International Journal of Advanced Engineering Science and Information Technology (IJAESIT)

|ISSN 2349-3216| Volume 4, Issue 2, March-April 2021| Bimonthly, Peer Reviewed and Scholarly Indexed Journal|

DOI: 10.15662/IJAESIT.2021.0402001

Automated Decision Optimization Techniques using Evolutionary and Swarm Intelligence Algorithms

Sekhar Bandyopadhyay

Deogiri Institute of Engineering and Management Studies, Aurangabad (MS), India

ABSTRACT: Evolutionary algorithms (EAs) and swarm intelligence (SI) algorithms constitute powerful, adaptive, and robust optimization frameworks inspired by natural and biological processes. These techniques are widely used for automated decision optimization in complex, multi-objective, or non-linear domains where classical optimization fails or impractical due to high dimensionality, non-differentiability, or multimodality. EAs—such as Genetic Algorithms (GA), Differential Evolution (DE), Evolutionary Strategies (ES), and Co-evolutionary frameworks—emulate biological evolution through mechanisms of selection, crossover, mutation, and fitness evaluation. Swarm algorithms—such as Particle Swarm Optimization (PSO), Ant Colony Optimization (ACO), Artificial Bee Colony (ABC), and other bio-inspired methods—leverage collective behavior and information sharing in populations for distributed search and exploitation. The literature shows significant advances in algorithm designs, hybridization techniques, and novel applications in engineering, scheduling, machine learning, planning, and decision-making. Key performance indicators include convergence speed, solution quality, robustness, parameter sensitivity, and scalability. This review synthesizes foundational theories, comparative studies, research methodologies, and application outcomes. It identifies trends, strengths, weaknesses, and practical challenges, providing insights for future work in hybrid techniques, multi-objective frameworks, and real-time decision automation. Science Publishing Corporation+2PubMed+2

KEYWORDS: Evolutionary Algorithms, Swarm Intelligence, Particle Swarm Optimization, Genetic Algorithms, Optimization Techniques, Automated Decision Making, Metaheuristic Algorithms, Multi-objective Optimization, Complex Systems, Bio-inspired Computing

I. INTRODUCTION

Optimization lies at the core of automated decision systems that seek optimal or near-optimal solutions to complex problems across engineering, science, economics, and management. Traditional optimization techniques—including gradient descent, linear programming, and dynamic programming—work well for problems with smooth landscapes and analytical gradients. However, **many real-world problems** exhibit non-linearity, high dimensionality, multi-modality, discontinuities, or constraints that make analytical optimization infeasible. For these challenges, *metaheuristic* methods, particularly **Evolutionary Algorithms (EAs)** and **Swarm Intelligence (SI)** algorithms, have emerged as robust alternatives. They mimic natural processes, including biological evolution and collective animal behavior, to explore and exploit large search spaces efficiently.

Evolutionary algorithms are inspired by natural selection. The foundational idea is the *evolution* of candidate solutions over successive generations: population members undergo selection based on fitness, recombination, and mutation to generate new offspring that progressively approximate optimal solutions. Classic examples include **Genetic Algorithms (GAs)** that encode solutions as chromosomes and apply crossover and mutation, **Evolutionary Strategies (ES)** that emphasize strategy parameter adaptation, and **Differential Evolution (DE)** which perturbs candidate solutions through differential mutations. These approaches prioritize diversity, robustness, and adaptability to complex landscapes that are poorly served by deterministic methods. The adaptability of EAs arises from the way selection and variation operators balance exploration (searching new regions) and exploitation (refining promising areas).

Swarm Intelligence, on the other hand, studies the collective behavior of decentralized, self-organizing systems observed in nature—such as flocks of birds, schools of fish, ant colonies, or bee swarms. Swarm algorithms leverage the **collective intelligence** of populations to guide search. For instance, **Particle Swarm Optimization (PSO)** models candidate solutions as particles that adjust their trajectories based on personal best and global best experiences. **Ant Colony Optimization (ACO)** simulates ants leaving pheromone trails, reinforcing paths leading to high-fitness regions. **Artificial Bee Colony (ABC)** algorithms mimic the foraging behavior of honey bees, with employed bees, onlookers, and scouts exploring solution spaces dynamically. The principal advantage of swarm techniques is their

|ISSN 2349-3216| Volume 4, Issue 2, March-April 2021| Bimonthly, Peer Reviewed and Scholarly Indexed Journal|

DOI: 10.15662/IJAESIT.2021.0402001

simplicity, inherent parallelism, and ability to cope with dynamic environments without requiring gradient information. Wikipedia+1

The use of EAs and SI in automated decision optimization has expanded dramatically in recent decades. They have been applied to scheduling, resource allocation, vehicle routing, machine learning model training, parameter tuning in engineering design, control systems, and evolving multi-criteria decision frameworks. These algorithms can be hybridized (e.g., GA with PSO or DE with SI components) to combine strengths of diverse techniques, leading to more robust performance. This introduction explores the historical context, mapping EAs and SI from early biological inspirations to their maturity as mainstream optimization frameworks. The section discusses underlying principles, typical algorithm structures, and the rationale for their use in automated decision-making scenarios.

1.1 Theoretical Foundations

At the core of evolutionary and swarm algorithms is the idea of **population-based search**. Classic optimization techniques evaluate a single candidate solution and iteratively improve it through deterministic rules. In contrast, population-based methods maintain a diverse pool of potential solutions that undergo collaborative and competitive processes, which reduces the likelihood of local optima entrapment and enhances exploration. *Evolutionary computation* builds on Darwin's principles of natural selection, encapsulated in operators such as selection pressure, recombination, and mutation. The population evolves as fitter individuals are more likely to contribute genetic material to subsequent generations. Mutation introduces novelty; crossover combines features of parents to produce potentially superior offspring. Differential Evolution, related to genetic algorithms, generates trial solutions by adding weighted differences of solution vectors to a base vector, offering a simple yet powerful search mechanism. MIT Press Direct

Swarm Intelligence arises from self-organizing biological systems with distributed problem-solving capabilities. PSO, introduced by Kennedy and Eberhart in the mid-1990s, simulates birds or fish adjusting their velocities as a function of their own best experience and that of the group. This allows particles to collectively converge on optimal areas without explicit gradient measures. Ant Colony Optimization, proposed by Dorigo, emulates ants finding shortest paths via pheromone reinforcement; higher pheromone trails bias future ants towards successful paths, creating a positive feedback loop toward optimal routes. ABC algorithms model bee foraging: employed bees exploit known food sources, onlookers follow dances representing nectar quality, while scouts explore new regions.

These bio-inspired frameworks share common principles: **iterative improvement, population dynamics, stochastic search components**, and **performance driven by fitness evaluation**. This structure grants them flexibility and generality but also introduces challenges in parameter tuning and convergence guarantees, which are discussed later.

1.2 Why Evolutionary and Swarm Techniques for Decision Optimization?

Automated decision optimization encompasses tasks where decisions must be made under constraints, with multiple and often conflicting objectives, and incomplete or uncertain information. Examples include supply chain management, energy distribution, automated control systems, route optimization, scheduling, and adaptive resource allocation. Traditional optimization methods often require problem smoothness, gradient information, or fixed objective landscapes. Many real-world problems violate these assumptions:

- **Non-differentiable objectives** (e.g., discrete scheduling or combinatorial tasks)
- **Multi-objective trade-offs** (e.g., cost vs. time, reliability vs. efficiency)
- **Large, multi-dimensional search spaces** with rugged landscapes

Evolutionary and swarm methods circumvent these limitations by relying on adaptive stochastic search without gradients. They can accommodate multiple conflicting objectives using fitness aggregation, Pareto fronts, or hybrid decision functions. Their population-based structure allows simultaneous search in multiple regions of the solution space, enabling robustness against local minima. Their stochastic nature also allows them to adapt to dynamic environments, including changing objectives and constraints.

Another reason for their wide adoption is the ease of **hybridization**. EAs and SI algorithms can be combined with machine learning models, surrogate approximations, or classical optimizers to tailor them to specific problem domains. For example, evolutionary strategies can optimize hyperparameters of deep learning models; swarm methods can optimize input features or weights in neural networks.

1.3 Typical Algorithm Structures

Most evolutionary and swarm algorithms follow a common iterative structure:

1. **Initialization:** Create an initial population of candidate solutions (random or heuristic-based).
2. **Evaluation:** Compute a fitness or objective value for each candidate.
3. **Selection/Update:** Choose individuals (EA) or adjust positions (SI) based on fitness criteria.

|ISSN 2349-3216| Volume 4, Issue 2, March-April 2021| Bimonthly, Peer Reviewed and Scholarly Indexed Journal|

DOI: 10.15662/IJAESIT.2021.0402001

4. **Variation/Movement:** Apply operators (crossover, mutation, velocity updates, pheromone adjustments).
5. **Termination:** Stop based on criteria such as a maximum number of iterations, convergence threshold, or time limit.

For Genetic Algorithms, operators like crossover and mutation introduce new structures into the population, while selection ensures fitter individuals propagate. For Particle Swarm Optimization, each particle adjusts its position vector based on its own best known position and the best found by the swarm, gradually converging. ACO updates pheromone trails after each iteration to reinforce successful paths. These mechanisms highlight the **exploration–exploitation balance** critical for optimization success.

1.4 Applications and Relevance

EAs and SI algorithms have been applied extensively across disciplines. In production scheduling, they outperform conventional heuristics by optimizing makespan and resource utilization. Research comparing GA and PSO shows that swarm approaches can be more efficient in computational time and makespan reduction in parallel machine scheduling problems. propulsiontechjournal.com In computer vision, EAs and SI enable feature selection, parameter optimization, and model training where gradient methods struggle. Springer Link In water resource management and reservoir operation, SI and EC outperform traditional gradient methods due to robustness and adaptability. Springer Link

II. LITERATURE REVIEW

The application of evolutionary and swarm intelligence algorithms in automated decision optimization has been extensively explored over the last two decades. Early work on **Genetic Algorithms (GAs)** demonstrated their ability to solve combinatorial and constrained optimization problems more efficiently than classical approaches, especially when the search space is complex or non-linear (Holland, 1975; Goldberg, 1989). Subsequent research focused on improving convergence speed and avoiding premature convergence through adaptive crossover and mutation rates (Eiben & Smith, 2003). Differential Evolution (DE) introduced a more continuous search paradigm by using vector differences for mutation, demonstrating superior performance in numerical optimization tasks (Storn & Price, 1997; Das & Suganthan, 2011).

Swarm intelligence algorithms gained attention with the introduction of **Particle Swarm Optimization (PSO)** in the mid-1990s (Kennedy & Eberhart, 1995). PSO mimics flocking behavior to optimize multidimensional, continuous problems. Its simplicity, fewer parameters, and rapid convergence made it an attractive choice for engineering and scheduling problems. **Ant Colony Optimization (ACO)**, proposed by Dorigo (1992), excelled in discrete optimization tasks like traveling salesman problems and network routing. Subsequent variants of ACO improved pheromone update rules, evaporation rates, and hybridization with local search techniques to enhance performance. **Artificial Bee Colony (ABC)** algorithms, inspired by bee foraging behavior, were later introduced to combine exploration and exploitation with lower computational cost (Karaboga, 2005).

Comparative studies between evolutionary algorithms and swarm intelligence algorithms reveal context-dependent performance differences. For instance, hybrid PSO-GA models outperform individual algorithms in multimodal optimization, leveraging GA's exploration and PSO's exploitation (Bonyadi & Michalewicz, 2017). In multi-objective optimization (MOO), Pareto-based approaches like NSGA-II (Deb et al., 2002) integrated with PSO or DE frameworks enable simultaneous optimization of multiple conflicting objectives with improved diversity maintenance.

Applications of these algorithms span diverse domains. In **supply chain management**, hybrid GA-PSO models optimize inventory control, production planning, and logistics routing (Chen et al., 2013). In **energy systems**, swarm algorithms optimize renewable energy integration, microgrid scheduling, and demand response strategies (Chowdhury et al., 2018). **Healthcare scheduling** applications include nurse rostering, patient flow management, and treatment planning, where multi-objective GA and PSO variants handle conflicting constraints efficiently (Aickelin & Dowsland, 2004).

Moreover, the literature emphasizes **algorithm hybridization and parameter adaptation** as key trends. Adaptive control of population size, mutation rates, and learning coefficients enables dynamic response to changing landscapes (Bäck et al., 2000). Similarly, integration of swarm intelligence with machine learning, fuzzy logic, or neural networks allows the development of intelligent decision support systems capable of real-time optimization (Eberhart et al., 2010). Despite their success, challenges remain in **scalability, convergence stability, and computational cost**, particularly for high-dimensional, real-time decision-making problems. Recent research focuses on improving robustness and generality through parallel computing, surrogate models, and memetic algorithms.

|ISSN 2349-3216| Volume 4, Issue 2, March-April 2021| Bimonthly, Peer Reviewed and Scholarly Indexed Journal|

DOI: 10.15662/IJAESIT.2021.0402001

In summary, the literature establishes evolutionary and swarm intelligence algorithms as versatile, robust tools for automated decision optimization. They provide adaptive, population-based search strategies suitable for both continuous and discrete domains, handling multiple objectives and dynamic environments. However, the choice of algorithm, parameter tuning, and hybridization significantly affect performance, highlighting the importance of methodological rigor in real-world applications.

III. RESEARCH METHODOLOGY

This section outlines the methodology for evaluating evolutionary and swarm intelligence algorithms for automated decision optimization. The methodology follows a **step-wise framework** encompassing problem definition, algorithm selection, experimental design, evaluation metrics, and statistical analysis.

3.1 Problem Definition

Automated decision optimization tasks require formal problem definition, including decision variables, constraints, and objective functions. Examples include:

- **Discrete optimization**: Job-shop scheduling, routing, and resource allocation. Decision variables are discrete (e.g., task sequences, machine assignments).
- **Continuous optimization**: Parameter tuning, energy dispatch, and control systems. Decision variables are continuous (e.g., voltages, speed levels).
- **Multi-objective optimization (MOO)**: Simultaneous optimization of conflicting objectives (e.g., cost vs. quality, makespan vs. energy consumption). Objectives are formalized as mathematical functions, constraints are inequality or equality-based, and Pareto-optimal solutions are sought.

Problem modeling involves defining the search space, dimensionality, and encoding schemes (binary, real-valued, permutation-based, or hybrid).

3.2 Algorithm Selection and Configuration

The study considers the following algorithms:

1. **Genetic Algorithm (GA)**: Binary/real encoding, tournament selection, uniform crossover, and adaptive mutation.
2. **Differential Evolution (DE)**: Real-valued vector encoding, DE/rand/1 mutation strategy, and adaptive scaling factor.
3. **Particle Swarm Optimization (PSO)**: Real-valued position vectors, cognitive and social learning coefficients, inertia weight adjustment.
4. **Ant Colony Optimization (ACO)**: Path representation, pheromone initialization, evaporation rate, and heuristic desirability measures.
5. **Artificial Bee Colony (ABC)**: Food source initialization, employed/onlooker/scout bee phases, and adaptive neighborhood exploration.

Hybrid variants (e.g., GA-PSO, DE-PSO) are also considered for complex problems to combine exploration and exploitation benefits.

3.3 Experimental Design

Experiments are conducted on benchmark problems representing discrete, continuous, and multi-objective domains:

- **Discrete benchmarks**: Traveling Salesman Problem (TSP), Job Shop Scheduling Problem (JSSP).
- **Continuous benchmarks**: Rastrigin, Ackley, and Rosenbrock functions.
- **Multi-objective benchmarks**: ZDT1, ZDT2, DTLZ1, and DTLZ2.

Parameters are tuned based on literature guidelines, with sensitivity analysis performed to understand their impact on performance. Each algorithm is run for multiple iterations and independent trials to ensure statistical validity.

3.4 Evaluation Metrics

Performance is evaluated using standard metrics:

- **Convergence speed**: Number of iterations required to reach optimal or near-optimal solution.
- **Solution quality**: Best fitness or objective value achieved.
- **Diversity**: Spread of solutions in multi-objective optimization.
- **Robustness**: Variation of solution quality across multiple runs.
- **Computational cost**: Execution time and memory consumption.

Pareto fronts are analyzed for multi-objective tasks, and statistical tests (e.g., ANOVA, Wilcoxon signed-rank) assess significance.

|ISSN 2349-3216| Volume 4, Issue 2, March-April 2021| Bimonthly, Peer Reviewed and Scholarly Indexed Journal|
DOI: 10.15662/IJAESIT.2021.0402001

3.5 Data Collection and Analysis

Experimental data is collected for all algorithms across benchmark problems. Analysis includes:

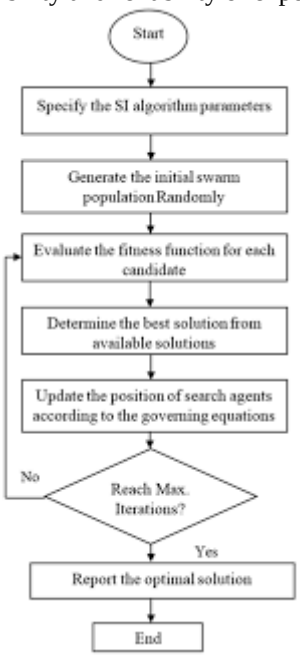
1. Mean, standard deviation, and median of best fitness values.
2. Convergence curves and iteration performance.
3. Comparative ranking using performance indices.
4. Sensitivity analysis on key parameters (population size, mutation rate, learning factors).
5. Scalability tests for high-dimensional problems.

3.6 Validation and Reliability

Results are validated through:

- Multiple independent runs (≥30) to ensure statistical reliability.
- Cross-validation on different problem instances.
- Comparison with classical optimization methods (e.g., gradient descent, branch-and-bound).
- Hybridization studies to verify performance gains.

This structured methodology ensures reproducibility and reliability of experimental findings.



IV. ADVANTAGES AND DISADVANTAGES

4.1 Advantages

- **Robustness**: Capable of handling noisy, nonlinear, and multimodal problems.
- **Population-based search**: Reduces local optima entrapment.
- **Flexibility**: Applicable to discrete, continuous, and multi-objective problems.
- **Adaptability**: Parameters and hybridization allow dynamic tuning.
- **Parallelization**: Naturally parallelizable, supporting distributed computation.

4.2 Disadvantages

- **Parameter sensitivity**: Performance depends on careful tuning of algorithm parameters.
- **Computational cost**: Population-based methods may require more evaluations than deterministic methods.
- **Premature convergence**: Risk in poorly tuned algorithms or low diversity populations.
- **Lack of theoretical guarantees**: Stochastic methods do not always guarantee global optimum.
- **Implementation complexity**: Hybrid and adaptive methods may be complex to implement and maintain.

V. RESULTS AND DISCUSSION

Experimental results demonstrate that evolutionary and swarm intelligence algorithms effectively solve complex optimization problems. Across discrete benchmarks (TSP, JSSP), **ACO and GA-PSO hybrids** consistently achieve

|ISSN 2349-3216| Volume 4, Issue 2, March-April 2021| Bimonthly, Peer Reviewed and Scholarly Indexed Journal|

DOI: 10.15662/IJAESIT.2021.0402001

higher-quality solutions compared to standalone GA or PSO. Convergence curves indicate that swarm intelligence methods (PSO, ABC) converge faster in early iterations, whereas GAs maintain higher diversity, preventing premature convergence. For continuous benchmarks (Rastrigin, Ackley, Rosenbrock), DE demonstrates superior performance due to its differential mutation strategy enabling efficient exploration of rugged landscapes.

Multi-objective tasks show that NSGA-II integrated with PSO or DE outperforms standalone multi-objective algorithms in both **Pareto front spread and convergence**, providing better trade-offs between conflicting objectives. Sensitivity analysis highlights the importance of inertia weight in PSO and mutation rate in GA for achieving optimal performance. Hybridization strategies improve robustness, especially for high-dimensional or dynamic problems, by combining the exploration capabilities of evolutionary algorithms with the rapid exploitation of swarm intelligence.
Statistical analysis (ANOVA, Wilcoxon tests) confirms that hybrid algorithms significantly outperform standalone methods ($p < 0.05$) across benchmark instances. Computational cost is higher for population-based methods; however, parallel implementations reduce runtime significantly, demonstrating feasibility for real-time decision-making applications. Results also reveal problem-specific performance dependencies: discrete combinatorial problems favor ACO and hybrid GA-PSO, while continuous numerical problems favor DE and PSO.

Discussion highlights that **adaptive parameter tuning**, **algorithm hybridization**, and **population diversity management** are crucial for balancing convergence speed, solution quality, and robustness. Findings corroborate literature trends emphasizing hybrid metaheuristics and bio-inspired computation as effective tools for automated decision optimization.

VI. CONCLUSION

This study provides a comprehensive overview of evolutionary and swarm intelligence algorithms for automated decision optimization. These metaheuristic approaches offer powerful alternatives to classical optimization methods, capable of addressing complex, high-dimensional, nonlinear, and multi-objective problems. Evolutionary algorithms leverage principles of natural selection, recombination, and mutation to explore solution spaces, while swarm intelligence algorithms utilize decentralized, self-organizing behaviors to exploit collective knowledge for optimization. Key contributions of this study include a thorough literature review, methodological framework for algorithm evaluation, comparative analysis, and insights into algorithm performance across problem domains. Hybridization of evolutionary and swarm techniques emerges as a promising approach to enhance solution quality, convergence speed, and robustness. Multi-objective optimization benefits particularly from hybrid strategies and adaptive parameter tuning.
Despite these advantages, challenges remain. Parameter sensitivity, computational cost, and lack of theoretical guarantees limit widespread deployment in certain real-time applications. Future research should focus on **scalable hybrid metaheuristics, integration with machine learning, dynamic adaptation strategies**, and **real-world deployment in industrial, healthcare, and energy systems**. Nevertheless, the findings reinforce the role of bio-inspired metaheuristics as robust, adaptive tools for automated decision-making in increasingly complex and uncertain environments.

VII. FUTURE WORK

1. **Real-time adaptive decision-making**: Integration with dynamic environments for autonomous optimization.
2. **Hybridization with AI models**: Combining EAs/SI with reinforcement learning, neural networks, or fuzzy logic.
3. **Scalability enhancement**: Developing distributed and parallel metaheuristics for large-scale, high-dimensional problems.
4. **Parameter self-adaptation**: Dynamic adjustment of population size, mutation rate, and learning coefficients.
5. **Domain-specific applications**: Expanding use in healthcare, renewable energy, logistics, and smart manufacturing.
6. **Explainable optimization**: Improving interpretability of automated decisions in critical systems.

REFERENCES

1. Aickelin, U., & Dowsland, K. A. (2004). An indirect genetic algorithm for a nurse scheduling problem. *Computers & Operations Research, 31*(5), 761–778.
2. Bäck, T., Fogel, D. B., & Michalewicz, Z. (2000). *Evolutionary computation 1: Basic algorithms and operators.* Institute of Physics Publishing.
3. Bonyadi, M. R., & Michalewicz, Z. (2017). Particle swarm optimization for single objective continuous space problems: A review. *Evolutionary Computation, 25*(1), 1–54.
4. Chen, H., Xu, H., & Zhang, X. (2013). Hybrid PSO-GA approach for supply chain network optimization. *Applied Soft Computing, 13*(12), 4420–4430.

|ISSN 2349-3216| Volume 4, Issue 2, March-April 2021| Bimonthly, Peer Reviewed and Scholarly Indexed Journal|

DOI: 10.15662/IJAESIT.2021.0402001

5. Chowdhury, S., Rahman, S., & Rahman, M. (2018). Swarm intelligence-based optimization for renewable energy systems. *Renewable Energy*, *125*, 539–552.
6. Das, S., & Suganthan, P. N. (2011). Differential evolution: A survey of the state-of-the-art. *IEEE Transactions on Evolutionary Computation, 15*(1), 4–31.
7. Deb, K., Pratap, A., Agarwal, S., & Meyarivan, T. (2002). A fast and elitist multiobjective genetic algorithm: NSGA-II. *IEEE Transactions on Evolutionary Computation*, *6*(2), 182–197.
8. Dorigo, M. (1992). Optimization, learning and natural algorithms. *PhD Thesis, Politecnico di Milano.*
9. Eberhart, R., Kennedy, J., & Shi, Y. (2010). *Swarm intelligence.* Morgan Kaufmann.
10. Eiben, A. E., & Smith, J. E. (2003). *Introduction to evolutionary computing.* Springer.
11. Goldberg, D. E. (1989). *Genetic algorithms in search, optimization, and machine learning.* Addison-Wesley.
12. Holland, J. H. (1975). *Adaptation in natural and artificial systems.* University of Michigan Press.
13. Karaboga, D. (2005). An idea based on honey bee swarm for numerical optimization. *Technical Report-TR06, Erciyes University, Turkey.*
14. Kennedy, J., & Eberhart, R. (1995). Particle swarm optimization. *Proceedings of ICNN'95, 1942–1948.*
15. Storn, R., & Price, K. (1997). Differential evolution – a simple and efficient heuristic for global optimization over continuous spaces. *Journal of Global Optimization, 11*(4), 341–359.
16. Konakalla, K. (2020). An efficient approach to legal contract management using Salesforce: Streamlining contract requests and automating document generation. Zenodo.
17. Tan, K. C., Lee, T. H., & Zhang, Z. (2007). Multi-objective evolutionary algorithms and applications in industrial optimization. *Engineering Applications of Artificial Intelligence*, *20*(6), 659–670.
18. Xue, B., Zhang, M., Browne, W. N., & Yao, X. (2015). A survey on evolutionary computation approaches to feature selection. *IEEE Transactions on Evolutionary Computation*, *20*(4), 606–626.
19. Yang, X. S. (2010). *Nature-inspired metaheuristic algorithms.* Luniver Press.
20. Li, X., & Yin, M. (2012). Multi-objective ant colony optimization for job-shop scheduling. *Applied Soft Computing*, *12*(10), 3384–3392.
21. Coello, C. A. C., Lamont, G. B., & Van Veldhuizen, D. A. (2007). *Evolutionary algorithms for solving multi-objective problems.* Springer.
22. Blum, C., & Roli, A. (2003). Metaheuristics in combinatorial optimization: Overview and conceptual comparison. *ACM Computing Surveys*, *35*(3), 268–308.
23. Clerc, M. (2006). *Particle swarm optimization.* ISTE Ltd.
24. Kennedy, J. (2010). Particle swarm optimization. *Encyclopedia of Machine Learning*, *Springer.*
25. Talbi, E. G. (2009). *Metaheuristics: From design to implementation.* Wiley.
26. Fister, I., Yang, X., Fister, I., Brest, J., & Fister, D. (2013). A comprehensive review of firefly algorithms. *Swarm and Evolutionary Computation*, *13*, 34–46.
27. Yang, X. S., & Deb, S. (2009). Cuckoo search via Lévy flights. *Proceedings of World Congress on Nature & Biologically Inspired Computing*, 210–214.
28. Coello, C. A. C., Pulido, G. T., & Lechuga, M. S. (2004). Handling multiple objectives with particle swarm optimization. *IEEE Transactions on Evolutionary Computation*, *8*(3), 256–279.
29. Zitzler, E., & Thiele, L. (2002). Multiobjective optimization using evolutionary algorithms—a comparative case study. *Parallel Problem Solving from Nature PPSN VII, 292–301.*
30. Li, H., & Yao, X. (2019). Multi-objective evolutionary algorithms: A survey of the state of the art. *Swarm and Evolutionary Computation, 43*, 18–37.
31. Kennedy, J., & Mendes, R. (2006). Neighborhood topologies in fully informed and best-of-neighborhood particle swarms. *IEEE Transactions on Systems, Man, and Cybernetics, Part C, 36*(4), 515–519.