Predicting Tours and Probabilistic Simulation for BKZ Lattice Reduction Algorithm

Md Mokammel Haque,‡,∗ Josef Pieprzyk, and Asaduzzaman

1Department of Computer Science and Engineering, Chittagong University of Engineering and Technology
Chittagong 4349, Bangladesh
2School of Electrical Engineering and Computer Science, Queensland University of Technology
Brisbane, QLD 4000, Australia
∗malin405@yahoo.com

Abstract—We investigate the terminating concept of BKZ reduction first introduced by Hanrot et al. [Crypto’11] and make extensive experiments to predict the number of tours necessary to obtain the best possible trade-off between reduction time and quality. Then, we improve Buchmann and Lindner’s result [Indocrypt’09] to find sub-lattice collision in SWIFFT. We illustrate that further improvement in time is possible through special setting of SWIFFT parameters and also through the combination of different reduction parameters adaptively. Our contribution also include a probabilistic simulation approach top-up deterministic simulation described by Chen and Nguyen [Asiacrypt’11] that can able to predict the Gram-Schmidt norms more accurately for large block sizes.

Index Terms—Cryptosystem, Lattice reduction, BKZ, SWIFFT, Hermite factor.

I. INTRODUCTION

The Lattices we are interested in are integer or point lattices; discrete subgroups of $\mathbb{R}^n$. A lattice $L$ is defined as all integer linear combinations of some linearly independent vectors $(b_1, ..., b_n)$ of the form: $\sum_{i=1}^{n} x_i \cdot b_i$, where $x_i \in \mathbb{Z}$ for all $i$. The vectors $(b_1, ..., b_n)$ are called a basis of the lattice. The number of vectors $n$ is the dimension of the lattice $L$.

The goal of lattice reduction is to find a “good” basis for the applications which is usually the one that contains short vectors. Clearly, it is an interesting problem to find a short basis or even to try to find a shortest basis. The SVP or Shortest Vector Problem is the most common and fundamental algorithmic lattice problem on which many cryptosystems rely. It is an NP-hard problem under randomized reductions [1].

Lattice problems are considered to be building blocks in many public-key cryptographic constructions. The most significant examples of cryptographic constructions that use lattices include: public key encryption based on knapsack problem; variant of RSA and DSA signatures schemes; encryption schemes based on LWE (Learning with Error) problems [2]; fully homomorphic cryptosystems [3]; the SWIFFT hash function [4] as well as many others. If one knows how to solve lattice problems, then one can break some cryptosystems.

Lattice reduction algorithms are one of the few exciting recent techniques in cryptography. The most popular lattice reduction algorithms are the LLL (Lenstra-Lenstra-Lovász) [5] and BKZ (Block Korkine-Zolotarev) [6] algorithms and their variants. They are called approximate algorithms; another category is known as exact algorithms includes Enumeration.

In this paper, we run experiments to predict the required number of tours (or iterations) in enumeration subroutine of aborted-BKZ reduction (also known as terminating BKZ as in [7]). We also show an approach of generating Gram-Schmidt sequence of vectors. This is called probabilistic approach different than the deterministic approach used in [8] for BKZ2.0 reduction. Finally, we perform experiments to find pseudo-collision in SWIFFT hash function (one of the contestants for SHA-3 competition) using above mentioned aborted-BKZ lattice reduction algorithm.

II. PRELIMINARIES

In this section we describe some preliminary concepts on lattice reduction.

A. Gram-Schmidt Orthogonalization

In the context of Euclidean vector spaces, Gram-Schmidt orthogonalization is used to compute an orthonormal basis i.e. to look for the unit vectors that are pairwise orthogonal in the basis. In the context of lattices it is a measure to guide the reduction, and assess its quality. For instance, any nonzero vector of the lattice generated by $B = (b_1, b_2, ..., b_n)$ must have length $\geq \min \|b_i\|$. The Gram-Schmidt orthogonalization is a well known procedure that takes any set of $n$ linearly independent vectors and creates a set of $n$ orthogonal vectors. It works by projecting each vector on the space orthogonal to the span of the previous vectors. Fig. 1 gives an illustration of this concept.

Therefore, the Gram-Schmidt Orthogonalization (GSO) can be defined as a method that can always transfer a basis $B = (b_1, ..., b_n)$ into the orthogonal basis $B^* = (b_1^*, ..., b_n^*)$ iteratively as follows:

1http://csrc.nist.gov/groups/ST/hash/sha-3
\[ b_1^* = b_1 \]

\[ b_2^* = b_2 - \mu_{2,1} b_1^*, \quad \text{where} \quad \mu_{2,1} = \langle b_2, b_1^* \rangle / \langle b_1^*, b_1^* \rangle \]

\[ b_i^* = b_i - \sum_{j=1}^{i-1} \mu_{i,j} b_j^*, \quad \text{where} \quad \mu_{i,j} = \langle b_i, b_j^* \rangle / \langle b_j^*, b_j^* \rangle, \forall 1 \leq j < i \leq n \]

We can also denote the orthogonal Gram-Schmidt vectors as \( b_i^* := \pi_i(b_i) \) for all non-negative \( i \leq n \), where \( \pi_i \) denotes the orthogonal projection over the orthogonal supplement of the linear span of \( b_1, \ldots, b_{i-1} \); \( \pi_i(b_i) \) is also known as the projection of lattice \( L \) or the projected lattice.

**B. BKZ Reduction**

Block-Korkine-Zolotarev reduction (aka BKZ reduction) is the most successful lattice reduction algorithm in practice. Schnorr and Euchner [6] introduced the following definition of BKZ reduction.

**Definition 1:** Let \( L \) be a lattice of rank \( n \) and \( k \) be an integer called the blocksize such that \( 2 \leq k < n \). A lattice basis \( B = (b_1, \ldots, b_n) \) of \( L \) is called BKZ-reduced with blocksize \( k \) and factor \( \delta \) if it is size-reduced and if \( \forall 1 \leq i \leq n \),

\[ \delta \|b_i^*\|^2 \leq \lambda_1(\pi_i(L(b_1, \ldots, b_{\min(i+k-1,n)})))^2 \]

If the blocksize \( k = 2 \), it is essentially a LLL reduction. Here, \( \lambda_1(L) \) means the first (successive) minimum of lattice \( L \).

**C. Enumeration**

Enumeration is a technique used in most lattice reduction algorithms to solve the exact SVP. If we imagine a lattice as a tree then the enumeration procedure is simply an exhaustive (depth first) search of all the possible combinations of nodes (or basis vectors) within distance \( \lambda_1(L) \). This procedure runs in exponential time with dimension \( n \), as opposed to the lattice reduction technique itself (which runs in polynomial time). The main idea dates back to the early 1980s and is attributed to Kannan [9].

**D. Random Lattice**

Random lattices are generally used for testing basis reduction algorithms. Goldstein and Mayer in [10] proposed one efficient way to generate random lattices of the following form:

\[
R_n^q = \begin{pmatrix}
q & 0 & 0 & \cdots & 0 \\
x_1 & 1 & 0 & \cdots & 0 \\
x_2 & 0 & 1 & \ddots & \vdots \\
\vdots & \ddots & \ddots & \ddots & \vdots \\
x_{n-1} & 0 & \cdots & 0 & 1
\end{pmatrix}
\]

where \( q \) (the volume of the \( n \)-dimensional lattice) is a large prime (bit length \( 10n \) is considered enough for time efficiency) and \( x_i \in \{0, \ldots, q - 1\} \) are chosen uniformly and independently. We used all full rank lattices normalized to have determinant 1 with uniform probability measure. See [11] for a method to generate random lattices and random bases for experiments.

---

**III. Predicting Tours**

During an entire reduction cycle BKZ performs number of tours where each tour is equivalent to \( (n-1) \) calls to the enumeration subroutine. Hanrot, Pujol and Stehlé in [7] first observed that reduction can be aborted or terminated long before its completion and still able to manage a very good quality. In fact, according to their analysis, BKZ reduction operates mainly in two distinct phases. In first phase exponential drop of hermite factor is seen for small number of tours whereas in second phase it drops slowly for greater number of tours on average. We revisited this scenario experimentally and found that one typical reduction cycle can be illustrated as three distinct phases as in Fig. 2. The points location will vary for randomly chosen different input basis but all three points are distinctive even for a single random instance with few exceptions.

**A. Experimental Result**

It is worth to investigate the average cost of stopping after certain phases of reduction and to estimate average amount of tours required. Our inputs consider random lattices (as in the Section II) of dimension 100 on BKZ-20 to BKZ-24 using NTL [12] package. All experiments are done in 2.67 GHz intel...
corei5 processor.

The results are plotted on Table I and Table II to see the stopping gain for point A and point B respectively. Here, \( T \) and \( \tau \) denotes the time and the number of tours required with respect to the points (B or C in case of Table I and C in case of Table II) respectively. We always find a point like B where stopping be made incur 0 loss of reduction quality \((\delta C - \delta B = 0)\). And, we save as high as 47% and as low as 27% on enumeration time based on block size. We require 29%-46% amount of tours on average to reach this point. If we stop at point A we have deficit on hermite factor constant(root-hermite factor) of about \( \approx 0.01 \) but can able to save 85%-96% on total enumeration time . The amount of tours required for this case is only about 2%-10% on average.

The above experiment gives us an idea about the trade off (between quality and time) we require.

IV. PROBABILISTIC SIMULATION

In [8] Chen and Nguyen presents a simulation algorithm to predict Gram Schmidt sequence of vectors \( \| b_i^* \|_{1 \leq i \leq n} \) for large block sizes. The main idea relies on the fact that for small block size approximately \( > 30 \) the first minimum of BKZ local blocks resembles to the random lattice of dimension equal to the block size. Then, in their simulation they consider hermite factor from Gaussian Heuristic. Precisely, using well known Minkowski bound the hermite factor becomes,

\[
\frac{2 \sqrt{h}}{V_n(1)^{1/n}} \geq 2 \sqrt{\frac{n}{\log n}} \Gamma(n/2 + 1) \tag{2}
\]

where \( h = \lambda_1(L^2)u \cdot vol(L)^{2/n} \) and \( u = 4/V_n(1)^{2/k} \) is the geometric upper bound from Minkowski on hermite constant. If we use \( h = 0.25 \), this is the Gaussian value. In fact, this is the average value of \( h \) derived form the experiment with many instances of random lattices. The distribution of \( h \) is normal for large block size. Instead of taking average value, at the start of each round in simulation algorithm we sample randomly from the density function with experimental average and standard deviation of \( h \). Then we use this sampled \( h \) to calculate hermite constant as in equation 2. This can predict the Gram-Schmidt sequence more accurately (see Fig. 3).

It suggests, a better root-hermite factor (reduction quality) corresponds to a block size can be possible. Therefore, same hermite factor can be achieved with smaller block size or reduced number of tours, thus decreasing the runtime on average.

V. PSEUDO-COLLISION IN SWIFFT

In this section we run an experiment to improve the current attack time (reported in [13]) on the SWIFFT hash function. We use the RBKZ reduction utilizing the aborted-BKZ reduction [7] with so-called sublattice attack.

A. SWIFFT Hash Function

The SWIFFT hash function is defined by the following compression function:

\[
\sum_{i=1}^{m} a_i \cdot x_i \in R
\]

where \( R = \mathbb{Z}_p[x]/(x^n + 1) \) is the ring of polynomials, \( a_i \in R (\forall i) \) are \( m \) fixed multipliers and \( x_i \in R (\forall i) \) with coefficients in \( \{0, 1\} \) are the input of length \( mn \). The two basic parameters are \( n \) and \( p \), where \( n \) is a power of 2 and \( p > 0 \) is a prime modulus. The algebraic formulation of SWIFFT is equivalent to a lattice of dimension \( mn \) and the security of such function depends on the infeasibility of finding relatively short vectors in such a lattice. Finding a collision in this function means looking for a nonzero vector \( x_i \in R \) with coefficients in \( \{-1, 0, 1\} \) such that \( \sum_{i=1}^{m} a_i \cdot x_i \equiv 0 \mod p \). In other words, if we can find a vector by reducing the lattice of Euclidean length \( \leq \sqrt{mn} \), then it is a pseudo-collision. We call it pseudo-collision because we consider a sublattice of dimension \( m' n' \leq mn \) of the full dimensional lattice.

B. Better Attack Time

The experiment by Micciancio and Regev [14] suggests that a shortest vector of length \( l = 2^{\sqrt{n \log p \log \delta}} \) could be found in an optimal dimension \( d = \sqrt{n \log p / \log \delta} \), where \( mn \) is the dimension of the lattice with modulus \( p \) and \( \delta \) is the root-Hermite factor. Again, finding a sub-lattice collision in the \( mn \) dimensional lattice is equivalent to finding a vector of length \( l = \sqrt{mn} \). In [13], Buchmann and Lindner use the above idea to find pseudo-collision for the SWIFFT compression function. They introduced a parameter generator algorithm that generates minimal possible parameters for SWIFFT required to satisfy the pseudo-collision condition. The smallest parameter set (in Table 1 of [13]) was \( m = 16, n = 64, p = 257 \), compression ratio \( \approx 2 \) with root-Hermite factor \( \delta \approx 1.0084 \) in sub-dimension \( d = 206 \). As this amount of the Hermite factor is hard for current reduction algorithms, they come up with this less hard problem with smaller \( p \) and \( \delta \) and successfully break 9 instances of this problem with the NTL implementation of BKZ, and RSR algorithm [15]. We use aborted-BKZ to break the harder 5 of these 9 instances. See the Table III for the scale-down parameters of these 5 instances.

<table>
<thead>
<tr>
<th>( m )</th>
<th>( n )</th>
<th>( p )</th>
<th>( \delta )</th>
<th>( d )</th>
</tr>
</thead>
<tbody>
<tr>
<td>16</td>
<td>64</td>
<td>45</td>
<td>1.0124</td>
<td>141</td>
</tr>
<tr>
<td>16</td>
<td>64</td>
<td>44</td>
<td>1.0121</td>
<td>144</td>
</tr>
<tr>
<td>16</td>
<td>64</td>
<td>54</td>
<td>1.0119</td>
<td>147</td>
</tr>
<tr>
<td>16</td>
<td>64</td>
<td>57</td>
<td>1.0117</td>
<td>150</td>
</tr>
<tr>
<td>16</td>
<td>64</td>
<td>61</td>
<td>1.0115</td>
<td>152</td>
</tr>
</tbody>
</table>
We search for the $\delta$ (root-HF) required from the table to break a particular instance by increasing the number of tours within a BKZ blocksize. To get average results, we use 10 random SWIFFT seeds/instances for each of these 5 instances. As soon as we found a required $\delta$ (after some tours) we stop and record time otherwise we move to the next blocksize. In Fig. 4 we plot the average runtime required for this approach comparing the original BKZ and RSR algorithms. It shows that aborted-BKZ can save a considerable amount of time comparing to BKZ and RSR.

C. Special case of $\|b_i^*\|$  

SWIFFT lattice basis can be represented by the following matrix:

$$B = \begin{pmatrix} I & 0 \\ H & p.I \end{pmatrix}$$  

(3)

where the $m \times n$ dimensional lattice is symbolized as a $m \times n \times m \times n$ matrix. $H$ is a $n \times n \cdot (m - 1)$ nega-circulant matrix in $\mathbb{Z}_p = \{0, \ldots, p-1\}; p > 0$ is a prime. $I$ is the $n \cdot (m - 1) \times n \cdot (m - 1)$ identity matrix. Right bottom is the $n \times n$ scalar matrix and right top is a $n \cdot (m - 1) \times n$ dimensional zero matrix.

According to [8], sufficiently random reduced bases have a typical shape for main algorithms like LLL and BKZ and the shape depends on the ratio $q = \|b_{i+1}^*\| / \|b_i^*\|$ of Gram-Schmidt vectors. This follows Schnorr’s Geometric Series Assumption(GSA) [15]. Experimentally we found that varying the parameter $m$ with fixed $mn$ (dimension of the lattice) we can have a special shape of $\log \|b_i^*\|$ vs. $i$ curve ($i <= mn$) (see Fig. 5) where slope $\log q$ becomes zero after some index $i$. This special shape takes significantly less reduction time than the typical shape. Experiment shows that when $m = 2$, problem becomes the hardest with only one circulant block (as in V-C) of large $n$ (as the dimension is fixed). Now, if we increase $m$ by fixing the overall dimension (that requires to decrease $n$ as well), the number of blocks of circulant matrix increases with smaller $n$ and the problem becomes easier. The reduction time is inversely proportional to the $m$.

To break all 5 instances of Table III we consider $m = 2$. With larger $m$ we can able to break all instances in better time. For example, the instance that require $\delta = 1.0124$ can be broken within $\approx 20$ seconds for $m = 2$ and for $m = 4$, it requires only $\approx 13$ seconds on average.

VI. CONCLUSIONS

The experiments in the Section III can be run with a variable number of tours. In particular, more experiments with larger blocks would give us a better understanding about the stopping points in an entire BKZ reduction and thus help us to choose the required trade-off. The limitation that we currently have is to locate properly the exact exponential drop (point $A$) and the linear drop (point $B$). As the number of tours for larger blocks is increasing exponentially, it is hard to locate those points. We are working on it.

ACKNOWLEDGMENT

The authors would like to thank Dr. Ron Steinfeld, Cryptography Research Fellow and Lecturer at Monash University, Australia for his suggestions and advice to conduct this research.

REFERENCES