Revista ELECTRO, Vol. 46, 2024, pp. 154 - 160
https://itchihuahua.mx/revista_electro
ISSN Electrónico: 3061-774X

BIG O COMO MARCO DE REFERENCIA PARA LA MEJORA DE LA EFICIENCIA DE UN SOFTWARE DETECTOR DE RÁFAGAS EXTREMAS DE VIENTO

Guillén Olivera Ángel de Jesús[1], Parra Velasco Laura Yazmín[1*], Dueñas Reyes Efraín[1*], Juárez Vázquez Sergio[1], Hernández Mayoral Emmanuel[2] y Pacheco Bautista Daniel[1]
[1]Universidad del Istmo – Cd. Universitaria s/n, 70760, Tehuantepec, Oax., México. *ing_lypv@live.com.mx, *edr@ier.com.mx. [2]Instituto de Energías Renovables – Xochicalco s/n, Azteca, 62588, Temixco, Mor., México.

RESUMEN.

Actualmente, el uso de sistemas computacionales ha incrementado significativamente la productividad de la humanidad. Sin embargo, esto ha traído consigo problemas, como el aumento de datos de entrada y la complejidad de los modelos diseñados para analizarlos, lo que exige una mayor capacidad computacional. Por ello, se buscan algoritmos cada vez más eficientes. Un ejemplo es el método detector de ráfagas extremas de viento (MDREV), creado para analizar una base de datos de 31 millones de registros de velocidades del viento obtenidos en La Ventosa, Oaxaca, México. Sin embargo, el análisis resultó inviable debido a la complejidad del algoritmo, lo que llevó al desarrollo de un nuevo algoritmo más eficiente para realizar dicho análisis. En este estudio, se analizan ambos algoritmos utilizando la notación Big O como marco de referencia, demostrando que la complejidad del nuevo algoritmo es menor y, por ende, más eficiente. Este resultado se pone a prueba con un estudio empírico de tiempos de ejecución de ambos algoritmos con bases de datos de distintos tamaños, mostrando una mejora del 95% en los tiempos de ejecución.

Palabras Clave: Análisis de algoritmos, ráfagas extremas de viento, notación Big O, eficiencia de algoritmos.

ABSTRACT.

Currently, using computer systems has significantly increased the productivity of humanity. However, this has brought problems, such as the increase in input data and the complexity of the models designed to analyze them, which requires greater computational capacity. Therefore, increasingly efficient algorithms are sought. An example is the Extreme Wind Gust Detector Method (MDREV), created to analyze a database of 31 million wind speed records obtained in La Ventosa, Oaxaca, Mexico. However, the analysis turned out to be infeasible due to the complexity of the algorithm, which led to the development of a new, more efficient algorithm to perform said analysis. In this study, both algorithms are analyzed using the Big O notation as a framework, demonstrating that the complexity of the new algorithm is lower and, therefore, more efficient. This result is tested with an empirical study of the execution times of both algorithms with databases of different sizes, showing a 95% improvement in execution times.

Keywords: Algorithm analysis, extreme wind gusts, Big O notation, algorithm efficiency.

1. INTRODUCCIÓN

Actualmente, el uso de herramientas computacionales para el análisis de datos es crucial para desarrollar nuevas tecnologías. La resolución de problemas actuales demanda un tratamiento eficiente de grandes volúmenes de datos, como ocurre en la detección de ráfagas de viento, donde se registra la velocidad a cada segundo y se analiza anualmente.

El Istmo de Tehuantepec, en Oaxaca, es clave para la energía eólica [1]. La Ventosa, Oaxaca, destaca por su calidad de viento y alta frecuencia de ráfagas, lo que puede afectar los aerogeneradores [2]. Para abordar esto, se creó el algoritmo MDREV, diseñado para detectar ráfagas extremas de viento, pero no es eficiente para analizar un año completo de datos.

En este contexto, el análisis de algoritmos es fundamental en la investigación científica. Por ejemplo, en [3], se analizaron veinticinco algoritmos para diseñar vigas de hormigón al menor costo. En [4], se compararon dos algoritmos con diferentes volúmenes de datos, mostrando que "Quadratic Search" es más rápido con menos de 100,000 datos, mientras "Binary Search" es superior para volúmenes mayores. En [5], se compararon los algoritmos de Dijkstra y Bellman-Ford utilizando la notación Big O como referencia de la complejidad, concluyendo que Dijkstra es superior en términos de tiempo de ejecución, En [6], se evaluaron tres algoritmos para determinar la ruta más corta en el juego “Maze Runner”, destacando también Dijkstra. En [7], se compararon algoritmos para clasificar series temporales multivariadas, evaluando los tiempos de clasificación. En [8], se analizaron varios algoritmos de extracción de características para reducir la dimensionalidad de datos grandes, considerando tiempos de ejecución.

La motivación de esta investigación es desarrollar un algoritmo más eficiente y rápido para detectar ráfagas de viento, lo que permitirá evaluar la viabilidad de granjas eólicas y medidas de protección para aerogeneradores. La contribución reside en el uso de la notación Big O para mejorar el algoritmo, y el desarrollo de un código que maneje grandes volúmenes de datos con bajo costo computacional.

El artículo se organiza así: la sección II presenta el marco teórico; la sección III, la metodología; la sección IV, el análisis de algoritmos; la sección V, los resultados; y la sección VI, las conclusiones.

2. MARCO TEÓRICO

2.1. Análisis de algoritmos.

Un algoritmo es una secuencia de pasos computacionales que transforman un dato de entrada en un valor de salida [9]. El análisis de algoritmos tiene como objetivo determinar la cantidad de “tiempo” que un algoritmo tarda en ejecutarse y, por tiempo se refiere a la cantidad de operaciones que se realizan, esta cantidad de operaciones se ve refleja en el tiempo de ejecución y en el uso de los recursos computacionales [10].

El orden de crecimiento del tiempo de ejecución de un algoritmo proporciona una manera simple de caracterizar la eficiencia del algoritmo y de igual manera permite que sea comparado con algoritmos alternativos. Y para ello existen tres notaciones asintóticas (Big Omega, Big O y Big Theta).

La notación Big O se utiliza para denotar un límite superior asintótico de una función, dentro de un factor constante multiplicativo, y está definida para una función dada $g(n)$, definida en el conjunto $O(g(n))$ del conjunto de funciones

$$O(g(n)) = \{f(n): existen\ constantes\ positivas\ c\ y\ n_0 \\ tal\ \ que\ 0 \leq f(n) \leq cg(n) para\ todo\ n > n_0\}$$

En otras palabras, la función $f(n)$ es $O(g(n))$ si existe una constante c y un valor n_0 a partir del cual $f(n)$ no crece más rápido que $g(n)$ multiplicado por c. Esto se representa matemáticamente como:

$\forall n \geq n_0, \quad f(n) \leq c * g(n)$

Este criterio establece que, para valores de n suficientemente grandes, $f(n)$ está acotada superiormente por $g(n)$ multiplicada por una constante c.

2.2. Ráfagas de viento extremas.

En [11], Branlard hace un trabajo exhaustivo por describir toda la información relacionada con las ráfagas extremas de viento y métodos de detección.

Denomina a una ráfaga de viento como una variación de la velocidad del viento en un corto plazo de tiempo dentro de un campo de viento turbulento. Habiendo dicho lo anterior, se puede establecer que un caso extremo de ráfaga, donde la velocidad del viento sube y baja repentinamente para alcanzar nuevamente la velocidad media del viento. A partir de esta definición describe cuatro tipos de ráfagas de viento: Ráfaga extrema operacional (EOG, por sus siglas en inglés), Cambio de dirección extremo (EDC, por sus siglas en inglés), Ráfaga extrema coherente (ECG, por sus siglas en inglés), Ráfaga extrema coherente con cambio de dirección (ECD, por sus siglas en inglés).

Y para realizar la detección de las ráfagas mencionadas Branlard recopila cinco métodos de detección, los cuales son: el procedimiento pico-pico, el método del incremento de velocidad, el pico sobre el umbral, el método de correlación y el incremento de velocidad por encima del umbral.

l método de detección de ráfagas MDREV se basa en el enfoque del "incremento de velocidad por encima del umbral", detectando ráfagas cuando la aceleración supera un valor umbral y finaliza cuando cae por debajo. A diferencia del método tradicional, MDREV introduce una restricción adicional en la amplitud de la ventana, además de la duración, lo que permite definir un umbral de aceleración más preciso. Esto significa que, en lugar de usar un intervalo fijo de 10 minutos y un umbral de 3 m/s, MDREV permite análisis en intervalos de segundos y umbrales mayores. Estas modificaciones no solo mejoran la precisión, sino que también permiten detectar ráfagas operacionales, algo que el método original, enfocado en ráfagas coherentes de viento, no lograba. Las modificaciones se ilustran en las ecuaciones (1) y (2).

$$u(t+\tau) - u(t) \quad (1)$$

$$u(t+\tau) - u(t+(2*\tau)+1) \quad (2)$$

donde $u(t)$ es la velocidad del viento y τ es el tiempo de evolución de la pendiente positiva de la ráfaga extrema.

3. METODOLOGÍA

La metodología para comparar dos algoritmos comienza con un análisis teórico, en el cual se estudian y comprenden los fundamentos teóricos de los algoritmos. En este análisis inicial se examinan las características, ventajas y desventajas de cada algoritmo desde una perspectiva teórica.

Luego, se procede a obtener las ecuaciones que describen la complejidad temporal y espacial de ambos algoritmos, utilizando notación Big O. Estas ecuaciones se derivan para evaluar el rendimiento teórico de los algoritmos.

Una vez obtenidas las fórmulas de complejidad, se realiza un análisis comparativo entre ambos algoritmos, utilizando estas ecuaciones para discutir cómo se comportan los algoritmos a medida que aumenta el tamaño de los datos de entrada. En esta etapa también se crea un gráfico comparativo que muestra visualmente la diferencia en la cantidad de operaciones que realiza cada algoritmo en función del tamaño de los datos de entrada.

El siguiente paso es el análisis empírico, que implica una evaluación práctica de los algoritmos mediante la implementación y ejecución de programas que los representen. Se realizan pruebas de rendimiento de cada uno de los programas, midiendo el tiempo de ejecución. Estos datos de tiempo de ejecución se recopilan para el análisis comparativo empírico.

Con las medidas de tiempo obtenidas, se lleva a cabo un análisis comparativo entre ambos algoritmos, utilizando los tiempos de ejecución como referencia. Se comparan los tiempos para determinar cuál algoritmo es más eficiente en la práctica.

Finalmente, se concluye el análisis comparativo, resumiendo los hallazgos tanto teóricos como empíricos y discutiendo las implicaciones de los resultados.

4. ANÁLISIS DE ALGORITMOS

4.1. Método de detección MDREV.

El método detector de ráfagas extremas de viento (MDREV), con numero de registro 3-2023-062610002000-01 ante el Registro Público del Derechos de Autor, está diseñado para la detección de EOGs y ECGs mostrando su taxonomía en la Figura 1 [11]. Este detector está basado en el método incremento de velocidad sobre cierto umbral, lo que permite detectar ráfagas de cualquier duración y amplitud. El programa procesa datos a una frecuencia de 1 Hz, lo que significa que maneja una gran cantidad de datos en intervalos de tiempo muy cortos. Dado que la base de datos que se desea analizar contiene más de 31 millones de registros, es esencial que estos datos se gestionen de manera eficiente. Los datos deben almacenarse en un archivo de tipo ".txt", organizado en formato de columnas. Específicamente, los datos de la velocidad del viento se deben de encontrar en la columna 5 del archivo.

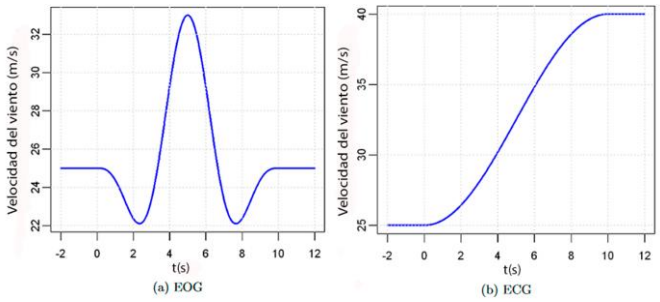

Figura 1. Ejemplos de ráfagas de viento EOG y ECG.

$$f_1 = O(6n) \quad (5)$$
$$f_2 = 24 * O(m) \quad (6)$$
$$f_3 = 12 * O(mn) \quad (7)$$
$$f(n) = f_1 + f_2 + f_3 \quad (8)$$
$$f(n) = O(6n) + 24 * O(m) + 12 * O(mn) \quad (9)$$
$$f(n) = O(6n) + 24 * O(50) + 12 * O(50n) \quad (10)$$
$$f(n) = O(606n + 1200) \quad (11)$$

4.2. Análisis teórico.

Debido a la complejidad del algoritmo MDREV, se decidió analizarlo en tres operaciones fundamentales: primero, un ciclo de análisis de duraciones de ráfagas; segundo, el procesamiento de casos para guardar datos en matrices correspondientes; y tercero, el análisis de estas matrices para graficar los puntos. Además, se realizaron diagramas de flujo explicativos, con un enfoque didáctico en lugar de computacional.

En la Figura 2 se muestra la primera operación del algoritmo MDREV. Esta operación comienza con un ciclo que busca determinar la amplitud ($dif\ y\ difr$) de las ráfagas de viento, cuyas duraciones van de 4 a 9 segundos. A continuación, se ejecuta otro ciclo que analiza todos los puntos i en la base de datos usando las ecuaciones (3) y (4).

$$dif = u(i+h) - u(i) \quad (3)$$
$$difr = u(i+h) - u(i+2*h+1) \quad (4)$$

Luego, se realizan seis operaciones condicionales que evalúan el valor de h en cada ciclo. Durante esta evaluación, se verifica si el punto cumple con las condiciones especificadas por las ecuaciones según los umbrales establecidos. Si se cumplen las condiciones, el punto se almacena en la matriz correspondiente según el valor de h y el tipo de ráfaga detectada.

Basándose en esta secuencia de operaciones, se llevó a cabo un análisis algorítmico para determinar la complejidad de la operación, dando como resultado la ecuación (5).

En la Figura 3 se muestra la segunda operación, que consiste en un conjunto de ciclos continuos que buscan crear la forma de las ráfagas tanto coherente como operacional. Esto se logra generando puntos adicionales a partir de los puntos que se guardaron en la misma matriz, para luego volver a guardarlos en la misma matriz. Debido a esto, el análisis de la operación arrojó que su ecuación (6), donde m es el tamaño de las matrices.

En la Figura 3 se observa la tercera operación, encargada de generar la gráfica a través de otro análisis y escritura de matrices. En esta parte, se debe dejar claro que, hasta ahora, el código está repleto de secciones copiadas, solo cambiando las matrices que se analizan, y esto no es la excepción. Debido a que se trata nuevamente de ciclos anidados, se forman de la siguiente manera: primero, un ciclo que analiza la matriz correspondiente donde m es el tamaño de las matrices, para luego inicializar otro ciclo que es de tamaño n con la intención de anexar los puntos de las ráfagas detectadas en esta nueva matriz y luego así graficarla. Esto se repite doce veces como se muestra en el ciclo de j, una vez por cada matriz correspondiente a una h y tipo de ráfaga, obteniendo la ecuación (7).

Teniendo las tres ecuaciones se procede a encontrar la ecuación de complejidad de todo el algoritmo teniendo en cuenta que m representa el tamaño de las matrices de guardado y, para este caso $m = 50$. Por lo tanto, la fórmula de complejidad del algoritmo antiguo es la ecuación (11).

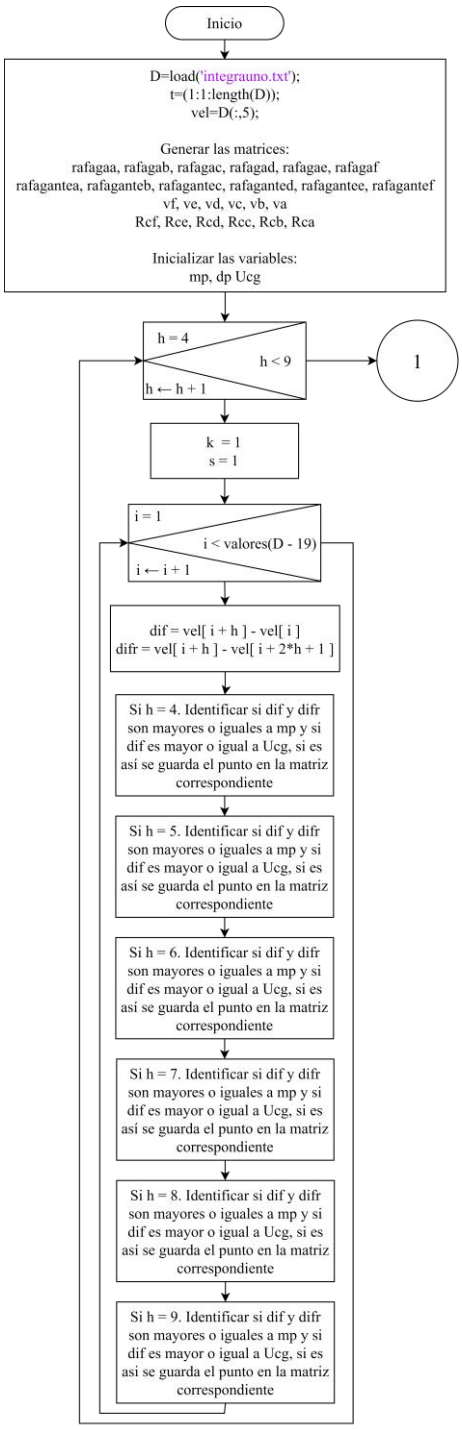

Figura 2. Primera parte del diagrama de flujo del programa MDREV antiguo.

En el caso del segundo algoritmo, como se puede observar en la Figura 5, los procesos que se realizan en el primer algoritmo de forma separada, en este algoritmo se realizan en conjunto. También se debe tener en cuenta que en este nuevo algoritmo solo se utilizan dos matrices, que son las encargadas de almacenar los puntos que posteriormente se graficarán. Lo que

anteriormente requería doce arreglos se puede solucionar con dos, y por ello el análisis da como resultado la ecuación (12).

$$g(n) = O(hn) \quad (12)$$

donde $h = 6$ debido a que es la cantidad de intervalos de tiempo a analizar, lo que da la ecuación (13).

$$g(n) = O(6n) \quad (13)$$

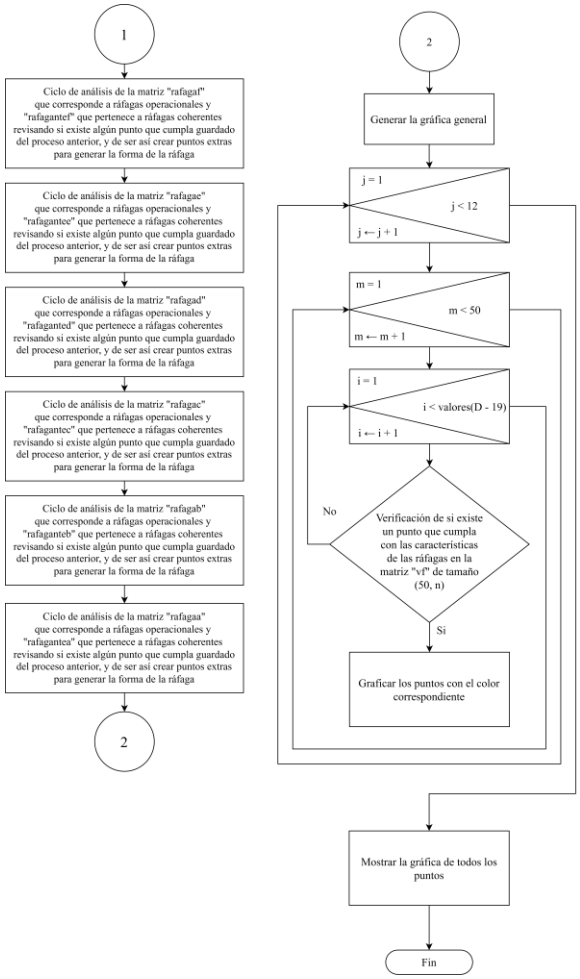

Figura 3. Segunda y Tercera parte del diagrama de flujo del programa MDREV antiguo.

En la Figura 4 se puede observar una comparativa de los pasos que deben de realizar ambos algoritmos dependiendo de la cantidad de datos que le son suministrados, ya que, aunque ambos algoritmos tengan un orden de complejidad n el número de operaciones que realizan para cumplir con su objetivo varia significativamente debido a todos los ciclos extra que contiene el antiguo algoritmo. Dando como resultado, que el segundo algoritmo demuestra ser 100 veces más eficiente, dada la magnitud de las ecuaciones.

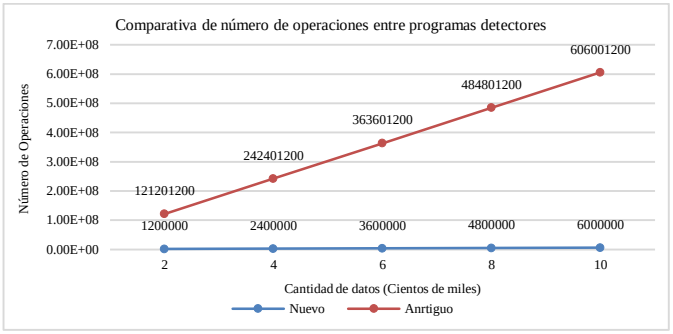

Figura 4. Gráfica comparativa sobre los pasos que realizan cada uno de los algoritmos.

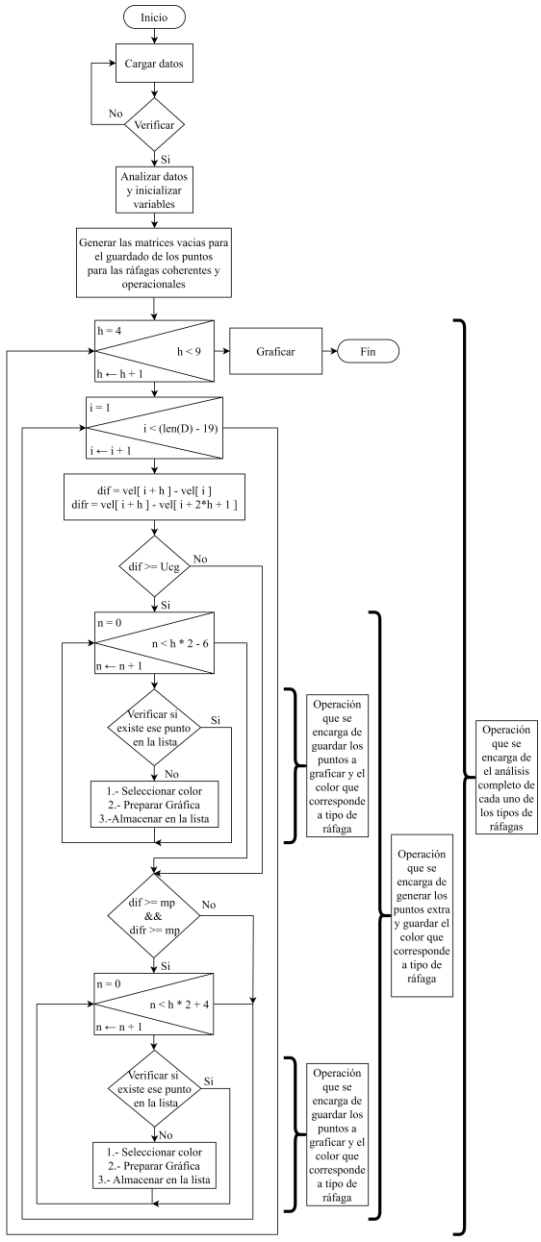

Figura 5. Diagrama de flujo del programa MDREV nuevo.

Es importante tomar en cuenta la cantidad de matrices que se requieren en el algoritmo antiguo y cómo se usaban dichas matrices, ya que el propio análisis se encontraba limitado debido al tamaño de las matrices. El método detector maneja un criterio de búsqueda de amplitudes de las ráfagas (umbral). Cuanto menor sea el umbral, más ráfagas serán detectadas y, por lo tanto, más puntos se analizarán y graficarán, y estos puntos se tienen que guardar en dichas matrices. Esto genera que la complejidad no solo se vea afectada por la cantidad de datos, sino también por los umbrales de amplitud.

Sabiendo esto, la minimización del uso de matrices por sí solo hace que el programa sea más eficiente, pero también lo hace escalable, ya que no se tiene que ajustar las dimensiones de las matrices, como se haría con el algoritmo anterior. De igual manera, la escalabilidad se ve mejorada debido a que, si se desea aumentar o disminuir los umbrales a analizar, se puede hacer de forma sencilla en el nuevo programa.

Por ello, en el nuevo algoritmo, la operación de procesamiento de casos para guardar datos en matrices correspondientes y el análisis de las matrices para el graficado de puntos se pueden hacer en una sola operación, como se puede observar en la Figura 5. Esto hace que la complejidad del algoritmo se vea disminuida debido a que se eliminan actividades repetitivas para cada uno de los tipos de ráfagas.

4.3. Análisis Empírico.

En el análisis empírico se utilizó el lenguaje de programación Python 3.12.3, se implementaron ambos algoritmos y se realizaron pruebas de rendimiento para comparar sus tiempos de ejecución con diferentes tamaños de bases de datos y también se llevaron a cabo pruebas de tiempo a diferentes umbrales de amplitud de ráfaga de viento a detectar, para observar cómo dichos umbrales afectan al rendimiento de ambos programas. Se llevaron a cabo cinco pruebas con cuatro bases de datos de distintos tamaños, dichas pruebas se llevaron a cabo una computadora con los siguientes componentes: un procesador Intel® Core™ i5-11400F, 16GB de memoria RAM DDR4 con una velocidad de 2400 MHz, una tarjeta madre MPG Z590 GAMING PLUS, un disco sólido SSD M.2 de un 1TB con una velocidad de escritura de 2100MB/s y una tarjeta gráfica GeForce RTX™ 3060 GAMING OC 12G con una arquitectura de 64 bits. Los resultados obtenidos proporcionaron una visión clara del rendimiento práctico de ambos algoritmos.

Los resultados del análisis empírico como se muestran en la Tabla 1, indican que el algoritmo nuevo es significativamente más eficiente que el algoritmo antiguo en términos de tiempo de ejecución. El algoritmo nuevo no solo completó todas las pruebas, sino que lo hizo en un tiempo considerablemente menor. Además, el algoritmo antiguo no pudo manejar las bases de datos más grandes debido a limitaciones de memoria, mientras que el algoritmo nuevo no presentó tales problemas. Estos resultados resaltan la superioridad del algoritmo nuevo en situaciones prácticas. Esto se puede observar de forma visual en la Figura 6.

Tabla 1. Resultados de tiempos medios de las pruebas de realizadas

Número de datos	MDREV nuevo (Tiempo promedio)	MDREV antiguo (Tiempo promedio)
86,400	0.38 s	5.95 s
6,291,456	22.21 s	454.31 s
12,582,912	45.08 s	No se puedo completar
18,874,368	68.78 s	No se puedo completar

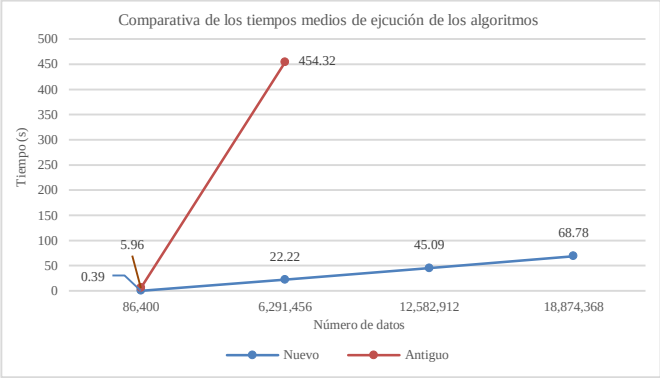

Figura 6. Gráfica comparativa sobre los tiempos medios de ejecución de ambos algoritmos.

En cuanto al uso de la memoria, se puede evaluar mediante una estimación del consumo de memoria asociado con las matrices. Primero, consideremos el algoritmo antiguo, que utiliza 12 matrices estáticas y 15 dinámicas. De estas matrices, solo una es de tipo flotante, mientras que las restantes son de tipo entero. El análisis se resume en la ecuación (14), donde n es el número de datos, "$float$" es el tamaño en bytes asignado por Python para los datos de tipo flotante, e "int" es el tamaño en bytes para los datos de tipo entero, que son 8 y 4 bytes respectivamente.

$$(9 * n) * (float) + (602 * n + 600) * (int) \quad (14)$$

En el nuevo algoritmo, se utilizan 2 matrices estáticas y 5 dinámicas, de las cuales 3 son de tipo flotante y 2 son de tipo entero. Esto da lugar a la ecuación (15), donde p es el número de puntos guardados en las matrices vacías. Dado que siempre se detectan los mismos puntos para todas las bases de datos, p es un valor constante de 60.

$$(9 * n + 2 * p) * (float) + (2 * n + 24) * (int) \quad (15)$$

Como se detalla en la Tabla 2 y la Figura 7, el algoritmo antiguo consume 15.250 GB para un número de datos de 6,291,456, mientras que el nuevo algoritmo utiliza solo 0.503 GB para la misma cantidad de datos. Esta diferencia en el consumo de memoria explica por qué, al intentar procesar 12,582,912 datos con el algoritmo antiguo, el sistema no tenía suficiente memoria para realizar el análisis ya que la estimación marca una de manda de más de 30 GB.

Tabla 2. Memoria estimada de cada algoritmo en función de diferentes bases de datos

Número de datos	MDREV nuevo (Memoria estimada)	MDREV antiguo (Memoria estimada)
86,400	0.007 GB	0.209 GB
6,291,456	0.503 GB	15.250 GB
12,582,912	1.007 GB	30.501 GB
18,874,368	1.510 GB	45.751 GB

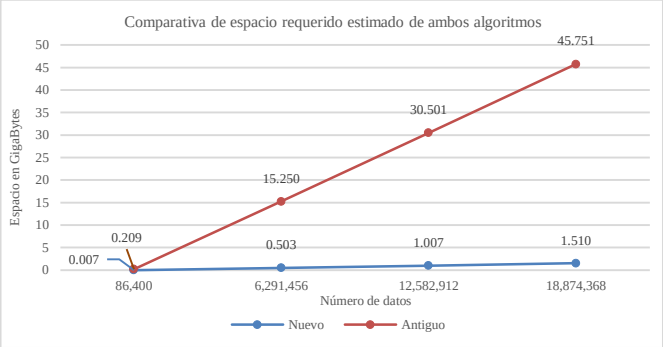

Figura 7. Gráfica comparativa sobre la memoria estimada de ambos algoritmos.

5. DISCUSIÓN DE RESULTADOS

En este estudio, se determinaron las ecuaciones de complejidad de los dos algoritmos involucrados. El algoritmo antiguo se caracteriza por realizar tres operaciones fundamentales: un ciclo de análisis de duraciones de ráfagas, el procesamiento de casos para guardar datos en matrices correspondientes, y el análisis de estas matrices para el graficado de puntos. La complejidad total del algoritmo antiguo se expresa como $f(n) = O(606n + 1200)$. En contraste, el segundo algoritmo realiza todas las operaciones en una única fase de análisis, lo que se refleja en una complejidad de $g(n) = O(6n)$. Al comparar ambos algoritmos, se puede concluir que el nuevo algoritmo es más eficiente en términos de pasos de ejecución, como se muestra en la Figura 4. La superioridad del segundo algoritmo sobre el antiguo radica en su enfoque más eficiente y simplificado para realizar las operaciones requeridas. El uso de una única operación frente a múltiples pasos del algoritmo antiguo, reduce significativamente la complejidad computacional y mejora el rendimiento general del algoritmo en términos de velocidad de ejecución y manejo de recursos. En el análisis empírico, se implementaron ambos algoritmos y se realizaron pruebas de rendimiento para comparar sus tiempos de ejecución en diferentes cantidades de datos. Se llevaron a cabo cinco pruebas con cuatro bases de datos de distintos tamaños. Los resultados obtenidos proporcionaron una visión clara del rendimiento práctico de ambos algoritmos como se puede observar en la Tabla 1 y en la Figura 6. Los resultados del análisis empírico indican que el algoritmo nuevo es significativamente más eficiente que el algoritmo antiguo en términos de tiempo de ejecución. El algoritmo nuevo no solo completó todas las pruebas, sino que lo hizo en un tiempo considerablemente menor. Por ejemplo, para la base de datos con 6,291,456 elementos, el nuevo algoritmo tardó 22.21 segundos, mientras que el algoritmo antiguo tardó 454.31 segundos, mostrando una mejora de aproximadamente el 95%. Además, el manejo ineficiente de los arreglos en el algoritmo antiguo generó problemas de ejecución como se muestra en la Figura 7, mientras que el algoritmo nuevo no presentó tales limitaciones. Estos hallazgos empíricos resaltan la superioridad del algoritmo nuevo en situaciones prácticas, especialmente cuando se trabaja con grandes volúmenes de datos. Este análisis práctico complementa el análisis teórico, proporcionando una visión completa y detallada del rendimiento de ambos algoritmos.

6. CONCLUSIONES

Este estudio comparativo entre dos algoritmos para el análisis de ráfagas de viento revela que la optimización del código puede resultar en mejoras significativas en eficiencia. El segundo algoritmo, con una complejidad de $g(n) = O(6n)$, supera ampliamente al primer algoritmo, cuya complejidad es $f(n) = O(606n + 1200)$. El uso de la notación Big O en algoritmos es una herramienta de gran utilidad para la comparación y mejora de algoritmos, con la cual se pueden obtener comparativas validas y ser la base para la mejora continua de algoritmos para cualquier aplicación. La implementación de algoritmos más eficientes es crucial para el análisis efectivo de grandes volúmenes de datos, permitiendo una toma de decisiones más rápida y precisa en aplicaciones críticas como la evaluación de sitios para la instalación de granjas eólicas. Este trabajo subraya la importancia de la optimización algorítmica en el desarrollo de soluciones tecnológicas avanzadas, promoviendo un enfoque más eficiente y rentable en el procesamiento de datos complejos. Igualmente se resalta la importancia de comprender como las variables pueden incrementar la complejidad de los algoritmos, en este estudio se demostró que la reducción de los umbrales de velocidad de las ráfagas, genera una mayor carga en el análisis debido a que entre menor sea el umbral, más ráfagas podrán ser detectadas eso en el primer algoritmo significa más puntos que guardar en las matrices y debido a que las matrices tienen dimensiones predeterminadas (estáticas) si los puntos sobrepasan estas dimensiones, las matrices no tendrán donde alojarlos y generara un error, pero no solo eso, sino que también se verá afectado el tiempo de análisis. La adopción del segundo algoritmo no solo mejora el rendimiento y la velocidad del análisis, sino que también facilita una gestión más eficiente de los recursos computacionales, impulsando así el desarrollo de tecnologías avanzadas en el sector eólico.

7. REFERENCIAS

[1] I. K. Ángeles-Pérez, G. Martínez-Reyes, R. Iracheta-Cortez, L. A. Otamendi-Cruz, J. d. L. Cruz-Soto and E. Dueñas-Reyes, "Comparison of wind resource among physical and virtual weather stations for analyzing the economic feasibility of wind farms," 2022 IEEE 40th Central America and Panama Convention (CONCAPAN), Panama, Panama, 2022, pp. 1-6, doi: 10.1109/CONCAPAN48024.2022.9997606.

[2] D. Elliot, M. Schwartz, G. Scott, S. Haymes, D. Heimiller, R. George, Atlas de recursos eólicos del estado de oaxaca, Tech. Rep. NREL/TP-500-35575, Laboratorio Nacional de Energía Renovable (NREL), USA654 (2004).

[3] Öztürk, H. T. (2024). Research on optimal solutions and algorithm stability analyses in RC continuous beam problems. *Structures*, *62*, 106239. https://doi.org/10.1016/j.istruc.2024.106239

[4] Hendy, K., & Istiono, W. (2020). Efficiency Analysis of Binary Search and Quadratic Search in Big and Small Data. *Computational Science and Techniques*, 7. https://doi.org/10.15181/csat.v7i1.2091

[5] AbuSalim, S. W. G., Ibrahim, R., Zainuri Saringat, M., Jamel, S., & Abdul Wahab, J. (2020). Comparative Analysis between Dijkstra and Bellman-Ford Algorithms in Shortest Path Optimization. *IOP Conference Series: Materials Science and Engineering*, *917*, 012077. https://doi.org/10.1088/1757-899x/917/1/012077

[6] Handy Permana, S. D., Yogha Bintoro, K. B., Arifitama, B., & Syahputra, A. (2018). Comparative Analysis of Pathfinding Algorithms A *, Dijkstra, and BFS on Maze Runner Game. *IJISTECH (International Journal Of Information System & Technology)*, *1*(2), 1. https://doi.org/10.30645/ijistech.v1i2.7

[7] Ruiz, A. P., Flynn, M., Large, J., Middlehurst, M., & Bagnall, A. (2020). The great multivariante time series classification bake off: a review and experimental evaluation of recent algorithmic advances. *Data Mining and Knowledge Discovery*. https://doi.org/10.1007/s10618-020-00727-3

[8] Anowar, F., Sadaoui, S., & Selim, B. (2021). Conceptual and empirical comparison of dimensionality reduction algorithms (PCA, KPCA, LDA, MDS, SVD, LLE, ISOMAP, LE, ICA, t-SNE). *Computer Science Review*, *40*, 100378. https://doi.org/10.1016/j.cosrev.2021.100378

[9] Cormen, T. H., Stein, C., Rivest, R. L., & Leiserson, C. E. (2009). *Introduction to Algorithms*. MIT Press.

[10] McConnell, J. J. (2008). *Analysis of algorithms: An active learning approach* (2a ed.). Jones and Bartlett Publishers.

[11] E. Branlard, Wind energy: On the statistics of gusts and their propagation through a wind farm, Tech.786 Rep. ECN-Wind-Memo-09-005, Energy Research Centre of the Netherlands, Petten, the Netherlands787 (2009).