

Compartir información confidencial de la cadena de suministro: Un estudio sobre las vulnerabilidades en los sistemas de cifrado RSA utilizando el algoritmo shor

*Sharing sensitive supply chain information: A study on
vulnerabilities in RSA encryption systems from the use of the shor
algorithm*

*Compartilhamento de informações sensíveis da cadeia de
suprimentos: Um estudo sobre vulnerabilidades nos sistemas de
criptografia RSA a partir do uso do algoritmo de shor*

Rodrigo de Souza¹

rodrigo.souza119@fatec.sp.gov.br

Antônio Cesar Galhardi²

antonio.galhardi@cpspos.sp.gov.br

1 – Fatec de Franco da Rocha

2 - Unidade de Pós-Graduação, Extensão e Pesquisa - Centro Estadual de Educação
Tecnológica Paula Souza

Resumen:

Este trabajo propone un análisis comparativo de algoritmos, basado en el algoritmo de Shor. En la obra se utiliza la herramienta de inteligencia artificial ChatGPT, pidiéndole que escriba códigos en el lenguaje Qiskit y la computación cuántica en dos momentos: agosto de 2023 y febrero de 2024. Los resultados mostraron que hubo un aprendizaje considerable de la herramienta ChatGPT en relación con las mejores estrategias para implementar el algoritmo, de tal manera que, considerando la última sugerencia, a partir de febrero de 2024, existe un riesgo considerable para el cifrado RSA, si este tipo de código se implementa sin supervisión en entornos de acceso remoto a computadoras cuánticas. Cabe destacar que el riesgo para el cifrado RSA tiene implicaciones directas en la logística contemporánea, especialmente en el intercambio de información sensible en la cadena de suministro, aumentando la necesidad de desarrollar soluciones de seguridad de la información que permitan prevenir o mitigar las consecuencias de dicho avance.

Palabras clave: criptografía, machine learning, algoritmo Shor, ciberseguridad.

Abstract:

This work proposes a comparative analysis of algorithms, considering Shor's Algorithm as a basis. The ChatGPT artificial intelligence tool is used in the work, asking it to create codes in the Qiskit language and quantum computing at two moments: August 2023 and February 2024. The results showed that there was considerable learning from the ChatGPT tool in relation to better strategies for implementing the algorithm, in such a way that, considering the last suggestion, from

Recebido
Received
Recibido
Jun. 2024

Aceito
Accepted
Aceptado
Dez. 2024

Publicado
Published
Publicado
Jan./Mar. 2025
Ene./Mar. 2025

<https://git.fateczl.edu.br>

e_ISSN
2965-3339

DOI
10.29327/2384439.3.2-3

São Paulo
v. 3 | n. 2
v. 3 | i. 2
e32015
Janeiro-Março
January-March
Enero-Marzo
2025



February 2024, there is a considerable risk to RSA encryption, if this type of code is implemented without supervision in remote access environments to quantum computers. It is noteworthy that the risk to RSA encryption has direct implications for contemporary logistics, especially in the sharing of sensitive information in the supply chain, increasing the need to develop information security solutions that make it possible to prevent or mitigate the consequences of such advancement.

Keywords: *cryptography, machine learning, Shor's algorithm, cybersecurity.*

Resumo:

Este trabalho propõe uma análise comparativa de algoritmos, considerando-se como base o Algoritmo de Shor. No trabalho emprega-se a ferramenta de inteligência artificial ChatGPT, solicitando-a que elabore códigos na linguagem e computação quântica Qiskit em dois momentos: agosto de 2023 e fevereiro de 2024. Os resultados mostraram que houve um aprendizado considerável da ferramenta ChatGPT em relação às melhores estratégias de implementação do algoritmo, de tal forma que, considerando a última sugestão, de fevereiro de 2024, tem-se um risco considerável à criptografia RSA, caso este tipo de código seja implementado sem supervisão nos ambientes de acesso remoto a computadores quânticos. Destaca-se que o risco à criptografia RSA tem implicações diretas na logística contemporânea, especialmente no compartilhamento de informações sensíveis na cadeia de suprimentos, aumentando a necessidade de desenvolvimento de soluções em segurança da informação que possibilitem prevenir ou atenuar as consequências de tal avanço.

Palavras-chave: *criptografia, aprendizado de máquina, algoritmo de Shor, cibersegurança.*

1. INTRODUCCIÓN

El cifrado es la técnica de cifrar (codificar) y descifrar (decodificar) un mensaje por medio de una clave. Su uso se remonta a períodos anteriores al advenimiento de la informática contemporánea, pero fue con el uso de computadoras electromecánicas durante la Segunda Guerra Mundial, especialmente la máquina alemana llamada Enigma, que la criptografía alcanzó un nuevo nivel. Con la aparición de los semiconductores y la arquitectura de circuitos integrados, alrededor de la década de 1960, la informática comenzó a tener mayor capacidad de procesamiento, almacenamiento y agilidad.

La criptografía RSA (*Rivest-Shamir-Adleman*) surgió en 1978 como una tecnología asimétrica, es decir, basada en dos claves, una de las cuales es pública y es producto de dos claves privadas, que otorgan acceso a los datos. El beneficio del modelo RSA es la dificultad de factorizar números primos grandes y es esta técnica la utilizada en la generación de la clave pública. Romper este tipo de criptografía requeriría factorizar un número en sus constituyentes principales, y esta operación requiere un nivel de procesamiento extremadamente alto, debido al tamaño del número utilizado en la clave pública (Uzeda et al., 2022).

La tecnología RSA estuvo libre de amenazas hasta que en 1994 el matemático estadounidense Peter Shor propuso un modelo de algoritmo que requeriría su implementación en ordenadores cuánticos y que sería capaz de romper los cifrados RSA con facilidad, ya que se apoya en la potencia de la computación cuántica para la factorización (Shor, 1994).

La creciente interconexión de las cadenas de suministro contemporáneas, impulsada por las tecnologías web y la necesidad de eficiencia operativa, ha sacado a la luz una serie de desafíos relacionados con la seguridad de la información. El intercambio de información sensible entre diferentes entidades, como proveedores, fabricantes, distribuidores y minoristas, ha sido una práctica común, ya que tiene como objetivo optimizar el flujo de mercancías. Sin embargo, esta práctica conlleva un riesgo sustancial, especialmente en un escenario en el que la ciberseguridad es una preocupación creciente. Las vulnerabilidades en los sistemas de TI, la falta de un cifrado sólido y los métodos de autenticación ineficientes pueden exponer la información a ciberataques, lo que resulta en fuga de datos, interrupciones de la cadena de suministro y pérdida de confianza de los clientes.

Este riesgo se amplifica cuando se considera la posibilidad de ataques de computadoras cuánticas, que podrían romper los cifrados tradicionales, como RSA, mucho más rápido que las computadoras clásicas. En este contexto, es fundamental que las organizaciones adopten medidas preventivas para garantizar la integridad de la información sensible en sus cadenas de suministro. Además, el uso de tecnologías emergentes como *blockchain* e inteligencia artificial puede ayudar a fortalecer la seguridad y la resiliencia de las cadenas de suministro, garantizando la integridad y confidencialidad de las informaciones compartidas.

Hasta mediados de la década de 2000, el Algoritmo de *Shor* era visto mucho más como una curiosidad académica que como un instrumento práctico, sin embargo,

dados los avances en el desarrollo de las computadoras cuánticas, su aplicabilidad ha pasado de una situación hipotética a una posibilidad (VIEIRA; ALBUQUERQUE, 2020). Además, con la irrupción de los sistemas de inteligencia artificial generativa, de los cuales el más famoso y quizás uno de los más avanzados es ChatGPT, lanzado a finales de 2022, se lanzó la posibilidad de traducir el Algoritmo de Shor a un lenguaje cuántico como *Qiskit*, que se puede implementar en el entorno cuántico de acceso remoto de IBM llamado IBM Quantum Experience, abre posibilidades inexploradas hasta ahora de uso inapropiado de estas herramientas en un intento de interrumpir los sistemas de ciberseguridad basados en el cifrado RSA.

En este estudio, la pregunta básica de investigación es: ¿Cuál es el nivel de asertividad del código *Qiskit* propuesto por ChatGPT para el algoritmo de *Shor*? Para responder a esta pregunta, este artículo se basó inicialmente en una búsqueda bibliográfica sobre el estado del arte de la computación cuántica y la IA generativa. Luego, se llevó a cabo una investigación empírica, en la que se le pidió a ChatGPT que propusiera un código *Qiskit* para el Algoritmo de *Shor*, el cual fue sometido a un análisis comparativo entre el código propuesto y el algoritmo original, con el fin de verificar el nivel de asertividad. En un tercer momento, se hizo la misma solicitud a ChatGPT, luego de 6 meses, con el fin de verificar el nivel de aprendizaje de la plataforma con relación a la construcción del algoritmo.

2. ANTECEDENTES TEÓRICOS

La computación actual se basa en la tecnología de semiconductores, utilizada en la construcción de transistores, que son dispositivos capaces de conducir y no conducir electricidad; en otras palabras: el nombre semiconductor proviene del hecho de que el tipo de material utilizado tiene propiedades conductoras y aislantes. La existencia de esta propiedad semiconductor es esencial para el funcionamiento de las computadoras, ya que el sistema informático se basa en bits, que son los valores de 0 (cero) y 1 (uno), utilizados en el código binario.

Los semiconductores se forman a partir de una estructura cristalina de silicio, un material aislante. Su estructura cristalina se dopa (adición de pequeñas cantidades) con elementos que la convertirán en semiconductor. El proceso de dopaje se puede realizar de dos maneras diferentes (Weste; Harris, 2009):

- a) Dopaje negativo: se insertan átomos de fósforo con un electrón más;
- b) Dopaje positivo: se insertan átomos de indio, con un electrón menos.

La ausencia de un electrón o la presencia de un electrón adicional hará que la estructura cristalina sea semiconductor, lo que permitirá el proceso de codificación binaria. Tenga en cuenta que el funcionamiento del transistor solo tiene un estado a la vez, o solo se puede leer un bit a la vez. En general, en el dopaje negativo, con el flujo de corriente, hay el bit 1 y en ausencia de corriente, hay el bit 0. En el dopaje positivo ocurre lo contrario (Weste; Harris, 2009).

En la computación cuántica, no se procesa con transistores. Los elementos utilizados son subpartículas atómicas, como electrones y fotones, manipuladas

individualmente. Dado que la agitación de las moléculas es un fenómeno termodinámico, un aspecto relevante es la necesidad de mantener dichos ordenadores en el cero absoluto, 0°K (-273°C), ya que a temperaturas más altas, habrá agitación de partículas subatómicas. En el cero absoluto, solo las partículas seleccionadas se agitan (Portugal; Marquezino, 2019).

El término "cuántico" se refiere al uso de los preceptos de la mecánica cuántica para determinar y explicar los procesos computacionales. En relación con estos conceptos, la idea de entrelazamiento cuántico es especialmente importante, ya que es la forma en la que se intercambia información en un ordenador cuántico. El entrelazamiento cuántico es la paridad de espín (momento angular) de dos partículas subatómicas que, una vez juntas, mantienen dicha paridad aunque luego se separen (Toneli, 2022).

Otro aspecto relevante es el principio de incertidumbre de la *Heisenberg*, según el cual, de forma simplificada, no es posible definir con precisión la posición y la velocidad de una partícula subatómica. La elección de una cantidad implica la determinación probabilística de la segunda. Desde la perspectiva de la computadora cuántica, esto se aplica a la posibilidad de que un bit cuántico (o *qubit*) puede asumir el valor 0, el valor 1 o ambos simultáneamente. Se estima que un ordenador de unos 100 *qubit* tiene la misma capacidad de procesamiento que la suma de todos los ordenadores del mundo actual (Parada, 2019).

2.1 El algoritmo de *Shor*

El algoritmo *de Shor* fue desarrollado por Peter *Shor* en 1994 y permite factorizar un número N en tiempo polinómico, utilizando un ordenador cuántico. Sus etapas son (Shor, 1994):

- a) Preparación del estado cuántico: Un estado cuántico debe prepararse en una superposición de todos los valores posibles para la entrada del algoritmo. Para ello, se utilizan *qubits* en dos registros, uno para la entrada y otro para la salida. El registro de entrada se inicializa en un estado superpuesto con todos los valores de entrada posibles, mientras que el registrador de salida se inicializa en un estado con un valor de 1.
- b) Transformada cuántica de Fourier: Se debe aplicar una transformada cuántica de Fourier al registro de entrada. Esta transformada convierte la superposición de todos los valores posibles en un conjunto de amplitudes que corresponden a los factores primos del número que se va a factorizar.
- c) Medición y posprocesamiento: debe medir el registro de salida y realizar el posprocesamiento clásico para encontrar los factores primos. La medición del registro de salida contrae la superposición de estados, lo que da como resultado un único valor que representa uno de los factores principales del número que se va a factorizar. Una vez que se encuentra un factor, se utiliza un algoritmo de factorización clásico para encontrar el otro factor.

El pseudocódigo del algoritmo, basado en el trabajo de Shor, (1994), se muestra a continuación:

Início

1. Elija un número entero aleatorio entre 1 y $N-1$.
2. Calcule el máximo común divisor de a y N . Si el m.d.c. es diferente de 1, entonces a y N tienen un factor en común y se encuentra la factorización de N .
3. Crie un circuito cuántico que actúe sobre la base $\{|0\rangle, |1\rangle, \dots, |N-1\rangle\}$ con una puerta unitaria $U(x)$ que calcula $a^x \bmod N$.
4. Prepare un registro cuántico en un estado superpuesto utilizando un número suficiente de *qubits* para almacenar N .
5. Aplique el puerto $U(x)$ al estado superpuesto. Esto crea una superposición de estados que representan todas las potencias posibles de un *mod* N .
6. Aplique la transformada cuántica de Fourier al estado superpuesto.
7. Mida el registro cuántico. El resultado de la medición es un número r que es un múltiplo entero del período de la función $a^x \bmod N$.
8. Calcule el máximo común divisor de N y $a^{(r/2)} + 1$. Si el m.d.c. es diferente de 1, entonces se encuentra la factorización de N . Si no es así, vuelva al paso 1 y elija un nuevo valor de a .

Fin

El pseudocódigo anterior se puede entender de la siguiente manera: El algoritmo comienza eligiendo un número entero aleatorio " a ", que se utiliza para calcular las posibles potencias de " a " *mod* N . A continuación, se calcula el máximo común divisor entre " a " y N . El siguiente paso es crear un circuito cuántico que actúe sobre la base $\{|0\rangle, |1\rangle, \dots, |N-1\rangle\}$ e incluyen una puerta unitaria $U(x)$ que calcula $a^x \bmod N$ para cada x de la base. A continuación, se prepara un registro cuántico en un estado superpuesto y la puerta $U(x)$ se aplica al estado superpuesto. Esto crea una superposición de estados que representan todas las potencias posibles de un *mod* N . La transformada cuántica de Fourier se aplica al estado superpuesto, y luego se realiza una medición del registro cuántico. El resultado de la medición es un número r que es un múltiplo entero del período de la función $a^x \bmod N$. El período es el entero positivo más pequeño r para el cual $a^r \bmod N = 1$. Finalmente, se calcula el máximo común divisor de N y $a^{(r/2)} + 1$. Si el m.d.c. es diferente de 1 y diferente de N , entonces se encuentra la factorización de N . De lo contrario, el algoritmo debe repetirse con un nuevo valor aleatorio de " a " (Shor, 1994). Cabe señalar que el propósito de la función "*mod*" es devolver el resto de una división. El resto de la sala es un elemento clave para los procesos de cifrado y descifrado.

2.2 Cifrado RSA

El sistema de encriptación RSA se basa en la siguiente suposición: dados los factores primos p y q , con valores grandes en cada uno, dichos valores forman una clave privada a partir de una clave pública $n = p \cdot q$ (CASTRO, 2019). Esto significa que tanto la codificación como la decodificación dependen de p y q . Este enfoque criptográfico sigue el Teorema Fundamental de la Aritmética en N , según el cual: considerando un entero positivo $n = 1$, podemos escribirlo de la siguiente manera $n = p_1^{e_1} \cdot \dots \cdot p_k^{e_k}$, donde $1 < p_1 < \dots < p_k$ y $p =$ primo. Es decir, en resumen, sabemos que cualquier número natural se puede descomponer en

constituyentes primos.

La aplicación del sistema RSA requiere la conversión de letras en números y la definición de un número para los espacios entre palabras. Una vez hecho esto, se elige un "exponente de codificación" y el número correspondiente al mensaje se eleva por este exponente. Poco después, el resultado se divide por un "módulo de cifrado". El proceso de decodificación sigue la lógica inversa (Teixeira, 2020).

2.3 ChatGPT

ChatGPT es un *chatbot*, es decir, un robot para conversaciones de chat, basado en Inteligencia Artificial generativa. La IA generativa suele basarse en una Red Neuronal Artificial, y actúa a través de la investigación libre sobre un tema determinado, realizando una síntesis en la respuesta, que también se almacena como aprendizaje. Este enfoque de inteligencia artificial se basa en el procesamiento del lenguaje natural y esto es relevante dada la necesidad de acercar la interfaz de la IA al usuario final (Ramesh et al., 2017).

3. DESARROLLO DEL TEMA

La investigación se basa en una comparación entre el Algoritmo *Shor* propuesto en el lenguaje cuántico *Qiskit*, desarrollado por ChatGPT y el algoritmo original, con el fin de verificar si existe una correspondencia entre ambos. Dicha comparación se produjo en dos momentos: el primero en agosto de 2023 y el segundo, en febrero de 2024. El uso de dos momentos se debe al objetivo de investigar el nivel de aprendizaje y mejora de la plataforma en relación con el algoritmo propuesto. En la plataforma ChatGPT, el texto provocador era "Haga el algoritmo de *Shor* en el lenguaje *Qiskit*". El resultado del primer código generado (agosto de 2023) se puede ver en la Figura 1.

Al comparar línea por línea la propuesta de ChatGPT con el algoritmo original de Peter Shor, en pseudocódigo, se observa un alto nivel de fidelidad. Hay que tener en cuenta que las tres primeras líneas del código de la figura 1 se refieren a las bibliotecas importadas. El código propuesto también utilizó un número para la factorización (15), como una forma de ejemplificar su operatividad. El análisis línea por línea es el siguiente:

1. Shor: Elija un número entero aleatorio entre 1 y $N-1$;
ChatGPT: $N = 15$;
2. Shor: Calcule el máximo común divisor de a y N . Si el m.d.c. es diferente de 1, entonces a y N tienen un factor en común y se encuentra la factorización de N ;
ChatGPT: No realizado;
3. Shor: Cree un circuito cuántico que actúe sobre la base $\{|0\rangle, |1\rangle, \dots, |N-1\rangle\}$ con una puerta unitaria $U(x)$ que calcula $a^x \text{ mod } N$;
ChatGPT: `qc = QuantumCircuit(4, 4)`;
4. Shor: Preparar un registro cuántico en un estado superpuesto utilizando un número suficiente de qubits para almacenar N ;

- ChatGPT: `qc.h(rango(3))`
`qc.x(3);`
5. Shor: Aplique el puerto $U(x)$ al estado superpuesto. Esto crea una superposición de estados que representan todas las potencias posibles de un mod N ;
 ChatGPT: `shor = Shor(N, a=2,`
`quantum_instance=Aer.get_backend('qasm_simulator')`
`qc.append(shor.circuit, qc.qubits[:3]);`
 6. Shor: Aplicar la transformada cuántica de Fourier al estado superpuesto;
 ChatGPT: `qc.append(QFT(3, inverse=True), qc.qubits[:3]);`
 7. Shor: Mida el registro cuántico. El resultado de la medición es un número r que es un múltiplo entero del período de la función $a^x \text{ mod } N$;
 ChatGPT: `qc.measure(rango(3), rango(3));`
 8. Shor: Calcule el máximo común divisor de N y $a^{(r/2) + 1}$. Si el m.d.c. es diferente de 1, entonces se encuentra la factorización de N . De lo contrario, vuelva al paso 1 y elija un nuevo valor de a ;
 ChatGPT: `backend = Aer.get_backend('qasm_simulator')`
`trabajo = ejecutar (control de calidad, backend, disparos = 1024)`

Figura 1 – Algoritmo de Shor en Qiskit propuesto por ChatGPT en agosto de 2023

```

from qiskit import QuantumCircuit, Aer, execute
from qiskit.algorithms import Shor
from qiskit.circuit.library import QFT

# Define o número a ser fatorado
N = 15

# Cria um circuito quântico para o algoritmo de Shor
qc = QuantumCircuit(4, 4)

# Aplica as portas necessárias no primeiro registrador
qc.h(range(3))
qc.x(3)

# Aplica o algoritmo de Shor
shor = Shor(N, a=2, quantum_instance=Aer.get_backend('qasm_simulator'))
qc.append(shor.circuit, qc.qubits[:3])

# Aplica a transformada de Fourier inversa no primeiro registrador
qc.append(QFT(3, inverse=True), qc.qubits[:3])

```

Fuente: elaboración propia (2024)

Con la excepción del ítem 2, se cumplieron los otros puntos del Algoritmo de Shor. El punto 2, aunque no es propuesto por ChatGPT, no es esencial para el resultado, ya que se trata de una verificación a priori de la factorización. En cuanto a los demás, desde el punto de vista de las funcionalidades, se cumplieron a cabalidad.

El segundo código generado (febrero de 2024) se muestra en la figura 2.

Figura 2 – Algoritmo de Shor en Qiskit propuesto por ChatGPT en febrero de 2024

```

from math import gcd
from random import randint
from qiskit import Aer, QuantumCircuit, transpile, assemble, execute
from qiskit.visualization import plot_histogram

def modular_exponentiation(a, exponent, N):
    """Perform modular exponentiation a^exponent mod N."""
    return pow(a, exponent, N)

def quantum_modular_exponentiation(qc, a, exponent, N, n):
    """Implement modular exponentiation a^exponent mod N using a quantum circuit."""
    for i in range(n):
        qc.x(i) if (exponent & (1 << i)) else None
    for i in range(n):
        quantum_multiply_by_a_mod_N(qc, a, 2 ** i, N, n)
    return qc

def quantum_multiply_by_a_mod_N(qc, a, exponent, N, n):
    """Implement the controlled modular multiplication by a."""
    for i in range(n):
        qc.swap(i, n + i)
    for i in range(n):
        qc.cx(i, n + i) if (exponent & (1 << i)) else None
    for i in range(n - 1, -1, -1):
        qc.cx(i, n + i) if (exponent & (1 << i)) else None
    return qc

def find_period(N, a, n):
    """Find the period r of the function f(x) = a^x mod N."""
    qc = QuantumCircuit(n * 2, n)

    quantum_modular_exponentiation(qc, a, 1, N, n)

    for q in range(n):
        qc.measure(q, q)

    backend = Aer.get_backend('qasm_simulator')
    result = execute(qc, backend, shots=1).result()
    counts = result.get_counts(qc)

    measured_period = int(list(counts.keys())[0], 2)
    measured_period = measured_period if measured_period else 2 ** n

    return measured_period

def shor(N, n_trials=20):
    """Shor's Algorithm to find prime factors of N."""
    if N % 2 == 0:
        return 2

    for _ in range(n_trials):
        a = randint(2, N - 1)
        if gcd(a, N) != 1:
            return gcd(a, N)
        else:
            measured_period = find_period(N, a, n)
            if measured_period % 2 != 0:
                continue

            if modular_exponentiation(a, measured_period // 2, N) != N - 1:
                p = gcd(modular_exponentiation(a, measured_period // 2, N) + 1, N)
                q = gcd(modular_exponentiation(a, measured_period // 2, N) - 1, N)
                return p, q

# Example usage
N = 15
n = 4 # Number of qubits needed to represent N
factors = shor(N)
print("Factors of", N, "are:", factors)

```

Fuente: elaboración propia (2024).

Tras la propuesta del nuevo código, por parte de la *ChatGPT*, se analizó en comparación con el algoritmo original. El nuevo código (febrero de 2024) propuesto por *ChatGPT*, además de los ítems ya cumplidos en el algoritmo anterior (agosto de 2023), también cumplía con el ítem 2 (máximo común divisor), que no se había cumplido en el primer código generado. El siguiente paso fue el análisis comparativo de los algoritmos, con el fin de verificar cuál de los dos códigos hace un uso más racional y optimizado de la herramienta *Qiskit*. Los criterios para este análisis fueron (Johnston; Harrigan; Segovia, 2019):

a) Control y flexibilidad: es la capacidad de controlar todos los aspectos del algoritmo y la implementación, así como su flexibilidad para ajustar y optimizar diferentes partes del código de acuerdo con las necesidades específicas del problema.

b) Implementación de la transformada cuántica de Fourier (QFT): comprueba cómo se implementa QFT en el código. Se considera que una implementación manual puede ofrecer un mayor control sobre cada detalle del QFT, permitiendo ajustes específicos según sea necesario. Por otro lado, al optar por el uso de una clase predefinida, se produce una simplificación del código, sin embargo, la precisión de los cálculos puede verse limitada.

c) Personalización y Optimización: es la capacidad que ofrece el modelo para personalizar el algoritmo en su conjunto, con el fin de satisfacer las necesidades específicas del problema en cuestión, es decir, dicha personalización incluye el ajuste de parámetros, la optimización de partes del código y la adaptación del algoritmo a diferentes escenarios de uso.

4. RESULTADOS Y DISCUSIÓN

En cuanto al control del código y su flexibilidad, se observa que, en el código de febrero de 2024, se implementa de manera manual el algoritmo de Shor, lo que ofrece un mayor control sobre cada paso del proceso, incluyendo valores de referencia para variables y eventuales aproximaciones. De esta manera, es posible optimizar cada paso del algoritmo de acuerdo con las características específicas del problema, como ajustar la elección de 'a' dinámicamente, maximizando la probabilidad de encontrar el período rápidamente.

En el código de agosto de 2023, el algoritmo de Shor se encapsula en una clase estandarizada proporcionada por la biblioteca *Qiskit*. Si bien esta estrategia ofrece comodidad, puede limitar el control sobre el proceso, ya que algunas optimizaciones o ajustes específicos pueden no ser posibles sin modificar directamente el código fuente de la clase "*Shor*".

En cuanto a la implementación de la Transformada Cuántica de Fourier (QFT), se observa que, en el código de febrero de 2024, la QFT se implementa manualmente mediante una función definida en el propio código. Esta solución permite un mayor control sobre la implementación de QFT y la capacidad de ajustarla a diferentes tamaños de registro o requisitos de precisión.

En el código de agosto de 2023, QFT se aplica utilizando una clase estandarizada

proporcionada por la biblioteca *Qiskit*. De nuevo, está la facilidad y agilidad de trabajar con una solución definida a priori, sin embargo, esta opción puede limitar la capacidad de personalización del QFT, especialmente en situaciones en las que existe la necesidad de una implementación específica.

En cuanto a la personalización y optimización del código, se observa que las características que ofrece la implementación manual del algoritmo de *Shor* y QFT en el código de febrero de 2024, permiten una mayor personalización y optimización para el problema específico en cuestión. Por ejemplo, puede ajustar los parámetros del algoritmo de *Shor*, como el número de intentos para encontrar factores primos o la precisión del QFT, para mejorar el rendimiento o la eficiencia del algoritmo.

En el código de agosto de 2023, las opciones de personalización pueden ser más limitadas debido a la encapsulación del algoritmo de *Shor* y QFT en clases proporcionadas por la biblioteca *Qiskit*. El uso de estas características estandarizadas puede dificultar o imposibilitar la realización de ajustes específicos para mejorar el rendimiento o la eficiencia del algoritmo.

5. CONSIDERACIONES FINALES

Los resultados mostraron que, en ambas situaciones, hubo llamadas del algoritmo de *Shor* en los códigos propuestos por ChatGPT, sin embargo, parece haber habido un aprendizaje por parte de la plataforma en el período de 6 meses entre solicitudes. Este aprendizaje parece haber mejorado la capacidad del código para hacer frente a situaciones específicas que factoriza, es decir, en la primera propuesta (agosto de 2023) hay un uso masivo de recursos estandarizados, ya puestos a disposición de los usuarios del lenguaje *Qiskit*, sin embargo, en la propuesta de febrero de 2023, se observa la implementación "manual" de la mayoría de estos recursos.

La opción por la implementación "manual" de los recursos es positiva si se considera que la estructura de los algoritmos RSA no es estándar en su totalidad, por lo tanto, para cada algoritmo RSA, existe una necesidad específica de factorización de los números primos, dado el nivel de complejidad de la técnica utilizada en el cifrado. Cuando se utiliza un código con muchos elementos estandarizados, como ocurrió en el código propuesto en agosto de 2023, se pueden encontrar problemas que no se pueden descomponer con este tipo de herramientas. Por otro lado, al optar por una solución personalizada, como se observa en el código de febrero de 2024, la posibilidad de no romper el cifrado es menor, ya que el código es ajustable a las especificaciones del problema en cuestión.

Lo que resulta es que la herramienta ChatGPT aparentemente ha "aprendido" que el algoritmo de ChatGPT *Shor* se implementa mejor cuando se dejan de lado las herramientas de lenguaje estándar *Qiskit* y si utiliza códigos personalizados y ajustables. Este aprendizaje, desde el punto de vista de la inteligencia artificial, es muy positivo, ya que las numerosas iteraciones han permitido una mejora significativa en la construcción del algoritmo, por otro lado, considerando

aspectos de ciberseguridad, existe un mayor riesgo, ya que la implementación de este código optimizado en una plataforma de acceso remoto a ordenadores cuánticos, podría abrir posibles brechas en los sistemas de encriptación que, En última instancia, podría comprometer operaciones sensibles como la cadena de bloques y las transacciones financieras tradicionales, por lo que existe una necesidad latente de mejorar los sistemas de cifrado que se utilizan en la actualidad.

Finalmente, se considera que la creciente dependencia de los sistemas logísticos en el intercambio de información sensible requiere la implementación de medidas preventivas para asegurar la integridad de los datos y la información. Las posibilidades por explorar incluyen la adopción de criptografía robusta, políticas de seguridad estrictas e inversión en tecnologías emergentes, como *blockchain* e inteligencia artificial, con el fin de fortalecer la seguridad y la resiliencia de las cadenas de suministro.

REFERENCIAS

CASTRO, C. C. **Criptografia RSA**. Trabalho de Conclusão de Curso - Universidade Federal de Santa Catarina, Departamento de Matemática (MAT), Centro de Blumenau, Curso de Licenciatura em Matemática, Blumenau, 2019.

JOHNSTON, E.R.; HARIGAN, N.; SEGOVIA, M.G. **Programming quantum computers**. O'Reilly Media, Inc., Sebastopol, 2029.

PARADA, I.U. **Aplicação de processamento quântico para aceleração da resolução de autômatos probabilísticos**. Trabalho de conclusão de curso (Bacharelado em Engenharia Mecatrônica) — Universidade de Brasília, Brasília, 2019.

PORTUGAL, R.; MARQUEZINO, F. **Introdução à Programação de Computadores Quânticos**. Sociedade Brasileira de Computação, 2019.

RAMESH, K.; RAVISHANKARAN, S.; JOSHI, A.; CHANDRASEKARAN, K. **A survey of design techniques for conversational agents**. International Conference on Information, Communication and Computing Technology, Springer, Singapore, 2017.

SHOR, P.W. **Polynomial-Time Algorithms for Prime Factorization and Discrete Logarithms on a Quantum Computer**. 35th Annual Symposium Foundations of Computer Science, Santa Fe, NM, Nov. 20–22, 1994, IEEE Computer Society Press, pp. 124–134, 1994.

TONELI, D.A. **Desenvolvimento de um tutorial para o ensino de computação quântica**. Trabalho de Conclusão de Curso – Unifesp, 2022.

TEIXEIRA, M.A.F. **Números inteiros e criptografia RSA**. Dissertação (mestrado) - Universidade Estadual Paulista (Unesp), Instituto de Geociências e Ciências Exatas, Rio Claro, 2020.

UZEDA, E. E. F.; DALLASEN, R. V.; SANTOS, M. C.; ROLLWAGEN, A. F.; RIBEIRO, D. D.; FIGUEIREDO, J. A. O. **Utilização do algoritmo de Shor para quebra de criptografia RSA em computadores quânticos**. Salão do Conhecimento, Unijuí, Ijuí, 2022.

VIEIRA, L.A.; ALBUQUERQUE, C.D. **Um estudo passo a passo do Algoritmo de Shor.**
Proceeding Series of the Brazilian Society of Computational and Applied Mathematics,
v. 7, n. 1, 2020.

WESTE, N.H.E., HARRIS, D. M. **CMOS VLSI Design A Circuits and Systems Perspective.**
Addison – Wesley, New York, 2009.

"Los contenidos expresados en la obra, así como los derechos de autor de las figuras y datos, así como su revisión ortográfica y las normas ABNT son de exclusiva responsabilidad del autor o autores."