Pedro Archila - 202421572

Requerimiento 1

Descripción

Breve descripción de como abordaron la implementación del requerimiento

|  |  |
| --- | --- |
| Entrada | Catalogo donde se encuentran todos los datos del csv, fecha inicial para el rango de búsqueda, fecha final para el rango de búsqueda. |
| Salidas | Lista de los crímenes que fueron cometidos entre las fechas estipuladas. Cada crimen muestra su código de identificación de reporte, la fecha en que ocurrió, la hora en que ocurrió, el nombre del área, el código del crimen y la dirección del crimen. |
| Implementado (Si/No) | Si - Grupal |

Análisis de complejidad

Análisis de complejidad de cada uno de los pasos del algoritmo

|  |  |
| --- | --- |
| Pasos | Complejidad |
| Paso 1 | O(1). Transforma las fechas a el formato datetime para que puedan ser comparadas |
| Paso 2 | O(1). Busca en el mapa del catálogo el árbol que representa las fechas en las que ocurre un crimen |
| Paso 3 | O(Log N + M). Donde N es la altura del árbol y M es la cantidad de nodos en el rango, hace la función values() de RBT para sacar los valores de los nodos cuyas llaves cumplen el rango de fechas. |
| Paso 4 | O(M^2) Se recorre toda la lista de valores la cuales, compuesta por otras listas, las cuales también se recorre para sacar cada dato y ponerlo en una nueva lista |
| Paso 5 | O(1). Revisa si la lista con los datos está vacía o no, si está vacía retorna None, si no, continúa. |
| Paso 6 | O(M). Pasa por cada uno de los datos y crea un mapa utilizando únicamente la fecha, el área y el índice. Se añade esta tabla de hash a una lista. |
| Paso 7 | O(Log M). Ordena la lista de mapas pequeños usando merge sort |
| Paso 8 | O(M). Se crea una lista donde se guardan los índices de los valores ya todo ordenado. |
| Paso 9 | O(M). Usando cada índice se crea una nueva lista y se mete el valor necesario de cada una de las tablas de hash en una lista final. |
| TOTAL | O(M^2 + 3M + Log (M) + Log (M+N)) en el peor de los casos que es cuando el rango de fechas es muy largo o cubre la mayor cantidad de los datos. O(Log(N)) en el mejor caso que es cuando el rango de fechas cubre muy pocos datos, esto haría que N sea mucho más grande que M por lo tanto se puede asumir únicamente N. Esto se podría mejorar ordenando directamente la lista completa en vez de hacer nuevos mapas y listas para poder ordenarlo. |

Pruebas Realizadas

Las pruebas son hechas con los siguientes datos:

•⁠ ⁠Fecha inicial: 2021/01/01

•⁠ ⁠⁠Fecha final: 2021/05/01

Se utilizan en el requerimiento las librerías de array list, sepárate chaining y RBT

Características del computador:

•⁠ ⁠8GB

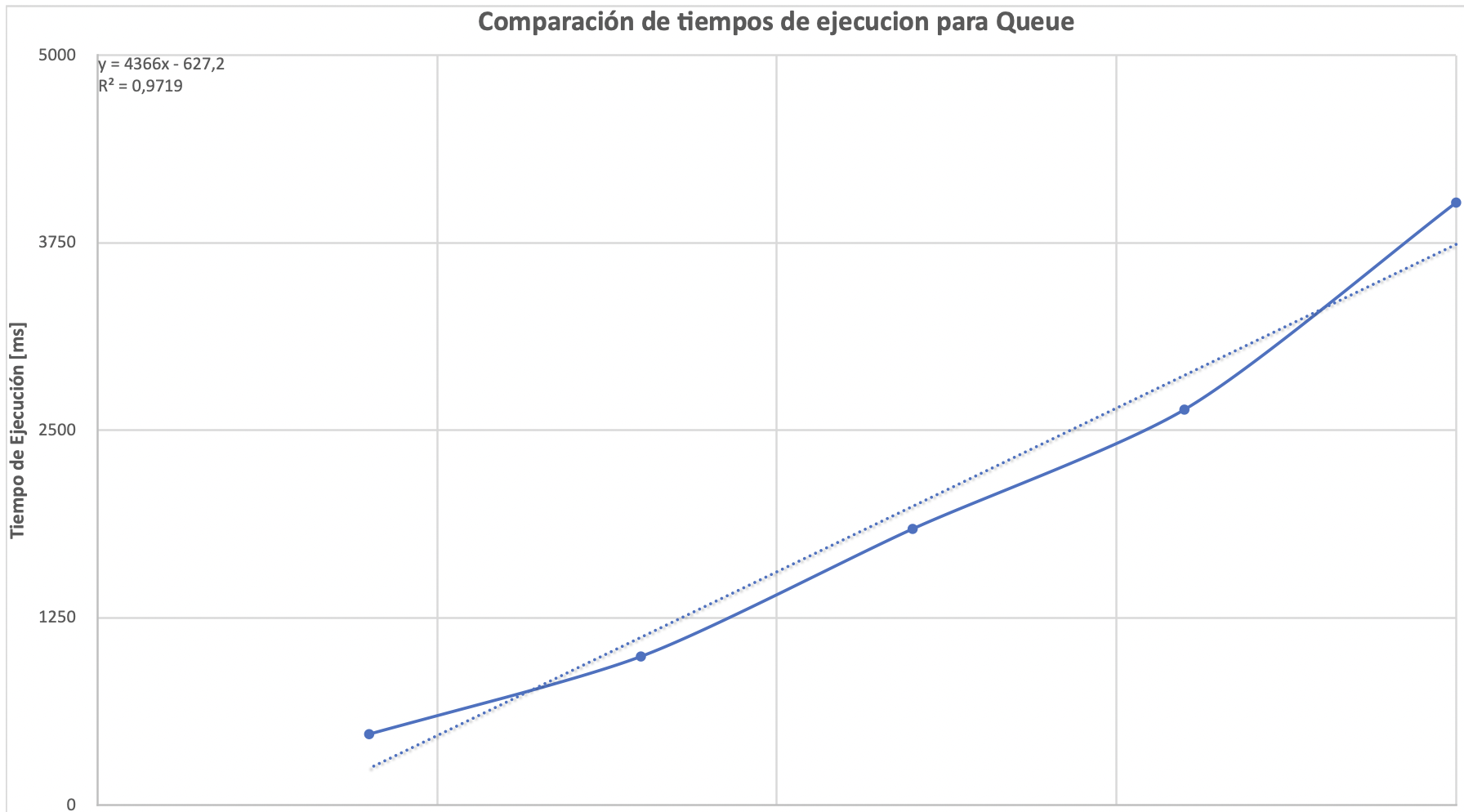
•⁠ ⁠⁠Chip M1

•⁠ ⁠⁠macOS sequoia 15.1

|  |  |
| --- | --- |
| Entrada | Tiempo (ms) |
| 20% | 473,91 |
| 40% | 991,72 |
| 60% | 1842,04 |
| 80% | 2637,15 |
| 100% | 4017,21 |

Gráficas

Las gráficas con la representación de las pruebas realizadas.



Análisis

El comportamiento de la gráfica es entre lineal y cuadrático, esto se debe a la complejidad logarítmica de algunas de las operaciones como el ordenamiento y la búsqueda en árboles. Además, el uso de dos for para recorrer por completo la lista de todos los datos entre las fechas dadas hace que con muchos datos la gráfica crezca rápidamente. En términos generales, los puntos están muy cerca a la línea de tendencia dando una apariencia de complejidad O(N).

Requerimiento 2

Descripción

Breve descripción de como abordaron la implementación del requerimiento

|  |  |
| --- | --- |
| Entrada | Catálogo donde se encuentran todos los datos del csv, fecha de reporte inicial del rango a revisar, fecha de reporte final del rango a revisar |
| Salidas | Lista de los crímenes graves resuelto que fueron reportados dentro del rango de fechas, si hay más de 10 crímenes que cumplen el filtro se entrega una lista de los primeros 5 y los últimos 5 crímenes en orden cronológico en cuanto a la fecha cuando fueron reportados. Cada crimen debe contener su identificador de reporte, fecha y hora de ocurrencia, número del area, subarea, gravedad del caso, código del crimen, estado del caso. Se imprimen también la fecha del reporte y el nombre del área para revisar que esté correctamente ordenado. |
| Implementado (Si/No) | Si - Grupal |

Análisis de complejidad

Análisis de complejidad de cada uno de los pasos del algoritmo

|  |  |
| --- | --- |
| Pasos | Complejidad |
| Paso 1 | O(1). Transforma las fechas a el formato datetime para que puedan ser comparadas |
| Paso 2 | O(1). Busca en el mapa del catálogo el árbol que representa las fechas en las que se reporta un crimen |
| Paso 3 | O(Log N + M). Donde N es la altura del árbol y M es la cantidad de nodos en el rango, hace la función values() de RBT para sacar los valores de los nodos cuyas llaves cumplen el rango de fechas. |
| Paso 4 | O(M^2) Se recorre toda la lista de valores la cuales, compuesta por otras listas, las cuales también se recorre para sacar cada dato y ponerlo en una nueva lista si es un crimen el cual no continúa en investigación (IC) y es muy grave (1). |
| Paso 5 | O(1). Revisa si la lista con los datos está vacía o no, si está vacía retorna None, si no, continúa. |
| Paso 6 | O(M). Pasa por cada uno de los datos y crea un mapa utilizando únicamente la fecha, el área y el índice. Se añade esta tabla de hash a una lista. |
| Paso 7 | O(Log M). Ordena la lista de mapas pequeños usando merge sort |
| Paso 8 | O(M). Se crea una lista donde se guardan los índices de los valores ya todo ordenado. |
| Paso 9 | O(M). Usando cada índice se crea una nueva lista y se mete el valor necesario de cada una de las tablas de hash en una lista final. |
| TOTAL | O(M^2 + 3M + Log (M) + Log (M+N)) en el peor de los casos que es cuando el rango de fechas es muy largo o cubre la mayor cantidad de los datos. O(Log(N)) en el mejor caso que es cuando el rango de fechas cubre muy pocos datos, esto haría que N sea mucho más grande que M por lo tanto se puede asumir únicamente N. Esto se podría mejorar ordenando directamente la lista completa en vez de hacer nuevos mapas y listas para poder ordenarlo. |

Pruebas Realizadas

Las pruebas son hechas con los siguientes datos:

•⁠ ⁠Fecha inicial: 2021/01/01

•⁠ ⁠⁠Fecha final: 2021/05/01

Se utilizan en el requerimiento las librerías de array list, sepárate chaining y RBT

Características del computador:

•⁠ ⁠8GB

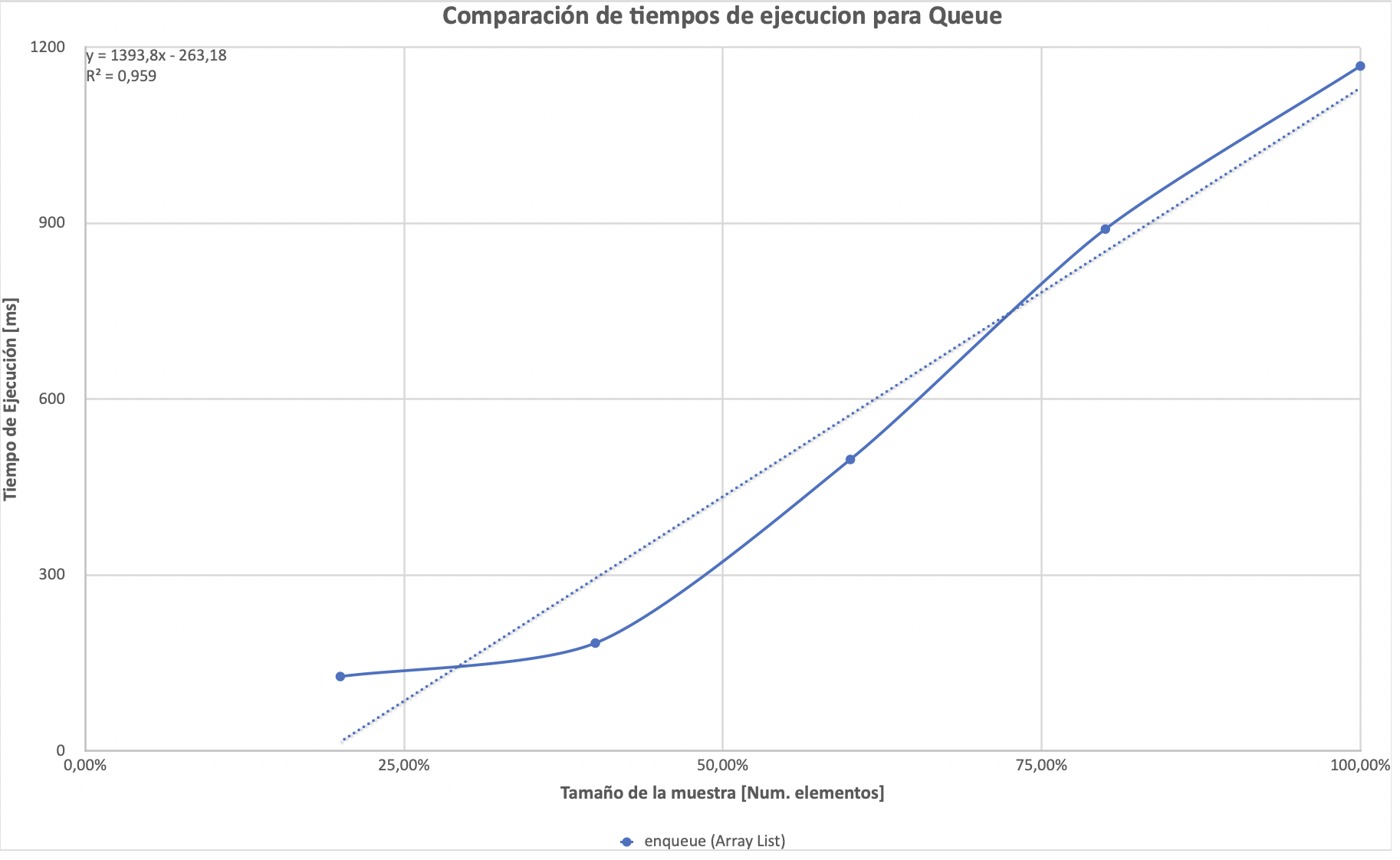
•⁠ ⁠⁠Chip M1

•⁠ ⁠⁠macOS sequoia 15.1

|  |  |
| --- | --- |
| Entrada | Tiempo (ms) |
| 20% | 126,94 |
| 40% | 183,92 |
| 60% | 497,01 |
| 80% | 889,75 |
| 100% | 1167,79 |

Gráficas

Las gráficas con la representación de las pruebas realizadas.



Análisis

Viendo la gráfica y en análisis de complejidad es posible decir que con pocos datos la función actúa con una complejidad temporal logarítmica, pues no aumenta mucho el tiempo que se demora, sin embargo, cuando los datos empiezan a ser muchos, el comportamiento de la gráfica se vuelve más lineal, lo cual tiene sentido gracias a las operaciones como merge\_sort y la búsqueda de lo valores dentro del rango.

Requerimiento 3

Descripción

Breve descripción de como abordaron la implementación del requerimiento

|  |  |
| --- | --- |
| Entrada | Catalogo donde se encuentran todos los datos del csv, Numero N de elementos que se desean analizar y el nombre del área de la cual se quieren ver los crimenes. |
| Salidas | Una lista de los primeros N crímenes en orden cronológico dentro del área dada. Cada crimen debe presentar su identificador de reporte, Fecha y hora que ocurrió, número de área, subárea, gravedad del caso, código del crimen, estado del crimen y la dirección |
| Implementado (Si/No) | Si - Pedro Archila |

Análisis de complejidad

Análisis de complejidad de cada uno de los pasos del algoritmo

|  |  |
| --- | --- |
| Pasos | Complejidad |
| Paso 1 | O(1). Saca el árbol correspondiente al nombre del área de la tabla de hash inicial |
| Paso 2 | O(Log N). Busca dentro del RBT de áreas el nombre del área que se desea encontrar y toma la lista de tablas de hash que se encuentra en ese nodo. |
| Paso 3 | O(1). Revisa si la lista está vacía o no, si está vacía retorna None, si tiene elementos continúa el código. |
| Paso 4 | O(M). Revisa todos los elementos de la lista de tablas de hash y saca de cada uno la fecha del reporte y el índice donde se encuentra. Mete cada tabla de hash a una lista |
| Paso 5 | O(Log M). Ordena la lista con respecto a la fecha de reporte usando merge sort. |
| Paso 6 | O(M). Recorre nuevamente la lista sacando cada valor de la tabla de hash en la llave “index” y mete ese valor a una nueva lista. |
| Paso 7 | O(M). Se recorre toda la lista de indices y por cada uno crea una lista donde se guarda toda la información de un crimen y luego esa lista se mete en un array. |
| Paso 8 | O(K). Se crea una sublista de la lista grande con la cantidad K de datos introducida como parámtero |
| TOTAL | O(3M + Log M + Log N). Sin embargo, M será mucho más pequeño que N por la  cantidad de datos dentro del csv. Por lo tanto la complejidad se podría considerar como O(Log N). |

Pruebas Realizadas

Las pruebas son hechas con los siguientes datos:

•⁠ ⁠N: 10

•⁠ Área = Pacific

Se utilizan en el requerimiento las librerías de array list, sepárate chaining y RBT

Características del computador:

•⁠ ⁠8GB

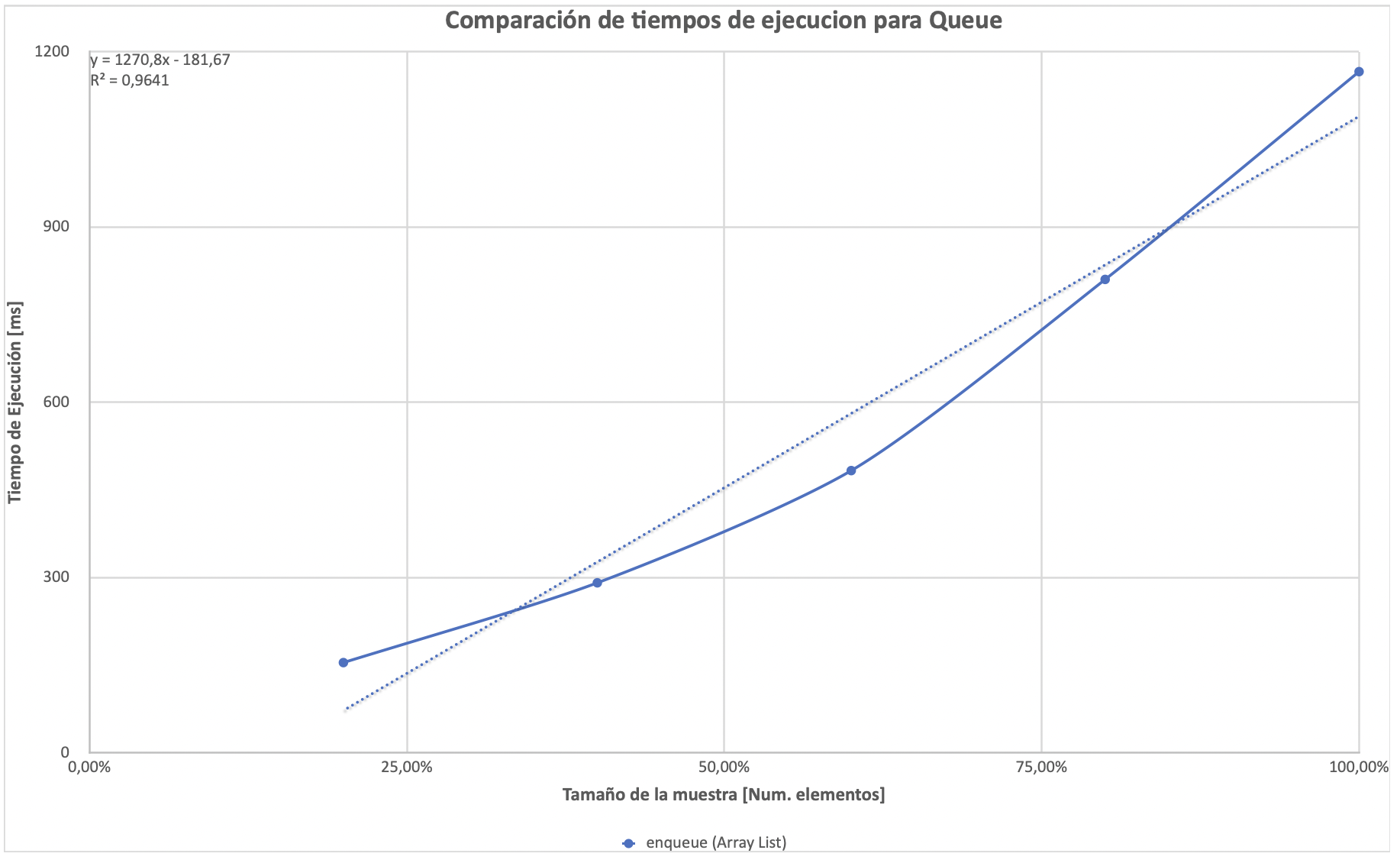
•⁠ ⁠⁠Chip M1

•⁠ ⁠⁠macOS sequoia 15.1

|  |  |
| --- | --- |
| Entrada | Tiempo (ms) |
| 20% | 154,41 |
| 40% | 290,93 |
| 60% | 482,94 |
| 80% | 810,08 |
| 100% | 1165,62 |

Gráficas

Las gráficas con la representación de las pruebas realizadas.



Análisis

Según la gráfica, el comportamiento de la función es cuadrática, sin embargo, según el análisis de complejidad y las operaciones utilizadas en la función, esta se debería comportarse de forma logarítmica. Esto se puede deber a la parte en la complejidad temporal donde aparece 3 veces M gracias a los for utilizados para organizar de manera correcta la lista.