Tarea 2 - Histograma de palabras multi-hilo

José Nicolás Olmedo Cabrera, jose.olmedo@alumnos.uv.cl

1. Introducción

En esta tarea, nos adentraremos en la implementación de una aplicación que realiza un histograma de palabras de un archivo de texto determinado utilizando multi-threading. Un histograma de palabras, en términos simples, es una representación que cuenta la ocurrencia de palabras en un texto. Esta aplicación es particularmente útil para analizar grandes volúmenes de texto y entender con rapidez qué palabras son las más recurrentes, lo que tiene aplicaciones prácticas en áreas como análisis de texto, procesamiento de lenguaje natural y búsqueda de información.

El objetivo principal es migrar una solución secuencial dada a una versión multi-thread para aprovechar el poder del procesamiento paralelo y, por ende, mejorar el rendimiento. Es importante señalar que la solución multi-thread debe ser consistente con la versión original, es decir, ambos métodos deben producir el mismo resultado. Además, la tarea implica la integración con herramientas de programación en C++17, tales como la librería getopt, para procesar argumentos de la línea de comandos, asegurando así una interacción efectiva y flexible con el usuario final.

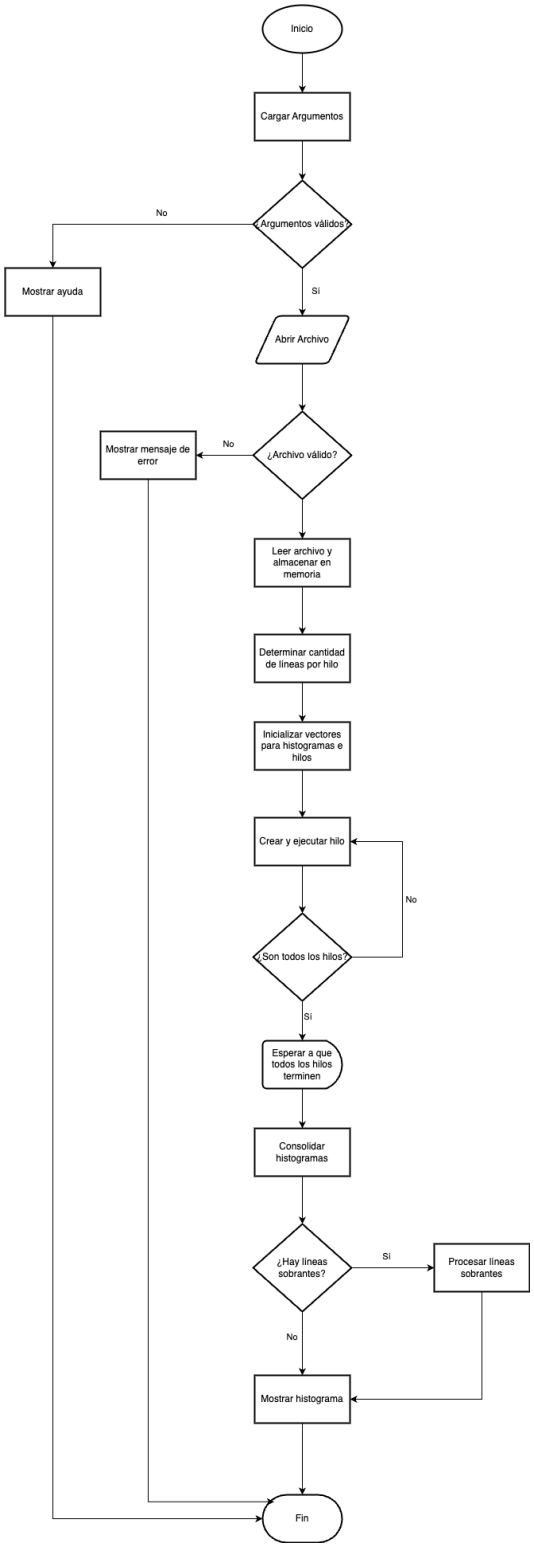
2. Materiales y Métodos

Se contó con varios materiales y herramientas que facilitaron la adaptación y mejora del código base proporcionado.

- **Código Base:** Se partió de un código fuente preexistente que ofrecía una solución secuencial para la generación del histograma de palabras. Este código estableció las bases funcionales sobre las cuales se diseñó y estructuró la versión multi-hilo.
- **Entorno de Desarrollo:** Para la edición, visualización y adaptación del código, se utilizó el entorno de desarrollo integrado (IDE) Visual Studio Code.
- **Repositorio en GitHub:** Con el objetivo de mantener un control de versiones y facilitar la colaboración y retroalimentación, el código se alojó en un repositorio de GitHub.

3. Resultados

Diagrama de Flujo. Ver diagrama completo aquí.



El proceso comienza con el inicio del programa. Lo primero que hace es cargar los argumentos proporcionados al ejecutarlo. Se realiza una verificación para determinar si los argumentos proporcionados son válidos. En caso de no serlo, el programa muestra un mensaje de ayuda para el usuario y termina su ejecución.

Si los argumentos son válidos, el programa intenta abrir el archivo especificado. Si encuentra algún problema al intentar abrir este archivo (por ejemplo, si el archivo no existe o no tiene los permisos adecuados), muestra un mensaje de error al usuario y finaliza.

Una vez que el archivo se abre con éxito, se procede a leer su contenido y almacenarlo en memoria para su posterior procesamiento. Luego, se determina la cantidad de líneas que cada hilo debe procesar basándose en el número total de líneas del archivo y el número de hilos especificado.

Con esta información, se inicializan estructuras de datos para almacenar histogramas individuales y se crean los hilos correspondientes. Cada hilo se encarga de procesar un segmento específico del archivo y generar su histograma de palabras. Una vez que todos los hilos han completado su tarea, el programa consolida los histogramas individuales en uno global.

Después de la consolidación, el programa verifica si hay líneas en el archivo que no fueron procesadas por los hilos (esto podría ocurrir si el número de líneas no es divisible exactamente por el número de hilos). Si hay líneas sobrantes, el programa las procesa de forma adicional.

Finalmente, con el histograma global completo, el programa muestra este histograma al usuario y finaliza su ejecución.

4. Discusión y conclusiones

La tarea de implementar un histograma de palabras utilizando un enfoque multi-hilo ha sido un ejercicio revelador sobre la importancia y eficacia del procesamiento paralelo en la era actual. Con el auge de los datos y la necesidad de procesar grandes volúmenes de información en tiempos reducidos, es esencial adoptar técnicas que optimicen y aceleren dichos procesos.

Al adaptar una solución secuencial a una versión multi-hilo, no solo enfrentamos el desafío técnico de dividir y distribuir el trabajo, sino que también nos vimos obligados a garantizar la coherencia y precisión del resultado final. Esto subraya la necesidad de un diseño cuidadoso y una implementación meticulosa cuando trabajamos con concurrencia.