Metaheuristic-Based Optimization of Monitoring System Architectures*

Hanna Livinska[1,†], Serhii Kostrytsia[2,*,†]

[1] Taras Shevchenko National University of Kyiv, Volodymyrska Street 64/13, 01601 Kyiv, Ukraine
[2] Taras Shevchenko National University of Kyiv, Volodymyrska Street 64/13, 01601 Kyiv, Ukraine

Abstract

This paper presents an in-depth study of metaheuristic methods for finding optimal architectures in real-time monitoring systems. Motivated by the increasing complexity of industrial, energy, and other critical infrastructure domains, we focus on designing efficient monitoring solutions that balance responsiveness, cost, and reliability. After providing an overview of existing metaheuristics—including genetic algorithms, evolutionary strategies, ant colony optimization, and particle swarm optimization—we introduce a dynamic Social Spider Optimization (SSO) algorithm. This novel approach integrates adaptive global exploration with local refinement and is supported by mathematical formulations of objective functions, constraints, and convergence criteria. Comparative experiments and simulations in the energy sector demonstrate that SSO achieves near-optimal configurations more efficiently than traditional methods, confirming its practical effectiveness and applicability in real-time monitoring applications.

Keywords

monitoring, metaheuristics, system architecture, evolutionary algorithms, real-time optimization

1. Introduction

In the era of accelerating digital transformation and growing demands for sustainable development, the real-time monitoring of critical infrastructures has become a cornerstone for ensuring reliability, safety, and efficiency. Sectors such as energy, transportation, water supply, and industrial manufacturing are increasingly dependent on the prompt acquisition, processing, and analysis of large volumes of heterogeneous data. This dependency has been further intensified by geopolitical challenges — notably the war-induced disruptions in Ukraine's energy sector — which have revealed the vulnerability of traditional infrastructure systems and the urgent need for resilient monitoring frameworks.

Modern monitoring systems must therefore fulfill a wide range of requirements: high temporal resolution, accuracy, fault-tolerance, low-latency data transmission, and interoperability with heterogeneous networks. Traditional approaches to system monitoring — including threshold-based alarms or deterministic models — often lack flexibility and do not cope with complex, non-stationary processes. Moreover, the integration of renewable energy sources, distributed control, and edge computing in critical infrastructure adds layers of complexity, pushing the boundaries of conventional optimization methods.

To address these challenges, the focus of recent research has shifted toward advanced computational intelligence techniques, particularly metaheuristic algorithms. These methods — including Genetic Algorithms (GA), Particle Swarm Optimization (PSO), and Ant Colony Optimization (ACO) are well-suited to solving large-scale, multi-objective, and nonlinear optimization problems typical for modern monitoring systems. Their adaptability, population-based

*International Workshop on Computational Intelligence, co-located with the IV International Scientific Symposium "Intelligent Solutions" (IntSol-2025), May 01-05, 2025, Kyiv-Uzhhorod, Ukraine
1,* Corresponding author.
hanna.livinska@knu.ua (H. Livinska); serhii.kostrytsia@univ.net.ua (S. Kostrytsia)
0000-0001-9676-7932 (H. Livinska); 0000-0002-7729-2844 (S. Kostrytsia)

structure, and global search capabilities make them applicable for exploring optimal configurations of monitoring system (MS) architectures under uncertain or dynamic operating conditions.

Designing the architecture of a monitoring system is inherently a multi-criteria optimization problem. It involves balancing competing constraints such as system reliability, data throughput, latency, resource consumption, fault detection coverage, and adaptability to network changes. Conventional optimization techniques, such as exhaustive search or gradient-based methods, are often computationally infeasible or prone to convergence to local minima. Randomized search approaches, on the other hand, lack convergence guarantees. Metaheuristic methods bridge this gap by efficiently navigating the solution space and offering acceptable trade-offs between accuracy and computational cost. This work focuses on employing methods of metaheuristic, specifically Social Spider Optimization - to overcome these limitations.

In recent years, numerous studies have advanced the metaheuristic approach to the investigation of monitoring system architectures. For example, Nassef et al. (2023) ([9]) provide a comprehensive review of metaheuristic optimization algorithms for power systems problems, emphasizing the importance of adaptive parameter control to overcome local optima and ensure robust convergence. Similarly, Dutta and Mahanand ([10]) showed that applying metaheuristic approaches to energy-intensive routing in large-scale energy networks yields significant improvements in resource allocation, fault detection, and operational efficiency. These contributions underscore the potential of metaheuristic techniques to effectively address the complex, multi-objective nature of modern monitoring system design.

This study contributes to this evolving area by generalizing and applying metaheuristic approaches to the optimization of monitoring system architectures, with a focus on energy-critical infrastructures. Emphasis is placed on the formalization of the problem: definition of objective functions, constraints, and adaptive mechanisms for parameter control. Particular attention is given to multi-objective formulations, where trade-offs between performance, cost, and robustness must be handled simultaneously.

By using real-world monitoring requirements and drawing on recent methodological developments ([8]), this research offers a structured approach for the design of intelligent and resilient monitoring systems. The proposed framework applies to various energy domains, including smart grids, substation monitoring, and wide-area situational awareness systems (WAMS), where prompt and reliable data acquisition is critical for operational decision-making.

The paper is organized as follows. In Section 2, an overview of modern monitoring system architectures is presented, along with a formulation of the general optimization problems inherent to these architectures. Section 3 provides an overview of several metaheuristic approaches. In Section 4, the Social Spider Optimization (SSO) algorithm is described in detail. Section 5 shows the application of the SSO approach to finding an optimal architecture in the energy sector, highlighting its advantages. Finally, Section 6 concludes the article and outlines some ideas for future investigations.

2. Overview of Modern Monitoring System Architectures

2.1. Classification and Characteristics of Monitoring Systems

Monitoring is defined as “specially organized systematic automatic observation of the state of objects or processes in order to evaluate and forecast them” [3]. Modern monitoring systems are designed with a multi-tier or distributed architecture that caters to a wide range of application requirements. At their core, these systems consist of several fundamental layers:

- Data Acquisition: this layer includes various sensors, transducers, and intelligent agents that are deployed throughout the monitored facilities to capture real-time data.
- Communication Infrastructure: here, data is transmitted across networks—both wired (such as IP-based networks) and wireless (including technologies like Zigbee, LoRa, and LTE)—ensuring that information flows seamlessly throughout the system.

- Processing Nodes: these are the centralized servers, cloud platforms, or edge devices responsible for aggregating, filtering, and analyzing the collected data.
- Visualization and Storage: finally, dashboards, databases, and reporting systems are used to store data and provide actionable insights for decision-making and long-term diagnostics.

Depending on specific application constraints, monitoring system (MS) architectures can vary considerably. In some cases, hierarchical architecture is employed, where data flows from lower-level nodes to centralized processing centers—this is common in SCADA systems and wide-area monitoring systems (WAMS). Alternatively, clustered architecture utilizes redundant agents with partial data sharing between regions, thereby enhancing fault tolerance and resilience. There is also the option of fully distributed or hybrid architecture, which leverages edge computing to ensure low-latency responses and maintains local autonomy in the event of network failures.

Design decisions in monitoring systems are influenced by a variety of factors such as the topology of the network, the coordination among nodes, the volume of data, fault detection latency, real-time operational constraints, and security requirements [4]. For example, in power systems, the need to manage the rapid dynamics of voltage, current, and frequency necessitates a careful balance between responsiveness and robustness. Overall, the architecture of a monitoring system is a critical determinant of its effectiveness in capturing, processing, and reacting to data in real time, and the chosen design must align with both the operational needs and the constraints of the application environment.

2.2. The Optimization Tasks of Monitoring-System Architecture

The objective of this work is to develop an optimal configuration for monitoring system architecture. In this context, "architecture configuration" refers to the strategic placement of functional modules, data routes, redundancy schemes, and node roles to achieve a balanced performance. Specifically, the goal is to construct a monitoring system that minimizes a multi-criteria objective function, thereby achieving an optimal trade-off between performance (response time), cost, and reliability.

A general formulation is given by:

$$J(x) = \alpha \cdot T_{\text{resp}}(x) + \beta \cdot C_{\text{impl}}(x) + \gamma \cdot (1 - R_{\text{rel}}(x)), \tag{1}$$

where:

- x is a vector of architectural design parameters (e.g., number and position of sensors, number of relay nodes, topology type such as star, mesh or tree structures, redundancy level).
- $T_{resp}(x)$ is an average response time (in seconds).
- $C_{impl}(x)$ is the total implementation cost (e.g., equipment + deployment + maintenance).
- $R_{rel}(x) \in [0,1]$ is reliability (probability of uninterrupted operation over a fixed period).
- α, β, γ are user-defined coefficients reflecting the importance of time sensitivity, cost-efficiency, and robustness, respectively.

This solution must also satisfy certain constraints:

$$T_{\text{resp}}(x) \le T_{max}, C_{\text{impl}}(x) \le B_{\text{budget}}, R_{\text{rel}}(x) \ge R_{min}, \tag{2}$$

where:

- T_{max} is the maximum allowable response time (in seconds).
- B_{budget} is available budget.
- R_{min} is minimum required reliability.

This multi-objective optimization problem belongs to the class of NP-hard problems, that is, problems that are at least as difficult as the hardest problems in NP (nondeterministic polynomial time), for which no algorithm is known to solve all instances in polynomial time. Consequently, finding an exact solution for large instances is often computationally infeasible. The solution space is highly non-convex, often discrete, and characterized by conflicting objectives. Furthermore, interdependencies between system components (e.g., sensor placement and transmission delays) complicate analytical treatment.

As a result, classical optimization methods — such as linear programming or gradient descent — are insufficient for solving this task effectively. Random search methods, while easy to implement, provide no convergence guarantees. This motivates the use of metaheuristic algorithms, which offer a balanced trade-off between exploration and exploitation of the solution space.

In the following section, we describe the theoretical foundations of metaheuristic optimization and prove how they can be effectively applied to architecture-level research in real-time monitoring systems.

3. Metaheuristic Approaches: Overview and Potential

3.1. Main Classes of Metaheuristics

Metaheuristics are high-level search strategies designed to guide subordinate heuristics toward optimal or near-optimal solutions in complex optimization landscapes [1, 2]. Their advantage lies in flexibility, ease of hybridization, and the ability to escape local optima. The primary categories include:

- Evolutionary algorithms: Inspired by the principles of natural selection and genetics, these include Genetic Algorithms (GA), Evolutionary Strategies (ES), and Genetic Programming (GP). They operate on populations of solutions using crossover, mutation, and selection.
- Swarm intelligence algorithms: Based on the collective behavior of decentralized systems. Examples include Particle Swarm Optimization (PSO), Ant Colony Optimization (ACO), Artificial Bee Colony (ABC), and Social Spider Optimization (SSO).
- Trajectory-based methods: These are local search algorithms enhanced with mechanisms to escape local minimum, such as Simulated Annealing (SA) and Tabu Search.
- Hybrid or memetic algorithms: These combine global search strategies (e.g., evolutionary algorithms) with problem-specific local search procedures. Scatter Search and Memetic Algorithms are typical representatives.

Metaheuristics are particularly effective for monitoring system optimization due to the discrete nature of architecture parameters, the presence of multiple objectives, and the lack of closed-form analytical models.

3.2. Applied Method: Social Spider Optimization (SSO)

Social Spider Optimization (SSO), introduced by Cuevas et al. (2013) [7], is a population-based algorithm inspired by the cooperative behavior of social spiders. In SSO, the concept of transmitting vibrations through a communal web is used as an analogy for information sharing among candidate solutions, with each spider representing a potential monitoring system (MS) architecture. We describe this method in detail because it forms the core of approach, and its adaptive mechanisms play a crucial role in achieving improved performance.

Key operational principles of SSO include:

- Solution encoding: Each spider in the population encodes a candidate MS architecture — including sensor placement, communication routes, and redundancy schemes.

- Information sharing: Spiders share information about their fitness through vibrations; stronger solutions generate stronger signals.
- Gender-based behavior: The population is divided into male and female spiders, which influence exploration and exploitation dynamics differently.
- Adaptive attraction: The influence of neighbors is modulated by an attraction coefficient, which changes over time to balance global and local search.
- Stochastic perturbation: Mutation-like mechanisms help escape premature convergence and keep diversity.

While alternative metaheuristic approaches such as Genetic Algorithms, Particle Swarm Optimization, and Ant Colony Optimization also have their merits, they face limitations in this context. Evolutionary algorithms often require meticulous tuning of crossover and mutation parameters and can converge prematurely in complex, multimodal search spaces. Swarm intelligence methods, though computationally efficient in continuous domains, may struggle with the discrete and highly non-convex nature of MS architecture parameters. Trajectory-based methods like Simulated Annealing and Tabu Search excel in local search but are prone to getting trapped in local optima without adequate diversification. In contrast, SSO's combination of adaptive attraction and stochastic perturbation provides a balanced trade-off between exploration and exploitation, making it especially suitable for multi-objective, discrete optimization tasks in monitoring system design [1, 2, 3, 7].

In this study, we adapt SSO to monitor architecture optimization problems using a fitness function that integrates latency, cost, and reliability criteria, as defined in Equation (1). The following section presents the implementation details and simulation results, demonstrating the practical benefits of SSO-based approach in real-time monitoring environments.

4. Search for Optimal Monitoring System Architectures

4.1. Mathematical Formulation of the Problem

Let x be a vector of decision variables describing potential monitoring system architecture. These variables may include the number and position of servers, type of communication topology (e.g., mesh, star), node redundancy schemes, and allocation of processing tasks.

The goal is to minimize the following objective function:

$$\min_{x} \left[\alpha \cdot T_{\text{resp}}(x) + \beta \cdot C_{\text{impl}}(x) + \gamma \cdot \left(1 - R_{\text{rel}}(x)\right) \right], \tag{3}$$

under the constraints (2).

This formulation allows for a weighted trade-off between responsiveness, cost-efficiency, and reliability — the three cornerstones of critical infrastructure monitoring.

To handle constraint violations during the search process, a penalty function is incorporated into the fitness evaluation. This function penalizes any solution that violates one or more constraints, thereby discouraging infeasible configurations. A common formulation is:

$$P(x) = \lambda_1 \cdot max\left(0, T_{resp}(x) - T_{max}\right) + \lambda_2 \cdot max\left(0, C_{impl}(x) - B_{budget}\right) + \lambda_3 \cdot max\left(0, R_{min} - R\right. \tag{4}$$

where $\lambda_1, \lambda_2, \lambda_3$ are penalty coefficients that determine the severity of each constraint violation.

The penalty coefficients are tuned based on domain knowledge and experimental sensitivity analysis. Higher values enforce stricter constraint adherence and drive the search into feasible regions faster, while lower values allow more exploration but may lead to slower convergence. Striking the right balance is crucial to ensure that penalties meaningfully influence the fitness without dominating it.

The Social Spider Optimization (SSO) algorithm exhibits several advantages in terms of convergence speed when solving the optimization task (3). One of its key features is the adaptive

attraction coefficient, which is decreased over time. This dynamic adjustment allows the algorithm to transition smoothly from a broad, global exploration phase to a focused, local refinement phase. As a result, SSO is capable of rapidly honing in on promising regions of the search space.

Empirical observations have indicated that SSO often converges to near-optimal solutions in significantly fewer iterations compared to classical metaheuristic methods. For instance, when benchmarked against methods like Genetic Algorithms (GA) and Differential Evolution (DE), SSO achieved comparable or superior fitness values with approximately 25% fewer iterations. This efficiency is attributed to SSO's balanced use of global exploration—via information sharing among individuals—and local exploitation—through stochastic perturbations that help escape local optima.

In contrast, while Genetic Algorithms are robust in global exploration, they typically require careful tuning of crossover and mutation operators and may exhibit slower convergence in highly non-convex, discrete solution spaces. Differential Evolution, although effective in continuous domains, can struggle when the problem space involves complex constraints and mixed-variable types. Trajectory-based methods such as Hill Climbing, on the other hand, converge quickly but are prone to premature convergence, often getting trapped in suboptimal local minima.

Other alternatives include Particle Swarm Optimization (PSO) and Ant Colony Optimization (ACO). PSO can sometimes be effective but may also converge prematurely if the balance between exploration and exploitation is not well maintained. ACO is well-suited for discrete problems, yet its convergence speed and scalability may not match the adaptive mechanisms found in SSO.

Overall, the convergence speed of SSO is one of its strongest attributes not only accelerates the search process by quickly focusing on high-quality regions of the solution space but also maintains diversity through its stochastic perturbation, thereby reducing the risk of getting trapped in local optima. These features make SSO a particularly attractive choice for the complex, multi-objective optimization tasks inherent in designing optimal monitoring system architectures.

4.2. Developed Social-Spider-Based Algorithm

The SSO algorithm applied here follows an iterative process with the following steps:

1. Generate an initial population of spiders (candidate solutions):

$$\left\{x_1^{(0)},\ldots,x_N^{(0)}\right\}, \tag{5}$$

where:

- N is the number of individuals in the population.
- (0) is the initial generation.

2. For each spider, compute the penalized fitness function:

$$F\left(x_i^{(t)}\right)=J\left(x_i^{(t)}\right)+P\left(x_i^{(t)}\right), \tag{6}$$

where:

- $i\in\{1,2,\ldots,N\}$ – index of spider (solution candidate) in the population.
- $t\in N_0$, $N_0=\{0,1,2,\ldots\}$ - current iteration (generation).
- $J(x)$ – the original objective (1).
- $P(x)$ – a penalty function for constraint violations (4).

3. Update the position of each spider by moving it toward a more fit neighbor in the population. The movement is influenced by a time-dependent attraction factor and random noise, promoting both exploration and convergence:

$$x_i^{(t+1)} = x_i^{(t)} + \phi(t) \cdot \left(x_j^{(t)} - x_i^{(t)}\right) + \epsilon \ , \tag{7}$$

where:

- $x_j^{(t)}$ – neighboring spider with better fitness in the same generation.
- $\varepsilon \sim N(0, \sigma^2)$, σ^2 - variance of the normal distribution (e.g., 0.01).
- $\phi(t)$ – a time-dependent attraction coefficient, defined as:

$$\phi(t) = \phi_0 \cdot \left(1 - \frac{t}{T_{max}}\right), \tag{8}$$

where:

- ϕ_0 – initial attraction strength (e.g., 1.0).
- t – current iteration index.
- t_{max} – maximum number of iterations.
- $k > 0$ – decay control parameter (e.g., $k = 2$).

4. Accept the updated solution $x_i^{(t+1)}$ only if it improves the fitness value. That is, if $F\left(x_i^{(t+1)}\right) < F\left(x_i^{(t)}\right)$, then the new position replaces the current one: $x_i^{(t)} \leftarrow x_i^{(t+1)}$.
5. Stop the algorithm if the number of iterations reaches a fixed maximum $t = t_{max}$, or if the fitness improvement becomes smaller than a threshold δ for k consecutive iterations,

$$/ F\left(x_i^{(t)}\right) - F\left(x_i^{(t-1)}\right) / < \delta \ , \ \forall i \ , \text{for } k \text{ successive iterations} . \tag{9}$$

This method allows the algorithm to begin with broad exploration and gradually shift toward intensive local search, reducing the likelihood of premature convergence.

4.3. Energy-Sector Example: Substation Monitoring

To demonstrate the applicability of the proposed approach, we consider a case study of designing a monitoring system for a high-voltage substation compliant with the IEC 61850 standard [6]. The architectural optimization must satisfy the following conditions:

- Response time $T_{\text{resp}}(x) \leq 1 \text{second}$.
- Reliability $R_{\text{rel}}(x) \geq 0.999$.
- Total implementation cost $C_{impl}(x) \leq \$1{,}000{,}000$.

The SSO algorithm was executed with a population of 60 individuals. The attraction coefficient $\phi(t)$ (8) and mutation probability ε were dynamically decreased over iterations to enhance convergence stability. In 50 to 70 iterations, the method achieved a 5–15% improvement in the fitness score compared to baseline Genetic Algorithms and Simulated Annealing approaches.

The resulting architecture typically featured balanced distributions of processing units, optimized redundancy patterns, and topology structures that ensured both low latency and robust fault tolerance. This confirms the efficacy of SSO in practical, cost-constrained energy infrastructure applications.

5. Experimental and Model Results

5.1. Reference Scenarios and Parameter Settings

To evaluate the effectiveness of the proposed Social Spider Optimization (SSO)-based approach, a series of simulated scenarios were constructed, reflecting real-world applications in energy and industry. Each scenario posed specific constraints in terms of latency, reliability, and resource usage:

- Scenario 1: A networked production-trading cluster with a requirement for sub-second transaction monitoring, typical in supply chain logistics and industrial automation.
- Scenario 2: A high-voltage substation infrastructure following the IEC 61850 standard [6], requiring strict response time and reliability thresholds.
- Scenario 3: A cloud-oriented internal auditing system for enterprise environments, where computational efficiency and cost-awareness dominate.

To ensure comparability and repeatability, the following metaheuristic settings were used across all scenarios:

- Population size: $N = 60$.
- Maximum iterations: 100.
- Attraction coefficient $\phi(t)$: linearly decreased from 1 to 0.1.
- Mutation probability: decreased from 0.2 to 0.05 over iterations.

All algorithms were implemented in Python and executed under identical hardware conditions using simulated network models with constraint-aware cost functions.

5.2. Analysis and Comparison of Results

The experimental results indicate that the SSO algorithm converges significantly faster than both GA and DE. In resulted tests, SSO reached near-optimal fitness values in fewer iterations and with less fluctuation, reflecting a more stable convergence behavior.

While Hill Climbing sometimes converges quickly, its greedy approach often leads to premature convergence in local optima, limiting its performance over multiple runs. In contrast, the adaptive exploration and exploitation balance of SSO allows it to effectively navigate the complex, non-convex solution space. For example, on average, SSO required about 25% fewer iterations to achieve comparable or better fitness scores than GA and DE, as detailed in Table 1.

This suggests that SSO not only enhances solution quality but also offers a more efficient optimization process in terms of convergence speed.

Table 1
Typical Optimization Performance Metrics

Method	Mean $\acute{J}$	Best	Number of iterations to Convergence
GA	0.278 ± 0.026	0.244	~70
DE	0.252 ± 0.030	0.235	~60
HC	0.420 ± 0.045	0.380	~120
SSO	**0.231 ± 0.018**	**0.220**	**50–55**

As we can see, SSO algorithm consistently outperformed its counterparts across all scenarios. Notably, it converged faster and showed lower variance, indicating stable performance and reduced sensitivity to initial conditions. The adaptive attraction parameter, combined with a decaying mutation rate, enabled a balance between exploration and exploitation throughout the search process.

This behavior is particularly helpful in monitoring applications where both speed and reliability are critical. The resulting configurations were found to meet system-level constraints more consistently than the other tested methods.

6. Conclusions and Future Work

In this work, we propose a metaheuristic-based approach for optimizing the architecture of monitoring systems, with a particular focus on critical infrastructure applications where responsiveness, reliability, and cost-efficiency are crucial. Presented method begins by formulating a multi-objective optimization problem that captures key performance metrics such as system response time, implementation cost, and overall reliability. This formulation is carefully designed to incorporate practical constraints, ensuring that the resulting solutions are both feasible and effective in real-world scenarios.

Building on this formulation, we introduce a dynamic Social Spider Optimization (SSO) algorithm. Unlike conventional methods, SSO algorithm combines global exploration with targeted local refinement by adaptively adjusting attraction coefficients and controlling mutation. This innovative strategy enables the algorithm to navigate the highly non-convex, often discrete solution space efficiently. Through simulated experiments set in industrial and energy-sector scenarios, we observed that the SSO algorithm converges faster and achieves higher-quality solutions compared to traditional approaches like Genetic Algorithms, Differential Evolution, and Hill Climbing.

The robustness and flexibility of presented approach highlights its potential as a practical tool for intelligent system design in real-time monitoring environments. We are confident that this metaheuristic framework not only meets the demanding requirements of critical infrastructures but also offers significant improvements over existing methods.

Looking ahead, there are several promising directions for future research. One possibility is to hybridize presented approach with machine learning model-using surrogate models to approximate the fitness function could reduce evaluation times, particularly in large-scale problems. Additionally, adopting advanced multi-objective methods, such as NSGA-II or SPEA2, could help in obtaining a Pareto front of non-dominated solutions, thereby enabling a more detailed trade-off analysis between conflicting objectives. We also see great potential in integrating presented optimization process with digital twins, which would allow for online testing and adaptive reconfiguration of monitoring systems based on real-time data. Finally, validating presented approach to larger datasets and across a broader range of industrial, smart-grid, and cloud-based scenarios would further demonstrate its scalability and practical applicability.

Overall, these enhancements promise to further expand the applicability of metaheuristic optimization in designing modern, intelligent monitoring architectures, paving the way for more efficient, robust, and cost-effective systems.

Declaration on Generative AI

The authors have not employed any Generative AI tools.

References

[1] S. Luke, Essentials of Metaheuristics, 2nd ed., Lulu, 2013. Available at: http://cs.gmu.edu/~sean/book/metaheuristics/

[2] I.V. Kozin, Ye.K. Selyutin, Classification of metaheuristic algorithms. In: Proceedings of the Conference "Mathematical Methods, Models and Information Technologies in Economics", Kyiv, 2023, pp. 31–34.
[3] T. Neskorodieva, Ye. Fedorov, Yu. Antonov, A. Neskorodieva, Methods and algorithms for detecting deviations in control and audit systems of information security. In: Proceedings of the International Scientific-Practical Conference "Topical Issues and Prospects of Development of Accounting, Control, Audit and Taxation in the Digital Economy", Kyiv, 2023, pp. 22–25.
[4] I.H. Falkovskyi, Monitoring systems in power engineering. In: Monitoring of Complex Energy Systems: Monograph, Kyiv, 2019, pp. 10–25.
[5] V. Slipchenko, V. Mamalyha, L. Polyahushko, Monitoring of CEESM: concept, methods, implementation. In: Analytical Models of Ecological-Economic-Energy Monitoring, Kyiv, 2021, pp. 45–67.
[6] M. Sopel, Monitoring in Power Engineering, Doctoral dissertation, Institute of Electrodynamics of the National Academy of Sciences of Ukraine, Kyiv, 2015.
[7] Cuevas E. and Cienfuegos M., *A new algorithm inspired in the behavior of the social-spider for constrained optimization*, *Expert Systems with Applications*, 2014, Vol. 41, No. 2, pp. 412–425. DOI: 10.1016/j.eswa.2013.07.067.
[8] G. Phadke and J. S. Thorp, *Computer Relaying for Power Systems*, 2nd ed., Wiley, Hoboken, NJ, 2009.
[9] A. M. Nassef, M. A. Abdelkareem, H. M. Maghrabie, A. Baroutaji, Metaheuristic optimization algorithms for power systems problems, *Sustainability* 15 (2023) 9434. doi:10.3390/su15129434.
[10] P. Dutta, B. S. Mahanand, Affordable energy-intensive routing using metaheuristics, in: *Cognitive Big Data Intelligence with a Metaheuristic Approach*, Elsevier, 2022, Chapter 9. doi:10.1016/B978-0-323-85117-6.00012-4.