jet INTL JOURNAL OF ELECTRONICS AND TELECOMMUNICATIONS, 2025, VOL. 71, NO. 1, PP. 39–46
Manuscript received December 2, 2024; revised March 2025. doi: 10.24425/ijet.2025.153542

Maximizing the practical achievability of quantum annealing attacks on factorization-based cryptography

Olgierd Żołnierczyk

***Abstract*—This work focuses on quantum methods for cryptanalysis of schemes based on the integer factorization problem and the discrete logarithm problem. We demonstrate how to practically solve the largest instances of the factorization problem by improving an approach that combines quantum and classical computations, assuming the use of the best publicly available special-class quantum computer: the quantum annealer. We achieve new computational experiment results by solving the largest instance of the factorization problem ever announced as solved using quantum annealing, with a size of 29 bits. The core idea of the improved approach is to leverage known sub-exponential classical method to break the problem down into many smaller computations and perform the most critical ones on a quantum computer. This approach does not reduce the complexity class, but it assesses the pragmatic capabilities of an attacker. It also marks a step forward in the development of hybrid methods, which in practice may surpass classical methods in terms of efficiency sooner than purely quantum computations will.**

***Keywords*—Cryptography; Integer Factorization Problem; Number Field Sieve; Hybrid quantum computations; Quantum annealing; QUBO; B-smooth numbers**

I. Introduction

CRYPTOSCHEMES, which are at most as secure as the factorization problem (factorization-based), can be compromised in standard, general approaches: classically with subexponential complexity through decomposition base methods [1] and quantumly with polynomial complexity through Shor's algorithm [2] or via other quantum methods of unspecified complexity. Importantly, fully quantum solutions in practice require too many resources: the largest practically solved examples include: 6 bits - Shor's algorithm [3], 10 bits - adiabatic quantum computation [4], and 23 bits - quantum annealing [5]. Therefore, hybrid methods such as [4], [6]–[8] have been invented, representing a compromise between computational complexity and the use of quantum resources.

In this work, we present experimental results of the efficiency of hybrid classical-quantum computations using quantum annealing. This is a transitional approach between fully quantum methods and best methods for classical hardware. It involves factoring numbers based on a subexponential factorization method, where part of the computations is performed using quantum techniques, specifically through quantum annealing. This phase is crucial for the complexity of the method and focuses on searching for numbers with special properties, known as B-smooth numbers. As a result, we obtain an improvement that has the potential to practically address the largest instance, ever computed on quantum machine. This is due to using fewer quantum resources than fully quantum approaches.

As a result, a 29-bit problem was solved, setting a record among all approaches with quantum annealing. This result improves upon the 23-bit result of a fully quantum annealing method, and the most effective to date, a hybrid approach, which factored a 26-bit number [8]. The contribution of this work to improving this record lies in using a more advanced algebraically decomposition base method (General Number Field Sieve) for factorization compared to [8]. The quantum resource occupancy $O(\frac{\log^2 h}{4})$ remains the same for h - values from the relation gathering step, yet h are smaller in this work due to the refined algebraic sieving of the number field. Among all hybrid approaches, this also represents the largest result, excluding only [9], where the type of scalability is unspecified.

Due to the lack of theoretical foundations (regarding the complexity of quantum annealing), the aim of this paper is not to fully compare this approach with known ones, including classical subexponential solutions. Although the complexity of presented calculations remains in the subexponential class, in practice, hybrid approaches may surpass classical solutions sooner than fully quantum methods

A. Paper organization

The article is divided into two main parts: the established theory and the description of the results of this work – the experiment. The following sections explain:

- subsection II-A The method of the General Number Field Sieve and the role of finding B-smooth numbers within.
- subsection II-B What is quantum annealing.
- subsection II-C How quantum annealing can be used to find B-smooth numbers.
- section III The contribution of this work, how the experiment was conducted, and the detailed results obtained.

This work was Supported by the Grant DOB/002/RON/ID1/2018 by The National Centre for Research and Development.

O. Żołnierczyk is with Faculty of Cybernetics, Military University of Technology, Warsaw, Poland (e-mail: olgierd.zolnierczyk@wat.edu.pl).

II. Theory

A. Classical factorization method

In cryptology integer factorization goal is to factor $N = pq$, where p, q are primes. The most effective classical general method for integer factorization is the General Number Field Sieve (GNFS), initially proposed as a generalized version of previous ideas in [10]. There are many improvements of this method; however, from the perspective of this work, it is sufficient to focus on the primary version. It achieves subexponential cost

$$O\left(\exp\left((1.923 + o(1))\,(\ln N)^{1/3}(\ln\ln N)^{2/3}\right)\right),$$

under various heuristic assumptions.

1) Essence of GNFS: This tool is highly sophisticated and operationalises the fundamental idea of factorization: establishing relationships of congruences of squares, defined as follows

$$X^2 \equiv Y^2 \pmod N. \tag{1}$$

If we can indicate such a pair, we then have: $(X - Y)(X + Y) \equiv 0 \pmod N$, and if $X \not\equiv \pm Y \pmod N$, we can easily factorize N by computing $\gcd(X \pm Y, N)$. Establishing relation from Equation 1 leads to factorization with probability at least $\frac{1}{2}$. This is accomplished through the following stages.

Firstly, finding an appropriate polynomial $\mathcal{F}(x) \in \mathbb{Z}[x]$, and an m such that

$$\mathcal{F}(m) \equiv 0 \pmod N. \tag{2}$$

The simplest way to do this is by: setting $m = \lfloor N^{\frac{1}{d}} \rfloor$, where d is an integer parameter that regulates the degree of the polynomial $\mathcal{F}$, then expressing N in base m, that is: $N = m^d + y_{d-1}m^{d-1} + \cdots + y_1 m + y_0$, where each of $y_{d-1}, \ldots, y_1, y_0$ is between 0 and $m - 1$ (for small d we have $(N > 2m^d)$), and finally assigning $\mathcal{F} = x^d + y_{d-1}x^{d-1} + \cdots + y_1 x + y_0$. Hence, it follows Equation 2. We assume that F is irreducible, because otherwise we immediately obtain the factorization of N. Moreover, if we denote ρ as a root of $\mathcal{F}$, this polynomial defines a ring homomorphism (proof in [11], Lemma. 11.19):

$$\begin{aligned} &\phi : \mathbb{Z}[\rho] \to \mathbb{Z}/N\mathbb{Z}, \\ &\sum_i z_i \rho^i \to \sum_i z_i m^i \pmod N, \end{aligned} \tag{3}$$

and let us recall that a homomorphism, by definition, has the following defining property:

$$\forall_{\gamma_1\gamma_2 \in \mathbb{Z}[\rho]} \phi(\gamma_1\gamma_2) = \phi(\gamma_1)\,\phi(\gamma_2) \in \mathbb{Z}/N\mathbb{Z}. \tag{4}$$

Secondly, identifying a set $\mathbb{S}$ of pairs (a, b) such that combined: $\prod_{(a,b)\in\mathbb{S}} (a + b\rho)$ is a square $\gamma^2 \in \mathbb{Z}[\rho]$ and $\prod_{(a,b)\in S} (a+bm)$ is a square $X^2 \in \mathbb{Z}$.

Third, by computing $\gamma \in \mathbb{Z}[\rho]$ as square root of γ^2. At this point, the factorization is almost complete because the following holds:

$$\begin{aligned} X^2 = \prod_{(a,b)\in\mathbb{S}} (a + bm) &\overset{\text{by Equation 3}}{\underset{(\text{mod } N)}{\equiv}} \phi\left(Y^2\right) \\ &\overset{\text{by Equation 4}}{=} \phi(\gamma)\phi(\gamma) \text{ mod N} \\ &= Y^2 \text{ mod N}, \end{aligned} \tag{5}$$

for some $Y \in \mathbb{Z}/N\mathbb{Z}$ (the symbol $\underset{(\text{mod } N)}{\equiv}$ here denotes congruence modulo N). Hence, we have relation from Equation 1 and can easily obtain X by taking the square root of X^2 in $\mathbb{Z}$ (and then reducing modulo N), and Y by projecting $\phi(\gamma)$. Equally important, when computing square roots in these two rings, there is no map that guarantees $X \equiv \pm Y \pmod N$, and in practice, this is often not the case. Then, we satisfy the conditions to determine a non-trivial divisor of N by computing $\gcd(X - Y, N)$ or $\gcd(X + Y, N)$. This is the outline of the entire method.

All three phases contain more solutions that address the challenges associated with each of them. However, in particular, the most important stage for this work and for the complexity of the entire method is the stage of determining the relation of congruent squares.

2) Linear algebra stage: The forms γ^2 and X^2 (being products of many simple expressions) are not insignificant here. This arises, among other things, from the fact that instead of checking each randomly selected pair belonging to $\mathbb{Z} \times \mathbb{Z}[\rho]$ to see whether it forms a pair of squares (a naive approach), we determine the desired square values by multiplying the appropriate combination of previously recorded simple expressions, represented by (a, b). This multiplication is understood as the multiplication of the expressions of the form $a + bm$ and $a + \rho m$ in their respective two rings. Thus, we treat this specific set of pairs (a, b) as a single representation of both elements from the two rings, as we need to obtain squares in both rings simultaneously.

If we succeed in identifying a sufficient number of pairs (a, b) that imply only those elements that can be expressed as a unique factorization into prime elements (this applies to both integers and the ring $\mathbb{Z}[\rho]$, in which we factor the element γ^2), we reduce the problem of obtaining congruent squares to solving a system of linear equations over the field $\mathbb{F}_2$. The assumption of unique factorization is unrealistic in $\mathbb{Z}[\rho]$, but it is close enough to practice to clearly present the main idea of this phase; hence, we leave the realistic refinements for later.

To illustrate this more clearly, let us assume we represent a fixed pair (a_1, b_1). We record the factorization in $\mathbb{Z}$: $a_1 + b_1 m = s_1^{e_1} s_2^{e_2} s_3^{e_3} \ldots s_k^{e_k}$, and the factorization in the ring of algebraic integers of the number field $\mathbb{Q}(\rho)$: $a_1 + b_1\rho = \mathfrak{p}_1^{v_1}, \mathfrak{p}_2^{v_2}, \mathfrak{p}_3^{v_3} \ldots \mathfrak{p}_t^{v_t}$. Now, the sequence $e_1 e_2 e_3 \ldots e_k v_1 v_2 v_3 v_t$ is a vector representing the pair (a_1, b_1), which we shall call $\vec{V}_1$. Furthermore, the assumption of unique factorization still applies. We know that, in such a case, the parity of all exponents in any vector $\vec{V}$ is equivalent to the fact that the numbers it represents are squares. Having many different vectors $\vec{V}$, representing the recorded pairs (a, b), we must determine their combination

such that the sum of the coefficients of these vectors for each dimension is even. This implies linear vector calculations over $\mathbb{F}_2$:

$$\mathrm{x}_1 V_1 + \mathrm{x}_2 V_2 + \ldots \mathrm{x}_l V_l = \vec{0}. \tag{6}$$

The vector $\mathrm{x}_1, \mathrm{x}_2, \ldots, \mathrm{x}_l$, where $\mathrm{x}_i \in \mathbb{F}_2$, encodes the solution to the system of linear equations defined by the matrix formed from the vectors $\vec{V}$, thereby indicating which pairs (a, b) should be multiplied together so that the result is a square in both rings.

3) Factorization of algebraic elements: In practice, to establish the vector representation of the screened pairs (a, b), we must determine a finite set of elements s and $\mathfrak{p}$ necessary for this, which we call the factor base $\mathbb{B}$. For integers, we rely on the unique factorization of integers and use the set of prime numbers smaller than a given bound B, which gives rise to the term B-smooth number, meaning

Definition 1. *A B-smooth number is one whose prime divisors are all less than or equal to B.*

In the second structure, we do not have unique factorization; instead, we use the unique factorization into so-called prime ideals in the ring of integers of the number field $\mathbb{Q}(\rho)$ (in this case, the smoothness of the screened algebraic elements also depends on the bound B, as described more thoroughly below).

All the concepts and facts from number field theory necessary for a complete understanding of the factorization of algebraic elements (from the ring $\mathbb{Z}[\rho]$) can be found in [11]. From the perspective of this work, two issues are important: firstly, how factorization into prime ideals relates to detecting B-smoothness, and secondly, what problems it poses and what practical steps are involved, ultimately aimed in this phase at determining the pair γ^2, X^2. The use of factorization into prime ideals makes verifying the smoothness of an algebraic element (i.e., confirming that the element factors into prime ideals from the factor base) more complex and requires a two-step process.

This is because:

Theorem 1. *The factorization into prime ideals of the ideal generated by $(a + b\rho)$, assuming that $\gcd(a, b) = 1$, contains only ideals generated by the pair: $(s, \rho - r)$, where:*

- *s is a prime number such that $\mathcal{F}$ has roots modulo s,*
- *r satisfies: $\mathcal{F}(r) \equiv 0 \pmod{s}$.*

(The proof of this theorem can be derived based on the proof in [11], Theorem 11.12). Therefore, the prime ideals from the factor base can be represented by pairs (L, r). Hence, the factor base is narrowed down to those prime ideals generated by primes from the factor base $\mathbb{B}$ that can occur in the factorization of $a + b\rho$, according to the theorem. Thus, the first and most important step in verifying the smoothness of an algebraic element is identifying all prime numbers that generate the prime ideals of that element. This is made possible by using the norm of the number field element, which intuitively has the following meaning: it generalizes the absolute value of a real number and serves as a measure of a certain type of magnitude of a given element. It is defined as follows:

Definition 2. *The norm of an element α of the number field $\mathbb{Q}(\rho)$, denoted $N_{\mathbb{Q}(\rho)}(\alpha)$, is:*

$$N_{\mathbb{Q}(\rho)}(\alpha) = \prod_i \sigma_i(\alpha), \tag{7}$$

where σ_i is the i-th field homomorphism, defined as:

$$\begin{aligned} &\sigma_i : \mathbb{Q}(\rho) \to \mathbb{Q}(\rho_i), \\ &\sum_j \mu_j \rho^j \to \sum_j \mu_j \rho_i^j \pmod{N}, \end{aligned} \tag{8}$$

and ρ_i is the i-th root of the minimal polynomial of ρ in the field $\mathbb{C}$.

Additionally, it turns out that:

Theorem 2. *If α belongs to the ring of integers of the number field $\mathbb{Q}(\rho)$, then $N_{\mathbb{Q}(\rho)}(\alpha) \in \mathbb{Z}$.*

(The proof of this theorem can be found in [11], Corollary 3.17), therefore in this case, we can discuss the prime factorization of the norm, especially since we can compute it efficiently, as:

$$\begin{aligned} N_{\mathbb{Q}(\rho)}(a + b\rho) &= (a + b\rho)(a + b\rho_2) \cdots (a + b\rho_d) \\ &= (-b)^d \left(\frac{a}{b} - \rho\right)\left(\frac{a}{b} - \rho_2\right) \cdots \left(\frac{a}{b} - \rho_d\right) \\ &= (-b)^d \mathcal{F}\left(\frac{a}{b}\right). \end{aligned}$$

Adding to this the even more significant fact that:

Theorem 3. *A prime number s divides $N_{\mathbb{Q}(\rho)}(\alpha)$ if and only if the ideal generated by α includes in its factorization a prime ideal generated by the pair (s, r).*

(The proof of this theorem can be found in [12], Preposition 5.3.), we can conclude that the determination of all prime ideals into which an algebraic element factors is performed in the first step by factoring the norm of elements of the form $a + b\rho$, where $\gcd(a, b) = 1$, into prime numbers. To fully represent these ideals, a second step is required — the determination of the corresponding r, which is done via simple linear modular calculations. For clarity, we will omit these calculations and refer to a more comprehensive description, along with other details of the GNFS method, in [13].

Even after determining the exponents, i.e., the vectors $\vec{V}$, the situation is not as straightforward as under the idealised assumption of unique factorization in $\mathbb{Z}[\rho]$. Factorization into ideals introduces two additional potential points of failure related to representing the element by an ideal and vice versa, as well as two more points of failure associated with the fact that the ideals are defined not in $\mathbb{Z}[\rho]$ but in the ring of integers of the number field $\mathbb{Q}(\rho)$. In practice, we deal with these issues by determining additional properties for each pair (a, b), in the form of extra columns in the matrix M (referred to as Adleman columns). These do not guarantee the success of identifying the correct pair of squares, but they significantly increase the probability of success to a sufficiently high level.

4) Closing notes: The remaining phases of the GNFS method, namely the selection of the polynomial $\mathcal{F}$ and the determination of the square root of γ^2, can be approached using various methods, which have been developed and refined over time by researchers. Further details on these phases, as well as improvements in the sieving process, can be found in works such as [14]–[16].

Crucially for this work, the factorization problem is effectively broken down into subproblems involving determining whether a given element is divisible by prime factors greater than a certain limit B (there is also the cost of additional computations, which does not increase the complexity of the entire method.). We say that we are checking the B-smoothness of a given number. Thus, identifying a sufficient amount of B-smooth numbers among those sieved provides a high probability of factorizing the N.

B. Quantum annealing and the QUBO problem

The most general concept of quantum annealing, introduced in [17], is that due to the quantum phenomena involved, the computations performed by a quantum annealer rely on solving a certain type of optimization problems, defined below. In this process, the minimum of a given objective function $\mathcal{K}$ is found with some probability, where the preimage of the function is represented by the state of the quantum computer's memory (the quantum annealer). From a physical perspective, the computations involve reaching the system's lowest energy level (ground state), which is achievable through quantum phenomena such as superposition, tunneling, and energy dissipation. This is a computational technology derived from the paradigm of adiabatic computing [18], [19].

In practice, a quantum annealing computer resolves computational tasks defined by the Ising problem, which can equivalently be transformed (through a linear transformation) into the form of Quadratic Unconstrained Binary Optimization (QUBO). QUBO is a second-degree, multivariate polynomial with real coefficients and binary variables:

$$\mathcal{Q}(\mathrm{u}_1, \ldots, \mathrm{u}_\mathrm{o}) = \sum_i \beta_i \mathrm{u}_\mathrm{i} + \sum_{i<j} \delta_{i,j} \mathrm{u}_\mathrm{i}\mathrm{u}_\mathrm{j}. \tag{9}$$

Transforming any problem into the QUBO allows for attempts to solve it via quantum annealing.

It has already been shown how to transform into the QUBO form: the discrete logarithm problem over a finite field [20], as well as the discrete logarithm problem in the group of points on an elliptic curve [21], block cipher equations [22], [23], and stream cipher equations [24], [25].

Some results suggest that the time complexity of quantum annealing falls within the sub-exponential class; however, so far, determining the full formal complexity requires further research. Another important issue related to the specification of quantum annealing is that the quantum annealing qubit should not be compared to the qubit in gate-based quantum computers, as these are different types of hardware components.

C. Quantum annealing methods for factorization and determining B-smooth Numbers

1) Direct factorization: Direct methods for factorization of $N = pq$ through quantum annealing are known for several years, see [26]–[28]. The main idea of transforming the problem involves the binary representation $\mathrm{p}_{\tau(\mathrm{p})} \ldots \mathrm{p}_3\mathrm{p}_2\mathrm{p}_1\mathrm{p}_0$ of p, the binary representation $\mathrm{q}_{\tau(\mathrm{q})} \ldots \mathrm{q}_3\mathrm{q}_2\mathrm{q}_1\mathrm{q}_0$ of q and defining a cost function of these variables

$$\begin{aligned} &\underset{dir}{\mathcal{K}}\left(\mathrm{p}_0, \mathrm{p}_1, \mathrm{p}_2, \ldots, \mathrm{p}_{\tau(\mathrm{p})}, \mathrm{q}_0, \mathrm{q}_1, \mathrm{q}_2, \ldots \mathrm{q}_{\tau(\mathrm{q})}\right) = \\ &\left(N - \left(\sum_i \mathrm{p}_\mathrm{i} 2^i\right)\left(\sum_i \mathrm{q}_\mathrm{i} 2^i\right)\right)^2 = (N - pq)^2, \end{aligned} \tag{10}$$

which gives us the optimization form of the factorization problem, but not the QUBO form.

2) Degree reduction of a polynomial: To reduce $\underset{dir}{\mathcal{K}}$ to the QUBO form $\mathcal{Q}$, we must reduce the degree of the polynomial $\underset{dir}{\mathcal{K}}$ to 2. The reduction process involves applying a series of transformations λ to each monomial of the original function to obtain a new function, which is a polynomial of the appropriate degree. The transformation λ is defined to preserve the minima of the function at the same points (values of the arguments) but reduces the degree of the given monomial. The transformation λ is performed by introducing a new auxiliary variable $\hat{\mathrm{a}} \in \{0, 1\}$ and adding a so-called penalty $\mathcal{P}_{\hat{\mathrm{a}}}$:

$$\begin{aligned} \mathrm{p}_0\mathrm{p}_1\mathrm{p}_2 &\rightarrow \hat{\mathrm{a}}\mathrm{p}_1 + \mathcal{P}_{\hat{\mathrm{a}}}, \\ \mathcal{P}_{\hat{\mathrm{a}}} &= 2(\mathrm{p}_0\mathrm{p}_1 - 2\hat{\mathrm{a}}(\mathrm{p}_0 + \mathrm{p}_1) + 3\hat{\mathrm{a}}) \end{aligned} \tag{11}$$

Considering the above expressions as functions, we find that the minimal values of the expressions (i.e., zero) on both sides of the transformation are achieved for the same values of $\mathrm{p}_0, \mathrm{p}_1, \mathrm{p}_2$, which can be easily verified through exhaustive search. This also holds true for expressions of the form $\psi\mathrm{p}_0\mathrm{p}_1\mathrm{p}_2$, where $\psi \in \mathbb{R}$ (thus for any monomial form), and obviously remains true for functions that are sums of such monomials.

The reduction can be performed on the form of the function before squaring (in which case we must reduce the degree of the polynomial to 1, linearization see [8]), and we denote it by $\mathring{\lambda}_{lin}$ or after squaring (in which case it is sufficient to reduce the degree to 2, quadratization see [26]), and we denote it by $\mathring{\lambda}_{quad}$.

3) Detecting smoothness: Factorization through quantum annealing is used to detect B-smooth numbers, whose identification is motivated as described in subsection II-A. This is done in a very intuitive and simple way. A potentially smooth number $h > B$ is broken down into two divisors

$$h = fg. \tag{12}$$

We define the lengths: $\tau(g)$ as the number of bits $\lfloor h/s_1 \rfloor$ and $\tau(f)$ as the number of bits s_k, where s_1 is the smallest and s_k is the largest prime in the chosen base $\mathbb{B}$. Additionally, instead of taking s_1 as a reference, we can choose a larger number from the base and check and record the divisibility of h by the smallest primes (for example $2, 3$) in an efficient manner

before the annealing process, which can increase the efficiency of the procedure.

The values $\tau(g)$ and $\tau(f)$ determine the number of binary variables used to represent g and f, respectively. If we obtain a non-trivial $f \mid h$, we can proceed to further factorize g (if necessary), recursively invoking the procedure until all divisors are less than or equal to B and we confirm smoothness. Otherwise, we treat the number h as non-smooth. This results in a simple procedure for finding B-smooth numbers through quantum annealing.

4) Multiplication table procedure: In practice, for smoothness detection, we use an enhanced method of factorization through quantum annealing. Specifically, a multiplication table is established, constructed exactly according to the long multiplication technique, where the binary variables of the table are h_i, f_i, g_i, representing the bits of h, f, g, respectively, and the carry bits c_i. The table is then divided into blocks, each covering several columns. The width of the i-th block (i.e., the number of its columns) is denoted by w_i. This gives us a Table I, where w_0 is always equal to 1, because, in practice, checking for divisibility by at least $s_1 = 2$ occurs before the smoothness verification procedure.

TABLE I
MULTIPLICATION TABLE

							$f_{\tau(f)}$	.	.	.	f_2	f_1	1
							$g_{\tau(g)}$	.	.	.	g_2	g_1	1
						c_0	$f_{\tau(f)}$	.	.	.	f_2	f_1	1
				c_1	$f_{\tau(f)}g_1$	.	.	.	f_2g_1		f_1g_1	g_1	
			.	$f_{\tau(f)}g_2$	.	.	.	f_2g_2	f_1g_2		g_2		
		.	.						.				
		.	.					.					
	c_k	.					.						
	$f_{\tau(f)}g_{\tau(g)}$	.	.	.	$f_2g_{\tau(g)}$	$f_1g_{\tau(g)}$	$g_{\tau(g)}$						
	$h_{\tau(h)}$			.			.		.		h_2	h_1	1
	L-th block			.			.		.		1. block		0. block

Instead of forming a single objective function $\mathcal{K}$ as before, we break down the problem by equating each expression $\mathcal{K}_i$ from the i-th block separately to the value read from the fragment of bits of the number h, as an independent binary representation: n_i. More specifically, we define K_i as follows:

$$\mathcal{K}_i = \overline{\mathcal{K}}_i(\mathrm{f}_0, \mathrm{f}_1, \ldots, \mathrm{f}_{\tau(\mathrm{f})}, \mathrm{g}_0, \mathrm{g}_1, \ldots, \mathrm{g}_{\tau(\mathrm{g})}) + \mathcal{C}_i(\mathrm{c}_0, \mathrm{c}_1, \ldots, \mathrm{c}_\mathrm{J}) - 2^{w_i}\mathcal{C}_{i+1}(\mathrm{c}_0, \mathrm{c}_1, \ldots, \mathrm{c}_\mathrm{J}) - n_i, \tag{13}$$

where:

- $\overline{\mathcal{K}}_i(\mathrm{f}_0, \mathrm{f}_1, \ldots, \mathrm{f}_{\tau(\mathrm{f})}, \mathrm{g}_0, \mathrm{g}_1, \ldots, \mathrm{g}_{\tau(\mathrm{g})})$ is the result of multiplying the bits $\mathrm{f_i}$ and $\mathrm{g_i}$ from the i-th block,
- $C_i(\mathrm{c}_0, \mathrm{c}_1, \ldots, \mathrm{c}_\mathrm{J})$ is an expression created by the shifted bits c_i in the i-th block,
- $\tau(c)$ is the number of carry bits into the i-th block.

In this way, we can formulate the QUBO form:

$$\mathcal{Q} = \mathring{\lambda}_{lin}\left(\mathcal{K}_1\right)^2 + \mathring{\lambda}_{lin}\left(\mathcal{K}_2\right)^2 + \cdots + \mathring{\lambda}_{lin}\left(\mathcal{K}_L\right)^2 \tag{14}$$

Additional improvements involve techniques for determining the total number of carry bits $J+1$, which have been explained more precisely in [8].

Finally, quantum annealing is used to find the solution to the problem Q. By applying this procedure to the steps described in subsubsection II-C3, we determine whether the number h is B-smooth.

III. EXPERIMENT

A. Contribution

The contribution of this work lies in proposing an improvement to the hybrid quadratic sieve method, known from [8], allowing for the assessment of the maximum potential of quantum annealing-based attacks on the integer factorization problem, and obtaining practical results through computational experiments.

The improvement involves the use of the known method for finding B-smooth numbers via quantum annealing, generally described in subsection II-C, to sieve pairs (a, b) in search of smooth elements in the GNFS method, as described in section II (in the previous work, a less effective method – the quadratic sieve – was used). While the sieving was conducted through quantum computations, all other steps, such as polynomial selection, solving the system of linear equations, square root extraction, and smaller calculations, were performed classically (without a quantum computer). The sieving of pairs (a, b) was done by attempting a quantum factorization of the corresponding values (see subsection II-A), preceded by a classical removal of the highest power of 2 dividing the value.

This improvement, which enabled achieving the maximum problem size, meant that the framework of hybrid (quantum-classical) computations, the most efficient of the known classical methods was utilized, while solutions to subproblems (searching for B-smooth relations) were sought through quantum annealing. This did not change the number of basic operations, thus the approach remains within the subexponential complexity class. The potential practical speed-up in full-scale attacks remains an unresolved issue due to the lack of precise knowledge about the complexity of quantum annealing. The answer to the question of whether the presented method will solve larger instances than the classical GNFS in full-scale applications lies in comparing the smoothness verification procedures in the classical and hybrid methods. Unfortunately, such a comparison is not possible when relying solely on computation time without determining the complexity of quantum annealing.

This practical example demonstrates how solving small subproblems by currently available quantum annealing computers, using all available results, realistically translates to the size of the factorization problem being solved. These are the implications of the possibility of a quantum solution for subproblems of this size.

The practical results of the experiment are described below.

B. Purpose of the tests

The goal of the experiment was to investigate the maximum size of the integer factorization problem in cryptography that can be solved using quantum annealing. This approach did not exclude the use of the best tools, both quantum and classical. By framing the problem in this way, we demonstrate how effectively cryptography based on the integer factorization problem and the discrete logarithm can already be attacked

using quantum computers, despite remaining in the same computational complexity class, which, for now, cannot be practically changed. This provides a very realistic and current view of the range of possibilities available to attackers, while also taking a practical step toward hybrid attack ideas with lower complexity than classical methods, such as those proposed in [6].

C. Methodology - classical part

The methodology of the experiment comprises: detailed solutions in GNFS, the method of selecting a computational example, and the allowable precision for annealing. Key details of the GNFS method pertain to the implementation of random relation sampling, examined for B-smoothness. In the experiment, the pairs (a, b) were limited by a constant constraint $0 < b \leq D$, and a restriction $|a| \leq A$, where A was increased when, despite exhausting available pairs, no solution to the system of equations existed, according to [1]. The degree d of the polynomial $\mathcal{F}$ was specified at the input, and a standard method for determining the polynomial $\mathcal{F}$ was adopted by expressing N in base m. The search for examples was designed to limit the sizes of the algebraic-side value: $N_{\mathbb{Q}[\rho]}(a+bm)$ and the integer-side value $a+bm$, to be written on W bits. Importantly, the value of W reflects the asymptotic size of the sieved numbers to $N^{\frac{1}{d}}$, where $d \sim \sqrt[3]{\frac{3\log N}{\log\log N}}$ (see [11]). Examples exceeding this limit in the relation search phase were rejected. This naturally results from the computational limitations assumed for quantum annealing.

D. Methodology - quantum part

The experiment was conducted using a quantum annealing computer, the most powerful commercially available model at the time. Specifically, the D-Wave Advantage system, model Advantage system 4.1, was utilized. The enhanced version of this model, Advantage system 2, was not commercially available at the time of the research (March 2024).

The computer used was characterized by the following basic technical data: a total of 5760 qubits, a Pegasus topology in a 16x16 unit cell configuration, and 40279 couplers. The number of couplers affects the ability to embed more complex problems, and therefore, with an increase in the number of couplers, the capability to sieve larger B-smooth numbers is expected. A more detailed specification of the device can be found in [29].

The quantum computer was queried using tools from the Ocean SDK and D-Wave system library. Annealing was carried out with a sampling number of 10,000, allowing a maximum of four attempts to check the B-smoothness of a given pair. The number of samples determines the precision of the result but increases the computation time. The annealing time was set at $20\mu s$, which is standard, well-known solution.

QUBO transforming into the QPU structure (embedding) were automatically handled using Ocean SDK tools, while the coefficients defining the specific problem were auto-scaled by the QPU solver API.

We allowed up to nine attempts to verify the smoothness of a single element, which was known in advance to be smooth. Even with nine attempts, we obtained a relatively low QPU working time, as shown below.

E. Outcomes

The most significant result achieved was the factorization of a 29-bit number: $448383577 = 20771 \times 21587$. This was accomplished with the input parameters set as follows:

- $D = 2$,
- $B = 224$,
- $d = 4$,
- $W = 13$,
- $M = 1$.

The polynomial $\mathcal{F}$, forming the homomorphism, was given by

$$\mathcal{F} = x^4 + 2x^3 + 11x^2 + 30x + 77.$$

Next, after performing the sieving of smooth elements, we determined the following set: $\{(-1, 1), (1, 1), (1, 2)\}$. These pairs represented the following elements in $\mathbb{Z}$: $-146, -144, -289$, and elements in $\mathbb{Z}[\rho]$ with the following norms: $57, 121, 1521$. By solving the system of equations, we determined the set $\mathbb{S}$, which included $\mathbb{S} = \{(1, 1), (1, 2)\}$. From this, we obtained the values: $X^2 = 41616$, $\phi(\gamma) = 224202378$, and thus $gcd(448383577, 224202378 - 204) = 20771$.

The following results were obtained in the quantum part of the experiment. These are the largest numbers of qubits, respectively logical or physical, for all QUBO problems generated during the smoothness detection procedure for, respectively, $a+bm$ (integer part) and $N_{\mathbb{Q}(\rho)}(a+b\rho)$ (algebraic part):

- logical qubits in integer part: 63,
- logical qubits in algebraic part: 81,
- physical qubits in integer part: 328,
- physical qubits in algebraic part: 471,
- QPU working time: 980 μs.

The "QPU working time" was summed only for the smoothness verification of the smooth elements.

The probability of success was influenced by the choice of N, for which the sieved numbers h were closer in size to the asymptotic values.

IV. Conclusions

The results of the experiment address the question of the maximum practical effectiveness of solving the factorization problem using quantum annealing, assuming a hybrid combination of computations, that is, the most effective classical general method combined with a quantum QUBO solver. The decomposition of the largest problem to date, a 29-bit size, was achieved for all quantum annealing approaches, thereby improving the previous best result [8].

Although the results demonstrate new achievements, there are certain significant limitations, mainly regarding the number of steps in the quantum phase (the number of required B-smooth elements). The number of elements that need to be searched does not change compared to the classical method, which makes this approach unsuitable for significantly larger examples than those addressed by the best classical methods.

To better present the context of this work's result, we provide below the collected parameters of quantum methods used to factor the largest numbers. The comparison is divided into four tables: Table II, Table III, Table IV, Table V, where corresponding rows refer to the same results. The general conclusions drawn from these are as follows. Quantum methods with better complexities are, so far, less practically efficient due to hardware limitations. On the other hand, hybrid methods currently demonstrate greater efficiency. We also observe that, to date, no hybrid method has been practically implemented that clearly improves the complexity compared to its classical counterpart.

TABLE II
FACTORING RECORDS: HARDWARE USED

ID	Problem size (bits)	Hardware	Paper
1	6	gate-based quantum computer	[3]
2	10	NMR adiabatic quantum computer	[4]
3	10	gate-based quantum computer	[30]
4	23	quantum annealer	[5]
5	26	quantum annealer	[8]
6	29	quantum annealer	this paper

TABLE III
FACTORING RECORDS: ALGORITHMS USED

ID	Algorithm
1	Shor's Algorithm
2	hybrid adiabatic quantum algorithm
3	quantum variational imaginary time evolution
4	quantum annealing
5	hybrid method with quantum annealing
6	hybrid method with quantum annealing

TABLE IV
FACTORING RECORDS: COMPLEXITY

ID	Complexity
1	polynomial
2	problem instance dependent
3	unspecified,
4	heuristicaly subexponential
5	heuristicaly subexponential
6	heuristicaly subexponential

TABLE V
FACTORING RECORDS: QUANTUM LOGICAL RESOURCES

ID	Quantum logical resources
1	$\mathcal{O}\left(\log^2 N\right)$ gate model qubits
2	unspecified
3	$\mathcal{O}(\log N)$ gate model qubits
4	unspecified
5	$\mathcal{O}\left(\frac{\log^2 N}{4\left(\frac{3\log N}{\log\log N}\right)^{\frac{2}{3}}}\right)$ quantum annealing qubits
6	$\mathcal{O}\left(\frac{\log^2 N}{16}\right)$ quantum annealing qubits

Good effectiveness in searching for B-smooth numbers was observed up to a maximum size of 11-12 bits. The measured qubit cost in terms of the maximum number of logical qubits (the number of variables in the QUBO problem), equal to 81, corresponds to the asymptotic assumptions of the logical qubit cost of the applied approach: $O(\frac{\log^2 n}{4})$,where n is a sieved number (see [8]). The actual number of used qubits on the quantum computer, i.e., the number of physical qubits, turned out to be more than 5 times greater, due to the necessity to represent one logical qubit by many physical qubits in order to model the QUBO problem within the physical architecture of the QPU chip.

The maximum size of the solved problem should also be related to the discrete logarithm problem. The GNFS method achieves the same size of sieved elements for the variant intended to solve the discrete logarithm problem (DLP). This means that a problem of the same size can be solved using an approach similar to that presented in this work.

In summary, the results of the experiment should be interpreted as an answer to the question: how far are we from attacks on cryptography based on the factorization problem and DLP, which are quantum enhancements of classic general methods? Such attempts may occur sooner than the direct use of quantum methods, due to the lower resource requirements of the former approaches.

REFERENCES

[1] R. Crandall and C. Pomerance, *Prime Numbers. A Computational Perspective*. Springer, 2005.

[2] P. W. Shor, "Algorithms for quantum computation: discrete logarithms and factoring," in *Proceedings 35th annual symposium on foundations of computer science*. Ieee, 1994, pp. 124–134.

[3] M. Amico, Z. H. Saleem, and M. Kumph, "Experimental study of shor's factoring algorithm using the ibm q experience," *Phys. Rev. A*, vol. 100, p. 012305, Jul 2019. [Online]. Available: https://link.aps.org/doi/10.1103/PhysRevA.100.012305

[4] S. Pal, S. Moitra, V. S. Anjusha, A. Kumar, and T. S. Mahesh, "Hybrid scheme for factorisation: Factoring 551 using a 3-qubit NMR quantum adiabatic processor," *Pramana - J Phys*, vol. 92, no. 2, Feb. 2019.

[5] J. Ding, G. Spallitta, and R. Sebastiani, "Experimenting with D-Wave quantum annealers on prime factorization problems," *Front. Comput. Sci.*, vol. 6, Jun. 2024.

[6] D. J. Bernstein, J.-F. Biasse, and M. Mosca, "A low-resource quantum factoring algorithm," Cryptology ePrint Archive, Paper 2017/352, 2017, https://eprint.iacr.org/2017/352. [Online]. Available: https://eprint.iacr.org/2017/352

[7] M. Wroński, *Index Calculus Method for Solving Elliptic Curve Discrete Logarithm Problem Using Quantum Annealing*. Springer International Publishing, 2021, p. 149–155. [Online]. Available: http://dx.doi.org/10.1007/978-3-030-77980-1_12

[8] O. Żołnierczyk and M. Wroński, "Searching b-smooth numbers using quantum annealing: Applications to factorization and discrete logarithm problem," in *Computational Science – ICCS 2023*. Springer Nature Switzerland, 2023, pp. 3–17. [Online]. Available: https://doi.org/10.1007/978-3-031-36030-5_1

[9] A. H. Karamlou, W. A. Simon, A. Katabarwa, T. L. Scholten, B. Peropadre, and Y. Cao, "Analyzing the performance of variational quantum factoring on a superconducting quantum processor," *npj Quantum Information*, vol. 7, no. 1, Oct. 2021. [Online]. Available: http://dx.doi.org/10.1038/s41534-021-00478-z

[10] A. K. Lenstra, H. W. Lenstra, M. S. Manasse, and J. M. Pollard, "The number field sieve," in *Proceedings of the Twenty-Second Annual ACM Symposium on Theory of Computing*, ser. STOC '90. New York, NY, USA: Association for Computing Machinery, 1990, p. 564–572. [Online]. Available: https://doi.org/10.1145/100216.100295

[11] F. Jarvis, *Algebraic number theory*. Springer, 2014.

[12] J. P. Buhler, H. W. Lenstra, and C. Pomerance, "Factoring integers with the number field sieve," in *The development of the number field sieve*, A. K. Lenstra and H. W. Lenstra, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 1993, pp. 50–94.

[13] A. K. Lenstra and H. W. Lenstra, *The Development of the Number Field Sieve*, ser. Lecture Notes in Mathematics. Springer-Verlag, Berlin, Heidelberg, 1993, vol. 1554.

[14] A. Guillevic and S. Singh, "On the alpha value of polynomials in the tower number field sieve algorithm," *Mathematical Cryptology*, vol. 1, no. 1, p. 1–39, Feb. 2021. [Online]. Available: https://journals.flvc.org/mathcryptology/article/view/125142

[15] E. Thomé, "Square root algorithms for the number field sieve," in *Arithmetic of Finite Fields*, F. Özbudak and F. Rodríguez-Henríquez, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2012, pp. 208–224.

[16] C. Bouillaguet, A. Fleury, P.-A. Fouque, and P. Kirchner, "We are on the same side. alternative sieving strategies for the number field sieve," in *Advances in Cryptology – ASIACRYPT 2023*, J. Guo and R. Steinfeld, Eds. Singapore: Springer Nature Singapore, 2023, pp. 138–166.

[17] T. Kadowaki and H. Nishimori, "Quantum annealing in the transverse ising model," *Physical Review E*, vol. 58, no. 5, p. 5355–5363, Nov. 1998.

[18] W. van Dam, M. Mosca, and U. Vazirani, "How powerful is adiabatic quantum computation?" in *Proceedings 42nd IEEE Symposium on Foundations of Computer Science*. IEEE, 2001, p. 279–287. [Online]. Available: http://dx.doi.org/10.1109/SFCS.2001.959902

[19] E. Farhi, J. Goldstone, S. Gutmann, and M. Sipser, "Quantum computation by adiabatic evolution," 2000. [Online]. Available: https://arxiv.org/abs/quant-ph/0001106

[20] M. Wroński, *Practical Solving of Discrete Logarithm Problem over Prime Fields Using Quantum Annealing*. Springer International Publishing, 2022, p. 93–106. [Online]. Available: http://dx.doi.org/10.1007/978-3-031-08760-8_8

[21] M. Wronski and L. Dzierzkowski, "Base of exponent representation matters – more efficient reduction of discrete logarithm problem and elliptic curve discrete logarithm problem to the qubo problem," *Quantum Information and Computation*, vol. 24, no. 7 & 8, p. 541–564, Jun. 2024. [Online]. Available: http://dx.doi.org/10.26421/QIC24.7-8-1

[22] E. Burek, K. Mańk, and M. Wroński, "Searching for an efficient system of equations defining the aes sbox for the qubo problem," *Journal of Telecommunications and Information Technology*, vol. 4, no. 2023, p. 30–37, Oct. 2023. [Online]. Available: http://dx.doi.org/10.26636/jtit.2023.4.1340

[23] E. Burek and M. Wroński, *Quantum Annealing and Algebraic Attack on Speck Cipher*. Springer International Publishing, 2022, p. 143–149. [Online]. Available: http://dx.doi.org/10.1007/978-3-031-08760-8_12

[24] M. Wroński, E. Burek, and M. Leśniak, "(in)security of stream ciphers against quantum annealing attacks on the example of the grain 128 and grain 128a ciphers," *IEEE Transactions on Emerging Topics in Computing*, p. 1–14, 2024. [Online]. Available: http://dx.doi.org/10.1109/TETC.2024.3474856

[25] M. Leśniak, E. Burek, and M. Wroński, *Unsafe Mechanisms of Bluetooth, E_0 Stream Cipher Cryptanalysis with Quantum Annealing*. Springer Nature Switzerland, 2024, p. 389–404. [Online]. Available: http://dx.doi.org/10.1007/978-3-031-63778-0_28

[26] S. Jiang, K. A. Britt, A. J. McCaskey, T. S. Humble, and S. Kais, "Quantum annealing for prime factorization," *Scientific reports*, vol. 8, no. 1, pp. 1–9, 2018.

[27] W. Peng, B. Wang, F. Hu, Y. Wang, X. Fang, X. Chen, and C. Wang, "Factoring larger integers with fewer qubits via quantum annealing with optimized parameters," *Science China Physics, Mechanics; Astronomy*, vol. 62, no. 6, Jan. 2019. [Online]. Available: http://dx.doi.org/10.1007/s11433-018-9307-1

[28] R. Mengoni, D. Ottaviani, and P. Iorio, "Breaking rsa security with a low noise d-wave 2000q quantum annealer: Computational times, limitations and prospects," 2020. [Online]. Available: https://arxiv.org/abs/2005.02268

[29] D.-W. S. Inc, "Qpu-specific physical properties: Advantage_system4.1, user manual." [Online]. Available: https://docs.dwavesys.com/docs/latest/_downloads/d9a3e404f2019e9f35f59c9d2ea2bcf8/09-1262A-D_QPU_Properties_Advantage_system4_1.pdf

[30] R. Selvarajan, V. Dixit, X. Cui, T. S. Humble, and S. Kais, "Prime factorization using quantum variational imaginary time evolution," *Scientific Reports*, vol. 11, no. 1, Oct. 2021. [Online]. Available: http://dx.doi.org/10.1038/s41598-021-00339-x