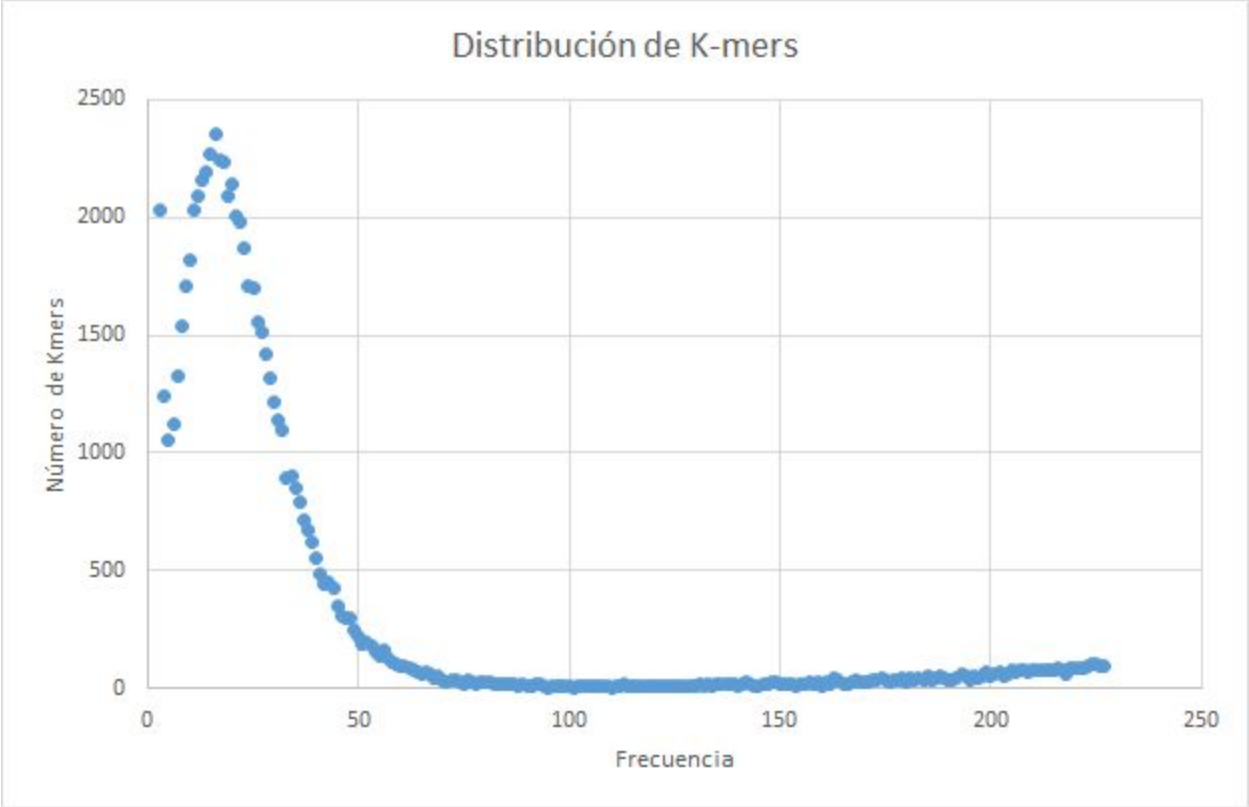

Figura 22: Distribución de Kmers (10X) de longitud 10 a partir de reads n=81 por Illumina de *B.subtilis*

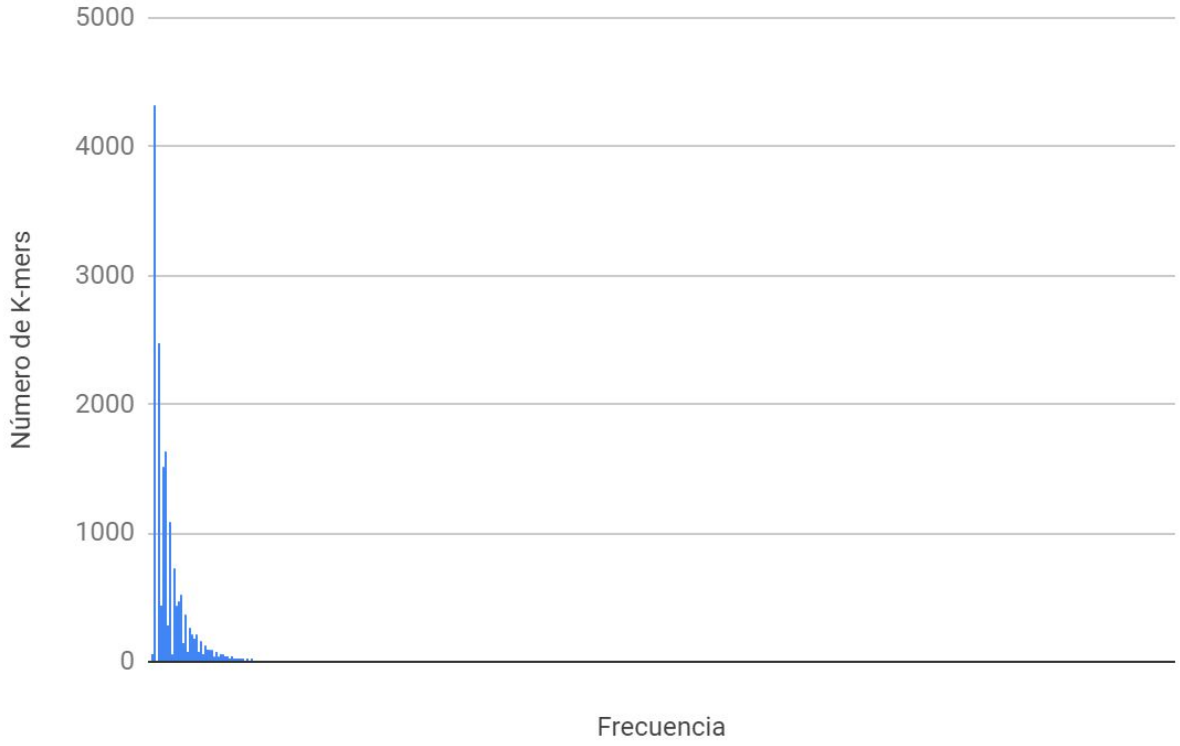

Figura 23: Distribución de Kmers (12X) de longitud 10 a partir de simulaciones del genoma ensamblado de *B.subtilis*.

A pesar que no se generaron mutaciones en las secuencias simuladas se observa que su distribución se asemeja a la distribución generada por el análisis de kmers sobre los reads originales obtenidos por Illumina. En ambos casos, la gran mayoría de kmers no se repiten muchas veces. Aun así, en el ensayo simulado se observa que hay una menor varianza de la distribución. Es importante mencionar que para esta simulación se requirió una gran cantidad de tiempo a comparación de otros ensayos realizados en esta investigación, pues tomó 430742 ms armando la tabla de kmers. Claramente, este proceso también tomó un tiempo considerable en el análisis del genoma sin ensamblar, equivalente a 933654ms, por lo que se observa que tomó más tiempo a partir de las lecturas originales a comparación de las simuladas, ambos procesos, para una profundidad de 10X y 12X respectivamente.

Referencias:

University of Connecticut. (2019). Genome Size Estimation Tutorial. Retrieved from https://bioinformatics.uconn.edu/genome-size-estimation-tutorial/#

Genoma secuenciado por Illumina *Bacillus cereus*:
https://www.ncbi.nlm.nih.gov/sra/SRX9114637[accn]

Genoma ensamblado *Bacillus cereus*:
https://www.ncbi.nlm.nih.gov/assembly/GCF_014389245.1