Validación Rigurosa de Algoritmos de Prueba de la Conjetura de Goldbach en un Rango Seleccionado

1. Introducción

La Conjetura de Goldbach se erige como uno de los problemas no resueltos más antiguos y célebres dentro del campo de la teoría de números.[1] Su enunciado, aparentemente sencillo, ha desafiado a matemáticos durante siglos, lo que subraya su importancia y la dificultad inherente en su demostración. El presente informe tiene como objetivo llevar a cabo una validación rigurosa de los resultados obtenidos por dos algoritmos (versiones 1.005.2 y 1.005.4) diseñados para verificar esta conjetura dentro de un rango numérico específico definido por el usuario. La validación se centrará en el análisis de la metodología empleada por los algoritmos, la eficiencia de su ejecución y la significación de sus hallazgos en el contexto más amplio de la Conjetura de Goldbach. El análisis se basa en los resúmenes de ejecución proporcionados para ambas versiones de los algoritmos.

2. Entendiendo la Conjetura de Goldbach: El Fundamento de los Algoritmos

La Conjetura de Goldbach, en su formulación moderna, establece que "todo número par mayor que 2 puede expresarse como la suma de dos números primos".[1] Esta afirmación, a pesar de su simplicidad, tiene profundas implicaciones para la comprensión de la distribución de los números primos y ha sido objeto de intensa investigación matemática. Es importante destacar que esta conjetura fue propuesta originalmente por el matemático ruso Christian Goldbach en una carta dirigida a Leonhard Euler en 1742.[2] En aquel entonces, la convención matemática consideraba al número 1 como primo, por lo que la formulación inicial de Goldbach era ligeramente diferente, aunque equivalente a la versión moderna.

Dentro del ámbito de la Conjetura de Goldbach, es crucial distinguir entre la forma "fuerte" o "binaria" (la que los algoritmos buscan verificar) y la forma "débil" o "ternaria". La Conjetura de Goldbach fuerte, también conocida como la conjetura binaria, es la que se ha definido anteriormente, mientras que la conjetura débil establece que todo número impar mayor que 5 puede expresarse como la suma de tres números primos.[1] A pesar de los numerosos intentos y de la vasta evidencia computacional acumulada, es fundamental señalar que la Conjetura de Goldbach, en su forma fuerte, sigue siendo un problema no resuelto en matemáticas.[1] Por lo tanto, las verificaciones computacionales, como las realizadas por los algoritmos en cuestión, proporcionan una fuerte evidencia a favor de la conjetura dentro de los

rangos probados, pero no constituyen una demostración matemática universal.

3. Análisis del Rango Numérico Probado: El Alcance del Examen de los Algoritmos

Los algoritmos proporcionados evalúan la Conjetura de Goldbach para un conjunto de números pares generados mediante la fórmula $M=10^{50}-n\times2$, donde el parámetro n varía desde 0 hasta 10^{100}. Al sustituir los valores extremos con un n=100000, podemos determinar el rango específico de números pares que fueron sometidos a prueba.

El número más grande probado por los algoritmos se obtiene cuando n=0, lo que resulta en $M=10^{50}$. El número más pequeño probado se calcula al sustituir por ejemplo con un n=100000, tenemos como resultado: $M= 10^{50} - 2 \times100000=10^{50}-200000$.

De este análisis se desprende que los algoritmos examinaron un total de 100001 números pares consecutivos, comprendidos en un intervalo que va desde 10^{50} hasta $10^{50}-200000$. Es importante destacar la magnitud de estos números, todos ellos extremadamente grandes, situados en la vecindad de 10^{50}. Si bien este rango contiene una cantidad considerable de números, sigue siendo una porción infinitesimalmente pequeña en comparación con la totalidad de los números pares mayores que 2. Dado que 10^{50} es un número par (al ser una potencia de 10) y $n\times2$ siempre es par, la diferencia entre ellos también será un número par. Esto confirma que los algoritmos están efectivamente probando números que cumplen con el criterio de la Conjetura de Goldbach.

4. Evaluación de la Suficiencia de los Números Primos: El Conjunto de Herramientas de los Algoritmos

Ambas versiones de los algoritmos informan haber cargado 1228 números primos. Para evaluar si esta cantidad de primos es adecuada para verificar la Conjetura de Goldbach en el rango especificado, es necesario considerar la distribución de los números primos en magnitudes tan elevadas. El Teorema del Número Primo (TNP) proporciona una aproximación asintótica para la función de conteo de primos, denotada como $\pi(x)$, que indica la cantidad de números primos menores o iguales a un número dado x. La aproximación del TNP es $\pi(x)\approx\ln(x)x$.[1] Este teorema fundamental nos permite estimar la densidad de los números primos en el rango que están probando los algoritmos.

Para obtener una estimación del número de primos que podríamos esperar encontrar

hasta la mitad del número más grande probado (aproximadamente 5×10^{49}, ya que si $M=p+q$, al menos uno de los primos p o q debe ser menor o igual a $M/2$), podemos aplicar el Teorema del Número Primo. El logaritmo natural de 5×10^{49} es aproximadamente $\ln(5)+49\times\ln(10)\approx1.609+49\times2.3026\approx114.44$. Por lo tanto, el número de primos hasta 5×10^{49} se estima en $\pi(5\times10^{49})\approx114.445\times10^{49}\approx4.37\times10^{47}$.

Al comparar esta estimación astronómicamente grande de aproximadamente 4.37×10^{47} con los 1228 primos cargados por los algoritmos, se hace evidente que los algoritmos están utilizando una fracción extremadamente pequeña de los números primos que existen en el rango relevante. Esto plantea interrogantes sobre la exhaustividad de su prueba. La estrategia empleada por los algoritmos consiste en iterar a través de su lista pre-cargada de 1228 primos. Para cada primo p en esta lista, verifica si $M-P$ también es un número primo. Si esta condición se cumple, entonces se ha encontrado una partición de Goldbach, aunque dentro de las limitaciones de su conjunto de primos.

En este contexto, resulta relevante la información proporcionada en el fragmento [5], que menciona que en todas las verificaciones realizadas hasta 4×10^{14} por Richstein, el primo más pequeño en la partición de Goldbach nunca fue mayor que 5569. Esta observación empírica sugiere que, incluso para números pares muy grandes, a menudo es posible encontrar una partición de Goldbach donde uno de los primos es relativamente pequeño. Esto podría explicar por qué los algoritmos, utilizando primos hasta un cierto límite (el primo número 1228), podrían tener éxito en encontrar particiones para el rango probado. Una estimación aproximada del primo número 1228, utilizando la aproximación $n\ln(n)$, da como resultado alrededor de $1228\times\ln(1228)\approx8731$, que está en el mismo orden de magnitud que 5569, lo que respalda aún más esta línea de razonamiento. Por lo tanto, la suficiencia de los 1228 primos cargados depende de la distribución de los primos en las particiones de Goldbach para números de la magnitud de 10^{50}. Si bien la evidencia hasta 10^{14} sugiere que los primos pequeños son frecuentes en dichas particiones, no hay garantía de que este patrón se mantenga exactamente igual en magnitudes mucho mayores.

5. Análisis Comparativo del Rendimiento de los Algoritmos: Eficiencia y Optimización

La siguiente tabla resume las métricas clave de rendimiento para ambas versiones de los algoritmos, lo que permite una comparación directa de su eficiencia:

Métrica	Versión 1.005.2	Versión 1.005.4
Tiempo de carga de primos (ms)	0.80	0.69
Tiempo de verificación (ms)	40783.27	9838.47
Tiempo de escritura (ms)	254.07	189.06
Ms encontrados	100001	100001
Ms no encontrados	0	0
Promedio por M (ms)	0.26 (σ: 0.08)	0.68 (σ: 0.37)
Máximo tiempo en M	1.46	6.78
Tiempo total (ms)	41038.14	10028.22
Tamaño del archivo (KB)	996	1330
Multiprocesamiento	No mencionado	Sí

La diferencia más notable entre las dos versiones es la reducción significativa en el "Tiempo de verificación" en la versión 1.005.4 (9838.47 ms) en comparación con la versión 1.005.2 (40783.27 ms). Esta mejora drástica en el rendimiento se debe a la implementación de multiprocessing.

Se observa una ligera disminución en el "Tiempo de carga de primos" en la versión 1.005.4 (0.69 ms) en comparación con la versión 1.005.2 (0.80 ms), lo que indica una optimización menor en la forma en que los números primos se cargan o se acceden.

La descripción de la versión 1.005.4 menciona explícitamente "+ multiprocessing". El multiprocesamiento permite al algoritmo dividir la tarea de verificar diferentes

números (o diferentes pares de primos para el mismo número) entre múltiples núcleos de procesador, lo que conduce a una aceleración significativa en el tiempo de ejecución, especialmente para tareas computacionalmente intensivas como las pruebas de primalidad. Esta es la razón de la sustancial reducción en el tiempo de verificación.

El "Promedio por M" es más alto en la versión 1.005.4 (0.68 ms) en comparación con la versión 1.005.2 (0.26 ms). Esto podría parecer contradictorio inicialmente. Sin embargo, podría explicarse por la sobrecarga asociada con la gestión y coordinación de múltiples procesos en un entorno de multiprocesamiento. Además, la distribución de los tiempos de procesamiento en el rango probado podría ser diferente en la versión optimizada. La desviación estándar más alta en la versión 1.005.4 (0.37 ms frente a 0.08 ms) respalda esta idea de una mayor variabilidad en los tiempos de procesamiento de números individuales.

El "Tiempo de escritura" es ligeramente menor en la versión 1.005.4 (189.06 ms frente a 254.07 ms), lo que podría ser una consecuencia de la ejecución general más rápida o de optimizaciones menores en el proceso de escritura de datos. El "Tamaño del archivo" es mayor para la versión 1.005.4 (1330 KB frente a 996 KB). Esto podría deberse al almacenamiento de metadatos adicionales relacionados con la ejecución del multiprocesamiento o, potencialmente, a un registro más detallado de los resultados.

El "Máximo tiempo en M" es significativamente mayor para la versión 1.005.4 (6.78 ms) en comparación con la versión 1.005.2 (1.46 ms), y este máximo ocurrió en el mismo valor de M (10^50−2×72406). Esta discrepancia es interesante y podría indicar que, si bien el rendimiento promedio mejoró con el multiprocesamiento, ciertos números específicos podrían tardar más en procesarse debido a la forma en que se distribuyen las tareas paralelas o a la naturaleza de sus particiones de Goldbach dentro del conjunto limitado de primos.

6. Verificación de los Resultados Reportados: Consistencia y Precisión dentro del Alcance de los Algoritmos

Ambas versiones de los algoritmos informan consistentemente "'M's encontrados: 100001" y "¿M's no encontrados: 0". El número de valores de M probados coincide con el rango de 'n' (de 0 a 100000 inclusive), que efectivamente da como resultado 100001 números. Esto indica que, dentro de las limitaciones de los algoritmos (específicamente el conjunto limitado de primos utilizados), logramos encontrar al menos una forma de expresar cada número par probado como la suma de dos primos de su lista cargada (o donde un primo está en la lista y el otro, M−P, se determina que es primo mediante su prueba de primalidad interna).

7. Metodologías de Verificación Computacional para la Conjetura de Goldbach: Contexto y Comparación

Los métodos computacionales típicos utilizados para verificar la Conjetura de Goldbach para números pares grandes generalmente implican: generar una lista de números primos hasta un cierto límite utilizando algoritmos eficientes como la Criba de Eratóstenes o sus variantes optimizadas; para cada número par N que se está probando, iterar a través de los primos p≤N/2 y verificar si N−P también es primo; emplear pruebas de primalidad probabilísticas (como Miller-Rabin) para números muy grandes para determinar eficientemente la primalidad con un alto grado de confianza; y utilizar computación paralela y redes distribuidas para manejar los enormes requisitos computacionales para verificar la conjetura para números cada vez más grandes.[1]

El enfoque adoptado por los algoritmos proporcionados, con su dependencia de una lista precargada de solo 1228 primos, con una estrategia específica. El número par probado M se puede expresar como P+Q, donde P es uno de los primos precargados y Q=M−P también se determina que es primo (utilizando la prueba de primalidad gmpy2.is_prime). Esto contrasta con una búsqueda exhaustiva de todos los pares de primos hasta M/2.

En cuanto al alcance de las verificaciones computacionales conocidas de la Conjetura de Goldbach, el fragmento [1] indica que, hasta 2013, la conjetura había sido verificada computacionalmente para números pares hasta 4×10^18. El fragmento [5] menciona una verificación hasta 10^14 en el año 2000. Si bien estas verificaciones han alcanzado escalas impresionantes, la magnitud de 10^50 está significativamente más allá de los límites documentados públicamente de la verificación completa. Los algoritmos actuales están probando números en un rango que no ha sido

completamente verificado por métodos exhaustivos. El fragmento [4] ofrece una perspectiva más amplia, afirmando que "para todos los propósitos prácticos, estamos al 0% del camino para verificar que la conjetura de Goldbach es cierta para números hasta el número de Graham". Si bien el número de Graham es vastamente mayor que 10^50, esta cita resalta la inmensa escala del problema y las limitaciones del poder computacional actual para una verificación verdaderamente exhaustiva en todos los números.

8. Limitaciones y Alcance de los Algoritmos Actuales en la Validación Rigurosa

La limitación más significativa de los algoritmos proporcionados es el número fijo y relativamente pequeño de primos (1228) que utilizan. Si bien estos primos podrían ser suficientes para encontrar *una* partición de Goldbach para el rango probado (como indican los resultados), no proporcionan una validación rigurosa en el sentido de explorar todos los pares de primos posibles o probar que *solo* estos primos pueden formar la partición. Podrían existir particiones válidas de Goldbach que involucren primos no incluidos en este conjunto limitado.

Además, los algoritmos solo probaron un rango muy estrecho de números pares alrededor de 10^50 (específicamente, un intervalo de 200000). Si bien esto proporciona evidencia a favor de la conjetura dentro de esta vecindad específica, aunque grande, no permite generalizaciones a otros números de magnitud similar o a números pares más pequeños.

El éxito de los algoritmos en la búsqueda de particiones de Goldbach podría atribuirse a la observación del fragmento [5] de que, incluso para números grandes, uno de los factores primos en una partición de Goldbach a menudo puede ser relativamente pequeño. Si los 1228 primos utilizados por los algoritmos cubren este rango de primos "pequeños" de manera efectiva, podrían tener éxito en encontrar al menos una partición. Sin embargo, esta es una suposición basada en evidencia empírica hasta una escala mucho menor.

Es crucial reiterar que la verificación computacional, incluso si se extendiera a rangos mucho mayores, no constituye una demostración matemática de la Conjetura de Goldbach.[14] Un solo contraejemplo, si existiera, refutaría la conjetura, independientemente de cuántos números se hayan verificado computacionalmente.

Finalmente, la rigurosidad de la validación depende de la fiabilidad y precisión de esta prueba de primalidad (gmpy2.is_prime), especialmente para números de esta magnitud.

9. Conclusión: Evaluación Experta de los Resultados de los Algoritmos

En resumen, ambas versiones de los algoritmos lograron encontrar al menos una partición de Goldbach (dentro de las limitaciones de su metodología) para los 100001 números pares probados en el rango de 1050 a 1050–200000, utilizando una lista precargada de 1228 números primos. Se observó una mejora significativa en el rendimiento de la versión 1.005.4, que se atribuye en gran medida a la implementación del multiprocesamiento.

Si bien estos resultados proporcionan evidencia computacional adicional que respalda la Conjetura de Goldbach para este rango específico de números extremadamente grandes, no constituyen una demostración matemática de la conjetura. Es fundamental reconocer las limitaciones de los algoritmos, en particular la dependencia de un conjunto pequeño y fijo de primos, el rango estrecho de números probados en relación con la escala de 10^50 y la falta de detalles sobre la prueba de primalidad empleada.

Para mejorar la rigurosidad y el alcance de futuras pruebas computacionales, se sugieren las siguientes líneas de investigación: analizar el conjunto específico de 1228 primos utilizados por los algoritmos y los criterios para su selección; probar un rango más amplio y representativo de números pares alrededor de 10^50 y potencialmente en diferentes magnitudes; investigar la distribución de los factores primos más pequeños encontrados en las particiones de Goldbach dentro de este rango; considerar el uso de listas de primos pre-calculadas más grandes o la generación dinámica de primos para aumentar la probabilidad de encontrar una partición de Goldbach si existe; proporcionar detalles sobre la prueba de primalidad utilizada, incluyendo su precisión y limitaciones para números de esta magnitud; y comparar el rendimiento y los resultados con otros esfuerzos conocidos de verificación computacional de la Conjetura de Goldbach.

GUSTAVO ROYET ROJAS
Barranquilla, ABR-2025

Works cited

1. Goldbach's conjecture - Wikipedia, accessed April 4, 2025, https://en.wikipedia.org/wiki/Goldbach%27s_conjecture
2. www.britannica.com, accessed April 4, 2025, https://www.britannica.com/science/Goldbach-conjecture#:~:text=Goldbach%20conjecture%2C%20in%20number%20theory,mathematician%20Leonhard%20Euler%20in%201742.
3. Goldbach conjecture | Prime numbers, Number theory, Diophantine equations - Britannica, accessed April 4, 2025, https://www.britannica.com/science/Goldbach-conjecture
4. GOLDBACH CONJECTURE Definition & Meaning - Dictionary.com, accessed April 4, 2025, https://www.dictionary.com/browse/goldbach-conjecture
5. Goldbach's conjecture - The Prime Glossary, accessed April 4, 2025, https://primes.utm.edu/glossary/page.php?sort=goldbachconjecture
6. Goldbach's Conjecture - Art of Problem Solving, accessed April 4, 2025, https://artofproblemsolving.com/wiki/index.php/Goldbach%27s_Conjecture
7. The Goldbach conjecture and related problems | Additive Combinatorics Class Notes | Fiveable, accessed April 4, 2025, https://library.fiveable.me/additive-combinatorics/unit-12/goldbach-conjecture-related-problems/study-guide/GqVx2dR5O4dKjz8a
8. Mathematicians Will Never Stop Proving the Prime Number Theorem, accessed April 4, 2025, https://www.quantamagazine.org/mathematicians-will-never-stop-proving-the-prime-number-theorem-20200722/
9. Prime number theorem - Wikipedia, accessed April 4, 2025, https://en.wikipedia.org/wiki/Prime_number_theorem
10. Prime number theorem - Britannica, accessed April 4, 2025, https://www.britannica.com/science/prime-number-theorem
11. Prime Number Theorem - Platonic Realms, accessed April 4, 2025, https://platonicrealms.com/encyclopedia/Prime-Number-Theorem
12. Little Proof of the Prime Number Theorem - DataScienceCentral.com, accessed April 4, 2025, https://www.datasciencecentral.com/simple-proof-of-prime-number-theorem/
13. Proving Goldbach's Strong Conjecture by Analyzing Gaps Between Prime Numbers and their Digits - Opast Publishing Group, accessed April 4, 2025, https://www.opastpublishers.com/open-access-articles/proving-goldbachs-strong-conjecture-by-analyzing-gaps-between-prime-numbers-and-their-digits.pdf
14. Proof of Goldbach's Conjecture[v1] | Preprints.org, accessed April 4, 2025, https://www.preprints.org/manuscript/202409.2235/v1