Generalized Implicit Factorization Problem

Yansong Feng[1,2], Abderrahmane Nitaj[3(✉)], and Yanbin Pan[1,2(✉)]

[1] Key Laboratory of Mathematics Mechanization, Academy of Mathematics and Systems Science, Chinese Academy of Sciences, Beijing, China
[2] School of Mathematical Sciences, University of Chinese Academy of Sciences, Beijing, China
ysfeng2023@163.com, panyanbin@amss.ac.cn
[3] Normandie Univ, UNICAEN, CNRS, LMNO, 14000 Caen, France
abderrahmane.nitaj@unicaen.fr

Abstract. The Implicit Factorization Problem (IFP) was first introduced by May and Ritzenhofen at PKC'09, which concerns the factorization of two RSA moduli $N_1 = p_1q_1$ and $N_2 = p_2q_2$, where p_1 and p_2 share a certain consecutive number of least significant bits. Since its introduction, many different variants of IFP have been considered, such as the cases where p_1 and p_2 share most significant bits or middle bits at the same positions. In this paper, we consider a more generalized case of IFP, in which the shared consecutive bits can be located at *any* positions in each prime, not necessarily required to be located at the same positions as before. We propose a lattice-based algorithm to solve this problem under specific conditions, and also provide some experimental results to verify our analysis.

Keywords: Implicit Factorization Problem · Lattice · LLL algorithm · Coppersmith's algorithm.

1 Introduction

In 1977, Rivest, Shamir, and Adleman proposed the famous RSA encryption scheme [18], whose security is based on the hardness of factoring large integers. RSA is now a very popular scheme with many applications in industry for information security protection. Therefore, its security has been widely analyzed. Although it seems infeasible to break RSA with large modulus entirely with a classical computer now, there still exist many vulnerable RSA instances. For instance, small public key [7,8] or small secret key [4] can lead to some attacks against RSA. In addition, side-channel attacks pose a great threat to RSA [2,5,6], targeting the decryption device to obtain more information about the private key.

It is well known that additional information on the private keys or the prime factors can help attack the RSA scheme efficiently. In 1997, Coppersmith [8,14] proposed an attack that can factor the RSA modulus $N = pq$ in polynomial time if at least half of the most (or least) significant bits of p are given. In 2013, by using Coppersmith's method, Bernstein et al. [3] showed that an attacker can efficiently factor 184 distinct RSA keys generated by government-issued smart cards.

At PKC 2009, May and Ritzenhofen [15] introduced the Implicit Factorization Problem (IFP). It concerns the question of factoring two n-bit RSA moduli $N_1 = p_1q_1$ and $N_2 = p_2q_2$, given the implicit information that p_1 and p_2 share γn of their consecutive least significant bits, while q_1 and q_2 are αn-bit. Using a two-dimensional lattice, May and Ritzenhofen obtained a heuristic result that this implicit information is sufficient to factor N_1 and N_2 with a lattice-based algorithm, provided that $\gamma n > 2\alpha n + 2$.

In a follow-up work at PKC 2010, Faugère *et al.* [9] generalized the Implicit Factorization Problem to the case where the most significant bits (MSBs) or the middle bits of p_1 and p_2 are shared. Specifically, they established the bound of $\gamma n > 2\alpha n + 2$ for the case where the MSBs are shared, using a two-dimensional lattice. For the case where the middle bits of p_1 and p_2 are shared, Faugère *et al.* obtained a heuristic result that q_1 and q_2 could be found from a three-dimensional lattice if $\gamma n > 4\alpha n + 6$.

In 2011, Sarkar and Maitra [21] further expanded the Implicit Factorization Problem by revealing the relations between the Approximate Common Divisor Problem (ACDP) and the Implicit Factorization Problem, and presented the bound of $\gamma > 2\alpha - \alpha^2$ for the following three cases.

1. the primes p_1, p_2 share an amount of the least significant bits (LSBs);
2. the primes p_1, p_2 share an amount of most significant bits (MSBs);
3. the primes p_1, p_2 share both an amount of least significant bits and an amount of most significant bits.

In 2016, Lu *et al.* [13] presented a novel algorithm and improved the bounds to $\gamma > 2\alpha - 2\alpha^2$ for all the above three cases of the Implicit Factorization Problem. In 2015, Peng *et al.* [17] revisited the Implicit Factorization Problem with shared middle bits and improved the bound of Faugère *et al.* [9] up to $\gamma > 4\alpha - 3\alpha^2$. The bound was further enhanced by Wang *et al.* [22] in 2018 up to $\gamma > 4\alpha - 4\alpha\sqrt{\alpha}$.

It is worth noting that in the previous cases, the shared bits are located at the same position for the primes p_1 and p_2.

In this paper, we present a more generalized case of the Implicit Factorization Problem that allows for arbitrary consecutive shared locations, rather than requiring them to be identical in the primes, as in previous research. More precisely, we propose the Generalized Implicit Factorization Problem (GIFP), which concerns the factorization of two n-bit RSA moduli $N_1 = p_1q_1$ and $N_2 = p_2q_2$ when p_1 and p_2 share γn consecutive bits, where the shared bits are not necessarily required to be located at the same positions. See Fig. 1 for an example, where the starting positions for the shared bits in p_1 and p_2 may be different.

We transform the GIFP into the Approximate Common Divisor Problem and then, employ Coppersmith's method with some optimization strategy, we propose a polynomial time algorithm to solve it when $\gamma > 4\alpha(1 - \sqrt{\alpha})$.

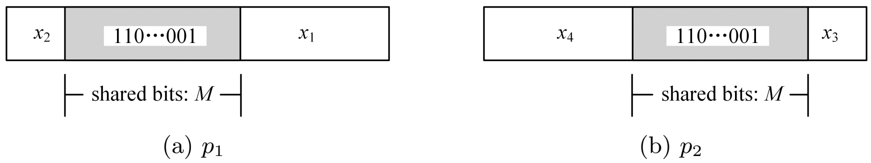

Fig. 1: Shared bits M for p_1 and p_2

In Table 1, we present a comparison of our new bound on γ with the known former bounds obtained by various methods to solve the Implicit Factorization Problem.

	LSBs	MSBs	both LSBs-MSBs	Middle bits	General
May, Ritzenhofen [15]	2α	-	-	-	-
Faugère, *et al.* [9]	2α	-	-	4α	-
Sarkar, Maitra [21]	$2\alpha - \alpha^2$	$2\alpha - \alpha^2$	$2\alpha - \alpha^2$	-	-
Lu, *et al.* [13]	$2\alpha - 2\alpha^2$	$2\alpha - 2\alpha^2$	$2\alpha - 2\alpha^2$	-	-
Peng, *et al.*[17]	-	-	-	$4\alpha - 3\alpha^2$	-
Wang, *et al.*[22]	-	-	-	$4\alpha(1-\sqrt{\alpha})$	-
This work	-	-	-	-	$4\alpha(1-\sqrt{\alpha})$

Table 1: Asymptotic lower bound of γ in the Implicit Factorization Problem for n-bit $N_1 = p_1 q_2$ and $N_2 = p_2 q_2$ where the number of shared bits is γn, q_1 and q_2 are αn-bit.

It can be seen in Table 1 that the bounds for the Implicit Factorization Problem for sharing middle bits are inferior to those of other variants. This is because the unshared bits in the Implicit Factorization Problem for LSBs or MSBs or both LSBs and MSBs are continuous, and only one variable is necessary to represent the unshared bits while at least two variables are needed to represent the unshared bits in the Implicit Factorization Problem sharing middle bits or GIFP. In addition, our bound for GIFP is identical to the variant of IFP sharing the middle bits located in the same position. However, it is obvious that the GIFP relaxes the constraints for the positions of the shared bits.

Therefore, with the same bound for the number of shared bits as in the IFP sharing middle bits at the same position, we show that the Implicit Factorization Problem can still be solved efficiently when the positions for the sharing bits are located differently.

There are still open problems, and the most important one is: can we improve our bound $4\alpha\left(1-\sqrt{\alpha}\right)$ for GIFP to $2\alpha - 2\alpha^2$ or even better? A positive answer seems not easy since the bound for GIFP directly yields a bound for any known variant of IFP. Improving the bound for GIFP to the one better than $4\alpha\left(1-\sqrt{\alpha}\right)$ means that we can improve the bound for the variant of IFP sharing the middle bits located in the same position, and improving the bound for GIFP to the one better than $2\alpha - 2\alpha^2$ means that we can improve the bound for any known variant of IFP.

Roadmap Our paper is structured as follows. Section 2 presents some required background for our approaches. In Section 3, we present our analysis of the Generalized Implicit Factorization Problem, which constitutes our main result. Section 4 details the experimental results used to validate our analysis. Finally, we provide a brief conclusion in Section 5.

2 Notations and Preliminaries

Notations Let $\mathbb{Z}$ denote the ring of integers, i.e., the set of all integers. We use lowercase bold letters (e.g., $\mathbf{v}$) for vectors and uppercase bold letters (e.g., $\mathbf{A}$) for matrices. The notation $\binom{n}{m}$ represents the number of ways to select m items out of n items, which is defined as $\frac{n!}{m!(n-m)!}$. If $m > n$, we set $\binom{n}{m} = 0$.

2.1 Lattices, SVP, and LLL

Let $m \geq 2$ be an integer. A lattice is a discrete additive subgroup of $\mathbb{R}^m$. A more explicit definition is presented as follows.

Definition 1 (Lattice). *Let $\mathbf{v_1}, \mathbf{v_2}, \ldots, \mathbf{v_n} \in \mathbb{R}^m$ be n linearly independent vectors with $n \leq m$. The lattice $\mathcal{L}$ spanned by $\{\mathbf{v_1}, \mathbf{v_2}, \ldots, \mathbf{v_n}\}$ is the set of all integer linear combinations of $\{\mathbf{v_1}, \mathbf{v_2}, \ldots, \mathbf{v_n}\}$, i.e.,*

$$\mathcal{L} = \left\{ \mathbf{v} \in \mathbb{R}^m \mid \mathbf{v} = \sum_{i=1}^{n} a_i \mathbf{v_i}, a_i \in \mathbb{Z} \right\}.$$

The integer n denotes the rank of the lattice $\mathcal{L}$, while m represents its dimension. The lattice $\mathcal{L}$ is said to be full rank if $n = m$. We use the matrix $\mathbf{B} \in \mathbb{R}^{n\times m}$, where each vector $\mathbf{v_i}$ contributes a row to $\mathbf{B}$. The determinant of $\mathcal{L}$ is defined as $\det(\mathcal{L}) = \sqrt{\det(\mathbf{B}\mathbf{B}^t)}$, where $\mathbf{B}^t$ is the transpose of $\mathbf{B}$. If $\mathcal{L}$ is full rank, this reduces to $\det(\mathcal{L}) = |\det(\mathbf{B})|$.

The Shortest Vector Problem (SVP) is one of the famous computational problems in lattices.

Definition 2 (Shortest Vector Problem (SVP)). *Given a lattice $\mathcal{L}$, the Shortest Vector Problem (SVP) asks to find a non-zero lattice vector $\mathbf{v} \in \mathcal{L}$ of minimum Euclidean norm, i.e., find $\mathbf{v} \in \mathcal{L}\backslash\{\mathbf{0}\}$ such that $\|\mathbf{v}\| \leq \|\mathbf{w}\|$ for all non-zero $\mathbf{w} \in \mathcal{L}$.*

Although SVP is NP-hard under randomized reductions [1], there exist algorithms that can find a relatively short vector, instead of the exactly shortest vector, in polynomial time, such as the famous LLL algorithm proposed by Lenstra, Lenstra, and Lovasz [12] in 1982. The following result is useful for our analysis[14].

Theorem 1 (LLL Algorithm). *Given an n-dimensional lattice $\mathcal{L}$, we can find an LLL-reduced basis $\{\mathbf{v_1}, \mathbf{v_2}, \ldots, \mathbf{v_n}\}$ of $\mathcal{L}$ in polynomial time, which satisfies*

$$\|\mathbf{v_i}\| \le 2^{\frac{n(n-1)}{4(n+1-i)}} \det(\mathcal{L})^{\frac{1}{n+1-i}}, \quad \textit{for} \quad i = 1, \ldots, n.$$

Theorem 1 presents the upper bounds for the norm of the i-th vector in the LLL-basis using the determinant of the lattice.

2.2 Coppersmith's method

In 1996, Coppersmith [8,14] proposed a lattice-based method for finding small solutions of univariate modular polynomial equations modulo a positive integer M, and another lattice-based method for finding the small roots of bivariate polynomial equations. The methods are based on finding short vectors in a lattice. We briefly sketch the idea below. More details can be found in [14].

Let M be a positive integer, and $f(x_1, \ldots, x_k)$ be a polynomial with integer coefficients. Suppose we want to find a small solution $(y_1, \ldots, y_k)$ of the modular equation $f(x_1, \ldots, x_k) \equiv 0 \pmod{M}$ with the bounds $y_i < X_i$ for $i = 1, \ldots, k$.

The first step is to construct a set G of k-variate polynomial equations such that, for each $g_i \in G$ with $i = 1, \ldots, k$, we have $g_i(y_1, \ldots, y_k) \equiv 0 \pmod{M}$. Then we use the coefficient vectors of $g_i(x_1X_1, \ldots, x_kX_k)$, $i = 1, \ldots, k$, to construct a k-dimensional lattice $\mathcal{L}$. Applying the LLL algorithm to $\mathcal{L}$, we get a new set H of k polynomial equations $h_i(x_1, \ldots, x_k)$, $i = 1, \ldots, k$, with integer coefficients such that $h_i(y_1, \ldots, y_k) \equiv 0 \pmod{M}$. The following result shows that one can get $h_i(y_1, \ldots, y_k) = 0$ over the integers in some cases, where for $h(x_1, \ldots, x_k) = \sum_{i_1 \ldots i_k} a_{i_1 \ldots i_k} x_1^{i_1} \cdots x_1^{i_k}$, the Euclidean norm is defined by $\|h(x_1, \ldots, x_k)\| = \sqrt{\sum_{i_1 \ldots i_k} a_{i_1 \ldots i_k}^2}$.

Theorem 2 (Howgrave-Graham [11]). *Let $h(x_1, \ldots, x_k) \in \mathbb{Z}[x_1, \ldots, x_k]$ be a polynomial with at most ω monomials. Let M be a positive integer. If there exist k integers $(y_1, \ldots, y_k)$ satisfying the following two conditions:*

1. *$h(y_1, \ldots, y_k) \equiv 0 \pmod{M}$, and there exist k positive integers $X_1, \ldots, X_k$ such that $|y_1| \le X_1, \ldots, |y_k| \le X_k$,*
2. *$\|h(x_1X_1, \ldots, x_kX_k)\| < \frac{M}{\sqrt{\omega}}$,*

then $h(y_1, \ldots, y_k) = 0$ holds over the integers.

From Theorem 1, we can obtain the vectors $\mathbf{v_1}, \mathbf{v_2}, \ldots, \mathbf{v_k}$ in the LLL reduced basis of $\mathcal{L}$. This yields k integer polynomials $h_1(x_1, \ldots, x_k), \ldots, h_k(x_1, \ldots, x_k)$, all of which share the desired solution $(y_1, \ldots, y_k)$, that is $h_i(y_1, \ldots, y_k) \equiv 0 \pmod{M}$ for $i = 1, \ldots, k$.

To combine Theorem 1 and Theorem 2, for $i = k$, we set

$$2^{\frac{n(n-1)}{4(n+1-i)}} \det(\mathcal{L})^{\frac{1}{n+1-i}} < \frac{M}{\sqrt{\dim(\mathcal{L})}}.$$

Ultimately, the attainment of the desired root hinges upon effectively resolving the system of integer polynomials using either the resultant method or the Gröbner basis approach. However, in order for a Gröbner basis computation to find the common root, the following heuristic assumption needs to hold.

Assumption 1 *The k polynomials $h_i(x_1, \cdots, x_k)$, $i = 1, \cdots, k$, that are derived from the reduced basis of the lattice in the Coppersmith method are algebraically independent. Equivalently, the common root of the polynomials $h_i(x_1, \cdots, x_k)$ can be found by computing the resultant or computing the Gröbner basis.*

Assumption 1 is often used in connection with Coppersmith's method in the multivariate scenario [4,14,21,13,22]. Since our attack in Section 3 relies on Assumption 1, it is heuristic. However, our experiments in Section 4 justify the validity of our attack and show that Assumption 1 perfectly holds true.

3 Generalized Implicit Factorization Problem

This section presents our analysis of the Generalized Implicit Factorization Problem (GIFP) in which p_1 and p_2 share an amount of consecutive bits at different positions.

3.1 Description of GIFP

This section proposes the Generalized Implicit Factorization Problem (GIFP), which concerns the factorization of two n-bit RSA moduli, $N_1 = p_1q_1$ and $N_2 = p_2q_2$, under the implicit hint that the primes p_1 and p_2 share a specific number, γn, of consecutive bits. In contrast to previous studies [9,13,15,19,20,22], where the shared bits were assumed to be located at the same positions in p_1 and p_2, the proposed GIFP considers a more general case where the shared bits can be situated at arbitrary positions.

Definition 3 (GIFP(n, α, γ)). *Given two n-bit RSA moduli $N_1 = p_1q_1$ and $N_2 = p_2q_2$, where q_1 and q_2 are αn-bit, assume that p_1 and p_2 share γn consecutive bits, where the shared bits may be located in different positions of p_1 and p_2. The Generalized Implicit Factorization Problem (GIFP) asks to factor N_1 and N_2.*

The introduction of GIFP expands the scope of the Implicit Factorization Problem and presents a more realistic and challenging scenario that can arise in practical applications. In real-world settings, it is more probable to encounter situations where the shared location of bits differs between primes. Therefore, it is essential to develop algorithms and analysis that can handle such cases where the shared bits are situated at different positions. By considering the Generalized Implicit Factorization Problem (GIFP), we need to avoid situations where the system that creates RSA keys lack entropy.

3.2 Algorithm for GIFP

We will show our analysis of the GIFP in this subsection. The main idea is also to relate the Approximate Common Divisor Problem (ACDP) to the Implicit Factorization Problem.

Theorem 3. *Under Assumption 1, GIFP(n, α, γ) can be solved in polynomial time when*

$$\gamma > 4\alpha\left(1 - \sqrt{\alpha}\right),$$

provided that $\alpha + \gamma \leq 1$.

Proof. Without loss of generality, we can assume that the starting and ending positions of the shared bits are known. When these positions are unknown, we can simply traverse the possible starting positions of the shared bits, which will just scale the time complexity for the case that we know the position by a factor $\mathcal{O}(n^2)$.

Hence, we suppose that p_1 shares γn-bits from the $\beta_1 n$-th bit to $(\beta_1+\gamma)n$-th bit, and p_2 shares bits from $\beta_2 n$-th bit to $(\beta_2+\gamma)n$-th bit, where β_1 and β_2 are known with $\beta_1 \leq \beta_2$ (see Fig. 1). Then we can write

$$p_1 = x_1 + M2^{\beta_1 n} + x_2 2^{(\beta_1+\gamma)n}, \quad p_2 = x_3 + M2^{\beta_2 n} + x_4 2^{(\beta_2+\gamma)n},$$

with $M < 2^{\gamma n}$, $x_1 < 2^{\beta_1 n}$, $x_2 < 2^{(\beta-\beta_1)n}$, $x_3 < 2^{\beta_2 n}$, $x_4 < 2^{(\beta-\beta_2)n}$ where $\beta = 1 - \alpha - \gamma$. From this, we deduce

$$\begin{aligned} 2^{(\beta_2-\beta_1)n} p_1 &= x_1 2^{(\beta_2-\beta_1)n} + M2^{\beta_2 n} + x_2 2^{(\beta_2+\gamma)n} \\ &= x_1 2^{(\beta_2-\beta_1)n} + (p_2 - x_3 - x_4 2^{(\beta_2+\gamma)n}) + x_2 2^{(\beta_2+\gamma)n} \\ &= p_2 + (x_1 2^{(\beta_2-\beta_1)n} - x_3) + (x_2 - x_4) 2^{(\beta_2+\gamma)n}. \end{aligned}$$

Then, multiplying by q_2, we get

$$N_2 + (x_1 2^{(\beta_2-\beta_1)n} - x_3) q_2 + (x_2 - x_4) q_2 2^{(\beta_2+\gamma)n} = 2^{(\beta_2-\beta_1)n} p_1 q_2.$$

Next, we define the polynomial

$$f(x, y, z) = xz + 2^{(\beta_2+\gamma)n} yz + N_2,$$

which shows that $(x_1 2^{(\beta_2-\beta_1)n} - x_3, x_2 - x_4, q_2)$ is a solutions of

$$f(x, y, z) \equiv 0 \pmod{2^{(\beta_2-\beta_1)n} p_1}.$$

Let m and t be integers to be optimized later with $0 \leq t \leq m$. To apply Coppersmith's method, we consider a family of polynomials $g_{i,j}(x, y, z)$ for $0 \leq i \leq m$ and $0 \leq j \leq m - i$:

$$g_{i,j}(x, y, z) = (yz)^j f(x, y, z)^i \left(2^{(\beta_2-\beta_1)n}\right)^{m-i} N_1^{\max(t-i,0)}.$$

These polynomials satisfy

$$\begin{aligned}
&g_{i,j}\left(x_1 2^{(\beta_2-\beta_1)n} - x_3, x_2 - x_4, q_2\right) \\
&= (x_2 - x_4)^j q_2^j \left(2^{(\beta_2-\beta_1)n} p_1 q_2\right)^i \left(2^{(\beta_2-\beta_1)n}\right)^{m-i} N_1^{\max(t-i,0)} \\
&= (x_2 - x_4)^j q_2^{j+i} q_1^{\max(t-i,0)} \left(2^{(\beta_2-\beta_1)n}\right)^m p_1^{\max(t-i,0)+i} \\
&\equiv 0 \pmod{\left(2^{(\beta_2-\beta_1)n}\right)^m p_1^t}.
\end{aligned}$$

On the other hand, we have

$$\begin{aligned}
\left|x_1 2^{(\beta_2-\beta_1)n} - x_3\right| &\le \max\left(x_1 2^{(\beta_2-\beta_1)n}, x_3\right) \\
&\le \max\left(2^{\beta_1 n} 2^{(\beta_2-\beta_1)n}, 2^{\beta_1 n}\right) \\
&= 2^{\beta_2 n},
\end{aligned}$$

and

$$|x_2 - x_4| \le \max(x_2, x_4) = 2^{(\beta-\beta_2)n}.$$

Also, we have $q_2 = 2^{\alpha n}$. We then set

$$X = 2^{\beta_2 n},\ Y = 2^{(\beta-\beta_1)n},\ Z = 2^{\alpha n}.$$

To reduce the determinant of the lattice, we introduce a new variable w for p_2, and multiply the polynomials $g_{i,j}(x, y, z)$ by a power w^s for some s that will be optimized later. Similar to t, we also require $0 \le s \le m$

Note that we can replace zw in $g_{i,j}(x, y, z)w^s$ by N_2. We want to eliminate this multiple. Since $\gcd(N_2, 2N_1) = 1$, there exists an inverse of N_2, denoted as N_2^{-1}, such that $N_2 N_2^{-1} \equiv 1 \pmod{\left(2^{(\beta_2-\beta_1)n}\right)^m N_1^t}$. We then eliminate $(zw)^i$ from the original polynomial by multiplying it by N_2^{-i}, while ensuring that the resulting polynomial evaluation is still a multiple of $\left(2^{(\beta_2-\beta_1)n}\right)^m p_1^t$. By selecting the appropriate parameter s, we aim to reduce the determinant of the lattice.

Remark 1. For simplicity, the results after the above treatment of $g_{i,j}(x, y, z)$ are denoted as $g_{i,j}(x, y, z, w)$.

Consider the lattice $\mathcal{L}$ spanned by the matrix $\mathbf{B}$ whose rows are the coefficients of the polynomials $g_{i,j}(x, y, z, w)$ for $0 \le i \le m$, $0 \le j \le m - i$. The rows are ordered following the rule that $g_{i,j}(x, y, z, w) \prec g_{i',j'}(x, y, z, w)$ if $i < i'$ or if $i = i'$ and $j < j'$. The columns are ordered following the monomials so that $x^i y^j z^{i+j} w^s \prec x^{i'} y^{j'} z^{i'+j'} w^s$ if $j < j'$ or if $j = j'$ and $i < i'$.

Table 2 presents a matrix $\mathbf{B}$ with $m = 2$, $s = 2$, $t = 2$ where $*$ represents a nonzero term.

By construction, the square matrix B is left triangular. Hence, the dimension of the lattice is

$$\omega = \sum_{i=0}^{m} \sum_{j=0}^{m-i} 1 = \sum_{i=0}^{m} (m - i + 1) = \frac{1}{2}(m+1)(m+2)$$

	w^2	yzw^2	$y^2z^2w^2$	xzw^2	xyz^2w^2	$x^2z^2w^2$
$g_{0,0}$	$M^4N_1^2W^2$	0	0	0	0	0
$g_{0,1}$	0	$YM^4N_1^2ZW^2N_2^{-1}$	0	0	0	0
$g_{0,2}$	0	0	$Y^2M^4N_1^2Z^2W^2N_2^{-2}$	0	0	0
$g_{1,0}$	*	*	0	$XM^3N_1ZW^2N_2^{-1}$	0	0
$g_{1,1}$	0	*	*	*	$YXM^3N_1Z^2W^2N_2^{-2}$	0
$g_{2,0}$	*	*	*	*	*	$X^2M^2Z^2W^2N_2^{-2}$

Table 2: The matrix of the lattice with $m = 2$, $s = 2$, $t = 2$ and $M = 2^{(\beta_2-\beta_1)n}$.

and its determinant is

$$\det(B) = \det(\mathcal{L}) = X^{e_X} Y^{e_Y} Z^{e_Z} W^{e_W} 2^{(\beta_2-\beta_1)n e_M} N_1^{e_N},$$

with

$$\begin{aligned}
e_X &= \sum_{i=0}^{m} \sum_{j=0}^{m-i} i = \frac{1}{6} m(m+1)(m+2), \\
e_Y &= \sum_{i=0}^{m} \sum_{j=0}^{m-i} j = \frac{1}{6} m(m+1)(m+2), \\
e_Z &= \sum_{i=0}^{m} \sum_{j=0}^{m-i} \max\{i+j-s, 0\} \\
&= \frac{1}{3} m(m+1)(m+2) + \frac{1}{6} s(s+1)(s+2) - \frac{1}{2} s(m+1)(m+2), \\
e_W &= \sum_{i=0}^{s} \sum_{j=0}^{s-i} j = \frac{1}{6} s(s+1)(s+2), \\
e_N &= \sum_{i=0}^{t} \sum_{j=0}^{m-i} (t-i) = \frac{1}{6} t(t+1)(3m-t+4), \\
e_M &= \sum_{i=0}^{m} \sum_{j=0}^{m-i} (m-i) = \frac{1}{3} m(m+1)(m+2).
\end{aligned}$$

The former results are detailed in Appendix A. To combine Theorem 1 and Theorem 2, we set

$$2^{\frac{\omega(\omega-1)}{4(\omega+1-i)}} \det(\mathcal{L})^{\frac{1}{\omega+1-i}} < \frac{\left(2^{(\beta_2-\beta_1)n}\right)^m p_1^t}{\sqrt{\omega}},$$

with $i = 2$. Then

$$\det(\mathcal{L}) < \frac{1}{2^{\frac{\omega-1}{4}} \sqrt{\omega}} \left(2^{(\beta_2-\beta_1)n}\right)^{\omega m} p_1^{t\omega},$$

and

$$X^{e_X}Y^{e_Y}Z^{e_Z}W^{e_W}2^{(\beta_2-\beta_1)ne_M}N_1^{e_N} < \frac{1}{2^{\frac{\omega-1}{4}}\sqrt{\omega}}\left(2^{(\beta_2-\beta_1)n}\right)^{\omega m}p_1^{t\omega}. \qquad (1)$$

Next, we set $s = \sigma m$ with $0 \le \sigma \le 1$, $t = \tau m$ with $0 \le \tau \le 1$, and we use $N \approx 2^n$, $p_1 \approx 2^{(1-\alpha)n}$, $X = 2^{\beta_2 n}$, $Y = 2^{(\beta-\beta_1)n}$, $Z = 2^{\alpha n}$, $W = 2^{(1-\alpha)n}$ and the most significant parts of e_X, e_Y, e_Z, e_W, e_N, e_M as

$$\begin{aligned}
e_X &= \frac{1}{6}m^3 + o\left(m^3\right), \\
e_Y &= \frac{1}{6}m^3 + o\left(m^3\right), \\
e_Z &= \frac{1}{3}m^3 + \frac{1}{6}\sigma^3 m^3 - \frac{1}{2}\sigma m^3 + o\left(m^3\right), \\
e_W &= \frac{1}{6}\sigma^3 m^3 + o\left(m^3\right), \\
e_N &= \frac{1}{6}\tau^2(3-\tau)m^3 + o\left(m^3\right), \\
e_M &= \frac{1}{3}m^3 + o\left(m^3\right).
\end{aligned}$$

Similarly, we use

$$m\omega = \frac{1}{2}m^3 + o\left(m^3\right).$$

Then, after taking logarithms, dividing by nm^3, and neglecting the very small terms, i.e., $o\left(m^3\right)$, the inequality (1) implies

$$\begin{aligned}
&\frac{1}{6}\beta_2 + \frac{1}{6}(\beta-\beta_1) + \alpha(\frac{1}{3} + \frac{1}{6}\sigma^3 - \frac{1}{2}\sigma) + \frac{1}{6}\sigma^3(1-\alpha) + \frac{1}{3}(\beta_2-\beta_1) + \frac{1}{6}\tau^2(3-\tau) \\
&< \frac{1}{2}(\beta_2-\beta_1) + \frac{1}{2}(1-\alpha)\tau.
\end{aligned}$$

Using $\beta = 1-\alpha-\gamma$, the former inequality is equivalent to

$$\tau^2(3-\tau) - 3(1-\alpha)\tau + \sigma^3 - 3\alpha\sigma + 1 - \gamma + \alpha < 0.$$

The left side is optimized for $\tau_0 = 1-\sqrt{\alpha}$ and $\sigma_0 = \sqrt{\alpha}$, which gives

$$3\alpha - 2\alpha\sqrt{\alpha} - 1 - 2\alpha\sqrt{\alpha} + 1 + \alpha - \gamma < 0,$$

and finally

$$\gamma > 4\alpha\left(1-\sqrt{\alpha}\right).$$

By Assumption 1, we can get $(x_0, y_0, z_0) = (x_1 2^{(\beta_2-\beta_1)n} - x_3, x_2 - x_4, q_2)$, so we have $q_2 = z_0$, and we calculate

$$p_2 = \frac{N_2}{q_2}.$$

Next, we have

$$2^{(\beta_2-\beta_1)n}p_1 = p_2 + (x_1 2^{(\beta_2-\beta_1)n} - x_3) + (x_2 - x_4)2^{(\beta_2+\gamma)n} = p_2 + y_0 + z_0 2^{(\beta_2+\gamma)n}.$$

Therefore, we can calculate p_1 and $q_1 = \frac{N_1}{p_1}$. This terminates the proof. □

4 Experimental Results

We provide some experiments to verify Assumption 1 and the correctness of our analysis.

The experiments were run on a computer configured with AMD Ryzen 5 2500U with Radeon Vega Mobile Gfx (2.00 GHz). We selected the parameter $n = \log(N)$ using gradients, validated our theory starting from small-scale experiments, and continually increased the scale of our experiments. The results are presented in Table 3:

n	αn	βn	$\beta_1 n$	$\beta_2 n$	γn	m	$\dim(\mathcal{L})$	Time for LLL(s)	Time for Gröbner Basis(s)
200	20	40	20	30	140	6	28	1.8620	0.0033
200	20	60	20	30	140	6	28	1.8046	0.0034
500	50	100	50	75	350	6	28	3.1158	0.0043
500	50	150	50	75	300	6	28	4.23898	0.0048
1000	100	200	100	150	700	6	28	8.2277	0.0147

Table 3: Some experimental results for the GIFP.

As can be seen from Table 3, we chose various values of n, αn, βn, $\beta_1 n$, $\beta_2 n$ and γn to investigate the behavior of our proposed algorithm. For each set of parameters, we recorded the time taken by the LLL algorithm and Gröbner basis algorithm to solve the Generalized Integer Factorization Problem (GIFP).

Our experiments confirm Assumption 1 and also the efficiency of our algorithm in handling various values of n and related parameters. As the size of the problem increases, the computation time for LLL and Gröbner basis algorithms also increases. Nevertheless, our algorithm's time complexity grows moderately compared to the problem size. Therefore, we can conclude that our algorithm is suitable for practical applications in the Generalized Integer Factorization Problem (GIFP).

Besides the Generalized Implicit Factoring Problem, we also conducted experiments on a special case, called the *least-most significant bits case* (LMSBs). This case is characterized by $\beta_1 = 0$ and $\beta_2 = \beta$. The results of these experiments are outlined below

5 Conclusion and Open Problem

In this paper, we considered the Generalized Implicit Factoring Problem (GIFP), where the shared bits are not necessarily required to be located at the same positions. We proposed a lattice-based algorithm that can efficiently factor two RSA moduli, $N_1 = p_1 q_1$ and $N_2 = p_2 q_2$, in polynomial time, when the primes share a sufficient number of bits.

Our analysis shows that if p_1 and p_2 share $\gamma n > 4\alpha\left(1 - \sqrt{\alpha}\right) n$ consecutive bits, not necessarily at the same positions, then N_1 and N_2 can be factored in

n	αn	βn	γn	m	dim($\mathcal{L}$)	Time for LLL(s)	Time for Gröbner Basis(s)
256	25	75	156	5	21	1.3068	0.0029
256	25	75	156	5	21	1.2325	0.0023
256	25	75	156	6	21	1.2931	0.0023
512	50	150	212	6	28	2.0612	0.0028
512	50	150	212	6	28	2.4889	0.0086
512	50	150	212	6	28	2.0193	0.0022

Table 4: Some experimental results for the LMSBs case.

polynomial time. However, this bound is valid when p_i and q_i, $i = 1, 2$, are not assumed to have the same bit length, i.e., N_1 and N_2 are unbalanced moduli [16].

So our work raises an open question on improving the bound $4\alpha\,(1-\sqrt{\alpha})$, which would lead to better bounds for specific cases such as sharing some middle bits. It is known that the unshared bits in the Most Significant Bits (MSBs) or the Least Significant Bits (LSBs) are continuous, and only one variable is required when using variables to represent the unshared bits. This makes the MSBs or LSBs case easier to solve than the generalized case and achieves a better bound of $2\alpha\,(1-\alpha)$. However, the bound of the MSBs is not linear with the bound of the GIFP, which is unnatural. We hope that the gap between the bounds of the MSBs or LSBs and the GIFP case can be reduced.

Acknowledgement

The authors would like to thank the reviewers of SAC 2023 for their helpful comments and suggestions. Yansong Feng and Yanbin Pan were supported in part by National Key Research and Development Project (No. 2018YFA0704705), National Natural Science Foundation of China (No. 62032009, 12226006) and Innovation Program for Quantum Science and Technology under Grant 2021ZD0302902.

References

1. Ajtai, M.: The shortest vector problem in L2 is NP-hard for randomized reductions (extended abstract). In: Symposium on the Theory of Computing (1998)
2. Bauer, A., Jaulmes, É., Lomné, V., Prouff, E., Roche, T.: Side-channel attack against RSA key generation algorithms. In: Batina, L., Robshaw, M. (eds.) Cryptographic Hardware and Embedded Systems - CHES 2014 - 16th International Workshop, Busan, South Korea, September 23-26, 2014. Proceedings. Lecture Notes in Computer Science, vol. 8731, pp. 223–241. Springer (2014). `https://doi.org/10.1007/978-3-662-44709-3_13`, `https://doi.org/10.1007/978-3-662-44709-3_13`

3. Bernstein, D.J., Chang, Y., Cheng, C., Chou, L., Heninger, N., Lange, T., van Someren, N.: Factoring RSA keys from certified smart cards: Coppersmith in the wild. In: Sako, K., Sarkar, P. (eds.) Advances in Cryptology - ASIACRYPT 2013 - 19th International Conference on the Theory and Application of Cryptology and Information Security, Bengaluru, India, December 1-5, 2013, Proceedings, Part II. Lecture Notes in Computer Science, vol. 8270, pp. 341–360. Springer (2013). https://doi.org/10.1007/978-3-642-42045-0_18, https://doi.org/10.1007/978-3-642-42045-0_18
4. Boneh, D., Durfee, G.: Cryptanalysis of RSA with private key d less than $N^{0.292}$. In: Stern, J. (ed.) Advances in Cryptology - EUROCRYPT '99, International Conference on the Theory and Application of Cryptographic Techniques, Prague, Czech Republic, May 2-6, 1999, Proceeding. Lecture Notes in Computer Science, vol. 1592, pp. 1–11. Springer (1999). https://doi.org/10.1007/3-540-48910-X_1, https://doi.org/10.1007/3-540-48910-X_1
5. Brumley, D., Boneh, D.: Remote timing attacks are practical. In: Proceedings of the 12th USENIX Security Symposium, Washington, D.C., USA, August 4-8, 2003. USENIX Association (2003), https://www.usenix.org/conference/12th-usenix-security-symposium/remote-timing-attacks-are-practical
6. Carmon, E., Seifert, J., Wool, A.: Photonic side channel attacks against RSA. In: 2017 IEEE International Symposium on Hardware Oriented Security and Trust, HOST 2017, McLean, VA, USA, May 1-5, 2017. pp. 74–78. IEEE Computer Society (2017). https://doi.org/10.1109/HST.2017.7951801, https://doi.org/10.1109/HST.2017.7951801
7. Coppersmith, D.: Finding a small root of a univariate modular equation. In: Maurer, U.M. (ed.) Advances in Cryptology - EUROCRYPT '96, International Conference on the Theory and Application of Cryptographic Techniques, Saragossa, Spain, May 12-16, 1996, Proceeding. Lecture Notes in Computer Science, vol. 1070, pp. 155–165. Springer (1996). https://doi.org/10.1007/3-540-68339-9_14, https://doi.org/10.1007/3-540-68339-9_14
8. Coppersmith, D.: Small solutions to polynomial equations, and low exponent RSA vulnerabilities. J. Cryptol. **10**(4), 233–260 (1997). https://doi.org/10.1007/s001459900030, https://doi.org/10.1007/s001459900030
9. Faugère, J., Marinier, R., Renault, G.: Implicit factoring with shared most significant and middle bits. In: Nguyen, P.Q., Pointcheval, D. (eds.) Public Key Cryptography - PKC 2010, 13th International Conference on Practice and Theory in Public Key Cryptography, Paris, France, May 26-28, 2010. Proceedings. Lecture Notes in Computer Science, vol. 6056, pp. 70–87. Springer (2010). https://doi.org/10.1007/978-3-642-13013-7_5, https://doi.org/10.1007/978-3-642-13013-7_5
10. Graham, R.L., Knuth, D.E., Patashnik, O.: Concrete Mathematics: A Foundation for Computer Science. Addison-Wesley Longman Publishing Co., Inc., USA, 2nd edn. (1994)
11. Howgrave-Graham, N.: Finding small roots of univariate modular equations revisited. In: IMA International Conference on Cryptography and Coding. pp. 131–142. Springer (1997)
12. Lenstra, A.K., Lenstra, H.W., Lovász, L.: Factoring polynomials with rational coefficients. Mathematische annalen **261**(ARTICLE), 515–534 (1982)
13. Lu, Y., Peng, L., Zhang, R., Hu, L., Lin, D.: Towards optimal bounds for implicit factorization problem. In: International Conference on Selected Areas in Cryptography. pp. 462–476. Springer (2016)

14. May, A.: New RSA vulnerabilities using lattice reduction methods. Ph.D. thesis, University of Paderborn (2003), http://ubdata.uni-paderborn.de/ediss/17/2003/may/disserta.pdf
15. May, A., Ritzenhofen, M.: Implicit factoring: On polynomial time factoring given only an implicit hint. In: Jarecki, S., Tsudik, G. (eds.) Public Key Cryptography - PKC 2009, 12th International Conference on Practice and Theory in Public Key Cryptography, Irvine, CA, USA, March 18-20, 2009. Proceedings. Lecture Notes in Computer Science, vol. 5443, pp. 1–14. Springer (2009). https://doi.org/10.1007/978-3-642-00468-1_1, https://doi.org/10.1007/978-3-642-00468-1_1
16. Nitaj, A., Ariffin, M.R.K.: Implicit factorization of unbalanced RSA moduli. IACR Cryptol. ePrint Arch. p. 548 (2014), http://eprint.iacr.org/2014/548
17. Peng, L., Hu, L., Lu, Y., Huang, Z., Xu, J.: Implicit factorization of RSA moduli revisited (short paper). In: Tanaka, K., Suga, Y. (eds.) Advances in Information and Computer Security - 10th International Workshop on Security, IWSEC 2015, Nara, Japan, August 26-28, 2015, Proceedings. Lecture Notes in Computer Science, vol. 9241, pp. 67–76. Springer (2015). https://doi.org/10.1007/978-3-319-22425-1_5, https://doi.org/10.1007/978-3-319-22425-1_5
18. Rivest, R.L., Shamir, A., Adleman, L.M.: Cryptographic communications system and method (Sep 20 1983), uS Patent 4,405,829
19. Sarkar, S., Maitra, S.: Further results on implicit factoring in polynomial time. Adv. Math. Commun. **3**(2), 205–217 (2009). https://doi.org/10.3934/amc.2009.3.205, https://doi.org/10.3934/amc.2009.3.205
20. Sarkar, S., Maitra, S.: Some applications of lattice based root finding techniques. Adv. Math. Commun. **4**(4), 519–531 (2010). https://doi.org/10.3934/amc.2010.4.519, https://doi.org/10.3934/amc.2010.4.519
21. Sarkar, S., Maitra, S.: Approximate integer common divisor problem relates to implicit factorization. IEEE Trans. Inf. Theory **57**(6), 4002–4013 (2011). https://doi.org/10.1109/TIT.2011.2137270, https://doi.org/10.1109/TIT.2011.2137270
22. Wang, S., Qu, L., Li, C., Fu, S.: A better bound for implicit factorization problem with shared middle bits. Sci. China Inf. Sci. **61**(3), 032109:1–032109:10 (2018). https://doi.org/10.1007/s11432-017-9176-5, https://doi.org/10.1007/s11432-017-9176-5

A Details of calculations in Section 3.2

In this appendix, we present the details of calculations for the quantities e_X, e_Y, e_Z, e_W, e_N, and e_M used in Section 3.2. We begin by a lemma that will be easily proven by induction. This lemma is well-known and can be found in many textbooks and references on combinatorics and discrete mathematics, such as Table 174 on page 174 in [10].

Lemma 1. *The equation $\sum_{i=0}^{n}\binom{i}{2}=\binom{n+1}{3}$ holds for any integer n.*

Moving on, we provide the calculations for e_X as:

$$
\begin{aligned}
e_X &= \sum_{i=0}^{m}\sum_{j=0}^{m-i} j = \sum_{i=0}^{m}\binom{m-i+1}{2} = \sum_{i=0}^{m}\binom{i+1}{2} \\
&= \binom{m+2}{3} = \frac{1}{6}m(m+1)(m+2).
\end{aligned}
$$

The calculation of e_Y and e_W are the same as e_X.
Next, we provide the calculation for e_Z:

$$
\begin{aligned}
e_Z &= \sum_{i=0}^{m}\sum_{j=0}^{m-i}\max\{i+j-s,0\} \\
&= \sum_{t=s+1}^{m}\sum_{j=0}^{t}(t-s) \quad (\text{Let } t=i+j) \\
&= \sum_{t=s+1}^{m}(t-s)(t+1) \\
&= \sum_{t=0}^{m}(t-s)(t+1) - \sum_{t=0}^{s}(t-s)(t+1) \\
&= \sum_{t=0}^{m}t(t+1) - \sum_{t=0}^{m}s(t+1) - \sum_{t=0}^{s}t(t+1) + \sum_{t=0}^{s}s(t+1) \\
&= 2\sum_{t=0}^{m}\binom{t+1}{2} - s\sum_{t=0}^{m}(t+1) - 2\sum_{t=0}^{s}\binom{t+1}{2} + s\sum_{t=0}^{s}(t+1) \\
&= 2\binom{m+2}{3} - s\binom{m+2}{2} + \frac{1}{6}\binom{s+2}{3} \\
&= \frac{1}{3}m(m+1)(m+2) + \frac{1}{6}s(s+1)(s+2) - \frac{1}{2}s(m+1)(m+2).
\end{aligned}
$$

Furthermore, we provide the calculation for e_N:

$$
\begin{aligned}
e_N &= \sum_{i=0}^{t}\sum_{j=0}^{m-i}(t-i) = \sum_{i=0}^{t}(t-i)(m-i+1) = \sum_{i=0}^{t}(t-i)(m+2-i-1) \\
&= (m+2)\sum_{i=0}^{t}(t-i) - \sum_{i=0}^{t}(t-i)(i+1) = (m+2)\binom{t+1}{2} - \sum_{i=0}^{t}t(i+1) + \sum_{i=0}^{t}i(i+1) \\
&= (m+2)\binom{t+1}{2} - t\binom{t+2}{2} + \sum_{i=0}^{t}2\binom{i+1}{2} = (m+2)\binom{t+1}{2} - t\binom{t+2}{2} + 2\binom{t+2}{3} \\
&= \frac{1}{6}t(t+1)(3m-t+4).
\end{aligned}
$$

Finally, we provide the calculation for e_M:

$$
\begin{aligned}
e_M &= \sum_{i=0}^{m}\sum_{j=0}^{m-i}(m-i) = \sum_{i=0}^{m}(m-i+1)(m-i) = \sum_{i=0}^{m} 2\binom{m-i+1}{2} \\
&= \sum_{i=0}^{m} 2\binom{i+1}{2} = 2\binom{m+2}{3} = \frac{1}{3}m(m+1)(m+2).
\end{aligned}
$$