Diseño de experimentos con tres factores

Lucas García García
Métodos estadístico
48118360Z
lucas.garcia.garcia@udc.es
3/12/2025

Índice

Introducción

En este experimento se evalúa el tiempo de búsqueda de un array utilizando tres algoritmos (Búsqueda Lineal, Búsqueda Binaria y Búsqueda de Salto). Los arrays serán de distintos tipos de datos (numéricos, alfabéticos y mixtos), y de tamaños variados (10.000, 100.000 y 1.000.000 elementos). He implementado un código Java que mide el rendimiento de la búsqueda bajo estas condiciones, para comparar la eficiencia y efectividad de los diferentes escenarios.
La motivación de este trabajo es contrastar la eficiencia teórica de estos algoritmos (conocidas por sus complejidades O(n), O($\sqrt{n}$) y O(log n)) con su rendimiento en un escenario práctico. Es fundamental conocer su desempeño real para poder seleccionar el método más adecuado en un contexto de uso específico, especialmente en aplicaciones que manejan grandes volúmenes de datos donde la eficiencia es un factor muy importante.

En este estudio, la población está compuesta por todos los posibles arrays de datos en los que se puede realizar una búsqueda, considerando sus diversas combinaciones de tamaño, tipo de datos y algoritmo de búsqueda. Cada individuo de la población es un experimento específico, definido por una combinación única de estos factores. En total, se generan 378 individuos, resultado de combinar 27 configuraciones diferentes de factores (3 algoritmos x 3 tamaños x 3 tipos) con 14 réplicas para cada combinación, proporcionando 378 observaciones para el análisis.

Variable respuesta

La variable respuesta en este experimento es el tiempo de ejecución de la búsqueda. Es importante destacar que no se mide el tiempo en “encontrar” un elemento, sino el tiempo total que tarda el algoritmo en ejecutarse.
En el diseño experimental, se busca un elemento que se sabe que no existe (simulando el “peor caso”). Por lo tanto, la variable mide el tiempo que tarda la función en procesar el array y determinar que el elemento no está presente.

- Unidad de medida: El tiempo se registra en nanosegundos (ns). Las búsquedas al ser tan rápidas, los milisegundos no capturarían las diferencias con suficiente detalle.
- Método de captura: La medición del tiempo se realiza a través de la función System.nanoTime() de Java. Esta función proporciona una alta precisión en la captura del tiempo, lo cual es fundamental para este experimento.

Factores del experimento

Algoritmos de Búsqueda

Se seleccionan tres algoritmos de búsqueda con complejidades teóricas fundamentalmente distintas:

- Búsqueda Lineal:Es el método más básico. Funciona recorriendo el arrays elemento por elemento, desde el inicio hasta el final, comparando cada uno con el valor buscado. Su complejidad temporal en el peor de los casos es O(n), lo que lo hace ineficiente para grandes volúmenes de datos, pero no requiere que el array esté ordenado.
- Búsqueda Binaria: Este algoritmo implementa una estrategia de "divide y vencerás". Requiere que el array esté previamente ordenado. Compara el elemento central del array con el valor buscado; si no es igual, divide el array por la mitad y repite el proceso en la sub-mitad correspondiente. Su complejidad temporal es O(log n), lo que lo convierte en un algoritmo extremadamente eficiente para arrays de gran tamaño.
- Búsqueda de Salto (Jump Search): También requiere un array ordenado y es un algoritmo intermedio entre el lineal y el binario. Funciona comprobando elementos en "saltos" de un tamaño fijo (basado en la raíz cuadrada del tamaño del arrays $\sqrt{n}$. Una vez que encuentra un bloque donde el elemento podría estar, realiza una búsqueda lineal solo dentro de ese bloque. Su complejidad temporal es $O(\sqrt{n})$, siendo más rápido que la búsqueda lineal pero más lento que la binaria.

Tipo de datos

El segundo factor es el tipo de datos que contienen los arrays. Este factor es relevante, ya que puede influir en el tiempo necesario para realizar las comparaciones. Para este experimento, todos los datos se han tratado como cadenas de texto (String) para permitir la ordenación y comparación en arrays mixtos.

- Array Numérico: Compuesto por cadenas de texto que representan números enteros (p.ej., "27", "8", "47").
- Array Alfabético: Compuesto por cadenas de texto de un solo carácter (letras mayúsculas de la A a la Z).
- Array Mixto: Compuesto por una combinación de cadenas de texto numéricas y alfabéticas.

Tamaño del array

El tercer factor es el tamaño total del array. Para este experimento, se han elegido tres niveles que representan diferentes órdenes de magnitud.

- 10.000 elementos
- 100.000 elementos
- 1.000.000 elementos

El objetivo de elegir estos tamaños es observar cómo escala el tiempo de ejecución de cada algoritmo. A diferencia de una progresión lineal (como 1000, 2000, 3000), el uso de órdenes de magnitud (aumentando 10x y 100x el tamaño) permite visualizar de forma mucho más clara el impacto de las complejidades teóricas O(n), O($\sqrt{n}$) y O(log n).

Metodología

Para este experimento, se evaluaron las diferentes combinaciones de los 3 factores (Algoritmo de Búsqueda, Tipo de Datos y Tamaño del Array), realizando 14 réplicas para cada combinación específica de ellos.

El propósito de realizar múltiples réplicas es aumentar la fiabilidad y la precisión de los resultados. En experimentos que miden el tiempo de ejecución en nanosegundos, pueden existir variaciones debido a diversos factores, como el estado del sistema, fluctuaciones en la carga de trabajo del procesador o pequeñas diferencias en el entorno de ejecución. Al realizar 14 réplicas, se busca suavizar los efectos de estos factores aleatorios, obteniendo así una media representativa del comportamiento del algoritmo. Esto permite que los resultados no estén sesgados por un valor atípico o por una ejecución ocasional que pueda haber sido influenciada por factores externos.

A continuación, se detallan los pasos que se siguen en el código Java implementado:

1. Generación y Preparación de Combinaciones

El código utiliza una serie de ciclos for anidados para generar y probar sistemáticamente todas las combinaciones posibles de los tres factores. En el primer ciclo itera sobre los tres algoritmos de búsqueda, en el segundo sobre los tamaños y en el tercero itera sobre los tipos de datos del array.
Como hay 3 niveles por cada uno de los factores, el número total de combinaciones únicas es de 27. Por cada una de estas 27 combinaciones, el código ejecuta 14 réplicas, lo que da un total de 378 ejecuciones en total.
Un paso metodológico clave ocurre antes de la medición: el código comprueba si el algoritmo a probar es "Binaria" o "Salto". Si es así, se ordena el array previamente usando Arrays.sort().

Esta ordenación es un requisito de preparación para que dichos algoritmos funcionen y no se incluye en la medición del tiempo de búsqueda.

2. Medición del tiempo de la Ejecución

Una vez que se ha generado y (si es necesario) ordenado el array, el siguiente paso es medir el tiempo que tarda el algoritmo en ejecutarse.
Para realizar esta medición de manera precisa, el código utiliza el método System.nanoTime() de Java. Este método ofrece la mayor precisión disponible para la medición de intervalos de tiempo. Se mide el tiempo justo antes de llamar a la función de búsqueda (startTime). Se ejecuta el algoritmo de búsqueda. En este experimento, se le pide buscar un elemento ("---") que se sabe que no existe, para medir de forma consistente el "Peor Caso" del algoritmo.
Se mide el tiempo inmediatamente después de que la función de búsqueda termina (endTime). El tiempo transcurrido se calcula restando el valor de la medición inicial al valor de la medición final (duracion = endTime - startTime). Y dado que los tiempos de búsqueda son extremadamente rápidos, el valor se registra y se mantiene en nanosegundos (ns) para el análisis.

3. Impresión de los resultados

Tras cada una de las 378 ejecuciones, cuando ya se tiene el tiempo total que tardó la búsqueda en nanosegundos, se imprime por terminal una línea con los resultados. Esta línea contiene el algoritmo empleado, el tamaño y tipo del array, el número de réplica y el tiempo que duró la búsqueda, todo en un formato separado por comas para su fácil importación a Statgraphics.

En los array mixtos, los he tratado como ‘String’, con lo que implica que he utilizado un orden lexicográfico estándar (ASCII). Por ejemplo, si tuviésemos el vector original: ["Z", "1", "D", "2", "3"], una vez ordenado quedaría: ["1", "2", "3", "D", "Z"]

Modelo matemático

El modelo matemático para este diseño de 3 factores (Algoritmo, Tamaño, Tipo) es un modelo factorial completo, que incluye los efectos principales y todas sus interacciones:

$$Y_{ijkr} = \mu + \alpha_i + \beta_j + \gamma_k + (\alpha\beta)_{ij} + (\alpha\gamma)_{ik} + (\beta\gamma)_{jk} + (\alpha\beta\gamma)_{ijk} + \epsilon_{ijkr}$$

Y_{ijkr} : El valor observado de la variable dependiente (el tiempo de búsqueda en nanosegundos) para la combinación del nivel i de Algoritmo, el nivel j de Tamaño, el nivel k de Tipo de Dato, en la réplica r.

μ: La media global o promedio del tiempo de búsqueda para todas las combinaciones.

α_i: El efecto principal del Algoritmo de Búsqueda en el nivel i.

β_j : El efecto principal del Tamaño del Array en el nivel j.

γ_k : El efecto principal del Tipo de Dato en el nivel k.

$(\alpha\beta)_{ij}$: El efecto de la interacción entre los factores Algoritmo y Tamaño.

$(\alpha\gamma)_{ik}$: El efecto de la interacción entre los factores Algoritmo y Tipo de Dato.

$(\beta\gamma)_{jk}$: El efecto de la interacción entre los factores Tamaño y Tipo de Dato.

$(\alpha\beta\gamma)_{ijk}$: El efecto de la interacción de los tres factores (Algoritmo, Tamaño y Tipo).

ϵ_{ijkr} : El error aleatorio asociado con cada observación. Se asume que este error sigue una distribución normal con media cero y varianza σ^2, es decir, $\epsilon_{ijkr} \sim N(0, \sigma^2)$. Este término captura las variaciones no explicadas por los factores y sus interacciones.

Índices

Los índices i, j, k, r toman los siguientes valores:

- i = 1, 2, 3. Representa los niveles del Algoritmo de Búsqueda.
  - i = 1 para Búsqueda Lineal.
  - i = 2 para Búsqueda Binaria.
  - i = 3 para Búsqueda de Salto.

- j = 1, 2, 3. Representa los niveles del Tamaño del Array.
  - j = 1 para 10.000 elementos.
  - j = 2 para 100.000 elementos.
  - j = 3 para 1.000.000 elementos.

- k = 1, 2, 3. Representa los niveles del Tipo de Datos.
  - k = 1 para Numéricos.
  - k = 2 para Alfabéticos.
  - k = 3 para Mixtos.

- r = 1, 2, …, 14. Representa el número de réplicas realizadas para cada combinación de factores.

Preparación y Aleatorización de los Datos

Antes de proceder con el análisis ANOVA, es fundamental preparar la hoja de datos para asegurar el cumplimiento del supuesto de independencia.

Los datos importados desde el script de Java se encuentran naturalmente agrupados por factor. Si se analizan en este orden, las pruebas de independencia de los residuos podrían mostrar una autocorrelación falsa, debida al orden de entrada y no al modelo.

Para evitar este sesgo y asegurar que las 378 observaciones son independientes, se aleatorizó el orden de las filas en Statgraphics. Este proceso se realizó de la siguiente manera:

- Creación de una Columna de Orden: Primero, se creó una columna orden utilizando la expresión: COUNT(1;378;1) Esto genera una secuencia de números del 1 al 378, permitiendo recuperar el orden original si fuese necesario.

- Creación de la Columna de Aleatorización: Se añadió una nueva columna llamada aleatorio y se rellenó utilizando la expresión: RUNIFORM (378;0;1) Esta función generó 378 números decimales aleatorios únicos, uno para cada fila.

- Ordenación: Finalmente, se ordenó la hoja de datos completa basándose en los valores de la columna aleatorio (ascendentemente).

Este procedimiento "baraja" eficazmente las 378 observaciones. El resultado es una hoja de datos donde las filas están en un orden completamente aleatorio, garantizando que el análisis ANOVA y el posterior chequeo de residuos sean válidos.

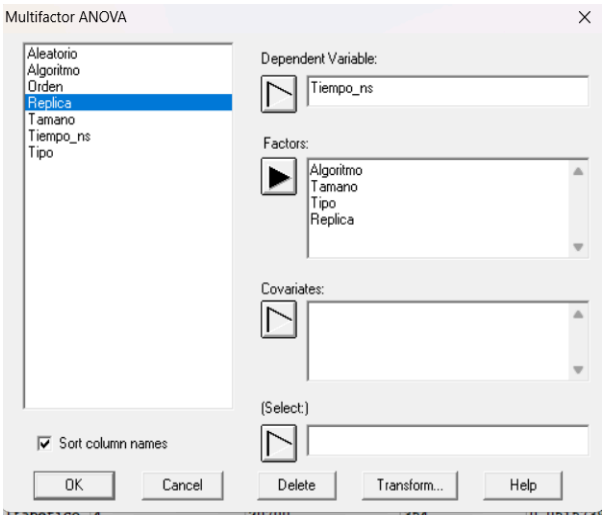
	Algoritmo	Tamano	Tipo	Replica	Tiempo_ns	Orden	Aleatorio
1	Binaria	100000	Numerico	4	7400	172	0,000457024395
2	Lineal	1000000	Mixto	1	5831400	113	0,000532629122
3	Lineal	10000	Mixto	6	110500	34	0,013240408641
4	Salto	100000	Alfabetico	7	16900	315	0,014186615135
5	Salto	100000	Numerico	6	34600	300	0,017834317540
6	Salto	100000	Numerico	11	17700	305	0,017840783295
7	Lineal	100000	Numerico	9	564900	51	0,01998881042
8	Salto	10000	Alfabetico	10	5900	276	0,022855964842
9	Lineal	10000	Numerico	3	399000	3	0,024706410392
10	Binaria	100000	Numerico	10	8900	178	0,025140894943
11	Binaria	10000	Alfabetico	1	1700	141	0,026485235434
12	Salto	100000	Mixto	2	17900	324	0,029504952157
13	Lineal	10000	Alfabetico	10	162000	24	0,032160272370
14	Binaria	10000	Mixto	14	2000	168	0,034975459524
15	Lineal	1000000	Alfabetico	11	4593000	109	0,036561386947
16	Lineal	100000	Alfabetico	2	424800	58	0,037484533411
17	Lineal	1000000	Alfabetico	7	3361500	105	0,040676015185
18	Lineal	10000	Numerico	6	376600	6	0,040689770115
19	Lineal	10000	Numerico	8	171800	8	0,043528757049
20	Lineal	100000	Mixto	5	436200	75	0,048675501266
21	Salto	1000000	Alfabetico	4	20700	354	0,051573911611

Análisis tabla ANOVA

Para el análisis de los datos, se utiliza el Análisis de Varianza (ANOVA) como herramienta estadística principal. El propósito del ANOVA es determinar si las diferencias observadas entre las medias de los diferentes grupos son estadísticamente significativas.

En el caso de este experimento, el objetivo es evaluar el tiempo de búsqueda en función de tres factores controlados: Algoritmo de Búsqueda, Tamaño del Array y Tipo de Dato. El análisis ANOVA nos permitirá cuantificar la influencia de cada uno de estos factores, así como el impacto de sus interacciones, sobre el tiempo de ejecución. De esta forma, podremos concluir qué factores son influyentes y si existen combinaciones específicas que alteren el rendimiento de forma significativa.

Como primer paso metodológico, se realiza una comprobación para asegurar que el factor Replica no tiene influencia en los resultados. Si la réplica tuviera un efecto significativo, indicaría un posible sesgo en el proceso de recogida de datos. Para verificar esto, se ajusta un primer modelo que incluye Réplica como un factor adicional.



El análisis inicial muestra que el p-valor para el factor "Replica" es de 1,0000. Este valor es mucho más alto que el umbral de significancia estándar (p.ej., 0.1), lo que indica que no hay evidencia estadística de que el orden de las réplicas tuviera un efecto en la medición.

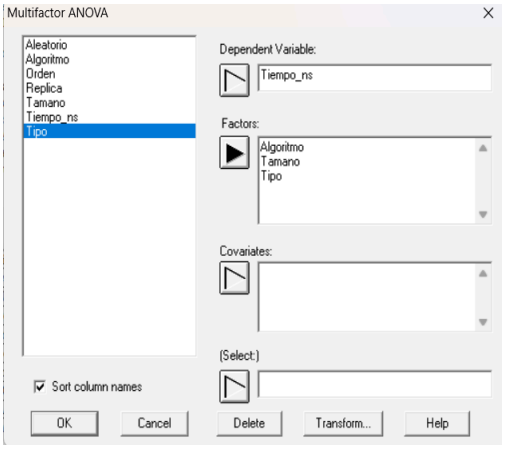
En consecuencia, aceptamos la hipótesis nula. Concluimos que el factor "Replica" no es significativo y, por lo tanto, el proceso de replicación fue robusto y se ejecutó correctamente.

Analysis of Variance for Tiempo_ns - Type III Sums of Squares

Source	*Sum of Squares*	*Df*	*Mean Square*	*F-Ratio*	*P-Value*
MAIN EFFECTS					
A:Algoritmo	2,24814E14	2	1,12407E14	123,47	0,0000
B:Tamano	1,5311E14	2	7,65548E13	84,09	0,0000
C:Tipo	8,0086E11	2	4,0043E11	0,44	0,6445
D:Replica	6,82401E11	13	5,24924E10	0,06	1,0000
RESIDUAL	3,25915E14	358	9,10377E11		
TOTAL (CORRECTED)	7,05322E14	377			

All F-ratios are based on the residual mean square error.

Tras esta validación, se procede a eliminar este factor del modelo para centrar el análisis en los efectos principales (Algoritmo, Tamaño, Tipo) y sus interacciones.



Analysis of Variance for Tiempo_ns - Type III Sums of Squares

Source	*Sum of Squares*	*Df*	*Mean Square*	*F-Ratio*	*P-Value*
MAIN EFFECTS					
A:Algoritmo	2,24814E14	2	1,12407E14	127,69	0,0000
B:Tamano	1,5311E14	2	7,65548E13	86,96	0,0000
C:Tipo	8,0086E11	2	4,0043E11	0,45	0,6349
RESIDUAL	3,26597E14	371	8,80317E11		
TOTAL (CORRECTED)	7,05322E14	377			

All F-ratios are based on the residual mean square error.

En esta primera tabla, analiza el modelo de efectos principales sin interacciones, y observamos varios puntos clave:

- La varianza residual es S_r^2 = 8,80317E11. Este es nuestro punto de referencia para la variabilidad no explicada.
- Los factores Algoritmo (P-Valor = 0,0000) y Tamaño (P-Valor = 0,0000) son altamente significativos.
- El factor Tipo (P-Valor = 0,6349) no es significativo en absoluto.

Ahora, para ver si el modelo mejora al incluir las combinaciones de factores, analizamos el modelo para 2 interacciones (considerando nula la interacción de orden 3).

Al añadir las interacciones de orden 2, obtenemos esta nueva tabla ANOVA, que es mucho mejor que la anterior por dos razones clave.
La Varianza Residual (el error) ha bajado a 7,40228E10. Una reducción tan grande significa que este nuevo modelo es mucho más preciso y explica mejor la variabilidad de los datos.

Analysis of Variance for Tiempo_ns - Type III Sums of Squares

Source	*Sum of Squares*	*Df*	*Mean Square*	*F-Ratio*	*P-Value*
MAIN EFFECTS					
A:Algoritmo	2,24814E14	2	1,12407E14	1518,54	0,0000
B:Tamano	1,5311E14	2	7,65548E13	1034,21	0,0000
C:Tipo	8,0086E11	2	4,0043E11	5,41	0,0048
INTERACTIONS					
AB	2,98045E14	4	7,45111E13	1006,60	0,0000
AC	1,46012E12	4	3,6503E11	4,93	0,0007
BC	5,18596E11	4	1,29649E11	1,75	0,1381
RESIDUAL	2,65742E13	359	7,40228E10		
TOTAL (CORRECTED)	7,05322E14	377			

All F-ratios are based on the residual mean square error.

Y las interacciones:

- AB (Algoritmo*Tamano): Es extremadamente significativa (P-Valor = 0,0000).
- AC (Algoritmo*Tipo): Es muy significativa (P-Valor = 0,0007).
- BC (Tamano*Tipo): No es significativa (P-Valor = 0,1381).

Los P-Valores tan bajos de AB y AC nos obligan a rechazar la hipótesis nula. Concluimos que las interacciones son influyentes y significativas.

Vemos que hay una diferencia notable en la variabilidad residual, se reduce en más de un orden de magnitud, pasando de 8,80E11 (en el modelo simple) a 7,40E10 cuando consideramos las interacciones.
Dado que los p-valores de las interacciones AB (Algoritmo*Tamano) y AC (Algoritmo*Tipo) son altamente significativos (0,0000 y 0,0007) y la variabilidad disminuye de forma tan considerable, concluimos que las interacciones son efectivamente significativas. Esto confirma que el modelo con interacciones es mejor y explica mucho más la variabilidad de los datos.

Más adelante, en el análisis gráfico, comprobaremos cómo los gráficos de interacción demuestran que, efectivamente, sí hay interacción.

Factor Algoritmo

Contraste principal:

- Hipótesis nula (H_0): El factor Algoritmo no influye en la variable dependiente ($\alpha_1 = \alpha_2 = \alpha_3 = 0$)
- Hipótesis alternativa (H_1): El factor Algoritmo influye en la variable dependiente ($\alpha_i \neq 0$)

Test de la F:

La varianza explicada por el factor es de $1,12407\ x\ 10^{14}$ y la varianza no explicada es de $7,40228\ x\ 10^{10}$. Con lo que el F- Ratio = $1,12407\ x\ 10^{14}$ / $7,40228\ x\ 10^{10}$ = 1518,54

El F-Ratio obtenido es F(2,359) = 1518,54. Este valor es extremadamente alto, lo que resulta en un P-Valor de 0,0000. Al ser P-Valor menor que el nivel de significancia (0,1), rechazamos la hipótesis nula. Concluimos que el algoritmo tiene un efecto significativo y determinante en el tiempo de ejecución.

Factor Tamaño

Contraste principal:

- Hipótesis nula (H_0): El factor Tamaño no influye en la variable dependiente ($\beta_1 = \beta_2 = \beta_3 = 0$)
- Hipótesis alternativa (H_1): El factor Tamaño influye en la variable dependiente ($\beta \neq 0$)

Test de la F:

La varianza explicada por el factor es de $7,65548\ x\ 10^{13}$ y la varianza no explicada es de $7,40228\ x\ 10^{10}$. Con lo que el F- Ratio = $7,65548\ x\ 10^{13}$/ $7,40228\ x\ 10^{10}$ = 1034,21

El F-Ratio obtenido es F(2,359) = 1034,21. Este valor es extremadamente alto, lo que resulta en un P-Valor de 0,0000. Al ser P-Valor menor que el nivel de significancia (0,1), rechazamos

la hipótesis nula. Concluimos que el tamaño del array tiene un efecto significativo y determinante en el tiempo de ejecución.

Factor Tipo

Contraste principal:

- Hipótesis nula (H_0): El factor Tipo no influye en la variable dependiente ($\gamma_1 = \gamma_2 = \gamma_3 = 0$)
- Hipótesis alternativa (H_1): El factor Tipo influye en la variable dependiente ($\gamma \neq 0$)

Test de la F:

La varianza explicada por el factor es de $4,0043\ x\ 10^{11}$ y la varianza no explicada es de $7,40228\ x\ 10^{10}$. Con lo que el F- Ratio = $4,0043\ x\ 10^{11}$ / $7,40228\ x\ 10^{10}$ = 5.41

El F-Ratio obtenido es F(2,359) = 5,41. Este valor resulta en un P-Valor de 0,0048. Al ser P-Valor menor que el nivel de significancia (0,1), rechazamos la hipótesis nula. Concluimos que el tipo de dato tiene un efecto estadísticamente significativo en el tiempo de ejecución, aunque su influencia es mucho menor que la de los otros factores.

Esto provoca que el factor 'Tipo' aparezca ahora como estadísticamente significativo. Sin embargo, como veremos en el análisis del R^2 y los Grupos Homogéneos, esta significancia es meramente matemática debido al gran tamaño muestral, pero carece de relevancia práctica, ya que su impacto real en el tiempo es casi nulo.

Comparación de Sr^2 y Sy^2

$Sr^2 = 7,40228\ x\ 10^{10}$	$Sr = \sqrt{7,40228\ x\ 10^{10}} = 272071,31$
$Sy^2 = 7,05322\ x\ 10^{14} / 377 = 1,87088\ x\ 10^{12}$	$Sy = \sqrt{1,87088\ x\ 10^{12}} = 1367801,16$

La comparación entre Sr^2 y Sy^2 muestra que la varianza residual del modelo es dos órdenes de magnitud menor que la varianza total de los datos (pasamos de 10^{12} a 10^{10}).

Al observar que Sr (272071,31) es considerablemente menor que Sy (1367801,16), verificamos que se reduce drásticamente la variabilidad. Con esto, concluimos que el modelo tiene una influencia muy significativa en la explicación de la variabilidad de los datos. Esto indica que el modelo es altamente efectivo, ya que captura la inmensa mayoría de la variabilidad total, dejando una parte muy pequeña debida al error aleatorio.

Bondad de Ajuste (R^2)

Coeficiente de determinación del modelo (R^2)

R^2 del modelo = $7,05322\ x\ 10^{14}\ -\ 2,65742\ x\ 10^{13}\ /\ 7,05322\ x\ 10^{14}\ =\ 0,9623$

El modelo tiene un R^2 de 0,9623, lo que significa que explica el 96,23% de la variabilidad en el tiempo de ejecución. Esto sugiere que el modelo es extremadamente efectivo en la predicción, ya que casi la totalidad de la variabilidad de los datos queda explicada por los factores controlados y sus interacciones, dejando muy poco margen al error aleatorio.

Coeficientes de determinación parciales de cada factor

R^2 del factor Algoritmo

$R^2\ =\ 2,24814\ x\ 10^{14}\ /\ 7,05322\ x\ 10^{14}\ =\ 0,3187$

El factor Algoritmo explica el 31,87% de la variabilidad. Es un impacto considerable, confirmando que la elección del método de búsqueda es determinante para el tiempo de ejecución.

R^2 del factor Tamaño $R^2\ =\ 1,5311\ x\ 10^{14}\ /\ 7,05322\ x\ 10^{14}\ =\ 0,2170$

El factor Tamaño explica el 21,70% de la variabilidad. Esto indica que la cantidad de elementos a procesar tiene un peso muy relevante en el rendimiento global.

R^2 del factor Tipo $R^2\ =\ 8,0086\ x\ 10^{11}\ /\ 7,05322\ x\ 10^{14}\ =\ 0,0011$

El factor Tipo explica solo el 0,11% de la variabilidad. Aunque su p-valor indicaba significancia estadística, su impacto real en el tiempo es prácticamente nulo en comparación con los otros factores.

Coeficientes de determinación parciales de las interacciones

R^2 de la interacción AB (Algoritmo-Tamaño)

$R^2\ =\ 2,98045x10^{14}/\ 7,05322x10^{14}\ =\ 0,4225$

La interacción entre Algoritmo y Tamaño tiene un R^2de 0,4225, explicando el 42,25% de la variabilidad.

Este es el elemento más importante del modelo, superando incluso a los efectos principales por separado. Esto demuestra matemáticamente que el rendimiento no depende solo de "qué

algoritmo usas" o "cuántos datos hay", sino de la complejidad computacional: cómo se comporta específicamente cada algoritmo al aumentar el tamaño.

R^2 de la interacción AC (Algoritmo-Tipo)
$R^2 = 1,46012x10^{12} / 7,05322x10^{14} = 0,0020$
Explica solo el 0,20% de la variabilidad. Aunque su p-valor (0,0007) indica que es significativa, su impacto es muy bajo.

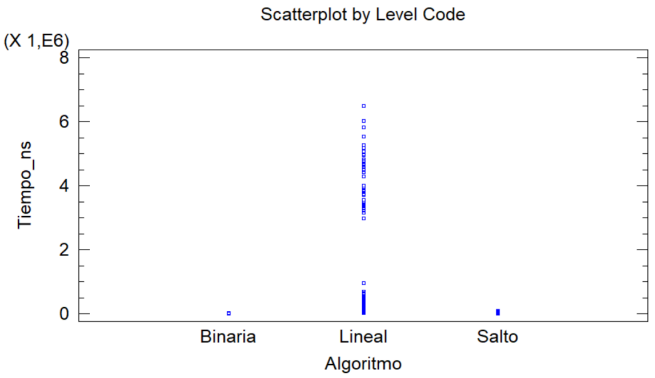
R^2 de la interacción BC (Tamaño-Tipo) $R^2 = 5,18596x10^{11} / 7,05322x10^{14} = 0,0007$

Esta interacción explica un insignificante 0,07% de la variabilidad y, como vimos en la tabla ANOVA, su p-valor no es significativo. Esto confirma que el tipo de dato no altera cómo escala el tiempo con el tamaño del array.

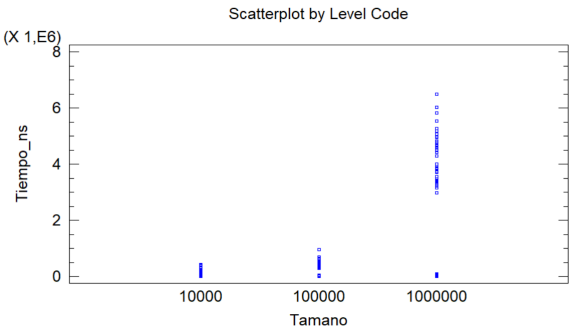
Análisis gráfico

Scatterplot

Los gráficos de dispersión nos permiten visualizar la distribución bruta de los datos y detectar patrones iniciales antes del análisis formal.



Se observa una diferencia drástica entre los grupos. Los algoritmos Binaria y Salto presentan tiempos extremadamente bajos, concentrados cerca de 0 en la escala vertical (millones de ns). Por el contrario, la Búsqueda Lineal muestra una dispersión enorme, con una columna de puntos que se extiende hasta los 6 millones de nanosegundos. Esto confirma visualmente que la Búsqueda Lineal es significativamente menos eficiente y más variable para los casos de prueba analizados.



El gráfico muestra claramente el efecto de escala. Para el tamaño de 10.000, todos los tiempos se mantienen bajos y compactos. Al aumentar a 100.000, se observa un leve incremento en la dispersión. Sin embargo, para 1.000.000, los tiempos se disparan, mostrando una gran variabilidad que alcanza el máximo de la escala. Esto indica que el aumento del tamaño del problema tiene un impacto directo y exponencial en el tiempo de ejecución, especialmente para los algoritmos menos eficientes.

Y el comportamiento del tiempo de búsqueda es uniforme independientemente del contenido del array.

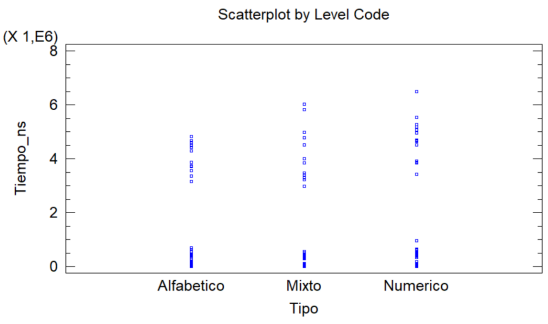
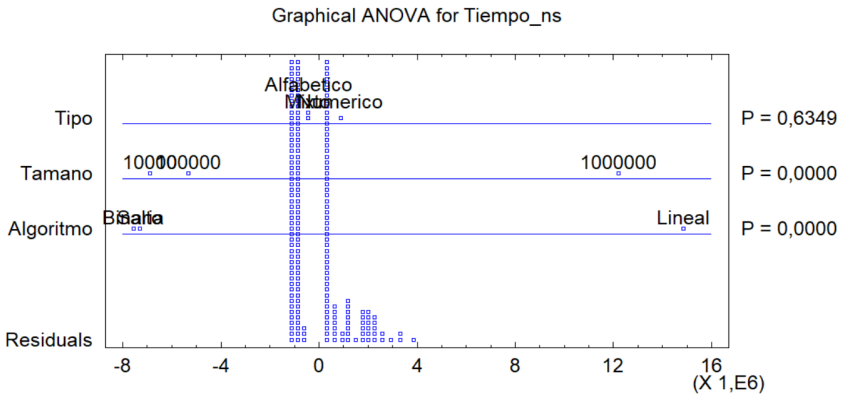


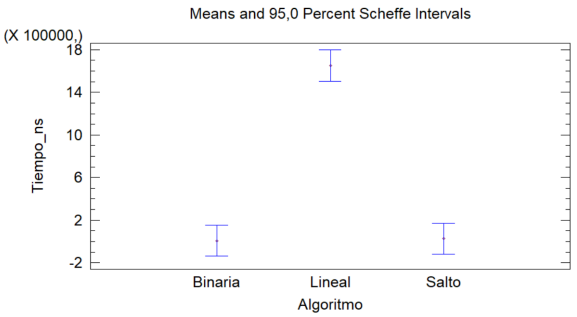
Tabla ANOVA



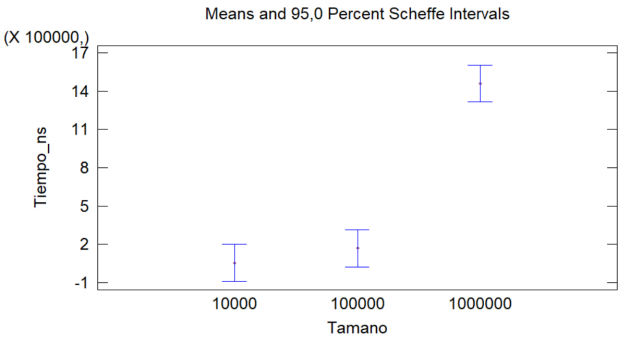
El gráfico ANOVA proporciona un resumen visual de la significancia estadística, mostrando una separación drástica para los factores Algoritmo y Tamaño, cuyos niveles "Lineal" y "1.000.000" aparecen muy desplazados hacia la derecha respecto a la media global, confirmando visualmente su impacto determinante en el aumento del tiempo de ejecución (coincidiendo con sus P-Valores de 0,0000). Por el contrario, los tres niveles del factor Tipo (Alfabético, Mixto, Numérico) aparecen apilados verticalmente casi en el mismo punto central con una línea de conexión mínima, lo que ratifica que la naturaleza del dato no influye en el rendimiento (P=0,6349), mientras que la distribución de los residuos en la parte inferior refleja la variabilidad remanente del modelo.

Gráficos de medias

Los gráficos de medias con intervalos de confianza (utilizando el método de Scheffé al 95%) nos permiten comparar visualmente si las diferencias entre los niveles de cada factor son estadísticamente significativas. Si los intervalos de dos niveles no se solapan, podemos concluir que sus medias son diferentes.



En este gráfico, la diferencia de rendimiento es evidente.
Lineal: Su media se sitúa extremadamente alta, con un intervalo de confianza muy alejado de los otros dos.
Binaria y Salto: Sus medias se encuentran muy próximas a cero en esta escala. Aunque visualmente parecen estar al mismo nivel debido a la magnitud de la búsqueda lineal, sus intervalos sugieren un rendimiento comparablemente eficiente frente al algoritmo lineal.
Conclusión: Se confirma que la Búsqueda Lineal es significativamente más lenta que los otros dos algoritmos.



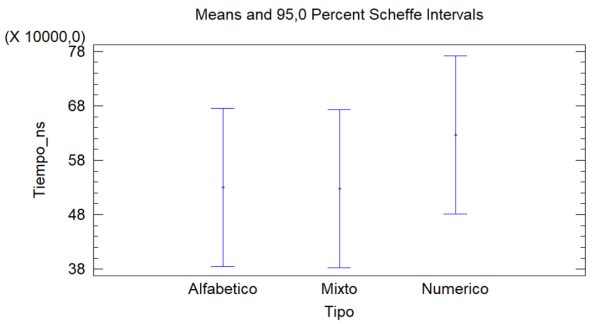
Este gráfico ilustra el impacto del volumen de datos.

10.000 y 100.000: Las medias para estos tamaños son bajas y sus intervalos de confianza se encuentran en la parte inferior del gráfico, relativamente cercanos entre sí.

1.000.000: Se observa un salto drástico. La media para el tamaño de un millón es significativamente superior, con un intervalo de confianza que no se solapa en absoluto con los tamaños menores.

Conclusión: El rendimiento se degrada significativamente al alcanzar el millón de elementos, confirmando el efecto de escala.

Análíticamente las medias de los tres tipos (Alfabético, Mixto, Numérico) están alineadas horizontalmente. Lo más destacable son los amplios intervalos de confianza (las barras verticales azules), que se solapan casi por completo entre sí.

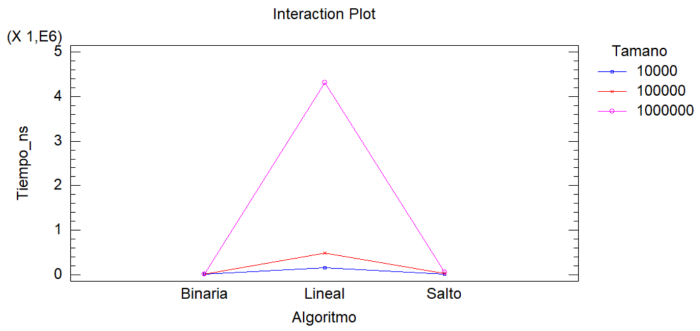


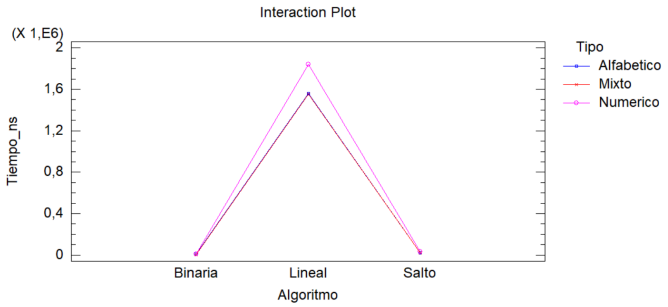
Conclusión: El solapamiento total de los intervalos confirma que no existen diferencias estadísticamente significativas entre los tres tipos de datos. Ratificamos la irrelevancia de este factor.

Gráficos de Interacción

Los gráficos de interacción nos permiten visualizar si el efecto de un factor depende del nivel de otro factor. Si las líneas del gráfico no son paralelas, indica que existe una interacción.

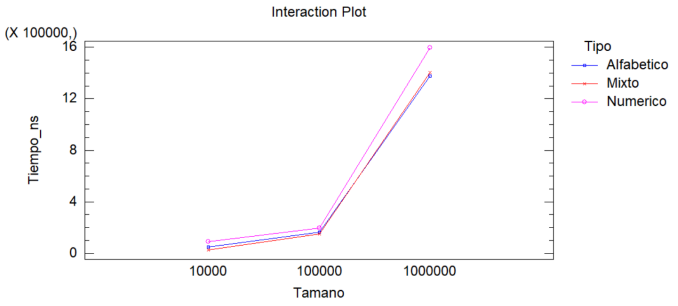
Algoritmo

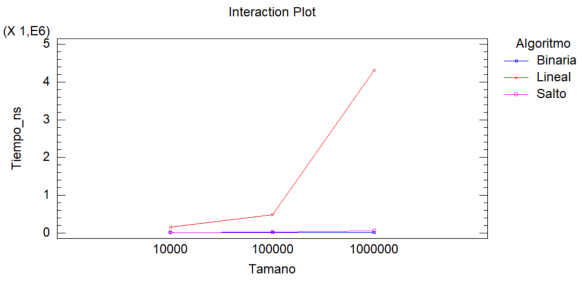




Se observa claramente la divergencia de comportamientos. Para los algoritmos Binaria y Salto, las líneas correspondientes a los tres tamaños (10k, 100k, 1M) aparecen colapsadas en la parte inferior, indicando que el aumento de tamaño apenas afecta su tiempo de ejecución en esta escala visual. Sin embargo, en el punto central (Lineal), la línea rosa (1.000.000 elementos) se dispara verticalmente, separándose drásticamente de las otras dos. Esto visualiza perfectamente la penalización de rendimiento que sufre la búsqueda lineal al aumentar el volumen de datos.
Y en el gráfico de la derecha muestra un patrón de "pirámide" donde las tres líneas (Alfabético, Mixto, Numérico) viajan completamente juntas y superpuestas. Aunque el punto central (Lineal) es mucho más alto que los extremos debido a su lentitud, el hecho de que las tres líneas sigan exactamente el mismo camino sin separarse confirma la ausencia de interacción. El algoritmo se comporta igual de rápido o lento sin importar el tipo de dato que esté procesando.

Tamaño

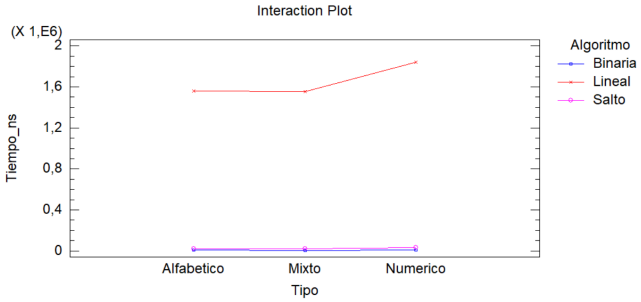


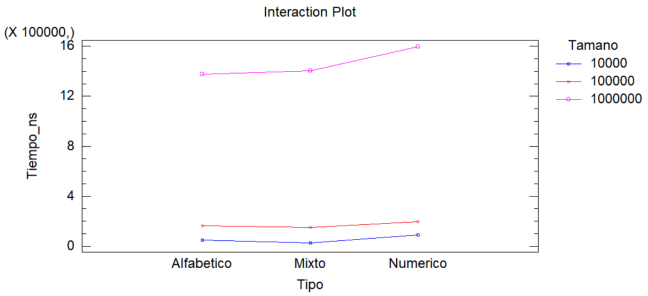


En el gráfico de la izquierda se observa un paralelismo casi perfecto entre las tres líneas (Alfabético, Mixto, Numérico). A medida que el tamaño aumenta de 100.000 a 1.000.000, el tiempo crece con la misma pendiente para todos los tipos de datos. El coste de aumentar el tamaño es independiente de la naturaleza de los datos.
Y en el gráfico de la derecha muestra cómo la Búsqueda Lineal sufre una degradación de rendimiento masiva al llegar al millón de elementos, disparándose hacia la parte superior del

gráfico. En contraste total, la Búsqueda Binaria y de Salto se mantienen planas en el eje horizontal. Esto demuestra empíricamente la diferencia de complejidad computacional: mientras que los algoritmos eficientes soportan el aumento de carga sin apenas coste temporal visible, el algoritmo lineal se vuelve ineficiente rápidamente.

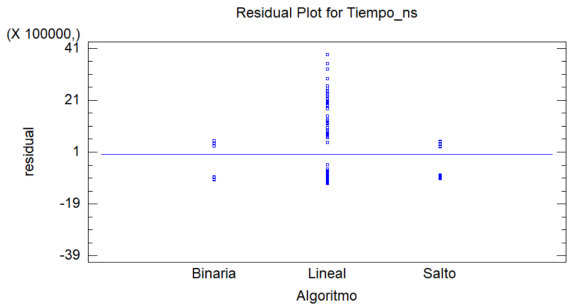
Tipo

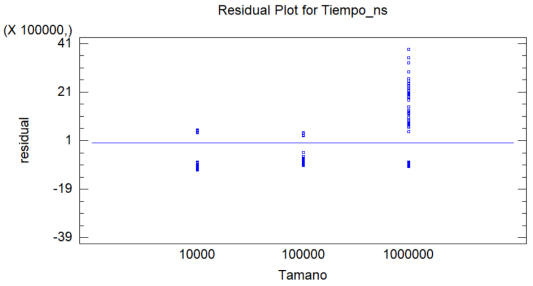


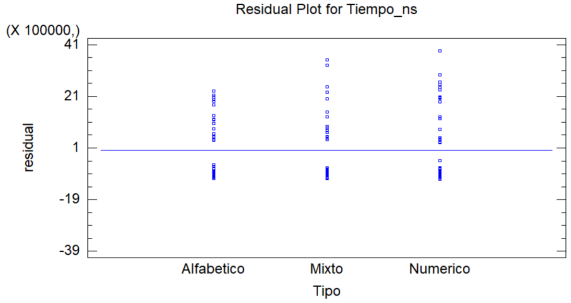


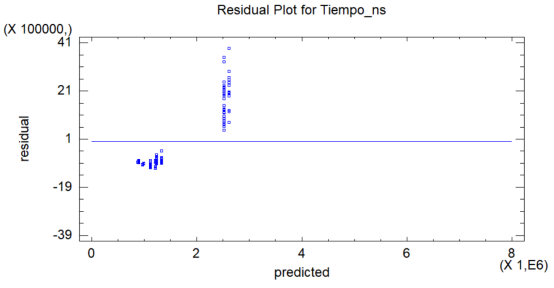
En el gráfico de la izquierda las líneas correspondientes a los tres algoritmos se muestran prácticamente horizontales y paralelas. Aunque el algoritmo Lineal (línea roja) presenta un ligero incremento en el nivel "Numérico", la estructura general se mantiene constante: la Búsqueda Lineal está muy separada superiormente, mientras que Binaria y Salto permanecen superpuestas en la base. Esto indica que la jerarquía de eficiencia de los algoritmos se mantiene intacta sin importar el tipo de dato.
Y en el de la derecha confirma la ausencia total de interacción entre el Tamaño y el Tipo . Las líneas que representan los tres tamaños son perfectamente paralelas y horizontales. Esto demuestra que el "coste" temporal de aumentar el tamaño del array es idéntico, independientemente de si estamos buscando números, letras o datos mixtos.

Gráficos de residuos









Análisis de Residuos por Factores: Al analizar la distribución de los residuos en función de los factores, detectamos patrones claros de varianza desigual. Mientras que el factor Tipo muestra una dispersión idéntica y estable en sus tres nivele, los factores Algoritmo y Tamaño revelan disparidades masivas. Específicamente, los residuos para la Búsqueda Lineal y el tamaño de 1.000.000 de elementos presentan una dispersión vertical enorme en comparación con los algoritmos eficientes (Binaria, Salto) y los tamaños menores, donde el error es prácticamente nulo; esto indica que la variabilidad del error no es constante en el experimento, sino que se dispara en los escenarios de mayor carga computacional.

Por su parte, el gráfico de residuos frente a los valores predichos ofrece el diagnóstico definitivo, mostrando una clara estructura de "embudo" hacia la derecha. Para los tiempos de ejecución cortos (valores predichos bajos), los residuos están extremadamente concentrados en torno al cero, pero a medida que aumenta el valor predicho correspondiente a las ejecuciones de la Búsqueda Lineal masiva, la nube de puntos se abre drásticamente. Este patrón confirma visualmente la existencia de heterocedasticidad, una consecuencia natural de comparar algoritmos cuyas escalas de tiempo difieren en varios órdenes de magnitud (O(n) vs $O(log n)).

Grupos homogéneos

Para profundizar en el análisis de las diferencias entre niveles, se ha realizado un Test de Rangos Múltiples (utilizando el método LSD con un nivel de confianza del 95%). Este análisis nos permite identificar qué niveles específicos de cada factor son estadísticamente equivalentes entre sí (forman un grupo homogéneo) y cuáles son significativamente diferentes.

Multiple Range Tests for Tiempo_ns by Algoritmo

Method: 95,0 percent LSD

Algoritmo	*Count*	*LS Mean*	*LS Sigma*	*Homogeneous Groups*
Binaria	126	7150,0	83586,1	X
Salto	126	25035,7	83586,1	X
Lineal	126	1,65198E6	83586,1	X

Contrast	*Sig.*	*Difference*	*+/- Limits*
Binaria - Lineal	*	-1,64483E6	232443,
Binaria - Salto		-17885,7	232443,
Lineal - Salto	*	1,62694E6	232443,

* denotes a statistically significant difference.

Al analizar el factor Algoritmo, el test distingue claramente dos grupos homogéneos.
Grupo 1: Formado por Búsqueda Binaria y Búsqueda de Salto. La alineación de las 'X' en la misma columna indica que, en el contexto global del experimento (dominado por la gran varianza de la búsqueda lineal), no existe una diferencia estadísticamente significativa entre las medias de estos dos algoritmos eficientes.
Grupo 2: Formado exclusivamente por la Búsqueda Lineal. Su media es tan elevada en comparación con los otros dos que se aísla en un grupo independiente.
Esto confirma que la Búsqueda Lineal es la única que presenta un rendimiento discrepante y significativamente inferior.

Para el factor Tamaño, el test identifica nuevamente dos grupos homogéneos, revelando el punto de inflexión del rendimiento.

Multiple Range Tests for Tiempo_ns by Tamano

Method: 95,0 percent LSD

Tamano	*Count*	*LS Mean*	*LS Sigma*	*Homogeneous Groups*
10000	126	54826,2	83586,1	X
100000	126	170366,	83586,1	X
1000000	126	1,45897E6	83586,1	X

Contrast	*Sig.*	*Difference*	*+/- Limits*
10000 - 100000		-115540,	232443,
10000 - 1000000	*	-1,40414E6	232443,
100000 - 1000000	*	-1,2886E6	232443,

* denotes a statistically significant difference.

Grupo 1: Los tamaños de 10.000 y 100.000 elementos se agrupan juntos. Esto indica que el incremento de tiempo al pasar de 10k a 100k no es lo suficientemente grande como para considerarlos estadísticamente diferentes en este modelo global.
Grupo 2: El tamaño de 1.000.000 se separa completamente en un grupo propio.

En el caso del factor Tipo de Dato, el resultado es contundente: se forma un único grupo homogéneo.

Multiple Range Tests for Tiempo_ns by Tipo

Method: 95,0 percent LSD

Tipo	*Count*	*LS Mean*	*LS Sigma*	*Homogeneous Groups*
Mixto	126	527613,	83586,1	X
Alfabetico	126	530083,	83586,1	X
Numerico	126	626467,	83586,1	X

Contrast	*Sig.*	*Difference*	*+/- Limits*
Alfabetico - Mixto		2469,84	232443,
Alfabetico - Numerico		-96384,1	232443,
Mixto - Numerico		-98854,0	232443,

* denotes a statistically significant difference.

Las columnas de 'X' muestran que Mixto, Alfabético y Numérico pertenecen al mismo grupo estadístico. Los contrastes entre pares muestran que ninguna diferencia supera el límite significativo.

Esto ratifica la naturaleza de los datos no introduce ninguna variación real en la velocidad de búsqueda de los algoritmos estudiados.

Intervalos de confianza al 90% para σ^2

A continuación, calculamos los intervalos de confianza para la varianza del error σ^2, utilizando la distribución Chi-Cuadrado con los grados de libertad del residuo (df = 359).

Analysis of Variance for Tiempo_ns - Type III Sums of Squares

Source	*Sum of Squares*	*Df*	*Mean Square*	*F-Ratio*	*P-Value*
MAIN EFFECTS					
A:Algoritmo	2,24814E14	2	1,12407E14	1518,54	0,0000
B:Tamano	1,5311E14	2	7,65548E13	1034,21	0,0000
C:Tipo	8,0086E11	2	4,0043E11	5,41	0,0048
INTERACTIONS					
AB	2,98045E14	4	7,45111E13	1006,60	0,0000
AC	1,46012E12	4	3,6503E11	4,93	0,0007
BC	5,18596E11	4	1,29649E11	1,75	0,1381
RESIDUAL	2,65742E13	359	7,40228E10		
TOTAL (CORRECTED)	7,05322E14	377			

All F-ratios are based on the residual mean square error.

Estimador: $Sr^2 = 7,40228\ x\ 10^{10}$

Estadístico Pivote: W = $2,65742\ x\ 10^{13}$ / σ² (para el grado de libertad 359)
A través de la tabla Inverse CDF de Statgraphics, obtenemos los valores críticos para 359 grados de libertad:

Inverse CDF
Distribution: Chi-Squared

CDF	*Dist. 1*	*Dist. 2*	*Dist. 3*
0,05	316,091		
0,1	325,119		
0,95	404,181		

Una cola

σ² < $2,65742\ x\ 10^{13}$ / 325,119 σ² < $8,1736\ x\ 10^{10}$

Nuestro valor estimado $7,40\ x\ 10^{10}$ es menor que el límite $8,17\ x\ 10^{10}$, por lo que es coherente.

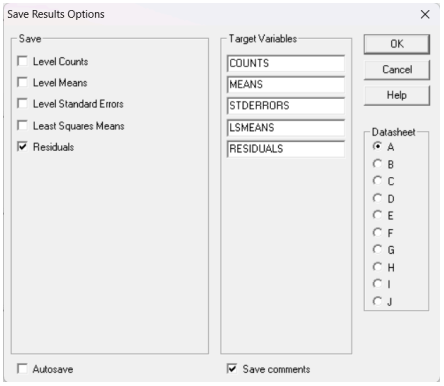
Dos colas

$2,65742\ x\ 10^{13}$ / 404,181 < σ² < $2,65742\ x\ 10^{13}$ / 316,091

Nuestro valor estimado $7,40\ x\ 10^{10}$ cae perfectamente entre $6,57\ y\ 8,40$, por lo que es coherente.

Reconocimiento de residuos

Primero vamos a salvar los residuos en la hoja A.



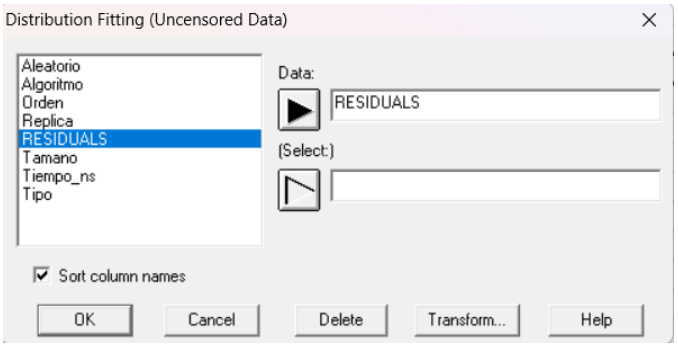
practica_Garcia_Garcia.sf6

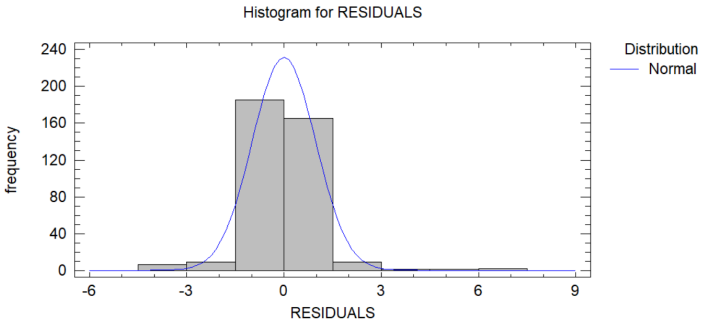
	Algoritmo	Tamano	Tipo	Replica	Tiempo_ns	Orden	Aleatorio	RESIDUALS
								Residuals standarizados
1	Binaria	100000	Numerico	4	7400	172	0,000457024395102	0,145261347222
2	Lineal	1000000	Mixto	1	5831400	113	0,00053262912274	5,97796163501
3	Lineal	10000	Mixto	6	110500	34	0,0132404086412	0,164791175833
4	Salto	100000	Alfabetico	7	16900	315	0,0141866151357	-0,114871485752
5	Salto	100000	Numerico	6	34600	300	0,0178343175405	0,16979097368
6	Salto	100000	Numerico	11	17700	305	0,0178407832956	0,107674834878
7	Lineal	100000	Numerico	9	564900	51	0,01998881042	-0,233613652319
8	Salto	10000	Alfabetico	10	5900	276	0,022855964842	-0,0875911067332
9	Lineal	10000	Numerico	3	399000	3	0,0247064103926	0,315790731096
10	Binaria	100000	Numerico	10	8900	178	0,0251408949435	0,150774613979
11	Binaria	10000	Alfabetico	1	1700	141	0,0264852354345	-0,0986962961874
12	Salto	100000	Mixto	2	17900	324	0,029504952157	-0,0521367584197
13	Lineal	10000	Alfabetico	10	162000	24	0,0321602723701	0,279198812075
14	Binaria	10000	Mixto	14	2000	168	0,0349754595243	-0,025381462927
15	Lineal	1000000	Alfabetico	11	4593000	109	0,0365613869471	1,53023291714
16	Lineal	100000	Alfabetico	2	424800	58	0,0374845334115	0,044129289781
17	Lineal	1000000	Alfabetico	7	3361500	105	0,0406760151859	-2,99615909083
18	Lineal	10000	Numerico	6	376600	6	0,0406897701159	0,233459280849
19	Lineal	10000	Numerico	8	171800	8	0,0435287570497	-0,519287244874
20	Lineal	100000	Mixto	5	436200	75	0,0486755012666	0,141387725998
21	Salto	1000000	Alfabetico	4	20700	354	0,051573911611	0,101018116595
22	Binaria	1000000	Mixto	9	7700	247	0,053455785216?	0,070342303554

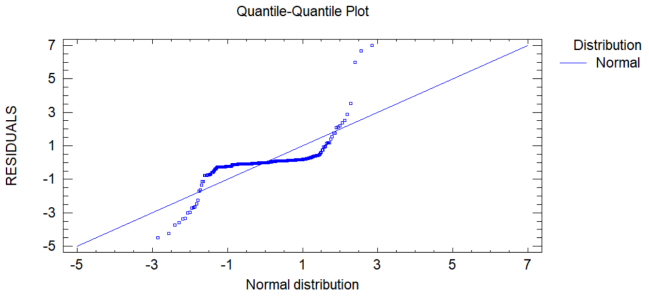
practica_Garcia_Garcia B C D E F G H I J

Añadimos la nueva columna de Residuos estandarizados.

Normalidad





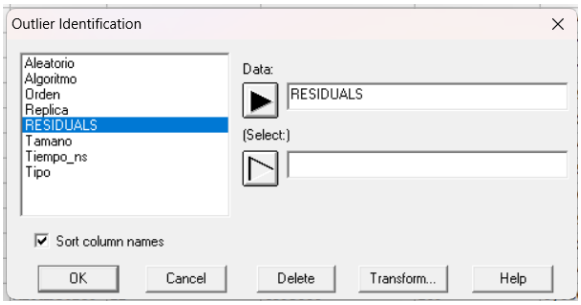


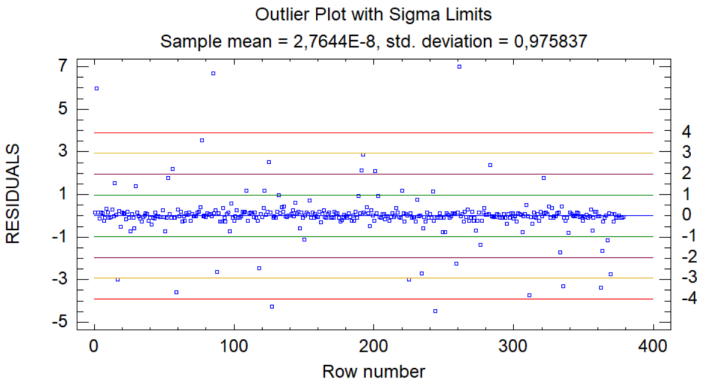
Como se observa en la imagen, la distribución de los residuos no se ajusta a la curva normal teórica (línea azul). Presenta una clara desviación: una concentración excesiva de datos en torno al valor central (cero) y colas más largas de lo esperado. Esto se debe a que la gran mayoría de las búsquedas (Binaria, Salto y tamaños pequeños) tienen errores mínimos, acumulándose en el centro.

El gráfico de probabilidad normal confirma esta desviación. Los puntos azules no siguen la línea recta diagonal, sino que forman una curva en forma de "S". Se aprecia una fuerte divergencia en los extremos (colas), lo que indica que los valores extremos de los residuos se alejan de lo que predeciría una distribución normal estricta.
Estas desviaciones gráficas se confirman numéricamente. El test de Shapiro-Wilk arroja un P-Valor de 0,0000, y otros tests como Kolmogorov-Smirnov confirman el resultado. Al ser el P-Valor menor que 0.05, rechazamos formalmente la hipótesis de normalidad

Esta falta de normalidad es una consecuencia estructural del diseño experimental y no un error de medición. Al comparar algoritmos con complejidades tan dispares, las escalas de variación de los residuos son radicalmente diferentes, lo que distorsiona la distribución global. A pesar de ello, dado el gran tamaño muestral (N=378) y el alto coeficiente de determinación, el análisis ANOVA es lo suficientemente robusto para validar los resultados sobre la significancia de los factores.

Datos Atípicos





Se observan varios puntos dispersos que exceden los límites de control de desviaciones estándar (líneas rojas). Estos puntos se corresponden con residuos positivos y negativos de gran magnitud relativa.

Sorted Values

Row	Value	Studentized Values Without Deletion	Studentized Values With Deletion	Modified MAD Z-Score
244	-4,48769	-4,59881	-4,74013	-23,8827
127	-4,25356	-4,35888	-4,47928	-22,6319
311	-3,71987	-3,81198	-3,89299	-19,7809
59	-3,59566	-3,68469	-3,75803	-19,1173
362	-3,38652	-3,47038	-3,53199	-18,0
...				
192	2,87655	2,94778	2,98638	15,4583
77	3,52933	3,61672	3,68619	18,9456
2	5,97796	6,12598	6,4652	32,0266
85	6,67116	6,83635	7,31509	35,7298
261	6,99939	7,1727	7,7305	37,4832

Grubbs' Test (assumes normality)
Test statistic = 7,1727
P-Value = 3,73501E-11

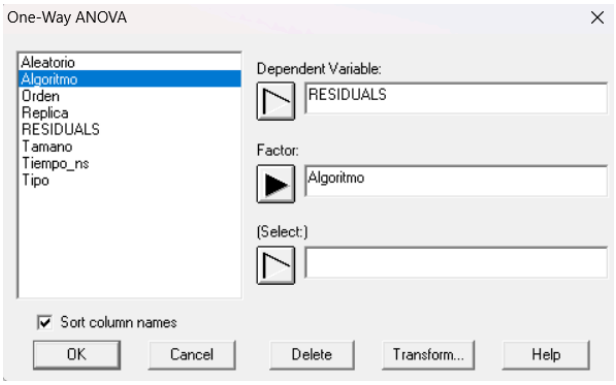
El contraste estadístico confirma numéricamente esta observación. El valor más extremo registrado (Fila 261, residuo estandarizado de 6,999) arroja un estadístico de prueba de 7,17 y un P-Valor extremadamente bajo, lo que lo clasifica formalmente como un "outlier" significativo bajo la asunción de normalidad.

Tras un análisis detallado, se ha verificado que estos supuestos "valores atípicos" corresponden sistemáticamente a las ejecuciones del algoritmo Búsqueda Lineal con el tamaño de array más grande (1.000.000).Dado que la complejidad de este algoritmo es lineal, es teóricamente esperable que sus tiempos de ejecución sean órdenes de magnitud superiores a los de los algoritmos logarítmicos y de raíz cuadrada. Esta diferencia de escala introduce naturalmente una varianza mucho mayor en ese subgrupo de datos. Por lo tanto, concluimos que estos valores no son errores experimentales ni fallos de medición, sino datos válidos que reflejan el comportamiento real e ineficiente del algoritmo bajo carga máxima. Se decide conservar estos datos en la muestra, ya que su eliminación sería artificial y ocultaría la verdadera magnitud de la diferencia de rendimiento entre los algoritmos, que es el objetivo principal de este estudio.

Homocedasticidad

Para analizar la homocedasticidad de los residuos miraremos el test de Levene.

- Algoritmo

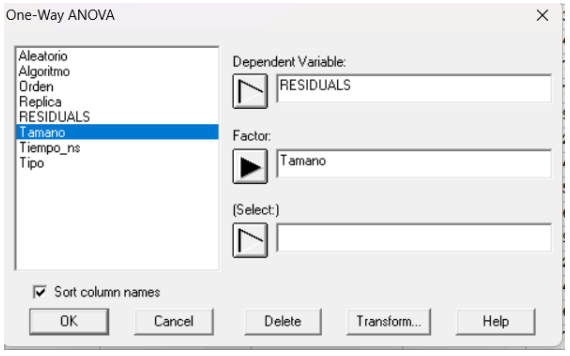


Variance Check

	Test	*P-Value*
Levene's	52,4389	0,0

Con base en el resultado de la prueba de Levene, el p-valor obtenido es 0,0, lo que es menor que 0.01. Esto implica que debemos rechazar la hipótesis nula de que las varianzas de los residuos son iguales entre los tres niveles del factor Algoritmo. Por lo tanto, hay heterocedasticidad en este factor.

- Tamaño

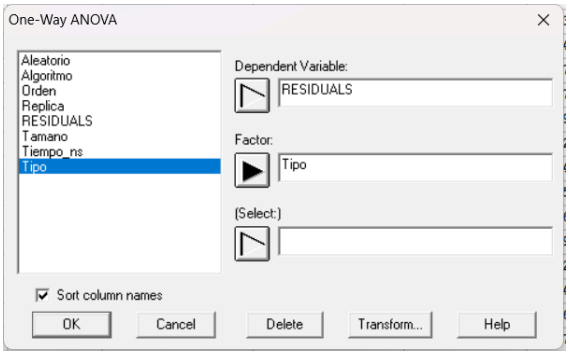


Variance Check

	Test	*P-Value*
Levene's	33,1976	0,0

Con base en el resultado de la prueba de Levene, el p-valor obtenido es 0,0, lo que es menor que 0.01. Esto implica que debemos rechazar la hipótesis nula de igualdad de varianzas. Existe una clara diferencia en la variabilidad de los residuos dependiendo del tamaño del array, confirmando que hay heterocedasticidad en este factor.

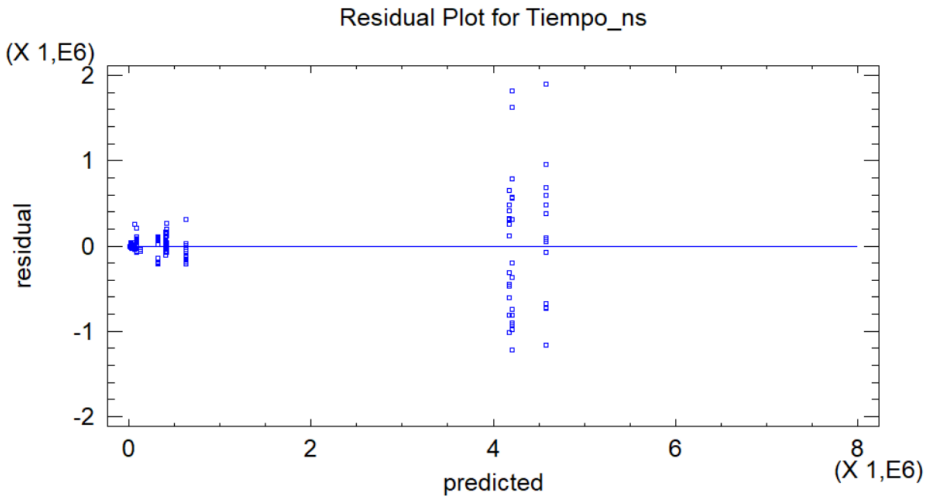
- Tipo



Variance Check

	Test	*P-Value*
Levene's	0,693635	0,500396

Con base en el resultado de la prueba de Levene, el p-valor obtenido es 0,5004, lo que es mayor que 0.01. Esto implica que no hay evidencia suficiente para rechazar la hipótesis nula. Por lo tanto, aceptamos que las varianzas de los residuos son iguales entre los tres niveles del factor Tipo. Concluimos que hay homocedasticidad en este factor.



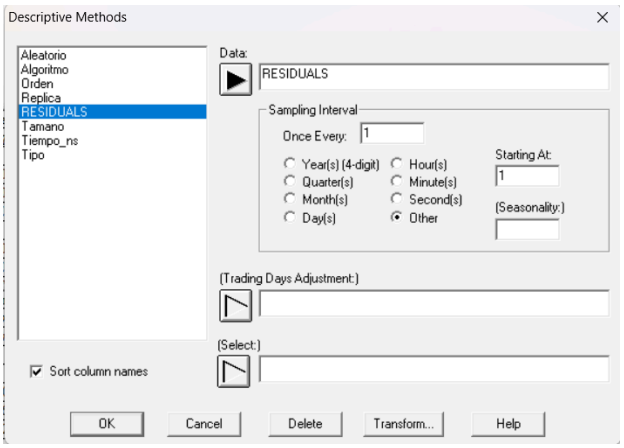
A pesar de la fuerte heterocedasticidad observada en los factores Algoritmo y Tamaño (confirmada por los tests de Levene con P=0,0), el gráfico de residuos frente a predicciones nos permite contextualizar este fenómeno.

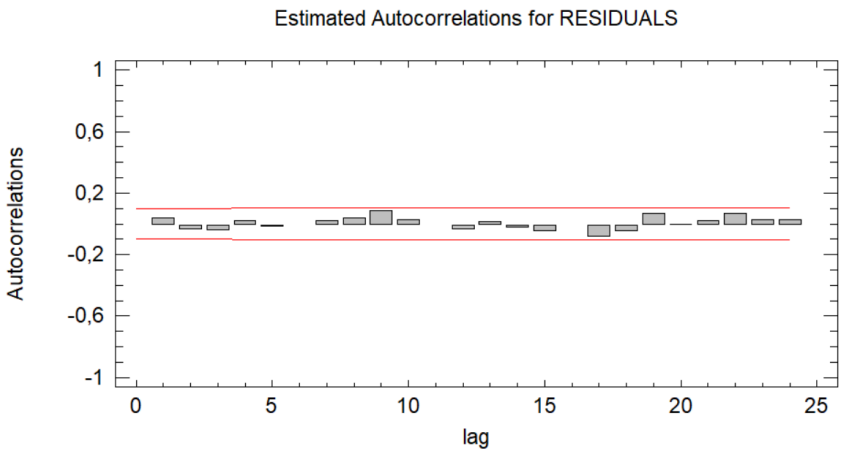
Se observa una clara estructura de "embudo": la varianza es mínima para los tiempos de ejecución bajos y se dispara para los tiempos altos. Esto indica que existe un problema de heterocedasticidad a nivel global. Sin embargo, este comportamiento es intrínseco al experimento: no podemos esperar la misma precisión absoluta (varianza) al medir 5.000 ns que al medir 5.000.000 ns.

Por lo tanto, aunque el supuesto de homocedasticidad no se cumple estrictamente, consideramos que el modelo sigue siendo válido para comparar los efectos principales,

siempre y cuando se tenga en cuenta que las predicciones para la Búsqueda Lineal tendrán un margen de error mucho mayor que para los otros algoritmos.

Independencia





En el correlograma de autocorrelaciones estimadas, observamos que todas las barras (incluida la del primer retardo o lag) se mantienen con un amplio margen dentro de los límites de confianza al 95% (líneas rojas).

A diferencia de otros casos donde el primer retardo podría mostrar valores limítrofes, aquí el coeficiente de autocorrelación es muy bajo, lo cual indica de forma clara que no existe una correlación significativa entre los residuos consecutivos. Esto confirma la efectividad del proceso de aleatorización de los datos realizado previamente.

Conclusiones finales

En este estudio, se ha diseñado y ejecutado un experimento completo para analizar el rendimiento de tres algoritmos de búsqueda (Lineal, Binaria y de Salto), evaluando su comportamiento bajo diferentes cargas de trabajo (Tamaño del array) y naturalezas de datos (Tipo). A partir de las 378 observaciones recogidas y su posterior análisis estadístico, se extraen las siguientes conclusiones:

Factores Determinantes del Rendimiento

El análisis ANOVA ha revelado que el Algoritmo empleado y el Tamaño del Array son los factores críticos que determinan el tiempo de ejecución (P-Valor = 0,0000).

- Se ha demostrado empíricamente la superioridad de la Búsqueda Binaria y la Búsqueda de Salto, cuyos tiempos se mantienen estables y bajos.
- Por el contrario, la Búsqueda Lineal ha mostrado un rendimiento drásticamente inferior, siendo el único algoritmo que sufre una degradación masiva al aumentar el volumen de datos.

La Irrelevancia del Tipo de Dato

Uno de los hallazgos más interesantes ha sido la nula influencia práctica del Tipo de Dato. Aunque el análisis detectó una significancia estadística marginal, el factor "Tipo" explica únicamente un 0,11% de la variabilidad total. Los gráficos de medias y dispersión confirman que buscar números, letras o datos mixtos tiene el mismo coste computacional.

Interacción Crítica: La Divergencia de Complejidad

El modelo ha identificado una fuerte interacción entre Algoritmo y Tamaño, que por sí sola explica el 42,25% de la variabilidad de los datos. Esto valida la teoría de la complejidad computacional: la diferencia entre algoritmos no es constante, sino que se amplifica exponencialmente a medida que crece el problema (de 10.000 a 1.000.000 de elementos).

Se observa que la Búsqueda de Salto es 3 veces más lenta en promedio, lo cual es coherente con su complejidad teórica.

Validez del Modelo y Análisis de Residuos

El modelo matemático obtenido explica el 96,23% de la variabilidad observada R^2, lo que indica un ajuste excelente.

El chequeo de residuos mostró desviaciones en la normalidad y la homocedasticidad (patrón de "embudo"). Sin embargo, se ha determinado que esto no es un fallo del diseño, sino una consecuencia natural de comparar procesos con escalas de tiempo tan dispares (microsegundos frente a milisegundos).

- Los valores identificados como "atípicos" en la Búsqueda Lineal se mantuvieron en el análisis por ser datos legítimos que reflejan su ineficiencia.
- Finalmente, la prueba de independencia (Box-Pierce) confirmó la ausencia de correlación serial, validando la eficacia de la aleatorización realizada durante la toma de datos.

En resumen, el experimento ha cumplido sus objetivos al cuantificar con precisión cómo la elección del algoritmo adecuado es, con diferencia, la decisión más importante en el diseño de sistemas de búsqueda eficientes para grandes volúmenes de datos.

Lucas García García