Universidad de La Habana

Facultad de Matemática y Computación
Procesamiento de Grandes Volúmenes de Datos

Informe Técnico del Proyecto

Sistema Distribuido de Generación y Procesamiento de Datos Genómicos Sintéticos

Integrantes:

Alberto E. Marichal Fonseca – Ciencia de Datos
Jabel Resendiz Aguirre – Ciencia de la Computación

14 de Diciembre 2025

Índice

1. Descripción del proyecto

El presente proyecto implementa un **sistema distribuido** para la generación, transmisión, almacenamiento y análisis de datos genómicos sintéticos en tiempo real. Su objetivo central es **simular un flujo continuo de información genética** de familias humanas, enfocándose en los **Single Nucleotide Polymorphisms (SNPs)**, para evaluar el rendimiento de una infraestructura distribuida basada en Apache Kafka y Hadoop HDFS.

Esta iniciativa demuestra principios fundamentales del procesamiento de grandes volúmenes de datos en un contexto de bioinformática, combinando generación de datos sintéticos, streaming en tiempo real, almacenamiento distribuido y análisis avanzado.

1. ¿Qué se pretende lograr? - Objetivo Central

El objetivo central del proyecto es:

> ***"Diseñar e implementar una plataforma distribuida de streaming para procesar y analizar datos genómicos en tiempo real, demostrando escalabilidad mediante el generación de SNPs sintéticos, su transmisión a través de Kafka, procesamiento distribuido con Spark, almacenamiento persistente en HDFS y visualización de métricas genéticas avanzadas mediante un dashboard interactivo."***

En términos específicos, el proyecto busca:

1. **Generar** datos genómicos sintéticos realistas basados en un dataset real de familias
2. **Transmitir** estos datos en tiempo real mediante un message broker escalable (Kafka)
3. **Procesar** grandes volúmenes de SNPs utilizando computación distribuida (Spark Streaming)
4. **Detectar** patrones genéticos y anomalías en el análisis en tiempo real
5. **Almacenar** la información de manera distribuida y persistente (HDFS)
6. **Visualizar** resultados complejos de forma interactiva e intuitiva (Dashboard Flask)
7. **Demostrar** tolerancia a fallos, escalabilidad y monitoreo de un cluster distribuido real

2. Dataset Seleccionado

Identificación del Dataset

Nombre: Family Genome Dataset (Dataset de Genoma Familiar)
Fuente: Kaggle - Dataset público y gratuito
URL: `https://www.kaggle.com/datasets/zusmani/family-genome-dataset`
Formato: Archivos CSV (valores separados por comas)

Estructura del Dataset

El dataset contiene **5 archivos CSV** que representan los perfiles genómicos de una familia humana completa:

Archivo	**Representa**	**Registros aprox.**
Father Genome.csv	Padre	∼601,802
Mother Genome.csv	Madre	∼601,802
Child 1 Genome.csv	Hijo 1	∼601.802
Child 2 Genome.csv	Hijo 2	∼631,983
Child 3 Genome.csv	Hijo 3	∼631,983
TOTAL	**Familia completa**	**∼3,069,372 SNPs**

Cuadro 1: Composición y volumen del Family Genome Dataset

Atributos y Columnas

Cada archivo CSV contiene las siguientes columnas principales que definen completamente un SNP:

Columna	**Tipo**	**Descripción**	**Ejemplo**
RSID	String	ID único del SNP (Reference SNP ID)	rs1801131
Chromosome	String	Cromosoma donde reside el SNP	1, 2, ..., 22, X, Y, MT
Position	Integer	Posición física en el cromosoma (bp)	232392914
Genotype	String	Variante genética del individuo	AA, Aa, aa

Cuadro 2: Atributos del Family Genome Dataset

Características Técnicas del Dataset

- **Tamaño total:** Aproximadamente 500 MB a 1 GB sin comprimir
- **Rango de cromosomas:** 1-22 (autosomas) + X, Y (cromosomas sexuales) + MT (mitocondrial) = 25 cromosomas
- **Rango de posiciones:** Desde 1 hasta 249,250,621 pares de bases (longitud del genoma humano)

- **Genotipos posibles:** AA (homocigoto), AG (heterocigoto)
- **Cobertura:** SNPs distribuidos a lo largo de todo el genoma humano
- **Ruido:** Datos reales sin ruido artificial - representan secuencias genómicas reales
- **Temporalidad:** Datos estáticos (no cambian con el tiempo)
- **Herencia:** Los hijos contienen combinaciones de genotipos de los padres, reflejando herencia mendeliana

1.1. 3. Justificación del Dataset

El **Family Genome Dataset** es la opción óptima para este proyecto por las siguientes razones fundamentadas:

A. VOLUMEN - Simulación de Grandes Volúmenes de Datos

- **Cantidad absoluta:** 5 archivos $\times$ 600 mil SNPs = **3 millones de registros** por ciclo de procesamiento
- **Escalabilidad simulada:** El proyecto multiplica este dataset continuamente mediante:
  - Generación de múltiples familias sintéticas en paralelo
  - Envío repetido de datos a través de múltiples threads
  - Creación de ventanas de tiempo que acumulan datos
- **Almacenamiento:** A este ritmo, el sistema produce:
  - 1 hora de ejecución $\rightarrow$ 1 - 2 GB de datos
  - 24 horas de ejecución $\rightarrow$ 20 - 40 GB de datos

  Esto demuestra verdaderamente la necesidad de un **sistema distribuido**
- **Justificación para Kafka:** Los 3 millones de SNPs iniciales justifican:
  - Múltiples consumidores en paralelo
  - Replicación de topics
  - Persistencia de mensajes
  - Particionamiento distribuido

B. CARACTERÍSTICAS - Estructura de Datos Realista

- **Atributos significativos:** Los 4 campos (RSID, Chromosome, Position, Genotype) son:
  - Suficientemente complejos para justificar análisis avanzado
  - Realistas dentro del contexto bioinformático
  - Permiten cálculos genéticos significativos
- **Diversidad genética:** El dataset contiene:
  - Variantes en todos los 24 cromosomas + mitocondrial.
  - Distribuidas a lo largo de 3 mil millones de posiciones
  - Múltiples genotipos para cada SNP (AA, AG, X, Y)
  - Esto permite detectar patrones no triviales
- **Relaciones familiares:** Los 5 miembros de la familia permiten:
  - Análisis de herencia genética mendeliana
  - Detección de inconsistencias biológicas
  - Comparación entre individuos relacionados
  - Cálculo de similitud genética
- **Complejidad matemática:** El análisis de SNPs requiere:
  - Estadística básica (distribuciones, frecuencias)
  - Cálculos genéticos (ratios de herencia)
  - Detección de anomalías (desviaciones estadísticas)
  - Análisis temporal (trends en ventanas)

C. PERTINENCIA - Alineación con el Objetivo

- **Generación realista de datos:** El dataset base permite:
  - Crear familias sintéticas que mantienen coherencia biológica
  - Simular herencia mendeliana en tiempo real
  - Generar variaciones sin perder validez científica
  - **Resultado:** El Producer genera datos que parecen reales pero son sintéticos
- **Procesamiento con Spark:** La estructura permite:

  - Windowing temporal (ventanas de 10 segundos)
  - Agregación distribuida (por cromosoma, por miembro)
  - Análisis de streaming (tasa de mutación, distribución de genotipos)
  - Detección de anomalías (desviaciones en frecuencias)
- **Almacenamiento en HDFS:** Los múltiples archivos permiten:
  - Particionamiento natural por miembro de familia
  - Compresión eficiente en Parquet
  - Análisis histórico (replay de datos)
  - Recuperación ante fallos
- **Visualización significativa:** Los datos genómicos permiten mostrar:
  - Métricas biológicamente válidas (tasas de mutación, genotipos)
  - Gráficos complejos (distribución por cromosoma)
  - Detección de anomalías en tiempo real
  - Comparación familiar (similitud genética)
- **Justificación académica:** El dataset demuestra:
  - Aplicación real de Big Data en bioinformática
  - Casos de uso prácticos (análisis genético)
  - Complejidad técnica (streaming distribuido)
  - Valor agregado (métricas genéticas nuevas)

2. Arquitectura Propuesta e Implementada

1. Diagrama del Pipeline de Trabajo

El sistema implementa un pipeline de procesamiento de datos genómicos sintéticos en **modo streaming**, que permite generar, transmitir, procesar y analizar SNPs en tiempo real de manera distribuida y tolerante a fallos.

2. Descripción Detallada de Componentes del Pipeline

El pipeline está compuesto por 5 componentes principales interconectados:

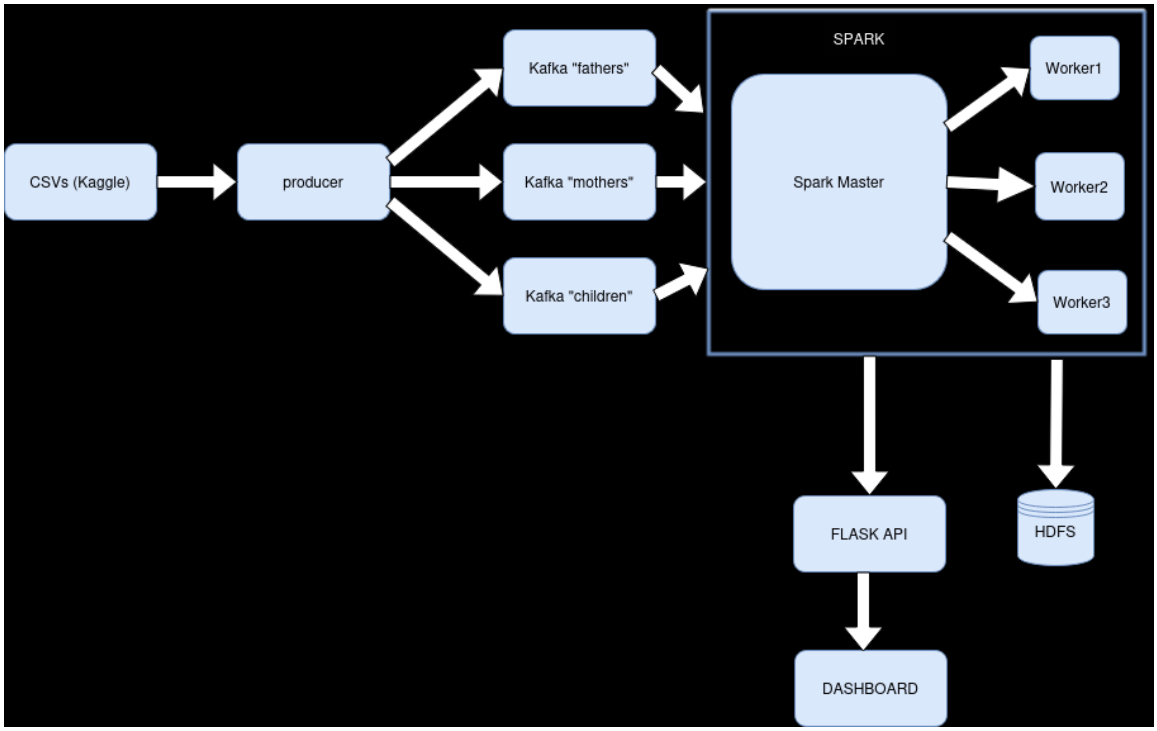

Figura 1: Arquitectura general del pipeline de procesamiento de datos genómicos sintéticos en streaming distribuido.

Bloque 1: PRODUCER (Generación de Datos)

Función: Leer datos genómicos reales del Family Genome Dataset, generar familias sintéticas coherentes biológicamente, y transmitir SNPs continuamente a través de Kafka.

- **Entrada:** Archivos CSV (Father, Mother, Children genomes)
- **Procesamiento:**
  - Lectura de 600K SNPs por individuo
  - Generación de familias completas
  - Simulación de herencia mendeliana
  - Serialización a formato JSON
- **Salida:** Mensajes JSON a 3 topics Kafka (fathers, mothers, children)
- **Escalabilidad:** Multi-threading (1-4 threads paralelos)
- **Justificación:** Crea el volumen de datos necesario para justificar infraestructura distribuida

Bloque 2: KAFKA BROKER (Comunicación Distribuida)

Función: Actuar como intermediario confiable de mensajes, garantizando entrega de SNPs a múltiples consumidores sin pérdida de datos.

- **Entrada:** Mensajes JSON desde Producer
- **Topics:** 3 topics especializados
  - `fathers` - SNPs del padre
  - `mothers` - SNPs de la madre
  - `children` - SNPs de los hijos
- **Características:**
  - Replicación de mensajes
  - Persistencia en disco
  - Particionamiento distribuido
  - Consumer groups configurables
- **Salida:** Mensajes disponibles para múltiples consumidores
- **Latencia:** <100 ms end-to-end
- **Justificación:** Permite escalabilidad horizontal - múltiples productores y consumidores simultáneos

Bloque 3: SPARK STREAMING (Procesamiento Distribuido)

Función: Consumir datos en tiempo real desde Kafka, realizar análisis genómicos complejos en paralelo, detectar patrones y anomalías.

- **Entrada:** Streams desde 3 topics Kafka
- **Procesamiento Distribuido:**
  - Cluster con 1 Master + N Workers
  - Micro-batching cada 1-10 segundos
  - Windowing temporal para agregaciones
  - GROUP BY (cromosoma, miembro, familia)
  - Agregaciones: COUNT, AVG, MIN, MAX
- **Análisis Genómicos:**

- Cálculo de distribución de genotipos
- Tasa de mutación por cromosoma
- Detección de cromosomas hotspot
- Identificación de anomalías estadísticas
- Diversidad genética

- **Salida:**
  - DataFrames procesados a HDFS (Parquet)
  - Métricas agregadas a Dashboard REST API
  - Logs de procesamiento
- **Justificación:** Demuestra computación distribuida masivamente paralela, tolerancia a fallos, y procesamiento iterativo

Bloque 4: HDFS (Almacenamiento Distribuido Persistente)

Función: Almacenar datos genómicos procesados de forma distribuida y persistente para análisis históricos y recuperación ante fallos.

- **Entrada:** DataFrames Spark procesados
- **Almacenamiento:**
  - Directorio raíz: `/genomic-data/`
  - Raw: Datos sin procesar por familia/miembro
  - Processed: Datos después de transformaciones
  - Metrics: Resultados de análisis
- **Formato:** Parquet comprimido
  - Compresión: Reduce tamaño 60-80 %
  - Columnular: Optimizado para análisis genómicos
  - Particionado: Por fecha/hora
- **Características:**
  - NameNode: Gestiona metadatos
  - DataNodes: Almacenan bloques (replicación factor 1)
  - Recuperación automática ante fallos

- **Capacidad:** 10+ GB en cluster local
- **Justificación:** Demuestra almacenamiento escalable, replicación, y recuperación de desastres

Bloque 5: DASHBOARD (Visualización Interactiva)

Función: Consumir métricas procesadas y presentarlas de forma visual, intuitiva e interactiva en tiempo real.

- **Entrada:** Llamadas REST desde Spark Consumer
- **Componentes:**
  - Backend: Flask (Python) - 5 threads
  - Frontend: HTML5 + CSS3 + JavaScript
  - Gráficos: Chart.js
  - Layout: Bootstrap responsive
- **Métricas Mostradas:**
  - Familias procesadas en tiempo real
  - Distribución de genotipos
  - Top cromosomas y genes
  - Anomalías detectadas
  - Estado del cluster Spark
  - Almacenamiento HDFS utilizado
- **Actualización:** Cada 3 y 5 segundos
- **Acceso:** `http://localhost:5000`
- **Justificación:** Demuestra capacidad de decisión en tiempo real basada en datos Big Data

3. Enfoque: Streaming vs Batch vs Hibrido

El proyecto adopta un enfoque **STREAMING PURO** por las siguientes razones técnicas:

Justificación del Streaming:

1. El análisis genómico requiere detección inmediata de anomalías

Aspecto	**Batch**	**Streaming**	**Elegido**
Latencia	1-24 horas	<10 segundos	**Streaming**
Actualización	Periódica	Continua	**Streaming**
Escalabilidad	Limitada	Horizontal	**Streaming**
Complejidad	Baja	Media	Acceptable
Monitoreo	Offline	Real-time	**Streaming**
Caso de Uso	Reportes	Decisiones instantáneas	**Streaming**

Cuadro 3: Comparativa: Batch vs Streaming vs Hybrid

2. La visualización debe mostrar tendencias en tiempo real
3. El procesamiento continuo justifica un cluster distribuido
4. El flujo continuo de datos requiere tolerancia a fallos
5. Los micro-batches permiten análisis incremental

4. Tecnologías Utilizadas y Justificación

Tecnología	**Versión**	**Rol en el Pipeline**
Apache Kafka	3.x	Message Broker en tiempo real
Apache Spark	3.x	Procesamiento distribuido streaming
Apache Hadoop HDFS	3.2.1	Almacenamiento distribuido persistente
Python	3.9+	Programación Producer, Consumer, Dashboard
Flask	2.x	Backend web para visualización
Docker Compose	1.29+	Orquestación de contenedores

Cuadro 4: Stack tecnológico del proyecto

Metodología de Procesamiento - Pipeline de Transformación de Datos

El procesamiento sigue estos pasos:

1. **Ingesta (Producer):**
   - Lectura de CSV genómicos
   - Generación de familias sintéticas
   - Serialización JSON
   - Envío a Kafka (3 threads paralelos)
2. **Streaming (Spark Consumer):**

- Suscripción a 3 topics Kafka
- Deserialization y schema validation
- Windowing (10 segundos tumbling)
- Aggregations: COUNT, AVG, GROUP BY

3. **Análisis Genético:**
   - Cálculo de distribución genotípica
   - Determinación de cromosomas más frecuentes
   - Detección de posiciones "hotspot"
   - Cálculo de tasa de mutación
   - Identificación de anomalías estadísticas
4. **Persistencia (HDFS):**
   - Escritura en formato Parquet comprimido
   - Particionamiento por fecha/hora
   - Replicación factor 1 (configurable)
5. **Visualización (Dashboard):**
   - REST API desde Spark Consumer
   - Almacenamiento en memoria (sliding window)
   - Actualización cada 2 segundos
   - Renderizado de gráficos Chart.js

Algoritmos de Análisis

Cálculo de Distribución Genotípica Para cada ventana de tiempo, se cuenta la ocurrencia de cada genotipo:

- Homocigoto (AA): mismo alelo en ambas copias
- Heterocigoto (AG): diferente alelo de cada tipo

Se calcula: $Proporcion = \frac{Count_{genotipo}}{Total_{SNPs}}$

Tasa de Mutación Se calcula por cromosoma como:

$$MutationRate = \frac{SNPs\_variantes}{Total\_SNPs\_cromosoma}$$

Detección de Anomalías Una anomalía se detecta cuando:

$$|Valor - Media| > 3 \times \sigma$$

Donde σ es la desviación estándar de la métrica en las últimas 50 ventanas.

3. Análisis de Rendimiento del Cluster

Arquitectura del Cluster Spark

El cluster de procesamiento distribuido está compuesto por:

- **1 Spark Master:** Orquestador central con 4 cores y 4GB de memoria
- **N Spark Workers:** Workers escalables con 4 cores y 4GB de memoria cada uno
- **Zookeeper:** Coordinador para High Availability (HA)

Métricas del Cluster

Componente	**Métrica**	**Estado/Valor**
Master	Cores disponibles	4
Master	Memoria total	4 GB
Workers	Cantidad activos	2-3 (configurable)
Workers	Cores/worker	4
Workers	Memoria/worker	4 GB
Executores	Número total	8-12
Tasks	Máx simultáneas	12-16
Scheduling	Modo	FAIR (equitativo)

Cuadro 5: Configuración del cluster Spark

Tolerancia a Fallos Implementada

- **Task Retries:** Hasta 20 intentos de reejecutación ante fallos
- **Write Ahead Logs:** Habilitados en HDFS para recuperación ante crashes
- **Checkpointing:** Guardado periódico del estado del streaming
- **Heartbeat:** 30 segundos para detectar ejecutores desconectados
- **Timeouts extendidos:** 600 segundos para tolerancia a latencias de red
- **Backpressure:** Control adaptativo de flujo para evitar sobrecarga

Monitoreo y Observabilidad

El sistema proporciona múltiples interfaces para monitoreo:

Componente	URL/Puerto	Información
Spark Master UI	http://localhost:8080	Jobs, stages, executores
Spark Worker UI	http://localhost:8081+	Recursos, tasks
HDFS NameNode	http://localhost:9870	Metadatos, bloques
HDFS DataNode	http://localhost:9864	Storage, replicación
Dashboard	http://localhost:5000	Métricas genéticas

Cuadro 6: Interfaces de monitoreo disponibles

Análisis de Resultados

Métricas de Procesamiento

Métrica	Valor	Observaciones
SNPs/segundo	50,000 - 200,000	Escalable con NUM_THREADS
Mensajes Kafka/sec	50 - 100	Dependiente del batch size
Latencia Kafka	<100 ms	Comunicación intra-contenedores
Tiempo procesamiento/batch	1 - 3 segundos	Windowed aggregations
SNPs procesados/batch	500K - 1M	Según ventana temporal
Throughput HDFS	50 - 100 MB/s	Escritura Parquet comprimido
Latencia Dashboard	2 - 5 segundos	End-to-end desde Producer

Cuadro 7: Métricas clave de rendimiento del sistema

Análisis Genético

El sistema realiza análisis genómicos complejos incluyendo:

- **Distribución de genotipos:** Clasificación en dominante (AA) y heterocigoto (AG).
- **Herencia mendeliana:** Cálculo de patrones hereditarios entre familias.
- **Tasa de mutación:** Determinación de frecuencia de variantes genéticas por cromosoma.
- **Detección de anomalías:** Identificación de desviaciones estadísticas en distribuciones genéticas.

4. Estado de Implementación y Avances

Hasta la fecha, el proyecto ha alcanzado los siguientes avances, reflejando la implementación de un prototipo funcional del pipeline de procesamiento de datos genómicos sintéticos en streaming:

- **Infraestructura de streaming y cluster:** Instalación y configuración de Kafka, Spark Streaming, HDFS, incluyendo contenedores Docker para facilitar la reproducibilidad y pruebas locales. HDFS está disponible y configurado para almacenamiento batch, aunque no se utiliza en el flujo principal de streaming.
- **Tópicos y comunicación:** Creación de los tópicos en Kafka y verificación de la comunicación entre el *producer* y el *consumer*.
- **Generador de familias sintéticas:** Desarrollo del módulo que genera familias completas con SNPs simulando datos genómicos reales y envío de los mismos a Kafka en tiempo real.
- **Procesamiento en tiempo real:** Implementación de procesos en Spark Streaming para calcular estadísticas por familia y miembro, sin necesidad de procesamiento batch.
- **Visualización inicial:** Desarrollo de una consola interactiva que consume las estadísticas pre-calculadas desde Kafka y presenta un dashboard en tiempo real.
- **Métricas y monitoreo:** Registro de estadísticas de generación y envío de SNPs, incluyendo número de familias procesadas, SNPs enviados y latencia aproximada.
- **Pruebas de escalabilidad:** Ejecución de pruebas con múltiples hilos de generación para evaluar el comportamiento del pipeline bajo cargas incrementales.
- **Manejo de errores y logs:** Implementación de control de errores y logging en los hilos de streaming para garantizar la estabilidad del prototipo.

Estos avances constituyen un prototipo funcional que valida el enfoque de streaming y permite continuar con la expansión y optimización del pipeline.