Babaï Round-Off CVP method in RNS Application to Lattice based cryptographic protocols
Jean-Claude Bajard, Julien Eynard, Nabil Merkiche, Thomas Plantard

To cite this version:

HAL Id: hal-01098802
https://hal.sorbonne-universite.fr/hal-01098802
Submitted on 29 Jan 2015
Abstract—Lattice based cryptography is claimed as a serious candidate for post quantum cryptography, it recently became an essential tool of modern cryptography. Nevertheless, if lattice based cryptography has made theoretical progresses, its chances to be adopted in practice are still low due to the cost of the computation. If some approaches like RSA and ECC have been strongly optimized - in particular their core arithmetic operations, the modular multiplication and/or the modular exponentiation - lattice based cryptography has not been arithmetically improved. This paper proposes to fill the gap with a new approach using Residue Number Systems, RNS, for one of the core arithmetic operation of lattice based cryptography: namely solving the Closest Vector Problem (CVP).

I. INTRODUCTION

The cryptography based on lattices appeared at the beginning of this century with initial propositions like GGH [10] and NTRU [12].

In few years, due to some properties of the lattices, some powerful cryptographic tools have been proposed for the first time: fully homomorphic encryption, multi-linear map and indistinguishability obfuscation [8]. Despite numerous attacks against one the historical propositions, countermeasure after countermeasure, these systems are still available [6]. Even after numerous evolutions, they stay based on some simple proposals where the encryption is obtained by adding an "error" to a vector of a lattice. This error represents the original message, and the vector obtained the ciphered one. All the security is based on the difficulty to reduce the public basis of the lattice in a Lovász reduced basis in which the Babaï algorithms can be performed [1].

Some recent approaches propose to use an oracle which gives some approximated closest vectors [14], [9], [18], and a Learning with Error method to find the closest vector. Nevertheless, they are not still sufficiently efficient in practice. Thus, an efficient computation of a closest vector remains a real challenge. As the Residue Number Systems (RNS) has been proved to be efficient for other cryptographic systems [11], [5], we suggest to study in this paper their use in lattice cryptography by implementing in RNS the Babaï Round-Off CVP method.

II. ABOUT BABAÏ ROUND-OFF CVP

The main idea can be summarized in the following way. We create a lattice using a strongly reduced basis given by a matrix $G$ and we construct another basis $H = UG$, where $U$ is a unimodular matrix such that $H$ is a bad basis in terms of lattice basis reduction. $H$ can be in Hermite Normal Form [15].
The encryption mode [10], obeys the following scheme: \( c = (m + kH) \) where \( m \) is the vector message composed of zeros and ones (or of small values with respect to the Lovász conditions), \( k \) is a vector such that \( c = (c_0, 0, \ldots, 0) \) with \( c_0 \) huge, or \( c = (c_0, c_1, \ldots, c_n) \), \( c_i \) smaller. The vector \( kH \) belongs to the lattice, and is a closest vector of \( c \). In the following, we will consider that all the coefficients of \( c \) are positive, which is possible modulo a translation via a vector of the lattice. As the coefficients of \( m \) are small and \( G \) is strongly orthogonal, the message \( m \) is found using the Rounding Off algorithm of Babai [1]. This operation is given by \( m = c - \left[ cG^{-1} \right] \times G \), where \( \left[ cG^{-1} \right] \times G \) represents the closest vector of the lattice. Since \( m \) is composed of small values, it is suggested to compute \( c - \left[ cG^{-1} \right] \times G \mod \beta \) where \( \beta \) is a small number, reducing by this way the complexity of the calculus. Nevertheless, though matrix \( G \) is an integer matrix, its inverse \( G^{-1} \) is not, i.e., is rational. The operation \( \left[ cG^{-1} \right] \) must be done sufficiently precisely for obtaining a good rounding.

### III. The RNS approach of the Rounding Off Babai's algorithm

In this work, we propose for this evaluation to use RNS systems which distribute the calculus on small values in a fully parallel way for additions and multiplications [21], [20]. These representations are based on the Chinese Remainder Theorem, a number \( \alpha \) is represented by its residues \((\alpha_1, \ldots, \alpha_n)\) modulo \( m \) called the RNS base. Hence, we are able to represent all the values from 0 to \( M = \prod_{i=1}^n m_i \). In this approach we use the modular reduction proposed by P. Montgomery [16] and adapted to RNS [17], [13], [2], both for the evaluation of \( \left[ cG^{-1} \right] \times G \) and for the final reduction \( \mod \beta \).

Our first purpose is to compute the value \( \left[ cG^{-1} \right] \) in RNS. For this, we will transform this calculus in complete integer operation using that \( G' = (\det G) \times G^{-1} \) is an integer matrix when \( G \) is one integer matrix. Thus we have: \( \left[ \frac{cG'}{\det G} \right] = \left[ cG^{-1} \right] \).

In RNS, the division by \( \det G \) is possible if it is an exact one and if \( \det G \) is co-prime with the RNS Base. In this case we have,

\[
\frac{cG' - (cG' \mod \det G)}{\det G} = \frac{cG'}{\det G}.
\]

As we want to compute \( \left[ \frac{cG'}{\det \alpha} \right] \), we will compute more precisely \( \left[ \frac{cG'}{\det \alpha} + \frac{1}{2} v_1 \right] \), where \( v_1 \) is the all-one vector (i.e. \( v_1 = (1, 1, \ldots, 1) \)).

If we develop this expression, we obtain:

\[
\left[ \frac{cG'}{\det \alpha} \right] = \left[ \frac{cG'}{\det \alpha} + \frac{1}{2} v_1 \right] = \left[ 2cG' + \det G, v_1 - \left[ (2cG' + \det G, v_1) \mod (2 \det G) \right] \right] \mod (2 \det G).
\]

The most delicate operation is due to the modulo \( \mod (2 \det G) \), which requires in RNS a particular attention. The other operations can be directly implemented in RNS as is.

We note \( D_G = (2 \det G) \).

#### A. Evaluation of \([(2cG' + (\det G)v_1) \mod D_G] \) in RNS

In this part, we consider the RNS bases \( B_1 \) and \( B_2 \) with \( M_1 = \prod_{m \in B_1} m \) and \( M_2 = \prod_{m \in B_2} m \). The bases are selected such that \( D_G < M_1, M_2 \), assuming that \( D_G \) is coprime with the elements of \( B_1 \) (which is generally the case, because \( \det G \) is frequently a prime number).

The modular reduction can be done in RNS using the Montgomery algorithm [2]. The particularity of the approach is that the reduced value is obtained multiplied by a factor depending of the RNS base (in our case \( M_1^{-1} \)). When some values are fixed, \( G \) in our case, we can use precomputed values to avoid this extra final factor \( M_1^{-1} \).

Thus, we let denote by \( G'' = 2G' \times M_1 \mod D_G \) (recall that \( G'' \) is not integer, but \( G'' = (\det G)G^{-1} \) is), and \( v'' = (M_2 \times \det G)v_1 \mod D_G \).

The "PreBabaiROffrms" has two modes, the one which gives the result on \( B_1 \) and \( B_2 \), and the other without option which gives the result modulo \( \beta \) adapted to a cryptographic context.
Algorithm 1 PreBabaiROff\textsubscript{rns}(option)

**Input**: $a = c \times G' + v''$, $a \in \mathbb{Z}^n$ given in the two bases $B_1$ and $B_2$, $|a|_\infty < M_1 \times D_G$, $2D_G < M_2$, all the values concerned by $G$ are considered as precomputed.

**Output**: $\left[(2cG' + (\det G)v_1) \mod D_G\right]$ in $B_1$ and $B_2$ if (option = rns), else $\mod \beta$.

1. $q_1 \leftarrow (-D_G)^{-1} \times a_1$ in $B_1$ (in other words, the evaluation is made modulo $M_1$),
2. $q_2 \leftarrow$ q\textsubscript{Extension1} from $B_1$ to $B_2$ of $q_1$,
3. $r_2 \leftarrow (a_2 + D_G \times q_2) \times M_1^{-1}$ in base $B_2$,
   hence $r_2 \equiv (2cG' + (\det G)v_1) \mod D_G$, with $|r_2|_\infty < 2D_G$.
4. Extension\textsubscript{2} of $r_2$ in the base $B_1$ if option=rns, else modulo $\beta$.

The “PreBabaiROff\textsubscript{rns}” algorithm uses the Montgomery reduction in the states 1 and 3 of the procedure. The state 1 computes $q_1$ modulo $M_1$ such that $(a_2 + D_G \times q_2)$ gives a multiple of $M_1$, thus, in state 3, the division by $M_1$ is equivalent to a multiplication by its inverse. This last operation is possible in the base $B_2$, since $M_1$ is coprime to $M_2$. Thus, base extensions are needed and correspond to states 2 and 4. Then, we obtain the value $r_2 \equiv [(2cG' + (\det G)v_1) \mod D_G]$, with $|r_2|_\infty < 2D_G$, which is converted in $B_1$ or modulo $\beta$ with respect to the option.

B. Analysis of the first extension

For Extension\textsubscript{1} we need to extend $q_1$ exactly. A first solution could be to use an intermediate representation: Mixed Radix System. But it is costly. So we can replace steps 2 and 3 by an approach where we use an extra modulo $\hat{m}$.

We recall that $D_G = (2 \det G)$.

In step 1, $q_2 = q_1 + \alpha M_1$, thus

$r_2 = (a_2 + D_G \times q_2) \times M_1^{-1}$

$= (a_2 + D_G \times (q_1 + \alpha M_1)) \times M_1^{-1}$

$= (a_2 + D_G \times q_1) \times M_1^{-1} + \alpha D_G$.

Hence, $r_2 < (2 + \alpha)D_G$, we need to reduce it a second time. For that we use the extra modulo $\hat{m}$ and we apply a second Montgomery reduction computing $\tilde{q}$, thus

$r_2' \equiv (a_2 \times M_1^{-1}) \times \hat{m}^{-1} \mod D_G$ with $r_2' < 2D_G$

when $\hat{m} > |B_1| + 1 \geq 2 + \alpha$.

Algorithm 2 Extension\textsubscript{1}Bis

**Input**: $a_2$ defined on $B_2$ and $a_\hat{m} = a \mod \hat{m}$.

**Output**: $q_2$ the extension of $q_1$ in $B_2$ with $q_2 < M_1$.

1. $q_2 \leftarrow \sum_{m \in B_1} q_{1,m} \frac{M_1}{m} \frac{1}{M_1} \mod \hat{m}$

2. $r_{\hat{m}} \leftarrow \sum_{m \in B_1} \left(q_{1,m} \frac{M_1}{m} \frac{1}{M_1} \frac{M_1}{\hat{m}} \mod \hat{m}\right)$

3. $q' \leftarrow (-D_G)^{-1} r_{\hat{m}} \mod \hat{m}$

4. Extension of $\tilde{q}$ in $B_2$ is just a duplication if $\tilde{m}$ smaller than all the elements of $B_2$.

5. $r_{\hat{m}}' \leftarrow (r_2 + D_G \times \tilde{q}) \times M_1^{-1} \mod \hat{m}$.

We replace $M_1$ by $M'_1 = M_1 \times \hat{m}$. Hence, the precomputed values become

$G'' = 2G' \times M'_1 \mod D_G$

and $v'' = (M'_1 \times \det G)v_1 \mod D_G$.

C. Analysis of the second extension

For the second base extension, we can use an extra modulo $\hat{m}$ with a Shenoy-Kumaresan approach [19]. But in this case, we cannot extract any information about the comparison of $r_2'$ with $D_G$. Thus, we obtain $r_2' \equiv (2cG' + (\det G)v_1) \mod D_G$ or $[(2cG' + (\det G)v_1) \mod D_G] + D_G$ which is not satisfying for our purpose.

Hence, the second extension can be done in MRS which is a positional number system. In this case, during the conversion, a comparison with $D_G$ is possible and if necessary we subtract $D_G$.

D. Complete “Rounding Off” Closest Vector in RNS

Now, we come back to our problem which is to compute a closest vector with round-off formula: $cG^{-1} \times G$. First we give a new version of the PreBabaiROff\textsubscript{rns} which includes the results of the extensions analysis.

NewPreBabaiROff\textsubscript{rns} algorithm gives $[cG^{-1}]$ in the two bases $B_1$ and $B_2$ or modulo $\beta$ with respect to the option, with as input $a = c \times G' + v''$ in which $G'' = 2G' \times M'_1 \mod D_G$ and
Algorithm 3 NewPreBabaiROff_rns(option)

Input: $a = c \times G^v + v^n$, $a \in \mathbb{Z}^n$ given in the bases $B_1$, $B_2$ and $\hat{m}$, $|a|_\infty < M_1 \times D_G$, $2D_G < M_2$, all the values concerned by $G$ are considered as precomputed.

Output: $[(2cG' + (\det G)v_1) \mod D_G]$ in $B_1$ and $B_2$ if (option = $rns$), else $\mod \beta$.

1. $q_1 \leftarrow (-D_G)^{-1} \times a_1$ in $B_1$ (in other words, the evaluation is made modulo $M_1$),
2. $r_2' \leftarrow \text{Extension}_1\text{Bis}(q_1, B_1, B_2, \hat{m})$,
3. $\hat{r}_2 \leftarrow r_2'$ conversion in mixed radix,
4. Comparison of $\hat{r}_2$ with $(2\det G)$,
5. Extension of $\hat{r}_2$ in the base $B_1$ if $rns$ else modulo $\beta$,

$v^n = (M_1' \times \det G)v_1 \mod D_G$. Thus we propose the following procedure for computing the Closest Vector $\lfloor cG^{-1} \rfloor \times G$.

Algorithm 4 BabaiROff_rns(option)

Input: $c \in \mathbb{Z}^n$ the ciphertext given in $B_1$, $B_2$ and $\hat{m}$, all the values concerned by $G$ are considered as precomputed.

Output: $r = \left\lfloor \frac{cG'}{\det G} \right\rfloor = \left\lfloor \frac{cG'}{\det G} + \frac{1}{2}v_1 \right\rfloor$, if (option = $rns$) then in the two RNS bases $B_1$ and $B_2$, else modulo $\beta$ (that is true for all the calculus of this procedure).

1. $a \leftarrow c \times G^n + v^n$ in $B_1$, $B_2$ and $\hat{m}$,
2. $b \leftarrow \text{NewPreBabaiROff_rns}(a, B_1, B_2, \hat{m})$,
3. $r \leftarrow (a - b)(2\det G)^{-1}$ in $B_1$, $B_2$ and $\hat{m}$.

IV. DISCUSSIONS

One interesting feature of this approach comes from the formulae of Extension$_1\text{Bis}$ which can be decomposed in matrix products where some fast algorithms like the Strassen one can be used. The main drawback of the current version is due to the necessity to compute exactly the result of the NewPreBabaiROff_rns. The solution of using MRS is not efficient, it would be more interesting to use a Shenoy-Kumaresan approach where the formulae are similar to the ones of Extension$_1\text{Bis}$.

REFERENCES