Two Applications of finding Approximate Common Divisor

Santanu Sarkar and Subhamoy Maitra

Applied Statistics Unit, Indian Statistical Institute, 203 B. T. Road, Kolkata 700 108, India.
{santanu_r, subho}@isical.ac.in

Abstract. In CaLC 2001, Howgrave-Graham proposed a method to find the Greatest Common Divisor (GCD) of two large integers when one of the integers is exactly known and the other one is known approximately. In this paper, we present two applications of the technique.
The first one is as follows. Consider $N = pq$, where p, q are large primes and $p > q$. The term $q^{-1} \bmod p$ is stored as a part of the secret key in PKCS #1 to expedite the decryption in CRT-RSA. Using lattice based technique, we show that factoring N is deterministic polynomial time equivalent to finding $q^{-1} \bmod p$. We are not aware of any earlier or trivial proof of this result.
Next, we consider the problem of finding smooth integers in a short interval as studied by Boneh in STOC 2000. We find slightly improved result using the idea of Howgrave-Graham, and it is different from the method proposed by Boneh.

Keywords: CRT-RSA, Greatest Common Divisor, Factorization, Integer Approximations, Lattice, LLL, RSA, Smooth Integers.

1 Introduction

It is well known that given two large integers a, b $(a > b)$, one can calculate the GCD efficiently in $O(\log^3 a)$ time. In [HOW01], Howgrave-Graham has shown that it is possible to calculate the GCD efficiently when some approximations of a, b are available. This problem was referred as "approximate common divisors". Using the strategy of [HOW01], Coron and May [COR07] proved the deterministic polynomial time equivalence of computing the RSA secret key and factoring. In this paper we present two other interesting applications of the technique presented in [HOW01].

First, we use the idea of [HOW01] to prove that factoring N is deterministic polynomial time equivalent to finding $q^{-1} \bmod p$ when $p > q$ or p, q are of same bit size. In the presentation of a recent paper [HEN09] at Crypto 2009, it has been asked how one can use $q^{-1} \bmod p$ towards factorization of N as $q^{-1} \bmod p$ is stored as a part of the secret key in PKCS #1 [PKCS].

In this direction, let us briefly explain RSA [RSA78] first. One needs to generate two large primes p, q, with (in general) $q < p < 2q$. Then we have $N = pq$ and $\phi(N) = (p-1)(q-1)$. Further, e, d are identified such that $ed = 1 + k\phi(N)$, $k \geq 1$. N, e are publicly available and the plaintext $M \in \mathbb{Z}_N$ is encrypted as $C = M^e \bmod N$. The secret key d is required to decrypt the ciphertext as $M = C^d \bmod N$.

To make the decryption process faster, the Chinese Remainder Theorem has been exploited and the model is well known as CRT-RSA [QUI82,WIE90]. The encryption technique

is same as RSA, but the decryption process is little different. Instead of one decryption exponent as in standard RSA, two decryption exponents (d_p, d_q) are required in this case, where $d_p \equiv d \bmod (p-1)$ and $d_q \equiv d \bmod (q-1)$. To decrypt the ciphertext C, one needs to calculate both $C_p \equiv C^{d_p} \bmod p$ and $C_q \equiv C^{d_q} \bmod q$. From C_p, C_q one can get the plaintext M by the application of CRT using $q^{-1} \bmod p$. This is the reason, $q^{-1} \bmod p$ is stored in the secret key part of PKCS #1 [PKCS].

One may be tempted to consider the following method to factorize N from the knowledge of $q^{-1} \bmod p$, which does not work. Consider $q_1 = q^{-1} \bmod p$. Now one can easily calculate $q_2 = q_1^{-1} \bmod N$ easily, as N is known. Now $q_2 q_1 - 1$ is divisible by N and hence $q_2 q_1 - 1$ is divisible by p. Thus, $q_2 \equiv q_1^{-1} \bmod p \equiv q \bmod p$. If q_2 would have been less than p, then $q_2 = q$ and the factorization will be immediate. However, in general, q_2 is of $O(N)$ and not less than p. Thus this method does not work and we need to look for a lattice based strategy which we explain in Section 2.

Next we consider the problem of finding smooth integers in a small interval [BON00]. Following [BON00], let us define two notions of smoothness.

Definition 1.

- *An integer N is called B smooth if N has no prime divisor greater than B.*
- *An integer N is called strongly B smooth if N is B smooth and p^m can not divide N for any m for which $p^m > B$.*

Finding smooth numbers is important for application in the well known factorization algorithms such as quadratic sieve [POM84] and number field sieve [LEN93]. We study the results of [BON00] and show that similar kinds of results could be achieved using a different strategy following the idea of [HOW01]. This is presented in Section 3.

2 Equivalence of finding $q^{-1} \bmod p$ and factorization

For our purpose we need the following two results. We first state the following one due to Howgrave-Graham [HOW97].

Lemma 1. *Let $h(x) \in \mathbb{Z}[x]$ be the sum of at most ω monomials. Suppose that $h(x^{(0)}) \equiv 0 \bmod N^m$ where $|x^{(0)}| \leq X$ and $||h(xX)|| < \frac{N^m}{\sqrt{\omega}}$. Then $h(x^{(0)}) = 0$.*

We also note that the basis vectors of an LLL-reduced basis fulfill the following property [LLL82].

Lemma 2. *Let L be an integer lattice of dimension ω. Given any basis of L, the LLL algorithm outputs a reduced basis $\{v_1, \ldots, v_\omega\}$ with $||v_1|| \leq 2^{\frac{(\omega-1)}{4}} det(L)^{\frac{1}{\omega}}$ in polynomial time of dimension ω and the bit size of the entries in L.*

Now we come to our main result.

Theorem 1. *Assume $N = pq$, where p, q are primes and $p \approx N^\gamma$. Suppose an approximation p_0 of p is known such that $|p - p_0| < N^\beta$. Given $q^{-1} \bmod p$, one can factor N deterministically in poly$(\log N)$ time when $\beta - 2\gamma^2 < 0$.*

Proof. Let $q_1 = q^{-1} \bmod p$. So we can write $qq_1 = 1 + k_1 p$ for some positive integer k_1. Multiplying both sides by p, we get $q_1 N = p + k_1 p^2$. That is, we have $q_1 N - p = k_1 p^2$. Let $x_0 = p - p_0$. Thus, we have $q_1 N - p_0 - x_0 = k_1 p^2$. Also we have $N^2 = p^2 q^2$. Our goal is to recover x_0 from $q_1 N - p_0$ and N^2.

Note that p^2 is the GCD of $q_1 N - p_0 - x_0$ and N^2. In this case $q_1 N - p_0$ and N^2 is known, i.e., one term N^2 is exactly known and the other term $q_1 N - p_0 - x_0$ is approximately known. This is exactly the Partially Approximate Common Divisor Problem (PACDP) [HOW01] and we follow a similar technique to solve this as explained below. This will provide the error term $-x_0$, which added to the approximation $q_1 N - p_0$, gives the exact term $q_1 N - p_0 - x_0$.

Take $X = N^\beta$ as an upper bound of x_0. Then we consider the shift polynomials

$$g_{ij}(x) = x^i (q_1 N - p_0 + x)^j N^{2(m-j)} \quad \text{for } i = 0,\ 0 \le j \le m \text{ and } j = m,\ 1 \le i \le t, \tag{1}$$

where m, t are fixed non-negative integers. Clearly, $g_{ij}(-x_0) \equiv 0 \bmod (p^{2m})$.

We construct the lattice L spanned by the coefficient vectors of the polynomials $g_{ij}(xX)$ in (1). One can check that the dimension of the lattice L is $\omega = m+t+1$ and the determinant of L is

$$det(L) = X^{\frac{(m+t)(m+t+1)}{2}} N^{2\frac{m(m+1)}{2}} = X^{\frac{(m+t)(m+t+1)}{2}} N^{m(m+1)}. \tag{2}$$

Using Lattice reduction on L by LLL algorithm [LLL82], one can find a non-zero vector b whose norm $||b||$ satisfies $||b|| \le 2^{\frac{\omega-1}{4}} (det(L))^{\frac{1}{\omega}}$. The vector b is the coefficient vector of the polynomial $h(xX)$ with $||h(xX)|| = ||b||$, where $h(x)$ is the integer linear combination of the polynomials $g_{ij}(x)$. Hence $h(-x_0) \equiv 0 \bmod (p^{2m})$. To apply Lemma 1 and Lemma 2 for finding the integer root of $h(x)$, we need

$$2^{\frac{\omega-1}{4}} (det(L))^{\frac{1}{\omega}} < \frac{p^{2m}}{\sqrt{\omega}}. \tag{3}$$

Neglecting small constant terms, we can rewrite (3) as $det(L) < p^{2m\omega}$. Substituting the expression of $det(L)$ from (2) and using $X = N^\beta, p \approx N^\gamma$ we get

$$\frac{(m+t)(m+t+1)}{2}\beta + m(m+1) < 2m(m+t+1)\gamma. \tag{4}$$

Let $t = \tau m$. Then neglecting the terms of $o(m^2)$ we can rewrite (4) as

$$\frac{\tau^2 \beta}{2} + (\beta - 2\gamma)\tau + \frac{\beta}{2} - 2\gamma + 1 < 0. \tag{5}$$

Now, the optimal value of τ to minimize the left hand side of (5) is $\frac{2\gamma-\beta}{\beta}$. Putting this optimal value in (5), we get $\beta - 2\gamma^2 < 0$.

Our strategy uses LLL [LLL82] algorithm to find $h(x)$ and then calculates the integer root of $h(x)$. Both these steps are deterministic polynomial time in $\log N$. Thus the result. $\square$

Corollary 1. *Factoring N is deterministic polynomial time equivalent to finding $q^{-1} \bmod p$, where $N = pq$ and $p > q$.*

Proof. When no approximation of p is given, then β in the Theorem 1 is equal to γ. Putting $\beta = \gamma$ in the condition $\beta - 2\gamma^2 < 0$, we get $\gamma > \frac{1}{2}$. This requirement forces the condition that $p > q$. Also, it is trivial to note that if the factorization of N is known then one can efficiently compute $q^{-1} \bmod p$. Thus the proof. □

Corollary 2. *Factoring N is deterministic polynomial time equivalent to finding $q^{-1} \bmod p$, where $N = pq$ and p, q are of same bit size.*

Proof. The proof of the case $p > q$ is already taken care in Corollary 1. Now consider $q > p$. When p, q are of same bit size and $p < q$, then $p < q < 2p$, i.e., $\sqrt{\frac{N}{2}} < p < \sqrt{N}$ and $\sqrt{N} < q < \sqrt{2N}$.

Now if we take $p_0 = \sqrt{N}$ then $|p - p_0| < (1 - \frac{1}{\sqrt{2}})\sqrt{N} < \frac{N^{\frac{1}{2}}}{2} = N^{\frac{1}{2} - \frac{\log 2}{\log N}}$. Also $p > N^{\frac{1}{2} - \frac{\log 2}{2 \log N}}$. So in this case we can take $\beta = \frac{1}{2} - \frac{\log 2}{\log N}$ and $\gamma > \frac{1}{2} - \frac{\log 2}{2 \log N}$. Thus, $\beta - 2\gamma^2 < \frac{1}{2} - \frac{\log 2}{\log N} - 2(\frac{1}{2} - \frac{\log 2}{2 \log N})^2 = -\frac{\log^2 2}{2 \log^2 N} < 0$. Hence in this situation one can factor N following Theorem 1. □

It needs to be studied how the situation can be tackled when p is significantly smaller than q.

Now let us describe the experimental result. We have implemented the program in SAGE 4.1 over Linux Ubuntu 8.10 on a laptop with Dual CORE Intel(R) Pentium(R) D CPU 1.83 GHz, 2 GB RAM and 2 MB Cache. Note that our result in Theorem 1 holds when the lattice dimension approaches to infinity. Since in practice we use finite lattice dimension, we may not reach the bound presented in Theorem 1. For experiments, we consider that small amount of Most Significant Bits (MSBs) of p is known. In Table 1, we provide some practical results. In the first three experiments, we take N as 1000-bit integer with p, q of the same bit size and $p > q$. Then in the next three experiments, we swapped p, q, i.e., q becomes larger than p. Given $q^{-1} \bmod p$, we could successfully recover p in all the cases.

p ? q	# MSBs of p known	Lattice Parameters (m, t)	Lattice Dimension	Time (in sec.)
$p > q$	46	(5,5)	11	1.41
$p > q$	24	(10,10)	21	66.33
$p > q$	20	(11,11)	23	119.72
$q > p$	47	(5,5)	11	1.42
$q > p$	24	(10,10)	21	66.56
$q > p$	20	(11,11)	23	120.00

Table 1. Experimental results following Theorem 1.

3 Finding smooth integers in a short interval

Let us denote the n-th prime by p_n, e.g., $p_1 = 2, p_2 = 3$ and so on. Suppose we want to find a strongly B smooth integer (as written in Definition 1) N in the interval $[U, V]$.

Now let us present our result.

Theorem 2. *Let $S = \prod_{i=1}^{n} p_i^{a_i}$ where $a_i = \lfloor \frac{\log B}{\log p_i} \rfloor$ and $p_1, \ldots, p_n$ are all distinct primes not exceeding B. Let $I = [U, V]$. One can find all strongly B smooth integers $N \in I$ for which $gcd(N, S) > d$ in poly$(\log S)$ time when $|I| < 2d^{\frac{\log d}{\log S}}$ and $d < 2V$.*

Proof. We will try to find N such that $\gcd(N, S) > d$. Let us take take $a_0 = \lfloor \frac{U+V}{2} \rfloor$. We consider a_0 as an approximation of N. Thus we will try to find the GCD of S, N, by knowing exactly S and some approximation of N, which is a_0 (but N is not known). Here we follow the idea of solving the Partially Approximate Common Divisor Problem (PACDP) as explained in [HOW01].

Let $x_0 = N - a_0$. We want to calculate x_0 from a_0, S. Assume $X = d^\beta$ is an upper bound of x_0. Let $S = d^\delta$. Using the same approach as in the proof of Theorem 1, we get the condition as

$$\frac{(m+t)(m+t+1)}{2}\beta + \frac{m(m+1)}{2}\delta < m(m+t+1). \tag{6}$$

Let $t = \tau m$. Then neglecting the terms of $o(m^2)$ we can rewrite (6) as

$$\frac{\beta}{2}\tau^2 + (\beta - 1)\tau + \frac{\beta}{2} + \frac{\delta}{2} - 1 < 0. \tag{7}$$

Now, the optimal value of τ to minimize the left hand side of (7) is $\frac{1-\beta}{\beta}$. Putting this optimal value in (7), we get $\beta < \frac{1}{\delta}$. Now $\delta = \frac{\log S}{\log d}$. So x_0 should be less than $d^{\frac{\log d}{\log S}}$.

Thus, we get x_0 and hence N in poly$(\log S)$ time. As, $V < 2d$, we have $N < 2d$ (since $U \leq N \leq V$). When $\gcd(N, S) > d$, then $\gcd(N, S) = N$ as $N < 2d$. Hence N divides S, i.e., N is strongly B smooth. □

B	$\log_2 d$	$\log_2(V-U)$	LD (Our), Time (sec.)	LD ([BON00]), Time (sec.)
1000	450	130	36, 15.51	32, 21.33
1000	496	156	29, 3.77	26, 8.06
1000	496	161	45, 36.88	41, 64.71

Table 2. Comparison of our experimental results with that of [BON00]. We have implemented the ideas of [BON00] for experimental comparison. LD denotes Lattice Dimension.

Asymptotically, our result is 8 times better than that of [BON00, Theorem 3.1], as that bound was $|I| < \frac{1}{4}d^{\frac{\log d}{\log S}}$. Below we present a few experimental results, where we find improved outcomes (in terms of execution time) using our strategy than that of [BON00]. One should also note, that the method of [BON00] requires the implementation of CRT on several primes, which is not included in the time mentioned in Table 2. Our strategy using the idea of [HOW01] does not require such computation.

4 Conclusion

In this paper we use the method of finding approximate common divisor, as proposed in [HOW01], for approaching two problems. The first one is to show the deterministic polynomial time equivalence between factorization of the RSA moduli and finding $q^{-1} \bmod p$. To the best of our knowledge, this equivalence has not been studied earlier. We also do not find any trivial method to prove it. Next, we revisit the problem of finding smooth integers in an interval as explained in [BON00]. We find slightly improved results than that of [BON00] using the technique presented by [HOW01]. The work of [HOW01] has earlier been exploited in [COR07] to prove the deterministic polynomial time equivalence of computing the RSA secret key and factoring. We present two more important applications of the work of [HOW01] here. Finding more applications of this work [HOW01] could be an interesting area to study in lattice based approaches for number theoretic problems.

References

[BON00] D. Boneh. Finding smooth integers in short intervals using CRT decoding. Proceedings of STOC 2000, pages 265–272, 2000.

[COP97] D. Coppersmith. Small Solutions to Polynomial Equations and Low Exponent Vulnerabilities. Journal of Cryptology, 10(4):223–260, 1997.

[COR07] J. -S. Coron and A. May. Deterministic polynomial-time equivalence of computing the RSA secret key and factoring. Journal of Cryptology, 20(1):39–50, 2007.

[HEN09] N. Heninger and H. Shacham. Reconstructing RSA Private Keys from Random Key Bits. Proceedings of Crypto 2009, Lecture Notes in Computer Science, Volume 5677, pages 1–17, Springer, 2009. The presentation is available at http://www.iacr.org/conferences/crypto2009/slides/p001-rsa-keys.pdf

[HOW97] N. Howgrave-Graham. Finding Small Roots of Univariate Modular Equations Revisited. Proceedings of Cryptography and Coding, Lecture Notes in Computer Science, Volume 1355, pages 131–142, Springer, 1997.

[HOW01] N. Howgrave-Graham. Approximate integer common divisors. Proceedings of CALC 2001, Lecture Notes in Computer Science, Volume 2146, pages 51–66, Springer, 2001.

[LLL82] A. K. Lenstra, H. W. Lenstra and L. Lovász. Factoring Polynomials with Rational Coefficients. Mathematische Annalen, 261:513–534, 1982.

[LEN93] A. K. Lenstra and H. W. Jr. Lenstra. The Development of the Number Field Sieve. Springer-Verlag, 1993.

[PKCS] http://www.rsa.com/rsalabs/node.asp?id=2125

[POM84] C. Pomerance. The Quadratic Sieve Factoring Algorithm. Proceedings of Eurocrypt 1984, Lecture Notes in Computer Science, Volume 209, pages 169–182, 1985.

[QUI82] J. -J. Quisquater and C. Couvreur. Fast decipherment algorithm for RSA public-key cryptosystem. Electronic Letters, volume 18, pages 905–907, 1982.

[RSA78] R. L. Rivest, A. Shamir and L. Adleman. A Method for Obtaining Digital Signatures and Public Key Cryptosystems. Communications of ACM, 21(2):158–164, February 1978.

[WIE90] M. Wiener. Cryptanalysis of Short RSA Secret Exponents. IEEE Transactions on Information Theory, 36(3):553–558, 1990.