Vol. 8, No. 3, June 2026

ISI Journal of Information Systems and Informatics

Published By Asosiasi Doktor Sistem Informasi Indonesia

Key Scalability Effects on Entropy and Computational Complexity in a GA-SA Hybrid Cryptosystem

Naufal Muzakki [1], Nur Rochmah Dyah Puji Astuti [2]

[1,2]Department of Informatics, Faculty of Industrial Technology, Universitas Ahmad Dahlan, Yogyakarta, Indonesia

Received:
October 13, 2025
Revised:
April 26, 2026
Accepted:
May 27, 2026
Published:
June 22, 2026

Corresponding Author:
Author Name*:
Nur Rochmah Dyah Puji Astuti
Email*:
rochmahdyah@tif.uad.ac.id

DOI:
10.63158/journalisi.v8i3.1607

Abstract. Digital data security demands robust encryption systems in which key randomness quality serves as the primary determining factor. Metaheuristic algorithms such as the Genetic Algorithm (GA) and Simulated Annealing (SA) exhibit significant potential for key generation optimization. However, each is individually susceptible to premature convergence and slow computational time, respectively, motivating their sequential hybridization. This study proposes a GA-SA hybrid cryptographic architecture with dynamic population sizing to optimize pseudo-random keystream generation in XOR encryption, evaluated using 15 PDF document datasets across three key configurations: 16 characters (128-bit), 32 characters (256-bit), and 64 characters (512-bit). The hybrid system consistently reduced local optima entrapment across all configurations, with the 64-character key achieving the highest randomness quality at a Shannon Entropy of 7.9288 bits/byte and a mean NIST SP 800-22 Monobit Frequency Test P-Value of 0.2999, though this does not constitute a full NIST SP 800-22 suite evaluation. Runtime analysis showed near-linear empirical growth within the tested range, from 0.0361 seconds to 0.1305 seconds, without exponential bottleneck effects, suggesting the proposed architecture is a promising candidate for pseudo-random keystream generation under tested conditions, with further validation recommended before production deployment.

Keywords: Metaheuristic Hybridization, Pseudo-Random Keystream Generation, XOR Encryption, Genetic Algorithm, Simulated Annealing, Shannon Entropy, Key Scalability, Cryptographic Randomness.

1. INTRODUCTION

The surge in digital data transfer volume during the era of technological transformation demands robust protection mechanisms capable of addressing increasingly sophisticated cybersecurity threats [1]. Among the various existing encryption techniques, algorithms based on the Exclusive-OR (XOR) operation remain an essential method due to their exceptionally high execution speed [2]. The performance of XOR algorithms offers significant advantages in terms of minimal memory consumption, making them an ideal solution for resource-constrained environments such as the Internet of Things and real-time systems [3]. Consequently, XOR continues to occupy a foundational role in lightweight cryptographic system design, particularly where processing efficiency is a primary constraint.

Despite its high efficiency, the security level of the XOR algorithm is absolutely dependent on the quality and degree of randomness of the encryption key, wherein a static key exposes a critical vulnerability to adversarial attacks [4]. Intelligent metaheuristic-based approaches such as the Genetic Algorithm (GA) have been proven reliable in solving solution-space search problems and generating adaptive cryptographic keys [5]. Nevertheless, conventional GA frequently encounters premature convergence, whereby the algorithm loses population diversity too rapidly and becomes trapped at a local optimum [6]. Keys generated under such conditions may exhibit suboptimal entropy, leaving residual statistical patterns that reduce the overall unpredictability of the keystream.

To overcome this limitation, algorithm hybridization has emerged as a promising solution by combining the exploratory strength of GA with the local refinement capability of complementary algorithms [7]. Simulated Annealing (SA) is selected as an effective complement due to its unique probabilistic mechanism that enables escape from local optimum traps [8]. The strategic sequential integration of genetic evolution and SA post-processing enhances the robustness of complex cryptographic systems [9]. This synergistic architecture positions GA-SA hybridization as a structurally sound candidate for generating high-quality keystreams with more uniform bit distributions than either method achieves independently.

The development of data security in digital transmission systems continues to demand cryptographic algorithms that are computationally efficient yet resilient against advanced cryptanalytic attacks [10]. While 128-bit keys currently remain within acceptable security margins under conventional threat models, sustained advances in computational hardware and the prospective emergence of quantum computing may gradually reduce the effective security margin of shorter key lengths over time [11]. The application of hybrid metaheuristic approaches to analyze computational scalability when keystream generation search space is incrementally expanded on fundamental ciphers such as XOR remains largely underexplored [12]. The primary challenge, therefore, lies in rigorously characterizing the trade-off between enhanced key security and computational overhead as key dimensionality scales upward.

An increase in chromosome dimension automatically expands the genetic search space and directly impacts algorithm performance [13]. A thorough evaluation of execution duration is essential to measure computational time stability when the system handles exponential increases in operational scale [14]. Furthermore, the information uncertainty level of large-scale keys must be continuously validated using standard entropy calculations to ensure that bit distribution approximates the ideal value [15]. The preceding analysis identifies a concrete research gap: while GA-SA hybridization has been applied in various optimization domains, its systematic evaluation as a keystream generator for XOR encryption specifically regarding how entropy quality and runtime complexity respond to increasing key length remains insufficiently addressed, motivating this study to propose a sequential GA-SA hybrid architecture with dynamic population sizing evaluated across 128-bit, 256-bit, and 512-bit configurations, with contributions spanning entropy-scalability characterization, runtime complexity profiling, and a baseline reference for computationally efficient metaheuristic-driven XOR encryption systems.

2. METHODS

This study employs an algorithmic experimental method to evaluate the computational scalability and cryptographic randomness quality of a sequential GA-SA hybrid keystream generation architecture across three key length configurations [16]. The research design proceeds chronologically through five stages: (1) dataset preparation and parameter

initialization, (2) stochastic population generation, (3) GA-driven global exploration, (4) SA-driven local exploitation, and (5) metric acquisition covering fitness convergence, Shannon Entropy, NIST Monobit validation, and runtime measurement [17]. The overall workflow is illustrated in Figure 1.

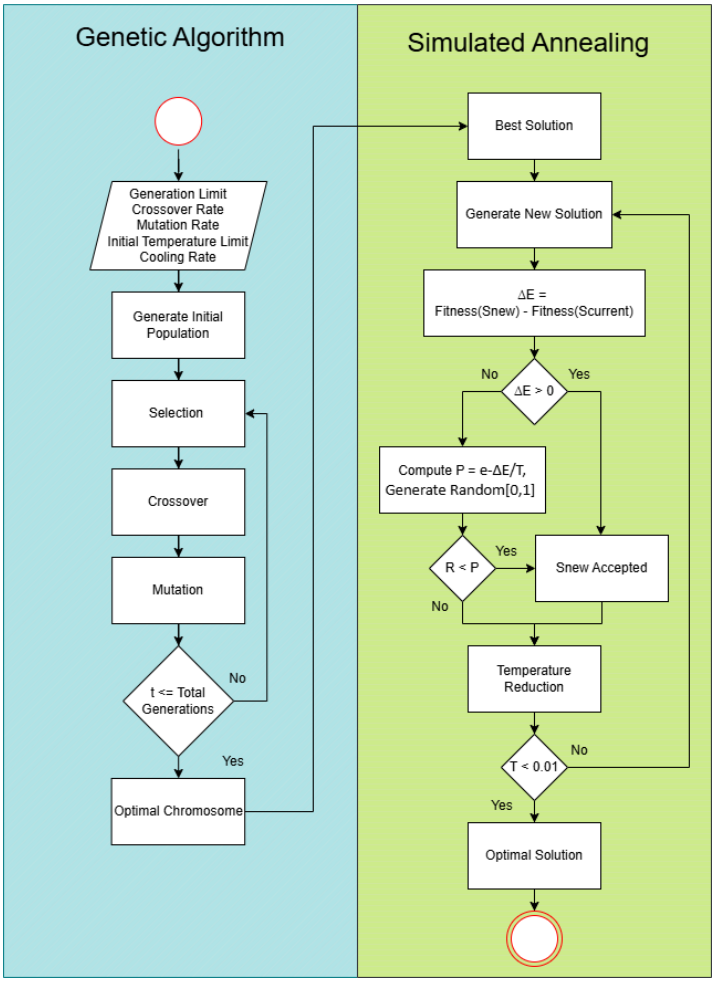

Figure 1. Flowchart of GA-SA hybrid cryptosystem

2.1. Algorithmic Workflow Narrative

The procedure begins with the initialization of all algorithm parameters and the construction of an initial population of candidate keys. Each individual in the population is represented as a chromosome of dynamic length L, where each gene encodes a single ASCII character of the keystream [5]. The initial population is generated stochastically without a fixed random seed, meaning that each experimental run begins from a randomly drawn starting state [18]. The implications of this choice for reproducibility are discussed further in Section 2.4.

During the GA phase, all chromosomes in the population are first evaluated using the fitness function defined in Equation 3. Selection is performed via Tournament Selection, in which a subset of individuals competes and the fittest candidate is retained as a parent [19]. Genetic recombination is then carried out through Two-Point Crossover, which exchanges a segment of genetic material between two parent chromosomes at two randomly chosen cut points [20]. This is followed by Hybrid Mutation, which applies two complementary operators sequentially: Inversion Mutation, which reverses the order of characters within a randomly selected chromosome segment to perturb local structure, and Random Resetting Mutation, which replaces selected gene positions with entirely new random character values to inject diversity and prevent premature convergence [21][22]. This full cycle of evaluation, selection, crossover, and mutation is repeated for a fixed number of generations *G*, after which the chromosome carrying the highest fitness value is extracted as the best solution from the GA phase [23].

The GA output is then passed directly as the initial solution $S_{current}$ to the SA phase, which performs local exploitation through a probabilistic perturbation mechanism governed by the Metropolis Criterion [24][8]. At each SA iteration, a neighboring solution S_{new} is generated by randomly modifying one or more characters of the current key. If the new solution yields a higher fitness value ($\Delta E > 0$), it is accepted unconditionally. If it yields a lower fitness value $\Delta E < 0$, it is accepted with a probability defined in Equation 1, allowing the algorithm to escape local optima:

$$P(accept) = \begin{cases} 1, & if\ \Delta E > 0 \\ e^{\Delta E/T}, & if\ \Delta E < 0 \end{cases} \quad (1)$$

Where P is the acceptance probability, ΔE represents the fitness difference $\left(Fitness(S_{new}) - Fitness(S_{current})\right)$, and T is the current system temperature. The temperature is reduced at the end of each iteration according to a geometric cooling schedule [25][26]:

$$T_{k+1} = T_k \times \alpha \quad (2)$$

Where T_{k+1} is the temperature at the next iteration, T_k is the current temperature, and α is the cooling rate constant where $0 < \alpha < 1$. This cooling mechanism ensures that the probability of accepting inferior solutions diminishes progressively as the search

converges. The SA phase terminates when the temperature falls below a minimum threshold T_{min}, at which point the current best solution is returned as the optimal cryptographic key K_{opt}. The complete sequential procedure is formalized as pseudocode in Table 1.

Table 1. Pseudocode of sequential hybrid GA-SA

1.	N : Population Size
2.	G : Generations
3.	L : Key Length
4.	T_0 : SA Initial Temperature
5.	α : Cooling Rate
6.	K_{opt} : Optimal Cryptographic Key
7.	
8.	Initialize population P with size N of length L
9.	For i = 1 to G do:
10.	Evaluate fitness of all individuals in P
11.	Apply Tournament Selection to pick parents
12.	Apply Two-Point Crossover & Hybrid Mutation to form P_{new}
13.	$P \leftarrow P_{new}$
14.	End For
15.	Extract S_{best} from P as initial solution for SA ($S_{current} \leftarrow S_{best}$)
16.	$T \leftarrow T_0$
17.	While $T > T_{min}$ do:
18.	Generate neighborS_{new} by perturbing $S_{current}$
19.	Calculate $\Delta E = Fitness(S_{new}) - Fitness(S_{current})$
20.	If $\Delta E > 0$ or Random $(0,1) < e^{\frac{\Delta E}{T}}$ then:
21.	$S_{current} \leftarrow S_{new}$
22.	End If
23.	$T \leftarrow T \times \alpha$
24.	End While
25.	Return $S_{current}$ as K_{opt}

2.2. Dataset Description

To validate algorithm robustness across varied input conditions, 15 PDF documents were used as the source material for keystream generation targets. The documents were

deliberately selected to represent a range of file characteristics: file sizes span from approximately 50 KB to 200 KB, covering short single-page technical documents, multi-page academic reports, and text-dense reference materials. Document content varies across domains including electronic health records covering patient data, medical reports, and clinical documentation, ensuring that the character distribution of the target material is not homogeneous. No preprocessing or normalization was applied to the PDF content prior to testing, preserving the natural byte-level diversity of each document.

2.3. Dynamic Population Sizing: Proposed Framework and Baseline Comparison

A central architectural contribution of this study is the dynamic population sizing strategy, in which the number of chromosomes in the initial population N is set proportionally equal to the key length L at each configuration (i.e., $N = L$). This stands in contrast to the conventional baseline approach of using a fixed population size regardless of problem dimensionality, a strategy that risks under-representation of the search space at higher key lengths or unnecessary computational overhead at shorter ones [27]. As key length increases, the combinatorial solution space grows exponentially, and a proportionally larger population improves the probability of adequate initial coverage of that space [28]. While a comprehensive head-to-head comparison with a fixed-population baseline falls outside the scope of this experiment, future work is recommended to directly benchmark the dynamic strategy against fixed-size configurations (e.g., $N = 50$ across all key lengths) to quantify the trade-off more precisely.

2.4. Experimental Setup and Reproducibility

All experiments were implemented in Python 3 and executed on a machine equipped with an Intel Core i7 processor and 16 GB RAM running Windows 11 [29][30]. Parameter values were determined through preliminary empirical tuning to identify the optimal balance between solution diversity and computational convergence speed for the tested datasets. The final configurations are listed in Table 2.

Regarding reproducibility, no fixed random seed was applied during either population initialization or mutation operations, meaning each trial begins from an independently drawn stochastic starting state [18]. To account for this non-determinism, each key length configuration was tested across 15 distinct PDF documents, and performance metrics are

reported as averages across all 15 runs. The absence of a fixed seed is acknowledged as a limitation in strict reproducibility. Future work should consider seeded execution to enable exact replication.

Table 2. Configuration of algorithm parameters

Parameter	Value
GA Parameters [19]	
1. Number of Generations	50
2. Crossover Rate	4
3. Mutation Rate	2
SA Parameters [31]	
1. Initial Temperature	100
2. Cooling Rate	0,95

2.5. Evaluation Metrics

The primary quantitative metrics evaluated in this study are the fitness objective function, Shannon Entropy, NIST Monobit $P-Value$, and computational runtime. Fitness guides the metaheuristic search by quantifying how closely a candidate keystream approximates an ideal target distribution, computed as defined in Equation 3 [32]:

$$Fitness = \frac{100{,}000}{\sum|Target-Chromosome|+1} \quad (3)$$

Where the summation accumulates the absolute ASCII value differences between each character position i of the target key and the corresponding character of the candidate chromosome, L is the key length, the constant numerator $100{,}000$ serves as a scaling factor, and the denominator offset of $+1$ prevents division-by-zero.

Cryptographic randomness quality is measured using Shannon Entropy [33], as represented in Equation 4:

$$H = -\sum_{i=1}^{n} p(x_i) log_2 p(x_i) \quad (4)$$

Where H is the information uncertainty in bits per byte, $n = 256$ for an 8-bit byte system, and $p(x_i)$ is the probability of occurrence of the i-th byte value in the keystream.

The theoretical maximum is 8.0 bits/byte, approached as byte distribution becomes perfectly uniform.

Statistical randomness is additionally evaluated using the NIST SP 800-22 Monobit Frequency Test, which assesses whether the proportions of ones and zeros in the binary key sequence are approximately equal a key sequence passes the test if it produces a $P - Value \geq 0.01$ [34]. It must be explicitly acknowledged that the Monobit test represents only one of fifteen statistical tests in the full NIST SP 800-22 suite, and passing it alone is not sufficient to constitute comprehensive cryptographic randomness certification. It provides indicative evidence of first-order bit-level uniformity only [34]. Evaluation against the complete NIST SP 800-22 battery is therefore recommended as a priority direction for future work.

2.6. Computational Time Complexity

Theoretical complexity analysis and empirical runtime measurement are treated as distinct but complementary evaluations in this study. From a theoretical standpoint, under the dynamic population sizing scheme where $N = L$, the dominant cost driver across both the GA and SA phases scales as a function of L. With a fixed number of generations G and a fixed number of SA iterations determined by the cooling schedule, the total operation count per run grows proportionally to L, yielding an asymptotic time complexity of $O(n)$ with respect to key length [17]. This theoretical expectation does not account for constant factors introduced by Python interpreter overhead, memory access patterns, or hardware-level caching effects.

Empirical runtime is measured independently using the internal system timer, capturing wall-clock execution time in seconds from the moment of population initialization through to the return of K_{opt} at the conclusion of the SA phase. These empirically measured values are reported in Section 3.3 and serve as the primary evidence for evaluating whether the theoretical $O(n)$ growth prediction holds in practice under the tested hardware and software environment.

3. RESULTS AND DISCUSSION

This section presents the empirical testing results from the implementation of the GA-SA hybrid cryptographic algorithm across three key length configurations. The evaluation

is organized into two parts: results reporting (Sections 3.1–3.3), which presents quantitative findings per metric, and a dedicated discussion (Section 3.4), which synthesizes the findings analytically and situates them within broader cryptographic considerations. The GA-SA hybrid architecture was selected as the foundational system for this scalability evaluation because preliminary testing demonstrated its capability to elevate fitness values and mitigate the premature convergence that frequently occurs in standalone GA [6]. Testing was executed using 15 distinct PDF document datasets per configuration, covering three key length scalability scenarios: 16 characters (128-bit), 32 characters (256-bit), and 64 characters (512-bit), implemented with a dynamic population sizing approach.

3.1. Fitness Convergence Stability Analysis

The first metric evaluated is the algorithm's ability to consistently identify near-optimal solutions across varied input document conditions. The stability profile of final fitness values across the three key length scenarios on 15 PDF files is visualized in Figure 2.

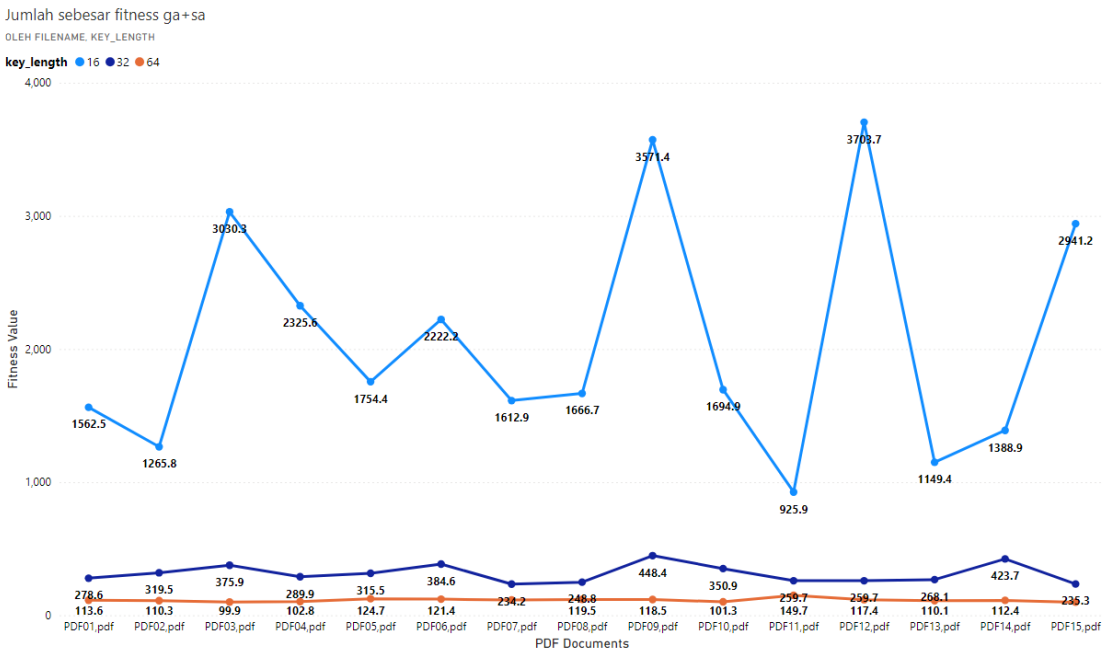

Figure 2. Fitness convergence stability across 15 different PDF datasets

As illustrated in Figure 2, a clear layering pattern is observable, wherein fitness values for the 16-character key occupy the highest position, followed by 32-character, and 64-character at the lowest. Notably, the standard deviation decreases substantially as key

ISI Journal of Information Systems and Informatics
Published By Asosiasi Doktor Sistem Informasi Indonesia

length increases from ± 879.36 at $L = 16$ down to ± 12.70 at $L = 64$, indicating that the algorithm produces more consistent, stable results as the search space expands. The analytical interpretation of the mean fitness decline across configurations is addressed in Section 3.4.

3.2. Entropy Evaluation and Statistical Randomness Testing

The cryptographic randomness quality of the generated keystream is measured using Shannon Entropy and validated using the NIST SP 800-22 Monobit Frequency Test. To provide a more complete picture of result variability across the 15 document trials, Table 3 reports both the mean and standard deviation for each metric per key length configuration.

Table 3. Average entropy and randomness evaluation results

Key Length	Mean Entropy	SD Entropy	Mean P-*Value*	SD P-Value	Status
16 Characters	7.8447	± 0.0121	0.7151	± 0.2532	Random
32 Characters	7.8984	± 0.0112	0.5921	± 0.2686	Random
64 Character	7.9288	± 0.0063	0.2999	± 0.2047	Random

All three configurations produced entropy values closely approaching the theoretical maximum of 8.0 bits/byte, with a monotonically increasing trend: $7.8447 \rightarrow 7.8984 \rightarrow 7.9288$. The standard deviation of entropy narrows progressively from ± 0.0121 to ± 0.0063, indicating increasingly consistent randomness quality as key length grows. All configurations passed the Monobit Frequency Test with mean $P - Values$ exceeding the 0.01 threshold. However, the minimum $P - Value$ recorded at $L = 64$ reached 0.0004 in one trial still above the rejection threshold of 0.01, though individual trials can produce borderline results. The broader cryptographic interpretation is addressed in Section 3.4 [34].

3.3. Computational Time Complexity Scalability

Execution time was measured from the GA population initialization phase through to the return of K_{opt} at the conclusion of the SA cooling process. Mean runtime values across 15 document trials per configuration are presented in Figure 3.

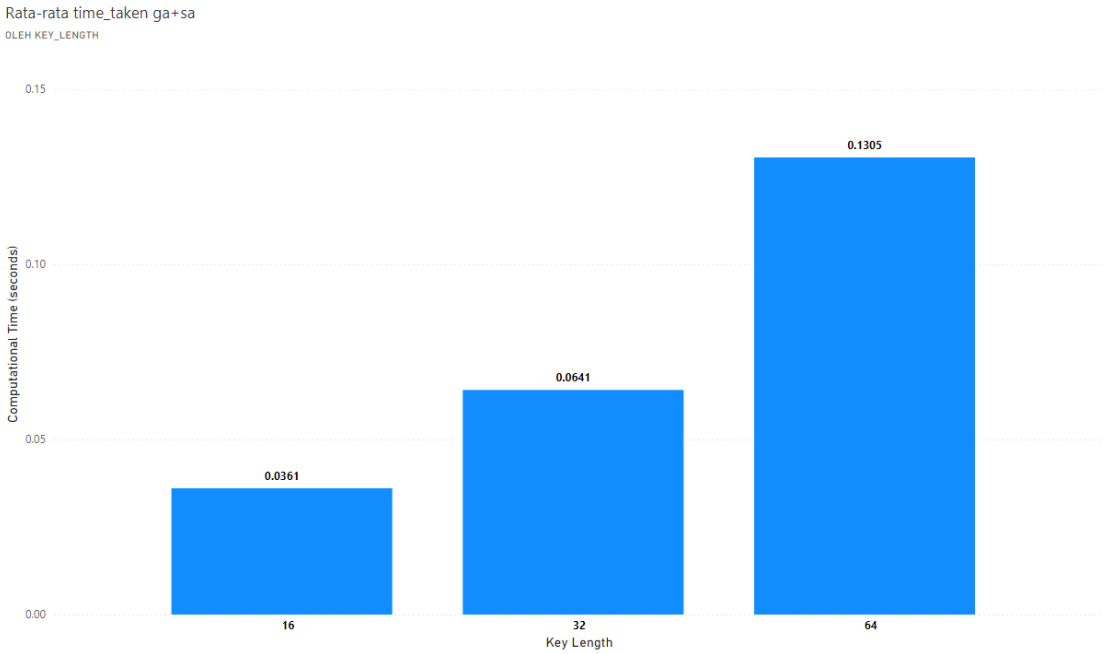

Figure 3. Average computational time scalability based on key dimensions

The recorded mean execution times are 0.0361 seconds ($L = 16$), 0.0641 seconds ($L = 32$), and 0.1305 seconds ($L = 64$). with notably small standard deviations across all configurations ($\leq \pm 0.0033$ seconds), confirming highly stable and predictable runtime behavior throughout all 15 document trials. Across these three tested configurations, the growth pattern is approximately proportional to the doubling of key length, suggesting near-linear scaling behavior within the range evaluated. It should be noted, however, that characterizing asymptotic complexity from three data points alone is methodologically insufficient to formally establish $O(n)$ behavior, as the observed trend is consistent with near-linear growth but cannot be generalized beyond the tested range without additional configurations. A broader runtime analysis spanning a wider range of key lengths is recommended to substantiate the scaling characterization more rigorously. The full analytical interpretation of this scaling pattern in relation to the dynamic population sizing mechanism is provided in Section 3.4.

3.4. Discussion

3.4.1. Interpreting Fitness Decline Under Key Length Expansion

The steep decline in mean fitness values from 2054.39 at $L = 16$ to 114.37 at $L = 64$ is a direct and mathematically predictable consequence of the fitness function structure in Equation 3, not an indicator of degraded algorithm performance. As key length L

increases, the denominator accumulates absolute ASCII value differences across proportionally more character positions. Even under constant per-character deviation, the summed denominator grows with L, causing the fitness score to decrease inversely. This is compounded by the combinatorial explosion of the solution space at $L = 64$ (256^{64} possible combinations), making exact convergence to a fitness maximum structurally improbable within 50 fixed generations.

Critically, the data in Section 3.1 reveal an important counterintuitive finding: while mean fitness declines with key length, standard deviation also narrows dramatically from ± 879.36 at $L = 16$ to just ± 12.70 at $L = 64$, indicating that the algorithm becomes more consistent, not less reliable, as dimensionality increases. The high variance at $L = 16$ reflects the narrow solution space where random initialization can land at vastly different starting distances from the optimum, producing highly variable absolute fitness scores. At $L = 64$, the dynamic population mechanism distributes individuals more evenly across a wider space, resulting in more uniform convergence behavior across all 15 document trials. This stability confirms that the GA-SA hybrid architecture successfully resists local optimum entrapment even as problem dimensionality scales substantially. In contrast to fixed-population GA studies where population size remains constant regardless of key length, the dynamic sizing strategy employed here appears to contribute to this improved consistency at higher dimensions, though a direct head-to-head benchmark remains a recommended direction for future work.

3.4.2. Entropy and Monobit Results: Scope and Cryptographic Limitations

The entropy results confirm a positive relationship between key length and randomness quality, with the 64-character configuration achieving 7.9288 bits/byte within 0.09 bits/byte of the theoretical maximum and a tight standard deviation of ± 0.0063, indicating high consistency across document inputs. These findings are consistent with stochastic distribution principles: a longer character sequence provides more opportunities for uniform byte coverage across the 256-value sample space [33].

However, these results must be interpreted within clearly defined methodological boundaries. High Shannon Entropy confirms approximate uniformity in the byte-level distribution, but does not capture higher-order sequential dependencies, inter-byte correlations, or structural patterns operating across multiple positions simultaneously.

The NIST Monobit Frequency Test evaluates only the first-order balance between binary ones and zeros one of fifteen tests in the full NIST SP 800-22 suite and passing this single test is not equivalent to cryptographic randomness certification [34]. A keystream may pass the Monobit test while still exhibiting detectable patterns under runs tests, spectral analysis, serial correlation tests, or linear complexity evaluations. Furthermore, the minimum $P-Value$ of 0.0004 observed in one $L = 64$ trial underscores the importance of evaluating the full statistical battery rather than relying on mean values alone. It must be explicitly noted that this study did not perform evaluation against the full NIST SP 800-22 suite, Dieharder, or TestU01 test batteries, and no cryptanalytic attack evaluation was conducted. These constitute recognized methodological limitations, and addressing them is a necessary next step before stronger cryptographic security claims can be made.

3.4.3. Runtime Scaling: Empirical Observations and Theoretical Alignment

The runtime data exhibit near-proportional growth across the three tested configurations, with mean execution times approximately doubling each time key length doubles: $0.0361\,s \rightarrow 0.0641\,s \rightarrow 0.1305\,s$. The very small standard deviations ($\leq \pm 0.0033\,s$) confirm that this growth pattern is stable and not driven by outliers, and is broadly consistent with the $O(n)$ theoretical expectation derived from the dynamic population sizing scheme described in Section 2.6. It should be noted, however, that three data points are methodologically insufficient to formally establish asymptotic complexity; the observed pattern is more precisely described as near-linear growth within the tested range of 128-bit to 512-bit key configurations. These results align with the efficiency characteristics of XOR-based systems reported by Shyaa and Al-Zubaidie [35] and provide empirical evidence that the dual-phase GA-SA overhead remains computationally manageable with the most intensive configuration completing within 0.1305 seconds suggesting practical viability for time-sensitive applications within the tested parameter bounds.

4. CONCLUSION

This study implemented and evaluated the scalability of a sequential GA-SA hybrid cryptographic algorithm with dynamic population sizing for XOR keystream generation, tested across 15 PDF document datasets at three key length configurations (128-bit, 256-

bit, and 512-bit). Within the tested range, increasing key length produced a predictable fitness decline attributable to combinatorial space expansion, while Shannon Entropy increased monotonically from 7.8447 to 7.9288 bits/byte with progressively narrowing standard deviations, and all configurations passed the NIST SP 800-22 Monobit Frequency Test with mean execution times exhibiting near-linear empirical growth from 0.0361 to 0.1305 seconds ($\leq \pm 0.0033$ seconds), consistent with the $O(n)$ theoretical expectation under the dynamic population sizing scheme. These findings suggest that the proposed architecture is a promising candidate for pseudo-random keystream generation under tested conditions, with manageable computational overhead across the evaluated configurations. However, randomness evaluation was limited to the Monobit test alone and does not constitute full cryptographic certification; no evaluation against the complete NIST SP 800-22 battery, Dieharder, or TestU01 suites, nor any cryptanalytic attack assessment, was performed in this study. Future work should therefore prioritize evaluation against the full NIST SP 800-22 suite, extended runtime profiling across broader key length ranges, and assessment under cryptanalytic attack scenarios to more comprehensively characterize the security properties of the proposed system before any consideration of production deployment.

REFERENCES

[1] L. B. Handoko and C. Umam, "A super encryption approach for enhancing digital security using column transposition-hill cipher for 3D image protection," *Kinet. Game Technol. Inf. Syst. Comput. Network, Comput. Electron. Control*, Jun. 2024, doi: 10.22219/kinetik.v9i3.1984.

[2] S. Liang, Z. Dong, J. Peng, C. Jiang, and X. Zhao, "Research and implementation of secure dual algorithm-based transmission system for sensitive data of large models in an open environment," *IEEE Access*, vol. 13, pp. 165247–165262, 2025, doi: 10.1109/ACCESS.2025.3605073.

[3] D. Jonas, I. Sembiring, A. Setiawan, D. Manongga, I. R. Widiasari, and D. Julianingsih, "XOR encryption based on index value enumeration used in IoT based healthcare," in *Proc. 11th Int. Conf. Cyber IT Service Manage. (CITSM)*, IEEE, Nov. 2023, pp. 1–5. doi: 10.1109/CITSM60085.2023.10455527.

ISI Journal of Information Systems and Informatics
Published By Asosiasi Doktor Sistem Informasi Indonesia

[4] R. Reymond, J. Manullang, J. Tamba, F. P. Sitohang, and E. N. Ginting, "Cryptography with one-time pad (OTP) algorithm XOR based," *J. Tek. Indones.*, vol. 3, no. 02, pp. 54–60, Nov. 2024, doi: 10.58471/ju-ti.v3i02.664.

[5] R. B. Abduljabbar, O. K. Hamid, and N. J. Alhyani, "Features of genetic algorithm for plain text encryption," *Int. J. Electr. Comput. Eng.*, vol. 11, no. 1, pp. 434–441, Feb. 2021, doi: 10.11591/ijece.v11i1.pp434-441.

[6] S. S. Sultana, T. Tanabe, T. Fausten, and M. Irie, "Avoiding premature convergence to local optima with adaptive exploration for genetic algorithms," in *Proc. Genetic Evol. Comput. Conf. Companion (GECCO)*, ACM, Jul. 2025, pp. 539–542. doi: 10.1145/3712255.3726640.

[7] W. Yu, Y. Huang, Y. Yang, and A. Garcia-Ortiz, "Hybrid genetic algorithm combining simulated annealing for task allocation with data security," in *Proc. 19th Int. Conf. Distrib. Comput. Smart Syst. Internet Things (DCOSS-IoT)*, IEEE, Jun. 2023, pp. 134–141. doi: 10.1109/DCOSS-IoT58021.2023.00033.

[8] T. Zhang, B. Zhu, Y. Ma, and X. Zhou, "A novel image encryption algorithm based on multiple random DNA coding and annealing," *Electronics*, vol. 12, no. 3, p. 501, Jan. 2023, doi: 10.3390/electronics12030501.

[9] M. K. Nayak and P. K. Swain, "Color image encryption using lightweight cryptography and genetic algorithm for secure internet of things," *SSRG Int. J. Electr. Electron. Eng.*, vol. 11, no. 7, pp. 271–279, Jul. 2024, doi: 10.14445/23488379/IJEEE-V11I7P124.

[10] G. Singla and G. Kaur, "New enhanced hybrid cryptographic algorithms in cloud network," in *Proc. 1st Int. Conf. Adv. Electr. Electron. Comput. Intell. (ICAEECI)*, IEEE, Oct. 2023, pp. 1–6. doi: 10.1109/ICAEECI58247.2023.10370940.

[11] J. S. Yalli *et al.*, "A systematic review for evaluating IoT security: A focus on authentication, protocols and enabling technologies," *IEEE Internet Things J.*, vol. 12, no. 12, pp. 18908–18928, Jun. 2025, doi: 10.1109/JIOT.2025.3545737.

[12] H. Saini, G. Singh, and M. Rohil, "Design of hybrid metaheuristic optimization algorithm for trust-aware privacy preservation in cloud computing," *Int. J. Comput. Netw. Appl.*, vol. 10, no. 6, pp. 934–946, Dec. 2023, doi: 10.22247/ijcna/2023/223690.

[13] P. Mukherjee, H. Garg, C. Pradhan, S. Ghosh, S. Chowdhury, and G. Srivastava, "Best fit DNA-based cryptographic keys: The genetic algorithm approach," *Sensors*, vol. 22, no. 19, p. 7332, Sep. 2022, doi: 10.3390/s22197332.

[14] A. Almomani, K. Mhaidat, O. Al-Khaleel, and M. Al-Khaleel, "An efficient and scalable Montgomery modular multiplier for cryptographic applications," *IEEE Open J. Comput. Soc.*, vol. 6, pp. 1822–1833, 2025, doi: 10.1109/OJCS.2025.3628878.

[15] P. Bagane and S. Kotrappa, "Enriching AES through the key generation from genetic algorithm," *Indian J. Comput. Sci. Eng.*, vol. 12, no. 4, pp. 955–963, Aug. 2021, doi: 10.21817/indjcse/2021/v12i4/211204141.

[16] R. A. A. Qasim and B. S. Mahdi, "Image protection using genetic algorithm and cipher technique," *Iraqi J. Comput. Commun. Control Syst. Eng.*, vol. 22, no. 4, pp. 160–166, 2022, doi: 10.33103/uot.ijccce.22.4.13.

[17] M. Mitchell, *An Introduction to Genetic Algorithms*. Cambridge, MA, USA: MIT Press, 1996. doi: 10.7551/mitpress/3927.001.0001.

[18] R. Martín-Santamaría, S. Cavero, A. Herrán, A. Duarte, and J. M. Colmenar, "A practical methodology for reproducible experimentation: An application to the double-row facility layout problem," *Evol. Comput.*, vol. 32, no. 1, pp. 69–104, Mar. 2024, doi: 10.1162/evco_a_00317.

[19] A. Abid, M. Jarjar, M. Kattass, H. Rrghout, A. Jarjar, and A. Benazzi, "Genetic algorithm using Feistel and genetic operator acting at the bit level for images encryption," *Int. J. Saf. Secur. Eng.*, vol. 14, no. 1, pp. 15–27, Feb. 2024, doi: 10.18280/ijsse.140102.

[20] S. Rimcharoen and N. Leelathakul, "Ring-based crossovers in genetic algorithms: Characteristic decomposition and their generalization," *IEEE Access*, vol. 9, pp. 137902–137922, 2021, doi: 10.1109/ACCESS.2021.3117987.

[21] P. Larrañaga, C. M. H. Kuijpers, R. H. Murga, I. Inza, and S. Dizdarevic, "Genetic algorithms for the travelling salesman problem: A review of representations and operators," *Artif. Intell. Rev.*, vol. 13, no. 2, pp. 129–170, Apr. 1999, doi: 10.1023/A:1006529012972.

[22] S. Kumar and D. Sharma, "Key generation in cryptography using elliptic-curve cryptography and genetic algorithm," in *Eng. Proc.*, MDPI, Dec. 2023, p. 59. doi: 10.3390/engproc2023059059.

[23] O. Tito-Corrioso, M. Borges-Quintana, M. A. Borges-Trenard, O. Rojas, and G. Sosa-Gómez, "On the fitness functions involved in genetic algorithms and the cryptanalysis of block ciphers," *Entropy*, vol. 25, no. 2, p. 261, Jan. 2023, doi: 10.3390/e25020261.

[24] A. Hemmak, "Optimal adjusting of simulated annealing parameters," *Mil. Tech. Cour.*, vol. 72, no. 1, pp. 80–93, 2024, doi: 10.5937/vojtehg72-47242.

[25] D. Henderson, S. H. Jacobson, and A. W. Johnson, "The theory and practice of simulated annealing," in *Handbook of Metaheuristics*, Boston, MA, USA: Kluwer Academic Publishers, 2006, pp. 287–319. doi: 10.1007/0-306-48056-5_10.

[26] A. Kuznetsov *et al.*, "Generation of nonlinear substitutions by simulated annealing algorithm," *Information*, vol. 14, no. 5, p. 259, Apr. 2023, doi: 10.3390/info14050259.

[27] C. Pokhrel, R. Ghimire, B. R. Dawadi, and P. Manzoni, "A machine learning-based hybrid encryption approach for securing messages in software-defined networking," *Network*, vol. 5, no. 1, p. 8, Mar. 2025, doi: 10.3390/network5010008.

[28] M. Borges-Quintana, M. A. Borges-Trenard, O. Tito-Corrioso, O. Rojas, and G. Sosa-Gómez, "Combined and general methodologies of key space partition for the cryptanalysis of block ciphers," *Cryptography*, vol. 8, no. 4, p. 45, Oct. 2024, doi: 10.3390/cryptography8040045.

[29] D. Ramakrishna and M. Ali Shaik, "A comprehensive analysis of cryptographic algorithms: Evaluating security, efficiency, and future challenges," *IEEE Access*, vol. 13, pp. 11576–11593, 2025, doi: 10.1109/ACCESS.2024.3518533.

[30] M. D. Asrofa, S. Bahri, and K. Kasliono, "One-time pad cryptography for secure data transmission in IoT smart door using QR code," *J. Media Inf. Teknol.*, vol. 2, no. 2, pp. 133–148, Oct. 2025, doi: 10.69616/mit.v2i2.248.

[31] H. M. Bahig, M. A. G. Hazber, and H. M. Bahig, "An efficient simulated annealing algorithm for short addition sequences," *AIMS Math.*, vol. 9, no. 5, pp. 11024–11038, 2024, doi: 10.3934/math.2024540.

[32] D. E. Goldberg, *Genetic algorithms in search, optimization, and machine learning*, Boston, MA, USA: Addison-Wesley, 1989. doi: 10.5860/choice.27-0936.

[33] C. E. Shannon, "A mathematical theory of communication," *Bell Syst. Tech. J.*, vol. 27, no. 3, pp. 379–423, Jul. 1948, doi: 10.1002/j.1538-7305.1948.tb01338.x.

[34] A. Rukhin, J. Soto, and J. Nechvatal, "A statistical test suite for random and pseudorandom number generators for cryptographic applications," Nat. Inst. Stand. Technol., Gaithersburg, MD, USA, NIST SP 800-22 Rev. 1a, 2010. doi: 10.6028/NIST.SP.800-22r1a.

[35] G. S. Shyaa and M. Al-Zubaidie, "Utilizing trusted lightweight ciphers to support electronic-commerce transaction cryptography," *Appl. Sci.*, vol. 13, no. 12, p. 7085, Jun. 2023, doi: 10.3390/app13127085.