DOI: http://dx.doi.org/10.26483/ijarcs.v15i1.7037

ISSN No. 0976-5697

Volume 15, No. 1, January-February 2024

International Journal of Advanced Research in Computer Science

REVIEW ARTICLE

Available Online at www.ijarcs.info

A Survey on Optimization Algorithms: Challenges and Future Opportunities

Subash Kumar, Sikander Singh Cheema
Department of Computer Science and Engineering,
Punjabi University, Patiala,India

Abstract: This paper conducts a comprehensive exploration of contemporary optimization algorithms, addressing challenges and outlining potential avenues for future research. The survey encompasses a wide spectrum of optimization techniques employed in various domains, ranging from mathematical programming to machine learning and artificial intelligence. It systematically analyses the inherent challenges faced by existing algorithms, including scalability issues, convergence speed, and adaptability to diverse problem spaces. Furthermore, the paper critically examines the impact of optimization algorithms on real-world applications, considering their effectiveness and limitations. The survey identifies emerging trends, such as hybrid approaches and metaheuristic methods that offer promising directions for overcoming current challenges. By synthesizing the state-of-the-art in optimization algorithms, this paper provides a valuable resource for researchers, practitioners, and decision-makers, guiding them towards addressing existing limitations and unlocking new opportunities in the evolving landscape of optimization research.

Keywords: Optimization; Artificial Intelligence; PSO; GA

1. Introduction

In recent years, data analysis of real-life applications has posed some critical challenges due to advances in internet technology and multimedia tools. These challenges include extracting information from a wide data set, analysing data from various dimensions, categorizing data, and summing up the data relationship. In order to address these challenges, data mining is a preferable technique. Generally speaking, data mining techniques are classified into two categories, such as direct and indirect data mining. Figure 1 shows the classification of data mining techniques.

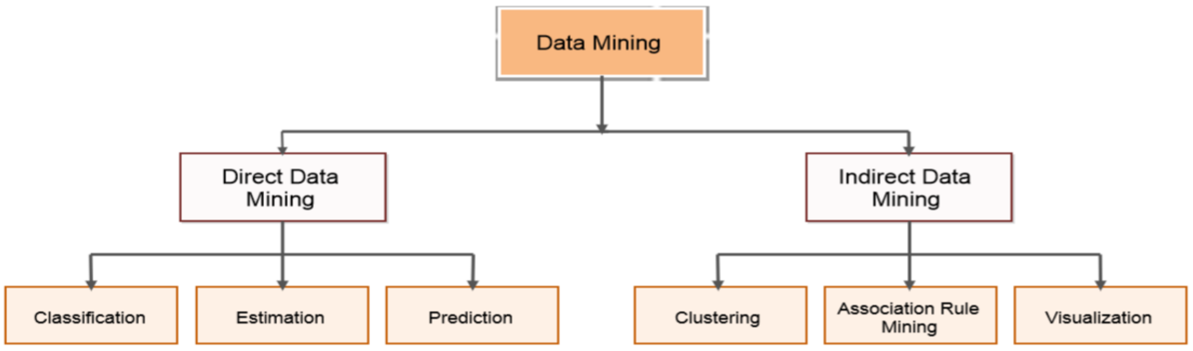

Figure 1: Classification of Data Mining Technique

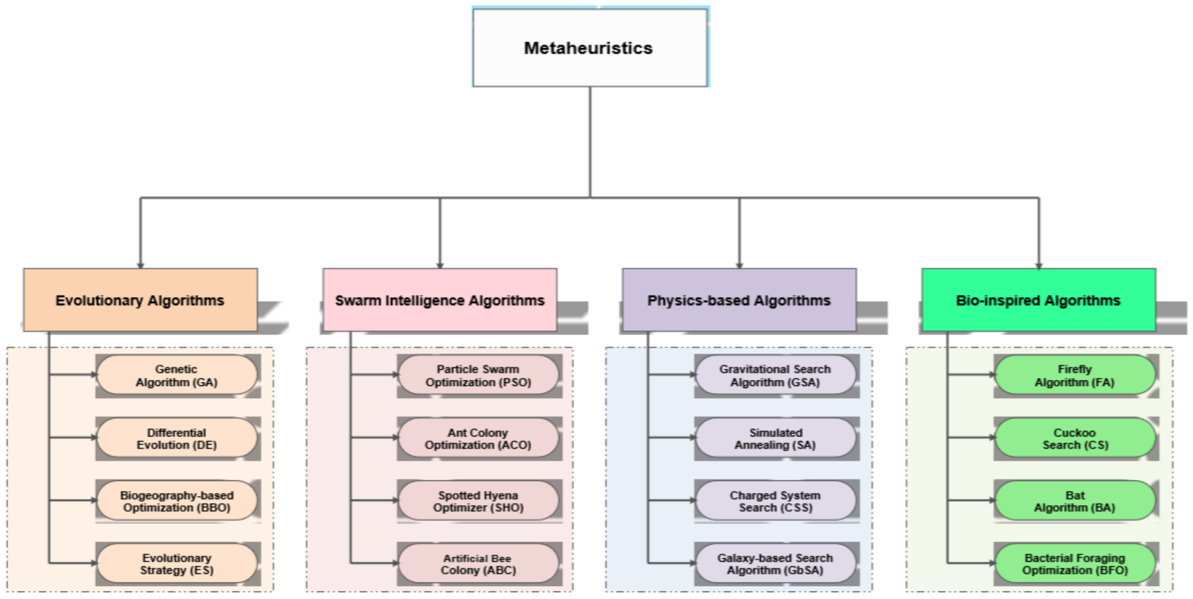
Metaheuristics
Evolutionary Algorithms
Genetic Algorithm (GA)
Differential Evolution (DE)
Biogeography-based Optimization (BBO)
Evolutionary Strategy (ES)
Swarm Intelligence Algorithms
Particle Swarm Optimization (PSO)
Ant Colony Optimization (ACO)
Spotted Hyena Optimizer (SHO)
Artificial Bee Colony (ABC)
Physics-based Algorithms
Gravitational Search Algorithm (GSA)
Simulated Annealing (SA)
Charged System Search (CSS)
Galaxy-based Search Algorithm (GbSA)
Bio-inspired Algorithms
Firefly Algorithm (FA)
Cuckoo Search (CS)
Bat Algorithm (BA)
Bacterial Foraging Optimization (BFO)

Figure 2: Classification of Meta-heuristic Techniques.

Direct data mining techniques are classified in three categories, including classification, estimation and prediction. Classification is a process to divide the set of data into different classes based on the knowledge of predefined classes or the structure of the set of data. It is also referred to as supervised learning. The number of classes in a given set of data is known in advance in supervised learning and assign, the class label of non-classified cases in a given set of data. Indirect data mining techniques are divided into three categories, such as clustering, mining and visualization. Clustering is widely used in indirect data mining. Clustering is a group of objects in a cluster that are very similar to each other or different from other clusters. The discovery of the set of relevant objects without prior knowledge is an unmonitored learning approach. It has been used in many fields of engineering, such as bioinformatics, data collection, and forecasting and image segmentation [5]. Clustering techniques are generally classified in two classes, such as classical and metaheuristic techniques [5]. Classical clustering algorithms are classified into five main classes, such as hierarchical clustering, grid-based clustering, clustering based on density, partitional clustering and clustering based on models. Due to its efficiency and simplicity, K-means is a widely used classic partitional clustering algorithm. However, premature convergence is suffering [6]. Meta-heuristic techniques (see Figure 2) are used in clustering to find the best global partition to overcome this problem. Clustering techniques based on meta-heuristics are divided into two categories. These are clustering techniques based on single and multi-objective.

The main focus of this research is single-objective hybrid method clustering technique based on partitions. It is used to address two major clustering problems, such as determining the number of clusters before the clustering process and finding the best partitions for the cluster. To solve these problems, the partitional single-objective clustering technique is repeatedly used with different number of classes as input and then the partitioning of the data leading to the best cluster validity indices is selected. The validity index of a cluster cannot detect the correct number of clusters. Therefore, a new fitness function must be built that includes more than one cluster validity indices to maintain the inter-cluster and intra-cluster properties. An efficient feature selection technique is required to improve the efficiency of the algorithm and minimize redundancy for uncontrolled data sets. The selection of features in unattended learning is a difficult problem because class labels cannot be used as advisors to search for relevant information. For efficient clustering a new single-objective hybrid approach meta-heuristic technique is therefore required.

1.1 Single-objective:

Minimize/Maximize: $F\,(\sim z) = f_1(\sim z)$	(1.21)	
Subject to:	(1.22)	
$g_j(\sim z) \geq 0,\ j = 1,2,\ldots,p$		
$h_j\,(\sim z) = 0,\ j = 1,2,\ldots,q$	(1.23)	

$Lb_j \leq z_j \leq ub_j, j = 1,2, \ldots, r$	(1.24)	

Many real-life issues need to achieve multiple goals such as minimizing risks, minimizing costs, maximizing reliability, etc.[30]. Single-objective optimization is aimed primarily at finding the best optimal solution and addressing only one goal to be minimized or maximized. The mathematical formulation of single-target optimization is as follows: Where p is the number of inequality constraints, g_j is j^{th} inequality constraints, q is the number of equality constraints, h_j is j^{th} equality constraints, r is the number of variables, lb_j and ub_j are lower and upper bounds of j^{th} variable, respectively.

1.2 Multi-objective:

Multi objective optimization refers to optimizing the functio n of a given problem with more than one objective (criterion). It can be described as [25][26]:

Minimize : $F(\sim z) = [f_1(\sim z), f_2(\sim z), \ldots, f_n(\sim z)]$ (1.25)

Subject to: (1.26)
$g_i(\sim z) \geq 0, i = 1,2, \ldots, m$

$h_i(\sim z) = 0, i = 1,2, \ldots, p$	(1.27)

Where $\sim z = [z_1, z, \ldots, z_k]^T$ is the vector of decision variables, m is the number of inequality constraints, p is the number of equality constraints, g_i is ith inequality constraints, h_i is ith equality constraints, and obj is the number of objective functions $f_i : R^{obj} \rightarrow R$. Because of multi-criterion comparison metrics [34], the solutions in a search space cannot be compared by relational operators.

1.3 Tabu search algorithm:

Glover formalized Tabu Search (TS) in 1986[1]. To manage an embedded local search algorithm, TS has been designed. It uses the search history explicitly, both to escape the local minima and to implement an exploratory strategy. Indeed, its main feature is based on the use of human memory-inspired mechanisms. From this point of view, it takes a path contrary to that of SA that does not use memory and is therefore unable to learn from the past. Different types of memory structures are commonly used through the search space the algorithm has undertaken to remember specific trajectory properties. A Tabu list (from which the name of the meta-heuristic framework derives) records the last encountered solutions (or some of their attributes) and prohibits the re-visitation of these solutions (or solutions containing one of these attributes) as long as they are in the list. This list can be viewed as a short-term memory, recording information about solutions recently visited. Its use prevents the return to the solutions recently visited, thus preventing endless cycling and forcing the search to accept even deteriorating movements. The Tabu list's length controls the search process's memory. The search will focus on small areas of the search space if the length of the list is low. On the contrary, a high length forces the search process to explore larger regions, as it prevents a higher number of solutions from being revisited. During the search, this length may vary, resulting in more robust algorithms, such as the Reactive.

Tabu Search algorithm [2].
Step 1: Choose an initial solution i in S. Set i* = i and k=0.
Step 2: Set k=k+1 and generate a subset V* of solution in N (i,k) such that neither one of the Tabu conditions is violated or at least one of the aspiration conditions holds.
Step 3: Choose a best j in V* and set i=j.
Step 4: If f (i) < f(i*) then set i* = i.
Step 5: Update Tabu and aspiration conditions.
Step 6: If a stopping condition is met then stop. Else go to Step 2.
Some immediate stopping conditions could be the following [2]:
N (i, K+1) = 0. (No feasible solution in the neighborhood of solution i)
K is larger than the maximum number of iterations allowed.
The number of iterations since the last improvement of i* is larger than a specified number.
Evidence can be given that an optimum solution has been obtained.

1.4 Evolutionary Computation algorithm

Evolutionary Computation (EC) is the general term for several optimization algorithms inspired by the Darwinian principles of the ability of nature to evolve well-adapted living beings to their environment. The fields of genetic algorithms [3], evolutionary strategies [4], evolutionary programming [5] and genetic programming [6] are usually found grouped under the term EC algorithms (also called Evolutionary Algorithms (EAs)). Despite the differences that will be shown later between these techniques, they all share a common underlying idea of simulating the evolution of individual structures through selection processes, Recombination and reproduction of mutations, thus creating better solutions. Each iteration of the algorithm corresponds to a generation in which a population of candidate solutions to a given problem of optimization, called individuals, is capable of reproduction and is subject to genetic variations followed by the environmental pressure which causes natural selection (survival of the most fitting). New solutions are created by applying recombination, which combines two or more selected individuals (so-called parents) to produce one or more new individuals (children or o spring) and mutation, which allows the appearance of new characteristics in the o spring to promote diversity. The fitness of the resulting solutions (how good the solutions are) is evaluated and then an appropriate selection strategy is applied to determine which solutions will be maintained in the next generation. A predefined number of generations (or function assessments) of simulated evolutionary processes are usually used as a termination condition, or some more complex stop criteria can be applied.

Evolutionary Computation algorithm:

```
1. Initialize the population with random individuals;
2. Evaluate each individual:
3. Repeat
4. Select parents;
5. Recombine pairs of parents;
6. Mutate the resulting o spring;
7. Evaluate new individuals;
8. Select individuals for the next generation;
Until a termination condition is satisfied;
```

1.5 Swarm Intelligence algorithm:

Swarm Intelligence (SI) is an innovative distributed intelligent paradigm for solving optimization issues inspired by a group of social insect colonies and other animal societies ‘ collective behaviour. Typically, SI systems consist of a population of simple agents (an entity capable of performing / performing certain operations) that interact locally with each other and their environment. These entities with very limited individual capacity can carry out many complex tasks that are necessary for their survival together (cooperatively).While there is normally no centralized control structure dictating how individual agents should behave, local interactions among such agents often lead to global and self-organized behaviour emerging. Several algorithms of optimization inspired by swarming behaviour metaphors are proposed in nature. Examples to this effect are ant colony optimization, particle swarm optimization, bacterial foraging optimization, bee colony optimization, anti-facial immune systems and biogeography-based optimization. Fundamentals of Computational Swarm Intelligence Book[7] introduces the reader to the collective behaviour of mathematical models of social insects and shows how they can be used to solve problems of optimization. Another book by Chan et al.[8] aims to present recent developments and applications related to SI optimization, focusing on the optimization of ant and particle swarm. Das et al. [9] provide a detailed survey of state-of – the-art research focused on bioinformatics applications of SI algorithms. Abraham et al. [10]’s book addresses SI’s use in data mining.

1.6 Ant Colony Optimization algorithm:

M. introduced ant colony optimization (ACO). Dorigo and colleagues [11, 12, 13] as a metaheuristic inspired by nature to solve difficult problems of combinatorial optimization. ACO is inspired by real ants ‘ foraging behaviour. These ants initially explore the area around their nest by performing a randomized walk when searching for food. Ants deposit a chemical pheromone trail on the ground along their path between food source and nest to mark a favourable path that should guide other ants to the food source [14].After some time, there is a higher concentration of pheromone in the shortest path between the nest and the food source and therefore it attracts more ants. Artificial ant colonies used this characteristic of real ant colonies to create solutions to an optimization problem and exchange quality information through a communication scheme that is reminiscent of that adopted by real ants [15].

Algorithm ACO:

```
1. Initialize
2. while termination condition not met do
3. Construct Ants Solutions;
4. Update Pheromones;
5. Daemon Actions;
6. End.
```

1.7 Bee colony optimization-based algorithms:

Bee colony-based optimization algorithms are a new type of algorithm inspired by honeybee colony behaviour that exhibits many features that can be used as models for smart systems and collective behaviour. These characteristics include waggle dance (communication), food foraging, queen bee, task selection, collective decision making, selection of nest sites, bee colony mating and marriage, floral / pheromone laying and navigation systems[16]. For a specific task, each model defines a particular behaviour. The bee hive colonies structure of Honeybee contains a single queen matted to a large number of males (drones) and thousands of workers. The queen is the only female egg-laying in a bee hive, secreting a pheromone that keeps all other females sterile in the colony. A fertile queen can lay fertilized or unfertilized eggs selectively. In workers or virgin queens, fertilized eggs hatch, while unfertilized eggs produce drones. Individual worker bees are always female because male drones, apart from matching with queens during marriage flights, do not contribute to social life. Workers perform various tasks as nurses tending, nest-building, hive defence, and as foragers by collecting nectar and pollen to make honey and feed the brood. A handle of algorithms such as Queen-bee Evolution Algorithm (QBE) [17] and Queen Bee-based crossover operator for GA [18] were developed in the literature on the basis of the Queen Bee concept. Bee Dance and Communication Bees exchange information about where food sources are located by performing a series of movements, often called waggle dance. Each hive has a so-called dance floor area where the bees who have discovered nectar sources dance to promote food locations and persuade their estimates to follow them. When a bee decides to leave the hive to collect nectar, one of the bee dancers follows to one of the nectar areas. The communicative procedures of honey bees such as Bee hive algorithm [19] and Discrete Bee Dance Algorithm [20] have inspired some bee colony-based algorithms.

BCO Algorithm:

```
Initialization: Read problem data, parameter values (B and NC), and stopping criterion.
Do
 1. Assign a (n) (empty) solution to each bee.
 2. For (i = 0; i < NC; i + +)
//forward pass
For (b = 0;
 b < B; b + +)
For (s = 0; s < f(NC);s + +)
//count moves
 (i) Evaluate possible moves;
(ii) Choose one move using the roulette wheel;
//backward pass
(b) For (b = 0; b < B; b + +)
 Evaluate the (partial/complete) solution of bee b;
(c) For (b = 0; b < B; b + +)
Loyalty decision for bee b;
(d) For (b = 0; b < B; b + +)
If (b is uncommitted), choose a recruiter by the roulette wheel
3.    Evaluate all solutions and find the best one. Update xbest
and f(xbest)
```

1.8 GENETIC ALGORITHM (GA):

This method is a search method which is based on the principle of natural selection and genetic [21, 22], GA concept was formalized for the first time by Holland [22]. It emulates the natural selection and the evolution mechanism of Darwin. It has been found and proven to be the most efficient, effective and powerful global optimization algorithm which in general forms combinational optimization problems while in particular the problems having discrete optimization parameters. There is no discontinuous or differentiable object function. The main and basic building blocks of the binary GA are chromosomes and genes. The optimization parameters are encoded by the conventional binary into binary code string. [23] To evolve and develop better solutions and ways perform the selection which is natural, a certain parameter is required to discriminate better from worse solutions. Concerning GA, the measure can function objectively and can be a computer simulation model based on mathematics or it can even be functioning subjectively where human may select finer and better ways and solutions other than ones which are worse Essentially, the measures for the fitness ought to find a fitness which is relative of a candidate solution, which GA will thus use to direct rise and emergence of better solutions [24]. There is one another essential of GA which is population belief. In contrast to the traditional search methods, GA depends on a population of candidate solutions. The population size, which is a parameter, determined by the user, is one of the important elements which influence the GA performance and the scalability. For instance, a small-sized population might result in an immature convergence and offers solutions which are below standard. But huge sized population might result in wasting valuable time in the computing process [25]. After encoding the problem manner of chromosomes and after choosing a parameter of fitness which differentiate the good and bad solutions, the GA becomes ready to find the best solution by means of using the steps bellow [26]:

Genetic Algorithm:

1. Initialization: the primary or starting solutions of candidates for population is mostly produced through the search space in random way.
2. Evaluation: Just when new population is generated or population is initialized, evaluation of the fitness values of the candidate solutions are carried out.
3. Selection: more copies solutions are allocated by means of selection with high level fitness and the idea of survival possibilities of being fittest is imposed on the solutions of the candidates. The main notion choice is to choose the solutions which are better and favouring them to other ones. Therefore, many procedures of selection were suggested to accomplish this notion. Amongst these procedures: the roulette-wheel selection, the stochastic universal selection, the ranking selection and tournament selection.
4. Recombination: Combining parts of two or more of the main solutions to have new and better solutions (i.e. offspring). There are several ways to achieve this and the efficient performance relies on the recombination mechanism which should be properly designed.
5. Mutation: At the time when two or more parental chromosomes are operated by recombining, it results in modification of mutation, a local and random way to solution. Various distinctive types of mutation, but mutation commonly include one change or more that occur in the individual feature(s). In other words, mutation performs a random walk in the candidate solution vicinity.
6. Replacement: The offspring population which is caused by selection, recombination, and mutation replaces the original parental population. Many replacements methods like the Steady state replacement, elitist replacement, generation wise replacement are utilized in GA.
7. Repeating all the steps from 2 till 6 till reach termination condition.

1.9 Nelder-Mead Algorithm:

This simple search method, first developed by Spendley, Hext and Himsworth (1962) [27] and subsequently refined by Nelder and Mead (1965)[28], is a derivative-free line search method used to find the minimum or maximum objective function. See, for example, Olsson and Nelson (1975) [29]. The fitness function value at (N+1) of the initial simplex is evaluated. In this, the function's value is high and new, and then replaced by a good point. Which can be located in a negative gradient form (direction)? These are considered a direct line search technique as one of the best resources. Four basic practices in this process Processing per- soil is applicable. Reflecting, diversifying, storing and reducing these local surface points can be more intensive and the general can make great progress in itself. Thus, in the example below, the function of two variables is minimized (N=2) the basic NM procedure is shown.

1. Sort the A, B, and C function values. Assume if(C) <f (B) <f (A) is the highest of the three function values and must be replaced. In this case, a reflection is made in point D to point E through the centre of BC.

2. If f (E) < f(C) is expanded to point J. Then we replace E or J with r for A, depending on which function value is lower.

3. If f(E)>f(C), there is a contraction to point G or H as a substitute for A, depending on which of f(A) and f(E) is lower, provided that f(G) or f(H) is lower than f(C). If either f(G) or f(H) is greater than f(C), the contraction failed and a shrinkage operation is carried out. The shrinkage procedure reduces the size of the simplex by moving everything but the best point C halfway to the best point C. We've got new points A and B. Return to step 1.

1.10 PSO algorithm:

PSO stands for particle swarms optimization (PSO) it is most popular evolutionary optimization techniques developed by Kennedy and Eberhart (1995)[30,31]in this algorithm population based and evolutionary in nature. It is inspired by the collective behaviour of birds flying around in the sky - those who are engaged in search of their food and are same as fish schooling [32]. This search space is applied to a fitness function to reach good results. The particles swarm through the fitness function solved to search space to find the maximum value return by the objective function. That is a used a number of particles constitute a swarm moving award in search space locking for the best solution. Each particle in search space adjusts its "swarm" according to own swarm experience as well as the swarm experience of the other particle. PSO is same as a genetic algorithm, but the main difference is that they cannot apply filtering. This means that all the members of the population Survive through the entire search process.

The following steps of the PSO algorithm:

1. Initialization process. Randomly generate 5N potential solutions called" particles',' N being the number of parameters to be optimized and a randomized velocity is assigned to each particle.
2. Velocity Update the particles then' fly' through hyperspace while updating their own velocity, which is achieved by taking into account their own past flight and that of their companions.' The velocity and position of the particle is dynamically updated with the following equations[33]:

$V_{id}^{New} = (W * V_{id}^{old}) + c_1 * r1 *$	$(p_{id} - x_{id}^{old}) + c_2 * r2 * (p_{gd} - x_{id}^{old})$ (4)
$X_{id}^{New} = x_{id}^{old} + V_{id}^{New}$	(5)

Where c_1 and c_2 are two positive constants, w is an inertia weight, and r1 and r2 are random number generated [34, 35].

Conclusion

This paper sheds light on the intricate landscape of optimization algorithms, offering a nuanced understanding of their current state and the hurdles they face. The comprehensive analysis of challenges, ranging from algorithmic efficiency to adaptability, underscores the need for continual innovation. Despite the existing impediments, the survey illuminates promising avenues for future research, emphasizing the potential of hybrid models and metaheuristic strategies to enhance optimization algorithm performance. The synthesis of insights from diverse domains, including mathematical programming and machine learning, provides a holistic perspective, facilitating a unified approach to addressing shared challenges. As the optimization field continues to evolve rapidly, this survey serves as a valuable roadmap for researchers and practitioners, guiding them towards novel solutions and fostering collaboration across disciplines. By recognizing and navigating through the identified challenges, the optimization community is poised to unlock unprecedented opportunities, contributing to advancements in diverse applications and enriching the broader landscape of algorithmic research.

References

[1] Garg, R. K., Soni, S. K., Vimal, S., & Dhiman, G. (2023). 3-D spatial correlation model for reducing the transmitting nodes in densely deployed WSN. Microprocessors and Microsystems, 103, 104963.

[2] Dehghani, M., Bektemyssova, G., Montazeri, Z., Shaikemelev, G., Malik, O. P., & Dhiman, G. (2023). Lyrebird Optimization Algorithm: A New Bio-Inspired Metaheuristic Algorithm for Solving Optimization Problems. Biomimetics, 8(6), 507.

[3] Mekala, M. S., Dhiman, G., Park, J. H., Jung, H. Y., & Viriyasitavat, W. (2023). ASXC $^{2} $ Approach: A Service-X Cost Optimization Strategy Based on Edge Orchestration for IIoT. IEEE Transactions on Industrial Informatics.

[4] Kumar, A., Misra, R., Singh, T. N., & Dhiman, G. (2023). APO-AN feature selection based Glorot Init Optimal TransCNN landslide detection from multi source satellite imagery. Multimedia Tools and Applications, 1-38.

[5] Rajinikanth, V., Razmjooy, N., Jamshidpour, E., Ghadimi, N., Dhiman, G., & Razmjooy, S. (2023). Technical and Economic Evaluation of the Optimal Placement of Fuel Cells in the Distribution System of Petrochemical Industries Based on Improved Firefly Algorithm. In Metaheuristics and Optimization in Computer and Electrical Engineering: Volume 2: Hybrid and Improved Algorithms (pp. 165-197). Cham: Springer International Publishing.

[6] Alferaidi, A., Yadav, K., Yasmeen, S., Alharbi, Y., Viriyasitavat, W., Dhiman, G., & Kaur, A. (2023). Node Multi-Attribute Network Community Healthcare Detection Based on Graphical Matrix Factorization. Journal of Circuits, Systems and Computers, 2450080.

[7] Singh, S. P., Dhiman, G., Juneja, S., Viriyasitavat, W., Singal, G., Kumar, N., & Johri, P. (2023). A New QoS Optimization in IoT-Smart Agriculture Using Rapid Adaption Based Nature-Inspired Approach. IEEE Internet of Things Journal.

[8] Singh, S. P., Piras, G., Viriyasitavat, W., Kariri, E., Yadav, K., Dhiman, G., ... & Khan, S. B. (2023). Cyber Security and 5G-assisted Industrial Internet of Things using Novel Artificial Adaption based Evolutionary Algorithm. Mobile Networks and Applications, 1-17.

[9] Khan, M., Kumar, R., Aledaily, A. N., Kariri, E., Viriyasitavat, W., Yadav, K., ... & Vimal, S. (2023). A Systematic Survey on Implementation of Fuzzy Regression Models for Real Life Applications. Archives of Computational Methods in Engineering, 1-21.

[10] Gulia, P., Kumar, R., Viriyasitavat, W., Aledaily, A. N., Yadav, K., Kaur, A., & Dhiman, G. (2023). A

Systematic Review on Fuzzy-Based Multi-objective Linear programming Methodologies: Concepts, Challenges and Applications. Archives of Computational Methods in Engineering, 1-40.

[11] Yadav, A. P., Davuluri, S. K., Charan, P., Keshta, I., Gavilán, J. C. O., & Dhiman, G. (2023, February). Probabilistic Scheme for Intelligent Jammer Localization for Wireless Sensor Networks. In International Conference on Intelligent Computing and Networking (pp. 453-463). Singapore: Springer Nature Singapore.

[12] Athawale, S. V., Soni, M., Murthy, K., Dhiman, G., & Singh, P. P. (2023, February). Weakly Supervised Learning Model for Clustering and Segmentation of 3D Point on Cloud Shape Data. In International Conference on Intelligent Computing and Networking (pp. 531-543). Singapore: Springer Nature Singapore.

[13] Pande, S. D., Kumaresan, T., Lanke, G. R., Degadwala, S., Dhiman, G., & Soni, M. (2023, February). Bidirectional Attention Mechanism-Based Deep Learning Model for Text Classification Under Natural Language Processing. In International Conference on Intelligent Computing and Networking (pp. 465-473). Singapore: Springer Nature Singapore.

[14] Singh, N., Virmani, D., Dhiman, G., & Vimal, S. (2023). Multi to binary class size based imbalance handling technique in wireless sensor networks. International Journal of Nanotechnology, 20(5-10), 477-511.

[15] Yadav, K., Al-Dhlan, K. A., Alreshidi, H. A., Dhiman, G., Viriyasitavat, W. G., Almankory, A. Z., ... & Rajinikanth, V. A novel coarse-to-fine computational method for three-dimensional landmark detection to perform hard-tissue cephalometric analysis. Expert Systems, e13365.

[16] Jiby, B. J., Sakhare, S., Kaur, M., & Dhiman, G. (2022). Multi-Criteria Decision Making in Healthcare: A Bibliometric Review. Demystifying Federated Learning for Blockchain and Industrial Internet of Things, 186-213.

[17] S. Jung, Queen-bee evolution for genetic algorithms, Electronics Letters 39 (2003) 575–576.

[18] A. Karci, Imitation of bee reproduction as a crossover operator in genetic algorithms, in: PRICAI, 2004, pp. 1015–1016.

[19] H.F. Wedde, M. Farooq, Y. Zhang, BeeHive: An e cient fault-tolerant routing algorithm inspired by honey bee behavior, in: M. Dorigo, M. Birattari, C. Blum, L.M. Gambardella, F. Mondada, T. Stutzle¨ (Eds.), Ant Colony Optimization and Swarm Intelligence, 4th International Workshop, ANTS 2004, Brussels, Belgium, September 5 - 8, 2004, Proceedings, number 3172 in Lecture Notes in Computer Science, Springer, 2004, pp. 83–94.

[20] N. Gordon, I.A. Wagner, A.M. Brucks, Discrete bee dance algorithms for pattern formation on a grid, in: Proceedings of the IEEE/WIC International Conference on Intelligent Agent Technology, IAT '03, IEEE Computer Society, Washington, DC, USA, 2003, pp. 545–549.

[21] M. Chau, R Cheng., & B. Kao (2005, December) Uncertain data mining: A new research direction. In Proceedings of the Workshop on the Sciences ofthe Artificial, Hualien, Taiwan (pp. 199-204).Applications, 2010. 37(7): p. 4966-4973.

[22] J. Barker, (1958). Simulation of Genetic Systems by Automatic Digital Computers. Australian Journal of Biological Sciences, 11(4), 603-612.

[23] H.J. Bremermann (1958). The evolution of intelligence: The nervous system as a model of its environment: University of Washington, Department of Mathematics.

[24] H. K Bansal, Simulation of Genetic Algorithm Processor, International Journal of Application or Innovation in Engineering and Management IJAIEM, 2012.

[25] P.G. Gonnade, & S. Bodkhe (2012). Geneticalgorithm for task scheduling in distributed heterogeneous system. International Journal ofAdvanced Research in Computer Science and Software Engineering, 2(10).

[26] K.N. Sastry (2007). Genetic algorithms and geneticprogramming for multiscale modeling: Applications in materials science and chemistry and advances in scalability: ProQuest.

[27]W. Spendley, Hext,G. R., & F.R.Himsworth (1962). Sequential application of simplex designs in optimization and evolutionary operation technometrics, 4, 441–461.

[28]J.A.Nelder, & R. Mead (1965) A simplex method for function minimization. Computer Journal, 7, 308–313.

[29]D.M.Olsson, & L.S. Nelson, (1975).The Nelder–Mead simplex procedure for function minimization. Technometrics, 17, 45–51.

[30] J.Kennedy, and R. C. Eberhart : Particle swarm optimization. In: Proceedings of IEEE International Conference on Neural Networks (1995) 1942-1948.

[31] R. C. Eberhart and Y. Shi.: Comparison between genetic algorithms and particle swarm optimization. In: Proceedings of the 7th Annual Conference on Evolutionary Programming (1998)

[32]J. Kennedy and R. C. Eberhart, Swarm intelligence. San Mateo: Morgan Kaufmann, 2001.69-73

[33] K.E. Parsopoulos, Particle Swarm Optimization and Intelligence: Advances and applications .Hershey, PA, USA: IGIGlobal,2010.

[34]R.C.Eberhart& Shi, Y. (2001). Tracking and optimizing dynamic systems with particle swarms. In Proceedings of the Congress on Evolutionary Computation, Seoul, Korea (pp. 9ems)
[35]Hu, X., & Eberhart, R. C.(2001). Tracking dynamic systems with PSO: where's the cheese? In Proceedings of the Workshop on Particle Swarm Optimization, Indianapolis, IN, USA.