Collision Resistant Hash Functions on Lattices

Nimish Mishra[1], SK Hafizul Islam[1,*], Daya Sagar Gupta[2], Sherali Zeadally[3]

[1]Department of Computer Science and Engineering, Indian Institute of Information Technology Kalyani, West Bengal 741235, India
[2]Department of Computer Science and Engineering, Shershah College of Engineering Sasaram, Bihar 821308, India
[3]College of Communication and Information, University of Kentucky Lexington, KY, USA 40506
nimish_bt18@iiitkalyani.ac.in, hafi786@gmail.com, dayasagar.ism@gmail.com, szeadally@uky.edu

Abstract

Hash functions have been centers of interest in cryptography because of their *hard to invert* nature, allowing information to be converted to indecipherable formats. However, all previous constructions of hash functions (and other cryptographic primitives) face the threat of being broken by the recent advancements in quantum technology and algorithms. The focus has thus shifted to developing cryptographic primitives on mathematical structures that are, in general, intractable by even quantum algorithms, and lattices are one of them. In this paper, we have reviewed advancements in developing hash functions based on the intractability assumption of the related mathematical problems on lattices. Of independent interest to the reader can be the techniques employed in the underlying reductions of these constructions.

Keywords: Hash function; Lattice; Worst-case hardness assumption; Shortest vector problem, Closest vector problem.

1. Introduction

With the rise of quantum algorithms and the steady advancement made in constructing practical quantum devices and simulators, efforts have been put into designing cryptographic primitives that are secure even under quantum attacks. We assume that the quantum algorithms can take advantage of superposition, entanglement, and other purely quantum phenomena that can process information faster, to up to non-negligible speedups, than their classical counterparts). One such basis of such constructions is lattice-based constructions. The use of lattices as the base for constructing cryptographic primitives received huge interests since the beginning of this century, starting from Ajtai's seminal paper [1] detailing ways to establish the security of lattice-based primitives.

Lattices have been studied extensively and independently among the cryptographic research communities, and the lack of algorithms attacking mathematical problems on lattices conjectures their hardness. One of the finest of such algorithms is the LLL (Lenstra-Lenstra-Lovàsz) [2] algorithm, which is a lattice-based reduction algorithm and its improved versions- achieve exponential approximation factors to lattice problems when running in polynomial time. Moreover, no polynomial-time bounded algorithm exists that can approximate lattice problems within linear factors in the rank of a lattice, for a large value of the rank. Such conjectures solidify the use of lattices in cryptography. However, lattices suffer from a shortcoming that lattice-based constructions allow less structure than their number-theoretic counterparts. For instance, lattice-based hash functions have domains that are subsets of rings and are neither closed under addition nor multiplication. Number-theoretic hash functions, on the other hand, have rings as domains. Quantum algorithms to solve the integer factorization and discrete logarithm problems [3] are inefficient on lattice-based constructions because of this very constrained structure. On the other hand, this constrained structure increases the difficulty in developing lattice-based primitives.

Ajtai's work [1] was the first breakthrough in the attempts to establish the security of lattice-based primitives on *worst-case* hardness assumption of lattice problems. Worst-case hardness assumption is theoretically safer than an average-case hardness assumption, primarily of experimental evidence than algorithms can sometimes perform better on certain random instances, even if the problem is provably secure on the average-case scenario. Extreme caution is then needed to disqualify choices of parameters that make the construction tractable on an average-case situation. The

[*]Corresponding author. (SK Hafizul Islam)

worst-case hardness assumption is based on the intractability of *any* instance of the problem, making it probably more secure than average-case hardness assumption. Thus, when Ajtai demonstrated the relation between an average-case hardness assumption to a worst-case reduction for lattice-based constructions, the interest of the research community sprung up.

In this paper, we described many collision-resistant hash functions based on the worst-case hardness assumption of lattice problems. We review the existing literature that constructs lattice-based hash functions and provide a brief discussion on correctness, the underlying hard problems, optimal parameter choices, weaknesses if any, and reduction techniques applied.

1.1. Organization of the paper

The remaining part of the article is arranged as follows. Section 2 gives a brief description of the concept of lattices and some computational problems on them. Section 3 contains a detailed discussion on the literature so far on lattice based hash functions, with subsection 3.1 dedicated to discussion on hash functions on integer lattices and subsection 3.2 dedicated to hash functions based on cyclic lattices. The section 4 discusses some recent ideas on hash functions. Finally, section 5 contains a comparative study of the hash functions described in the paper and their concrete instantiations in C++.

2. Preliminaries

We indicate n as the security parameter, the prime number $q \in \mathbb{Z}$ as modulus, a capital bold letter (e.g., $\mathbf{A}$) as matrix, $\mathbf{A}^\mathrm{T}$ is the transpose of $\mathbf{A}$, a small bold letter (e,g., $\mathbf{x}$) as a column vector, and $\| \mathbf{x} \|_2 = \sqrt{x_1^2 + x_2^2 + \cdots + x_n^2}$ is the L_2 norm (Euclidean norm) of a vector $\mathbf{x} = (x_1, x_2, \cdots, x_n) \in \mathbb{R}^n$. We also denote $a \leftarrow_R X$ to signify that a is selected uniformly at random from the set X.

Table 1: Notations used in the paper

Notation	Meaning
$\mathbb{R}$	Set of real numbers
$\mathbb{Z}$	Set of integers
$\mathbb{N}$	Set of natural numbers
$\mathcal{R}$	Ring
$\mathcal{L}(\mathbf{B})$	Lattice
$\mathbf{B}$	Basis matrix of $\mathcal{L}(\mathbf{B})$
$\mathbb{R}^n$	Euclidean space of n-dimension
n	Dimension of a lattice
m	Rank of a lattice
$\mathcal{H}$	A hash function family
q	Security parameter
$det(\mathbf{A})$	Determinant of the matrix $\mathbf{A}$
$\mathbf{A}^T$	Transpose of the matrix $\mathbf{A}$
$\mathbf{A}^{-1}$	Inverse of the matrix $\mathbf{A}$

2.1. Integer lattice

Integer lattices are considered as discrete subsets of the n-dimensional Euclidean space $\mathbb{R}^n$. Formally, given a set of m linearly independent vectors in $\mathbb{R}^n$, say $\{\mathbf{b_1}, \mathbf{b_2}, \mathbf{b_3}, \cdots, \mathbf{b_m}\}$, arranged as columns of a matrix $\mathbf{B} = [\mathbf{b_1}, \mathbf{b_2}, \mathbf{b_3}, \cdots, \mathbf{b_m}]$ such that $\mathbf{B} \in \mathbb{R}^{n \times m}$, the integer lattice $\mathcal{L}(\mathbf{B}) \subseteq \mathbb{R}^n$ is defined as the set $\mathcal{L}(\mathbf{B}) = \{\sum_{i=1}^{m} x_i \mathbf{b}_i : x_i \in \mathbb{Z}\}$, or equivalently, $\mathcal{L}(\mathbf{B}) = \{\mathbf{Bx} : x \in \mathbb{Z}^m\}$ for the usual matrix-vector multiplication. The integers m and n signify the *rank* and *dimension* of $\mathcal{L}(\mathbf{B})$. The matrix $\mathbf{B}$ is defined as the basis of $\mathcal{L}(\mathbf{B})$. From a computational point of view, the underlying assumption is that $\mathbf{B}$ has rational coordinates; every such rational matrix can be converted into an integer matrix by multiplying by a suitable scaling factor. Without loss of generality, integer matrices are thus considered as the basis matrices throughout the paper.

Definition 1 (Half-open parallelepiped)**.** Given a basis matrix $\mathbf{B} \in \mathbb{Z}^{n\times m}$, the half-open parallelepiped for a lattice $\mathcal{L}(\mathbf{B})$ is defined as $\mathcal{P}(\mathbf{B}) = \{\mathbf{B}\mathbf{x} : 0 \leq x_i < 1\}$.

$P(B)$ intuitively describes the smallest *unit* of lattice that is repeated over the entire space rendering the lattice its periodic structure.

Lemma 1. An arbitrarily defined matrix $\mathbf{B} \in \mathbb{Z}^{n\times m}$ is a basis for the lattice $\mathcal{L}(\mathbf{B})$ if and only if the half-open parallelepiped $\mathcal{P}(\mathbf{B})$ for $\mathcal{L}(\mathbf{B})$ contains no other lattice point except the origin.

An integer lattice can have several bases, each of which creates a different fundamental parallelepiped while spanning the same set of discrete lattice points.

Definition 2 (Equivalent bases)**.** Two arbitrary and distinct bases $\mathbf{B}$ and $\mathbf{B}'$, both in $\mathbb{Z}^{n\times m}$, are equivalent if and only if there exists a unimodular matrix $\mathbf{U} \in \mathbb{Z}^{m\times m}$, i.e., $det(\mathbf{U}) = \pm 1$ such that $\mathbf{B}' = \mathbf{B}\mathbf{U}$.

The determinant of $\mathcal{L}(\mathbf{B})$ is given as the volume of $\mathcal{P}(\mathbf{B})$ for some basis $\mathbf{B}$. For definition of the same lattice under equivalent bases, volume of $\mathcal{P}(\mathbf{B})$ remains constant since only the shape of the half-open parallelepiped changes under inter-conversion between equivalent bases, further implying that determinant is invariant under change of bases. Geometrically, the determinant is the inverse of density of lattice points.

Definition 3 (Norm)**.** Given a finite n-dimensional vector $\mathbf{x}$, the l_p norm of $\mathbf{x}$ for $p \geq 1$ is given as $\parallel \mathbf{x} \parallel_p = (x_1^p + x_2^p + \cdots + x_n^p)^{1/p}$, where $\mathbf{x} = (x_1, x_2, \cdots, x_n)$.

The value of p can be infinite. l_∞ is then defined as the *max-norm*: $\parallel \mathbf{x} \parallel_\infty = \mathbf{Max}\{x_1, x_2, ..., x_n\}$.

Definition 4 (Open ball)**.** An n-dimensional open ball of radius r centered in $\mathbf{x}$ is defined as $\mathscr{B}(\mathbf{x}, r) = \{\mathbf{y} \in \mathbb{R}^n : \parallel \mathbf{x} - \mathbf{y} \parallel < r\}$.

Definition 5 (Successive minima)**.** Successive minima for a lattice $\mathcal{L}(\mathbf{B})$, definable for any norm $p \geq 1$ and arbitrary basis $\mathbf{B} \in \mathbb{Z}^{n\times m}$, are given as $\lambda_i(\mathbf{B}) = \mathbf{Min}\{r : dim(\mathcal{L}(\mathbf{B}) \cap \mathscr{B}(\mathbf{x}, r)) \geq i\}$.

Informally, the value of i-th successive minimum represents the minimum length r such that a ball of radius r centered about a point $\mathbf{x}$ contains at least i independent vectors. Several upper bounds on $\lambda_i(\mathbf{B})$ can be expressed in the form of the determinant of the lattice, formally known as Minkowski's theorem.

Theorem 1 (Minkowski's theorem)**.** For any lattice $\mathcal{L}(\mathbf{B})$,

$$\{ \prod_{i=1}^{n} \lambda_i(\mathcal{L}(\mathbf{B})) \}^{1/n} \leq \sqrt{\gamma} det(\mathcal{L}(\mathbf{B}))^{\frac{1}{n}}$$

where the approximation factor $\gamma \leq n$ is a function of the dimension only.

2.2. q-ary lattices

Given an integer q and a basis $\mathbf{B} \in \mathbb{Z}^{n\times m}$, the q-ary lattice is an integer lattice $\mathcal{L}(\mathbf{B})$ such that $\mathbb{Z}_q^n \subseteq \mathcal{L}(\mathbf{B}) \subseteq \mathbb{Z}^n$. Any integer lattice is q-ary for some q. There exists a one-to-one correspondence between q-ary lattices of dimension n and subgroups of $\mathbb{Z}_q^n$. Inclusion on any vector $\mathbf{x} \in \mathbb{Z}^n$ in a q-ary lattice depends only upon $\mathbf{x}$ *mod q*.

Definition 6. Let $\mathbf{A} \in \mathbb{Z}_q^{n\times m}$ be an integral matrix with modulo q, then the following *q-ary* lattices are defined:

$$\Lambda_q^{\perp} = \{\mathbf{t} \in \mathbb{Z}^n : \mathbf{A}\mathbf{t} = \mathbf{0} \bmod q\}$$

$$\Lambda_q = \{\mathbf{t} \in \mathbb{Z}^n \text{ and } \mathbf{u} \in \mathbb{Z}^m : \mathbf{t} = \mathbf{A}^{\mathbf{T}}\mathbf{u} \bmod q\}$$

2.3. Dual lattice

Definition 7 (Dual lattice)**.** Given a basis $\mathbf{B} \in \mathbb{Z}^{n\times m}$ and the consequent lattice $\mathcal{L}(\mathbf{B})$, the dual lattice to $\mathcal{L}(\mathbf{B})$ is defined as $\mathcal{L}^*(\mathbf{B}^*) = \{\mathbf{x} : \langle \mathbf{x}, \mathbf{y} \rangle \in \mathbb{Z}, \forall \mathbf{y} \in \mathcal{L}(\mathbf{B})\}$.

Lemma 2. The dual of a lattice $\mathcal{L}(\mathbf{B})$ is a lattice $\mathcal{L}(\mathbf{B}^*)$ with the basis $\mathbf{B}^*$, where $\mathbf{B}^* = \mathbf{B}(\mathbf{B}^T\mathbf{B})^{-1}$.

2.4. Cyclic and ideal lattices

Definition 8 (Rotation shift operation)**.** For an arbitrary vector $\mathbf{x} \in \mathbb{R}^n$, the rotation shift operator, represented as $\mathbf{Rot}(\mathbf{x})$, is defined as $\mathbf{Rot}(\mathbf{x}) = x_n, x_1, x_2, \cdots, x_{n-1}$, where $\mathbf{x} = (x_1, x_2, \cdots, x_n)$.

Repeated application of the operation can be represented as a raised whole number index on **Rot** such as $\mathbf{Rot}^0(\mathbf{x})$, $\mathbf{Rot}^1(\mathbf{x})$, or $\mathbf{Rot}^k(\mathbf{x})$.

Definition 9 (Cyclic lattice)**.** For a given basis $\mathbf{B} \in \mathbb{Z}^{n\times m}$, the lattice $\mathcal{L}(\mathbf{B})$ is said to be cyclic if and only if, for any positive integer k, $\mathbf{Rot}^k(x) \in \mathcal{L}(\mathbf{B})$.

For the usual definitions of rings, isomorphism, and polynomials, ideal lattices are lattices corresponding to ideals in quotient polynomial rings $\mathbb{Z}[x]/\langle f(x)\rangle$ for some irreducible **monic** polynomial $f(x) \in \mathbb{Z}[x]$. A polynomial $f(x)$ is called a ***monic polynomial*** if the leading coefficient of $f(x)$ is 1.

Definition 10 (Ideal lattice)**.** Given an arbitrary basis $\mathbf{B} \in \mathbb{Z}^{n\times m}$, an ideal lattice is an integer lattice $\mathcal{L}(\mathbf{B}) \subseteq \mathbb{Z}^n$ such that $\mathcal{L}(\mathbf{B}) = \{g(x) \bmod f(x) : g(x) \in I(x)\}$ for $I(x) \in \mathbb{Z}[x]/\langle f(x)\rangle$ for some ***monic*** polynomial $f(x)$ of degree n.

There exists an isomorphism between the integer lattice $\mathbb{Z}^n$ and $\mathbb{Z}[x]/\langle f(x)\rangle$. Multivariate extension is given as $\mathbb{Z}[x_1, x_2, \cdots, x_n]/\langle x_1^{r_1} - 1, x_2^{r_2} - 1, \cdots, x_n^{r_n} - 1, \rangle$ such that $r_1 \times r_2 \times r_3 \times \cdots \times r_k = n$.

2.5. Computational Problems

Computational problems can be defined in three equivalent versions: search version, optimization version, and the decision version. Search version requires to *search* for the solution amongst a set of possible solutions, the optimization version requires to search for a solution that optimizes a certain criterion, while the decision version requires outputting a binary decision on whether a specified property is met or not.

For the usual meaning of complexity classes *P* and *NP* (*NP*-hard and *NP*-complete), a reduction from a problem *A* to another problem *B* implies converting any instance of *A* to some instance of *B*. This means given access to an oracle of *B*, *A* can be solved; it is thus stated that *A* is not harder than *B*. In all the succeeding sections, the notion of polynomial-time reductions is used: for any two decision problems *A* and *B*, *A* is said to be reducible to *B* if and only if there exists a polynomial-time bounded function f such that $x \in A$ if and only if $f(x) \in B$ (in this explanation, a decisional problem is defined as determining whether an arbitrary input x belongs to the language represented by the problem *A*, where the reader is referred to the concepts of *language* as used in the theory of computation).

All definitions henceforth are the approximate versions of the corresponding problems. Exact versions are derivable by fixing the approximation factor $\gamma = 1$.

Definition 11 (Minimum distance)**.** The **minimum distance** of a lattice $\mathcal{L}(\mathbf{B})$ can be expressed as:

$$d_{min}(\mathcal{L}) = \underset{\mathbf{v}\in\mathcal{L}\setminus\{\mathbf{0}\}}{\mathbf{Min}} \parallel \mathbf{v} \parallel$$

Definition 12 (Shortest vector problem (SVP))**.** Let $\mathbf{B} \in \mathbb{Z}^{n\times m}$ be a basis matrix of the lattice $\mathcal{L}(\mathbf{B})$, finding a non-zero vector $\mathbf{v} \in \mathcal{L}(\mathbf{B})$ such that $\parallel \mathbf{v} \parallel = d_{min}(\mathcal{L})$ is hard.

Definition 13 (Closest vector problem (CVP))**.** Let $\mathbf{B} \in \mathbb{Z}^{n\times m}$ be a basis matrix of the lattice $\mathcal{L}(\mathbf{B})$ and $\mathbf{v}$ be a vector such that $\mathbf{u} \notin \mathcal{L}$, finding a non-zero vector $\mathbf{v} \in \mathcal{L}$ such that $\parallel \mathbf{u} - \mathbf{v} \parallel = d_{min}(\mathcal{L})$ is hard.

Definition 14 (Approximate SVP)**.** Given a basis $\mathbf{B} \in \mathbb{Z}^{n\times m}$ of a lattice $\mathcal{L}(\mathbf{B})$, and an approximation factor $\gamma \geq 1$ as function of the dimension n of $\mathcal{L}(\mathbf{B})$, approximate versions of SVP are given as:

- **Search SVP$_\gamma$:** Find a non-zero vector $\mathbf{v} \in \mathcal{L}(\mathbf{B})$ such that $\parallel \mathbf{v} \parallel_2 \leq \gamma \cdot \lambda_1(\mathcal{L}(\mathbf{B}))$
- **Optimization SVP$_\gamma$ (OptSVP$_\gamma$):** Find d such that $d \leq \lambda_1 \cdot (\mathcal{L}(\mathbf{B})) < \gamma{\cdot}d$.
- **Decisional SVP$_\gamma$ (GapSVP$_\gamma$):** Given a positive rational number r, determine whether $\lambda_1 \cdot (\mathcal{L}(\mathbf{B})) \leq r$ or $\lambda_1(\mathcal{L}(\mathbf{B})) > \gamma{\cdot}r$

Definition 15 (Approximate CVP)**.** Given a lattice basis $\mathbf{B} \in \mathbb{Z}^{n\times m}$, a target vector $\mathbf{t} \in \mathbb{R}^n$, and an approximation factor γ as function of the dimension n of $\mathcal{L}(\mathbf{B})$, approximate versions of CVP are given as:

- **Search CVP$_\gamma$:** Find a vector $\mathbf{v} \in \mathcal{L}(\mathbf{B})$ such that $\| \mathbf{t} - \mathbf{v} \|_2 \leq \gamma \cdot \mathbf{dist}(\mathbf{t}, \mathcal{L}(\mathbf{B}))$, where $\mathbf{dist}(\mathbf{t}, \mathcal{L}(\mathbf{B}))$ is the distance of $\mathbf{t}$ from $\mathcal{L}(\mathbf{B})$.

- **Optimization CVP$_\gamma$ (OptCVP$_\gamma$):** Find d such that $d \leq \mathbf{dist}(\mathbf{t}, \mathcal{L}(\mathbf{B})) < \gamma \cdot d$.

- **Decisional CVP$_\gamma$ (GapCVP$_\gamma$):** Given a positive rational number r, distinguish between $\mathbf{dist}(\mathbf{t}, \mathcal{L}(\mathbf{B})) \leq r$ and $\mathbf{dist}(\mathbf{t}, \mathcal{L}(\mathbf{B})) > \gamma \cdot r$.

Definition 16 (Shortest independent vectors problem (SIVP))**.** Given a basis matrix $\mathbf{B} = [\mathbf{b_1}, \mathbf{b_2}, \mathbf{b_3}, \cdots, \mathbf{b_m}]$ of the lattice $\mathcal{L}(\mathbf{B})$ of dimension n, find n linearly independent vectors $\mathbf{v}_1, \mathbf{v}_2, \cdots, \mathbf{v}_n$ such that $\mathbf{Max}\|\mathbf{v}_i\| \leq \mathbf{Max_B}\|\mathbf{b}_i\|$.

Definition 17 (Approximate SIVP)**.** Given a lattice basis $\mathbf{B} \in \mathbb{Z}^{n \times m}$, and an approximation factor γ as function of the dimension n of $\mathcal{L}(\mathbf{B})$, approximate versions of SIVP are given as:

- **Search SIVP$_\gamma$ (SIVP$_\gamma$):** Find n linearly independent vectors $\mathbf{v}_1, \mathbf{v}_2, \cdots, \mathbf{v}_n \in \mathcal{L}(\mathbf{B})$ of length $\mathbf{Max} \| \mathbf{v}_i \|_2 \leq \gamma \cdot \lambda_n(\mathcal{L}(\mathbf{B}))$.

- **Optimization SIVP$_\gamma$ (OptSIVP$_\gamma$):** Find d such that $d \leq \lambda_n \cdot (\mathcal{L}(\mathbf{B})) < \gamma \cdot d$.

- **Decisional SIVP$_\gamma$ (GapSIVP$_\gamma$):** Given additionally a positive rational number r, distinguish between $\lambda_n(\mathcal{L}(\mathbf{B})) \leq r$ or $\lambda_n(\mathcal{L}(\mathbf{B})) > \gamma \cdot r$.

Definition 18 (Covering radius)**.** The covering radius in the l_p norm of a full-rank lattice $\mathcal{L}(\mathbf{B}) \in \mathbb{R}^n$ is defined as

$$\rho(\mathcal{L}(\mathbf{B})) = \mathbf{Max}_{\mathbf{x} \in \mathbb{R}^n} \mathbf{dist}(\mathbf{x}, \mathcal{L}(\mathbf{B}))$$

The covering radius ρ of $\mathcal{L}(\mathbf{B})$ is defined as the minimum radius of the (closed) spheres when centered at all lattice points cover the entire space, i.e., any point in $\mathbf{span}(\mathbf{B})$ is within distance ρ from $\mathcal{L}(\mathbf{B})$. Note that $\mathbf{span}(\mathbf{B})$ for any $\mathbf{B} \in \mathbb{Z}^{n \times m}$ is the set of points $\{\mathbf{Bx} : \mathbf{x} \in \mathbb{R}^m\}$. $\mathbf{span}(\mathbf{B})$ covers the entire $\mathbb{R}^n$ space, with n being the dimensionality of $\mathbf{B}$ if $n = m$.

Definition 19 (Approximate covering radius problem)**.** Given a lattice basis $\mathbf{B}$, and an approximation factor γ as function of the dimension n of $\mathcal{L}(\mathbf{B})$, approximate versions of CRP are given as:

- **Search CRP$_\gamma$):** No known problem formulation whose solution is verifiable in polynomial time [4].

- **Decisional CRP$_\gamma$ (GapCRP$_\gamma$):** Given additionally a positive rational number r, distinguish between $\rho(\mathcal{L}(\mathbf{B})) \leq r$ and $\rho(\mathcal{L}(\mathbf{B})) > \gamma \cdot r$ where ρ is the covering radius.

Definition 20 (Approximate unique shortest vector problem (u-SVP))**.** Given a basis matrix $\mathbf{B}$ of $\mathcal{L}(\mathbf{B})$, the promise that $\lambda_2(\mathbf{B}) > \gamma \cdot \lambda_1(\mathbf{B})$, and an approximation factor γ as function of the dimension n of $\mathcal{L}(\mathbf{B})$, approximate search version of u-SVP is given as:

- **Search u-SVP$_\gamma$:** Find a non-zero vector $\mathbf{v} \in \mathcal{L}(\mathbf{B})$ such that $\| \mathbf{v} \|_2 \leq \gamma \cdot \lambda_1(\mathbf{B})$

Definition 21 (Approximate guaranteed distance decoding problem (GDD))**.** Given a basis matrix $\mathbf{B}$ of $\mathcal{L}(\mathbf{B})$, a target vector $\mathbf{t} \in \mathbf{span}(\mathbf{B})$, and an approximation factor γ as function of the dimension n of $\mathcal{L}(\mathbf{B})$, approximate search version of GDD is given as:

- **Search GDD$_\gamma$:** Find a vector $\mathbf{v} \in \mathcal{L}(\mathbf{B})$ such that $\mathbf{dist}(\mathbf{t}, \mathbf{v}) \leq \gamma \cdot \rho(\mathcal{L}(\mathbf{B}))$, where ρ is the covering radius of$\mathcal{L}(\mathbf{B})$.

In these and other similar lattice problems, the approximate factor γ is considered a function of the dimension of the lattice. Several years of research has strengthened the conjecture that achieving polynomial approximation factors γ from polynomial time algorithms is intractable, allowing these problems to be the basis of security of various lattice based cryptographic primitives.

Definition 22 (Negligible function)**.** A function $\varepsilon(k) : \mathbb{N} \to \mathbb{R}$ is said to be negligible if, for every integer $v > 0$, there exists an integer u such that $\varepsilon(k) \leq \frac{1}{k^v}$ holds $\forall k \geq u$

3. Discussion on collision-free hash function

Theoretically, cryptographic schemes are never secure in the face of an adversary with unbounded computational capacity. Practically, however, all adversaries are limited by the computational capacity they have. A cryptographic scheme is considered secure if an adversary can not, with non-negligible probability, break it efficiently. To formally quantify *non-negligible* probability, negligible functions are considered.

Definition 23 (Collision-free hash function)**.** A Collision resistant hash function $h(\cdot)$ is defined as a keyed function $h : \{0,1\}^a \to \{0,1\}^b$ if it satisfies the following two properties:

(1). Compression: $a > b$

(2). Collision resistance: there exists a negligible function $\varepsilon(n)$ for all security parameters $n \in \mathbb{N}$ such that:

$$\mathbf{Pr}[x, y \leftarrow_R \{0,1\}^a : \ x \neq y \text{ and } h(x) = h(y)] \leq \varepsilon(n)$$

where $x \leftarrow_R \{0,1\}^a$ represents a sample x is drawn at random from $\{0,1\}^a$ with uniform probability distribution.

Or, the conditional probability of finding distinct x and y from $\{0,1\}^a$ such that $h(x) = h(y)$ is negligible. More concretely, collision resistance property is required on a family $\mathcal{H}$ of hash functions, so that a non-uniform adversary $\mathcal{A}$ can not efficiently find collisions for a Collision resistant hash function $h(\cdot)$ drawn at random from $\mathcal{H}$.

3.1. Collision-free hash function proposed by Goldreich et al. [5] and associated improvements

The first breakthrough in constructing lattice-based primitives was Ajtai's seminal works [1, 6] in which was presented a family of one-way functions based on the worst-case hardness of n^c-approximate SVP for some constant $c > 0$. Being able to invert a function from this family with non-negligible probability means solving *any* instance (since the assumption was worst-case hardness) of the n^c-approximate SVP problem. Goldreich *et. al.* [5] modifies Ajtai's work [1] to construct a collision-free hash function family on nearly the same lines as Ajtai did. The main result is the construction of two similar hash function families based on modular linear equations. Of independent interest to the reader is the mathematics for the procedure to establish worst-case to average-case reductions: a sampling procedure wherein a large space is considered and is divided into sub-regions ensuring *almost* uniform distribution of lattice points, such that sampling from such sub-regions yields useful *short* vectors while being *almost* uniformly distributed over the concerned group, usually $\mathbb{Z}_q^n$.

For a given security parameter n, a matrix $\mathbf{A} \in \mathbb{Z}_q^{n\times m}$ is chosen uniformly at random, m and n are chosen such that $n \log q < m < \frac{q}{2n^4}$, $q = \mathcal{O}(n^c)$ (for instance $m = \mathcal{O}(n^2)$, $q = \mathcal{O}(n^7)$) for some constant $c > 0$. The hash function family $h_{\mathbf{A}} : \{0,1\}^m \leftarrow \mathbb{Z}_q^n$ is defined for a binary message $\mathbf{x} = \{\mathbf{x}_1, \mathbf{x}_1, \cdots, \mathbf{x}_m\} \in \{0,1\}^m$ as

$$h_{\mathbf{A}}(\mathbf{x}) = \mathbf{A}\mathbf{x} \pmod q$$

Since $m > n \log q$, therefore collisions are necessarily existent in $h_{\mathbf{A}}$. With regards to the underlying SIS problem definition, the main difference from Ajtai's formulation is the constraint on the input domain to be $\{-1, 0, 1\}$. To find collisions for distinct lattice vectors say $\mathbf{a}$ and $\mathbf{b}$ such that $\mathbf{x} = \mathbf{a} - \mathbf{b}$, Ajtai considered $\{\mathbf{x} \in \mathbb{Z}_q^m : \| \mathbf{x} \|_2 < n; \mathbf{A}\mathbf{x} \equiv 0 (\text{mod } q)\}$. However, Goldreich *et. al.* [5] consider $\mathbf{x}$ to be strictly from $\{-1, 0, 1\}$, a harder constraint on the length of the vector, which is now $\mathcal{O}(\sqrt{m})$.

Goldreich *et. al.* [5] also define family as $h_{\mathbf{A},\mathbf{r}}(\mathbf{x}) = \mathbf{A}\mathbf{x} + \mathbf{r}$ for an additional random vector $\mathbf{r} \in \mathbb{Z}_q^n$. This ensures the construction to be *universal* or any two images are spread uniformly over the range in a pairwise independent manner. Both these families are collision resistant due to the constraint on the input vector, as discussed above.

Remark: The constant c in Ajtai's work [1] and Goldreich *et. al.* [5] was tightened further by Cai *et. al.* [7], [8] reducing $c > 8$ in the former works to $c = 4 + \varepsilon$ in the latter. Further tightening of the bound was done by Micciancio [9] where hardness of one-way and collision resistant functions respectively was still based on n^c-approximate SVP with reductions in the approximation factor further to $\mathcal{O}(\tau n^3 \log n)$ with $\tau \in [1, \sqrt{n}]$ or roughly to a factor of $3 + \varepsilon$ (τ is the ratio of covering radius of the lattice to the packing radius of the lattice, for usual definitions of covering and packing radii for lattices). This improvement is based on *covering radius problem* (Definition 19) on lattices, and the reductions that follow therein. No deterministic algorithm is known that can solve CRP in *polynomial time*. *CRP* is not even known to be in NP and is conjectured to be Π_2 hard [10]. A direct implication is that CRP is probably harder than SVP or other well-studied lattice-based problems. A construction based on CRP can thus be considered secure based on assumed hardness of CRP.

Let the output of a hash function $h_{\mathbf{A}}(\mathbf{x})$ be in $\mathbb{Z}_q^n$, for some appropriate integer q. This is a finite, abelian group that can be considered equivalent as the quotient group $\mathbb{Z}^n/q\mathbb{Z}^n$. $q\mathbb{Z}^n$ is naturally partitioned into q^n hyper-cubes, with each hyper-cube corresponding to some element in $\mathbb{Z}^n/q\mathbb{Z}^n$, and thus to some element in $\mathbb{Z}_q^n$. Consider another lattice $\mathcal{L}(\mathbf{B})$ generated by basis $\mathbf{B}$ and let each partitioned hyper-cube of $q\mathbb{Z}^n$ contain roughly the same number of lattice points in $\mathcal{L}(\mathbf{B})$. We want to estimate number of such points in each hyper-cube, and connect hardness of doing so with the security of the described hash functions. However, the main improvement in approximation factor in this work, as discussed earlier, comes from replacing such hyper-cubes with another construction called Voronoi cells. For lattices considered in this work, Voronoi cells are *almost* spherical, thereby saving up on space occupied with cubes (precisely hyper-cubes) and causing an efficient partition of space.

Definition 24 (Voronoi cell)**.** Voronoi cell for a certain lattice point $\mathbf{x} \in \mathcal{L}(\mathbf{B})$ is described as $V(\mathbf{x},\mathcal{L}) = \{\mathbf{z} \in \mathbf{span}(\mathcal{L}) : \forall \mathbf{y} \in \mathcal{L}(\mathbf{B}), \| \mathbf{z}-\mathbf{x} \|_2 \leq \| \mathbf{z}-\mathbf{y} \|_2\}$, i.e., the set of points that are closer to $\mathbf{x}$ than to any other point in $\mathcal{L}(\mathbf{B})$. The main property of Voronoi cells:

$$\text{sphere of radius } \frac{\lambda_1}{2} \subset V(\mathbf{x},\mathcal{L}) \subset \text{sphere of radius } \rho$$

where ρ and $\frac{\lambda_1}{2}$ are the is the covering radius and packing radius of of $\mathcal{L}(\mathbf{B})$, respectively, for the usual definition of λ_1 in definition 14. When τ is balanced, the packing and covering radii are almost equal implying $V(\mathbf{x},\mathcal{L})$ are *almost* spherical. The idea about Voronoi cells is essential for detailed proofs, for which we refer the reader to the original work.

For an arbitrary lattice $\mathbf{A}$ and a scaling factor α, a sub-lattice $\mathcal{L}(\mathbf{M}) \subset \mathbf{A}$ is defined as $\mathbf{M} = \alpha\rho(\mathbf{A})\mathbf{I}+\mathbf{R}$, where $\mathbf{I}$ is the identity matrix while $\mathbf{R}$ is a matrix chosen at random such that the lengths of columns of $\mathbf{R}$ are upper-bounded by $\rho(\mathbf{A})$, or informally lattice points in $\mathcal{L}(\mathbf{M})$ that are within a distance of $\rho(\mathbf{A})$ of lattice points in $\mathbf{A}$. Define a finite abelian group $G = \mathbf{A}/\mathcal{L}(\mathbf{M})$. Representation of G is possible such that elements of G can be represented as $\log_2 | G |$ bits. The hash function is then given as $h_{\mathbf{a}} : \{0,1\}^m \to G$ for input binary string $\mathbf{x}$ of length m.

$$h_{\mathbf{a}}(\mathbf{x}) = \sum_{i=1}^{m} a_i x_i$$

for some key $\mathbf{a} \in G^m$. Evidently, collisions exist for $m > \log_2 | G |$. Also note the following relation holds $\log_2 | G | < n(\log_2 n + \log_2 \alpha)$. For parameter α chosen to be asymptotically similar to n, this expression reduces to $\mathcal{O}(n\log_2 n)$ and gives an upper bound on the output domain of the hash function. Finally, finding collisions for any $\mathbf{a}$ chosen uniformly at random from G^m is at least as hard as approximating covering radius for any lattice up to factors of γ, thereby establishing security of the hash function.

Remark: Micciancio and Regev [11] further reduced the approximation factor to linear in n using Gaussian measures, and improved on the assumptions in [12] where the hardness is based on approximate *unique*-SVP that is not known to be NP-hard for small approximation factors. This work is important in a number of respects: connecting hardness of defined hash function family with worst case hardness of SVP, SIVP, and CRP for approximation factors up to $\tilde{\mathcal{O}}(n)$, introduce a general abstraction of procedure for worst-case to average-case reduction that might be of independent interest to the reader, and introduce techniques based on Gaussian measures that help in reductions among lattice problems, again of probable independent interest to the reader.

In brief, the authors [11] designed a hash function family on SIS problem. Given a hash function $h_{\mathbf{a}}$ taken uniformly at random from the hash function family such that

$$h_{\mathbf{a}}(\mathbf{x}) = \mathbf{A}\mathbf{x} \bmod q$$

for $\mathbf{A} \in \mathbb{Z}_q^{n\times m}$, $\mathbf{x} \in \{0,1\}^m$, $q = n^{\mathcal{O}(1)}$ and $m > n\log_2 q$ for collisions to necessarily exist. For distinct input vectors $\mathbf{a}$ and $\mathbf{b}$, existence of collision implies $\mathbf{Aa} - \mathbf{Ab} \equiv \mathbf{0} \bmod q$ i.e., $\mathbf{Az} \equiv \mathbf{0} \pmod q$ such that $\mathbf{z} \neq \mathbf{0}$. Given the input domain to be $\{0,1\}^m$, it is easy to see that $\| \mathbf{z} \|_\infty = 1$. The problem of finding collisions is thus reduced to finding short integer solutions to this homogeneous equation, which is intractable in polynomial time. The requirement in the entire setup is that $\mathbf{A}$ must be chosen uniformly at random from $\mathbb{Z}_q^{n\times m}$. This work is also interesting for a general take on approaches to worst-case to average-case reductions (for true average-case instantiations of the hash functions). Almost all the previous works [1], [5], [7], [12], [13] follow a similar approach that is abstracted here. Note that outputs of the hash function are usually in $\mathbb{Z}_q^n$. The basic idea is to search for the existence of a polynomial-time sampling procedure that samples group elements and corresponding *offset* vectors (this offset vector doesn't need to be in the lattice defined). Should this offset vector (compare with $\mathbf{z} = \mathbf{a} - \mathbf{b}$) be the solution of the homogeneous modular linear equation, it implies the offset vector maps to an actual lattice point. Depending on the norm of the sampled vector (compare with the norm, especially with l_2 norm, of $\mathbf{a}$ and $\mathbf{b}$), the offset vector can be considered short, implying it maps to a short vector in the lattice.

The difference between works lies in the way the sampling procedure works. This work details a method of division of a large hypercube into smaller hyper-cubes such that each hypercube corresponds to some element of $\mathbb{Z}_q^n$. The sampling procedure is constructed around this idea, such that for every sample from $\mathcal{L}(\mathbf{B}) \cap q\mathbb{Z}^n$, there is a corresponding smaller hypercube C wherein that sample lies and a corresponding offset vector concerning the center of C. The construction of these smaller hypercubes plays an important part: smaller hypercubes are small enough that the offset vectors are small, and large enough that almost equal number of lattice points in $\mathcal{L}(\mathbf{B})$ lie in each of the smaller hyper-cubes, thereby ensuring uniformly at random distribution over $\mathbb{Z}_q^n$. The reader is also referred to an earlier section for improved sampling procedure wherein Voronoi cells instead of smaller hypercubes were considered, and lattice $\mathcal{L}(\mathbf{B})$ was chosen such that these cells are almost spherical ensuring tighter packing (and thus even shorter offset vectors) and better approximation factors and improved security.

The sampling procedure from this work takes a different path, which is primarily responsible for efficient approximation factors. Instead of dividing large space into sub-spaces, the authors take a random lattice point and a random noise vector with a Gaussian distribution and reduce the noise vector modulo the basis such that a uniform distribution is obtained the fundamental parallelepiped. This fundamental parallelepiped is divided into q^n sub-regions to correspond to group elements in $\mathbb{Z}_q^n$; the reduced noise vector then induces almost uniform distribution over $\mathbb{Z}_q^n$. Such additions of Gaussian blur to smooth a discrete lattice into (almost) uniform distribution is also seen in Regev[12], which is described next. The work of Regev[12] is based on the assumed hardness of n^c-uSVP problem (to find the shortest non-zero vector in a n-dimensional lattice with the promise that it is shorter than all other non-parallel vectors by a factor of n^c). It introduces the idea of Fourier analysis as an integral part of lattice-based constructions. Earlier indirect applications of Fourier analysis was through transference theorems in the works of Cai *et. al.*[8].

The hash function family considered is the modular subset sum function (whose security is based on the worst case hardness of $\mathcal{O}(n^{1.5})$-uSVP), also appearing independently in Impagliazzo *et. al.*[14] involving an average-case to average-case reduction of the function. Modular subset sum problem is a specialized version of the subset sum problem but with the added constraint that the subset sum is modulo some positive integer. It was noted by Ajtai[1] that results of hash functions based on random lattices can be extended to modular subset sum functions, thus connecting author's present work with past constructions of hash functions. Ajtai's work [1] requires four reductions from instances of the aforementioned $\mathcal{O}(n^{1.5})$-uSVP lattice problem to the problem of distinguishing between two distributions (a uniform distribution and a special distribution concentrated around integral multiples of $\frac{1}{h}$ for some large *unknown h*, where h is functionally related to the shortest vector in the lattice). The main reduction involves four reductions and contains ideas about making the lattice sparse without losing the shortest vector; using this sparse nature of the lattice to ensure if there exists a short vector of length $\frac{1}{n}$ such that all non-parallel vectors to this short vector are of length at least $\frac{n^{1.5}}{n} = \sqrt{n}$; the third step uses a prime idea of Ajtai and Dwork [15] (based on a lemma of Banaszczyk [16]) that combines a *random* lattice point in the dual lattice with a Gaussian of radius $\sqrt{n}$ to produce two n-dimensional distributions (uniform and specialized as discussed in the statement of the problem in the first sentence of this paragraph); the last reduction transforms these n-dimensional distributions to one-dimensional distributions and completes the initial distinguishability problem described. Here, n-dimensional distributions are *random* vectors whose n components are random variables. There is another property for the *specialized* distribution centered around integral multiples of $\frac{1}{h}$. An algorithm able to distinguish between the distributions as mentioned above for a non-negligible fraction of values of h can distinguish them for all values of h. The authors provide a detailed proof of the hardness of this *average case* problem.

A general case hash function on the modular subset set problem is formulated as

$$f(\mathbf{b}) = \sum_{i=1}^{m} b_i a_i \bmod N$$

where $\mathbf{b}$ denotes the actual binary vector to be hashed ($b_i \in \{0,1\}$ for $i = 1,2,3,\cdots,m$) and $m = \mathcal{O}(logN)$ and a_i for all i denote the components of the key used in the hash function. A collision finding algorithm needs to search, with non-negligible probability, a non-zero vector $\mathbf{a}$ such that $\| \mathbf{a} \|_2 \leq \sqrt{m}$ (sufficiently *short* vector) and $\sum_{i=1}^{m} a_i a_i \equiv \mathbf{0} \pmod N$. This algorithm then has a solution to n^c-uSVP.

3.2. Collision-free hash function on Cyclic and ideal lattices

Regarding all suggested schemes discussed before, while the representation of hash function parameters is feasible in present memory constraints, the size of the key is a major bottleneck: the key $\mathbf{A} \in \mathbb{Z}_q^{n\times m}$ is chosen uniformly at random. With growing n, m also grows. It was soon observed that by imposing a special structure on the lattice used, efficient schemes could be designed. One such scheme proceeds by not choosing the key $\mathbf{A}$ at random as done in previous integer lattice based constructions [5] and the improvements thereafter, but with special cyclic structure. The key would be a block matrix where each block ($\mathbf{A}^i \in \mathbb{Z}_q^{n\times m}$) is a *circulant* matrix. So there exist $\frac{m}{n}$ such blocks. Two improvements are straightforward: reduced storage size since only the first column need to be stored (rest of the columns are simply cyclic permutations of the first column), and the running time of matrix vector product is reduced to asymptotically linear time (Fast Fourier Transform (FFT) is applicable due to the cyclic structure). This structure however invalidates the proofs of security established by various other works [7], [9], [11] since all of them assume a uniform distribution of key $\mathbf{A}$ over $\mathbb{Z}_q^{n\times m}$. One proof of security was, however, given by Micciancio [17] who showed the average hardness of inverting hash functions built upon the circulant property, but only for few classes of lattices that were invariant under cyclic rotations of coordinates. It turned out that finding collisions was not difficult [18], [19]. In a way, the existence of collisions for these functions demonstrates the importance of theoretical security proofs whenever a cryptographic construction is modified. The problem of describing collision-resistant hash functions on lattices was tackled by Lyubashevsky *et. al* [18] and Peikert *et. al.* [19]. Both works [18, 19] consider collision-resistant hash function families built on the generalized knapsack function on ideal lattices; Lyubashevsky *et. al* [18] takes a more generalized approach than Peikert [19] in a sense that it considers general monic polynomials in the definition of the *ideals* instead of a specific polynomial (($\alpha^n - 1$) for variable α) taken in the latter. This improves upon the scheme in [19] since choices of the polynomial other than ($\alpha^n - 1$) lead to better hash function families in some cases.

The work of Lyubashevsky *et. al* [18] is an extension to the detailed work done on building hash function families around generalized knapsack functions. The primary security assumption is the lack of polynomial-time algorithms to attack lattice problems on ideal lattices, partly motivated by the fact that algorithms attacking lattice problems have been unable to take advantage of the cyclic structure of lattices. Generalized knapsack functions have several attacks detailed in the works of Joux [20], Shamir [21], Vaudenay [22] but these attacks are useless against knapsacks based on ideal lattices. The collision-resistant hash family $\mathcal{H}(\mathcal{R}, D, m)$ from message domain D^m to output domain $\mathcal{R}$ is defined for $\mathcal{R} = \mathbb{Z}_p[\alpha]/\langle f \rangle$ where α is the variable, $D = \{g \in \mathcal{R} : \| g \|_f \leq d\}$ for some positive integer d and polynomial modulo norm $\| g \|_f$ [18], and rank m of the lattice. Conditions for collision-resistance requires f to be *irreducible* over the polynomial ring and a certain ratio, called *expansion factor* to be as low as possible.

The authors [18] define a new hard problem: approximate *shortest polynomial problem* to establish hardness of their hash function family. For the usual definition of infinite norm of a vector $\mathbf{x}$, the infinite norm of a set S is given as $\lambda_1^\infty(S) = min\{\| x \|_\infty : x \in S\}$. Given a monic polynomial f of degree n and an ideal $I \subseteq \mathbb{Z}[\alpha]/\langle f \rangle$, the problem is to find a non-zero $g \in I$ such that

$$\| g \|_f \leq \gamma \lambda_1^\infty(I)$$

for some approximate factor γ. The authors provide a detailed proof of polynomial time reduction from this problem to finding collisions, thereby establishing security of the hash function.

For a hash function $h \in \mathcal{H}$ such that $h_{\mathbf{a}}(\mathbf{b}) = \sum_{i=1}^m a_i b_i$ for key $\mathbf{a} \in \mathcal{R}^m$ and input $\mathbf{b} \in D^m$ where h is uniformly sampled from function family $\mathcal{H}$ at random. Collisions are inevitable for $m > \frac{log\, p}{log\, 2d}$ where d is the parameter in the definition of D such that $| D^m | = (2d+1)^{nm}$ and p is such that $| \mathcal{R} | = p^n$ for dimensionality n and rank m ($| . |$ denotes the cardinality of the given set, in the usual sense of cardinality). The size of the key in the hash function is $\mathcal{O}(n\, log\, n)$ and hashing of the message can be done in $\mathcal{O}(n\, log\, n\, log\, log\, n)$.

The more specific approach of Peikert *et.al.*[19] extends Micciancio's work and shows there exist collision-resistant hash functions derived from certain instantiations of generalized knapsack function based on the assumed hardness of approximate *shortest vector problem* in cyclic lattices. This work has a stronger *worst-case* assumption than Micciancio's: SVP is hard on cyclic lattices for all sufficiently large *prime* dimension. The authors formulate the proof using a chain of reductions, thereby introducing several interesting generalised problems on the way which are worth looking at in their own regard. The reader is referred to the original article for detailed discussion.

Since integer cyclic lattices are isomorphic to *ideals* in $\mathbb{Z}[\alpha]/(\alpha^n - 1)$ with α being the indeterminate, $(\alpha^n - 1)$ can be represented in terms of product of cyclotomic polynomials. A cyclotomic polynomial $\Phi_k(\alpha)$ is a *minimal* polynomial in the kth primitive root of unity (ζ),

$$\Phi_k(\alpha) = \prod_{j\in[1,k];gcd(j,k)=1} (\alpha - \zeta^j)$$

such that these polynomials are irreducible over $\mathbb{Z}[\alpha]$ and $(\alpha^n - 1)$ can be represented as product of such cyclotomic polynomials. For any given vector $\mathbf{x}$ belonging to a field F, $x(\alpha) \in F[\alpha]$, or a polynomial in α whose coefficients are the components of x. Then, the cyclotomic subspace H_Φ is defined as the *linear* subspace under $\mathbb{R}^n$:

$$H_\Phi = \{\mathbf{x} \in \mathbb{R}^n : \Phi(\alpha)\ divides\ x(\alpha)\ over\ \mathbb{R}[\alpha]\}$$

where $\Phi(\alpha)$ is the product of $\Phi_k(\alpha)$ for some values of *k*. It is not necessary that $\Phi(\alpha) = (\alpha^n - 1)$.

The cyclotomic versions of worst case lattice problems on cyclic lattices are then described: *subSIVP* and *subSVP*. Both of these problems require a n dimensional full-rank ($rank = dimension$) cyclic lattice basis B, a polynomial $\Phi(\alpha)$ *not* equal to $(\alpha^n - 1)$ but dividing the latter. Given these conditions, *subSIVP* asks a set of vectors S such that $\| S \| \leq \gamma(n).\zeta(D)$ and *subSVP* asks for a single vector $\mathbf{m}$ such that $\| \mathbf{m} \| \leq \gamma(n).\zeta(D)$. Here D is the domain under consideration (might be the entire lattice or the space defined by H_Φ) and ζ is an arbitrary function (*n*th successive minimum or any other function defined on lattices). The authors note there exist worst-case to worst-case reductions among these versions of computational problems.

The authors base their hash functions on generalised knapsack function defined as $f_{\mathbf{a}}(\mathbf{x}) = \sum_{i=1}^{m} a_i x_i$ for some $a_i \in \mathcal{R}$ where $\mathcal{R}$ is some ring, and $\mathbf{x} \in S \subseteq R$. The security parameter for the function is the *dimension* of $\mathcal{R}$: n as $\mathcal{R} = (\mathbb{Z}_p^n, +, \times)$, where p is the positive integer modulo to which integers are present in the n dimensional ring. S is chosen to be $[0, p^{\Theta(1)}]$. Such a choice leads to efficient implementation of the hash function: convolution ($\mathcal{O}(n\,log\,n)$), addition of vectors ($\mathcal{O}(n\,log\,n)$), and so on.

The generalised knapsack function is linear: $f_{\mathbf{a}}(\mathbf{d}) + f_{\mathbf{a}}(\mathbf{d}') = f_{\mathbf{a}}(\mathbf{d} + \mathbf{d}')$. For $f_{\mathbf{a}}(\mathbf{d}') = 0$, collision is detected. Thus given a fixed $\mathbf{d} \in S$, we need to find $\mathbf{d}'$ such that the condition is satisfied. Note however a condition on $\mathbf{d}$: $\| \mathbf{d} \|_\infty < p^{\Theta(1)}$, or an upper bound on the infinite norm of $\mathbf{d}$ giving the following relation:

$$\| \mathbf{d} \| \leq \sqrt{n} \| \mathbf{d} \|_\infty < p^{\Theta(1)}$$

and $\| \mathbf{d}' \|_\infty = 1$, thereby implying we are trying to find relatively short vectors. Such special constraints on S can prevent such an attack. For an efficient hash function, security parameter n is chosen as prime and $\Phi(\alpha) = \alpha - 1$; the hash function has outputs compressed by factors of $\frac{m\,log\,(p^{\Theta(1)})}{log\,p}$. The security of the above hash function is established by worst case reduction from an incremental version of *subSVP* where $\zeta = \eta_\varepsilon$ to finding collisions (η_ε is described as the the lattice function tasked with finding the value of the *smoothing* parameter for the given lattice, or informally, finding the radius of the Gaussian noise whose uniform samples when added to the lattice can transform it into a *near* uniform distribution, i.e. *loss of discreteness* due to added *blur*). Hardness is therefore established by further reduction from *SVP* to the above chosen cyclotomic version implying hardness is based on solutions to *SVP* in the worst case. For instance, for *prime* n, $m = \Theta(log\,n)$ and $p = n^{2.5+\Theta(1)}$, finding collisions is as hard as solving *SVP* up to factors of $n.poly(log\,n)$ with non-negligible probability.

Despite the security guarantee, there exist trade-offs between the provable security of a certain hash function family and its efficient implementation. SWIFFT is one such proposal that is provably secure as well as efficient to implement. First introduced in Lyubashevsky *et. al.*[23] as an extension to the works of Micciancio[17], SWIFFT is a collision resistant hash function with security based on the worst-case problems in lattices corresponding to ideals in $\mathbb{Z}_p[x]/\langle x^n + 1\rangle$ (for variable x) with comparable performance to ad-hoc hash functions in use today. SWIFFT arranges the input of binary strings of length mn as a block matrix $\in \{0, 1\}^{m\times n}$, for suitable choices of security parameters m and n. Efficient implementation of this scheme is achieved by using column Fast Fourier transform on the aforementioned matrix. The construction of the function ensures computation is parallelizable and connected to well-studied cryptographic problems. Formally, the function is described as $h_{\mathbf{a}}(\mathbf{x}) = \sum_{i=1}^{n} a_i x_i$ where a_i is sampled uniformly at random from the ring $\mathcal{R} = \mathbb{Z}_p[x]/\langle x^n + 1\rangle$ and x_i are sampled from ideals in $\mathcal{R}$. Use of FFT eases implementation of polynomial products implicit in the function definition. Security considerations are based on the irreducible nature of $x^n + 1$ over the defined ring, fact that p should be prime and $p - 1$ should be a multiple of $2n$ (leads to optimised running time for FFT), and uniform distribution of a_i over the ring. Further improvements include storing initial computation of FFT on binary vectors in lookup tables, and taking better advantage of parallelism offered by modern processors.

4. Recent Advancements

4.1. Programmable Hash Funtions

The work of Zhang [24] proposed a recent construction- *programmable hash functions*- originally introduced by Hofheinz and Kiltz [25] achieving short signature schemes over bilinear maps. Other works on PHFs include Yamada *et. al.* [26] which significantly reduce the size of the public key used in Hofheinz [25] while maintaining short signatures as in the original construction, Hanaoka *et. al.*[27] which derives a lower bound on the hash key size and discuss the impossibility of existence of algebraic PHFs in prime order groups for any $n \in \mathbb{Z}$, Hofheinz [28], Freire *et. al.* [29] which treat PHFs in multi-linear settings and in noisy multi-linear maps, Catalano *et. al.* [30] which introduces asymmetric PHFs that include two main new ideas: publicly computable isomorphic copy of the function and embedding a separate pseudo-random value as the output of the function for constructing signatures with shorter public keys. Most of these constructions are based on groups where the *discrete logarithm* problem (DL) is hard.

Formally, PHFs are keyed group hash functions over a finite group $\mathbb{G}$ that behave in statistically indistinguishable ways based on the mode of key generation: *normal* mode such that $h: \{0,1\}^m \to \mathbb{G}$ and a *trapdoor* mode where additional information $a, b \in \mathbb{Z}$ is obtained such that for pre-fixed $g, h \in \mathbb{G}$, $h: \{0,1\}^m = g^a.h^b$. Given the hardness of the DL problem, it is straightforward to note the hardness of inversing this additional trapdoor mode information outputted.

Zhang *et. al.* [24] derives lattice-based PHFs that are collision resistant under *inhomogeneous small integer solutions* or ISIS hardness assumption. Such hash functions are no longer group hash functions, but retain the statistically indistinguishable operations under the mode of key generation. On lattices, the normal operation of the hash function is given as $h(\mathbf{x}) \in \mathbb{Z}_q^{n\times m}$ for usual n, m, q in the definition of lattices. While the trapdoor operation involves outputting trapdoor information $\mathbf{A}$ and $\mathbf{B}$ such that for any *generators* $\mathbf{P} \in \mathbb{Z}_q^{n\times p}$ and $\mathbf{Q} \in \mathbb{Z}_q^{n\times m}$, $h(x) = \mathbf{PA} + \mathbf{BQ}$. Under dimensionality checks, it is straightforward to see $\mathbf{A} \in \mathbb{Z}_p^{p\times m}$ and $\mathbf{B} \in \mathbb{Z}_q^{n\times n}$. The generator $\mathbf{P}$ embeds the ISIS problem while the generator $\mathbf{Q}$ embeds a variant of short vector search problem, leading to provable hardness of the described hash function family.

4.2. On Gröbner bases and Ideal lattices

Francis *et. al* [31] developed a construction of hash function based on Gröbner bases and ideal lattices. The hardness of the construction is based on the *Smallest Polynomial Problem- SPP*- for multivariate ideal lattices. In the univariate case, *SPP* is reducible to *Shortest Conjugate Problem- SCP*- whose hardness is based on the isomorphism of number fields. In the multivariate case however, the authors introduce another problem named *Smallest Substitution Problem- SSP* whose hardness is based on determining if two functional fields are isomorphic, and *SCP* can be polynomially reduced to SSP, thereby establishing the hardness for SPP.

The generalized hash functions based on multivariate ideal lattices are defined as $\mathcal{H}(\mathcal{R}, D, m)$ where $\mathcal{R}$ is the ring $\mathbb{Z}_p[x_1, x_2, ..., x_n]/I$ such that p is of the order n^2. D is a properly chosen subset of $\mathcal{R}$ having $\{g \in \mathcal{R}\}$ such that the norm of g with respect to the ideal I is upper bounded by a parameter d. The security of the generated hash functions requires the ideal I in a finitely generated residue class polynomial ring $\mathbb{Z}[x_1, x_2, ..., x_n]/I$ to be a prime ideal, which forces the generated multivariate lattice to be full rank or *lattice rank = lattice dimension* (this is exactly like the full rank requirement of univariate lattices that aids in preventing development of collision attacks against primitives developed on such lattices). There are other additional conditions for which we refer the reader to the original paper. For the parameter $m \geq \frac{logp}{log2d}$, the constructed hash function has collisions, and any algorithm to find such collisions is equivalent to solving approximate-*SPP*. Earlier discussion on reductions of worst case instances of these problems establishes the security of such constructions.

4.3. Chameleon Hash Functions

A recent construction by Mohassel *et.al.*[32]- Chameleon hash functions- are collision resistant hash functions dependent upon certain probability distributions and having a pair of keys: public key and private (trapdoor) key. A chameleon hash function satisfies three properties: it is computable using the public key, it is collision resistant if trapdoor is not known, and collisions are easy to find if trapdoor information is known.

Generalised chameleon hash functions are sets of three algorithms (say *Gen*, *h*, and h^{-1}). *Gen* generates the pair of keys: the public key k_{public} and the trapdoor k_{trap}. h converts a given message m to its hash, i.e. output hash = $h(m, k_{public}, r)$ where $r \in S$ and r is drawn from P where P is some probability distribution over some set S. Finally, h^{-1} outputs r' over S given $h^{-1}(m, m', k_{trap}, r)$ such that for distinct messages m and m' and $r \in S$ as discussed before, $h(m, k_{public}, r) = h(m', k_{public}, r')$.

Lattice based chameleon hash functions can be constructed in similar ways. For the security parameter k, define $\mathbf{A} \in \mathbb{Z}_q^{k \times m_1}$ and $\mathbf{B} \in \mathbb{Z}_q^{k \times m_2}$ as well as the message domain $M = \{\mathbf{x} \in \mathbb{Z}_q^{m_1} : \| \mathbf{x} \|_2 \leq \beta_1\}$ and randomness domain (discrete Gaussian) $R = \{\mathbf{x} \in \mathbb{Z}_q^{m_2} : \| \mathbf{x} \|_2 \leq \beta_2\}$. β_1 and β_2 are parameters that directly affect the length of such domains. The chameleon hash function is defined as $h(\mathbf{m}, \mathbf{r}) = \mathbf{Am} + \mathbf{Br}$ where $\mathbf{m} \in M$ and $\mathbf{r} \in R$ and output $\mathbf{y} \in Z_p^k$. The hardness of $h(m,r)$ is based on assumed hardness of *SIS* (short integer solutions) in the worst case (which in turn depends on SVP (shortest vector problem) on arbitrary lattices). Trapdoor information is a short basis for the lattice whose parity check matrix is $\mathbf{B}$ (used in $h(\mathbf{m}, \mathbf{r})$). Good choice of parameters include q as odd prime and = *poly(k)*. m_1 and m_2 (dimensions of subspaces from which $\mathbf{A}$ and $\mathbf{B}$ are sampled), must be of the order $\mathcal{O}(k\,log\,q)$.

5. Comparative analysis

Table 2 gives a comparative analysis of all the schemes discussed in this review. Key for table- CR: collision resistant, SVP: shortest vector problem, CRP: covering radius problem, SIVP: shortest independent vectors problem, GDD: guaranteed distance decoding problem, SPP: shortest polynomial problem (we refer the reader to Lyubashevsky[18] for exact definition of ε).

Table 2: Comparative analysis of the various hash function construction discussed in the paper.

Ref.	Security	Lattice used	Hardness assumption	Approximation factor γ
[1]	One-way	Integer lattice	SVP	$n^c; c > 8$
[5]	One-way and CR	Integer lattice	SVP	$n^c; c > 8$
[7]	One-way and CR	Integer lattice	SVP	$n^{4+\varepsilon}$
[9]	One-way and CR	Integer lattice	SVP	$n^{3+\varepsilon}$
[11]	One-way and CR	Integer lattice	SVP, SIVP, CRP	$\tilde{\mathcal{O}}(n)$
[12]	One-way and CR	Integer lattice	unique-SVP	$\mathcal{O}(n^{1.5})$
[17]	One-way	Cyclic lattice	GDD	$n^{1+\varepsilon}$
[18]	One-way and CR	Cyclic lattice	SPP	$\tilde{\mathcal{O}}(n)\varepsilon^2$
[19]	One-way and CR	Cyclic lattice	SVP	$\tilde{\mathcal{O}}(n)$

1. All hardness assumptions are the approximate versions of the problems depicted in the table. γ represents the approximation factor for the respective problem(s).
2. The family of one way hash functions proposed by [1] was converted into a family of collision resistant hash functions by [5]. Refer subsection 3.1 for details.
3. The hash function proposed in [17] was proved to be non-collision resistant by [18] and [19]. Refer subsection 3.2 for details.

Table 3 provides a comparative analysis of the implementation times of various hash functions discussed so far. All implementations were done using FLINT [33] in C++ on MacBook Air running macOS High Sierra (version 10.13.6) with a 1.8 GHz Intel Core i5 processor, Intel HD Graphics 6000 1536 MB graphics, and 8 GB 1600 MHz DDR3 RAM.

Table 3: Comparative analysis of the various hash function construction discussed in the paper.

Ref.	Lattice used	Parameters	Setup time (msec)	Hashing time(msec)
[1]	Integer lattice	n=500; m=(n^2); q=prime and $\mathcal{O}(n^7)$	4069.23	308.36
[5]	Integer lattice	n=500; m=(n^2); q=prime and $\mathcal{O}(n^7)$	3994.13	345.11
[9]	Integer lattice	n=50; m=$\mathcal{O}(2n)$; q=$\mathcal{O}(n^7)$; $\alpha = n^{1.5}.log\, n$	106293.06	7288.10
[12]	Integer lattice	n=2.80624304008; N=$2^{8\times n^2}$ m=$9 \times n \times n$	17779.21×10^{-3}	2786.59×10^{-3}
[18]	Cyclic lattice	n=257; $p = 2^{25}$; $\mathbf{f} = x^{256} + 1$	20063.724	8276×10^{-3}

1. *Setup time* in milliseconds denotes any pre-processing that might be necessary to begin hashing, like key generation.
2. *Hashing time* in milliseconds denotes the time to hash a given message using a given key.
3. Both the times are averaged over 100 iterations.
4. All the parameters depicted here are standard as used in respective schemes. We refer the reader to respective portions of the preceding subsections for a discussion on the parameters.
5. The cyclic lattice based hash function in [19] is similar to to that of [18] (refer subsection 3.2). We omit the details thereby.

In table 3, the implementation of [5] is $h_{\mathbf{A},\mathbf{r}}(\mathbf{x}) = \mathbf{A}\mathbf{x} + \mathbf{r}$ as defined in subsection 3.1. The implementation of [9] proceeds in the following stages: beginning with a random full rank matrix, checking if it is *almost* perfect, forming the group G as defined in respective discussion in subsection 3.1, creating the key, and hashing. The bottleneck is checking for the *almost perfect* lattice, or concretely for arbitrary full rank lattice $\mathbf{A} \in \mathbb{Z}^{n\times n}$, the check involves $\frac{2\rho}{\lambda_1}$ (where ρ denotes the covering radius of $\mathbf{A}$ and λ_1 denotes the length of the shortest vector in $\mathbf{A}$) and thus involves approximating both the covering radius as well as the shortest vector. In our implementation, the shortest vector was approximated using ideas from lemma 2.8 and Theorem 10 in [4] while the covering radius was approximated using the ideas as on Pg. 137 of [4]. Concretely, to approximate the covering radius, we choose a target vector as far away from the lattice as possible; the distance of the target vector from the lattice becomes our covering radius. Both the approximations of the shortest vector as well as the covering radius involve *LLL* reduction at the start and are within exponential approximation factors. Moreover, the idea of distance of a target vector from the lattice in covering radius approximation involves to solve *CVP* up to exponential approximation factors (we used the nearest plane algorithm for this task). Finally, forming the group G is achieved by reducing arbitrary vector in the span of $\mathbf{A}$ with respect to the lattice $\mathbf{M}$ by using the vector-modulo-basis reduction as seen in [34]. Evidently, this setup explains the high setup time requirement for this implementation.

6. Conclusion

The research community in mathematical cryptography has successfully evaded the presumed threat from quantum devices, even before the latter are constructed to a level where such threats are realised. One such cryptographic primitive- collision resistant hash functions on lattices- is described in this review. Several constructions are presented, and their advantages and disadvantages discussed. These provide a base for advancements both in the implementation of hash functions as well as their integration in richer cryptographic primitives such as digital signatures and encryption schemes. In the research timeline, these works also paved their way into our understanding of computational problems by coming up with new mathematical techniques for reductions, suggesting ways for manipulation of lattices for establishing security, providing tighter bounds on approximations of lattice based problems, and enriching the independent field of lattices in general. Several of these ideas manifested themselves in many later works on lattices and on cryptography.

References

[1] M. Ajtai. Generating hard instances of lattice problems (extended abstract). In *Proceedings of the Twenty-Eighth Annual ACM Symposium on Theory of Computing*, STOC ’96, pages 99–108, New York, NY, USA, 1996. Association for Computing Machinery.

[2] Arjen K Lenstra, Hendrik Willem Lenstra, and László Lovász. Factoring polynomials with rational coefficients. *Mathematische annalen*, 261(ARTICLE):515–534, 1982.

[3] Peter W Shor. Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer. *SIAM review*, 41(2):303–332, 1999.

[4] Daniele Micciancio and Shafi Goldwasser. *Complexity of lattice problems: a cryptographic perspective*, volume 671. Springer Science & Business Media, 2012.

[5] Oded Goldreich, Shafi Goldwasser, and Shai Halevi. Collision-free hashing from lattice problems. *IACR Cryptol. ePrint Arch.*, 1996:9, 1996.

[6] D Bernstein, J Buchmann, and Shafi Goldwasser. *Post-Quantum Cryptograhy*. Springer, 2000.

[7] Jin-Yi Cai and Ajay P Nerurkar. An improved worst-case to average-case connection for lattice problems. In *Proceedings 38th Annual Symposium on Foundations of Computer Science*, pages 468–477. IEEE, 1997.

[8] Jin-Yi Cai. Applications of a new transference theorem to ajtai's connection factor. In *Proceedings. Fourteenth Annual IEEE Conference on Computational Complexity (Formerly: Structure in Complexity Theory Conference)(Cat. No. 99CB36317)*, pages 205–214. IEEE, 1999.

[9] Daniele Micciancio. Improved cryptographic hash functions with worst-case/average-case connection. In *Proceedings of the thiry-fourth annual ACM symposium on Theory of computing*, pages 609–618, 2002.

[10] Daniele Micciancio and Panagiotis Voulgaris. A deterministic single exponential time algorithm for most lattice problems based on voronoi cell computations. *SIAM Journal on Computing*, 42(3):1364–1391, 2013.

[11] Daniele Micciancio and Oded Regev. Worst-case to average-case reductions based on gaussian measures. *SIAM Journal on Computing*, 37(1):267–302, 2007.

[12] Oded Regev. New lattice-based cryptographic constructions. *Journal of the ACM (JACM)*, 51(6):899–942, 2004.

[13] Daniele Micciancio. Almost perfect lattices, the covering radius problem, and applications to ajtai's connection factor. *SIAM Journal on Computing*, 34(1):118–169, 2004.

[14] Russell Impagliazzo and Moni Naor. Efficient cryptographic schemes provably as secure as subset sum. *Journal of cryptology*, 9(4):199–216, 1996.

[15] Miklós Ajtai and Cynthia Dwork. A public-key cryptosystem with worst-case/average-case equivalence. In *Proceedings of the twenty-ninth annual ACM symposium on Theory of computing*, pages 284–293, 1997.

[16] Wojciech Banaszczyk. New bounds in some transference theorems in the geometry of numbers. *Mathematische Annalen*, 296(1):625–635, 1993.

[17] Daniele Micciancio. Generalized compact knapsacks, cyclic lattices, and efficient one-way functions. *computational complexity*, 16(4):365–411, 2007.

[18] Vadim Lyubashevsky and Daniele Micciancio. Generalized compact knapsacks are collision resistant. In *International Colloquium on Automata, Languages, and Programming*, pages 144–155. Springer, 2006.

[19] Chris Peikert and Alon Rosen. Efficient collision-resistant hashing from worst-case assumptions on cyclic lattices. In *Theory of Cryptography Conference*, pages 145–166. Springer, 2006.

[20] Antoine Joux and Louis Granboulan. A practical attack against knapsack based hash functions. In *Workshop on the Theory and Application of of Cryptographic Techniques*, pages 58–66. Springer, 1994.

[21] Adi Shamir. A polynomial time algorithm for breaking the basic merkle-hellman cryptosystem. In *23rd Annual Symposium on Foundations of Computer Science (sfcs 1982)*, pages 145–152. IEEE, 1982.

[22] Serge Vaudenay. Cryptanalysis of the chor,Äîrivest cryptosystem. *Journal of Cryptology*, 14(2):87–100, 2001.

[23] Vadim Lyubashevsky, Daniele Micciancio, Chris Peikert, and Alon Rosen. Swifft: A modest proposal for fft hashing. In *International Workshop on Fast Software Encryption*, pages 54–72. Springer, 2008.

[24] Jiang Zhang, Yu Chen, and Zhenfeng Zhang. Programmable hash functions from lattices: short signatures and ibes with small key sizes. In *Annual international cryptology conference*, pages 303–332. Springer, 2016.

[25] Dennis Hofheinz and Eike Kiltz. Programmable hash functions and their applications. In *Annual International Cryptology Conference*, pages 21–38. Springer, 2008.

[26] Shota Yamada, Goichiro Hanaoka, and Noboru Kunihiro. Two-dimensional representation of cover free families and its applications: Short signatures and more. In *Cryptographers‚Äô Track at the RSA Conference*, pages 260–277. Springer, 2012.

[27] Goichiro Hanaoka, Takahiro Matsuda, and Jacob CN Schuldt. On the impossibility of constructing efficient key encapsulation and programmable hash functions in prime order groups. In *Annual Cryptology Conference*, pages 812–831. Springer, 2012.

[28] Dennis Hofheinz and Eike Kiltz. Programmable hash functions and their applications. *Journal of Cryptology*, 25(3):484–527, 2012.

[29] Eduarda SV Freire, Dennis Hofheinz, Kenneth G Paterson, and Christoph Striecks. Programmable hash functions in the multilinear setting. In *Annual Cryptology Conference*, pages 513–530. Springer, 2013.

[30] Dario Catalano, Dario Fiore, and Luca Nizzardo. Programmable hash functions go private: constructions and applications to (homomorphic) signatures with shorter public keys. In *Annual Cryptology Conference*, pages 254–274. Springer, 2015.

[31] Maria Francis and Ambedkar Dukkipati. On ideal lattices, gröbner bases and generalized hash functions. *Journal of Algebra and Its Applications*, 17(06):1850112, 2018.

[32] Payman Mohassel. One-time signatures and chameleon hash functions. In *International Workshop on Selected Areas in Cryptography*, pages 302–319. Springer, 2010.

[33] W. B. Hart. Fast library for number theory: An introduction. In *Proceedings of the Third International Congress on Mathematical Software*, ICMS'10, pages 88–91, Berlin, Heidelberg, 2010. Springer-Verlag. `http://flintlib.org`.

[34] Daniele Micciancio. Cryptographic functions from worst-case complexity assumptions. In *The LLL algorithm*, pages 427–452. Springer, 2009.