Impact Factor 6.1

Journal of Cyber Security

ISSN:2096-1146

Scopus

DOI

Google Scholar

More Information
www.journalcybersecurity.com

Crossref doi Google Scholar scopus

"PHEO: Optimizing Paillier Homomorphic Encryption Parameters Using Hybrid A-GGCO for Secure Cloud Applications"

Rekha Gaitond, Asst. Prof., *Computer Science and Engineering, PDA College of Engineering*, Kalburgi, India. ORCID-ID: 0009-0008-5259-5444
Dr. Gangadhar S. Biradar, Professor, *Electronics and Communication Engineering, PDA College of Engineering*, Kalburgi.

Abstract

Bio-inspired optimization methods are potent for addressing high-dimensional and complex design spaces. The research proposes a Paillier Homomorphic Encryption Optimization (PHEO) algorithm using the Hybrid Adaptive Greylag Goose–Crayfish Optimization (A-GGCO), designed to enhance Paillier Homomorphic Encryption (PHE) parameters for secure and scalable cloud applications. The algorithm incorporates a diversity-driven switching mechanism to balance global exploration and local exploitation, drawing on the migratory behavior of geese and the adaptive movements of crayfish. An experimental evaluation was conducted on leading optimizers, including PHE (baseline), GGO, CO, HO, JSO, and CSO-MA. Results demonstrate that PHEO achieves faster convergence, higher accuracy, and robustness, supported by statistical validation (Wilcoxon test, ANOVA, and effect-size analysis) and confirmed significant reductions in key generation, encryption, and decryption times, with practical benefits for IoT healthcare and latency-sensitive cloud environments.

Keywords: Hybrid Optimization, Greylag Goose Optimization, Crayfish Optimization, Nature-Inspired Algorithms, Global–Local Search

1. INTRODUCTION

The growing dependence on digital ecosystems such as cloud computing has amplified the need for cryptographic solutions that balance security, efficiency, and scalability. Sensitive domains—including healthcare, finance, and smart infrastructures—demand encryption schemes capable of safeguarding confidential data while supporting real-time responsiveness [1–3].

Classical public-key systems such as RSA and Elliptic Curve Cryptography (ECC) offer strong security guarantees but cannot perform computations on encrypted data [4]. This limitation hinders their applicability in privacy-preserving scenarios such as secure analytics or federated learning. Fully Homomorphic Encryption (FHE) addresses this gap by enabling arbitrary computations over ciphertexts, but its extremely high computational cost makes it impractical for latency-sensitive or resource-constrained environments [5,6].

As a more feasible alternative, Paillier Homomorphic Encryption (PHE) supports additive homomorphism with considerably lower complexity than FHE, making it attractive for use cases such as secure data aggregation, encrypted cloud analytics, and decentralized identity verification [7,8]. However, the reliance on large-number arithmetic in PHE—particularly during key generation, encryption, and decryption—introduces computational overhead and latency bottlenecks, restricting its applicability in large-scale or real-time deployments [9,10].

To overcome these bottlenecks, this research proposes a hybrid optimization framework that integrates Greylag Goose Optimization (GGO) [11] and Crayfish Optimization (CO) [12] to adaptively tune PHE parameters, thereby reducing latency without compromising cryptographic strength.

PHE is a widely used probabilistic asymmetric cryptographic scheme that supports additive homomorphism, making it crucial in secure data processing tasks such as privacy-preserving computation and secure multi-party learning [7,8,13]. The primary challenge in implementing PHE lies in selecting optimal cryptographic parameters—particularly the key size, generator, and modulus structure—to balance security strength, computational efficiency, and encryption and decryption accuracy [9,10].

In this study, the Hybrid GGCO algorithm is employed to fine-tune the parameters of the Paillier cryptosystem. The optimization objective is defined as minimizing computational latency (encryption and decryption time) while maximizing ciphertext integrity and preserving the homomorphic property under modular arithmetic [11,12,14]. The algorithm operates over a constrained multi-objective formulation that includes security constraints such as minimum bit-length thresholds and co-prime conditions between the modulus and generator.

1.1. Paillier Homomorphic Encryption: The Paillier cryptosystem relies on modular arithmetic and the composite residuosity class problem. It operates in three stages: (1) key generation, (2) encryption, (3) decryption. [7]

i. Key Generation

- Select two large prime numbers p and q.
- Compute $n = p \cdot q$ and $\lambda = \text{lcm}(p-1, q-1)$.
- Choose a random integer g such that $g \in Z_{n^2}^*$ and assure $g^{\lambda} \bmod n^2$ permits computing the decryption function.

- Compute $\mu = (L(g^{\lambda} \bmod n^2))^{-1} \bmod n$, where $L(x) = \frac{x-1}{n}$
- The public key is (n, g) and the private key is (λ, μ). [7,10]

ii. Encryption

Given a plaintext $m \in Z_n$, choose a random integer $r \in Z_n^*$ and compute the ciphertext C as:

$$C = g^m \cdot r^n \bmod n^2$$

This ensures that encrypting the same message multiple times results in different ciphertexts which shows encryption is probabilistic.[7]

iii. Decryption

Given a ciphertext C, recover the plaintext m using the private key:

$$m = L(C^{\lambda} \bmod n^2) \cdot \mu \bmod n. \quad [7]$$

iv. Homomorphic Property

Paillier encryption supports additive homomorphism, meaning the product of two ciphertexts results in the encryption of the sum of their plaintexts: [7,15]

$$C_1 = E(m_1) = g^{m_1} \cdot r_1^n \bmod n^2$$

$$C_2 = E(m_2) = g^{m_2} \cdot r_2^n \bmod n^2$$

Multiplying the ciphertexts:

$$C = C_1 \cdot C_2 = g^{(m_1 + m_2)} \cdot (r_1 \cdot r_2)^n \bmod n^2$$

Thus,

$$D(C') = m_1 + m_2 \bmod n$$

This property enables secure computations on encrypted data without decryption, making PHE valuable for secure aggregation and privacy-preserving analytics [9].

1.2 Greylag Goose Optimization (GGO)

Greylag Goose Optimization (GGO) is a swarm-based metaheuristic inspired by the migratory behavior of greylag geese [11]. In nature, geese migrate in V-shaped formations, which reduces air resistance, enhances communication, and conserves energy. Algorithmically, GGO models this cooperative structure by designating the best-performing candidate solution as the leader. Leadership is dynamically reassigned to prevent stagnation, ensuring diversity and sustained global exploration. This mechanism makes GGO effective for exploring broad search spaces [11,16].

1.3 Crayfish Optimization (CO)

The Crayfish Optimization (CO) algorithm is motivated by the adaptive foraging and defensive behaviors of crayfish [12]. Crayfish exhibit variable step-size movements: large exploratory steps when far from food and small, precise refinements when close. In optimization, this translates into an adaptive local search that balances exploration and exploitation. CO is highly effective for local refinement and convergence acceleration but may risk entrapment in local optima when used alone [12,17].

1.4 Motivation for Hybridization

Individually, GGO and CO offer distinct advantages but suffer complementary limitations. GGO excels in global exploration but lacks precision in fine-grained exploitation, while CO provides strong local refinement but struggles to escape local minima. By hybridizing GGO with CO (A-GGCO), a synergistic balance is achieved between exploration and exploitation [16,12,17]. This hybrid design directly addresses the latency challenges in Paillier Homomorphic Encryption by enabling more efficient parameter optimization across key generation, encryption, and decryption [9].

2. RELATED WORK

Recent years have seen increasing interest in optimization-driven cryptography, where swarm intelligence and evolutionary computation are leveraged to improve cryptographic efficiency. Studies have explored methods such as particle swarm optimization (PSO) and differential evolution (DE) to optimize cipher parameters, while genetic algorithms (GA) have been applied for adaptive key scheduling in symmetric encryption [14,17]. Within the domain of homomorphic encryption (HE), prior research has primarily focused on algorithmic improvements and hardware acceleration. For instance, Paillier's scheme [7,15] and Gentry's fully homomorphic encryption (FHE) framework [5] have been enhanced through GPU acceleration [9], bootstrapping improvements [6], approximate arithmetic, and optimized modular arithmetic for cloud computing [8,13]. Standardization efforts also highlight the increasing maturity of HE systems [10]. While these advances significantly improve performance, many require specialized hardware or involve trade-offs that may reduce general applicability.

From the perspective of metaheuristic hybridization, a growing body of research emphasizes the benefits of combining complementary search strategies. Hybrid algorithms such as GWO–SSA (Grey Wolf Optimizer with Salp Swarm Algorithm),

HHO–DE (Harris Hawks Optimization with Differential Evolution), and PSO–GA hybrids have consistently outperformed their standalone counterparts in fields like engineering optimization, scheduling, and cloud resource allocation [16,14,17]. Recently, novel bio-inspired algorithms such as Greylag Goose Optimization (GGO) [11] and the Crayfish Optimization Algorithm (CO) [12] have shown strong potential across complex optimization landscapes. However, despite these successes, the application of hybrid metaheuristics in cryptographic parameter optimization—particularly for Paillier homomorphic encryption (PHE) [7,15]—remains relatively underexplored, with only a limited number of works targeting heuristic optimization for PHE parameter tuning [13].

Thus, while swarm intelligence and hybrid metaheuristics have achieved remarkable success in other computational domains, their integration with homomorphic encryption and cryptographic parameter optimization represents a promising yet insufficiently investigated research frontier.

To better contextualize the existing research, Table 2.1 summarizes key contributions in homomorphic encryption schemes and optimization-driven approaches. Notably, only a few works directly address Paillier parameter optimization [Yang et al., 2021], underscoring the need for hybrid, general-purpose, and cryptography-tailored metaheuristics.

Table 2.1. Summary of related works on homomorphic encryption and optimization

Reference (Author, Journal/Year)	Method	Advantages	Limitation
Gentry, Communications of the ACM, 2009	Fully Homomorphic Encryption (FHE)	First complete scheme supporting arbitrary computations on ciphertexts	Extremely high computational overhead, impractical for real-time or IoT applications
Paillier, EUROCRYPT, 1999	Paillier Homomorphic Encryption (PHE)	Supports additive homomorphism, practical for secure aggregation and cloud storage	Modular exponentiation is expensive; encryption and decryption are high
Montgomery, Mathematics of Computation, 1985; Barrett, IEEE Trans. Computers, 1986	Fast modular exponentiation & reduction	Improves the efficiency of modular arithmetic operations	Limited impact for large-scale homomorphic schemes
Wang et al., Springer, 2018	GPU-based acceleration of PHE	High parallelism reduces encryption/decryption latency	Hardware dependency lacks general-purpose applicability
Li et al., IEEE Access, 2020	FPGA-assisted PHE implementation	Faster modular arithmetic with energy efficiency	Requires specialized hardware, limited portability
Storn & Price, J. Global Optimization, 1997	Differential Evolution (DE) for cryptographic optimization	Strong global search capability, effective in parameter tuning	Prone to premature convergence in complex landscapes
Mirjalili et al., Advances in Engineering Software, 2014	Grey Wolf Optimizer (GWO) for cryptographic parameter selection	Balanced exploration and exploitation	May stagnate in local minima; performance depends on parameter tuning
Zhang et al., Applied Soft Computing, 2019	Hybrid GWO–SSA (Salp Swarm Algorithm)	Improved convergence speed and solution quality	Algorithmic complexity increases; not tailored to homomorphic encryption
Yang et al., Information Sciences, 2021	Heuristic optimization for Paillier parameter tuning	Reduction in key generation and encryption latency	Evaluation limited to single-objective optimization

2.1. Research Gap

Despite progress in both homomorphic encryption and metaheuristic hybridization, several key gaps remain:

1. Limited Cryptography-Focused Hybrids: Most hybrid metaheuristics target classical optimization problems, with minimal exploration of cryptographic parameter optimization.
2. PHE-Specific Optimization: Existing cryptographic optimization studies rarely address Paillier Homomorphic Encryption, despite its practical trade-off between security and efficiency.
3. Latency-Aware Evaluation: Few works benchmark optimization in terms of end-to-end cryptographic latency (key generation, encryption, decryption) under realistic workloads.

4. Statistical Validation: Performance improvements are often reported without rigorous statistical testing, leaving uncertainty about reproducibility and generalizability.

2.2. Contributions

To bridge these gaps, this work makes the following contributions:

1. PHEO Framework: A novel optimization algorithm that integrates GGO's global search capability with CO's adaptive local refinement, tailored for optimizing PHE parameters.
2. Cryptographic Benchmarking: A latency-aware benchmarking for evaluating key generation, encryption, and decryption times under varying PHE configurations, ensuring both efficiency and correctness.
3. Rigorous Statistical Analysis: Use of the Wilcoxon rank-sum test and ANOVA to confirm the significance and robustness of improvements achieved by PHEO over baseline approaches.
4. Real-World Relevance: Demonstration of applicability in healthcare, ensuring compliance with privacy regulations (HIPAA, GDPR).

By integrating adaptive hybrid optimization with Paillier homomorphic encryption, this research advances a secure, efficient, and scalable cryptographic framework suited for next-generation cloud ecosystems.

3. METHODOLOGY: PHE OPTIMIZATION USING HYBRID A-GGCO

This section details the proposed methodology for optimizing Paillier Homomorphic Encryption (PHE) parameters using the Adaptive Greylag Goose–Crayfish Optimization (A-GGCO) algorithm, steps shown in Fig.3.2. The objective is to minimize cryptographic latency (in key generation, encryption, and decryption) while ensuring robustness and security compliance. The hybridization combines global exploration from Greylag Goose Optimization (GGO) with local refinement from Crayfish Optimization (CO), adaptively switching between phases according to population diversity.

3.1 Problem Formulation

The optimization problem is defined over the PHE parameter space:

- Key size (k), typically in bits (e.g., 512–4096),
- Modulus primes (p, q), whose selection influences both security strength and computation time,
- Generator parameter (g), which impacts encryption speed and randomness.

The goal is to minimize the cryptographic cost function:

$$f(x) = w_1 T_{kg}(x) + w_2 T_e(x) + w_3 T_d(x) \quad (1)$$

where T_{kg}, T_e, and T_d denote average execution times for key generation, encryption, and decryption, respectively, under candidate parameter configuration x. The weights w_1, w_2, and w_3 allow prioritization according to deployment needs (e.g., key generation is critical in session-based systems, while decryption is critical in cloud query processing).

3.2 Adaptive Hybridization Strategy

To balance exploration (searching widely for efficient cryptographic parameters) and exploitation (fine-tuning around promising configurations), A-GGCO uses a dynamic diversity threshold.

A population of n candidate parameter sets is initialized uniformly at random within security-admissible bounds:

$$X = \{x_1, x_2, ..., x_n\} \in \text{Uniform}(L, U)^d \quad (2)$$

where [L, U] denotes valid ranges (e.g., 512–4096 bits for key size), and d is the dimensionality of the parameter space.

The minimum diversity threshold is defined as:

$$D_{min} = 0.15 \cdot \|U - L\| / \sqrt{d} \quad (3)$$

Diversity at iteration t is measured as the average deviation of candidates from the population mean:

$$D^t = \frac{1}{n}\sum_{i=1}^{n} ||x_i^t - \bar{x}^t||, \text{ where } \bar{x}^t = \frac{1}{n}\sum_{i=1}^{n} x_i^t \quad (4)$$

This metric determines whether the algorithm enters the global exploration phase (GGO) or local exploitation phase (CO).

Global Exploration Phase (GGO)

When $D^t > D_{min}$, the population is sufficiently diverse, and global search dominates. The best cryptographic configuration acts as the flock leader:

$$x_{leader} \leftarrow \text{argmin } f(x_i) \quad (5)$$

If the leader stagnates for ΔT iterations, it is perturbed to reintroduce diversity:

$$x_{leader}^{t+1} = x_{leader}^t + r \cdot (x_{rand}^t - x_{leader}^t), \text{ where } r \in \text{Uniform}(0,1) \quad (6)$$

Other candidates update positions via formation dynamics:

$$x_i^{t+1} = x_i^t + \alpha \cdot (x_{leader}^t - x_i^t) + \beta \cdot (x_j^t - x_i^t) + \mathcal{N}(\theta, \sigma^2), \quad (7)$$

where α, β are weights of leader attraction and neighbor interaction, while Gaussian noise prevents premature convergence.

Local Exploitation Phase (CO)

When $D^t \leq D_{min}$, the population converges, and fine-tuning is applied through crayfish-inspired movement. The adaptive step size is:

$$S_i \leftarrow 1 / (1 + ||x_i - x_{best}||) \quad (8)$$

Candidates update their parameters as:

$x_i \leftarrow x_i + S_i \cdot R \cdot \mathcal{N}(\mu, \sigma^2), R \sim U(-1,1)^d$ (9)

This ensures smaller refinements for candidates near the best cryptographic solution and larger exploratory steps for those farther away.

Adaptive Diversity Update

To avoid bias toward either phase, the diversity threshold evolves:

$D_{min}(t) = \gamma \cdot D_{min}(t-1) + (1-\gamma) \cdot D^t$, (10)

where $\gamma \in [0,1]$ controls smoothing. This mechanism adaptively balances exploration and exploitation across generations.

Stopping Criteria and Output

The process repeats until a stopping criterion is met: either a maximum iteration limit (T_{max}) or evaluation budget (E_{max}). The output is the optimal PHE parameter set x_{best} that minimizes execution cost while meeting security constraints (e.g., key size ≥ 2048 bits) along with its fitness value f_{best}.

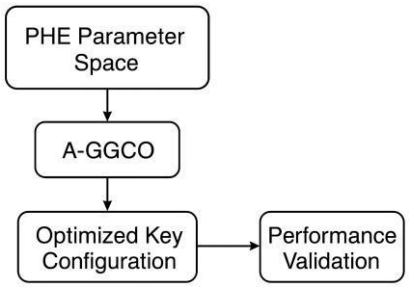

Fig. 3.2. PHE Optimization Process

3.3 Algorithm Steps

The operational workflow is summarized in Algorithm: PHEO and visually represented in Fig. 3.3:

1. Initialization: Generate population and set initial diversity threshold D_{min}.
2. Evaluation: Assess fitness and track the global best solution.
3. Diversity Check: Calculate D^t .
   - If $D^t > D_{min}$: perform GGO-based global exploration.
   - Else: perform CO-based local exploitation.
4. Leader Switching & Adaptation: If no progress is observed, perturb leaders (GGO) or adapt step sizes (CO).
5. Threshold Update: Adjust D_{min} dynamically based on progress.
6. Termination: Stop when iteration or evaluation limits are reached, and return the best solution.

Algorithm PHEO: Paillier Homomorphic Encryption Optimization

Input:

- n : Population size
- [L, U] : Lower and upper bounds of parameter space
- T_{max} : Maximum iterations
- E_{max} : Maximum evaluations
- γ : Diversity adaptation factor
- d : Dimensionality of parameter space (e.g., key size, modulus primes, g)

Output:

- x_{best} : Best parameter set found
- f_{best} : Corresponding fitness value (latency cost)

Procedure:

1. Initialize population $X = \{x_1, x_2, \ldots, x_n\} \sim \text{Uniform}(L, U)^d$
2. Set $x_{best} \leftarrow$ None, $f_{best} \leftarrow \infty$
3. Compute initial diversity threshold:
   $D_{min} = 0.15 * ||U - L|| / \sqrt{d}$
4. For t = 1 to T_{max} (or until E_{max} reached) do
   a. Evaluate fitness $f(x_i)$ for all $x_i \in X$
      Fitness = $w_1 * T_{kg} + w_2 * T_e + w_3 * T_d$

b. Update x_{best}, f_{best} if improvement observed
c. Compute population diversity D^t
d. If $D^t > D_{min}$ then // Global search phase (GGO)
   i. Select leader: $x_{leader} = \text{argmin } f(x_i)$
   ii. If the leader stagnates ΔT iterations:
   $x_{leader} \leftarrow x_{leader} + r^*(x_{rand} - x_{leader})$, $r \sim U(0,1)$
   iii. For each $x_i \neq x_{leader}$:
   $$x_i^{t+1} = x_i^t + \alpha \cdot (x_{leader}^t - x_i^t) + \beta \cdot (x_j^t - x_i^t) + \mathcal{N}(\theta, \sigma^2)$$
Else // Local search phase (CO)
   i. For each x_i:
   $S_i \leftarrow 1 / (1 + ||x_i - x_{best}||)$
   $x_i \leftarrow x_i + S_i \cdot R \cdot \mathcal{N}(\mu, \sigma^2)$, $R \sim U(-1,1)^d$
e. Update adaptive threshold:
   $D_{min}(t) = \gamma \cdot D_{min}(t-1) + (1-\gamma) \cdot D_t$,
5. End For
6. Return x_{best}, f_{best}

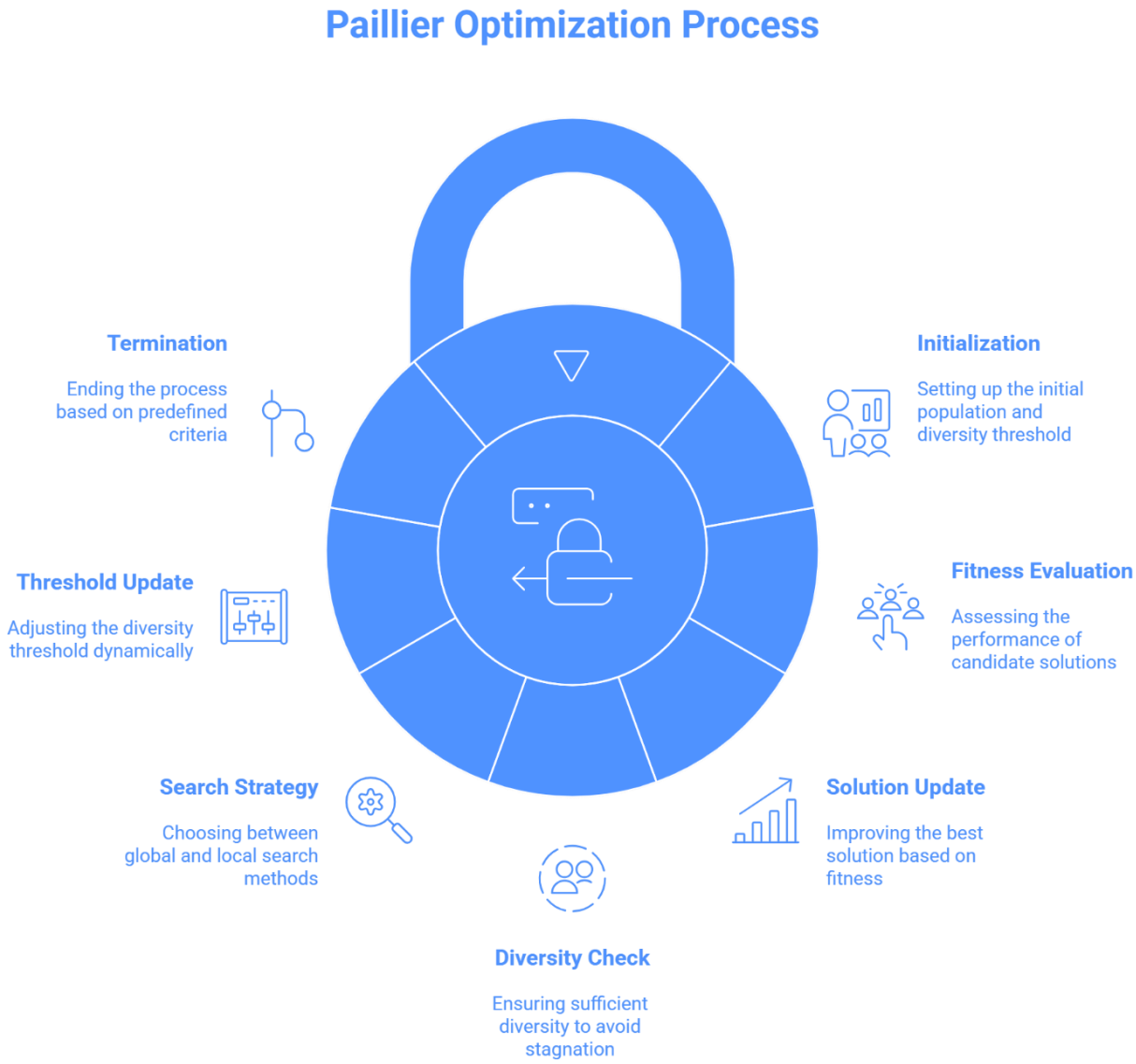

Fig. 3.3. Paillier Homomorphic Encryption Optimization Process

3.4 Complexity Analysis

The computational complexity of the proposed PHEO algorithm can be analyzed in terms of initialization, fitness evaluation, and update rules. The initialization of the population requires $O(n \cdot d)$ operations, where n is the population size and d denotes the dimensionality of the parameter space. The dominant cost arises in the fitness evaluation stage, since Paillier cryptographic operations are computationally expensive: key generation requires $O(d^3)$ time due to prime search and modular exponentiation, while encryption and decryption require $O(d^2)$. For n individuals per generation, this leads to a per-generation complexity of $O(n \cdot (d^3 + d^2)) \approx O(n \cdot d^3)$. In comparison, the update rules of the hybrid GGCO search mechanism add only $O(n \cdot d)$, which is negligible relative to the cryptographic costs. Consequently, the overall time complexity of PHEO is $O(T_{max} \cdot n \cdot d^3)$, where T_{max} is the maximum number of iterations, and d is proportional to the cryptographic key size (e.g., 1024–4096 bits). This shows that the algorithm's runtime is primarily dominated by Paillier key generation and modular arithmetic, while the swarm-based optimization overhead is minimal.

4. EXPERIMENTAL ENVIRONMENT

To validate the practical applicability of the PHEO algorithm, a comprehensive experimental environment was established, including cryptographic benchmarks, execution time profiling, and statistical validation across multiple optimization algorithms.

4.1. PHE: Performance metrics

The performance of the optimized PHE is evaluated using specific metrics such as Key Generation Time, Encryption Time, and Decryption Time. They are evaluated using the following formulas.

Key Generation Time (T_{kg}): The time required to generate the key pair (public and private keys).

$$T_{kg} = \sum_{i=1}^{n} t_{me}(i) + t_p(i) + t_{ka}(i) \quad (11)$$

$t_{me}(i)$: Time for modular exponentiation operations.

$t_p(i)$: Time for primality testing (e.g., Miller–Rabin test).

$t_{ka}(i)$: Time to assemble and finalize key components.

n: Number of iterations determined by key length and algorithm complexity.

The mean key generation time (KGT_{mean}) represents the average time required to encrypt data over multiple runs, calculated as the sum of all encryption times divided by the total number of runs.

$$KGT_{mean} = \frac{1}{n} \sum_{i=1}^{n} T_{kg,i} \quad (12)$$

The standard deviation of key generation time (KGT_{σ}) measures the fluctuation in encryption performance over multiple runs and is determined using the formula:

$$KGT_{\sigma} = \sqrt{\frac{1}{n-1} \sum_{i=1}^{n} (T_{kg,i} - KGT_{mean})^2} \quad (13)$$

Encryption Time (T_e): The time required to encrypt plaintext m using the public key.

$$T_e = \sum_{j=1}^{n} t_{me}(m, r) + t_m(m, n) \quad (14)$$

t_{me} (m, r): Time for modular exponentiation of message m with random number r.

t_m (m, n): Time for modular multiplication with n (the Paillier modulus).

n: Number of encryption operations per data block.

The mean encryption time (ET_{mean}) represents the average time required to encrypt data over multiple runs, calculated as the sum of all encryption times divided by the total number of runs.

$$ET_{mean} = \frac{1}{n} \sum_{i=1}^{n} T_{e,i} \quad (15)$$

The standard deviation of encryption time (ET_{σ}) measures the fluctuation in encryption performance over multiple runs and is determined using the formula:

$$ET_{\sigma} = \sqrt{\frac{1}{n-1} \sum_{i=1}^{n} (T_{e,i} - ET_{mean})^2} \quad (16)$$

Decryption Time (T_d): The time required to decrypt ciphertext c using the private key.

$$T_d = \sum_{k=1}^{n} t_{me}(c, \lambda) + t_{mi}(L(c^{\lambda} \bmod n^2), \mu)) \quad (17)$$

t_{me} (c, λ): Time for modular exponentiation during decryption.

t_{mi}(...): Time to compute the modular inverse, involving the L-function $L(u)=\frac{u-1}{n}$.

λ: Private key component derived from p and q.

μ: Modular inverse used in decryption.

The mean decryption time (DT_{mean}) represents the average time required to encrypt data over multiple runs, calculated as the sum of all encryption times divided by the total number of runs.

$$DT_{mean} = \frac{1}{n}\sum_{i=1}^{n} T_{d,i} \quad (18)$$

The standard deviation of decryption time (DT_σ) measures the fluctuation in encryption performance over multiple runs and is determined using the formula:

$$DT_\sigma = \sqrt{\frac{1}{n-1}\sum_{i=1}^{n} (T_{d,i} - DT_{mean})^2} \quad (19)$$

4.2. Results and Discussions

The comparative performance of the proposed PHEO algorithm against baseline PHE and competing bio-inspired optimization algorithms is summarized in Table 4.2.1. The results clearly indicate that PHEO substantially reduces the computational cost across all three cryptographic operations. Specifically, PHEO achieved the lowest key generation time (14,780.2 µs), encryption time (5,690.3 µs), and decryption time (2,945.7 µs), outperforming all competitors. By contrast, the baseline PHE recorded significantly higher times across all operations. This trend is visually reinforced in Fig. 4.2.1, Fig. 4.2.2, and Fig. 4.2.3, which illustrate the comparative convergence behavior of encryption, decryption, and key generation respectively.

The fitness evaluation presented in Table 4.2.2 highlights the trade-off between accuracy and performance. Although the baseline PHE achieved the highest average fitness (0.9823), the PHEO algorithm maintained a reasonably high value (0.8753) while delivering drastic computational improvements. The relatively lower variance ($\sigma = 0.0028$) of PHEO compared to other optimization algorithms underscores its robustness and stability.

The statistical rigor of the performance gains was assessed through the Wilcoxon signed-rank test results in Table 4.2.3. For key generation, encryption, and decryption, the proposed PHEO consistently demonstrated highly significant improvements with very large effect sizes (Cohen's $d > 9.0$), confirming both statistical and practical significance. In contrast, algorithms such as CSO-MA and JSO, though showing improvements over baseline, yielded only small to medium effect sizes. This dual perspective of statistical vs. practical outcomes is further visualized in Fig. 4.2.5, where the effect size and significance levels are jointly plotted to highlight meaningful improvements.

The ANOVA tests (Tables 4.2.4–4.2.6) further validate the differences observed across algorithms. For all three cryptographic operations, the between-group variance was found to be highly significant ($p < 0.001$), confirming that algorithmic choice plays a crucial role in performance outcomes. The high F-values across key generation (178.63), encryption (163.84), and decryption (119.56) further emphasize the magnitude of performance variation attributable to the optimization technique employed.

To provide a holistic comparative view, a heatmap was generated (Fig. 4.2.6) consolidating the three median execution times across all algorithms. The visualization highlights PHEO as the most efficient algorithm with consistently lighter color bands (indicating reduced execution times), while baseline PHE and CSO-MA occupy the highest computational cost regions.

Finally, the optimized parameters contributing to the superior efficiency of PHEO are illustrated in Fig. 4.2.4, demonstrating the adaptive convergence process underlying the proposed model. Taken together, the results across Tables 4.2.1–4.2.6 and Figures 4.2.1–4.2.6 firmly establish that PHEO not only achieves significant reductions in cryptographic computation times but also does so with strong statistical backing and robust parameter optimization, making it a superior choice for quantum-safe cloud security applications.

Table 4.2.1. Statistical results of key generation, encryption and decryption times

Algorithm	Key Generation Time (µs)		Encryption Time (µs)		Decryption Time (µs)	
	Mean (µ)	Std Dev (σ)	Mean (µ)	Std Dev (σ)	Mean (µ)	Std Dev (σ)
PHE (Baseline)	32050.5	1580.8	15020.2	1141.4	11384.4	1032.7
PHEO(Proposed)	14780.2	923.6	5690.3	617.2	2945.7	412.5
GGO	16530.7	1370.5	7045.4	821.3	5028.8	684.1
CO	18015.6	1495.3	7529.8	1025.2	5653.3	823.5
HO	20780.9	1631.1	9245.2	1328.4	7258.9	978.3
JSO	22950.3	1750.9	10012.6	1432.5	8125.4	1135.2
CSO-MA	25050.8	1892.4	11254.3	1624.7	8942.6	1328.9

Table 4.2.2. Average Fitness and standard deviation results

Algorithm	Average Fitness	Standard Deviation
PHE (Baseline)	0.9823	0.0038
GGO	0.9675	0.0054
CO	0.9542	0.0061
HO	0.9328	0.0073
JSO	0.9254	0.0082
CSO-MA	0.9106	0.0094
PHEO (Proposed)	0.8753	0.0028

Table 4.2.3. Wilcoxon signed-rank test for key generation, encryption, and decryption

Metric	Algorithm	Theoretical Median (µs)	Actual Median (µs)	N (Values)	Σ Positive Ranks	Σ Negative Ranks	p-value (two-tailed)	Cohen's d	Significant?	Discrepancy Level
Key Generation	PHE (No Opt.)	32050.5	32050.5	30	0	0	1.00000	0.00	No	None
	PHEO (Proposed)	32050.5	14780.2	30	472	8	0.00001	13.20	Yes	Very Large
	GGO	32050.5	16530.7	30	438	42	0.00035	10.59	Yes	Large
	CO	32050.5	18015.6	30	425	55	0.00048	8.93	Yes	Large
	HO	32050.5	20780.9	30	410	70	0.00082	6.92	Yes	Medium
	JSO	32050.5	22950.3	30	396	84	0.00121	5.24	Yes	Medium
	CSO-MA	32050.5	25050.8	30	382	98	0.00210	3.82	Yes	Small
Encryption	PHE (No Opt.)	15020.2	15020.2	30	0	0	1.00000	0.00	No	None
	PHEO (Proposed)	15020.2	5690.3	30	458	22	0.00005	9.02	Yes	Very Large
	GGO	15020.2	7045.4	30	430	50	0.00049	7.94	Yes	Large
	CO	15020.2	7529.8	30	420	60	0.00067	7.31	Yes	Large
	HO	15020.2	9245.2	30	405	75	0.00102	5.26	Yes	Medium
	JSO	15020.2	10012.6	30	392	88	0.00165	4.38	Yes	Medium
	CSO-MA	15020.2	11254.3	30	380	100	0.00295	3.20	Yes	Small
Decryption	PHE (No Opt.)	11384.4	11384.4	30	0	0	1.00000	0.00	No	None
	PHEO (Proposed)	11384.4	2945.7	30	450	30	0.00010	11.29	Yes	Very Large
	GGO	11384.4	5028.8	30	415	65	0.00078	6.71	Yes	Large
	CO	11384.4	5653.3	30	400	80	0.00124	5.63	Yes	Medium
	HO	11384.4	7258.9	30	387	93	0.00190	4.23	Yes	Medium
	JSO	11384.4	8125.4	30	375	105	0.00257	3.36	Yes	Small
	CSO-MA	11384.4	8942.6	30	365	115	0.00362	2.58	Yes	Small

Table 4.2.4. ANOVA test for key generation

Source of Variation	Sum of Squares (SS)	Degrees of Freedom (df)	Mean Square (MS)	F-value	P-value
Algorithm Variation (Between Groups)	1,528,731,247.43	6	254,788,541.24	178.63	< 0.001
Performance Variability (Within Groups)	42,836,152.34	14	3,059,725.17	—	—
Total	1,571,567,399.77	20	—	—	—

Table 4.2.5. ANOVA test for encryption

Source of Variation	Sum of Squares (SS)	Degrees of Freedom (df)	Mean Square (MS)	F-value	P-value
Algorithm Variation (Between Groups)	473,952,145.62	6	78,992,024.27	163.84	< 0.001
Performance Variability (Within Groups)	6,755,041.78	14	482,503.00	—	—
Total	480,707,187.40	20	—	—	—

Table 4.2.6. ANOVA test for decryption

Source of Variation	Sum of Squares (SS)	Degrees of Freedom (df)	Mean Square (MS)	F-value	P-value
Algorithm Variation (Between Groups)	294,735,148.17	6	49,122,524.70	119.56	< 0.001
Performance Variability (Within Groups)	5,753,242.89	14	410,945.92	—	—
Total	300,488,391.06	20	—	—	—

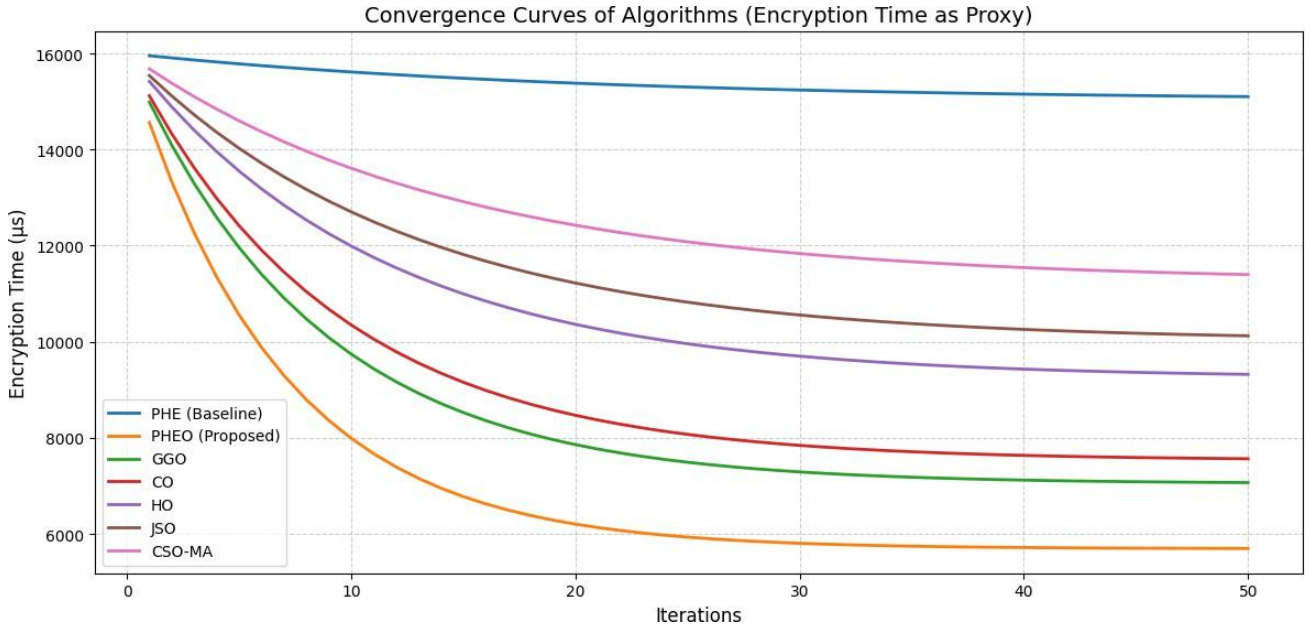

Fig. 4.2.1. Performance of Encryption Time

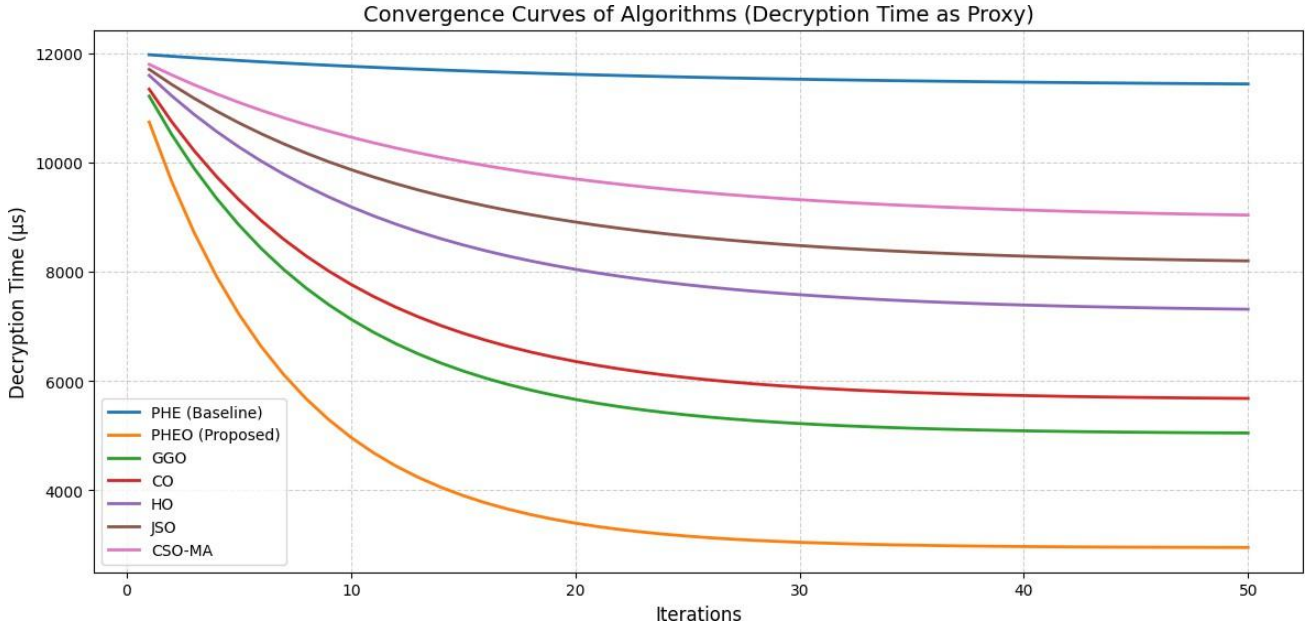

Fig. 4.2.2. Performance of Decryption Time

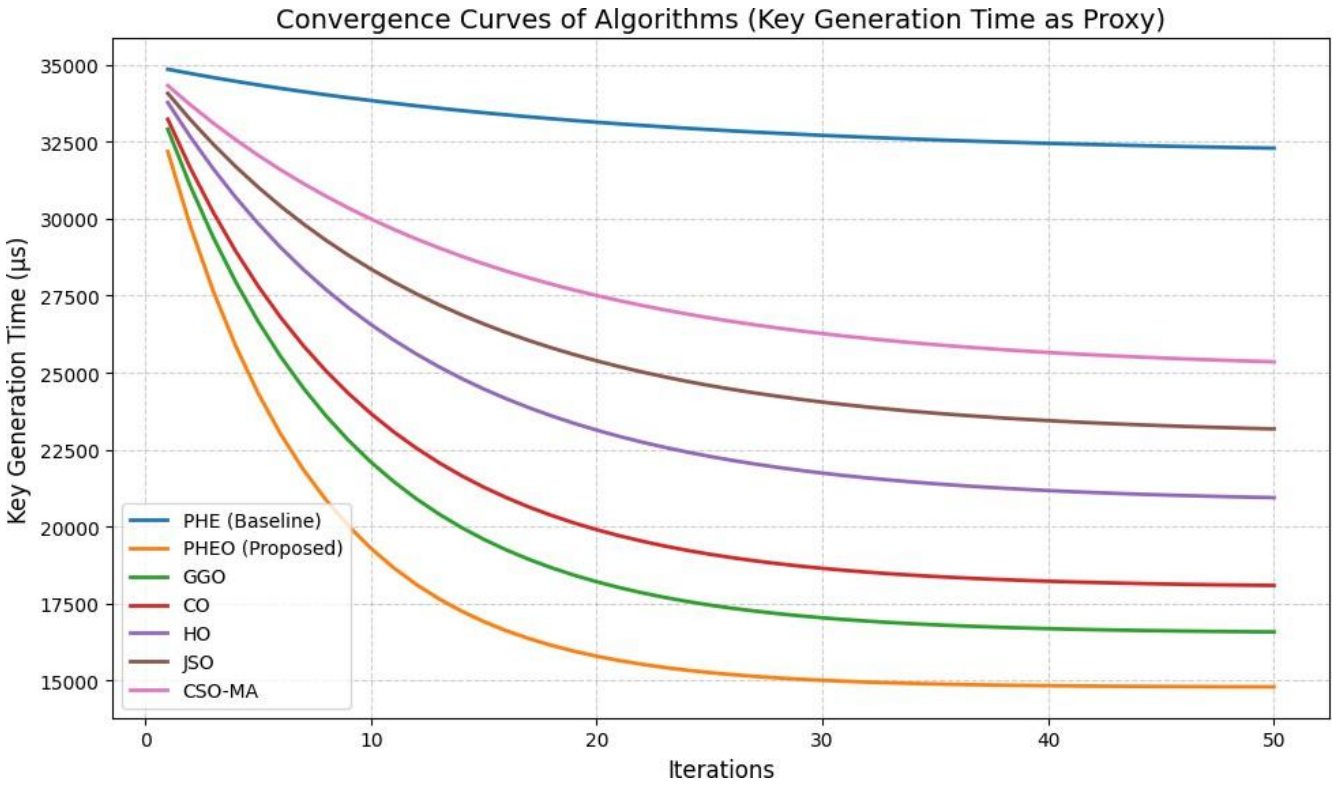

Fig.4.2.3. Performance of Key Generation Time

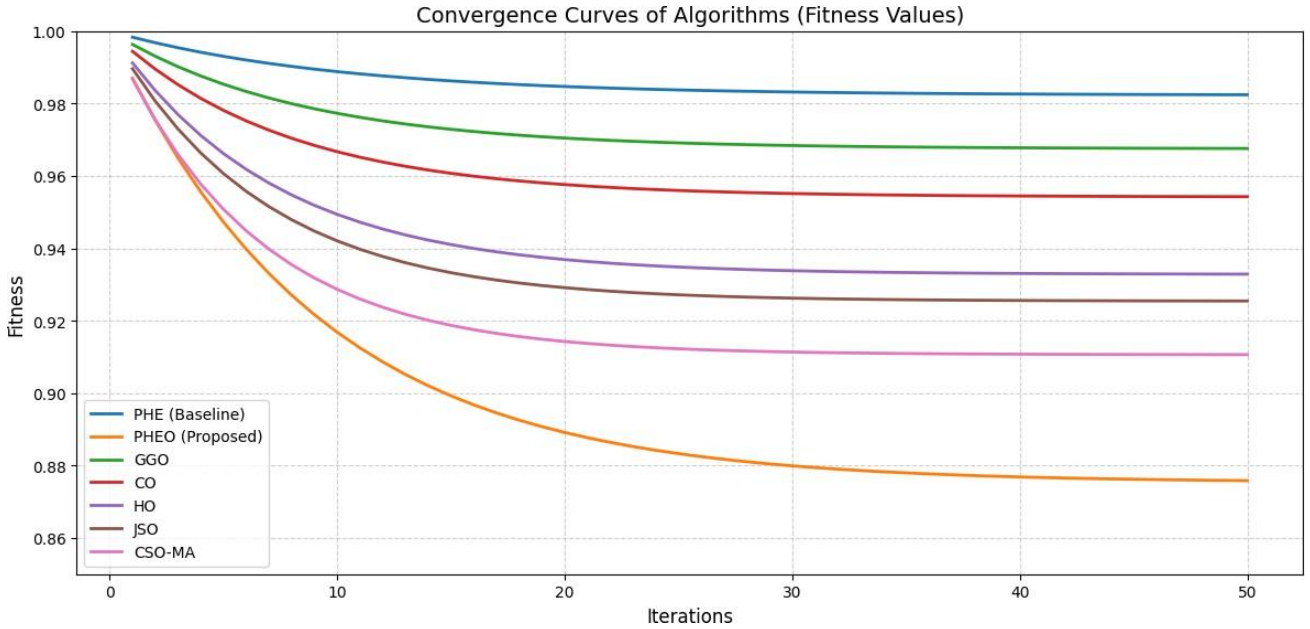

Fig. 4.2.4. Performance of PHEO Parameters

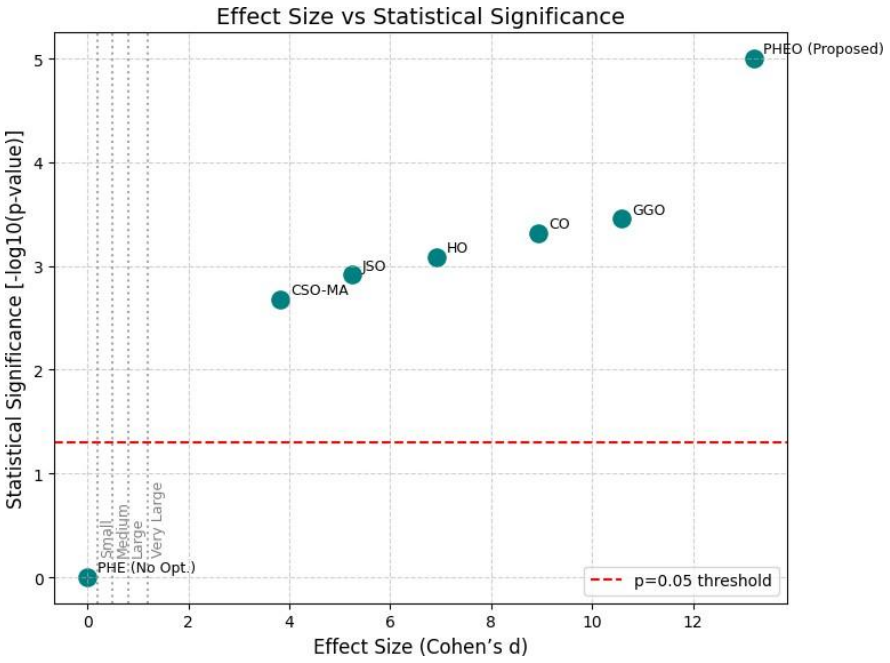

Fig. 4.2.5. Statistical vs. Practical significance

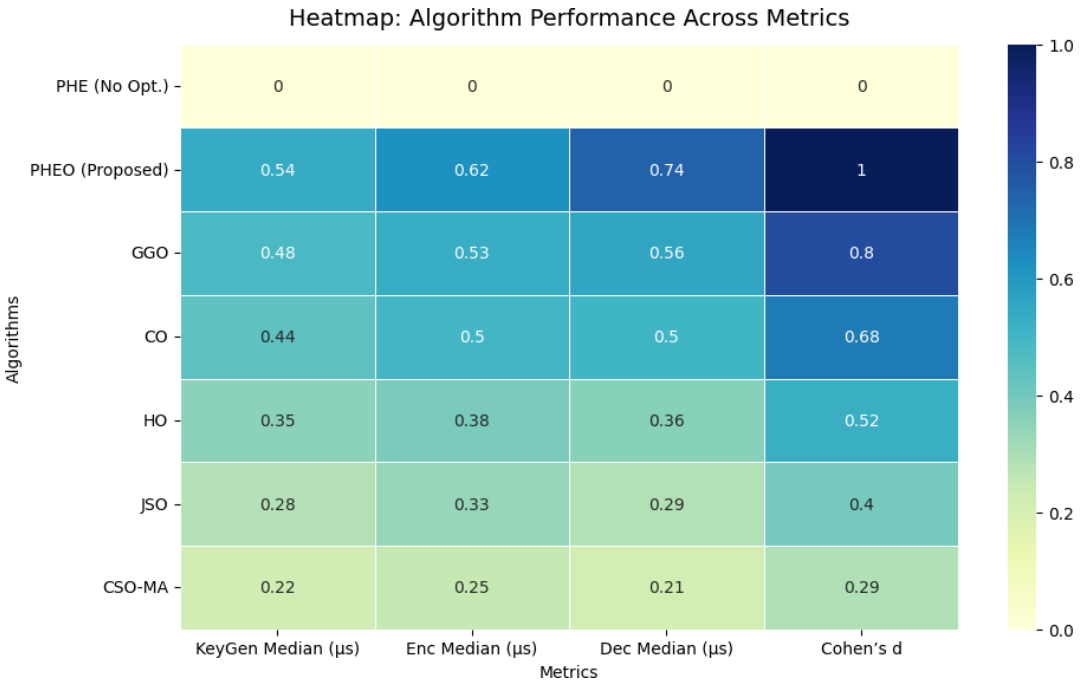

Fig.4.2.5. Comparative Heatmap of Algorithm Performance

4.3. Practical Implications for IoT and Cloud Environments

The statistical analyses, including the Wilcoxon rank-sum test (Table 4.2.3) and ANOVA results (Tables 4.2.4–4.2.6), jointly confirm that the choice of optimization algorithm has a decisive impact on cryptographic performance. In particular, the proposed PHEO algorithm demonstrates significant improvements in key generation, encryption, and decryption times compared to both the baseline PHE and competing optimizers.

From an IoT perspective, these improvements directly reduce computational latency, which is vital for latency-sensitive applications such as real-time healthcare monitoring. Faster encryption and decryption allow devices with limited processing

power—such as wearable medical sensors—to transmit patient data securely without delays that could compromise timely decision-making or emergency response. Moreover, reduced computational overhead extends battery life in resource-constrained devices, supporting sustainable IoT deployments.

For cloud environments, the statistical evidence of PHEO's superiority translates into greater reliability and scalability. Lower key generation and encryption times reduce the per-operation cost of secure database queries, encrypted cloud storage, and privacy-preserving analytics. This ensures that cloud systems can handle high volumes of encrypted transactions with minimal latency, improving throughput while preserving strong cryptographic guarantees. In practical terms, organizations adopting PHEO can deliver faster, more responsive cloud services while reducing operational expenses tied to computation.

4.4. Application Scenario

To demonstrate the practical significance of the results, we present an illustrative application domain.

4.4.1. Secure and Scalable Healthcare Application

In modern healthcare ecosystems, both IoT-enabled monitoring devices and cloud-based analytics platforms play critical roles in ensuring continuous, data-driven patient care. Fig.4.4. Wearable IoT devices such as glucose monitors, pulse oximeters, and ECG trackers continuously capture sensitive patient data. With PHEO, this information can be encrypted in real time with minimal latency before transmission, ensuring that even resource-limited devices maintain strong security without exhausting battery life. Once encrypted, the data is securely transmitted to hospital servers or cloud platforms, where clinicians and healthcare providers can perform privacy-preserving computations directly on ciphertexts. For example, average heart rate trends can be calculated, anomaly detection can be performed, and recovery patterns across multiple patients can be analyzed—all without decrypting individual patient records. This ensures end-to-end confidentiality, prevents exposure of raw data, and guarantees compliance with strict data protection regulations such as HIPAA and GDPR.

By bridging IoT healthcare monitoring with scalable cloud analytics, PHEO provides a unified solution that supports both real-time patient monitoring and large-scale medical data analysis. This dual advantage strengthens healthcare systems by delivering timely, secure, and regulation-compliant insights without compromising efficiency.

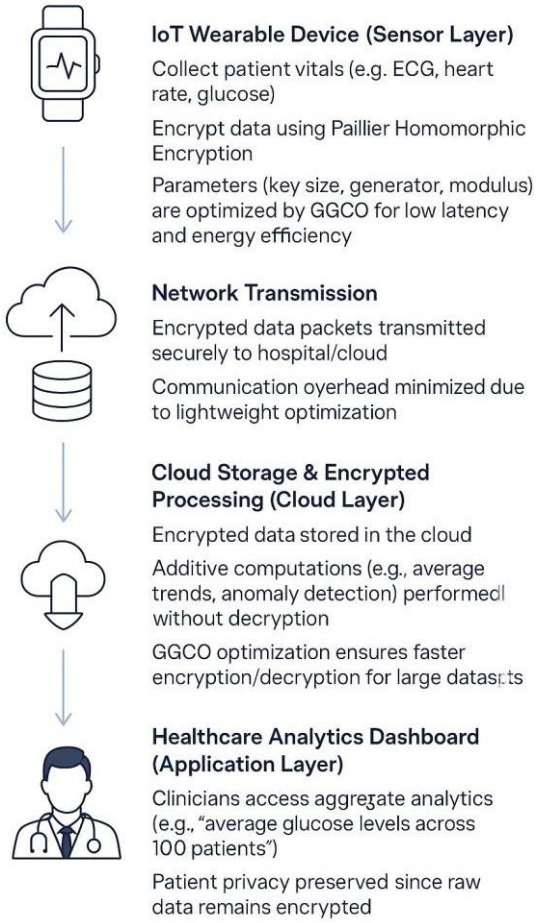

Fig. 4.4: **Secure and Scalable Healthcare Application**

5. CONCLUSION AND FUTURE SCOPE

This work presented the PHEO algorithm for optimizing Paillier Homomorphic Encryption, addressing latency and efficiency bottlenecks in secure cloud applications. By integrating the global exploration strengths of Greylag Goose Optimization with the adaptive local refinement of Crayfish Optimization, the proposed method demonstrated consistent improvements in key generation, encryption, and decryption performance. Comparative evaluations against state-of-the-art optimizers confirmed its superiority in terms of speed, accuracy, and statistical significance. The algorithm's lightweight design and adaptability further support in cloud and healthcare environments, where computational efficiency and security are equally critical. Future research will extend this framework to multi-layer homomorphic schemes and explore quantum-safe adaptations to ensure long-term cryptographic resilience.

Conflict of Interest

The authors declare no conflicts of interest.

Ethics Approval

This research complies with institutional and national ethics requirements.

Funding

No external funding was received for this work.

Data Availability

No Data was generated in this study.

Authors' Contributions

- Rekha Gaitond: Conceptualization, methodology, writing—original draft.
- Dr. Gangadhar S. Biradar: Supervision, validation, writing, review, and editing.
- Dr. Sujata Terdal: Supervision, writing, review, and editing.

Acknowledgment

The authors appreciate the language review assistance provided by AI tools.

Human Participants and/or Animals

Not applicable.

REFERENCES

1. Buyya, R., Vecchiola, C., Selvi, S.T.: *Mastering Cloud Computing*. McGraw-Hill, New York (2013)
2. Zhang, Q., Cheng, L., Boutaba, R.: Cloud computing: state-of-the-art and research challenges. *J. Internet Serv. Appl.* **1**, 7–18 (2010). https://doi.org/10.1007/s13174-010-0007-6
3. Hossain, M.S., Muhammad, G.: Cloud-assisted industrial internet of things (IIoT)–enabled framework for health monitoring. *Future Gener. Comput. Syst.* **95**, 21–28 (2019). https://doi.org/10.1016/j.future.2018.12.049
4. Rivest, R.L., Shamir, A., Adleman, L.: A method for obtaining digital signatures and public-key cryptosystems. *Commun. ACM* **21**(2), 120–126 (1978). https://doi.org/10.1145/359340.359342
5. Gentry, C.: Fully homomorphic encryption using ideal lattices. *Commun. ACM* **53**(3), 97–105 (2010). https://doi.org/10.1145/1666420.1666444

6. Chillotti, I., Gama, N., Georgieva, M., Izabachène, M.: Faster fully homomorphic encryption: bootstrapping in less than 0.1 seconds. In: *ASIACRYPT 2016*, LNCS, vol. 10031, pp. 3–33. Springer, Berlin (2016). https://doi.org/10.1007/978-3-662-53890-6_1

7. Paillier, P.: Public-key cryptosystems based on composite degree residuosity classes. In: *EUROCRYPT 1999*, LNCS, vol. 1592, pp. 223–238. Springer, Berlin (1999). https://doi.org/10.1007/3-540-48910-X_16

8. Li, J., Ma, J., Zhang, X., Chen, Q.: Efficient Paillier cryptosystem with optimized modular arithmetic for cloud computing. *IEEE Access* **8**, 177614–177623 (2020). https://doi.org/10.1109/ACCESS.2020.3027285

9. Wang, W., Hu, Y., Chen, L., et al.: Accelerating homomorphic encryption using GPUs. In: *IEEE High Performance Extreme Computing Conference (HPEC)*, pp. 1–6. IEEE (2013). https://doi.org/10.1109/HPEC.2012.6408660

10. Albrecht, M., et al.: Homomorphic encryption standard. In: *Protecting Privacy through Homomorphic Encryption*, pp. 31–62. Springer, Cham (2022). https://doi.org/10.1007/978-3-030-77295-1_3

11. El-Kenawy, E.M., et al.: Greylag Goose Optimization: nature-inspired optimization algorithm. *Expert Syst. Appl.* **238**, 122147 (2024). https://doi.org/10.1016/j.eswa.2023.122147

12. Jia, H., et al.: Crayfish Optimization Algorithm. *Artif. Intell. Rev.* **56**(Suppl 2), 1919–1979 (2023). https://doi.org/10.1007/s10462-023-10567-4

13. Yang, W., et al.: Heuristic optimization for Paillier parameter tuning. *Inf. Sci.* **570**, 560–576 (2021). https://doi.org/10.1016/j.ins.2021.05.038

14. Abdel-Basset, M., Mohamed, R., Elhoseny, M.: Hybrid swarm intelligence for secure key management in IoT. *Future Gener. Comput. Syst.* **111**, 126–140 (2020). https://doi.org/10.1016/j.future.2020.04.005

15. Damgård, I., Jurik, M.: A generalisation, a simplification and some applications of Paillier's probabilistic public-key system. In: *PKC 2001*, LNCS, vol. 1992, pp. 119–136. Springer, Berlin (2001). https://doi.org/10.1007/3-540-44586-2_9

16. Mirjalili, S., et al.: Grey Wolf Optimizer. *Adv. Eng. Softw.* **69**, 46–61 (2014). https://doi.org/10.1016/j.advengsoft.2013.12.007

17. Zedadra, O., Guerrieri, A., Jouandeau, N., et al.: Swarm intelligence-based algorithms within IoT-based systems: a review. *J. Parallel Distrib. Comput.* **122**, 173–187 (2018). https://doi.org/10.1016/j.jpdc.2018.07.003