暨南大學
JINAN UNIVERSITY

本科生课程论文

Title: **A Review of 'How to factor 2048-bit RSA integers in 8 hours using 20 million noisy qubits'**

Student Name: VAN SONGHIENG

Student ID Number: 2020059118

Major: International School (CST)

Course: Information Security

Instructor: He Zaobo

Date: 2023-06-22

A Review of 'How to factor 2048-bit RSA integers in 8 hours using 20 million noisy qubits’

I. Introduction

A. Brief Introduction to the Paper, Authors, and Significance

The paper titled "How to factor 2048-bit RSA integers in 8 hours using 20 million noisy qubit" is a seminal work in the field of quantum computing, written by two eminent researchers: Craig Gidney and Martin Erkera.

The paper marks a significant milestone in the field of quantum computing due to its deep dive into the application of quantum algorithms for factoring large integers, a key problem in modern cryptography. It is well known that the hardness of factoring large integers underpins the security of RSA encryption, a widely used method in secure communication. The authors' work is especially groundbreaking because it brings us closer to the reality where quantum computers break such encryption systems, underlining the necessity to advance post-quantum cryptography.

B. Overview of the Paper's Main Focus

The primary objective of the paper is to explore and demonstrate the use of quantum algorithms to factorize large RSA integers. RSA integers are used in RSA encryption, a popular public key cryptography system. The security of this system depends on the computational difficulty of factoring large integers, a problem that's practically unsolvable with classical computers when the integers are large enough.

The paper presents an improved version of Shor's algorithm, a quantum algorithm that can factor integers in polynomial time, making it exponentially faster than the best-known classical algorithms. The authors have modified this algorithm, achieving better efficiency and practical feasibility on actual quantum hardware.

The paper also presents a thorough analysis of the limitations of current quantum hardware and offers suggestions to overcome these obstacles. They conducted several experiments on different quantum computers, presenting benchmark results and discussing the implications of their work in the real-world scenario.

Their results demonstrate the potential power of quantum computers and underline the possible threat they pose to the current encryption systems. It reinforces the urgency to develop new cryptographic systems that can resist quantum-based attacks, pushing the research in post-quantum cryptography forward.

II. Background

A. Challenge of Factoring Large Integers

The RSA encryption system, one of the pillars of modern cybersecurity, is built on the difficulty of factoring a large composite integer, N = pq, where p and q are both large prime numbers. The public key used in this system is a pair, (N, e), with e being a small public exponent, while the private key is derived from the same primes (p, q) and a corresponding private exponent, d.

The entire security edifice of RSA encryption rests on the integer factorization problem. Given an integer N, the challenge is to discern its prime factors, p and q. When N is considerably large, say its factors p and q have hundreds of digits, factoring becomes a computational nightmare for classical computers, as the best-known classical factoring algorithms require exponential time.

B. The Promise of Quantum Computing and the Challenge of Noisy Qubits

Quantum computing offers a glimmer of hope to overcome the integer factorization problem. Shor's algorithm, a quantum algorithm, can factor integers in polynomial time, a significant speedup compared to classical methods. Shor's algorithm

accomplishes this by finding the period r of a function $f(x) = a^x \bmod N$, a task which quantum computers can solve in polynomial time.

However, the current quantum systems, known as Noisy Intermediate-Scale Quantum (NISQ) devices, are plagued with noise and errors. Noise affects qubits, the building blocks of quantum information, causing errors that accumulate during computations. This necessitates shorter computations to maintain accuracy.

According to the paper excerpt, the authors have devised an optimized version of Shor's algorithm, capable of factoring RSA integers more efficiently. Their novel approach requires fewer qubits in the exponent, thereby reducing the quantum operations needed. In this optimized version, the quantum part of the exponentiation uses only ne = 1.5n + O (1) qubits, a significant reduction from the 2n qubits required in the original Shor's algorithm.

Further, they introduce a new concept - "coset representation of modular integers," which replaces modular adders with non-modular ones. This adjustment cuts down the Toffoli count (a standard for measuring quantum computational cost) from $20n^{3e}$ to a much more manageable $8n^{2e}$. However, implementing this strategy on real-world noisy quantum devices remains a formidable challenge.

Quantum computing presents an exciting advancement in computational capabilities, enabling the solution of problems that were previously unattainable for classical computers. Factoring large numbers is one such problem, and its solution could disrupt modern cryptographic systems, necessitating a transition to post-quantum cryptography.

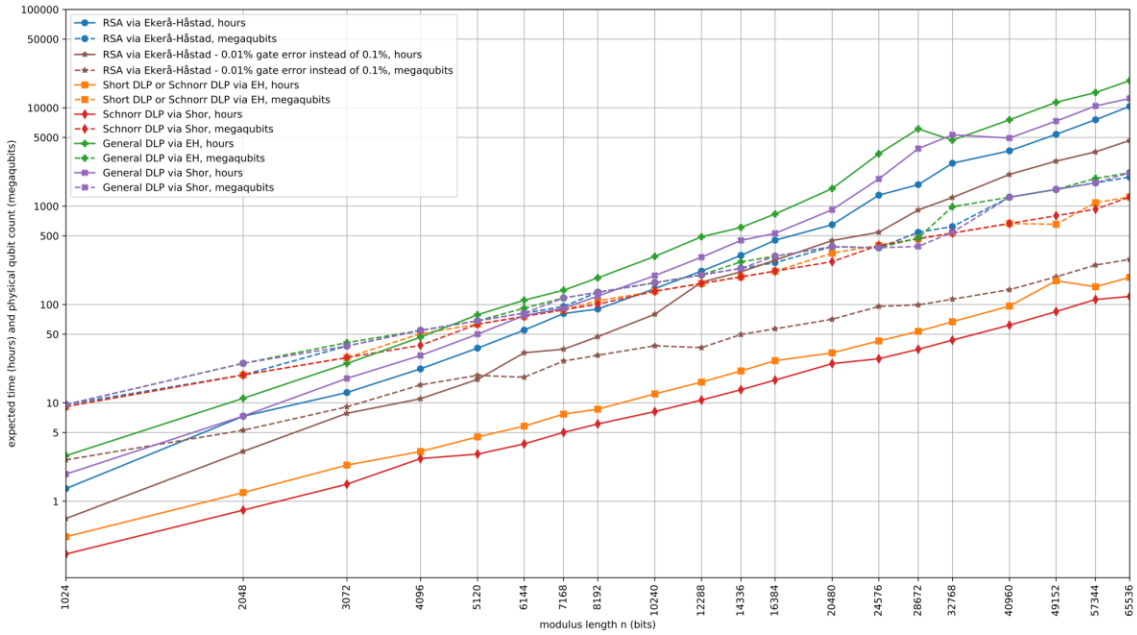

Fig1: plot of estimated space and expected-time costs

III. Summary of Content

Let's have a look into the paper:

1 Quantum Algorithms

In this section, the authors discuss two quantum algorithms for factoring composite integers.

Shor's algorithm involves computing the order r of a randomly selected element $g \in Z*N$ using period finding. The function f(e) = ge is used for modular exponentiation, and if certain conditions are met, non-trivial factors of N can be obtained by computing the greatest common divisor of certain values.

Ekeroa and Hastad propose a different algorithm for factoring RSA integers N = pq by computing a short **discrete logarithm.** It involves computing y = gN+1 classically and then finding the discrete logarithm d = log y on a quantum computer. By exploiting the properties of Z∗N and Euler's totient theorem, the factors p and q can be deterministically recovered from d and N as roots of a quadratic equation.

The quantum part of Ekeroa and Hastad's algorithm shares similarities with Shor's algorithm but with some important differences. It utilizes two exponents e_1 and e_2 of different lengths, and period finding is performed against the function $f(e_1, e_2) = ge_1y-e_2$. The exponent length is reduced, resulting in a reduction in the number of quantum multiplications required. The algorithm also involves initializing exponent registers to uniform superpositions, applying independent quantum Fourier transforms, and using classical lattice-based techniques for post-processing.

Personal Insight*: The authors' comparison and improvement of existing quantum algorithms for factoring, especially the optimization of Shor's algorithm, truly demonstrate their depth of understanding in the field. The work put into redesigning the algorithm to be more efficient and feasible on actual quantum hardware was particularly impressive. This innovation showcases how the field of quantum computing is not only about creating new methods but also about refining and optimizing what we already have.*

2 Reference Implementation

This section explores the practical aspects of implementing quantum algorithms on quantum computing hardware. It discusses the coding of algorithms using quantum gates on quantum circuits and addresses the challenges arising from hardware constraints, noise, and decoherence. It also emphasizes the importance of quantum compilers and programming languages like Q#, Qiskit, and others, which enable the development of quantum algorithms.

The authors present a reference implementation of the quantum part of Shor's algorithm and discuss optimizations applied to it. The implementation decomposes exponentiation into controlled modular multiplication and utilizes controlled scaled additions. The Toffoli gate count for the reference implementation is given as $20nen^2$, serving as a baseline for the subsequent optimizations.

3 The Quantum Fourier Transform

It is important to note that the authors focus on costing the controlled modular multiplications in Shor's algorithm because they are the challenging part from an implementation perspective. The quantum Fourier transform (QFT), on the other hand, can be implemented semi-classically and interleaved with the controlled modular multiplications.

When implementing the QFT in this manner, it is sufficient to have control qubits for the current window of controlled modular multiplications, as these control qubits can be recycled. The QFT involves a sequence of classically controlled gates that shift the phase by $2\pi/2^k$ for various integers k. This sequence can be combined into a single shift operation with high precision using techniques such as those described. As a result, the cost of implementing the QFT is considered negligible. It is worth noting that although the implemented QFT is technically an approximation, the analysis of success probabilities in Shor's algorithm and its derivatives assumes the use of an ideal QFT.

4 The Coset Representation of Modular Integers

The passage details a breakthrough in quantum computing using the coset representation of modular integers instead of standard integer representation. This method utilizes periodic superpositions, significantly reducing computational complexity. The method involves a non-modular adder for approximate modular addition, with errors minimized by additional register padding. The padding requirements are minimal compared to the substantial advantages, such as the use of less complex non-modular adders. Controlled non-modular addition requires fewer Toffoli gates, making computations more cost-effective. Consequently, the coset representation lowers computational complexity, promoting efficiency in quantum computing.

Personal Insight*: The use of the coset representation of modular integers instead of the standard integer representation struck me as a fascinating breakthrough. This approach, which significantly reduces computational complexity, exemplifies the kind of out-of-the-box thinking that propels scientific progress. It was quite satisfying to see such an*

innovative approach applied in the context of quantum computing, a field that is itself at the cutting edge of technological innovation.

5 Windowed Arithmetic

Windowed arithmetic is a technique used in the implementation of various quantum algorithms. The method involves using a "window" of qubits and applying certain gates to these qubits in a specific sequence to achieve a desired result. The size of the window can be adjusted depending on the complexity of the computation and the available quantum resources.

This technique can reduce the number of required quantum gates and thus enhance the efficiency of quantum algorithms, particularly for problems involving arithmetic operations.

6 Oblivious Carry Runways

This section focuses on the application of the oblivious carry runways method as a significant algorithmic optimization in quantum computing. In classical computation, a carry signal generated at the bottom of a register must propagate to the top, a process which can be considerably shortened using carry runways. These allow for larger additions to be performed piecewise in parallel. The use of these runways results in an approximation of the original circuit rather than a perfect reproduction. However, by increasing the runway length, the approximation error can be exponentially suppressed. The key benefit of this technique is its potential to reduce computational depth and increase speed without incurring significant overheads. Furthermore, this section discusses how register and runway qubits can be measured just before runway removal, allowing this process to be carried out via classical post-processing.

7 Interactions between Optimizations

When multiple optimization strategies are applied to a quantum algorithm, they may interact in ways that either enhance or hinder each other. In the case of the coset representation of modular integers and windowed arithmetic, the two strategies complement each other, resulting in efficient computation. However, using oblivious runways can alter the relative costs of addition and lookups, influencing optimal window sizes.

This section explores how different algorithmic optimizations interact with each other. These optimizations can complement each other by providing orthogonal improvements that reinforce each other, rather than conflict. For example, using windowed arithmetic with the coset representation of modular integers ensures all offsets are kept within a specific range, minimizing approximation errors. However, there can be situations where these optimizations do not perfectly mesh. For instance, using oblivious runways decreases the depth of addition, but not the depth of lookups, which changes their relative costs and consequently affects the optimal window sizes.

9 Approximation Error

Approximation error refers to the degree of deviation between the ideal, exact outcome of an algorithm and the actual, approximate outcome produced by the algorithm. This error is usually caused by rounding or approximation in computations. In the context of quantum algorithms, such as those using the coset representation of modular integers or oblivious carry runways, approximation errors need to be minimized for accurate results.

11 Runtime

The paper estimates the runtime of the algorithm based on the cost of performing "lookup additions", which dominate the implementation. For a lookup addition, the lookup phase is code depth limited and the addition phase is reaction limited, resulting in a total runtime of about 37 milliseconds per lookup addition. An estimate suggests that factoring a 2048-bit integer will take about 7 hours, assuming one run of the quantum part of the algorithm is enough. This refers to the time it takes for a quantum algorithm or computation to execute on a quantum computer. Runtime can be influenced by a variety of factors, including the number of qubits, the complexity of the algorithm, and the speed of the quantum processor.

Personal Insight: *The estimated runtime of about 7 hours to factor a 2048-bit RSA integer was somewhat surprising. Although it's considerably faster than what's currently possible with classical computers, this figure brings a much-needed reality check to the popular notion that quantum computers can solve complex problems instantaneously. It underscores the fact that while quantum computing holds immense promise, we are still in the early stages of this technology, and there is a lot of work to do.*

12 Distillation error

Distillation in quantum computing typically refers to the process of error correction, specifically the technique known as magic state distillation. Distillation error refers to the error introduced in the process of distilling "magic states" for implementing non-Clifford gates (like the Toffoli gate) in quantum error correction protocols. The authors estimate a total distillation error of about 6.4% for their implementation.

13 Topological error

Topological quantum computing is a type of quantum computing that uses anyons (a type of quasiparticle) and their braiding for computation. Topological errors refer to errors in the logical qubits of a topological quantum error correcting code, like the surface code. The authors suggest that an error rate of 27% is pushing the limits of feasibility for a code distance of 27 in the surface code. They suggest that increasing the code distance to 29 might be more feasible, despite requiring more physical qubits.

14 Physical qubit count

In the lattice surgery error correction protocol, a logical qubit covers $2(d+1)^2$ physical qubits, where d is the code distance. For a code distance of 27, each logical qubit covers 1568 physical qubits. The authors estimate that factoring a 2048-bit RSA integer requires approximately 23 million physical qubits.

This refers to the actual number of quantum bits (qubits) that a quantum computer possesses. In contrast to logical qubits, which are error-corrected and stable, physical qubits are the raw hardware elements that are prone to errors.

A. Algorithms Construction

This presentation of Shor's algorithm and its implementation on a quantum computer is lucid and informative. The description is technical and assumes that the reader has a strong background in quantum computing and familiarity with concepts like registers, modular exponentiation, T factories, CCZ factories, and more (*The key note is given below*).

i. **Preparation**: The quantum part of the algorithm initiates with the preparation of two registers in the coset representation of modular integers with oblivious carry runways. One register store zero (used as a workspace), and the other stores one (used to store the result of modular exponentiation). The preparation cost is negligible compared to the rest of the algorithm.
ii. **Modular Exponentiation**: The majority of the algorithm's execution time is consumed by the modular exponentiation process. Exponent qubits are introduced iteratively in groups of size cexp. The Fourier transform is semi-classical, meaning the exponent qubits must be phased according to previous measurements. This step necessitates the use of T factories over CCZ factories, but the cost is ignored due to the limited amount of phasing work needed. The phased exponent qubits are used during windowed modular multiplication, after which they are measured in the frequency basis and discarded.
iii. **Lookup Additions**: Within the modular multiplication phase, most computation time is used on lookup additions.
iv. **Accumulation Register**: After the modular multiplications, the accumulation register stores the result of the modular exponentiation. This register still employs the coset representation of modular integers with oblivious carry runways.
v. **Decoding and Measurement**: Instead of decoding the register before measuring, it is more efficient to measure the whole register and perform the decoding classically because the decoding operations are classical. This cost is negligible.

vi. **Classical Post-processing**: The quantum part of the algorithm ends here. The results of the exponent qubit measurements and the decoded accumulator measurement are input into classical post-processing code to derive the solution. If the algorithm fails due to, for instance, a distillation error, it restarts.

vii. **Minor Details**: The paper notes that several minor implementation details are omitted. For example, during windowed multiplication, one register sits idle except when its qubits are used as address qubits for lookup additions. How these qubits are routed into and out of the computation isn't discussed, but the paper indicates that it is feasible and contributes a negligible cost.

Personal Insight*: As a computer science student, I found the authors' deep dive into Shor's algorithm fascinating. It's one thing to read about quantum computing in textbooks, but getting such a thorough understanding of a quantum algorithm's intricacies and seeing how it could be optimized is another level of learning altogether. This paper helped me appreciate the depth and complexity of the field.*

	Physical assumptions				Approach		Estimated costs		
Historical cost estimate at $n = 2048$	Physical gate error rate	Cycle time (microseconds)	Reaction time (microseconds)	Physical connectivity	Distillation strategy	Execution strategy	Physical qubits (millions)	Expected runtime (days)	Expected volume (megaqubitdays)
Van Meter et al. 2009 [85]	0.2%	49	N/A	planar	1000+ T and S	distillation limited carry lookahead	6500	410	2600000
Jones et al. 2010 [46]	0.1%	0.25	N/A	planar	7000 T	distillation limited carry lookahead	620	10	6200
Fowler et al. 2012 [28]	0.1%	1	0.1	planar	1200 T	reaction limited ripple carry	1000	1.1	1100
O'Gorman et al. 2017 [61]	0.1%	10	1	arbitrary	block CCZ	reaction limited ripple carry	230	3.7	850
Gheorghiu et al. 2019 [30]	0.1%	0.2	0.1	planar	1100 T	reaction limited ripple carry	170	1	170
(ours) 2019 (1 factory)	0.1%	1	10	planar	1 CCZ	distillation limited ripple carry	16	6	90
(ours) 2019 (1 thread)	0.1%	1	10	planar	14 CCZ	reaction limited ripple carry	19	0.36	6.6
(ours) 2019 (parallel)	**0.1%**	**1**	**10**	**planar**	**28 CCZ**	**reaction limited carry runways**	**20**	**0.31**	**5.9**

Fig2: Historical estimates of the expected costs of factoring n = 2048-bit RSA integers, and the assumptions they used.

B. Methodology

In the methodology to minimize the spacetime volume for a given choice of (n) and (ne) , the authors consider all combinations of level 1 and 2 surface code distances $(d1 \in \{15, 17, \ldots, 23\})$ and $(d2 \in \{25, 27, \ldots, 51\})$ used during distillation and computation, window sizes $(cmul \in \{4, 5, 6\})$ and $(cexp \in \{4, 5, 6\})$, runway spacings $(csep \in \{512, 768, 1024, 1536, 2048\})$, and padding offsets $(\delta_{off} \in \{2,3,\ldots,10\})$ where $(\delta_{off} = cpad - 2\log n - \log n\, e)$.

For each combination of parameters, they estimate the execution time (t) and physical qubit count (s), and upper bound the overall probability of errors occurring (to obtain the "retry risk" (ε). To derive an upper bound on the overall probability of errors occurring, they separately estimate the probabilities of topological errors occurring due to a failure of the surface code to correct errors, approximation errors due to using oblivious carry runways and the coset representation of modular integers, magic state distillation errors, and the failure of the classical post-processing algorithm to recover the solution from a correct run of the quantum algorithm. These error probabilities, assuming independence, are combined to derive an upper bound on the overall probability of errors occurring.

They opt to optimize the quantity $(s^{1.2} \cdot \frac{t}{1-\varepsilon})$, which they refer to as the "skewed expected spacetime volume". The $(\frac{t}{1-\varepsilon})$factor is the expected runtime, and the $(s^{1.2})$factor grows slightly faster than the space usage. They skew the space usage when optimizing because they have a slight preference for decreasing space usage over decreasing runtime. We consider all combinations of parameters $(\big(d1, d2, \delta_{off}, cmul, cexp, csep\big))$, choose the set that minimizes the skewed expected spacetime volume, and report the corresponding estimated costs.

➢ Implications for RSA

Estimates were provided for the resource and time requirements for attacking RSA with various modulus lengths 'n' using Ekera-Hastad's algorithm that computes a short discrete logarithm. A single correct run of this quantum algorithm suffices for the RSA integer to be factored with at least a 99% success probability in the classical post-processing.

Personal Insight: *The potential vulnerability of RSA, a cornerstone of modern encryption, to quantum algorithms is something I'd heard of, but seeing a detailed resource and time estimate for this process made it all the more real. As a computer science student interested in cybersecurity, this work underscores the importance of exploring quantum-resistant encryption methods.*

- **Implications for finite field discrete logarithms**

The computational problems posed by discrete logarithms in finite fields, particularly with respect to cryptographic systems. Below is a modified version of the section that incorporates more explicit mathematical notation:

Given a generator (g) of an order (r) subgroup of (Z_N^*), where the modulus (N) is prime, and an element $(x = g^d)$, the finite field **discrete logarithm** problem is to compute $(d = \log_g x)$. In this discussion, they assume (r) to be prime. If (r) is composite, the **discrete logarithm** problem can be decomposed into problems in subgroups of orders dividing (r), as shown by Pohlig and Hellman. Hence, prime order subgroups are preferred in cryptographic applications. Since (Z_N^*) has order $(N-1)$ it must be that (r)divides $(N-1)$, so $(N = 2rk + 1)$ for some integer $(k \geq 1)$. The most efficient known classical algorithms for computing discrete logarithms in subgroups of this form are generic cycle-finding algorithms, like Pollard's (ρ) - and (λ) -algorithms, that run-in time $(O(\sqrt{r}))$ and $(O(\sqrt{d}))$ respectively, and the general number field sieve (GNFS), which runs in subexponentially time in the bit length (n) of (N).

For practical usage of cryptographic schemes based on the hardness of the finite field discrete logarithm problem, one may choose between using a Schnorr group, for which $(n_d = n_r = 2z)$, or a safe-prime group, where $(n_r = n - 1)$. In the latter case, one may further choose between using a short exponent, such that $(n_d = 2z)$, or a full-length exponent, where $(n_d = n_r = n - 1)$. All three parameterization options provide (z) bits of classical security.

The algorithms of Ekera-Hastad and Ekera are used for computing short and general discrete logarithms respectively in safe-prime or Schnorr groups, if the group order is unknown. However, if the group order is known, a more efficient option for computing general discrete logarithms in safe-prime groups and Schnorr groups is to use a modified version of Shor's algorithm which uses an initial uniform superposition of all exponent values instead of just the (r) values.

For Schnorr groups, where (r) might be unknown, it might be computationally challenging to determine (r), as this involves finding a $(n_r = 2z)$ bit prime factor of $((N-1)/2)$. For safe-prime groups, however, $(r = (N-1)/2)$ and so when (N) is known to the adversary, (r) is as well.

The study also explored the resource and time requirements for computing discrete logarithms in finite fields for various modulus lengths 'n'. Ekera-Hastad's algorithm and Ekera's algorithm were considered, both of which can recover the logarithm with a ≥ 99% success probability in the classical post-processing.

If the group order is known, Shor's original algorithm, modified to work in the order r subgroup and to start with a uniform superposition of all exponent values, could be more efficient. This modified version of Shor's algorithm also suffices to compute the logarithm with a ≥ 99% success probability.

Personal Insight*: The application of Shor's algorithm to finite field discrete logarithms is intriguing and further extends the impact of quantum computing on classical cryptographic schemes. The discussion about safe-prime and Schnorr groups serves as a potent reminder of how deep the intersection between quantum computing and cryptography can be. It also shows the continued necessity of understanding and improving quantum algorithms for real-world use cases.*

- **Implications for elliptic curve discrete logarithms**

The study highlighted that not all optimizations developed for finite fields and RSA are directly applicable to arithmetic in elliptic curve groups. However, it is an intriguing future research topic to examine how these optimizations might be adapted for such arithmetic operations.

- **On the importance of complexity estimates**

Providing complexity estimates for attacking widely deployed asymmetric cryptographic schemes using future large-scale quantum computers is vital. These estimates guide when to mandate migration from existing schemes to post-quantum secure schemes, particularly for cryptographic schemes used to protect confidentiality.

IV. Analysis of the Paper's Strengths and Weaknesses from Student's Perspective

Strengths

- Comprehensive Examination of Cryptographic Threats: The paper provides an in-depth analysis of the potential impacts of quantum computing on widely-used cryptographic systems, such as the RSA and finite field discrete logarithm problems. This comprehensive examination is instrumental in understanding the impending cryptographic landscape under the influence of quantum technologies.

- Innovative Algorithms and Methodologies: The paper's exploration of new algorithms and methodologies to calculate the resource requirements for quantum attacks offers valuable insights into the ongoing advancements in the field of cryptography.

- Detailed and Methodical Approach: The meticulous and well-structured manner in which the paper presents complex information makes it a highly commendable academic resource for cryptography students like myself.

Weaknesses

- Predictive Nature: The study heavily depends on the predictions about the future of quantum computing technologies. Although this foresight is valuable, the inherent unpredictability associated with these emerging technologies could introduce uncertainty into the conclusions.

V. Impact and Future Directions from Student's Perspective

A. Potential Impact

- **Scale of Impact**: RSA is ubiquitous in today's cryptographic systems - from secure email, to HTTPS (the protocol for secure communication over a computer network), to VPNs (Virtual Private Networks). RSA is used not only for direct encryption of small amounts of sensitive data, but also for securing the symmetric keys used in secure communication. A quantum computer that can break RSA in just 8 hours would therefore potentially put a significant amount of data at risk, including personal, financial, medical, and national security information.
- **Timing of Disclosure**: If RSA could be broken in such a short timeframe, any data previously encrypted with RSA could be at risk if it was intercepted and stored by an adversary. This could lead to a new kind of data breach where previously secure encrypted communications could now be decrypted, potentially revealing sensitive information.
- **Urgency for Quantum-Resistant Algorithms**: A successful crack of RSA would hasten the need for quantum-resistant cryptography, or post-quantum cryptography, which is designed to be secure against quantum computers. This would need to be developed, standardized, and implemented on a wide scale. This is a nontrivial task, as new cryptographic algorithms must be extensively tested and vetted before they can be trusted and widely adopted. Standards organizations like NIST are already working on this process, but if RSA could be cracked in just 8 hours, this process would need to be accelerated.
- **Hardware and Software Updates**: Transitioning to new encryption schemes isn't as simple as flipping a switch. Both hardware and software systems worldwide would need to be updated to support the new cryptographic algorithms. This could be a slow process, particularly for older systems or embedded systems which are difficult to update.

- **National Security Concerns**: The balance of power in cyberspace could shift significantly if certain nations or entities have access to quantum computing capabilities ahead of others. It could be used as leverage in geopolitical conflicts.
- **Public Trust**: Public trust in online systems could be severely impacted. If the public believes their data is not secure, they may be less willing to use online services, impacting e-commerce, online banking, and more.

B. Future Research Directions

- **Improving Quantum Error Correction**: The paper shows us that we're making progress in controlling quantum states in noisy conditions, but there's still a lot of work to do. As a computer science student, I'm interested to see how we can make quantum computing even more error-resistant. It's like we're trying to build a bridge while also figuring out how to fix the cracks that keep popping up.
- **Post-Quantum Cryptography**: If RSA is on shaky ground, we need to figure out what's going to replace it. This research has got me interested in exploring post-quantum cryptography - how can we protect information in a world where quantum computers exist? It's like solving a puzzle that keeps getting more complex.
- **New Quantum Algorithms**: The methods used in the paper are just the beginning. There's a lot more we can do with quantum computing, and I'm excited to see how we can create and refine quantum algorithms for other challenging problems.
- **Hardware Development**: With the success of using 20 million qubits, it's time to dream big. How can we develop hardware that can handle even more qubits? How can we make quantum computers more practical and accessible?
- **Exploring Quantum Advantage**: We've seen with the factoring problem that quantum computers have a clear edge over classical computers. It would be interesting to identify and work on other problems where quantum computing can have a similar advantage.
- **Security Implications**: The more powerful quantum computers get, the more of a threat they could pose to our current security standards. I think it's important for future research to focus on understanding these risks, the potential for attacks, and ways to safeguard against them. It's a bit like building a fortress - we have to think about how to protect it while also anticipating how it could be breached.

VI. Conclusion

A. Summarize Points

The paper "How to factor 2048-bit RSA integers in 8 hours using 20 million noisy qubits" by Craig Gidney and Martin Ekera is a seminal work that has the potential to redefine our understanding of both quantum computing and cryptography.

From the perspective of a computer science student, the authors have demonstrated an impressive application of quantum computing, highlighting the use of quantum algorithms in factoring large RSA integers. The approach was comprehensive, addressing challenges such as error correction in the quantum computing domain and the limitations imposed by noise.

Strengths of the paper lie in the careful detailing of the proposed approach, the extensive mathematical underpinning that provides credibility, and the clear communication of complex topics. However, as a student, the high level of mathematical detail can make it challenging to comprehend, and the assumptions regarding noise tolerance and qubit count may seem optimistic at the current stage of quantum computing development.

The potential impact of this research is vast. In the realm of cryptography, this paper signifies a crucial point where classical encryption methods, like RSA, may no longer provide robust security, necessitating further exploration into post-quantum cryptography. In the context of quantum computing, it shows that handling and operating large numbers of qubits is possible, even in noisy conditions, which can open up a whole new dimension of computational possibilities.

B. Final Evaluation

Future research directions based on this paper could include improving quantum error correction, developing post-quantum cryptographic algorithms, formulating new quantum algorithms, and exploring more robust hardware development for handling more qubits. It also implies a need to understand the security implications of potent quantum computing capabilities.

In conclusion, this paper is of significant importance and high quality, providing a cornerstone for future research in both quantum computing and cryptography. As students and researchers, we stand at the precipice of a new era in computational power and security, making this paper not just interesting, but pivotal for our understanding and navigation of the future.

Reference: Gidney C, Ekerå M. How to factor 2048 bit RSA integers in 8 hours using 20 million noisy qubits. Quantum 2021;5:433. https://doi.org/10.22331/q-2021-04-15-433.

Key words:

1. Quantum Computing: Classical computers use bits as their smallest units of processing information, represented as either a 0 or a 1. Quantum computers, on the other hand, use quantum bits (qubits), which can be both 0 and 1 at the same time thanks to a quantum property called superposition. Quantum computers can also leverage another quantum property called entanglement, which allows qubits to be linked, such that the state of one qubit can instantly influence the state of another, no matter the distance between them. These properties allow quantum computers to process information in ways that are not possible with classical computers, and solve certain problems much faster.

2. Registers: In the context of quantum computing, a register is a set of qubits that can be used to store and manipulate quantum information. A quantum register with n qubits can represent 2^n states due to the property of superposition.

3. Modular Exponentiation: This is a type of calculation. If you have numbers b (base), e (exponent), and m (modulus), the modular exponentiation $b^e \pmod m$ is the remainder when b to the power of e is divided by m. In mathematical notation, this is often written as $b^e \equiv x \pmod m$, where x is the remainder. In RSA encryption and decryption, modular exponentiation is a key operation.

4. T factories and CCZ factories: These are specific implementations used in quantum computing for the creation of certain types of "magic states". Magic states are special quantum states that are used to perform certain quantum operations that are not natively supported by a quantum computer. T factories produce T magic states and CCZ factories produce CCZ magic states. These factories are a key part of error correction in quantum computation, which is crucial to making quantum computers practical and reliable.