Improved Polynomial-Memory Algorithms via Nested Collision Search: Applications to Small Max-Norm LWE and Subset-Sum

Abul Kalam, Sudeshna Karmakar, and Santanu Sarkar

Indian Institute of Technology Madras, India
abulkalam.sunny@gmail.com, sudeshnakarmakar.dgp@gmail.com,
sarkar.santanu.bir1@gmail.com

Abstract. The Learning With Errors (LWE) problem with small max-norm secrets and errors forms the foundation of several practical lattice-based cryptographic schemes, including NTRU-based constructions and the recently standardized CRYSTALS-Kyber and CRYSTALS-Dilithium. While recent combinatorial attacks have significantly improved the asymptotic complexity of solving small max-norm LWE, the fastest known methods require prohibitively large memory, often comparable to their runtime. To address this issue, Esser *et al.* (ASIACRYPT 2023) introduced the first substantially improved polynomial-memory algorithms for recovering small max-norm LWE secrets via nested collision search.
In this work, we further improve the asymptotic complexity of polynomial-memory attacks on small max-norm LWE. We introduce a refined nested collision search framework together with new representation structures that increase the number of valid representations while preserving polynomial-memory complexity. This leads to improved asymptotic runtimes across the entire range of relative secret weights. For uniformly random ternary secrets of length n, our best algorithm reduces the polynomial-memory runtime from $2^{0.926n}$ to $2^{0.8595n}$, corresponding to an asymptotic speedup of approximately $2^{0.0665n}$. We further extend our framework to the secret distributions used in CRYSTALS-Kyber and CRYSTALS-Dilithium, obtaining improved complexity exponents.
Additionally, we apply the proposed refinement to the polynomial-memory nested collision search framework of Esser and May (EUROCRYPT 2020) for random subset-sum problem, reducing its asymptotic runtime from $2^{0.645n}$ to $2^{0.6432n}$. Although the latter improvement is modest, it provides an independent application of our framework and demonstrates its applicability beyond the LWE setting.

Keywords: Learning With Errors · small max-norm LWE · subset-sum · polynomial-memory · collision search

1 Introduction

Lattice-based cryptography has emerged as one of the leading paradigms for post-quantum public-key cryptography. Among the underlying hardness assumptions,

the Learning With Errors (LWE) problem has become one of the most central and versatile foundations for constructing quantum-resistant cryptographic primitives. Introduced by Regev [38], the LWE problem enjoys strong theoretical foundations through reductions from worst-case lattice problems, providing compelling evidence for its hardness. In the LWE problem, one is given a uniformly random matrix $\mathbf{A} \in \mathbb{Z}_q^{m \times n}$ together with a vector $\mathbf{b} = \mathbf{A}\mathbf{s} + \mathbf{e} \bmod q$, where $\mathbf{s} \in \mathbb{Z}_q^n$ is a secret vector and $\mathbf{e} \in \mathbb{Z}_q^m$ is a short error vector sampled from an appropriate distribution. The objective is to recover the secret vector $\mathbf{s}$ from $(\mathbf{A}, \mathbf{b})$.

The LWE problem, together with its ring and module variants, has become a fundamental building block for a wide range of cryptographic constructions, including encryption schemes, digital signatures, fully homomorphic encryption, identity-based encryption, and many other advanced primitives [19, 20, 38]. In particular, several lattice-based schemes selected in the NIST post-quantum standardization process rely directly on the presumed hardness of LWE-type problems.

Classical security reductions for LWE typically assume that the error vector $\mathbf{e}$, and often also the secret vector $\mathbf{s}$, is sampled from a discrete Gaussian distribution. In practice, however, many implementations employ significantly smaller and more structured secret distributions in order to improve efficiency. The size and structure of the secret key influence not only the security level of the scheme but also its computational and implementation costs. This has led to the widespread adoption of small max-norm variants of LWE, in which the secret and error vectors are sampled from distributions with small coefficient magnitudes, including binary, ternary, and quinary distributions.

LWE-based schemes with small max-norm secrets. A particularly important instance is ternary LWE, where the entries of both $\mathbf{s}$ and $\mathbf{e}$ are restricted to max-norm 1. This variant is widely used in practical lattice-based cryptography, including NTRU-based constructions [1, 5, 25, 28] and signature schemes such as BLISS [12] and GLP [22]. Furthermore, many practical schemes employ secret and error distributions with small max-norm. For example, the recently standardized NIST scheme CRYSTALS-Kyber [7] samples secrets and errors from centered binomial distributions with parameter $\eta \in \{2, 3\}$, while CRYSTALS -Dilithium [13] uses secret distributions supported on small coefficients of max-norm 2.

State-of-the-Art cryptanalysis of small max-norm LWE. The reduced search space induced by small max-norm secret distributions enables combinatorial attacks that exploit the structure of the secret vector by enumerating candidate representations. One of the earliest approaches is Odlyzko's meet-in-the-middle attack [25], achieving runtime approximately $|\mathcal{D}|^{1/2}$, where $\mathcal{D}$ denotes the secret search space. Later, Howgrave-Graham [26] introduced the hybrid attack framework, combining combinatorial search with lattice reduction by guessing part of the secret and solving the remaining instance using lattice techniques. Hybrid attacks subsequently became the dominant paradigm for small max-norm LWE cryptanalysis. More recently, May [32] adapted advanced subset-sum techniques [3, 6, 27] to small max-norm LWE, reducing the asymptotic

complexity for ternary LWE from roughly $|\mathcal{D}|^{1/2}$ to $|\mathcal{D}|^{1/4}$. These techniques were later extended to the secret distributions used in KYBER and DILITHIUM [21]. However, the fastest known combinatorial attacks require enormous memory, typically comparable to their runtime, which substantially limits their practical relevance. To address this issue, Esser *et al.* [14] introduced memory-efficient nested collision search algorithms for recovering small max-norm LWE keys. Their best instantiation, NESTED-2*, achieves runtime approximately $|\mathcal{D}|^{0.58}$ while requiring only polynomial-memory, improving the previous polynomial-memory complexity for uniformly random ternary secrets from $2^{1.063n}$ to $2^{0.926n}$. They further extended this framework in [16] by establishing continuous time–memory trade-offs ranging from negligible memory to the unrestricted-memory regime.

Subset-Sum-based Schemes and State-of-the-Art Cryptanalysis. The subset-sum problem, also known as the knapsack problem, is one of the most fundamental and well-studied NP-hard problems [23]. Given $a_1, \ldots, a_n, t \in \mathbb{Z}_{2^n}$, the subset-sum problem asks to find a subset $S \subseteq \{1, \ldots, n\}$ of a prescribed size such that $\sum_{i \in S} a_i = t \bmod 2^n$. The problem was among the early candidates for constructing public-key cryptosystems in the 1980s [10,24]. However, several such constructions were subsequently broken using lattice-based techniques [30,35]. More recently, subset-sum has regained attention in cryptography [18,31], in part because important lattice- and code-based problems admit formulations as vectorial variants of subset-sum. In particular, instances of LWE and SIS [39], as well as LPN [2], can be viewed through the lens of subset-sum.

This work focuses on the random subset-sum problem (RSSP) at density one, that is, $|S| = n/2$. A significant breakthrough was achieved by Howgrave-Graham and Joux (HGJ) [27] at EUROCRYPT 2010, who proposed a heuristic algorithm with time complexity $\tilde{\mathcal{O}}(2^{0.337n})$ and space complexity $\tilde{\mathcal{O}}(2^{0.311n})$. Their approach represents the solution as a sum of vectors from $\{0,1\}^n$, thereby breaking the long-standing $\tilde{\mathcal{O}}(2^{n/2})$ time-complexity barrier. Subsequently, Becker, Coron, and Joux (BCJ) [3] further reduced both the time and space complexities to $\tilde{\mathcal{O}}(2^{0.291n})$. More recently, Bonnetain, Bricout, Schrottenloher, and Shen (BBSS) [6] further improved the time and space complexities to $\tilde{\mathcal{O}}(2^{0.283n})$ by relaxing some of the constraints in the collision search framework. These representation techniques were subsequently adapted to the syndrome decoding problem [4,8,33].

While these algorithms achieve substantially lower runtimes, they require exponential memory. This motivates the study of subset-sum algorithms with polynomial-memory. In this direction, Esser and May [15] introduced, at EUROCRYPT 2020, a nested collision search framework for the RSSP that achieves a runtime of $\tilde{\mathcal{O}}(2^{0.645n})$ using polynomial-memory. More recently, Esser and Zweydinger [17] investigated time–memory trade-offs for the RSSP, providing a broader range of algorithms that interpolate between different time and memory complexities.

In this work, we develop an improved nested collision search framework for polynomial-memory attacks on small max-norm LWE and the random subset-sum problem (RSSP), building upon the frameworks introduced by Esser *et al.* [14] and Esser and May [15].

1.1 Our Contributions

For completeness and to establish the necessary foundations, we first revisit the nested collision search framework for ternary LWE introduced by Esser *et al.* [14], together with their best polynomial-memory instantiation, Nested-$2^\star$. Throughout the paper, we refer to this framework as EGMS. These constructions serve both as the starting point for our developments and as the baseline for all subsequent comparisons.

The central contribution of this work is a representation-entropy principle for nested collision search. Rather than balancing the projection size solely against the cardinality of the secret domain, we exploit the fact that a larger projection space permits larger intermediate representation domains. This increases the number of valid representations at the intermediate levels, which can reduce the overall search cost even though the collision search cost itself increases. We formulate this trade-off explicitly and incorporate it into the complexity optimization.

As a first application of the framework, we incorporate the Nested-$2^\star$ instantiation of [14] into our improved setting, which we call Improved-Nested-$2^\star$. We show that this modification yields a consistent reduction in the asymptotic complexity exponent across the entire range of relative secret weights. We then introduce a new shifted representation technique, leading to the Shifted-3 instantiation. The central idea is to shift a portion of the secret vector from the ternary distribution $\{-1, 0, 1\}$ to the asymmetric distribution $\{0, 1, 2\}$. This transformation creates a more favorable representation structure and leads to further nontrivial complexity improvements, particularly in the cryptographically relevant parameter regime. Finally, we combine the advantages of both approaches into a unified Hybrid instantiation, which simultaneously incorporates the Improved-Nested-$2^\star$ and Shifted-3 strategies. This hybrid construction achieves the best asymptotic complexity currently known across the full range of relative secret weights. Figure 1 compares the optimized runtime exponents $\frac{\log T}{n}$ of our framework with those of Esser *et al.* [14]. The figure shows that our approach consistently improves the complexity exponent across the entire parameter range. For the balanced ternary secret case $\omega = 0.667$, Figure 2 illustrates the progression of the best known polynomial-memory complexity exponents. Our framework reduces the exponent for Nested-$2^\star$ [14] from 0.926 to 0.8595, corresponding to an asymptotic improvement of approximately $2^{0.0665n}$.

We further extend our framework beyond ternary secrets to the secret distributions used in Kyber and Dilithium. For the Kyber secret distribution, our methods improve the complexity from $|\mathcal{D}|^{0.513}$ to $|\mathcal{D}|^{0.507}$, and from $|\mathcal{D}|^{0.508}$ to $|\mathcal{D}|^{0.505}$, depending on the parameter set.

As an additional application, we revisit the polynomial-memory nested collision search algorithm of Esser and May [15] for the random subset-sum problem. By incorporating the enlarged projection spaces into their construction, we obtain additional representations at the intermediate collision levels. The resulting complexity expression can then be optimized over the parameters governing the weight distribution and projection size. For the balanced RSSP, this optimiza-

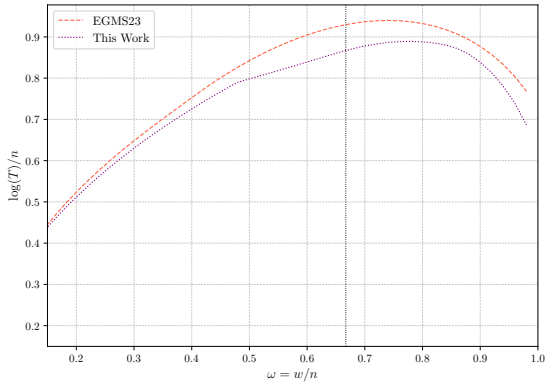
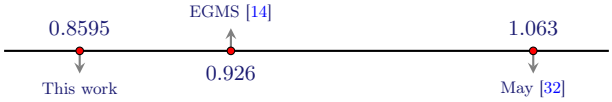


Fig. 1. Comparison of the optimized asymptotic runtime exponents of NESTED-2* [14] and the proposed framework as a function of the relative secret weight $\omega = w/n$.

Fig. 2. Comparison of the asymptotic runtime exponents for balanced-weight ternary secret LWE under polynomial-memory constraints.



tion improves the polynomial-memory runtime exponent from 0.645 to 0.6432. Although the resulting improvement in the exponent is modest, this application demonstrates that the proposed refinement is not specific to the LWE setting and can also be applied to other nested collision search frameworks.

Our work relies on the same heuristic assumptions introduced and experimentally validated in [14, 15]; we do not introduce any additional heuristic assumptions beyond those underlying the original frameworks. Nevertheless, to provide further evidence for the validity of our analysis, we conduct toy-scale experiments for small max-norm LWE in Appendix C. These experiments examine the random-projection and representation-count heuristics underlying our complexity analysis. All source codes and experimental scripts accompanying this work are available at `https://github.com/AnonymousCryptoSub/Improved-Polynomial-Memory-Algorithms-via-Nested-Collision-Search`.

Paper Layout. Section 2 introduces the necessary preliminaries, including collision search techniques, the ternary LWE problem and the random subset-sum problem. In Section 3, we revisit the EGMS framework and discuss their NESTED-2* instantiation. Section 4 presents our improved framework together with several new instantiations. In Section 5, we analyze the asymptotic complexity of our algorithms and compare them with the EGMS framework for ternary secret LWE. Section 6 extends our techniques to the secret distributions used in CRYSTALS-KYBER and CRYSTALS-DILITHIUM. Section 7 applies our refinement to the

polynomial-memory subset-sum algorithm of Esser and May. Finally, Section 8 concludes the paper.

2 Preliminaries

We begin by introducing the notation and background required throughout the paper. Since our work builds upon the frameworks of [14] and [15], we largely adopt their notation for consistency and ease of comparison.

2.1 Collision Search and the Rho Method

A collision in an n-to-n bit function $f : S \to S$ is a pair $(y_1, y_2) \in S^2$ with $y_1 \neq y_2$ and $f(y_1) = f(y_2)$, and is known to be computable with roughly $2^{n/2}$ evaluations of f and polynomial-memory [37]. Standard collision search algorithms [29, 34, 37] achieve this by iterating f from a seed $x \in S$, tracing the orbit $f(x), f^2(x), f^3(x), \ldots$ until a cycle is detected via a linear-space routine such as Floyd's or Brent's algorithm [9]. If $f^k(x)$ denotes the first repeated element and $f^{k+l}(x)$ its recurrence, then $f^{k-1}(x)$ and $f^{k+l-1}(x)$ are distinct preimages of $f^k(x)$ under the function f, constituting a collision.

To find a collision across two independent functions $f_1, f_2 : S \to S$, i.e., a pair (x_1, x_2) with $f_1(x_1) = f_2(x_2)$, van Oorschot and Wiener [36] introduced a merged function $F : S \to S$ governed by a random selector $g : S \to \{0, 1\}$:

$$F(x) := \begin{cases} f_1(x) & \text{if } g(x) = 0, \\ f_2(x) & \text{if } g(x) = 1. \end{cases}$$

A collision in F yields a valid cross-collision between f_1 and f_2 with probability $\frac{1}{2}$; else the seed is deterministically updated and the search restarts, incurring only a constant factor overhead in expectation. We write $\text{Rho}(f_1, f_2, x) := (x_1, x_2)$ for the cross-collision reachable from x, at cost $\mathcal{O}(\sqrt{|S|})$ evaluations of F.

Flavors and independence. Since the Rho-walk is deterministic, all seeds on the tail of a given orbit collapse to the same collision, creating dependencies when Rho is invoked repeatedly [3, 11]. To address this, we follow prior work [3, 11] and introduce *flavors*: a keyed family $f^{[t]}$ of parametric perturbations of f indexed by t, such that calls under distinct flavors are heuristically independent. All invocations of Rho in this work are implicitly parameterized by a fresh flavor.

Definition 1 (Flavor of a Function). *Let $\Pi = \{P_t\}_{t \in \mathbb{N}}$ be a permutation family acting on S. Given $f : S \to S$, we associate to each index t a perturbed variant $f^{[t]} : S \to S$ via:*

$$f^{[t]} := P_t \circ f.$$

We refer to $f^{[t]}$ as the t-th flavor of f relative to Π.

Flavoring preserves the collision set of f exactly. This follows immediately from the invertibility of P_t: two points $y_1, y_2 \in S$ satisfy $f^{[t]}(y_1) = f^{[t]}(y_2)$ precisely when $f(y_1) = f(y_2)$, since P_t, being a bijection, acts as a relabeling of the output space of f. Hence, the preimage partition induced by $f^{[t]}$ is identical to that of f, and the two functions share precisely the same collision set.

2.2 Complexity Notation

Before describing our algorithms, we fix the asymptotic framework used throughout. All runtimes are expressed in $\tilde{\mathcal{O}}$-notation, which suppresses factors polylogarithmic in n. Counting arguments rely heavily on the following approximation for multinomial coefficients, obtained via Stirling's formula:

$$\binom{n}{k_1 n,\ \ldots,\ k_p n} \;=\; \tilde{\mathcal{O}}\Big(2^{nH(k_1,\ldots,k_p)}\Big),$$

where $H(k_1, \ldots, k_p) = -\sum_{i=1}^{p} k_i \log_2 k_i$ is the Shannon entropy of the distribution $(k_1, \ldots, k_p)$ satisfying $\sum_i k_i = 1$. Whenever the last parameter k_p is determined by the remaining ones, we adopt the shorthand $\binom{n}{k_1 n, \ldots, k_{p-1} n, .}$ to avoid redundancy.

2.3 LWE with Ternary Secrets

An LWE instance $(\mathbf{A}, \mathbf{b})$ with $\mathbf{A} \in \mathbb{Z}_q^{n \times n}$ and $\mathbf{b} \in \mathbb{Z}_q^n$ is constructed by fixing a secret $\mathbf{s}$ and an error $\mathbf{e}$, and setting $\mathbf{As} = \mathbf{b} + \mathbf{e} \bmod q$. We work in the regime where $\mathbf{s}, \mathbf{e} \in \{-1, 0, 1\}^n$, which is the prescribed key format in prominent NTRU-type schemes including NTRU, NTRU Prime, GLP, and BLISS [5,12,22,25]. The modulus is fixed at $q = \mathrm{poly}(n)$, a range that is cryptographically hard [39], and all entries of $\mathbf{A}$ are treated as independent uniform samples from $\mathbb{Z}_q$, without exploiting any ring structure.

Definition 2 (Ternary LWE). *Let $n \in \mathbb{N}$, $q = \mathrm{poly}(n)$, and $w \leqslant n$. Given $(\mathbf{A}, \mathbf{b}) \in \mathbb{Z}_q^{n \times n} \times \mathbb{Z}_q^n$, the* ternary LWE problem *asks to recover a vector $\mathbf{s} \in \{-1, 0, 1\}^n$ of Hamming weight w together with some $\mathbf{e} \in \{-1, 0, 1\}^n$ satisfying:*

$$\mathbf{As} \;=\; \mathbf{b} + \mathbf{e} \bmod q.$$

We additionally confine attention to *balanced* solutions, in which the w nonzero coordinates of $\mathbf{s}$ are evenly divided between $+1$ and -1. This balance condition is built into the key generation of most NTRU-type constructions; when it is not, one can enumerate all possible splits at a polynomial overhead. Setting $w = \omega n$, the resulting search space is:

$$\tau^n(w/2) \;:=\; \big\{\mathbf{s} \in \{0, \pm 1\}^n \;\big|\; |\mathbf{s}|_{-1} = |\mathbf{s}|_1 = w/2\big\},$$

whose cardinality satisfies: $\big|\tau^n(w/2)\big| \;=\; \binom{n}{w/2, w/2, n-w} = \tilde{\mathcal{O}}\big(2^{H(\omega/2,\, \omega/2,\, 1-\omega)\cdot n}\big)$.

2.4 Random Subset-Sum Problem (RSSP)

Definition 3 (Random Subset-Sum). *Let $n \in \mathbb{N}$ and let $\mathbf{a} = (a_1, \ldots, a_n) \in (\mathbb{Z}_{2^n})^n$ be sampled uniformly at random. Let $\mathbf{e} \in \{0,1\}^n$ be sampled uniformly from the set of vectors of Hamming weight w. Define $t = \langle \mathbf{a}, \mathbf{e} \rangle = \sum_{i=1}^{n} a_i e_i \bmod 2^n$. The corresponding random subset-sum instance $(\mathbf{a}, t, w)$ asks to recover a vector $\mathbf{e}' \in \{0,1\}^n$ of Hamming weight w such that*

$$\langle \mathbf{a}, \mathbf{e}' \rangle = t \bmod 2^n.$$

Following [15], we focus on the balanced case $w = n/2$. This choice corresponds to the largest binary search space and therefore provides the most challenging setting for our analysis. Accordingly, we define

$$\mathcal{B}^n(\omega) := \{\mathbf{v} \in \{0,1\}^n \mid \mathrm{wt}(\mathbf{v}) = \omega n\}, \qquad 0 < \omega < 1.$$

Its cardinality is $|\mathcal{B}^n(\omega)| = \binom{n}{\omega n} = \tilde{\mathcal{O}}\left(2^{nH_2(\omega)}\right)$, where $H_2(\omega) := H(\omega, \cdot)$ denotes the binary entropy function.

3 The EGMS Nested Collision Search Framework

In this section, we revisit the nested collision search framework introduced by Esser, Girme, Mukherjee and Sarkar (EGMS) [14], focusing in particular on their most efficient variant, namely Nested-2*. Below, we briefly describe the underlying algorithm and its main ideas.

3.1 Nested Collision Search

Recall the ternary LWE relation

$$\mathbf{As} = \mathbf{b} + \mathbf{e} \bmod q, \tag{1}$$

where $\mathbf{A} \in \mathbb{Z}_q^{n \times n}$ and $\mathbf{b} \in \mathbb{Z}_q^n$ are public, while $\mathbf{s} \in \mathbb{Z}_q^n$ and $\mathbf{e} \in \mathbb{Z}_q^n$ denote the ternary secret of weight w and ternary error vectors, respectively. The nested collision search algorithm begins by decomposing the secret vector into four components,

$$\mathbf{s} = \underbrace{\mathbf{s}_1 + \mathbf{s}_2}_{\mathbf{a}_1} + \underbrace{\mathbf{s}_3 + \mathbf{s}_4}_{\mathbf{a}_2}.$$

Substituting this decomposition into (1) yields

$$\mathbf{A}(\mathbf{s}_1 + \mathbf{s}_2) = \mathbf{b} + \mathbf{e} - \mathbf{A}(\mathbf{s}_3 + \mathbf{s}_4) \bmod q. \tag{2}$$

The attack proceeds by guessing the first 2ℓ coordinates of the error vector $\mathbf{e}$. Since the error entries are sampled from the ternary distribution, this can be achieved via exhaustive search over these coordinates. Once these values are fixed,

the corresponding coordinates of the noisy LWE equation become deterministic, giving rise to the projected relation

$$\pi_{2\ell}(\mathbf{A}(\mathbf{s}_1 + \mathbf{s}_2)) = \mathbf{b}' - \pi_{2\ell}(\mathbf{A}(\mathbf{s}_3 + \mathbf{s}_4)), \tag{3}$$

where $\mathbf{b}' = \pi_{2\ell}(\mathbf{b}+\mathbf{e})$, and $\pi_{2\ell}$ denotes the projection onto the first 2ℓ coordinates. The parameter ℓ is chosen so that the projected system $\pi_{2\ell}(\mathbf{As}) = \mathbf{b}'$ admits, heuristically, a unique solution in the secret domain. Assuming that $\mathbf{A}$ behaves as a random matrix over $\mathbb{Z}_q$, this condition is equivalent to requiring

$$\frac{|\mathcal{D}|}{q^{2\ell}} = 1, \tag{C1}$$

where $\mathcal{D}$ denotes the set of all possible secret vectors[1]. Consequently, the choice of ℓ satisfies $\ell = \frac{1}{2}\log_q |\mathcal{D}|$.

The nested collision search algorithm further introduces an intermediate collision layer. More precisely, a vector $\mathbf{r} \in \mathbb{Z}_q^\ell$ is sampled uniformly at random and the lower ℓ coordinates of (2) are fixed according to

$$\pi_\ell(\mathbf{A}(\mathbf{s}_1 + \mathbf{s}_2)) = \mathbf{r}.$$

Substituting this condition into (2) yields the two relations

$$\pi_\ell(\mathbf{As}_1) = \mathbf{r} - \pi_\ell(\mathbf{As}_2), \tag{4}$$

and

$$\pi_\ell(\mathbf{As}_3) = \pi_\ell(\mathbf{b}') - \mathbf{r} - \pi_\ell(\mathbf{As}_4). \tag{5}$$

Esser *et al.* refer to (4) and (5) as the *Layer-1* identities. Solving these identities produces valid pairs $(\mathbf{s}_1, \mathbf{s}_2)$ and $(\mathbf{s}_3, \mathbf{s}_4)$. The projected relation (3) is then viewed as the *Layer-2* identity, whose solution combines the Layer-1 outputs into a valid representation of the secret vector.

To solve the Layer-1 identities, the following functions

$$f_i : \mathcal{T}_i \longrightarrow \mathbb{Z}_q^\ell, \qquad i \in \{1, 2, 3, 4\},$$

are defined:

$$f_1, f_3 : \mathbf{x} \mapsto \pi_\ell(\mathbf{Ax}), \quad f_2 : \mathbf{x} \mapsto \mathbf{r} - \pi_\ell(\mathbf{Ax}), \quad f_4 : \mathbf{x} \mapsto \pi_\ell(\mathbf{b}') - \mathbf{r} - \pi_\ell(\mathbf{Ax}), \tag{6}$$

where $\mathcal{T}_i$ denotes the domain of $\mathbf{s}_i$ for each $i \in \{1, 2, 3, 4\}$.

Any decomposition

$$\mathbf{s} = \mathbf{s}_1 + \mathbf{s}_2 + \mathbf{s}_3 + \mathbf{s}_4$$

[1]Recall from Section 2.3 that, in the ternary setting, the secret vector typically satisfies $\mathbf{s} \in \tau^n(w/2)$. However, in the NESTED-2* variant of Esser *et al.*, the search space is restricted to a suitable subset of $\tau^n(w/2)$. To keep the discussion sufficiently general and independent of the specific choice of search space, we denote the restricted secret domain of the secret vector abstractly by $\mathcal{D}$ throughout this paper.

satisfying the Layer-2 identity necessarily induces valid Layer-1 pairs. Consequently, the pairs $(\mathbf{s}_1, \mathbf{s}_2)$ and $(\mathbf{s}_3, \mathbf{s}_4)$ can be recovered through collision search between the function pairs (f_1, f_2) and (f_3, f_4), respectively.

To resolve the Layer-2 identity, the following second-layer functions are introduced:

$$\begin{aligned} g_1 &: \mathbf{x} \mapsto \vartheta_\ell\left(\mathbf{A}(\mathbf{z}_1 + \mathbf{z}_2)\right), & (\mathbf{z}_1, \mathbf{z}_2) &\leftarrow \textsc{Rho}(f_1^{[\mathbf{x}]}, f_2^{[\mathbf{x}]}, \mathbf{x}) \\ g_2 &: \mathbf{x} \mapsto \vartheta_\ell(\mathbf{b}') - \vartheta_\ell\left(\mathbf{A}(\mathbf{z}_3 + \mathbf{z}_4)\right), & (\mathbf{z}_3, \mathbf{z}_4) &\leftarrow \textsc{Rho}(f_3^{[\mathbf{x}]}, f_4^{[\mathbf{x}]}, \mathbf{x}), \end{aligned} \tag{7}$$

where each f_i is flavored via the starting point and ϑ_ℓ denotes the projection onto coordinates indexed from $\ell + 1$ to 2ℓ, i.e.,

$$\vartheta_\ell((x_1, \ldots, x_k)) = (x_{\ell+1}, \ldots, x_{2\ell}), \qquad \text{where } k \geqslant 2\ell.$$

A collision between the functions g_1 and g_2 therefore yields a solution to the Layer-2 identity over all 2ℓ projected coordinates.

Observe that, in order to apply Pollard's Rho collision search to the pairs (f_1, f_2), (f_3, f_4), and subsequently (g_1, g_2), the corresponding domains and codomains must have comparable cardinalities. This leads to the condition

$$q^\ell = |\mathcal{T}_1|, \qquad |\mathcal{T}_i| = |\mathcal{T}_j| \quad \text{for all } 1 \leqslant i, j \leqslant 4.$$

Combining this with the choice of ℓ obtained above yields

$$|\mathcal{T}_i| = q^\ell = |\mathcal{D}|^{1/2}, \qquad 1 \leqslant i \leqslant 4. \tag{C2}$$

A pseudocode description of the algorithm is given in Algorithm 1.

Algorithm 1: Nested Collision Search

Input: $(\mathbf{A}, \mathbf{b}) \in \mathbb{Z}_q^{n \times n} \times \mathbb{Z}_q^n$ and weight of the secret $w \leqslant n$

Set $\ell \leftarrow \frac{1}{2} \log_q |\mathcal{D}|$;
repeat
  Randomly select permutation matrix $\mathbf{P}$;
  Set $\mathbf{A}' \leftarrow \mathbf{AP}$;
  Randomly select $\pi_{2\ell}(\mathbf{e}) \in \{-1, 0, 1\}^{2\ell}$;
  Set $\mathbf{b}' \leftarrow \pi_{2\ell}(\mathbf{b} + \mathbf{e})$;
  Randomly select $\mathbf{r} \in \mathbb{Z}_q^\ell$;
  Define the layer functions f_i and g_j as in (6) and (7) using $\mathbf{A}'$, $\mathbf{b}'$ and $\mathbf{r}$;
  Choose random flavor for g_1 and g_2;
  Choose random starting point $\mathbf{y}$ and compute $(\mathbf{y}_1, \mathbf{y}_2) \leftarrow \textsc{Rho}(g_1, g_2, \mathbf{y})$;
  Compute $(\mathbf{z}_1, \mathbf{z}_2) \leftarrow \textsc{Rho}(f_1, f_2, \mathbf{y}_1)$;
  Compute $(\mathbf{z}_3, \mathbf{z}_4) \leftarrow \textsc{Rho}(f_3, f_4, \mathbf{y}_2)$;
  Set $\tilde{\mathbf{s}} \leftarrow \mathbf{z}_1 + \mathbf{z}_2 + \mathbf{z}_3 + \mathbf{z}_4$ and $\mathbf{s}^\star \leftarrow \mathbf{P}\tilde{\mathbf{s}}$;
until $\tilde{\mathbf{s}} \in \mathcal{D}$ *and* $\mathbf{As}^\star - \mathbf{b} \in \{-1, 0, 1\}^n$;
return $\mathbf{s}^\star$

Remark 1. More precisely, the maps $f_i : \mathcal{T}_i \to \mathbb{Z}_q^\ell$ are not endomaps. To implement the RHO iteration, we choose a bijection $\iota_i : \mathbb{Z}_q^\ell \to \mathcal{T}_i$ whenever $q^\ell = |\mathcal{T}_i|$. The actual maps iterated by the RHO procedure are therefore endomaps on $\mathbb{Z}_q^\ell$, namely

$$F_i = f_i \circ \iota_i : \mathbb{Z}_q^\ell \to \mathbb{Z}_q^\ell.$$

For simplicity of notation, we nevertheless continue to write $\text{RHO}(f_i, f_j, \cdot)$.

Complexity Analysis. The overall complexity of Algorithm 1 is obtained by multiplying the cost of a single iteration with the expected number of repetitions required for success. We first analyze the number of repetitions.

For the algorithm to succeed, the following events must simultaneously hold. First, the secret vector must attain the desired weight distribution. This is achieved by iterating the algorithm over independent random permutations $\mathbf{P}$ (see Line 4 of Algorithm 1). Let $\tilde{\mathbf{s}} := \mathbf{P}^{-1}\mathbf{s}$. We define the following events:

- $E_{\mathbf{P}}$: the event that $\tilde{\mathbf{s}}$ belongs to the restricted domain $\mathcal{D}$;
- $E_{\mathbf{e}}$: the event that the guessed 2ℓ coordinates of the error vector $\mathbf{e}$ are correct;
- $E_{\mathbf{r}}$: the event that, for the randomly chosen value of $\mathbf{r}$, there exists a representation $(\mathbf{a}_1, \mathbf{a}_2)$ of $\tilde{\mathbf{s}}$ satisfying $\pi_\ell(\mathbf{A}'\mathbf{a}_1) = \mathbf{r}$;
- $E_{\tilde{\mathbf{s}}}$: the event that the tuple $(\mathbf{z}_1, \mathbf{z}_2, \mathbf{z}_3, \mathbf{z}_4)$, obtained in Lines 11–12 of Algorithm 1, forms a valid representation of $\tilde{\mathbf{s}}$.

The algorithm succeeds only if all of the above events occur simultaneously. Hence, the expected number of repetitions required for success is $\mathbb{P}[E_{\mathbf{P}} \cap E_{\mathbf{e}} \cap E_{\mathbf{r}} \cap E_{\tilde{\mathbf{s}}}]^{-1}$. By the chain rule, this can be rewritten as

$$\left(\mathbb{P}[E_{\mathbf{P}}] \cdot \mathbb{P}[E_{\mathbf{e}} \mid E_{\mathbf{P}}] \cdot \mathbb{P}[E_{\mathbf{r}} \mid E_{\mathbf{P}} \cap E_{\mathbf{e}}] \cdot \mathbb{P}[E_{\tilde{\mathbf{s}}} \mid E_{\mathbf{P}} \cap E_{\mathbf{e}} \cap E_{\mathbf{r}}]\right)^{-1}.$$

We now evaluate each term individually.

The probability that $\tilde{\mathbf{s}}$ satisfies the desired weight distribution, equivalently $\tilde{\mathbf{s}} \in \mathcal{D}$, is

$$\mathbb{P}[E_{\mathbf{P}}] = \frac{|\mathcal{D}|}{|\tau^n(w/2)|}.$$

Conditioned on $E_{\mathbf{P}}$, the probability of correctly guessing the 2ℓ coordinates of the error vector is

$$\mathbb{P}[E_{\mathbf{e}} \mid E_{\mathbf{P}}] = 3^{-2\ell}.$$

Since $\ell = \frac{1}{2}\log_q |\mathcal{D}|$, $q = \text{poly}(n)$, and $|\mathcal{D}| = 2^{cn}$ for some constant $c > 0$, we obtain $3^{-2\ell} = 2^{-\Theta\left(\frac{n}{\log n}\right)}$.

Let R_1 denote the number of middle-layer representations $(\mathbf{a}_1, \mathbf{a}_2)$ of the secret $\mathbf{s}$ and let R_2 denote the number of lower-layer representations $(\mathbf{s}_1, \mathbf{s}_2)$ (resp. $(\mathbf{s}_3, \mathbf{s}_4)$) of $\mathbf{a}_1$ (resp. $\mathbf{a}_2$). By the randomness heuristic on $\mathbf{A}'$, the probability that a suitable representation $(\mathbf{a}_1, \mathbf{a}_2)$ satisfies the Layer-1 identity ($\pi_\ell(\mathbf{A}'\mathbf{a}_1) = \mathbf{r}$) is

$$\mathbb{P}[E_{\mathbf{r}} \mid E_{\mathbf{P}} \cap E_{\mathbf{e}}] = \frac{R_1}{q^\ell}.$$

For this expression, we require $R_1 < q^\ell$.

Finally, conditioned on the events $E_{\mathbf{P}} \cap E_{\mathbf{e}} \cap E_{\mathbf{r}}$, there exists a valid representation $(\mathbf{a}_1, \mathbf{a}_2)$ of $\tilde{\mathbf{s}}$. Since each of $\mathbf{a}_1$ and $\mathbf{a}_2$ admits R_2 representations into lower-layer vectors, the total number of valid tuples $(\mathbf{s}_1, \mathbf{s}_2, \mathbf{s}_3, \mathbf{s}_4)$ leading to $\tilde{\mathbf{s}}$ equals R_2^2. Therefore,

$$\mathbb{P}[E_{\tilde{\mathbf{s}}} \mid E_{\mathbf{P}} \cap E_{\mathbf{e}} \cap E_{\mathbf{r}}] = \frac{R_2^2}{q^\ell}.$$

As above, we require $R_2^2 < q^\ell$.

We now analyze the cost of a single iteration. This cost is dominated by the collision search between the second-layer functions g_1 and g_2. Obtaining one collision between these functions requires $\tilde{\mathcal{O}}(q^{\ell/2})$ evaluations. Each such evaluation, in turn, invokes a collision search between (f_1, f_2) or (f_3, f_4), each having cost $\tilde{\mathcal{O}}(q^{\ell/2})$. Consequently, the cost of a single iteration is $\tilde{\mathcal{O}}(q^{\ell/2} \cdot q^{\ell/2}) = \tilde{\mathcal{O}}(q^\ell)$.

Combining the success probability with the cost of a single iteration, and using Condition C1, the overall time complexity becomes

$$T_{\text{EGMS}} = \frac{|\tau^n(w/2)|}{|\mathcal{D}|} \cdot 2^{\Theta\left(\frac{n}{\log n}\right)} \cdot \frac{q^\ell}{R_1} \cdot \frac{q^\ell}{R_2^2} \cdot q^\ell = \left(\frac{|\mathcal{D}|^{1/2} \, |\tau^n(w/2)|}{R_1 R_2^2} \right)^{1+o(1)}. \quad (8)$$

In the following subsection, we describe the specific choices of the secret domain $\mathcal{D}$ and the corresponding sets $\mathcal{T}_i$ used in the Nested-2* instantiation of [14]. To the best of our knowledge, this variant currently provides the most memory-efficient nested collision search attack against ternary LWE keys.

3.2 The Nested-2* Instantiation

In the Nested-2* instantiation, the secret vector $\mathbf{s}$ is partitioned according to two optimized parameters $\gamma, \beta \in [0, 1]$. More precisely, the coordinates of $\mathbf{s}$ are divided into two parts:

- a *joint part* of length γn, in which the secret has Hamming weight $\beta\gamma w$, i.e., the number of nonzero entries is $\beta\gamma w$, consisting equally of $+1$ and -1 entries. In this part, the vectors $\mathbf{s}_i$ are allowed to take values in $\{0, \pm 1, \pm 2\}$, resulting in a quinary representation;
- a *disjoint part* of length $\bar{\gamma} n := (1 - \gamma)n$, where the vectors $\mathbf{s}_1, \ldots, \mathbf{s}_4$ are pairwise disjoint. The remaining secret weight $\alpha w := (1 - \beta\gamma)w$ is distributed equally among the four vectors.

Under this decomposition, the original search space of the secret vector change from $\tau^n(w/2)$ to restricted space

$$\mathcal{D} = \left(\tau^{\bar{\gamma} n/4}(\alpha w/8)\right)^4 \times \tau^{\gamma n}(\beta\gamma w/2)$$

with size $|\mathcal{D}| = \binom{\bar{\gamma}n/4}{\alpha w/8, \alpha w/8, \cdot}^4 \cdot \binom{\gamma n}{\beta\gamma w/2, \beta\gamma w/2, \cdot}$. To ensure that the secret vector satisfies this prescribed weight distribution, the algorithm is repeated over independent random permutations of the coordinates (see Line 4 of Algorithm 1). Accordingly, for a random permutation matrix $\mathbf{P}$, the LWE relation transforms into

$$\mathbf{A}'\tilde{\mathbf{s}} = \mathbf{b} + \mathbf{e}, \qquad \text{where } \mathbf{A}' = \mathbf{AP} \text{ and } \tilde{\mathbf{s}} = \mathbf{P}^{-1}\mathbf{s}.$$

Moreover, for the weight distribution in the disjoint part to be feasible, the available number of coordinates must be sufficient to support the prescribed weight. Hence, we require $\alpha w/4 \leqslant \bar{\gamma}n/4$.

Furthermore, the domains of the vectors $\mathbf{s}_i$ are then defined as

$$\begin{array}{llll}
\mathcal{T}_1 = \tau^{\bar{\gamma}n/4}(\alpha w/8) & \times \mathbf{0} & \times \mathbf{0} & \times \mathbf{0} \quad \times \tau_2^{\gamma n}(n_1^{(2)}, n_2^{(2)}) \\
\mathcal{T}_2 = \mathbf{0} & \times \tau^{\bar{\gamma}n/4}(\alpha w/8) & \times \mathbf{0} & \times \mathbf{0} \quad \times \tau_2^{\gamma n}(n_1^{(2)}, n_2^{(2)}) \\
\mathcal{T}_3 = \mathbf{0} & \times \mathbf{0} & \times \tau^{\bar{\gamma}n/4}(\alpha w/8) & \times \mathbf{0} \quad \times \tau_2^{\gamma n}(n_1^{(2)}, n_2^{(2)}) \\
\mathcal{T}_4 = \mathbf{0} & \times \mathbf{0} & \times \mathbf{0} & \times \tau^{\bar{\gamma}n/4}(\alpha w/8) \quad \times \tau_2^{\gamma n}(n_1^{(2)}, n_2^{(2)})
\end{array}$$

where the parameters $n_1^{(2)}$ and $n_2^{(2)}$ will be specified later. The set $\tau_2^n(n_1, n_2)$ is defined by

$$\tau_2^n(n_1, n_2) = \{\mathbf{x} \in \{0, \pm 1, \pm 2\}^n : |\mathbf{x}|_{-1} = |\mathbf{x}|_1 = n_1,\ |\mathbf{x}|_{-2} = |\mathbf{x}|_2 = n_2\}.$$

Consequently, each domain $\mathcal{T}_i$ has cardinality

$$|\mathcal{T}_i| = \binom{\bar{\gamma}n/4}{\alpha w/8, \alpha w/8, \cdot} \cdot \binom{\gamma n}{n_1^{(2)}, n_1^{(2)}, n_2^{(2)}, n_2^{(2)}, \cdot}.$$

Corresponding to these choices of $\mathcal{T}_i$, the intermediate domains associated with the sums $\mathbf{a}_1 (= \mathbf{s}_1 + \mathbf{s}_2)$ and $\mathbf{a}_2 (= \mathbf{s}_3 + \mathbf{s}_4)$ are given by

$$\begin{array}{llll}
\mathcal{D}_1 = \tau^{\bar{\gamma}n/4}(\alpha w/8) & \times \tau^{\bar{\gamma}n/4}(\alpha w/8) & \times \mathbf{0} & \times \mathbf{0} \quad \times \tau_2^{\gamma n}(n_1^{(1)}, n_2^{(1)}) \\
\mathcal{D}_2 = \mathbf{0} & \times \mathbf{0} & \times \tau^{\bar{\gamma}n/4}(\alpha w/8) & \times \tau^{\bar{\gamma}n/4}(\alpha w/8) \quad \times \tau_2^{\gamma n}(n_1^{(1)}, n_2^{(1)}),
\end{array}$$

respectively. We now explicitly define the parameters $n_i^{(j)}$. We first determine the middle-layer parameters $n_i^{(1)}$, followed later by the parameters $n_i^{(2)}$ corresponding to the lower-layer.

By construction, the joint part (γn-portion) of the secret vector contains

$$n_0^{(0)} = \gamma n - \beta\gamma w \qquad \text{and} \qquad n_1^{(0)} = \frac{\beta\gamma w}{2} \tag{9}$$

coordinates equal to 0 and 1 (resp. -1), respectively. To represent a coordinate taking values in $\{-1, 0, 1\}$, the following decompositions are possible:

	-2	-1	0	1	2
$-1:$	$\underbrace{-2+1}_{o_1^{(0)}}$	$\underbrace{-1+0}_{\frac{n_1^{(0)}}{2}-o_1^{(0)}}$	$\underbrace{0-1}_{\frac{n_1^{(0)}}{2}-o_1^{(0)}}$	$\underbrace{1-2}_{o_1^{(0)}}$	
$0:$	$\underbrace{-2+2}_{z_2^{(0)}}$	$\underbrace{-1+1}_{z_1^{(0)}}$	$\underbrace{0+0}_{n_0^{(0)}-2(z_1^{(0)}+z_2^{(0)})}$	$\underbrace{1-1}_{z_1^{(0)}}$	$\underbrace{2-2}_{z_2^{(0)}}$
$1:$		$\underbrace{-1+2}_{o_1^{(0)}}$	$\underbrace{0+1}_{\frac{n_1^{(0)}}{2}-o_1^{(0)}}$	$\underbrace{1+0}_{\frac{n_1^{(0)}}{2}-o_1^{(0)}}$	$\underbrace{2-1}_{o_1^{(0)}}$

Here, the parameters $o_1^{(0)}, z_1^{(0)}$ and $z_2^{(0)}$ are optimization variables. Summing the corresponding contributions from the two lower-layer vectors yields the distribution of coordinates in the middle-layer. Consequently, the numbers of coordinates in the joint part equal to 0, ± 1 and ± 2 in the vectors $\mathbf{a}_1$ and $\mathbf{a}_2$ are given by

$$n_0^{(1)} = n_0^{(0)} + n_1^{(0)} - 2(o_1^{(0)} + z_1^{(0)} + z_2^{(0)}), \quad n_1^{(1)} = n_1^{(0)}/2 + z_1^{(0)}, \quad n_2^{(1)} = o_1^{(0)} + z_2^{(0)}. \tag{10}$$

Moreover, the total number of representations $(\mathbf{a}_1, \mathbf{a}_2)$ of the secret vector $\mathbf{s}$ is

$$R_1 = \binom{n_1^{(0)}}{o_1^{(0)}, o_1^{(0)}, n_1^{(0)}/2 - o_1^{(0)}, .}^2 \binom{n_0^{(0)}}{z_1^{(0)}, z_1^{(0)}, z_2^{(0)}, z_2^{(0)}, .}. \tag{11}$$

We now describe the decomposition of the intermediate vectors $\mathbf{a}_1$ and $\mathbf{a}_2$ into

$$\mathbf{a}_1 = \mathbf{s}_1 + \mathbf{s}_2 \qquad \text{and} \qquad \mathbf{a}_2 = \mathbf{s}_3 + \mathbf{s}_4,$$

respectively. As in the previous layer, each coordinate taking values in $\{-2, -1, 0, 1, 2\}$ admits several possible decompositions. These possibilities are summarized below:

	-2	-1	0	1	2
$-2:$	$\underbrace{-2+0}_{t^{(1)}}$	$\underbrace{-1-1}_{n_2^{(1)}-2t^{(1)}}$	$\underbrace{0-2}_{t^{(1)}}$		
$-1:$	$\underbrace{-2+1}_{o_1^{(1)}}$	$\underbrace{-1+0}_{\frac{n_1^{(1)}}{2}-o_1^{(1)}}$	$\underbrace{0-1}_{\frac{n_1^{(1)}}{2}-o_1^{(1)}}$	$\underbrace{1-2}_{o_1^{(1)}}$	
$0:$	$\underbrace{-2+2}_{z_2^{(1)}}$	$\underbrace{-1+1}_{z_1^{(1)}}$	$\underbrace{0+0}_{n_0^{(1)}-2(z_1^{(1)}+z_2^{(1)})}$	$\underbrace{1-1}_{z_1^{(1)}}$	$\underbrace{2-2}_{z_2^{(1)}}$
$1:$		$\underbrace{-1+2}_{o_1^{(1)}}$	$\underbrace{0+1}_{\frac{n_1^{(1)}}{2}-o_1^{(1)}}$	$\underbrace{1+0}_{\frac{n_1^{(1)}}{2}-o_1^{(1)}}$	$\underbrace{2-1}_{o_1^{(1)}}$
$2:$			$\underbrace{0+2}_{t^{(1)}}$	$\underbrace{1+1}_{n_2^{(1)}-2t^{(1)}}$	$\underbrace{2+0}_{t^{(1)}}$

The parameters $o_1^{(1)}, z_1^{(1)}, z_2^{(1)}$ and $t^{(1)}$ are treated as optimization variables. By summing the corresponding lower-layer contributions, the numbers of coordinates in the joint part equal to 0, ± 1 and ± 2 in the vectors $\mathbf{s}_i$ are obtained as

$$n_0^{(2)} = n_0^{(1)} + n_1^{(1)} - 2\big(o_1^{(1)} + z_1^{(1)} + z_2^{(1)} - t^{(1)}\big), \qquad n_1^{(2)} = \tfrac{n_1^{(1)}}{2} + n_2^{(1)} + z_1^{(1)} - 2t^{(1)},$$
$$n_2^{(2)} = o_1^{(1)} + z_2^{(1)} + t^{(1)}. \tag{12}$$

Furthermore, the total number of representations $(\mathbf{s}_1, \mathbf{s}_2)$(resp., $(\mathbf{s}_3, \mathbf{s}_4)$) of the vector $\mathbf{a}_1$ (resp., $\mathbf{a}_2$) is given by

$$R_2 = \binom{n_2^{(1)}}{t^{(1)}, t^{(1)}, .}^2 \binom{n_1^{(1)}}{o_1^{(1)}, o_1^{(1)}, n_1^{(1)}/2 - o_1^{(1)}, .}^2 \binom{n_0^{(1)}}{z_1^{(1)}, z_1^{(1)}, z_2^{(1)}, z_2^{(1)}, .}. \tag{13}$$

By substituting the corresponding values of $|\mathcal{D}|$, R_1 and R_2 into (8), we obtain the asymptotic runtime exponent for the NESTED-2* instantiation.

4 An Improved Nested Collision Search Framework

In this section, we introduce our improved nested collision search framework. The main idea is to relax the balancing condition between the projection space and the secret domain used in the EGMS framework. This relaxation allows us to work with larger intermediate domains and, consequently, obtain more representations. While a larger projection space increases the cost of the collision search, it also provides more representations, and the resulting gain can outweigh this additional cost. We first describe this modified nested collision search procedure and subsequently we instantiate the framework using different representation structures.

Recall from Section 3.1 that Esser *et al.* choose the parameter ℓ so that the projected system $\pi_{2\ell}(\mathbf{As}) = \mathbf{b}'$ admits, heuristically, a unique solution. Accordingly, they impose the condition

$$q^\ell = |\mathcal{D}|^{1/2} \qquad \text{(Condition C1)}.$$

However, in the planted regime of the ternary LWE problem, the projected system is guaranteed to admit at least one solution, namely the planted secret itself. Consequently, even if we choose ℓ such that $q^\ell > |\mathcal{D}|^{1/2}$, the system is still expected, heuristically, to admit a unique solution with high probability. Motivated by this observation, we introduce an additional optimization parameter $\lambda \in [0, 1/2]$, and replace Condition C1 with

$$q^\ell = |\mathcal{D}|^{1/2+\lambda}. \tag{C3}$$

Under the random-matrix heuristic, for any fixed $\mathbf{s}' \in \mathcal{D}\backslash\{\mathbf{s}\}$ the projected value $\pi_{2\ell}(\mathbf{As}')$ is uniformly distributed in $\mathbb{Z}_q^{2\ell}$. Hence the expected number of false projected solutions is

$$\frac{|\mathcal{D}| - 1}{q^{2\ell}} = |\mathcal{D}|^{-2\lambda} + o(1).$$

Thus, for constant $\lambda > 0$ and $|\mathcal{D}| = 2^{\Theta(n)}$, the probability of a false projected solution is heuristically negligible. The case $\lambda = 0$ recovers the EGMS balancing condition, where the expected number of false projected solutions is constant. Importantly, the planted nature of the solution does not increase the probability that the permuted secret belongs to $\mathcal{D}$, nor does it increase its number of representations. Its role is instead to guarantee the existence of a solution to the projected equation, even when the projection space is larger than the search domain.

At first glance, increasing the value of q^ℓ may appear disastrous, since the collision search between the second-layer functions g_1 and g_2 requires time $\tilde{\mathcal{O}}(q^\ell)$. However, recall that Pollard's Rho collision search requires the domain and codomain sizes to be balanced, implying

$$q^\ell = |\mathcal{T}_i|. \tag{C4}$$

Thus, increasing q^ℓ simultaneously enlarges the domains $\mathcal{T}_i$. This additional entropy allows the numbers of representations R_1 and R_2 to increase significantly. Since the overall complexity (see (8)) depends inversely on the product $R_1 R_2^2$, the gain obtained from the increased number of representations can outweigh the additional collision search cost, ultimately leading to an improved overall complexity.

Accordingly, our improved algorithm is a wrapper around Algorithm 1: it runs the same nested collision search, but with the value of ℓ chosen according to Condition C3 and with layer function domains satisfying Condition C4. Therefore, the correctness of our improved algorithm follows directly from the correctness of Algorithm 1. We provide the resulting pseudocode in Algoritm 2.

Algorithm 2: Improved Nested Collision Search

Input: $(\mathbf{A}, \mathbf{b}) \in \mathbb{Z}_q^{n \times n} \times \mathbb{Z}_q^n$ and weight of the secret $w \leqslant n$

Choose optimal λ such that $0 \leqslant \lambda \leqslant 1/2$;

Choose optimized domain $\mathcal{D}$ and set $\ell \leftarrow \left(\frac{1}{2} + \lambda\right) \log_q |\mathcal{D}|$;

Choose optimized lower-level function domains $\mathcal{T}_1, \ldots, \mathcal{T}_4$ such that

$$|\mathcal{T}_i| = q^\ell = |\mathcal{D}|^{1/2+\lambda}, \qquad (1 \leqslant i \leqslant 4).$$

Run Algorithm 1 with this value of ℓ and these optimized domains;

return *the secret vector* $\mathbf{s}^\star$

Complexity Analysis. Since the overall framework of Algorithm 2 remains identical to that of Algorithm 1, the complexity analysis follows along the same lines. The only difference is that the projection dimension is now chosen as

$$\ell = \left(\frac{1}{2} + \lambda\right) \log_q |\mathcal{D}|, \qquad 0 \leqslant \lambda \leqslant \frac{1}{2}.$$

As before, since $q = \mathrm{poly}(n)$ and $|\mathcal{D}| = 2^{cn}$ for some constant $c > 0$, we obtain

$$\mathbb{P}[E_{\mathbf{e}} \mid E_{\mathbf{P}}] = 3^{-2\ell} = 2^{-\Theta\left(\frac{n}{\log n}\right)}.$$

Hence, the contribution arising from the error-guessing step remains subexponential and is absorbed into the lower-order term.

The main difference compared to Algorithm 1 is that every occurrence of q^ℓ, and consequently the balanced component-domain size $|\mathcal{T}_i|$, is now evaluated under the relation

$$q^\ell = |\mathcal{D}|^{1/2+\lambda}.$$

As a consequence, both the collision search cost and the optimized representation counts R_1 and R_2 change accordingly.

Therefore, combining the success probability with the cost of a single iteration exactly as in the analysis of Algorithm 1, and using Condition C3, the overall time complexity becomes

$$T_{\text{our}} = \frac{|\tau^n(w/2)|}{|\mathcal{D}|} \cdot 2^{\Theta\left(\frac{n}{\log n}\right)} \cdot \frac{q^\ell}{R_1} \cdot \frac{q^\ell}{R_2^2} \cdot q^\ell = \left(\frac{|\mathcal{D}|^{(1/2+3\lambda)}\,|\tau^n(w/2)|}{R_1 R_2^2}\right)^{1+o(1)}. \quad (14)$$

Following the complexity analysis of Algorithm 1, the above complexity expression is subject to the constraints

$$R_1 < |\mathcal{T}_i|, \qquad R_2^2 < |\mathcal{T}_i|, \qquad |\mathcal{T}_i| = |\mathcal{D}|^{1/2+\lambda}, \quad (C5)$$

where the inequalities and the equality follow from Conditions C3 and C4.

We now present several concrete instantiations of the improved nested collision search framework, together with their resulting time complexities under Algorithm 2.

4.1 Improved-Nested-2* Instantiation

We now analyze the Nested-2* instantiation by combining with Algorithm 2. The resulting variant will be referred to as Improved-Nested-2*.

To compute the complexity $T_{\text{Improved-Nested-2}^\star}$, we substitute the expressions for $|\mathcal{D}|$, R_1 and R_2 derived in Section 3.2 into (14). To obtain the asymptotic exponent

$$T_{\text{Improved-Nested-2}^\star} = 2^{cn},$$

we approximate all multinomial coefficients using the entropy-based formulas from Section 2.2. Furthermore, all optimization parameters are normalized as $\mu = \tilde{\mu} n$, $\tilde{\mu} \in [0, 1]$ for each $\mu \in \{o_1^{(0)}, z_1^{(0)}, z_2^{(0)}, o_1^{(1)}, z_1^{(1)}, z_2^{(1)}, t^{(1)}, \gamma, \beta, \lambda\}$. The exponent c is then obtained by minimizing the resulting expression subject to the constraints in C5. For this optimization, we use the numerical optimizer provided by the `SciPy` Python library, inspired by the code used for numerical optimization in [14].

Figure 3 compares the complexities of Nested-2* and Improved-Nested-2* for various values of the relative weight $\omega := \frac{w}{n}$. We observe that Improved-Nested-2* consistently outperforms Nested-2* across the entire range of relative weights. The two complexities become close when the relative weight is either very small or very large. However, in the cryptographically relevant regime, the improvement is significant. In particular, for the balanced-weight setting $\omega = 0.667$, the original Nested-2* attack has complexity $T_{\text{Nested-2}^\star} = 2^{0.9297n}$, whereas the improved framework achieves $T_{\text{Improved-Nested-2}^\star} = 2^{0.8847n}$, yielding a substantial asymptotic improvement. For the balanced-weight setting, the optimized Improved-Nested-2* parameters give domain sizes $|\mathcal{D}| = 2^{1.5754744n}$ and $|\mathcal{T}_i| = 2^{0.8752471n}$, and representation counts $R_1 = 2^{0.8752470n}$ and $R_2 = 2^{0.4376235n}$. In comparison, the optimized parameters for Nested-2* give $|\mathcal{D}| = 2^{1.5658838n}$, $|\mathcal{T}_i| = 2^{0.7829419n}$, with representation counts $R_1 = 2^{0.6658295n}$, $R_2 = 2^{0.3861696n}$. The main reason for the improvement is that our framework uses

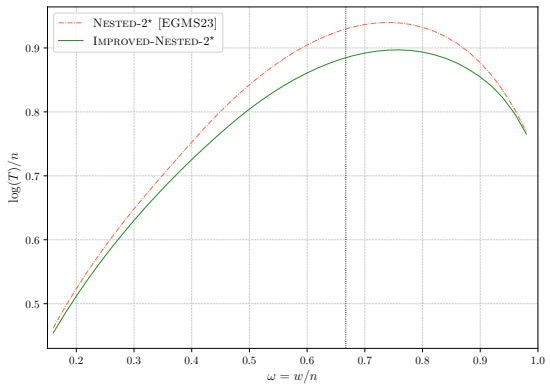


Fig. 3. Comparison between Nested-2* and Improved-Nested-2* for varying relative weight $\omega = w/n$.

larger search domains, which gives more representations at the intermediate levels. These additional representations make it more likely to find useful collisions and therefore reduce the overall runtime.

Remark 2. The numerical optimization routines used in our experiments do not provide guarantees of convergence to a global optimum. Consequently, the returned parameters may correspond to a local optimum or may deviate slightly from the true optimal values. A similar limitation also arises in the optimization procedures used in [14]. To increase confidence in the reported results, we performed the optimization over a large number of randomly chosen initial points and experimented with multiple equivalent formulations of the optimization problem. The reported values correspond to the best solutions obtained across all runs.

4.2 Shifted-3 Instantiation

In this instantiation, we retain the same secret domain as in the Nested-2* framework, namely

$$\left(\tau^{\bar{\gamma}n/4}(\alpha w/8)\right)^4 \times \tau^{\gamma n}(\beta\gamma w/2),$$

together with the transformed LWE relation induced by the random permutation matrix $\mathbf{P}$,

$$\mathbf{A}'\tilde{\mathbf{s}} = \mathbf{b} + \mathbf{e}, \qquad \text{where } \mathbf{A}' = \mathbf{AP} \text{ and } \tilde{\mathbf{s}} = \mathbf{P}^{-1}\mathbf{s}.$$

However, instead of working directly with the secret vector $\tilde{\mathbf{s}}$, we introduce a shifted secret

$$\mathbf{s}' = \tilde{\mathbf{s}} + \mathbf{v}_\gamma, \qquad \text{where } \mathbf{v}_\gamma = (0, \ldots, 0, \underbrace{1, \ldots, 1}_{\gamma n})^T.$$

That is, the shift is applied only to the joint part of length γn. The shifted secret satisfies the modified LWE relation

$$\mathbf{A}'\mathbf{s}' = \mathbf{b} + \mathbf{A}'\mathbf{v}_\gamma + \mathbf{e}. \tag{15}$$

Our objective is therefore to recover the shifted secret $\mathbf{s}'$ via nested collision search, from which the original secret vector $\mathbf{s}$ can be recovered immediately. Accordingly, in Algorithm 2, after guessing the first 2ℓ coordinates of the error vector $\mathbf{e}$, we set $\mathbf{b}' = \pi_{2\ell}(\mathbf{b} + \mathbf{A}'\mathbf{v}_\gamma + \mathbf{e})$, and after recovering the shifted candidate $\mathbf{s}'$, we compute $\mathbf{s}^\star = \mathbf{P}(\mathbf{s}' - \mathbf{v}_\gamma)$, which is accepted only if $(\mathbf{A}\mathbf{s}^\star - \mathbf{b} \mod q) \in \{-1, 0, 1\}^n$.

A crucial consequence of this transformation is that the weight distribution in the joint part becomes asymmetric. More precisely, after shifting, the numbers of coordinates equal to 0, 1 and 2 in the joint part become

$$n_0^{(0)} = \frac{\beta\gamma w}{2}, \qquad n_1^{(0)} = \gamma n - \beta\gamma w, \qquad n_2^{(0)} = \frac{\beta\gamma w}{2}, \tag{16}$$

respectively. This asymmetric distribution introduces additional flexibility in the representation structure. In particular, for certain parameter ranges, it yields a more favorable balance between the representation counts and the collision search cost, leading to improved asymptotic complexity exponents. We analyze this advantage in detail below.

To describe these asymmetric distributions, we introduce the notation

$$\bar{\tau}^n(a_{-3}, a_{-2}, a_{-1}, a_0, a_1, a_2, a_3) := \{\mathbf{x} \in \{-3, -2, -1, 0, 1, 2, 3\}^n : |\mathbf{x}|_i = a_i \text{ for all } i \in \{-3, \dots, 3\}\}.$$

Under the above shift, the secret domain transforms into

$$\mathcal{D} = \left(\tau^{\bar{\gamma}n/4}(\alpha w/8)\right)^4 \times \bar{\tau}^{\gamma n}\left(0, 0, 0, n_0^{(0)}, n_1^{(0)}, n_2^{(0)}, 0\right).$$

Observe that this transformation preserves the size of the search space, i.e.,

$$|\mathcal{D}| = \binom{\bar{\gamma}n/4}{\alpha w/8, \alpha w/8, \cdot}^4 \cdot \binom{\gamma n}{\beta\gamma w/2, \beta\gamma w/2, \cdot}.$$

We now define the lower-layer domains $\mathcal{T}_i$ as (for an illustration see Figure 4)

$$\begin{aligned}
\mathcal{T}_1 &= \tau^{\bar{\gamma}n/4}(\alpha w/8) \times \mathbf{0} \times \mathbf{0} \times \mathbf{0} \times \bar{\tau}^{\gamma n}(n_{-3}^{(2)}, n_{-2}^{(2)}, n_{-1}^{(2)}, n_0^{(2)}, n_1^{(2)}, n_2^{(2)}, n_3^{(2)}) \\
\mathcal{T}_2 &= \mathbf{0} \times \tau^{\bar{\gamma}n/4}(\alpha w/8) \times \mathbf{0} \times \mathbf{0} \times \bar{\tau}^{\gamma n}(n_{-3}^{(2)}, n_{-2}^{(2)}, n_{-1}^{(2)}, n_0^{(2)}, n_1^{(2)}, n_2^{(2)}, n_3^{(2)}) \\
\mathcal{T}_3 &= \mathbf{0} \times \mathbf{0} \times \tau^{\bar{\gamma}n/4}(\alpha w/8) \times \mathbf{0} \times \bar{\tau}^{\gamma n}(n_{-3}^{(2)}, n_{-2}^{(2)}, n_{-1}^{(2)}, n_0^{(2)}, n_1^{(2)}, n_2^{(2)}, n_3^{(2)}) \\
\mathcal{T}_4 &= \mathbf{0} \times \mathbf{0} \times \mathbf{0} \times \tau^{\bar{\gamma}n/4}(\alpha w/8) \times \bar{\tau}^{\gamma n}(n_{-3}^{(2)}, n_{-2}^{(2)}, n_{-1}^{(2)}, n_0^{(2)}, n_1^{(2)}, n_2^{(2)}, n_3^{(2)})
\end{aligned}$$

Analogously to the previous instantiation, we define the corresponding middle-layer domains by

$$\begin{aligned}
\mathcal{D}_1 &= \tau^{\bar{\gamma}n/4}(\alpha w/8) \times \tau^{\bar{\gamma}n/4}(\alpha w/8) \times \mathbf{0} \times \mathbf{0} \times \bar{\tau}^{\gamma n}(n_{-3}^{(1)}, n_{-2}^{(1)}, n_{-1}^{(1)}, n_0^{(1)}, n_1^{(1)}, n_2^{(1)}, n_3^{(1)}) \\
\mathcal{D}_2 &= \mathbf{0} \times \mathbf{0} \times \tau^{\bar{\gamma}n/4}(\alpha w/8) \times \tau^{\bar{\gamma}n/4}(\alpha w/8) \times \bar{\tau}^{\gamma n}(n_{-3}^{(1)}, n_{-2}^{(1)}, n_{-1}^{(1)}, n_0^{(1)}, n_1^{(1)}, n_2^{(1)}, n_3^{(1)})
\end{aligned}$$

We now determine the parameters $n_i^{(j)}$. We begin with the middle-layer parameters $n_i^{(1)}$.

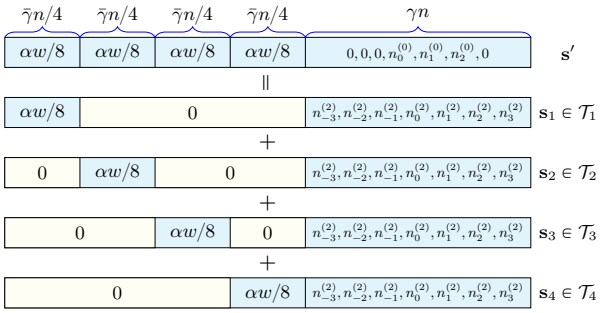


Fig. 4. Weight distribution of lower-level function domains for SHIFTED-3 instantiation.

In the joint part, the shifted secret vector contains only coordinates equal to 0, 1 and 2. These coordinates are represented using values from $\{0, \pm1, \pm2, \pm3\}$, and the corresponding decomposition possibilities are illustrated below:

	-3	-2	-1	0	1	2	3
0 :	$\underbrace{-3+3}_{z_3^{(0)}}$	$\underbrace{-2+2}_{z_2^{(0)}}$	$\underbrace{-1+1}_{z_1^{(0)}}$	$\underbrace{0+0}_{n_0^{(0)}-2(z_1^{(0)}+z_2^{(0)}+z_3^{(0)})}$	$\underbrace{1-1}_{z_1^{(0)}}$	$\underbrace{2-2}_{z_2^{(0)}}$	$\underbrace{3-3}_{z_3^{(0)}}$
1 :		$\underbrace{-2+3}_{o_2^{(0)}}$	$\underbrace{-1+2}_{o_1^{(0)}}$	$\underbrace{0+1}_{\frac{n_1^{(0)}}{2}-o_1^{(0)}-o_2^{(0)}}$	$\underbrace{1+0}_{\frac{n_1^{(0)}}{2}-o_1^{(0)}-o_2^{(0)}}$	$\underbrace{2-1}_{o_1^{(0)}}$	$\underbrace{3-2}_{o_2^{(0)}}$
2 :			$\underbrace{-1+3}_{t_1^{(0)}}$	$\underbrace{0+2}_{t_0^{(0)}}$	$\underbrace{1+1}_{n_2^{(0)}-2(t_1^{(0)}+t_0^{(0)})}$	$\underbrace{2+0}_{t_0^{(0)}}$	$\underbrace{3-1}_{t_1^{(0)}}$

The parameters $z_3^{(0)}$, $z_2^{(0)}$, $z_1^{(0)}$, $o_2^{(0)}$, $o_1^{(0)}$, $t_1^{(0)}$, $t_0^{(0)}$ are treated as optimization variables. Summing the corresponding contributions yields the numbers of coordinates equal to $-3, -2, -1, 0, 1, 2, 3$ in the middle-layer vectors $\mathbf{a}_1$ and $\mathbf{a}_2$:

$$\begin{aligned}
&n_{-3}^{(1)} = z_3^{(0)}, \quad n_{-2}^{(1)} = z_2^{(0)} + o_2^{(0)}, \quad n_{-1}^{(1)} = z_1^{(0)} + o_1^{(0)} + t_1^{(0)}, \\
&n_0^{(1)} = n_0^{(0)} - 2\big(z_1^{(0)} + z_2^{(0)} + z_3^{(0)}\big) + \frac{n_1^{(0)}}{2} - o_1^{(0)} - o_2^{(0)} + t_0^{(0)}, \\
&n_1^{(1)} = z_1^{(0)} + \frac{n_1^{(0)}}{2} - o_1^{(0)} - o_2^{(0)} + n_2^{(0)} - 2\big(t_1^{(0)} + t_0^{(0)}\big), \\
&n_2^{(1)} = z_2^{(0)} + o_1^{(0)} + t_0^{(0)}, \quad \text{and} \quad n_3^{(1)} = z_3^{(0)} + o_2^{(0)} + t_1^{(0)}.
\end{aligned} \tag{17}$$

Furthermore, the total number of representations $(\mathbf{a}_1, \mathbf{a}_2)$ of the shifted secret vector $\mathbf{s}'$ is

$$R_1 = \binom{n_0^{(0)}}{z_1^{(0)}, z_2^{(0)}, z_3^{(0)}, z_1^{(0)}, z_2^{(0)}, z_3^{(0)}, .}\binom{n_1^{(0)}}{o_1^{(0)}, o_2^{(0)}, o_1^{(0)}, o_2^{(0)}, n_1^{(0)}/2 - o_1^{(0)} - o_2^{(0)}, .}\binom{n_2^{(0)}}{t_0^{(0)}, t_1^{(0)}, t_0^{(0)}, t_1^{(0)}, .}. \tag{18}$$

We now analyze the lower-layer decomposition. Each coordinate of the vectors $\mathbf{a}_1$ and $\mathbf{a}_2$ is further represented using elements from $\{0, \pm1, \pm2, \pm3\}$, and the corresponding decomposition patterns are illustrated below:

	−3	−2	−1	0	1	2	3
−3 :	$\underbrace{-3+0}_{thm^{(1)}}$	$\underbrace{-2-1}_{\frac{n_{-3}^{(1)}}{2}-thm^{(1)}}$	$\underbrace{-1-2}_{\frac{n_{-3}^{(1)}}{2}-thm^{(1)}}$	$\underbrace{0-3}_{thm^{(1)}}$			
−2 :	$\underbrace{-3+1}_{tm_1^{(1)}}$	$\underbrace{-2+0}_{tm_0^{(1)}}$	$\underbrace{-1-1}_{n_{-2}^{(1)}-2(tm_0^{(1)}+tm_1^{(1)})}$	$\underbrace{0-2}_{tm_0^{(1)}}$	$\underbrace{1-3}_{tm_1^{(1)}}$		
−1 :	$\underbrace{-3+2}_{om_2^{(1)}}$	$\underbrace{-2+1}_{om_1^{(1)}}$	$\underbrace{-1+0}_{\frac{n_{-1}^{(1)}}{2}-om_1^{(1)}-om_2^{(1)}}$	$\underbrace{0-1}_{\frac{n_{-1}^{(1)}}{2}-om_1^{(1)}-om_2^{(1)}}$	$\underbrace{1-2}_{om_1^{(1)}}$	$\underbrace{2-3}_{om_2^{(1)}}$	
0 :	$\underbrace{-3+3}_{z_3^{(1)}}$	$\underbrace{-2+2}_{z_2^{(1)}}$	$\underbrace{-1+1}_{z_1^{(1)}}$	$\underbrace{0+0}_{n_0^{(1)}-2(z_1^{(1)}+z_2^{(1)}+z_3^{(1)})}$	$\underbrace{1-1}_{z_1^{(1)}}$	$\underbrace{2-2}_{z_2^{(1)}}$	$\underbrace{3-3}_{z_3^{(1)}}$
1 :		$\underbrace{-2+3}_{o_2^{(1)}}$	$\underbrace{-1+2}_{o_1^{(1)}}$	$\underbrace{0+1}_{\frac{n_1^{(1)}}{2}-o_1^{(1)}-o_2^{(1)}}$	$\underbrace{1+0}_{\frac{n_1^{(1)}}{2}-o_1^{(1)}-o_2^{(1)}}$	$\underbrace{2-1}_{o_1^{(1)}}$	$\underbrace{3-2}_{o_2^{(1)}}$
2 :			$\underbrace{-1+3}_{t_1^{(1)}}$	$\underbrace{0+2}_{t_0^{(1)}}$	$\underbrace{1+1}_{n_2^{(1)}-2(t_0^{(1)}+t_1^{(1)})}$	$\underbrace{2+0}_{t_0^{(1)}}$	$\underbrace{3-1}_{t_1^{(1)}}$
3 :				$\underbrace{0+3}_{th^{(1)}}$	$\underbrace{1+2}_{\frac{n_3^{(1)}}{2}-th^{(1)}}$	$\underbrace{2+1}_{\frac{n_3^{(1)}}{2}-th^{(1)}}$	$\underbrace{3+0}_{th^{(1)}}$

The parameters $thm^{(1)}$, $tm_1^{(1)}$, $tm_0^{(1)}$, $om_2^{(1)}$, $om_1^{(1)}$, $z_3^{(0)}$, $z_2^{(0)}$, $z_1^{(0)}$, $o_2^{(0)}$, $o_1^{(0)}$, $t_1^{(0)}$, $t_0^{(0)}$, $th^{(1)}$ are treated as optimization variables. By summing the corresponding lower-layer contributions, the numbers of coordinates equal to $-3, -2, -1, 0, 1, 2, 3$ in the vectors $\mathbf{s}_i$ are obtained as

$$
\begin{aligned}
&n_{-3}^{(2)} = thm^{(1)} + tm_1^{(1)} + om_2^{(1)} + z_3^{(1)}, \quad n_{-2}^{(2)} = \frac{n_{-3}^{(1)}}{2} - thm^{(1)} + tm_0^{(1)} + om_1^{(1)} + z_2^{(1)} + o_2^{(1)},\\
&n_{-1}^{(2)} = \frac{n_{-3}^{(1)}}{2} - thm^{(1)} + n_{-2}^{(1)} - 2(tm_0^{(1)} + tm_1^{(1)}) + \frac{n_{-1}^{(1)}}{2} - om_1^{(1)} - om_2^{(1)} + z_1^{(1)} + o_1^{(1)} + t_1^{(1)},\\
&n_0^{(2)} = thm^{(1)} + tm_0^{(1)} + \frac{n_{-1}^{(1)}}{2} - om_1^{(1)} - om_2^{(1)} + n_0^{(1)} - 2(z_1^{(1)} + z_2^{(1)} + z_3^{(1)}) + \frac{n_1^{(1)}}{2} - o_1^{(1)} - o_2^{(1)} + t_0^{(1)} + th^{(1)},\\
&n_1^{(2)} = tm_1^{(1)} + om_1^{(1)} + z_1^{(1)} + \frac{n_1^{(1)}}{2} - o_1^{(1)} - o_2^{(1)} + n_2^{(1)} - 2(t_0^{(1)} + t_1^{(1)}) + \frac{n_3^{(1)}}{2} - th^{(1)},\\
&n_2^{(2)} = om_2^{(1)} + z_2^{(1)} + o_1^{(1)} + t_0^{(1)} + \frac{n_3^{(1)}}{2} - th^{(1)} \quad \text{and} \quad n_3^{(2)} = z_3^{(1)} + o_2^{(1)} + t_1^{(1)} + th^{(1)}.
\end{aligned}
\tag{19}
$$

Consequently, the total number of lower-layer representations $(\mathbf{s}_1, \mathbf{s}_2)$(resp., $(\mathbf{s}_3, \mathbf{s}_4)$) of the middle-layer vector $\mathbf{a}_1$ (resp. $\mathbf{a}_2$) is given by

$$
\begin{aligned}
R_2 = &\binom{n_{-3}^{(1)}}{thm^{(1)}, thm^{(1)}, \frac{n_{-3}^{(1)}}{2} - thm^{(1)}, .}\binom{n_{-2}^{(1)}}{tm_1^{(1)}, tm_0^{(1)}, tm_1^{(1)}, tm_0^{(1)}, .}\binom{n_{-1}^{(1)}}{om_1^{(1)}, om_2^{(1)}, om_1^{(1)}, om_2^{(1)}, \frac{n_{-1}^{(1)}}{2} - om_1^{(1)} - om_2^{(1)}, .}\\
&\binom{n_0^{(1)}}{z_3^{(1)}, z_2^{(1)}, z_1^{(1)}, z_3^{(1)}, z_2^{(1)}, z_1^{(1)}, .}\binom{n_1^{(1)}}{o_1^{(1)}, o_2^{(1)}, o_1^{(1)}, o_2^{(1)}, \frac{n_1^{(1)}}{2} - o_1^{(1)} - o_2^{(1)}, .}\binom{n_2^{(1)}}{t_1^{(1)}, t_0^{(1)}, t_1^{(1)}, t_0^{(1)}, .}\binom{n_3^{(1)}}{th^{(1)}, th^{(1)}, \frac{n_3^{(1)}}{2} - th^{(1)}, .}.
\end{aligned}
\tag{20}
$$

Substituting the expressions for $|\mathcal{D}|$, R_1 and R_2 into (14) yields the time complexity of the resulting attack, which we denote by $T_{\text{SHIFTED-3}}$. As in the previous instantiations, to derive the asymptotic exponent in the expression $T_{\text{SHIFTED-3}} = 2^{cn}$, we approximate all multinomial coefficients using the entropy-based formulas from Section 2.2. Furthermore, all optimization parameters are normalized as $\mu = \tilde{\mu} n$, $\tilde{\mu} \in [0, 1]$.

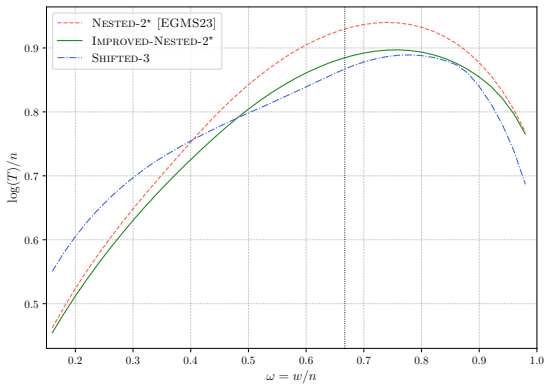


Fig. 5. Comparison of the asymptotic runtime exponents of Nested-2*, Improved-Nested-2*, and Shifted-3 for various relative weights $\omega = w/n$.

Figure 5 illustrates the optimized runtime exponents of Shifted-3 and compares them with those of Improved-Nested-2* and the original Nested-2* framework of [14]. We observe that Shifted-3 outperforms both of the instantiations whenever the relative weight satisfies $\omega \geqslant 0.48$.

Most notably, in the balanced-weight setting $\omega = 0.667$, the Shifted-3 instantiation reduces the asymptotic time exponent from 0.8847 achieved by Improved-Nested-2* to 0.8670, yielding a further nontrivial improvement over the previous instantiations. For the corresponding optimized parameter choices, the domain sizes are $|\mathcal{D}| = 2^{1.5822062n}$ and $|\mathcal{T}_i| = 2^{0.8642535n}$, while the representation counts are $R_1 = 2^{0.8642310n}$ and $R_2 = 2^{0.4321238n}$.

4.3 Hybrid Instantiation

We now present a unified instantiation that combines the advantages of Improved-Nested-2* and Shifted-3. The main idea is to apply the two strategies simultaneously on different portions of the joint part of the secret vector.

More precisely, we further partition the joint part of length γn into two subparts of lengths $\gamma_1 n$ and $\gamma_2 n$ where $\gamma_1, \gamma_2 \in [0, 1]$ are optimization parameters satisfying $0 \leqslant \gamma_1 + \gamma_2 \leqslant 1$. For additional optimization parameters $\beta_1, \beta_2 \in [0, 1]$, we impose the following weight distribution:

- the first joint subpart of length $\gamma_1 n$ contains $\frac{\beta_1 \gamma_1 w}{2}$ coordinates equal to $+1$ and the same number equal to -1;
- the second joint subpart of length $\gamma_2 n$ contains $\frac{\beta_2 \gamma_2 w}{2}$ coordinates equal to $+1$ and the same number equal to -1.

Under this distribution, the secret domain becomes

$$\left(\tau^{\bar{\gamma} n/4}(\alpha w/8)\right)^4 \times \tau^{\gamma_2 n}(\beta_2 \gamma_2 w/2) \times \tau^{\gamma_1 n}(\beta_1 \gamma_1 w/2),$$

where $\alpha = 1 - \beta_1\gamma_1 - \beta_2\gamma_2$, $\bar{\gamma} = 1 - \gamma_1 - \gamma_2$. For this weight distribution to be feasible, we additionally require $0 \leqslant \alpha \leqslant 1$ and $\frac{\alpha w}{4} \leqslant \frac{\bar{\gamma} n}{4}$.

We now apply the Improved-Nested-2* strategy on the $\gamma_2 n$ portion and the Shifted-3 strategy on the $\gamma_1 n$ portion. Accordingly, we introduce the shifted secret

$$\mathbf{s}' = \tilde{\mathbf{s}} + \mathbf{v}_{\gamma_1}, \qquad \text{where } \mathbf{v}_{\gamma_1} = (0, \ldots, 0, \underbrace{1, \ldots, 1}_{\gamma_1 n})^T.$$

The corresponding LWE relation becomes

$$\mathbf{A}'\mathbf{s}' = \mathbf{b} + \mathbf{A}'\mathbf{v}_{\gamma_1} + \mathbf{e},$$

and the search space of the shifted secret vector is

$$\mathcal{D} = \left(\tau^{\bar{\gamma}n/4}(\alpha w/8)\right)^4 \times \tau^{\gamma_2 n}(\beta_2\gamma_2 w/2) \times \bar{\tau}^{\gamma_1 n}\left(0, 0, 0, \tfrac{\beta_1\gamma_1 w}{2}, \gamma_1 n - \beta_1\gamma_1 w, \tfrac{\beta_1\gamma_1 w}{2}, 0\right).$$

The derivations of the domains $\mathcal{T}_i$ and the representation counts R_1 and R_2 follow analogously from the analyses in Section 3.2 and Section 4.2. To avoid excessive technical repetition, we present only the results here; the complete derivations are deferred to Appendix A.

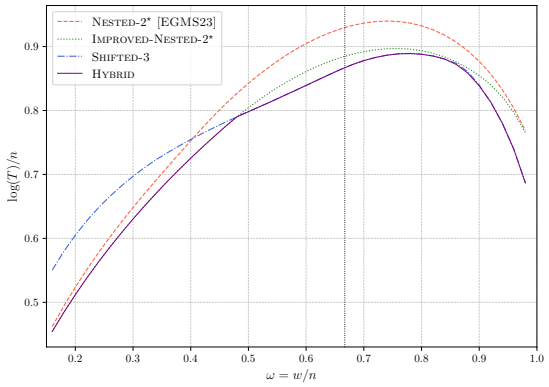


Fig. 6. Comparison of the asymptotic runtime exponents of Nested-2*, Improved-Nested-2*, Shifted-3, and the Hybrid instantiation as a function of the relative secret weight $\omega = w/n$. The Hybrid instantiation achieves the best known asymptotic complexity over the full parameter range.

Figure 6 compares the asymptotic runtime exponents of the Hybrid instantiation with those of the previous approaches. We observe that, for relative weights $\omega < 0.48$, the optimized parameters force the Hybrid instantiation to coincide with Improved-Nested-2*. Similarly, for $\omega \geqslant 0.48$ the optimized configuration coincides with Shifted-3. Although the Hybrid instantiation does not improve the complexity exponent, its primary significance lies in providing a unified

framework that attains the best currently known asymptotic complexity across the entire range of relative weights.

4.4 Known-Error-Weight ISD Preprocessing

We describe a conditional ISD-style preprocessing technique applicable in settings where the error distribution, or at least the expected error weight, is known. The goal of this preprocessing is not to reduce the projected-error guessing cost, which remains subexponential when $q = \mathrm{poly}(n)$, but rather to redistribute the total weight between the secret and the error so that the transformed secret vector has smaller relative weight.

Observe that, the ternary LWE instance can be rewritten as

$$(\mathbf{I} \mid \mathbf{A})(-\mathbf{e}, \mathbf{s}) = \mathbf{b} \bmod q, \qquad \mathbf{s}, \mathbf{e} \in \{-1, 0, 1\}^n.$$

Let $\mathbf{M} := (\mathbf{I} \mid \mathbf{A}) \in \mathbb{Z}_q^{n \times 2n}$. Choose a uniformly random permutation matrix $\mathbf{P} \in \mathbb{Z}^{2n \times 2n}$, and decompose $\mathbf{MP} = (\mathbf{M}_0 \mid \mathbf{M}_1)$, and $\mathbf{P}^{-1}(-\mathbf{e}, \mathbf{s}) = (-\mathbf{e}', \mathbf{s}')$. If $\mathbf{M}_0$ is not invertible over $\mathbb{Z}_q$, we resample the permutation matrix $\mathbf{P}$. Otherwise, we define $\mathbf{A}' = \mathbf{M}_0^{-1}\mathbf{M}_1$ and $\mathbf{b}' = \mathbf{M}_0^{-1}\mathbf{b}$. This yields the transformed relation

$$(\mathbf{I} \mid \mathbf{A}')(-\mathbf{e}', \mathbf{s}') = \mathbf{b}', \qquad \text{or equivalently,} \qquad \mathbf{A}'\mathbf{s}' = \mathbf{b}' + \mathbf{e}' \bmod q.$$

Hence, the original LWE instance is transformed into another LWE instance whose secret and error weights depend on how the permutation redistributes the coordinates of $(-\mathbf{e}, \mathbf{s})$. Assume that the original secret vector has relative weight $\omega := \frac{w}{n}$, and that the error vector follows the balanced uniform ternary distribution. Then, in expectation, $|\mathbf{e}|_0 = |\mathbf{e}|_1 = |\mathbf{e}|_{-1} = \frac{n}{3}$. Our objective is to construct a transformed secret vector $\mathbf{s}'$ with reduced relative weight, satisfying

$$|\mathbf{s}'|_1 = \left(\frac{\omega}{2} - \delta_1\right) n, \quad \text{and} \quad |\mathbf{s}'|_{-1} = \left(\frac{\omega}{2} - \delta_2\right) n$$

where $\delta_1, \delta_2 \leqslant \frac{w}{2}$ and $\delta_1 + \delta_2 \leqslant \frac{1}{3}$. Consequently, the zero entries in $\mathbf{s}'$ satisfies $|\mathbf{s}'|_0 = (1 - \omega + \delta_1 + \delta_2)$. Since the total number of nonzero coordinates is preserved under permutation, the transformed error vector necessarily satisfies

$$|\mathbf{e}'|_1 = \left(\frac{1}{3} + \delta_1\right) n, \quad |\mathbf{e}'|_{-1} = \left(\frac{1}{3} + \delta_2\right) n, \qquad \text{and} \quad |\mathbf{e}'|_0 = \left(\frac{1}{3} - \delta_1 - \delta_2\right) n.$$

The probability that a uniformly random permutation produces such a split is therefore

$$\frac{\binom{n}{(\omega/2-\delta_1)n,(\omega/2-\delta_2)n,\cdot}\binom{n}{(1/3+\delta_1)n,(1/3+\delta_2)n,\cdot}}{\binom{2n}{(\omega/2+1/3)n,(\omega/2+1/3)n,\cdot}} = \tilde{\mathcal{O}}\left(2^{-\varphi(\delta_1,\delta_2)n}\right),$$

where the exponent $\varphi(\delta_1, \delta_2)$ is obtained using the multinomial approximation formulas from Section 2.2. More precisely,

$$\varphi(\delta_1, \delta_2) = 2H\left(\frac{1}{6} + \frac{\omega}{4}, \frac{1}{6} + \frac{\omega}{4}, \cdot\right) - H\left(\frac{\omega}{2} - \delta_1, \frac{\omega}{2} - \delta_2, \cdot\right) - H\left(\frac{1}{3} + \delta_1, \frac{1}{3} + \delta_2, \cdot\right).$$

After a successful permutation, we apply the HYBRID attack to the transformed secret vector of relative weight $(\omega'_1, \omega'_{-1}) = (\omega/2 - \delta_1, \omega/2 - \delta_2)$. If $c_{\text{HYBRID}}(\omega'_1, \omega'_{-1})$ denotes the optimized HYBRID exponent corresponding to relative weight $(\omega'_1, \omega'_{-1})$, then the resulting ISD-enhanced exponent becomes

$$c_{\text{HYBRID+ISD}} = \min_{\substack{0 \leqslant \delta_1, \delta_2 \leqslant \omega/2 \\ \delta_1 + \delta_2 \leqslant 1/3}} \{\varphi(\delta_1, \delta_2) + c_{\text{HYBRID}}(\omega/2 - \delta_1, \omega/2 - \delta_2)\}.$$

For the case where both the secret and the error follow the uniform ternary distribution, we have $\omega = 0.667$. In this regime, the ISD preprocessing step further improves the asymptotic complexity exponent of the HYBRID instantiation, reducing it from 0.867 to 0.8595.

Moreover, when the error vector follows the uniform ternary distribution, we may choose the permutation matrix $\mathbf{P}$ so that $\mathbf{P}(-\mathbf{e}, \mathbf{s}) = (\mathbf{s}, -\mathbf{e})$. Under this transformation, an LWE instance with secret relative weight ω is converted into an equivalent instance whose transformed secret vector follows the uniform ternary distribution. Consequently, the preprocessing becomes advantageous whenever the original secret relative weight satisfies $0.667 < \omega < 0.87$, see Figure 7.

Remark 3. The optimized preprocessing does not preserve the balance between the positive and negative entries of the secret. Imposing the balanced condition $\delta_1 = \delta_2$ gives a time exponent of 0.8606, whereas allowing an unrestricted optimization yields $\delta_1 > \delta_2$ and reduces the exponent to 0.8595. Thus, allowing an asymmetric weight distribution provides additional flexibility in the optimization and leads to a more favorable complexity.

5 Improved Complexity Bounds for Ternary LWE with Polynomial-Memory

In this section, we evaluate the asymptotic complexity of our best-performing instantiation, namely the HYBRID instantiation, and compare it with the currently best known polynomial-memory algorithm for ternary LWE, namely NESTED-2* of Esser *et al.* [14].

Table 1 reports the optimized runtime exponents for several representative relative weights $\omega = w/n$ relevant to ternary secret schemes in the NTRU Cryptosystem family. We observe that the HYBRID instantiation consistently outperforms NESTED-2* across the entire range of considered relative weights. In particular, the improvement becomes especially pronounced in the cryptographically relevant regime around balanced ternary secrets.

For the balanced-weight case $\omega = 0.667$, the complexity exponent decreases from 0.926 for NESTED-2* (with ISD preprocessing) to 0.8595 using our HYBRID framework combined with the ISD preprocessing technique from Section 4.4. This corresponds to an asymptotic improvement of approximately $2^{0.0665n}$. Table 2 summarizes the comparison for balanced-weight case. Further, Figure 7 provides a visual comparison between the two approaches. The left panel plots the optimized asymptotic runtime exponents of NESTED-2* and the proposed

Table 1. Comparison of the optimized asymptotic runtime exponents $\log(T)/n$ for Nested-$2^\star$ [14] and the proposed Hybrid instantiation over several representative relative secret weights $\omega = w/n$.

Relative weight(ω)	Nested-$2^\star$ [14]	Hybrid
0.300	0.6482	**0.6303**
0.375	0.7272	**0.7032**
0.441	0.7928	**0.7597**
0.500	0.8425	**0.7981**
0.621	0.9140	**0.8481**
0.668	0.930	**0.8674**

Hybrid instantiation as functions of the relative secret weight ω. The right panel illustrates the improvement achieved by the Hybrid framework, measured as the difference in the corresponding exponents. The figure shows that, although both approaches exhibit comparable performance for very small relative weights, the advantage of the Hybrid framework grows substantially in the cryptographically relevant region. In particular, the improvement reaches approximately $2^{0.05n}$ over a cryptographic relevant parameters, demonstrating that the additional flexibility introduced by the shifted and hybrid decompositions yields a significant asymptotic gain while preserving polynomial-memory complexity.

Table 2. Comparison of asymptotic runtime exponents for balanced ternary secret LWE ($\omega = 0.667$).

Algorithms	May [32]	May [32]	Nested-$2^\star$+ISD [14]	Hybrid	Hybrid+ISD
log(T)/n	**0.379**	1.063	0.926	0.867	**0.8595**
Memory	**Exp**	Poly	Poly	Poly	**Poly**

6 Applications to Kyber and Dilithium Secret Distributions

The secret keys used in Kyber [7] and Dilithium [13] follow significantly different distributions from the ternary setting considered previously. More precisely, the secret distribution in Kyber follows the centered binomial distribution $\mathcal{B}(\eta)$, $\eta \in \{2, 3\}$, whereas the secret distribution in Dilithium is uniform over $\{0, \pm 1, \pm 2\}$, which we denote by $\mathcal{U}(2)$.

In these settings, the secret vector no longer has a fixed Hamming weight, and hence we do not have explicit prior knowledge of its precise weight distribution. On the other hand, both Algorithm 1 and Algorithm 2 require structured information about the distribution of secret coordinates.

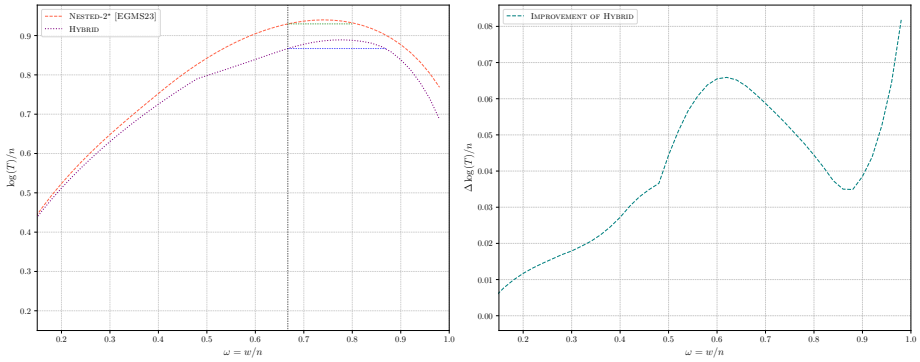


Fig. 7. Comparison of the asymptotic runtime exponents of Nested-2* [14] and the proposed Hybrid instantiation as a function of the relative secret weight $\omega = w/n$. The left panel shows the optimized exponents $\log(T)/n$, while the right panel illustrates the improvement achieved by the Hybrid framework over Nested-2*.

To address this issue, we follow the approach introduced by Glaser and May [21], which allows one to restrict the secret vector to its expected weight distribution with only subexponential overhead. More precisely, for a secret distribution D, the technique enforces $|\mathbf{s}|_i = n\mathbb{P}_{X\leftarrow D}[X = i]$ for each coordinate value i. Esser *et al.* [14] adopt the same heuristic assumption in their analysis, and we follow this approach throughout this section.

Table 3. Comparison of the asymptotic runtime exponents for Nested-3* and Improved-Nested-3* for the secret distributions used in Kyber and Dilithium.

Key-Distribution	Nested-3* [14]		Improved-Nested-3*	
	$\frac{\log T}{n}$	$\log_{\lvert\mathrm{Dom}(\mathbf{s})\rvert} T$	$\frac{\log T}{n}$	$\log_{\lvert\mathrm{Dom}(\mathbf{s})\rvert} T$
$\mathcal{U}(1)$	0.9297	0.5866	**0.8847**	**0.5582**
$\mathcal{U}(2)$	1.2815	0.5519	**1.2483**	**0.5376**
$\mathcal{U}(3)$	1.5352	0.5468	**1.4960**	**0.5329**
$\mathcal{B}(1)$	0.8425	0.5617	**0.8044**	**0.5363**
$\mathcal{B}(2)$	1.0404	0.5124	**1.0294**	**0.5069**
$\mathcal{B}(3)$	1.1854	0.5080	**1.1778**	**0.5048**

To analyze the complexity for the secret distributions used in Kyber and Dilithium, we extend our previous instantiations accordingly. In particular, we extend Improved-Nested-2* to Improved-Nested-3*, where vector coordinates are represented using $\{0, \pm1, \pm2, \pm3\}$, and we extend Shifted-3 to Shifted-4, where vector coordinates are represented using $\{0, \pm1, \pm2, \pm3, \pm4\}$.

Our observations indicate that the Shifted-4 instantiation does not improve the asymptotic complexity exponent beyond that achieved by Improved-Nested-3*. Consequently, we focus only on the latter instantiation in the main text.

The Improved-Nested-3* framework is obtained by applying the Nested-3* instantiation of Esser *et al.* [14] within our improved framework given by Algorithm 2. However, for completeness, we provide the full derivation and parameter analysis in Appendix B.

Table 3 reports both the asymptotic runtime exponents $\frac{\log T}{n}$ and the normalized exponents $\log_{|\mathrm{Dom}(\mathbf{s})|} T$ for Nested-3*[2] and Improved-Nested-3*. In our computation, we consider $\mathrm{Dom}(\mathbf{s})$ as the original (given) secret domain, for example in the case of $\mathcal{U}(3)$,

$$\mathrm{Dom}(\mathbf{s}) = \left\{\mathbf{x} \in \{0, \pm 1, \pm 2, \pm 3\}^n : |\mathbf{x}|_{\pm i} = \frac{n}{7}, i = 0, 1, 2, 3\right\}.$$

We observe that Improved-Nested-3* consistently outperforms Nested-3* across all considered parameter settings. However, the improvement gradually decreases as the parameter η increases. A plausible explanation is that larger values of η already provide sufficient entropy in the representation counts, reducing the additional advantage obtained from enlarging the search domains in the improved framework.

7 Improved Nested Framework for Subset-Sum

The nested collision search framework of Esser and May [15], which we refer to as the EM framework, and the EGMS nested collision search framework introduced in Section 3 share a similar underlying structure. We refer the reader to [15] for a detailed description of the EM framework. In this section, we adapt the refinement developed in Section 4 to the subset-sum setting.

Recall from Definition 3 that, given a subset-sum instance $(\mathbf{a}, t) \in (\mathbb{Z}_{2^n})^{n+1}$, the goal is to find a vector $\mathbf{e} \in \{0,1\}^n$ with $\mathrm{wt}(\mathbf{e}) = n/2$ such that $\langle \mathbf{a}, \mathbf{e} \rangle = t \bmod 2^n$. As in Section 3.1, we first restrict the search space of $\mathbf{e}$ to

$$\mathcal{D} = \left(\mathcal{B}^{\bar{\gamma} n/4}\left(\frac{1-\beta\gamma}{2\bar{\gamma}}\right)\right)^4 \times \mathcal{B}^{\gamma n}\left(\frac{\beta}{2}\right),$$

where $\beta, \gamma \in [0,1]$ are optimization parameters and $\bar{\gamma} := 1 - \gamma$. This restriction is obtained by applying a random permutation to the coordinates of $\mathbf{a}$. We then partition $\mathbf{e}$ into four components, $\mathbf{e} = \mathbf{e}_1 + \mathbf{e}_2 + \mathbf{e}_3 + \mathbf{e}_4$. The domains of the four

[2]Note that the complexity values corresponding to $\mathcal{U}(3)$ and $\mathcal{B}(3)$ reported in Table 3 are slightly larger than those stated in Section 6 of [14]. This discrepancy originates from a minor implementation issue in the optimization code of [14]: in the computation of the number of middle-layer representations, the parameter `x.r` was used instead of `x.mr`. The code is available on: https://github.com/arindamIITM/Small-LWE-Keys.

components are defined by

$$\mathcal{T}_1 = \mathbf{0}^{\bar{\gamma}n/4} \times \mathbf{0}^{\bar{\gamma}n/4} \times \mathbf{0}^{\bar{\gamma}n/4} \times \mathcal{B}^{\bar{\gamma}n/4}\left(\frac{1-\beta\gamma}{2\bar{\gamma}}\right) \times \mathcal{B}^{\gamma n}\left(\frac{\beta}{8}\right),$$
$$\mathcal{T}_2 = \mathbf{0}^{\bar{\gamma}n/4} \times \mathbf{0}^{\bar{\gamma}n/4} \times \mathcal{B}^{\bar{\gamma}n/4}\left(\frac{1-\beta\gamma}{2\bar{\gamma}}\right) \times \mathbf{0}^{\bar{\gamma}n/4} \times \mathcal{B}^{\gamma n}\left(\frac{\beta}{8}\right),$$
$$\mathcal{T}_3 = \mathbf{0}^{\bar{\gamma}n/4} \times \mathcal{B}^{\bar{\gamma}n/4}\left(\frac{1-\beta\gamma}{2\bar{\gamma}}\right) \times \mathbf{0}^{\bar{\gamma}n/4} \times \mathbf{0}^{\bar{\gamma}n/4} \times \mathcal{B}^{\gamma n}\left(\frac{\beta}{8}\right),$$
$$\mathcal{T}_4 = \mathcal{B}^{\bar{\gamma}n/4}\left(\frac{1-\beta\gamma}{2\bar{\gamma}}\right) \times \mathbf{0}^{\bar{\gamma}n/4} \times \mathbf{0}^{\bar{\gamma}n/4} \times \mathbf{0}^{\bar{\gamma}n/4} \times \mathcal{B}^{\gamma n}\left(\frac{\beta}{8}\right).$$

For this weight distribution to be feasible, we require $\frac{(1-\beta\gamma)n}{8} \leqslant \frac{\bar{\gamma}n}{4}$. The corresponding domain sizes are

$$|\mathcal{D}| = \binom{\bar{\gamma}n/4}{(1-\beta\gamma)n/8}^4 \binom{\gamma n}{\beta\gamma n/2}, \quad \text{and} \quad |\mathcal{T}_i| = \binom{\bar{\gamma}n/4}{(1-\beta\gamma)n/8}\binom{\gamma n}{\beta\gamma n/8}, \qquad i \in \{1,\ldots,4\}.$$

Using the entropy approximation for binomial coefficients, we write $|\mathcal{D}| = 2^{\delta n}$, where $\delta = \bar{\gamma}H_2\left(\frac{1-\beta\gamma}{2\bar{\gamma}}\right) + \gamma H_2\left(\frac{\beta}{2}\right)$. In the original EM framework, the modulus of the projected equation is chosen as $\langle \mathbf{a}, \mathbf{x}\rangle = t \bmod 2^{\delta n}$, which heuristically suffices to uniquely determine the secret vector. The Pollard-Rho collision searches are then balanced by imposing $|\mathcal{T}_i| = 2^{\delta n/2}$.

Following the approach of Section 4, we relax this balancing condition by enlarging the projection space. More precisely, for an optimization parameter $\lambda \geqslant \delta$, we consider the projected subset-sum equation $\langle \mathbf{a}, \mathbf{x}\rangle = t \bmod 2^{\lambda n}$, and impose $|\mathcal{T}_i| = 2^{\lambda n/2}$, $i \in \{1,\ldots,4\}$. As in Section 4, increasing the projection size helps to increase the number of representations at the intermediate collision levels. We therefore optimize λ jointly with β and γ. Since the ambient modulus is 2^n, we require $\delta \leqslant \lambda \leqslant 1$.

We consider the projected subset-sum equation

$$\langle \mathbf{a}, \mathbf{x}_1 + \mathbf{x}_2 + \mathbf{x}_3 + \mathbf{x}_4\rangle = t \bmod 2^{\lambda n}.$$

To construct the nested collision search, we introduce an intermediate value $r \in \mathbb{Z}_{2^{\lambda n/2}}$ chosen uniformly at random from $\mathbb{Z}_{2^{\lambda n/2}}$ and split the above equation into the two conditions

$$\langle \mathbf{a}, \mathbf{x}_1 + \mathbf{x}_2\rangle = r \bmod 2^{\lambda n/2}, \quad \text{and} \quad \langle \mathbf{a}, \mathbf{x}_3 + \mathbf{x}_4\rangle = t - r \bmod 2^{\lambda n/2}.$$

We define the Layer-1 functions $f_i : \mathcal{T}_i \longrightarrow \mathbb{Z}_{2^{\lambda n/2}}$, $i \in \{1,2,3,4\}$, as follows:

$$f_1, f_3 : \mathbf{x} \mapsto \langle \mathbf{a}, \mathbf{x}\rangle \bmod 2^{\lambda n/2}, \quad f_2 : \mathbf{x} \mapsto r - \langle \mathbf{a}, \mathbf{x}\rangle \bmod 2^{\lambda n/2}, \quad f_4 : \mathbf{x} \mapsto t - r - \langle \mathbf{a}, \mathbf{x}\rangle \bmod 2^{\lambda n/2},$$

and Layer-2 functions as follows:

$$g_1 : x \mapsto \langle \mathbf{a}, \mathbf{z}_1 + \mathbf{z}_2\rangle_{[\lambda n/2:\lambda n]}, \qquad (\mathbf{z}_1, \mathbf{z}_2) \leftarrow \textsc{Rho}(f_1, f_2, x)$$
$$g_2 : x \mapsto t - \langle \mathbf{a}, \mathbf{z}_3 + \mathbf{z}_4\rangle_{[\lambda n/2:\lambda n]}, \qquad (\mathbf{z}_3, \mathbf{z}_4) \leftarrow \textsc{Rho}(f_3, f_4, x).$$

Here, as in Section 3, we omit the flavors to ease the notations.

The nested collision search proceeds as follows. We first choose a uniformly random starting point $z \in \mathbb{Z}_{2^{\lambda n/2}}$ and perform $(y_1, y_2) \leftarrow \textsc{Rho}(g_1, g_2, z)$. The resulting values y_1 and y_2 determine the corresponding Layer-1 collisions through

$(\mathbf{z}_1, \mathbf{z}_2) \leftarrow \textsc{Rho}(f_1, f_2, y_1)$ and $(\mathbf{z}_3, \mathbf{z}_4) \leftarrow \textsc{Rho}(f_3, f_4, y_2)$. We then test whether $\mathbf{z}_1 + \mathbf{z}_2 + \mathbf{z}_3 + \mathbf{z}_4 \in \{0,1\}^n$ and, if so, whether it satisfies the original subset-sum equation. If the resulting vector is a valid solution, we return it; otherwise, we repeat the procedure with a fresh starting point.

Complexity Analysis. We first consider the probability that the solution $\mathbf{e}$ has the prescribed weight distribution. Since the original solution is chosen uniformly from $\mathcal{B}^n(1/2)$, this probability is $p_1 = \frac{|\mathcal{D}|}{\binom{n}{n/2}} = \widetilde{\mathcal{O}}\left(2^{(\delta-1)n}\right)$.

Conditioned on $\mathbf{e} \in \mathcal{D}$, consider the representation of $\mathbf{e}$ as $(\mathbf{e}'_1, \mathbf{e}'_2) = (\mathbf{e}_1 + \mathbf{e}_2, \mathbf{e}_3 + \mathbf{e}_4)$. The number of such representations is $\binom{\beta\gamma n/2}{\beta\gamma n/4} = \widetilde{\mathcal{O}}\left(2^{\beta\gamma n/2}\right)$. Since the intermediate value r is chosen uniformly from $\mathbb{Z}_{2^{\lambda n/2}}$, the probability that the chosen value of r corresponds to a representation of the solution is therefore $p_2 = \widetilde{\mathcal{O}}\left(\frac{2^{\beta\gamma n/2}}{2^{\lambda n/2}}\right) = \widetilde{\mathcal{O}}\left(2^{(\beta\gamma-\lambda)n/2}\right)$.

Next, for a fixed valid pair $(\mathbf{e}'_1, \mathbf{e}'_2)$, the number of representations in terms of $(\mathbf{e}_1, \mathbf{e}_2, \mathbf{e}_3, \mathbf{e}_4)$ is $\binom{\beta\gamma n/4}{\beta\gamma n/8}^2 = \widetilde{\mathcal{O}}\left(2^{\beta\gamma n/2}\right)$. Since g_1 and g_2 has a total $\tilde{\mathcal{O}}(2^{\lambda n/2})$ number of collisions, the probability that a randomly obtained collision corresponds to a valid representation of the solution is $p_3 = \widetilde{\mathcal{O}}\left(\frac{2^{\beta\gamma n/2}}{2^{\lambda n/2}}\right) = \widetilde{\mathcal{O}}\left(2^{(\beta\gamma-\lambda)n/2}\right)$.

Consequently, the overall success probability of one iteration is $p_1 p_2 p_3$. Thus, the expected number of iterations required to recover the solution is $(p_1 p_2 p_3)^{-1}$ and each iteration requires $\widetilde{\mathcal{O}}(2^{\lambda n/2})$ time for the nested collision search. Therefore, the overall runtime is $T = \widetilde{\mathcal{O}}\left(2^{(1-\delta-\beta\gamma+3\lambda/2)n}\right)$.

Optimizing the exponent over the parameters β, γ, and λ yields $\beta = 0.956$, $\gamma = 0.913$, $\lambda = 1$. For these parameter values, we obtain $T = \tilde{\mathcal{O}}\left(2^{0.6432n}\right)$. This improves upon the $\tilde{\mathcal{O}}(2^{0.645n})$ runtime of the EM framework [15].

8 Conclusion

In this work, we have presented improved polynomial-memory algorithms for solving small max-norm variants of the LWE problem building upon the EGMS framework. In particular, for uniformly random ternary secrets of length n, our best polynomial-memory attack improves the asymptotic runtime from $2^{0.926n}$ to $2^{0.8595n}$. We have also demonstrated that the refinement introduced in this work is not restricted to the LWE setting. By applying the same principle of enlarging the projection space to the nested collision search EM framework, we obtain an improved polynomial-memory algorithm for random subset-sum problem, reducing the asymptotic runtime from $2^{0.645n}$ to $2^{0.6432n}$. Although the improvement for random subset-sum is modest, this result provides an independent application of our approach and illustrates the broader applicability of the proposed refinement to nested collision search algorithms.

Beyond the concrete complexity improvements, our work highlights the effectiveness of refined representation structures in the design of memory-efficient combinatorial attacks. We hope that the techniques introduced in this work will contribute toward a deeper understanding of the complexity landscape and inspire further progress on polynomial-memory collision search algorithms.

References

1. Ieee standard specification for public key cryptographic techniques based on hard problems over lattices. IEEE Std 1363.1-2008 pp. 1–81 (2009). https://doi.org/10.1109/IEEESTD.2009.4800404
2. Alekhnovich, M.: More on average case vs approximation complexity. In: 44th Annual IEEE Symposium on Foundations of Computer Science, 2003. Proceedings. pp. 298–307. IEEE (2003)
3. Becker, A., Coron, J.S., Joux, A.: Improved generic algorithms for hard knapsacks. In: Paterson, K.G. (ed.) Advances in Cryptology – EUROCRYPT 2011. Lecture Notes in Computer Science, vol. 6632, pp. 364–385. Springer, Heidelberg (2011). https://doi.org/10.1007/978-3-642-20465-4_21
4. Becker, A., Joux, A., May, A., Meurer, A.: Decoding random binary linear codes in 2 n/20: How 1+ 1= 0 improves information set decoding. In: Annual international conference on the theory and applications of cryptographic techniques. pp. 520–536. Springer (2012)
5. Bernstein, D.J., Chuengsatiansup, C., Lange, T., van Vredendaal, C.: NTRU prime: Reducing attack surface at low cost. In: Selected Areas in Cryptography – SAC 2017. pp. 235–260. Lecture Notes in Computer Science, Springer (2017)
6. Bonnetain, X., Bricout, R., Schrottenloher, A., Shen, Y.: Improved classical and quantum algorithms for subset-sum. In: Moriai, S., Wang, H. (eds.) Advances in Cryptology – ASIACRYPT 2020. pp. 633–666. Springer International Publishing, Cham (2020)
7. Bos, J., Ducas, L., Kiltz, E., Lepoint, T., Lyubashevsky, V., Schanck, J.M., Schwabe, P., Seiler, G., Stehle, D.: Crystals - kyber: A cca-secure module-lattice-based kem. In: 2018 IEEE European Symposium on Security and Privacy (EuroS&P). pp. 353–367 (2018). https://doi.org/10.1109/EuroSP.2018.00032
8. Both, L., May, A.: Decoding linear codes with high error rate and its impact for lpn security. In: International Conference on Post-Quantum Cryptography. pp. 25–46. Springer (2018)
9. Brent, R.P.: An improved Monte Carlo factorization algorithm. BIT Numerical Mathematics **20**(2), 176–184 (1980). https://doi.org/10.1007/BF01933190
10. Chor, B., Rivest, R.L.: A knapsack-type public key cryptosystem based on arithmetic in finite fields. IEEE Transactions on Information Theory **34**(5), 901–909 (1988)
11. Dinur, I., Dunkelman, O., Keller, N., Shamir, A.: Memory-efficient algorithms for finding needles in haystacks. In: Robshaw, M., Katz, J. (eds.) Advances in Cryptology – CRYPTO 2016, Part II. Lecture Notes in Computer Science, vol. 9815, pp. 185–206. Springer, Heidelberg (2016). https://doi.org/10.1007/978-3-662-53008-5_7
12. Ducas, L., Durmus, A., Lepoint, T., Lyubashevsky, V.: Lattice signatures and bimodal Gaussians. In: Canetti, R., Garay, J.A. (eds.) Advances in Cryptology – CRYPTO 2013, Part I. Lecture Notes in Computer Science, vol. 8042, pp. 40–56. Springer, Heidelberg (Aug 2013)
13. Ducas, L., Kiltz, E., Lepoint, T., Lyubashevsky, V., Schwabe, P., Seiler, G., Stehlé, D.: Crystals-dilithium: A lattice-based digital signature scheme. IACR Transactions on Cryptographic Hardware and Embedded Systems **2018**(1), 238–268 (Feb 2018). https://doi.org/10.13154/tches.v2018.i1.238-268, https://tches.iacr.org/index.php/TCHES/article/view/839
14. Esser, A., Girme, R., Mukherjee, A., Sarkar, S.: Memory-efficient attacks on small lwe keys. In: Guo, J., Steinfeld, R. (eds.) Advances in Cryptology – ASIACRYPT 2023. pp. 72–105. Springer Nature Singapore, Singapore (2023)

15. Esser, A., May, A.: Low weight discrete logarithm and subset sum in $2^{0.65}$with polynomial memory. In: Canteaut, A., Ishai, Y. (eds.) Advances in Cryptology – EUROCRYPT 2020. pp. 94–122. Springer International Publishing, Cham (2020)
16. Esser, A., Mukherjee, A., Sarkar, S.: Memory-efficient attacks on small lwe keys. Journal of Cryptology **37**(4), 36 (2024). https://doi.org/10.1007/s00145-024-09516-3, https://doi.org/10.1007/s00145-024-09516-3
17. Esser, A., Zweydinger, F.: New time-memory trade-offs for subset sum–improving isd in theory and practice. In: Annual International Conference on the Theory and Applications of Cryptographic Techniques. pp. 360–390. Springer (2023)
18. Faust, S., Masny, D., Venturi, D.: Chosen-ciphertext security from subset sum. In: Public-Key Cryptography–PKC 2016: 19th IACR International Conference on Practice and Theory in Public-Key Cryptography, Taipei, Taiwan, March 6-9, 2016, Proceedings, Part I. pp. 35–46. Springer (2016)
19. Gentry, C.: Fully homomorphic encryption using ideal lattices. p. 169–178. STOC '09, Association for Computing Machinery, New York, NY, USA (2009). https://doi.org/10.1145/1536414.1536440, https://doi.org/10.1145/1536414.1536440
20. Gentry, C., Peikert, C., Vaikuntanathan, V.: Trapdoors for hard lattices and new cryptographic constructions. In: Proceedings of the Fortieth Annual ACM Symposium on Theory of Computing. p. 197–206. STOC '08, Association for Computing Machinery, New York, NY, USA (2008). https://doi.org/10.1145/1374376.1374407, https://doi.org/10.1145/1374376.1374407
21. Glaser, T., May, A.: How to enumerate lwe keys as narrow as in kyber/dilithium. In: Deng, J., Kolesnikov, V., Schwarzmann, A.A. (eds.) Cryptology and Network Security. pp. 75–100. Springer Nature Singapore, Singapore (2023)
22. Güneysu, T., Lyubashevsky, V., Pöppelmann, T.: Practical lattice-based cryptography: A signature scheme for embedded systems. In: Prouff, E., Schaumont, P. (eds.) Cryptographic Hardware and Embedded Systems – CHES 2012. Lecture Notes in Computer Science, vol. 7428, pp. 530–547. Springer, Heidelberg (Sep 2012)
23. Hartmanis, J.: Computers and intractability: a guide to the theory of np-completeness (michael r. garey and david s. johnson). Siam Review **24**(1), 90 (1982)
24. Hellman, M.: Hiding information and signatures in trapdoor knapsacks. IEEE Transactions on Information Theory **24**(5), 525–530 (1978)
25. Hoffstein, J., Pipher, J., Silverman, J.H.: NTRU: A ring-based public key cryptosystem. In: Algorithmic Number Theory Symposium – ANTS-III. pp. 267–288. Lecture Notes in Computer Science, Springer (1998)
26. Howgrave-Graham, N.: A hybrid lattice-reduction and meet-in-the-middle attack against NTRU. In: Advances in Cryptology – CRYPTO 2007. Lecture Notes in Computer Science, Springer, Heidelberg (2007)
27. Howgrave-Graham, N., Joux, A.: New generic algorithms for hard knapsacks. In: Advances in Cryptology – EUROCRYPT 2010. Lecture Notes in Computer Science, Springer, Heidelberg (2010)
28. Hülsing, A., Rijneveld, J., Schanck, J., Schwabe, P.: High-speed key encapsulation from ntru. In: Fischer, W., Homma, N. (eds.) Cryptographic Hardware and Embedded Systems – CHES 2017. pp. 232–252. Springer International Publishing, Cham (2017)
29. Knuth, D.E.: The Art of Computer Programming, Volume II: Seminumerical Algorithms. Addison-Wesley, 3rd edn. (1998)
30. Lagarias, J.C., Odlyzko, A.M.: Solving low-density subset sum problems. Journal of the ACM (JACM) **32**(1), 229–246 (1985)

31. Lyubashevsky, V., Palacio, A., Segev, G.: Public-key cryptographic primitives provably as secure as subset sum. In: Theory of Cryptography Conference. pp. 382–400. Springer (2010)
32. May, A.: How to meet ternary LWE keys. In: Advances in Cryptology – CRYPTO 2021, Part II. Lecture Notes in Computer Science, Springer, Heidelberg (2021)
33. May, A., Meurer, A., Thomae, E.: Decoding random linear codes in $\tilde{\mathcal{O}}(2^{0.054n})$. In: Lee, D.H., Wang, X. (eds.) Advances in Cryptology – ASIACRYPT 2011. pp. 107–124. Springer Berlin Heidelberg, Berlin, Heidelberg (2011)
34. Nivasch, G.: Cycle detection using a stack. Information Processing Letters **90**(3), 135–140 (2004)
35. Odlyzko, A.M.: The rise and fall of knapsack cryptosystems. Cryptology and computational number theory **42**(2) (1990)
36. van Oorschot, P.C., Wiener, M.J.: Parallel collision search with application to hash functions and discrete logarithms. In: Proceedings of the 2nd ACM Conference on Computer and Communications Security. pp. 210–218. ACM (1994)
37. Pollard, J.M.: A Monte Carlo method for factorization. BIT Numerical Mathematics **15**(3), 331–334 (1975). https://doi.org/10.1007/BF01933667
38. Regev, O.: New lattice-based cryptographic constructions. J. ACM **51**(6), 899–942 (Nov 2004). https://doi.org/10.1145/1039488.1039490, https://doi.org/10.1145/1039488.1039490
39. Regev, O.: On lattices, learning with errors, random linear codes, and cryptography. Journal of the ACM **56**(6), 34:1–34:40 (2009). https://doi.org/10.1145/1568318.1568324, preliminary version in STOC 2005

Appendix

A Analysis of HYBRID Instantiation

For the HYBRID instantiation, the search space of the shifted secret vector is

$$\mathcal{D} = \left(\tau^{\bar{\gamma}n/4}(\alpha w/8)\right)^4 \times \tau^{\gamma_2 n}(\beta_2\gamma_2 w/2) \times \bar{\tau}^{\gamma_1 n}\left(0,0,0,\frac{\beta_1\gamma_1 w}{2},\gamma_1 n-\beta_1\gamma_1 w,\frac{\beta_1\gamma_1 w}{2},0\right),$$

whose cardinality is

$$|\mathcal{D}| = \binom{\bar{\gamma}n/4}{\alpha w/8,\alpha w/8,\cdot}^4 \binom{\gamma_2 n}{\beta_2\gamma_2 w/2,\beta_2\gamma_2 w/2,\cdot}\binom{\gamma_1 n}{\beta_1\gamma_1 w/2,\beta_1\gamma_1 w/2,\cdot}.$$

The analysis naturally decomposes into two independent components:

- the *shifted component*, corresponding to the $\gamma_1 n$ coordinates; and
- the *nested component*, corresponding to the $\gamma_2 n$ coordinates.

The overall representation counts are then obtained by combining the contributions from these two parts.

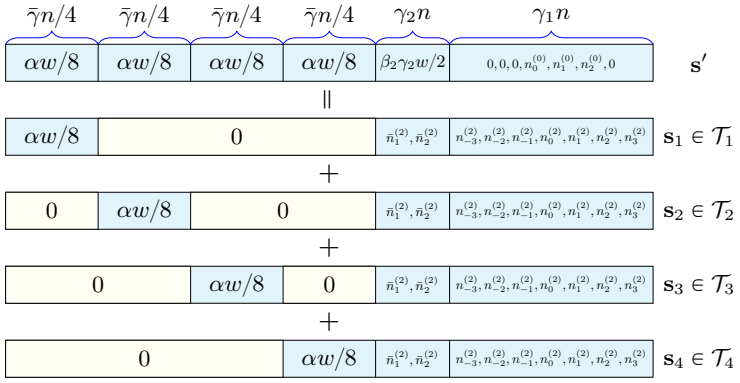


Fig. 8. Weight distribution of lower-level function domains for SHIFTED-3 instantiation.

The domains of the vectors $\mathbf{s}_i$ are defined by (see Figure 8)

$$\begin{aligned}
\mathcal{T}_1 &= \tau^{\bar{\gamma}n/4}(\alpha w/8) \times \mathbf{0} \times \mathbf{0} \times \mathbf{0} \times \tau_2^{\gamma_2 n}(\bar{n}_1^{(2)},\bar{n}_2^{(2)}) \times \bar{\tau}^{\gamma_1 n}(n_{-3}^{(2)},n_{-2}^{(2)},n_{-1}^{(2)},n_0^{(2)},n_1^{(2)},n_2^{(2)},n_3^{(2)})\\
\mathcal{T}_2 &= \mathbf{0} \times \tau^{\bar{\gamma}n/4}(\alpha w/8) \times \mathbf{0} \times \mathbf{0} \times \tau_2^{\gamma_2 n}(\bar{n}_1^{(2)},\bar{n}_2^{(2)}) \times \bar{\tau}^{\gamma_1 n}(n_{-3}^{(2)},n_{-2}^{(2)},n_{-1}^{(2)},n_0^{(2)},n_1^{(2)},n_2^{(2)},n_3^{(2)})\\
\mathcal{T}_3 &= \mathbf{0} \times \mathbf{0} \times \tau^{\bar{\gamma}n/4}(\alpha w/8) \times \mathbf{0} \times \tau_2^{\gamma_2 n}(\bar{n}_1^{(2)},\bar{n}_2^{(2)}) \times \bar{\tau}^{\gamma_1 n}(n_{-3}^{(2)},n_{-2}^{(2)},n_{-1}^{(2)},n_0^{(2)},n_1^{(2)},n_2^{(2)},n_3^{(2)})\\
\mathcal{T}_4 &= \mathbf{0} \times \mathbf{0} \times \mathbf{0} \times \tau^{\bar{\gamma}n/4}(\alpha w/8) \times \tau_2^{\gamma_2 n}(\bar{n}_1^{(2)},\bar{n}_2^{(2)}) \times \bar{\tau}^{\gamma_1 n}(n_{-3}^{(2)},n_{-2}^{(2)},n_{-1}^{(2)},n_0^{(2)},n_1^{(2)},n_2^{(2)},n_3^{(2)})
\end{aligned}$$

with cardinalities

$$|\mathcal{T}_i| = \binom{\bar{\gamma}n/4}{\alpha w/8,\alpha w/8,\cdot}\binom{\gamma_2 n}{\bar{n}_1^{(2)},\bar{n}_2^{(2)},\bar{n}_1^{(2)},\bar{n}_2^{(2)},\cdot}\binom{\gamma_1 n}{n_{-3}^{(2)},n_{-2}^{(2)},n_{-1}^{(2)},n_0^{(2)},n_1^{(2)},n_2^{(2)},n_3^{(2)}}.$$

The corresponding middle-layer domains are

$$\begin{array}{llllll}
\mathcal{D}_1 = \tau^{\bar{\gamma}n/4}(\alpha w/8) & \times \tau^{\bar{\gamma}n/4}(\alpha w/8) & \times \mathbf{0} & \times \mathbf{0} & \times \tau_2^{\gamma_2 n}(\bar{n}_1^{(1)}, \bar{n}_2^{(1)}) & \times \bar{\tau}^{\gamma_1 n}(n_{-3}^{(1)}, n_{-2}^{(1)}, n_{-1}^{(1)}, n_0^{(1)}, n_1^{(1)}, n_2^{(1)}, n_3^{(1)}) \\
\mathcal{D}_2 = \mathbf{0} & \times \mathbf{0} & \times \tau^{\bar{\gamma}n/4}(\alpha w/8) & \times \tau^{\bar{\gamma}n/4}(\alpha w/8) & \times \tau_2^{\gamma_2 n}(\bar{n}_1^{(1)}, \bar{n}_2^{(1)}) & \times \bar{\tau}^{\gamma_1 n}(n_{-3}^{(1)}, n_{-2}^{(1)}, n_{-1}^{(1)}, n_0^{(1)}, n_1^{(1)}, n_2^{(1)}, n_3^{(1)})
\end{array}$$

We now determine the parameters $n_i^{(j)}$ and $\bar{n}_i^{(j)}$.

We first analyze the shifted component corresponding to the $\gamma_1 n$ coordinates. From Section 4.2,

$$n_0^{(0)} = \frac{\beta_1\gamma_1 w}{2}, \qquad n_1^{(0)} = \gamma_1 n - \beta_1\gamma_1 w, \qquad n_2^{(0)} = \frac{\beta_1\gamma_1 w}{2}.$$

The corresponding middle-layer parameters are given by

$$\begin{aligned}
n_{-3}^{(1)} &= z_3^{(0)}, \qquad n_{-2}^{(1)} = z_2^{(0)} + o_2^{(0)}, \qquad n_{-1}^{(1)} = z_1^{(0)} + o_1^{(0)} + t_1^{(0)}, \\
n_0^{(1)} &= n_0^{(0)} - 2\big(z_1^{(0)} + z_2^{(0)} + z_3^{(0)}\big) + \frac{n_1^{(0)}}{2} - o_1^{(0)} - o_2^{(0)} + t_0^{(0)}, \\
n_1^{(1)} &= z_1^{(0)} + \frac{n_1^{(0)}}{2} - o_1^{(0)} - o_2^{(0)} + n_2^{(0)} - 2\big(t_1^{(0)} + t_0^{(0)}\big), \\
n_2^{(1)} &= z_2^{(0)} + o_1^{(0)} + t_0^{(0)}, \qquad n_3^{(1)} = z_3^{(0)} + o_2^{(0)} + t_1^{(0)}.
\end{aligned}$$

The corresponding lower-layer parameters are

$$\begin{aligned}
n_{-3}^{(2)} &= thm^{(1)} + tm_1^{(1)} + om_2^{(1)} + z_3^{(1)}, \\
n_{-2}^{(2)} &= \frac{n_{-3}^{(1)}}{2} - thm^{(1)} + tm_0^{(1)} + om_1^{(1)} + z_2^{(1)} + o_2^{(1)}, \\
n_{-1}^{(2)} &= \frac{n_{-3}^{(1)}}{2} - thm^{(1)} + n_{-2}^{(1)} - 2(tm_0^{(1)} + tm_1^{(1)}) + \frac{n_{-1}^{(1)}}{2} - om_1^{(1)} - om_2^{(1)} + z_1^{(1)} + o_1^{(1)} + t_1^{(1)}, \\
n_0^{(2)} &= thm^{(1)} + tm_0^{(1)} + \frac{n_{-1}^{(1)}}{2} - om_1^{(1)} - om_2^{(1)} + n_0^{(1)} - 2(z_1^{(1)} + z_2^{(1)} + z_3^{(1)}) + \frac{n_1^{(1)}}{2} - o_1^{(1)} - o_2^{(1)} + t_0^{(1)} + th^{(1)}, \\
n_1^{(2)} &= tm_1^{(1)} + om_1^{(1)} + z_1^{(1)} + \frac{n_1^{(1)}}{2} - o_1^{(1)} - o_2^{(1)} + n_2^{(1)} - 2(t_0^{(1)} + t_1^{(1)}) + \frac{n_3^{(1)}}{2} - th^{(1)}, \\
n_2^{(2)} &= om_2^{(1)} + z_2^{(1)} + o_1^{(1)} + t_0^{(1)} + \frac{n_3^{(1)}}{2} - th^{(1)}, \\
n_3^{(2)} &= z_3^{(1)} + o_2^{(1)} + t_1^{(1)} + th^{(1)}.
\end{aligned}$$

The total number of middle-layer representations $(\mathbf{a}_1, \mathbf{a}_2)$ corresponding to the shifted component is

$$R_1^{(S)} = \binom{n_0^{(0)}}{z_1^{(0)}, z_2^{(0)}, z_3^{(0)}, z_1^{(0)}, z_2^{(0)}, z_3^{(0)}, .} \binom{n_1^{(0)}}{o_1^{(0)}, o_2^{(0)}, o_1^{(0)}, o_2^{(0)}, n_1^{(0)}/2 - o_1^{(0)} - o_2^{(0)}, .} \binom{n_2^{(0)}}{t_0^{(0)}, t_1^{(0)}, t_0^{(0)}, t_1^{(0)}, .}.$$

Similarly, the total number of lower-layer representations corresponding to the shifted component is

$$\begin{aligned}
R_2^{(S)} = &\binom{n_{-3}^{(1)}}{thm^{(1)}, thm^{(1)}, \frac{n_{-3}^{(1)}}{2} - thm^{(1)}, .} \binom{n_{-2}^{(1)}}{tm_1^{(1)}, tm_0^{(1)}, tm_1^{(1)}, tm_0^{(1)}, .} \\
&\binom{n_{-1}^{(1)}}{om_1^{(1)}, om_2^{(1)}, om_1^{(1)}, om_2^{(1)}, \frac{n_{-1}^{(1)}}{2} - om_1^{(1)} - om_2^{(1)}, .} \binom{n_0^{(1)}}{z_3^{(1)}, z_2^{(1)}, z_1^{(1)}, z_3^{(1)}, z_2^{(1)}, z_1^{(1)}, .} \\
&\binom{n_1^{(1)}}{o_1^{(1)}, o_2^{(1)}, o_1^{(1)}, o_2^{(1)}, \frac{n_1^{(1)}}{2} - o_1^{(1)} - o_2^{(1)}, .} \binom{n_2^{(1)}}{t_1^{(1)}, t_0^{(1)}, t_1^{(1)}, t_0^{(1)}, .} \binom{n_3^{(1)}}{th^{(1)}, th^{(1)}, \frac{n_3^{(1)}}{2} - th^{(1)}, .}.
\end{aligned}$$

We now analyze the nested component corresponding to the $\gamma_2 n$ coordinates. Analogously to Section 3.2,

$$\bar{n}_0^{(0)} = \gamma_2 n - \beta_2\gamma_2 w, \qquad \bar{n}_1^{(0)} = \frac{\beta_2\gamma_2 w}{2}.$$

The middle-layer parameters are

$$\bar{n}_0^{(1)} = \bar{n}_0^{(0)} + \bar{n}_1^{(0)} - 2(\bar{o}_1^{(0)} + \bar{z}_1^{(0)} + \bar{z}_2^{(0)}),$$

$$\bar{n}_1^{(1)} = \frac{\bar{n}_1^{(0)}}{2} + \bar{z}_1^{(0)}, \qquad \bar{n}_2^{(1)} = \bar{o}_1^{(0)} + \bar{z}_2^{(0)}.$$

The lower-layer parameters are

$$\begin{aligned}
\bar{n}_0^{(2)} &= \bar{n}_0^{(1)} + \bar{n}_1^{(1)} - 2\big(\bar{o}_1^{(1)} + \bar{z}_1^{(1)} + \bar{z}_2^{(1)} - \bar{t}^{(1)}\big),\\
\bar{n}_1^{(2)} &= \frac{\bar{n}_1^{(1)}}{2} + \bar{n}_2^{(1)} + \bar{z}_1^{(1)} - 2\bar{t}^{(1)},\\
\bar{n}_2^{(2)} &= \bar{o}_1^{(1)} + \bar{z}_2^{(1)} + \bar{t}^{(1)}.
\end{aligned}$$

The corresponding representation counts are

$$R_1^{(N)} = \binom{\bar{n}_1^{(0)}}{\bar{o}_1^{(0)}, \bar{o}_1^{(0)}, \bar{n}_1^{(0)}/2 - \bar{o}_1^{(0)}, .}^2 \binom{\bar{n}_0^{(0)}}{\bar{z}_1^{(0)}, \bar{z}_1^{(0)}, \bar{z}_2^{(0)}, \bar{z}_2^{(0)}, .},$$

and

$$R_2^{(N)} = \binom{\bar{n}_2^{(1)}}{\bar{t}^{(1)}, \bar{t}^{(1)}, .}^2 \binom{\bar{n}_1^{(1)}}{\bar{o}_1^{(1)}, \bar{o}_1^{(1)}, \bar{n}_1^{(1)}/2 - \bar{o}_1^{(1)}, .}^2 \binom{\bar{n}_0^{(1)}}{\bar{z}_1^{(1)}, \bar{z}_1^{(1)}, \bar{z}_2^{(1)}, \bar{z}_2^{(1)}, .}.$$

All introduced parameters are optimization variables. Therefore, the total number of middle-layer representations of the shifted secret vector is

$$R_1 = R_1^{(S)} R_1^{(N)},$$

and the total number of lower-layer representations is

$$R_2 = R_2^{(S)} R_2^{(N)}.$$

B Analysis of the Improved-Nested-3* Instantiation

Consider a secret vector $\mathbf{s}$ with weight distribution

$$|\mathbf{s}|_{-1} = |\mathbf{s}|_1 = w_1, \qquad |\mathbf{s}|_{-2} = |\mathbf{s}|_2 = w_2, \qquad |\mathbf{s}|_{-3} = |\mathbf{s}|_3 = w_3,$$

and

$$|\mathbf{s}|_0 = n - 2(w_1 + w_2 + w_3).$$

Analogously to the Nested-2* instantiation, we restrict the search domain of the secret vector by partitioning the coordinates into a joint part and a disjoint part:

– The *joint part* has length γn and contains

$$n_1^{(0)} := \beta_1 \gamma w_1, \quad n_2^{(0)} := \beta_2 \gamma w_2, \quad n_3^{(0)} := \beta_3 \gamma w_3$$

coordinates equal to each of ± 1, ± 2 and ± 3, respectively. The remaining $n_0^{(0)} := \gamma n - 2\big(n_1^{(0)} + n_2^{(0)} + n_3^{(0)}\big)$ coordinates are equal to zero.

– The *disjoint part* has length $\bar{\gamma} n := (1-\gamma)n$, where the vectors $\mathbf{s}_1, \ldots, \mathbf{s}_4$ are pairwise disjoint. The remaining weights are distributed equally among the four vectors. More precisely, each vector contains

$$\alpha_1 := \frac{w_1 - n_1^{(0)}}{4}, \qquad \alpha_2 := \frac{w_2 - n_2^{(0)}}{4}, \qquad \alpha_3 := \frac{w_3 - n_3^{(0)}}{4}$$

coordinates equal to each of ± 1, ± 2 and ± 3, respectively.

Under this restriction, the restricted search domain of the secret vector becomes

$$\mathcal{D} = \Big(\tau_3^{\bar{\gamma} n/4}(\alpha_1, \alpha_2, \alpha_3)\Big)^4 \times \tau_3^{\gamma n}(n_1^{(0)}, n_2^{(0)}, n_3^{(0)}).$$

Here, the set $\tau_3^n(n_1, n_2, n_3)$ is defined by

$$\tau_3^n(n_1, n_2, n_3) = \{\mathbf{x} \in \{0, \pm 1, \pm 2, \pm 3\}^n : |\mathbf{x}|_{\pm i} = n_i,\ i = 1, 2, 3\}\,.$$

Consequently, the cardinality of the restricted search space is

$$|\mathcal{D}| = \binom{\bar{\gamma} n/4}{\alpha_1, \alpha_2, \alpha_3, \alpha_1, \alpha_2, \alpha_3, .}^4 \cdot \binom{\gamma n}{n_1^{(0)}, n_2^{(0)}, n_3^{(0)}, n_1^{(0)}, n_2^{(0)}, n_3^{(0)}, .}.$$

For the weight distribution in the disjoint part to be feasible, we require

$$2(\alpha_1 + \alpha_2 + \alpha_3) \leqslant \frac{\bar{\gamma} n}{4}.$$

The domains of the vectors $\mathbf{s}_i$ are defined by

$$\begin{array}{llllll}
\mathcal{T}_1 = \tau_3^{\bar{\gamma} n/4}(\alpha_1, \alpha_2, \alpha_3) & \times \mathbf{0} & \times \mathbf{0} & \times \mathbf{0} & \times \tau_3^{\gamma n}(n_1^{(2)}, n_2^{(2)}, n_3^{(2)}) \\
\mathcal{T}_2 = \mathbf{0} & \times \tau_3^{\bar{\gamma} n/4}(\alpha_1, \alpha_2, \alpha_3) & \times \mathbf{0} & \times \mathbf{0} & \times \tau_3^{\gamma n}(n_1^{(2)}, n_2^{(2)}, n_3^{(2)}) \\
\mathcal{T}_3 = \mathbf{0} & \times \mathbf{0} & \times \tau_3^{\bar{\gamma} n/4}(\alpha_1, \alpha_2, \alpha_3) & \times \mathbf{0} & \times \tau_3^{\gamma n}(n_1^{(2)}, n_2^{(2)}, n_3^{(2)}) \\
\mathcal{T}_4 = \mathbf{0} & \times \mathbf{0} & \times \mathbf{0} & \times \tau_3^{\bar{\gamma} n/4}(\alpha_1, \alpha_2, \alpha_3) & \times \tau_3^{\gamma n}(n_1^{(2)}, n_2^{(2)}, n_3^{(2)})
\end{array}$$

and the corresponding middle-layer domains are

$$\begin{array}{llllll}
\mathcal{D}_1 = \tau_3^{\bar{\gamma} n/4}(\alpha_1, \alpha_2, \alpha_3) & \times \tau_3^{\bar{\gamma} n/4}(\alpha_1, \alpha_2, \alpha_3) & \times \mathbf{0} & \times \mathbf{0} & \times \tau_3^{\gamma n}(n_1^{(1)}, n_2^{(1)}, n_3^{(1)}) \\
\mathcal{D}_2 = \mathbf{0} & \times \mathbf{0} & \times \tau_3^{\bar{\gamma} n/4}(\alpha_1, \alpha_2, \alpha_3) & \times \tau_3^{\bar{\gamma} n/4}(\alpha_1, \alpha_2, \alpha_3) & \times \tau_3^{\gamma n}(n_1^{(1)}, n_2^{(1)}, n_3^{(1)})
\end{array}$$

As in the previous instantiations, we first determine the middle-layer parameters $n_i^{(1)}$, followed by the lower-layer parameters $n_i^{(2)}$.

The possible middle-layer decompositions of the values $\{-3,-2,-1,0,1,2,3\}$ using vectors over $\{0,\pm1,\pm2,\pm3\}$ are illustrated as follows:

	-3	-2	-1	0	1	2	3
$-3:$	$\underbrace{-3+0}_{th^{(0)}}$	$\underbrace{-2-1}_{\frac{n_3^{(0)}}{2}-th^{(0)}}$	$\underbrace{-1-2}_{\frac{n_3^{(0)}}{2}-th^{(0)}}$	$\underbrace{0-3}_{th^{(0)}}$			
$-2:$	$\underbrace{-3+1}_{t_1^{(0)}}$	$\underbrace{-2+0}_{t_0^{(0)}}$	$\underbrace{-1-1}_{n_2^{(0)}-2(t_0^{(0)}+t_1^{(0)})}$	$\underbrace{0-2}_{t_0^{(0)}}$	$\underbrace{1-3}_{t_1^{(0)}}$		
$-1:$	$\underbrace{-3+2}_{o_2^{(0)}}$	$\underbrace{-2+1}_{o_1^{(0)}}$	$\underbrace{-1+0}_{\frac{n_1^{(0)}}{2}-o_1^{(0)}-o_2^{(0)}}$	$\underbrace{0-1}_{\frac{n_1^{(0)}}{2}-o_1^{(0)}-o_2^{(0)}}$	$\underbrace{1-2}_{o_1^{(0)}}$	$\underbrace{2-3}_{o_2^{(0)}}$	
$0:$	$\underbrace{-3+3}_{z_3^{(0)}}$	$\underbrace{-2+2}_{z_2^{(0)}}$	$\underbrace{-1+1}_{z_1^{(0)}}$	$\underbrace{0+0}_{n_0^{(0)}-2(z_1^{(0)}+z_2^{(0)}+z_3^{(0)})}$	$\underbrace{1-1}_{z_1^{(0)}}$	$\underbrace{2-2}_{z_2^{(0)}}$	$\underbrace{3-3}_{z_3^{(0)}}$
$1:$		$\underbrace{-2+3}_{o_2^{(0)}}$	$\underbrace{-1+2}_{o_1^{(0)}}$	$\underbrace{0+1}_{\frac{n_1^{(0)}}{2}-o_1^{(0)}-o_2^{(0)}}$	$\underbrace{1+0}_{\frac{n_1^{(0)}}{2}-o_1^{(0)}-o_2^{(0)}}$	$\underbrace{2-1}_{o_1^{(0)}}$	$\underbrace{3-2}_{o_2^{(0)}}$
$2:$			$\underbrace{-1+3}_{t_1^{(0)}}$	$\underbrace{0+2}_{t_0^{(0)}}$	$\underbrace{1+1}_{n_2^{(0)}-2(t_1^{(0)}+t_0^{(0)})}$	$\underbrace{2+0}_{t_0^{(0)}}$	$\underbrace{3-1}_{t_1^{(0)}}$
$3:$				$\underbrace{0+3}_{th^{(0)}}$	$\underbrace{1+2}_{\frac{n_3^{(0)}}{2}-th^{(0)}}$	$\underbrace{2+1}_{\frac{n_3^{(0)}}{2}-th^{(0)}}$	$\underbrace{3+0}_{th^{(0)}}$

All introduced parameters are treated as optimization variables. By summing the corresponding contributions, we obtain the numbers of coordinates equal to $0,\pm1,\pm2,\pm3$ in the middle-layer vectors:

$$
\begin{aligned}
n_0^{(1)} &= n_1^{(0)} + n_0^{(0)} - 2(z_1^{(0)} + z_2^{(0)} + z_3^{(0)} + o_1^{(0)} + o_2^{(0)} - t_0^{(0)} - th^{(0)}),\\
n_1^{(1)} &= \frac{n_3^{(0)}}{2} - th^{(0)} + n_2^{(0)} - 2(t_0^{(0)} + t_1^{(0)}) + \frac{n_1^{(0)}}{2} - o_2^{(0)} + z_1^{(0)} + t_1^{(0)},\\
n_2^{(1)} &= \frac{n_3^{(0)}}{2} - th^{(0)} + t_0^{(0)} + o_1^{(0)} + z_2^{(0)} + o_2^{(0)},\\
n_3^{(1)} &= th^{(0)} + t_1^{(0)} + o_2^{(0)} + z_3^{(0)}.
\end{aligned}
$$

The total number of middle-layer representations $(\mathbf{a}_1,\mathbf{a}_2)$ of the secret vector is

$$
\begin{aligned}
R_1 = &\binom{n_0^{(0)}}{z_3^{(0)}, z_2^{(0)}, z_1^{(0)}, z_3^{(0)}, z_2^{(0)}, z_1^{(0)}, .}
\binom{n_1^{(0)}}{o_1^{(0)}, o_2^{(0)}, o_1^{(0)}, o_2^{(0)}, \frac{n_1^{(0)}}{2} - o_1^{(0)} - o_2^{(0)}, .}^2\\
&\binom{n_2^{(0)}}{t_1^{(0)}, t_0^{(0)}, t_1^{(0)}, t_0^{(0)}, .}^2
\binom{n_3^{(0)}}{th^{(0)}, th^{(0)}, \frac{n_3^{(0)}}{2} - th^{(0)}, .}^2.
\end{aligned}
$$

Similarly, the possible lower-layer decompositions are illustrated below:

	-3	-2	-1	0	1	2	3
$-3:$	$\underbrace{-3+0}_{th^{(1)}}$	$\underbrace{-2-1}_{\frac{n_3^{(1)}}{2}-th^{(1)}}$	$\underbrace{-1-2}_{\frac{n_3^{(1)}}{2}-th^{(1)}}$	$\underbrace{0-3}_{th^{(1)}}$			
$-2:$	$\underbrace{-3+1}_{t_1^{(1)}}$	$\underbrace{-2+0}_{t_0^{(1)}}$	$\underbrace{-1-1}_{n_2^{(1)}-2(t_0^{(1)}+t_1^{(1)})}$	$\underbrace{0-2}_{t_0^{(1)}}$	$\underbrace{1-3}_{t_1^{(1)}}$		
$-1:$	$\underbrace{-3+2}_{o_2^{(1)}}$	$\underbrace{-2+1}_{o_1^{(1)}}$	$\underbrace{-1+0}_{\frac{n_1^{(1)}}{2}-o_1^{(1)}-o_2^{(1)}}$	$\underbrace{0-1}_{\frac{n_1^{(1)}}{2}-o_1^{(1)}-o_2^{(1)}}$	$\underbrace{1-2}_{o_1^{(1)}}$	$\underbrace{2-3}_{o_2^{(1)}}$	
$0:$	$\underbrace{-3+3}_{z_3^{(1)}}$	$\underbrace{-2+2}_{z_2^{(1)}}$	$\underbrace{-1+1}_{z_1^{(1)}}$	$\underbrace{0+0}_{n_0^{(1)}-2(z_1^{(1)}+z_2^{(1)}+z_3^{(1)})}$	$\underbrace{1-1}_{z_1^{(1)}}$	$\underbrace{2-2}_{z_2^{(1)}}$	$\underbrace{3-3}_{z_3^{(1)}}$
$1:$		$\underbrace{-2+3}_{o_2^{(1)}}$	$\underbrace{-1+2}_{o_1^{(1)}}$	$\underbrace{0+1}_{\frac{n_1^{(1)}}{2}-o_1^{(1)}-o_2^{(1)}}$	$\underbrace{1+0}_{\frac{n_1^{(1)}}{2}-o_1^{(1)}-o_2^{(1)}}$	$\underbrace{2-1}_{o_1^{(1)}}$	$\underbrace{3-2}_{o_2^{(1)}}$
$2:$			$\underbrace{-1+3}_{t_1^{(1)}}$	$\underbrace{0+2}_{t_0^{(1)}}$	$\underbrace{1+1}_{n_2^{(1)}-2(t_1^{(1)}+t_0^{(1)})}$	$\underbrace{2+0}_{t_0^{(1)}}$	$\underbrace{3-1}_{t_1^{(1)}}$
$3:$				$\underbrace{0+3}_{th^{(1)}}$	$\underbrace{1+2}_{\frac{n_3^{(1)}}{2}-th^{(1)}}$	$\underbrace{2+1}_{\frac{n_3^{(1)}}{2}-th^{(1)}}$	$\underbrace{3+0}_{th^{(1)}}$

Again, all introduced parameters are optimization variables. The corresponding lower-layer coordinate counts are

$$
\begin{aligned}
n_0^{(2)} &= n_1^{(1)} + n_0^{(1)} - 2(z_1^{(1)} + z_2^{(1)} + z_3^{(1)} + o_1^{(1)} + o_2^{(1)} - t_0^{(1)} - th^{(1)}),\\
n_1^{(2)} &= \frac{n_3^{(1)}}{2} - th^{(1)} + n_2^{(1)} - 2(t_0^{(1)} + t_1^{(1)}) + \frac{n_1^{(1)}}{2} - o_2^{(1)} + z_1^{(1)} + t_1^{(1)},\\
n_2^{(2)} &= \frac{n_3^{(1)}}{2} - th^{(1)} + t_0^{(1)} + o_1^{(1)} + z_2^{(1)} + o_2^{(1)},\\
n_3^{(2)} &= th^{(1)} + t_1^{(1)} + o_2^{(1)} + z_3^{(1)}.
\end{aligned}
$$

and the total number of lower-layer representations of the vectors $\mathbf{a}_1$ and $\mathbf{a}_2$ is

$$
\begin{aligned}
R_2 = &\binom{n_0^{(1)}}{z_3^{(1)}, z_2^{(1)}, z_1^{(1)}, z_3^{(1)}, z_2^{(1)}, z_1^{(1)}, .}\binom{n_1^{(1)}}{o_1^{(1)}, o_2^{(1)}, o_1^{(1)}, o_2^{(1)}, \frac{n_1^{(1)}}{2} - o_1^{(1)} - o_2^{(1)}, .}^2\\
&\binom{n_2^{(1)}}{t_1^{(1)}, t_0^{(1)}, t_1^{(1)}, t_0^{(1)}, .}^2\binom{n_3^{(1)}}{th^{(1)}, th^{(1)}, \frac{n_3^{(1)}}{2} - th^{(1)}, .}^2.
\end{aligned}
$$

Finally, the original (given) secret domain is $\tau_3^n(w_1, w_2, w_3)$. Substituting all corresponding quantities into the complexity estimate (14) yields

$$
T_{\text{Improved-Nested-3}^\star} = \left(\frac{|\mathcal{D}|^{(1/2+3\lambda)}\,|\tau_3^n(w_1, w_2, w_3)|}{R_1 R_2^2}\right)^{1+o(1)}.
$$

C Experimental Validation

We performed a toy-scale experiment to validate the heuristic assumptions underlying the analysis of the Improved-Nested-2* framework. The goal of these experiments is not to benchmark the asymptotic attack complexity, but rather to verify that the projected LWE mappings exhibit the random-function behavior assumed throughout the analysis. In particular, we validate the projected uniqueness heuristic, the Layer-1 collision estimates, the middle-layer event probability $E_{\mathbf{r}}$ and the lower-layer representation heuristics related to the event $E_{\tilde{\mathbf{s}}}$.

We consider the standard Nested-2* decomposition

$$\mathbf{s} = \mathbf{s}_1 + \mathbf{s}_2 + \mathbf{s}_3 + \mathbf{s}_4, \qquad \mathbf{a}_1 = \mathbf{s}_1 + \mathbf{s}_2, \qquad \mathbf{a}_2 = \mathbf{s}_3 + \mathbf{s}_4,$$

with

$$\mathbf{s} \in \mathcal{D}, \qquad \mathbf{s}_i \in \mathcal{T}_i, \qquad \mathbf{a}_i \in \mathcal{D}_i.$$

The experiment uses the parameters

$$n = 24, \qquad q = 149, \qquad \ell = 2, \qquad q^\ell = 22{,}201.$$

The lower-level domains are

$$\begin{aligned}
\mathcal{T}_1 &= \tau^4(1) \times \mathbf{0} \times \mathbf{0} \times \mathbf{0} \times \tau_2^8(1,1),\\
\mathcal{T}_2 &= \mathbf{0} \times \tau^4(1) \times \mathbf{0} \times \mathbf{0} \times \tau_2^8(1,1),\\
\mathcal{T}_3 &= \mathbf{0} \times \mathbf{0} \times \tau^4(1) \times \mathbf{0} \times \tau_2^8(1,1),\\
\mathcal{T}_4 &= \mathbf{0} \times \mathbf{0} \times \mathbf{0} \times \tau^4(1) \times \tau_2^8(1,1).
\end{aligned}$$

Accordingly, the middle-layer domains are

$$\begin{aligned}
\mathcal{D}_1 &= \tau^4(1) \times \tau^4(1) \times \mathbf{0} \times \mathbf{0} \times \tau_2^8(2,1),\\
\mathcal{D}_2 &= \mathbf{0} \times \mathbf{0} \times \tau^4(1) \times \tau^4(1) \times \tau_2^8(2,1),
\end{aligned}$$

and the secret domain is

$$\mathcal{D} = \left(\tau^4(1)\right)^4 \times \tau^8(2).$$

The corresponding domain sizes are

$$|\mathcal{D}| = 8{,}709{,}120, \qquad |\mathcal{T}_i| = 20{,}160, \qquad |\mathcal{D}_i| = 725{,}760.$$

Hence, $\frac{q^\ell}{|\mathcal{T}_i|} = 1.101240$, showing that the Pollard-Rho balancing condition is nearly exact. We also have

$$q^\ell = |\mathcal{D}|^{1/2+\lambda_{\text{eff}}}, \qquad \lambda_{\text{eff}} = 0.126281,$$

placing the experiment in the nontrivial regime $\lambda > 0$.

Exact enumeration gives

$$R_2(\mathbf{a}_i) = 32, \qquad R_1(\mathbf{s}) = 456,$$

which satisfy

$$R_1 < q^\ell, \qquad R_2(\mathbf{a}_i)^2 < q^\ell.$$

Table 4. Experimental validation of the heuristic assumptions underlying the Improved-Nested-$2^\star$ framework over 10,000 random matrices.

Experiment	Empirical	Theory
Average false projected solutions in $\pi_{2\ell}(\mathcal{D})$	0.0160	0.0177
Unique projected solution fraction	0.9844	0.9825
Layer-1 collisions — projected LWE maps	18306.29	18306.63
Layer-1 collisions — random functions	18307.68	18306.63
Middle-layer event probability $\mathbb{P}[E_{\mathbf{r}}]$	0.0203	0.0203
Total distinguished collisions	18340.48	18338.63
Samples required for distinguished collision	573.55	573.08

The experiments were performed over 10,000 independently sampled random matrices $\mathbf{A} \in \mathbb{Z}_q^{n\times n}$. For each matrix, all measurements were averaged.

The theoretical values in Table 4 are obtained from the same random-function heuristics used throughout the complexity analysis. Under the projection heuristic, an incorrect candidate $\mathbf{x} \in \mathcal{D}\backslash\{\mathbf{s}\}$ survives the projection with probability $1/q^{2\ell}$, leading to the estimate

$$\frac{|\mathcal{D}|-1}{q^{2\ell}}$$

for the expected number of false projected solutions. Similarly, the Layer-1 collision estimate is given by

$$\frac{|\mathcal{T}_1||\mathcal{T}_2|}{q^\ell}.$$

The middle-layer event probability is predicted by

$$1-\left(1-\frac{1}{q^\ell}\right)^{R_1},$$

which corresponds to the event $E_{\mathbf{r}}$. Finally,

$$R_2+\frac{|\mathcal{T}_1||\mathcal{T}_2|-R_2}{q^\ell}$$

gives the expected number of distinguished collisions, while

$$\frac{R_2+\left(|\mathcal{T}_1||\mathcal{T}_2|-R_2\right)/q^\ell}{R_2}$$

estimates the average number of samples required to recover a planted representation.

The empirical measurements are in very close agreement with the theoretical predictions. In particular, the projected LWE mappings behave statistically like uniformly random functions over the structured domains considered in the experiment. The observed collision counts closely match the estimates, the probability of the middle-layer event $E_{\mathbf{r}}$ agrees with the predicted value, and the distinguished collision statistics follow the expected lower-layer representation

model. Overall, the experiments provide strong empirical support for the heuristic assumptions underlying the IMPROVED-NESTED-2* framework.