Universidad de Concepción
Departamento de Informática y Cs. de la Computación
Ingeniería Civil Informática

Proyecto Semestral: Detección de Variaciones en el Número de Copias (CNVs) mediante Análisis de Cobertura Genómica en Archivos BAM de Alta Dimensión

Gabriel Huerta Torres
Diego Oyarzo Navia

19 de Diciembre de 2025

Resumen

Las variaciones en el número de copias (Copy Number Variations, CNVs) corresponden a regiones del genoma donde se producen deleciones o duplicaciones de segmentos de ADN. La detección de estas variantes es un problema relevante tanto en investigación genómica como en aplicaciones clínicas. Sin embargo, el análisis de CNVs implica el procesamiento de archivos genómicos de gran tamaño, como archivos BAM de alta cobertura, lo que plantea importantes desafíos computacionales.

En este trabajo se presenta el diseño e implementación de un sistema para la detección de CNVs basado en el análisis de profundidad de cobertura a partir de archivos BAM de alta cobertura (aproximadamente 300x). El sistema utiliza un enfoque de procesamiento por bins genómicos, cálculo de estadísticas robustas y normalización de cobertura, permitiendo un manejo eficiente de grandes volúmenes de datos. La solución fue implementada en C++ utilizando la biblioteca `htslib`, y los resultados son exportados en formato CSV para su posterior análisis.

Índice

1. Introducción

Las tecnologías de secuenciación de nueva generación han permitido generar grandes volúmenes de datos genómicos a un costo cada vez menor. Como consecuencia, el análisis eficiente de estos datos se ha convertido en un desafío central tanto para la bioinformática como para la ingeniería informática orientada al manejo de datos masivos.

Dentro de las variaciones estructurales del genoma, las variaciones en el número de copias (CNVs) representan una categoría particularmente relevante, ya que pueden afectar regiones extensas del ADN y estar asociadas a diversas patologías. Una estrategia común para la detección de CNVs consiste en analizar la profundidad de cobertura obtenida a partir de la alineación de lecturas de secuenciación contra un genoma de referencia.

El presente proyecto se enmarca en el ramo *Tópicos en Manejo de Grandes Volúmenes de Datos* y tiene como objetivo principal diseñar e implementar un sistema capaz de procesar archivos BAM de alta cobertura de forma eficiente, aplicando técnicas de binning, estadística descriptiva y normalización para detectar regiones candidatas a CNVs.

2. Problema

Actualmente, las herramientas comúnmente utilizadas para detectar CNVs en los genomas humanos son costosas desde el punto de vista computacional (en memoria y complejidad temporal), algunas de las mas comunes pueden ser encontradas en el paper (3) sobre el Benchmarking de estas para casos clinicos que hemos decidido analizar. Por ello, nosotros proponemos una solución de bajo consumo de recursos, logarítmica y probabilística, como una forma de análisis preventivo —no como un reemplazo completo— para la detección de CNVs antes del uso de herramientas exactas pero más costosas.

3. Conceptos importantes

- **Genoma:** Conjunto completo de ADN de un organismo. En humanos son aproximadamente **3 mil millones de pares de bases** (letras A, T, G, C).
- **Cobertura:** Número promedio de veces que cada base del genoma es leída por la secuenciación. **Alta cobertura** implica mayor confianza en las variantes detectadas. **Cobertura desigual** puede indicar **CNVs**.
- **BAM file:** Formato de archivo **binario** que contiene **lecturas (reads) alineadas** al genoma de referencia.
- **CNV (Copy Number Variation):** Tipo de **variante estructural** donde una región del genoma tiene un **número de copias diferente al normal** (2 en organismos diploides).
- **Bin: Ventana o segmento del genoma** de tamaño fijo (por ejemplo, 1 kb o 10 kb) usada para **agrupar lecturas o datos de cobertura**.
- **Quantile sketch: Estructura de datos probabilística** que **resume una distribución de valores**, permitiendo **consultas aproximadas de cuantiles** con **memoria constante**.

4. Propuesta de Proyecto

El objetivo de este trabajo es detectar **variaciones en el número de copias** (CNVs) en genomas humanos a partir de la **distribución de coberturas de secuenciación**, utilizando estructuras de datos compactas denominadas *quantile sketches*.

En secuenciación masiva, cada fragmento de ADN del genoma es leído múltiples veces por el secuenciador. Estas lecturas (*reads*) se almacenan en archivos `FASTQ` y posteriormente se alinean contra un genoma de referencia (`FASTA`) mediante un alineador como BWA o Bowtie. El resultado es un archivo `BAM`, que contiene la posición exacta de cada lectura en el genoma y permite calcular la **cobertura**.

La **cobertura** representa cuántas veces cada posición del genoma fue leída. Si el genoma se secuenció a una profundidad de 30×, significa que, en promedio, cada base fue leída 30 veces. Sin embargo, esta medida global no es suficiente para detectar variaciones estructurales, ya que las duplicaciones y deleciones afectan solo regiones específicas del genoma.

4.1. Cobertura local y detección de CNVs

Para analizar la variación local, el genoma se divide en ventanas de tamaño fijo llamadas **bins** (por ejemplo, de 1 kb). Para cada bin se calcula su cobertura promedio, es decir, el número total de bases de lecturas que caen dentro del bin dividido por su tamaño. Regiones con cobertura significativamente mayor o menor que la mediana global pueden indicar **duplicaciones** o **deleciones**, respectivamente.

$$C_i = \frac{\text{bases leídas dentro del bin}_i}{\text{tamaño del bin}} \tag{1}$$

Donde C_i es la cobertura del bin i. A partir de todos los C_i del cromosoma, se obtiene una distribución de coberturas $\{C_1, C_2, \ldots, C_n\}$.

4.2. Uso de quantile sketches

Guardar la cobertura de millones de bins puede ser costoso en memoria. Por ello, se utilizarán **quantile sketches** (KLL), que permiten construir una representación compacta de la distribución de coberturas sin almacenar todos los valores. Estos sketches permiten consultas como percentiles o cuantiles de forma aproximada, con errores acotados.

- El **percentil 50 (p50)** representa la cobertura típica del cromosoma.
- Los percentiles altos (p95, p99) indican bins con cobertura elevada → posibles duplicaciones.
- Los percentiles bajos (p5, p1) indican bins con cobertura reducida → posibles deleciones.

A partir de los cuantiles obtenidos se pueden definir indicadores simples para detectar anomalías en la distribución de coberturas. La mediana (p50) representa la cobertura típica del cromosoma, mientras que las colas (p5, p95) reflejan desviaciones locales.

Un desplazamiento pronunciado de la cola superior ($p95 \gg p50$) sugiere la existencia de regiones con cobertura anormalmente alta, indicativas de **duplicaciones**, mientras que un desplazamiento de la cola inferior ($p5 \ll p50$) indica **deleciones**.

En este trabajo se generará un **sketch por cromosoma**, de modo que cada distribución refleje el comportamiento de las coberturas dentro de ese cromosoma en particular.

El análisis no busca determinar la posición exacta de las duplicaciones o deleciones, sino detectar su **presencia o evidencia estadística** a partir de la forma de la distribución de coberturas.
Esto permite identificar cromosomas que presentan señales de CNVs sin requerir un análisis base por base ni almacenar todos los datos.
Estos indicadores permiten evaluar la existencia de duplicaciones o deleciones sin almacenar todas las coberturas individuales, sino únicamente un resumen probabilístico de su distribución.

4.3. Resumen del flujo de trabajo

1. **Entrada:** archivo `BAM` con lecturas alineadas al genoma.
2. **Procesamiento:**
   - Leer lecturas por cromosoma.
   - Dividir el cromosoma en bins de tamaño fijo (1 kb).
   - Contar cuántas lecturas cubren cada bin.
   - Calcular la cobertura promedio por bin.
   - Insertar cada cobertura en un quantile sketch (KLL).
3. **Salida:**
   - Distribución de coberturas por cromosoma (percentiles).
   - Identificación de cromosomas con patrones anómalos (presencia de CNVs).

4.4. Resultados esperados

El sistema debe permitir identificar la distribución de coberturas del genoma que difiera de la distribución típica.

La interpretación se centrará en la **detección global**, es decir, determinar si un genoma completo muestra evidencia estadística de duplicaciones o deleciones, obteniendo las estadísticas de ese CNV

Se espera observar desplazamientos en los cuantiles o colas de la distribución asociados a duplicaciones o deleciones, demostrando que las estructuras de resumen probabilístico (sketches) pueden utilizarse como una alternativa eficiente para la detección preliminar de CNVs.

5. Experimentación

La experimentación se realizó utilizando datos externos de secuenciación completa del genoma humano (Whole Genome Sequencing, WGS) (5), alineados contra el genoma de referencia GRCh38 (6). Se usaron los primeros 5 chromosomas del archivo debido a su gran peso en GB. El objetivo principal de esta fase fue evaluar el comportamiento del enfoque propuesto en términos de eficiencia computacional, uso de memoria y precisión aproximada en la estimación de cuantiles de cobertura.

5.1. Dataset utilizado

Se utilizó un archivo BAM generado por el National Institute of Standards and Technology (NIST) (4) como parte del proyecto *Genome in a Bottle*. El archivo fue alineado utilizando NovoAlign y presenta una cobertura promedio aproximada de 300×.

Las principales características del dataset son:

- Tamaño del archivo BAM: 191 GB.
- Número total de lecturas: 3 107 255 000.
- Cromosomas procesados: 5.
- Genoma de referencia: GRCh38.

5.2. Configuración experimental

El genoma fue dividido en bins de tamaño fijo, y para cada bin se calculó la cobertura promedio. Estos valores fueron insertados incrementalmente en un *quantile sketch* KLL, permitiendo resumir la distribución de coberturas de cada cromosoma sin almacenar todos los valores explícitamente.

Se evaluaron distintos parámetros:

- Tamaño de bin: 1000 bases.
- Parámetro K del sketch KLL: valores variables, con énfasis en $K = 400$.

La elección del tamaño de bin se basó en un análisis exploratorio de la dispersión de la cobertura, utilizando el rango intercuartílico (IQR) y la ganancia marginal de dispersión entre configuraciones consecutivas.

5.3. Metodología

La experimentación se dividió en dos fases:

1. **Estimación global:** se recorre el archivo BAM completo, se calcula la cobertura por bin y se alimenta el sketch KLL para obtener cuantiles representativos de la distribución.
2. **Detección local:** utilizando los cuantiles estimados, se aplican umbrales dinámicos para detectar secuencias consecutivas de bins con cobertura anómala, denominadas *runs*, asociadas a posibles CNVs.

6. Diseño de solución

El sistema desarrollado sigue un pipeline de procesamiento en tres etapas principales:

1. **Lectura y filtrado:** Procesamiento del archivo BAM utilizando libreria externa para la lectura especifica de estos tipos de archivos, htslib (1), con filtrado de reads no mapeados, secundarios, suplementarios y duplicados (flags `BAM_FUNMAP`, `BAM_FSECONDARY`, `BAM_FSUPPLEMENTARY`, `BAM_FDUP`).

2. **Conteo por bins:** División del genoma en ventanas de tamaño configurable (1-10 kb) y acumulación de cobertura mediante una tabla hash (`unordered_map<uint32_t, uint32_t>`).

3. **Construcción del sketch:** Inserción incremental de coberturas en el KLL sketch por cromosoma, con liberación de memoria tras procesar cada cromosoma (estrategia de flush).

6.1. Algoritmo KLL (Karnin-Lang-Liberty)

Se utilizó la implementación de KLL sketch de Apache DataSketches (2) con parámetro $k = 400$, que garantiza un error relativo $\epsilon < 1\,\%$ en la estimación de cuantiles con alta probabilidad, como también es el mejor en relación error y uso de memoria como se declara en la sección de resultados.

El algoritmo mantiene una estructura jerárquica de compactadores que resumen la distribución de valores mediante muestreo probabilístico, logrando una complejidad espacial de:

$$\mathcal{O}\left(\frac{1}{\epsilon}\log^2\log\frac{1}{\delta}\right) \tag{2}$$

donde δ es la probabilidad de fallo. Esta complejidad corresponde a la versión mergeable del KLL sketch implementada en Apache DataSketches, que permite combinar múltiples sketches de forma eficiente.

6.2. Optimizaciones Implementadas

- **Procesamiento por cromosoma:** En lugar de acumular todos los bins del genoma en memoria, se procesan cromosoma por cromosoma, insertando las coberturas en el sketch y liberando la memoria inmediatamente.
- **Pre-asignación de memoria:** Uso de `reserve(100000)` en el `unordered_map` para evitar rehashing costosos durante la inserción.
- **Keys numéricas:** Uso de identificadores enteros para bins en lugar de strings, reduciendo el overhead de hashing.

Esta estrategia reduce la memoria peak de $\mathcal{O}(n)$ (todos los bins) a $\mathcal{O}(\text{máx}\{\text{bins por cromosoma}\})$, aproximadamente 400 KB vs 1.8 MB para un genoma completo.

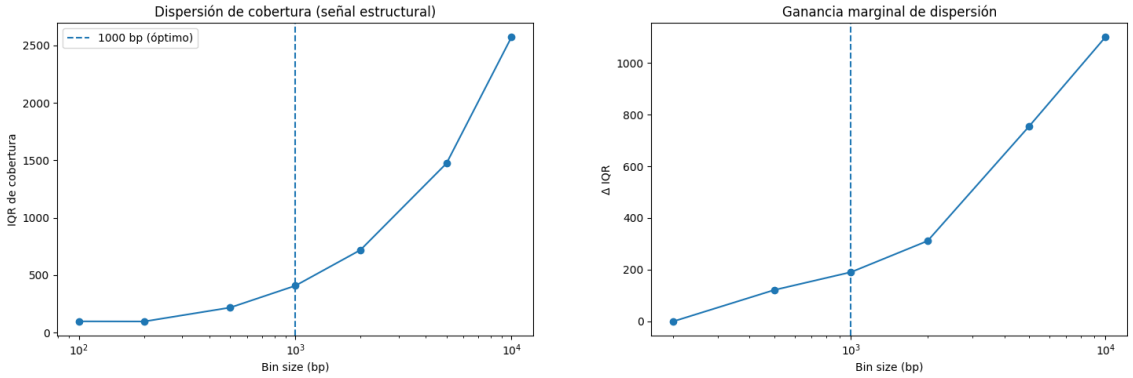
7. Análisis Experimental y Visualización de Resultados

En esta sección se presentan y analizan los principales resultados obtenidos durante la fase de experimentación, apoyándose en visualizaciones gráficas que permiten comprender el impacto de los parámetros del sistema sobre la estabilidad estadística, el error aproximado y la eficiencia computacional del enfoque propuesto.

7.1. Impacto del tamaño de bin en la dispersión de cobertura

La Figura 1a muestra el comportamiento del rango intercuartílico (IQR) de la cobertura al variar el tamaño de bin. El IQR se define como la diferencia entre los percentiles P75 y P25, y representa la variabilidad de la cobertura en la mayoría de las regiones genómicas.

La Figura 1b muestra la ganancia marginal de dispersión entre configuraciones consecutivas de tamaño de bin.



(a) Dispersión de cobertura en función del tamaño del bin.

(b) Ganancia marginal del IQR en función del tamaño del bin.

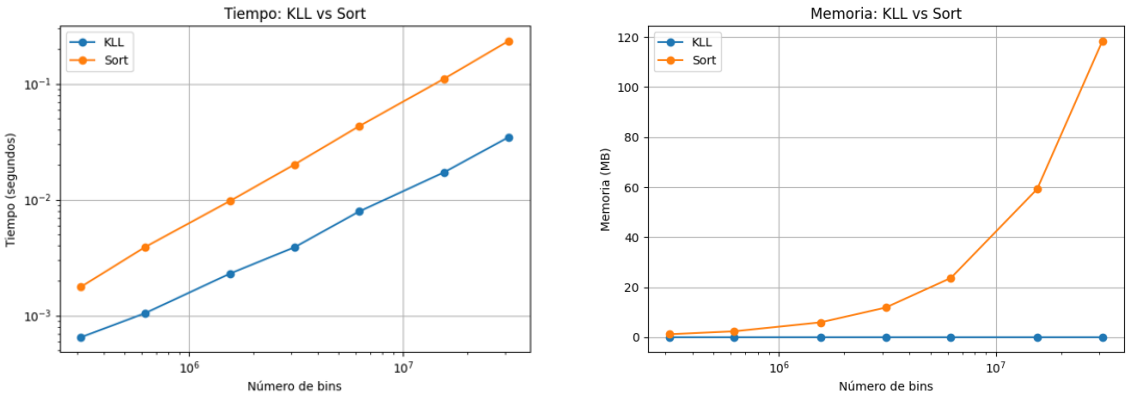
Figura 1: Decisión del tamaño de bins basado en IQR de cobertura.

Se observa que para tamaños de bin pequeños la dispersión de la cobertura es elevada, reflejando un mayor ruido local. A medida que el tamaño de bin aumenta, el IQR disminuye y se estabiliza, indicando una mayor robustez estadística.

La disminución progresiva de la ganancia marginal justifica la elección de un tamaño de bin de 1000 bases, ya que tamaños mayores no aportan mejoras significativas en estabilidad, pero sí reducen la resolución espacial del análisis.

7.2. Impacto en el KLL según el N° de bins

Para poder terminar de tomar la decisión del bin óptimo también quisimos ver como se relaciona el número de bins tanto para el KLL como para un sort convencional, dando así los siguientes resultados, donde la figura 2a nos muestra como varía el tiempo del KLL y del sort a partir del n° de bins y la figura 2b nos muestra como varía la memoria.



(a) KLL vs Sort (Tiempo)

(b) KLL vs Sort (Memoria)

Figura 2: Decisión del tamaño de bins basado en Espacio y Tiempo

Donde podemos notar que el KLL es superior en cuanto a memoria sobre el sort convencional dependiendo del número de bins, pero en tiempo no representa una mejora significativa.

7.3. Error aproximado del sketch KLL

El error de rank utilizado en este trabajo corresponde a la garantía teórica del algoritmo KLL (*Karnin, Lang y Liberty, 2016*), y no a una característica particular de la implementación de Apache DataSketches.

En el modelo teórico, el sketch KLL entrega una estimación aproximada del rank de un elemento, con un error acotado de forma probabilística. En particular, para un sketch con parámetro K, se cumple que:

$$\Pr\left(|\hat{r} - r| > \varepsilon N\right) \leq \delta$$

donde N es el número total de elementos procesados, $\varepsilon = O(1/\sqrt{K})$ es el error relativo máximo y δ es una probabilidad de fallo pequeña.

Expresando el error de forma normalizada, se obtiene el siguiente bound:

$$\text{RankError}(r) = \frac{|\hat{r} - r|}{N} \leq \frac{c}{\sqrt{K}}$$

donde c es una constante asociada al algoritmo.

La implementación de Apache DataSketches utiliza este mismo fundamento teórico, fijando constantes conservadoras y proporcionando estimaciones empíricas del error esperado. Por ejemplo,

se reporta que para $K = 200$ el error relativo en la mediana es aproximadamente 0.67 % con un 99 % de confianza.

En la experimentación realizada, el error de rank corresponde a una medición empírica obtenida al comparar los cuantiles aproximados entregados por el sketch KLL con los cuantiles exactos calculados mediante ordenamiento completo, observándose errores consistentemente inferiores a los límites teóricos. Como se muestra en la siguiente figura:

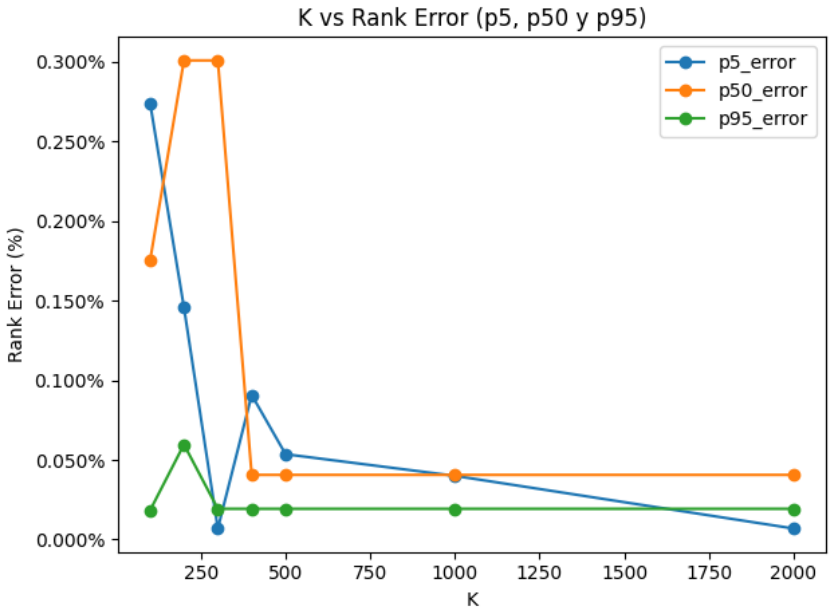


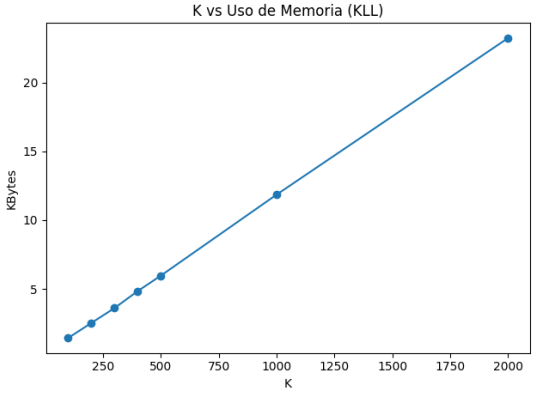
Figura 3: Error de rank del sketch KLL en función del parámetro K.

Los resultados muestran que el error disminuye rápidamente al aumentar K, alcanzando valores inferiores al 0.3 % para configuraciones moderadas. Este comportamiento concuerda con las garantías teóricas del algoritmo KLL y con las recomendaciones del proyecto Apache DataSketches, que sugieren valores cercanos a $K = 200$ como un buen compromiso entre precisión y memoria. Y debido a la figura 3 tomaremos $K = 400$ por su bajo error 0.05 %

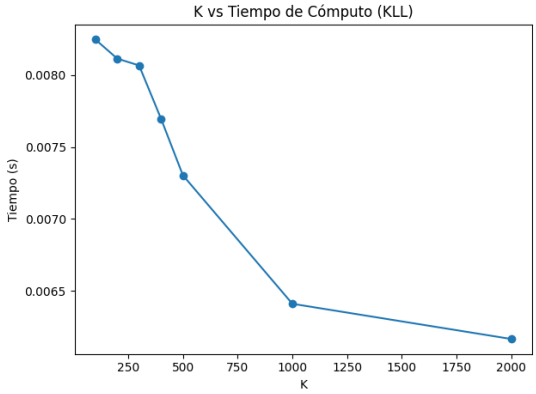
7.4. Eficiencia computacional

La Figura 4a presenta el tiempo de ejecución del sistema para distintos valores de K, manteniendo fijo el tamaño de bin en 1000 bases.

La Figura 4b presenta el uso de memoria del sistema para distintos valores de K, manteniendo fijo el tamaño de bin en 1000 bases.



(a) Uso de memoria del KLL para distintos valores K



(b) Tiempo de ejecución del KLL para distintos valores K

Figura 4: Tiempo de ejecución y espacio utilizado por el KLL

El tiempo de actualización del sketch se mantiene prácticamente constante para los valores de K evaluados, indicando que el costo computacional está dominado por la lectura de datos más que por el mantenimiento del sketch.

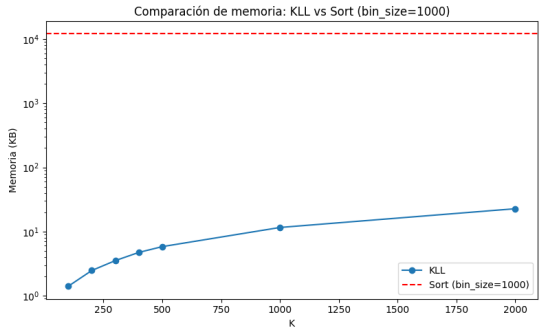
El uso de memoria del KLL sketch crece linealmente con K, confirmando el comportamiento teórico. Resultados altamente precisos se obtienen con menos de 5 KB de memoria.

7.5. KLL vs Sort

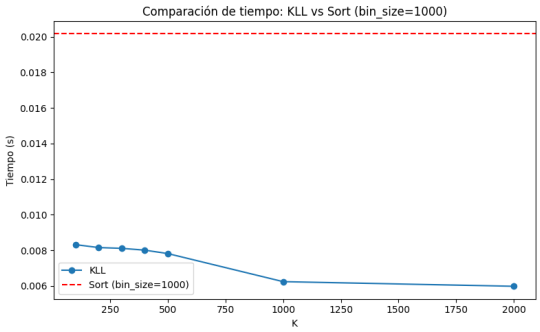
Debido a que metodos actuales usan Sort para la distribución de cobertura del genoma, para comprobar si el KLL es eficiente en tiempo y en espacio lo comparamos con el KLL para el genoma que usamos en nuestra experimentación, obteniendo así los siguientes resultados:

La Figura 5a muestra el uso de memoria asociado al sketch KLL en comparación con un enfoque exacto basado en ordenamiento completo.

La Figura 5b muestra el tiempo de ejecución asociado al sketch KLL en comparación con un enfoque exacto basado en ordenamiento completo.



(a) Uso de memoria del sketch KLL comparado con sort.



(b) Tiempo de ejecución del sketch KLL comparado con sort.

Figura 5: Comparación entre KLL y Sort para distintos valores K

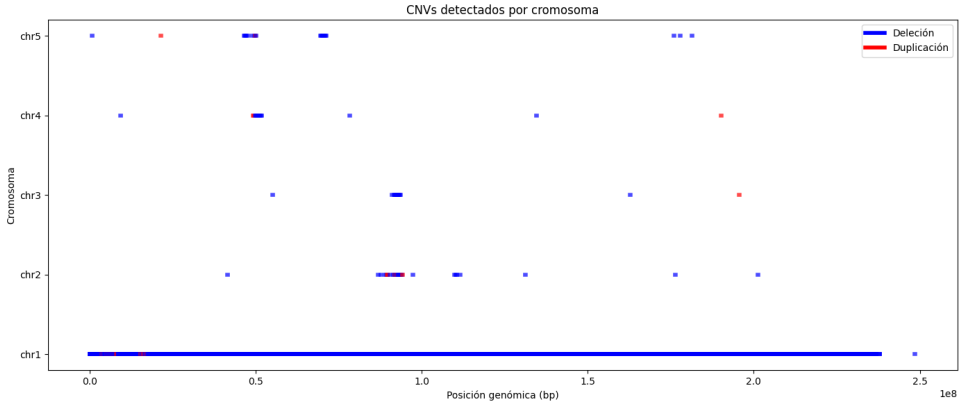
Mientras que el enfoque basado en ordenamiento no escala a genomas completos debido a su alto consumo de memoria, el sketch KLL mantiene un uso acotado y prácticamente constante, independiente del número total de bins procesados.

7.6. Análisis de detección de CNVs

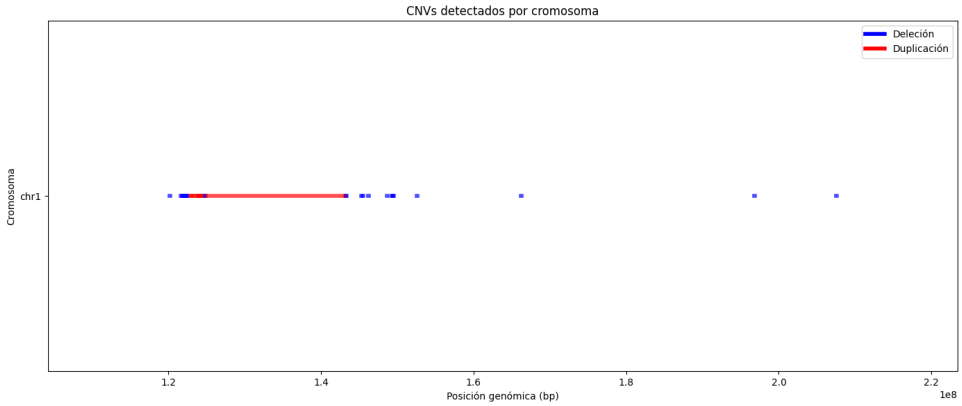
Una vez estimados los cuantiles globales de cobertura, se aplicaron umbrales dinámicos definidos como fracciones de la mediana de cobertura (0,5× y 1,5×) para detectar bins anómalos.

La Figura 6a muestra la cantidad de bins clasificados como deleciones y duplicaciones bajo longitúd minima de 1 run

La Figura 6b muestra la cantidad de bins clasificados como deleciones y duplicaciones bajo longitúd minima de 5 runs



(a) CNVs detectados para runs `>= 1`.



(b) CNVs detectados para runs `>= 5`.

Figura 6: Comparación de resultados de CNVs para runs distintos.

El uso de runs consecutivos permite filtrar falsos positivos y priorizar eventos más consistentes, reduciendo significativamente el número de regiones candidatas al aumentar el umbral mínimo de longitud.

8. Resultados

8.1. Selección del tamaño de bin

La experimentación mostró que un tamaño de bin de 1000 bases ofrece un balance adecuado entre resolución y estabilidad estadística. A tamaños menores, la cobertura presenta mayor ruido, mientras que bins mayores reducen la sensibilidad a variaciones locales relevantes.

El análisis del rango intercuartílico (IQR) evidenció que la ganancia marginal de dispersión disminuye significativamente al aumentar el tamaño de bin más allá de 1000 bases, justificando esta

elección.

8.2. Evaluación del parámetro K del sketch KLL

Se evaluó el impacto del parámetro K del sketch KLL en términos de error de rank, tiempo de ejecución y uso de memoria. El *rank error* mide la diferencia normalizada entre la posición real de un cuantil y la estimada por el sketch.

Los resultados mostraron que valores moderados de K permiten obtener errores bajos con un consumo de memoria reducido. En particular, se seleccionó $K = 400$ por ofrecer una buena relación entre precisión y eficiencia, manteniendo errores de rank inferiores al 0.3 %.

8.3. Rendimiento computacional

La evaluación del rendimiento computacional se realizó utilizando una configuración fija de tamaño de bin de 1000 bases y un sketch KLL con parámetro $K = 400$. Bajo esta configuración, el sistema procesó un archivo BAM de 191 GB en aproximadamente 15 minutos, demostrando una alta eficiencia en el manejo de grandes volúmenes de datos.

Durante la ejecución se procesaron un total de 3 107 255 000 lecturas, las cuales fueron agregadas en 3 107 255 bins genómicos. A partir de la distribución de coberturas obtenida mediante el sketch KLL, se calcularon diversos cuantiles representativos, los cuales se resumen en la Tabla 1.

El uso de memoria del sistema se mantuvo constante y acotado durante toda la ejecución, siendo independiente del número total de bins procesados, lo que confirma el comportamiento esperado del enfoque basado en sketches probabilísticos.

Cuadro 1: Cuantiles de cobertura estimados mediante KLL para bins de 1000 bases y $K = 400$.

Cuantil	**Cobertura**
P1	598
P5	1523
P25	1850
P50 (Mediana)	2060
P75	2259
P95	2521
P99	2707

La comparación con un enfoque exacto basado en ordenamiento completo mostró que este último no escala adecuadamente a genomas completos, tanto en tiempo como en memoria, mientras que el sketch KLL mantiene un comportamiento estable.

La mediana de cobertura cercana a 2000 concuerda con la profundidad esperada del dataset, mientras que la separación entre los cuantiles superiores e inferiores evidencia la presencia de regiones con cobertura significativamente anómala, consistentes con posibles eventos de duplicación y deleción.

8.4. Detección de CNVs

Utilizando umbrales dinámicos definidos como fracciones de la mediana de cobertura (0,5× para deleciones y 1,5× para duplicaciones), se obtuvieron los siguientes resultados:

- Para runs de bins anómalos de longitud mayor o igual a 1:
  - 14 070 bins clasificados como deleciones.
  - 2 378 bins clasificados como duplicaciones.
- Para runs de bins anómalos de longitud mayor o igual a 5 (cromosoma 1):
  - 185 bins clasificados como deleciones.
  - 294 bins clasificados como duplicaciones.

Estos resultados evidencian que el método es capaz de identificar regiones con cobertura anómala de forma eficiente, priorizando la detección de eventos más consistentes mediante el uso de runs consecutivos.

9. Conclusiones

El presente trabajo abordó el problema de la detección eficiente de Variaciones en el Número de Copias (CNVs) en archivos de secuenciación de alta cobertura, proponiendo una solución basada en algoritmos de streaming y estructuras de datos probabilísticas.

Los resultados experimentales permiten afirmar que el uso del *Sketch KLL* constituye una alternativa viable y escalable frente a los métodos exactos tradicionales. Se demostró que esta estructura permite estimar los cuantiles de cobertura con una reducción drástica en el consumo de memoria, manteniendo un error de rank inferior al 0.3 % para valores moderados de K ($K = 400$). Esta precisión resultó suficiente para identificar posibles duplicaciones y deleciones.

Asimismo, la elección de un tamaño de bin de 1000 bases ofreció un balance óptimo entre resolución espacial y estabilidad estadística, minimizando el ruido local sin perder la capacidad de detectar eventos relevantes. La implementación en C++ con `htslib` validó la factibilidad técnica de procesar archivos BAM masivos sin la necesidad de cargar la totalidad de los datos en memoria como lo podría ser con otras herramientas para detectar cnvs.

Si bien el sistema demostró un rendimiento robusto, la validación experimental se restringió a los primeros 5 cromosomas del genoma humano debido a limitaciones de hardware por almacenamiento para procesar la totalidad del archivo WGS que contenga el genoma humano completo (500 GB). Aunque los resultados parciales sugieren que la escalabilidad del método se mantendría para el genoma completo, una validación exhaustiva con los 23 pares de cromosomas es necesaria para confirmar la consistencia global de los hallazgos. Por lo que se propone:

- Ejecutar código sobre la totalidad del un archivo .bam de un genoma humano completo en un entorno de alto rendimiento para evaluar el comportamiento del sketch KLL ante la variabilidad completa del genoma.
- Comparar la precisión de la solucion del conjunto de cnvs generados contra herramientas estándar de la industria (como CNVnator) para cuantificar la sensibilidad y especificidad del método propuesto.

- Explorar la posibilidad de aporte o optimización del KLL en una de las herramientas para la detección de cnvs.

10. Disponibilidad de Código y Reproducibilidad

Con el objetivo facilitar la replicación de los experimentos, todo el código fuente desarrollado en C++, los scripts de análisis y las instrucciones detalladas de ejecución se encuentran disponibles públicamente en el repositorio de GitHub (7) En este repositorio también se incluyen los resultados obtenidos durante la experimentación, permitiendo validar los hallazgos o utilizar la implementación como base para trabajos futuros.

Referencias

[1] Bonfield, J. K., et al. (2021). *HTSlib: C library for reading/writing high-throughput sequencing data*. GigaScience, 10(2), giab007.

[2] Apache DataSketches. *A software library of stochastic streaming algorithms*. `https://datasketches.apache.org/`

[3] De La Vega, F. M., et al. (2024). *Benchmarking of Germline Copy Number Variant Callers from Whole Genome Sequencing Data for Clinical Applications*. medRxiv (preprint). DOI: 10.1101/2024.07.12.24310338.

[4] National Institute of Standards and Technology (NIST). (2015). *Genome in a Bottle (GIAB): Ashkenazim Trio HG002 alignment (NovoAlign, GRCh38)*. Recuperado de: `https://github.com/genome-in-a-bottle/giab_data_indexes`

[5] Zook, J. M., et al. (Genome in a Bottle Consortium). *NIST HiSeq HG002 Homogeneity: High Coverage (300x) Raw Sequencing Data (FASTQ)*. National Institute of Standards and Technology. Disponible en: `https://ftp-trace.ncbi.nlm.nih.gov/giab/ftp/data/AshkenazimTrio/HG002_NA24385_son/NIST_HiSeq_HG002_Homogeneity-10953946/HG002_HiSeq300x_fastq/`

[6] UCSC Genome Browser Graphics. (2013). *Human Genome Assembly GRCh38/hg38*. Recuperado de: `http://hgdownload.soe.ucsc.edu/goldenPath/hg38/bigZips/`

[7] Repositorio proyecto. *Repositorio alojado en Github con código utilizado para su reproducibilidad y sus resultados presentados en el informe*. `https://github.com/diegoOgh/Proyecto---CNVs-analisis-con-KLL`