A classification and improvement method of metaheuristic algorithms based on complex networks

Yifei Yang[1] Chaofeng Zhang[2] Wenbin Wang[3] Haichuan Yang[4*] Yuichi Nagata[4*]

[1]University of Toyama
[2]Advanced Institute of Industrial Technology
[3]Tohoku University
[4]Tokushima University
[*]Corresponding author: Haichuan Yang, yokaisen1994@gmail.com; Yuichi Nagata, nagata@is.tokushima-u.ac.jp

Abstract Optimization problems aim to find the best solution from a set of potential alternatives while maximizing or minimizing key metrics. Certain optimization problems exhibit impracticality in deriving exact solutions due to the intricate or colossal solution spaces. Metaheuristic approaches, inspired by natural phenomena, employ strategic, often stochastic, search processes to explore the vast solution spaces of optimization problems, frequently finding near-optimal solutions within a relatively reasonable computational timeframe. Nonetheless, the ”No Free Lunch” theorem asserts that no single algorithm can excel across all optimization problems, thereby necessitating meticulous adaptation, tuning, or hybridization of individual algorithms for effective problem addressing. Traditional methods for algorithm classification and selection are progressively found lacking in navigating through the ever-expanding domain of optimization methodologies. Recently, methods grounded in complex network theory have begun to be explored as potential avenues to negotiate the challenges tied to categorizing and selecting optimization algorithms. By mapping the relationships and performance linkages between various algorithms and problem types, these approaches aspire to equip practitioners with profound insights that adeptly guide the selection, adaptation, and application of algorithms to specific optimization problems.

Keywords metaheuristics; complex network; algorithm selection

Table 1: Nomenclatures used in this paper.

Nomenclature	description
TSP	Traveling Salesman Problem
VRP	Vehicle Routing Problem
GAs	Genetic Algorithms
DE	Differential Evolution
SE	Spherical Evolution
ACO	Ant Colony Optimization
PSO	Particle swarm optimization
SA	Simulated Annealing
PIN	Population Interaction Network

1 Introduction

Optimization problems involve seeking the best solution from a range of possible choices. These problems find extensive applications in mathematics, engineering, computer science, and numerous other domains. The objective of addressing optimization problems is to maximize or minimize a target function under certain constraints. This function could relate to cost, profit, efficiency, or various other metrics. Such problems are ubiquitous in real-world applications like logistics, manufacturing, finance, and many other sectors. However, not all optimization problems are easy to solve. Some, due to their inherent complexity, make it challenging to obtain exact solutions, especially those with high degrees of intricacy or vast solution spaces. For these issues, finding an exact solution might demand exponential computational time, rendering them impractical to solve within a reasonable timeframe.

Metaheuristic approaches have emerged as effective strategies for addressing the daunting challenges posed by complex optimization problems. These approaches, which are inspired by various phenomena in nature, such as the evolution of species, the behavior of ant colonies, or the cooling process of solids, employ strategic, often stochastic, search processes to explore the vast solution spaces of optimization problems. By doing so, they can often find near-optimal solutions in a comparatively reasonable amount of computational time, particularly for problems where an exact solution is practically elusive due to its high computational cost.

However, the ”No Free Lunch theorem” [1] poses a significant theoretical impediment to the universal application of metaheuristics. It posits that no one algorithm is universally superior across all possible optimization problems. Thus, despite the general efficacy of metaheuristics, individual algorithms still need to be meticulously adapted, tuned, or hybridized to address specific problems effectively. The necessity for such tailored adjustments arises from the varied nature and structure of different optimization landscapes.

In the deluge of algorithms, many with their unique configurations and specialties, categorizing and determining the most apt for a given scenario has become an exceedingly intricate task. The traditional methods of classification and selection of algorithms are increasingly found to be insufficient in navigating through the burgeoning domain of optimization methodologies. Thus, choosing, adapting, and improving good algorithms for specific problems have become progressively difficult, leading the research on metaheuristics into an ”alchemy” dilemma where devising effective algorithms often involves a substantial amount of experimentation, trial, and error.

In recent years, methods based on complex network theory have begun to be explored as potential pathways to navigate through the challenges associated with categorizing and selecting metaheuristic algorithms. Complex network-based methods seek to understand the manifold interconnections and dependencies among different algorithms, problem instances, and performance metrics, providing a more nuanced view of the metaheuristic algorithm landscape. Through mapping the relationships and performance linkages between various algorithms and problem types, such approaches aim to furnish practitioners with insights that can guide the selection, adaptation, and application of algorithms to specific optimization problems more adeptly. Table 1 shows the nomenclatures used in this paper.

2 Related Works

Some classic examples of problems hard to precisely solve include: Traveling Salesman Problem (TSP) [2]: Given a series of cities and distances between each pair, the goal is to identify the shortest possible route visiting each city once and returning to the starting point. Knapsack Problem [3]: With a set of items, each having a weight and value, the challenge is selecting

Received: 2023-10-29; Published online: 2024-02-29

items to pack into a fixed-capacity bag so that the total value is maximized without exceeding the weight limit. Job Scheduling Problem [4]: Considering limited machines and a series of jobs, each with a processing time and a deadline, the objective is to determine the order of job execution to minimize total delay or maximize the number of jobs completed on time. Graph Coloring Problem [5]: Given a graph, the aim is to color each node using the fewest colors possible, ensuring that no two adjacent nodes share the same color. Integer Programming [6]: Similar to linear programming, but the decision variables are restricted to integer values. This adds complexity since standard linear programming techniques aren't applicable. Vehicle Routing Problem (VRP) [7]: Given customer locations and demands and one or more distribution centers with vehicles, the goal is to find optimal routes to service all customers, adhering to vehicle capacity limits and other potential constraints. All the aforementioned problems are classified as NP-hard [8].

Furthermore, there are practical issues that, although their difficulty might differ from traditional NP-hard or NP-complete problems, present significant computational challenges. An example is wind farm layout optimization [9], aiming to determine optimal positions for wind turbines to maximize power output and minimize system cost. The problem is intricate because relative turbine positions can influence turbulence effects on each, affecting performance. Depending on constraints (like land availability, roads, and other infrastructure), the problem's difficulty may vary. While the problem is complex, there isn't a consensus on whether it's formally classified as NP-hard. A similar example involves neuron parameter training [10]. Training neural networks is a non-convex optimization issue, meaning multiple local minima could exist. Seeking a global minimum is tremendously challenging, but many local minima can offer decent performance for practical tasks.

There are currently popular solutions addressing some of the above problems. For certain problems, polynomial-time exact algorithms might exist. For example, branch and bound algorithms can handle some small-scale problems [11]. Heuristic methods tailored to a specific problem can be more effective than generic metaheuristic algorithms. For some problems, specialized greedy algorithms [12] or local search algorithms [13] can be designed based on structural characteristics. For large-scale problems, decomposing them into smaller sub-problems is an option. Approaches like column generation [14], Lagrangian relaxation [15], and Benders decomposition [16] are examples. Notably, in recent years, deep learning methods, especially reinforcement learning [17], have been applied to some combinatorial optimization problems, yielding impressive results. Neural networks and reinforcement learning methods have been used to generate approximate solutions for problems like TSP and VRP [18,19]. Parallel and distributed computing methods also exist, employing multiple processors or computers to simultaneously tackle different parts of a problem or search space, accelerating the solving process. However, the methods mentioned are more suited for solving small-scale instances or those with specific structures. For NP-hard or NP-complete issues, and highly challenging complex problems, metaheuristic algorithms often prove the most effective solution.

3 Metaheuristic algorithms

Metaheuristic algorithms are advanced heuristic algorithms designed to tackle large-scale or intricate optimization problems [20]. A defining characteristic of these algorithms is that they are not solely designed for specific problems but provide a framework that can be employed across various problems. They often attempt to mimic processes found in biology, society, or other natural phenomena. Some well-known metaheuristic algorithms include: Genetic Algorithms (GAs) [21] are inspired by the process of natural selection and genetics. This method uses "chromosomes" to encode potential solutions and produces new solutions via operations such as crossover, mutation, and selection. On the basis of GA, two algorithms, Differential Evolution (DE) [20] and Spherical Evolution (SE) [22], were proposed. The DE family of algorithms has proven notably successful [23], with its improved version regularly clinching top positions in the IEEE CEC competition [24]. The SE algorithm has extremely high potential in more complex and high-dimensional problems. Ant Colony Optimization (ACO) [25] is inspired by the process of ants searching for food. Ants communicate by releasing and following pheromone trails to identify the shortest paths. Particle Swarm Optimization (PSO) [26] mimics the social behavior of bird flocks or schools of fish. Each "particle" moves, updating its velocity and position based on its individual and the swarm's best-known positions. Simulated Annealing (SA) [27] is inspired by the cooling process of solids and crystal arrangements. Solutions undergo random alterations under a controlled temperature parameter, permitting acceptance of less optimal solutions early in the search to escape local optima. In addition, the Evolution Strategy with Covariance Matrix Adaptation [28] has been highly praised by researchers due to its complete mathematical foundation.

The broad attention and application of metaheuristic algorithms stem from their many advantages: They are generally universal and suitable for diverse problems without extensive customization. They often provide reasonable solutions, irrespective of specific problem structures or characteristics. Some can adjust their strategies or parameters during the search, adapting to the problem's nature [29]. For NP-hard problems or those where precise solutions are unattainable in a reasonable timeframe, metaheuristics can often find satisfactory approximations. They can be coupled with other algorithms, like local searches or greedy strategies, to yield improved solutions. Compared to basic local search methods, they tend to explore the solution space more extensively, offering a greater chance of identifying global or near-optimal solutions. Some, like GA or PSO, have parallel structures, facilitating parallel searches on multi-processors or clusters [30].

3.1 Improvements in metaheuristics

Due to their impressive performance in addressing various real-world optimization and decision-making problems, enhancements and studies of these algorithms continue to captivate researchers and practitioners alike. Some key areas of focus in current metaheuristic research include:

1) Parameter tuning [32]: This method is vital for optimizing metaheuristic performance. Adaptive methods, hyperparameter optimization, and Automated Machine Learning (AutoML) strategies have been employed to automatically identify optimal parameters. It should be emphasized that powerful metaheuristics generally require the use of various parameter adaptation techniques.

2) Hybrid methods [33,34]: Combining different metaheuristics or merging them with other optimization strategies like linear programming or local searches can enhance both search efficiency and solution quality. Such improved techniques have become increasingly difficult to publish in high-quality journals, but in fact, many new competitive metaheuristics are also based on a mixture of various mechanisms.

3) Memetic computing manner [35,36]: Memetic computing delves into intricate formations arising from the blend of basic entities and memes. As they evolve and interact,

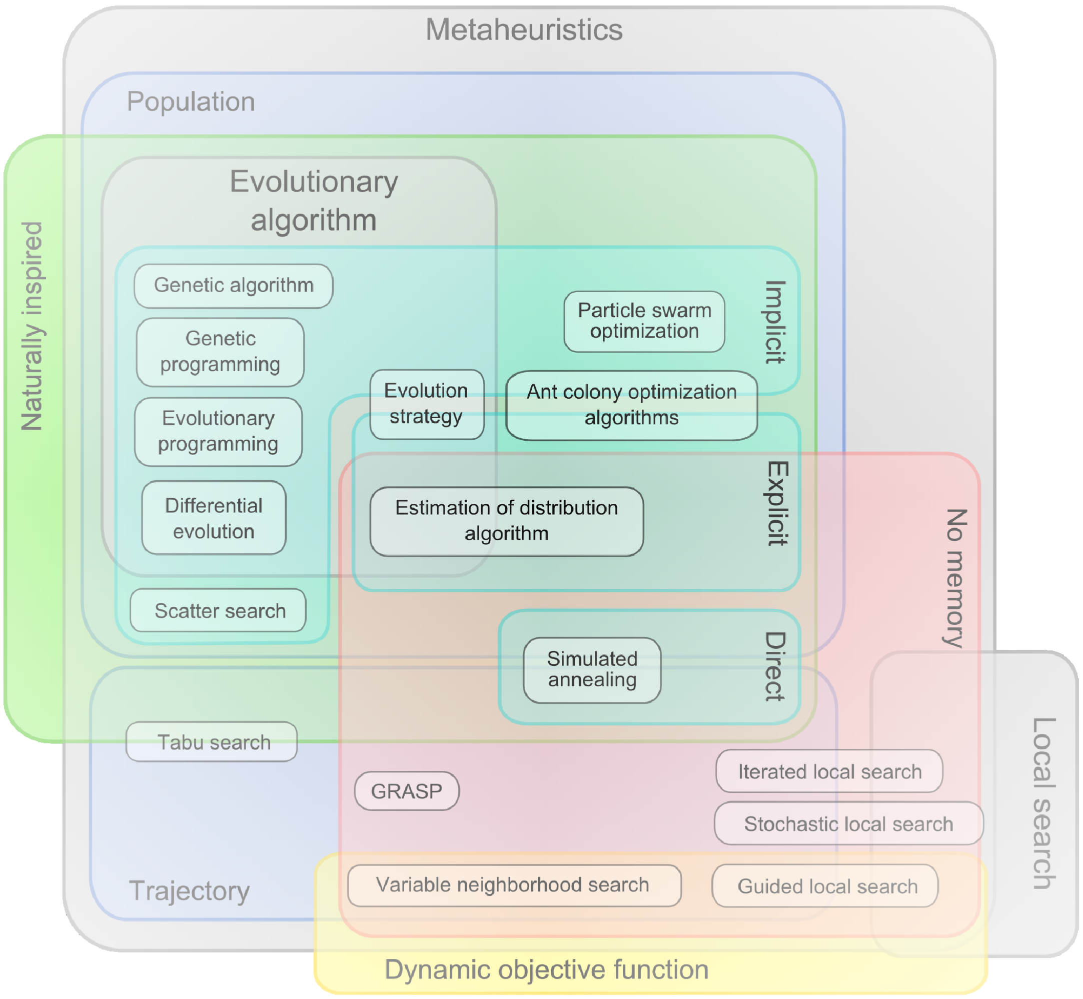


Figure 1: Some classification methods of metaheuristic algorithms [31].

they form sophisticated systems proficient in addressing challenges. The foundational principle of this area lies in memetic algorithms, a category of optimization techniques marked by their evolutionary underpinnings and an assortment of localized search mechanisms.

4) Parallelization and distributed computation: Given the inherent parallelism in many metaheuristics (e.g., GAs, PSO), leveraging modern computational hardware for parallel and distributed computing is pivotal.

5) Multi-objective optimization [37]: Many real-life problems encompass multiple conflicting objectives. Multi-objective versions of several metaheuristics, like multi-objective GAs or multi-objective PSO, have been developed in response.

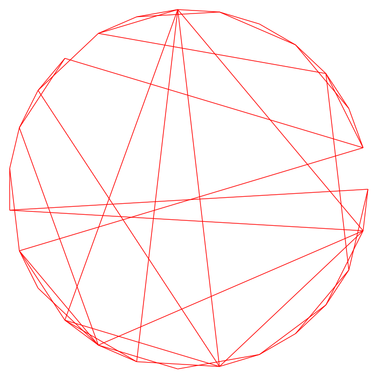
6) Theoretical analysis [38 – 41]: Even though metaheuristics are largely empirical, theoretical analysis remains a crucial area of study. This can aid in understanding algorithm behavior and direct improvements.

7) Application-driven research: New real-world applications and challenges often stimulate the development or refinement of existing metaheuristics. For instance, new problems in the domains of transportation, logistics, and energy system optimization have galvanized algorithmic research.

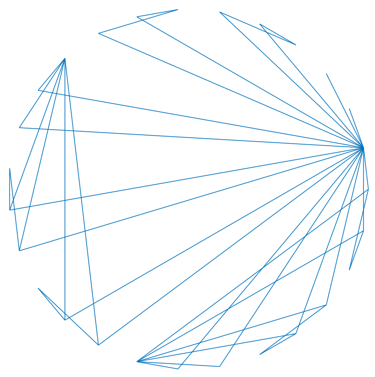
8) Integration with machine learning [42]: The convergence of machine learning and metaheuristics, especially the amalgamation of deep learning and reinforcement learning, has emerged as a hot research topic recently. Examples include employing neural networks to guide search strategies or using reinforcement learning to tweak algorithm parameters.

3.2 Classification of metaheuristics

Moreover, with the emergence of more and more metaheuristic algorithms, selecting a suitable algorithm for practical problems has become increasingly complex. To determine which algorithm is best for a particular problem, benchmark testing of multiple algorithms might be necessary, which could require significant time and computational resources. Classifying metaheuristic algorithms based on their characteristics can help in systematically and specifically understanding and choosing the appropriate algorithm. Generally, metaheuristic algorithms are optimization algorithms inspired by natural phenomena, they can be grouped into several basic categories based on their operations and features: Population-based algorithms: These algorithms typically maintain a set of potential solutions and improve these solutions through iterative evolution. The ”quality” or ”fitness” of the solutions determines their probability of survival and reproduction in subsequent iterations. Examples include GAs, PSO, and DE. Simulation-based algorithms: These algorithms search the solution space by simulating a certain natural or physical process. An example is SA, which simulates the annealing process of solids. Memory or learning-based algorithms: These algorithms accumulate knowledge during the search process and use this knowledge to guide subsequent searches. An example is the ACO, where ants use pheromone trails when searching for food paths. Local search-based algorithms: These



(a) Poisson degree distribution.



(b) power-law degree distribution.

Figure 2: Two typical population interaction networks of metaheuristics.

algorithms start from an initial solution and then search for better solutions in their neighborhood. An example is Tabu Search [43]. Cooperation and competition-based algorithms: These algorithms often simulate interactions between individuals who can cooperate, compete, or do both to achieve some objective. Examples include artificial life and evolutionary strategies [44]. Additionally, some literature categorizes metaheuristics into algorithms based on evolution, algorithms based on swarm intelligence, algorithms based on human behavior, and algorithms based on physics and chemistry [45]. Some other classification methods of metaheuristic algorithms are shown in Fig. 1. Classifying metaheuristic algorithms by their inherent characteristics can aid in understanding the advantages and applicability of each algorithm, helping to choose the right algorithm for specific problems. For instance, for problems requiring global search capabilities, population-based algorithms might be more suitable, while for solutions that need fine-tuning, local search-based algorithms may be better.

4 Challenges faced by metaheuristic research

The proliferation of improvements to metaheuristic algorithms and the continuous emergence of new metaheuristic algorithms have also ignited controversy. Particularly, many of the recently proposed metaheuristics draw inspiration from natural or social phenomena, such as fire propagation, animal migration, and human social behavior. While the names and concepts of these algorithms might seem appealing, the pivotal question is whether they truly provide novel and effective means to address real-world problems. Do these new algorithms truly outperform established, time-tested ones such as GAs, SA, and ACO? Only when a new algorithm consistently demonstrates superior performance across various benchmark problems and practical applications can it be deemed valuable. Does the new algorithm bring genuinely innovative ideas, or is it a slight variation of existing methods? To gain widespread acceptance, a new algorithm should make distinct contributions and demonstrate innovative elements. Moreover, although most metaheuristic algorithms are empirical in nature, providing some theoretical analysis and proof for the new algorithm can bolster its credibility. This can help explain why the algorithm is effective and under which conditions it might excel. Another important evaluation criterion for a new algorithm is its universality—does it apply across different problems or is it tailored for specific challenges? Generally, a versatile algorithm effective for a range of problems is more well-received. Lastly, is the new algorithm easily understood and implemented? An overly complex algorithm, even with marginal performance advantages, may not achieve broad adoption, especially if it's more intricate or harder to implement than existing algorithms.

These classifications of metaheuristics also have certain limitations. Many metaheuristic algorithms might possess characteristics from multiple categories. For instance, a population-based algorithm might also have some form of local search mechanism, making it challenging to classify it under a specific category. Moreover, classifying algorithms into a few fixed categories might restrict our understanding of their potential variations and adaptability. Especially as algorithms evolve, new strategies and techniques might be integrated into existing ones, making them surpass their original classifications. Therefore, categorizing algorithms might lead to an overly simplified perspective, potentially overlooking key details and mechanisms within certain algorithms. These classification methods might not always perfectly apply to all new algorithms or variations. It's worth noting that the aforementioned classification overly reliant on the heuristic source of algorithms (like natural phenomena) might not always be helpful. Some algorithms, even if inspired by the same phenomena, might differ significantly in their operations and performances. Lastly, even if we can classify algorithms based on their features, this doesn't always directly guide the selection of the best algorithm. In practical applications, the performance of algorithms might be affected by factors like problem characteristics, parameter settings, etc. Currently, there are studies focusing on this issue [46,47], attempting to categorize algorithms from an exploitation and exploration perspective. However, exploitation and exploration are broad concepts that warrant further in-depth research. In light of the shortcomings present in the aforementioned studies, some researchers have proposed a more radical viewpoint, attempting to find a unified model for all metaheuristic algorithms, and have indeed achieved some success [22]. However, this research more significantly advances the theoretical study of algorithms and does not assist researchers in filtering and utilizing algorithms.

5 Complex network-based research methods

In recent years, complex systems have emerged as a focal point of research, and notably, due to their ”groundbreaking contributions to our understanding of complex systems,” the 2021 Nobel Prize in Physics was awarded to three scientists. This spotlight not only underscores the significance and potential breakthroughs in the realm of complex systems research but also pro-

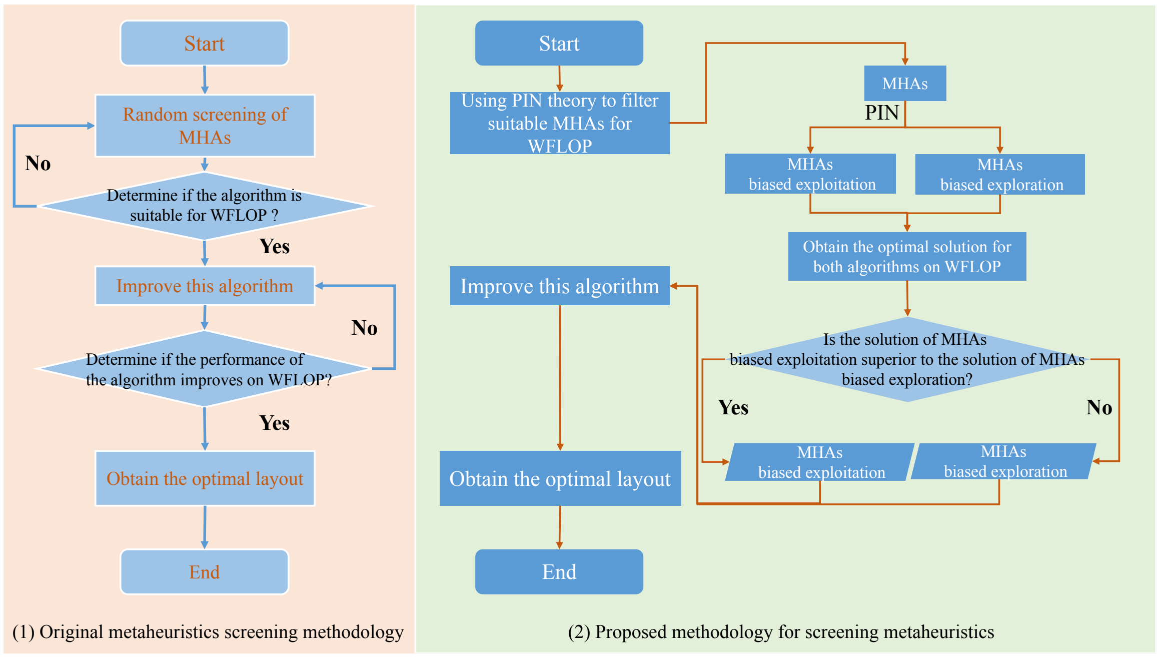

Start
Random screening of MHAs
No
Determine if the algorithm is suitable for WFLOP ?
Yes
Improve this algorithm
No
Determine if the performance of the algorithm improves on WFLOP?
Yes
Obtain the optimal layout
End
(1) Original metaheuristics screening methodology
Start
Using PIN theory to filter suitable MHAs for WFLOP
MHAs
PIN
MHAs biased exploitation
MHAs biased exploration
Obtain the optimal solution for both algorithms on WFLOP
Is the solution of MHAs biased exploitation superior to the solution of MHAs biased exploration?
Yes
No
MHAs biased exploitation
MHAs biased exploration
Improve this algorithm
Obtain the optimal layout
End
(2) Proposed methodology for screening metaheuristics

Figure 3: Flowchart of the proposed complex network-based research methods [48].

pels forward scientific inquiry and exploration into the intricate and multifaceted networks that govern various phenomena within these systems. Moreover, employing complex network theory to classify metaheuristic algorithms, and even to guide the enhancement of metaheuristic algorithms, emerges as an intriguing and challenging idea [49 – 51], linking foundational physics and algorithm development through innovative approaches to understanding and navigating complexity. Complex network theory investigates the structure and behavior among a multitude of interconnected elements, while metaheuristic algorithms involve the search and optimization of potential solutions in the solution space. In metaheuristic algorithms, each potential solution can be regarded as a node within the network. Edges, on the other hand, can represent some type of relationship between solutions, such as mutation, crossover, or neighborhood transitions. During the search process in metaheuristic algorithms, their populations can generate certain network structures within the solution space. For example, random searches might result in a random network structure, while neighborhood-based searches might produce networks with specific topological characteristics. Different metaheuristic algorithms might lead to networks with different statistical properties, which can aid in understanding the search behavior and efficiency of the algorithms.

Recently, our research has further revealed the correlation between the network structure of Population Interaction Network (PIN) within algorithms and the algorithm's performance on specific problems. The core idea of this research approach is to utilize complex networks to categorize the search strategy (i.e., exploration and exploitation) of algorithms, thereby providing guidance for algorithm selection and improvement. In complex networks, there are two typical network structures: one is the network structure with a Poisson distribution, such as small-world networks, and the other is the network structure with a power-law distribution, like scale-free networks. In the network structure with a Poisson distribution, interactions between vertices and other vertices are more evenly distributed and random. Meanwhile, in the network structure with a power-law distribution, some vertices have more edges than others, as shown in Fig. 2. Metaheuristics with a Poisson distribution PIN tend to explore the solution space, while those with a power-law distribution PIN are more inclined to exploit in potentially promising areas. When facing black-box problems, we can conjecture the attributes of the problem itself by running algorithms with different PIN structures, thereby providing possible directions for further selection and improvement of algorithms, with the specific process seen in Fig. 3. We analyzed the training of neuronal model parameters [52], optimization of wind farm layouts [48], and the IEEE CEC2017 benchmark function [53], obtaining the following insights:

1) Matching Algorithm Characteristics with Problem Characteristics: The research results show that, for wind farm layout optimization problems, algorithms that tend to have a Poisson distribution PIN generally perform better than those tending to have a power-law distribution PIN. The opposite is true for the IEEE CEC 2017 standard function set and dendritic neuron training problems. This indicates that certain attributes of algorithms may be more crucial when solving some optimization problems.

2) Complex Networks as a Tool for Categorizing Metaheuristic Algorithms: By constructing and analyzing population interaction networks, researchers can gain a deeper understanding of the internal working mechanisms of algorithms. This method provides a new perspective for understanding and improving algorithms.

3) Providing Theoretical Guidance for Algorithm Selection and Improvement: Traditional metaheuristic algorithm selection methods usually require a lot of experiments and computations. Utilizing complex network theory, researchers can predict its performance on specific problems based on the network characteristics of algorithms, thus providing more targeted guidance for algorithm selection and improvement.

6 Conclusion

The application of complex networks to the realm of optimization algorithm classification and selection not only represents a novel, systematic method for navigating the expansive solution spaces of intricate problems but also offers a platform for potentially unraveling the deeper intricacies of problem-algorithm dynamics. The goal is to transcend beyond the current heuristic-based paradigm and forge a pathway towards a more structured, systematic, and theoretically grounded approach to algorithm selection and adaptation, thereby transforming the ”alchemy” of metaheuristic research into a more exact science.

7 Acknowledgement

This work is partially supported by JSPS KAKENHI Grant Numbers JP22K17884, JP20H04174, JP22K11989, and the Tokushima University Tenure-Track Faculty Development Support System, Japan.

References

1. Wolpert DH, Macready WG. No free lunch theorems for optimization. IEEE Transactions on Evolutionary Computation. 1997;1: 67 – 82.
2. Jünger M, Reinelt G, Rinaldi G. The traveling salesman problem. Handbooks in operations research and management science. 1995;7: 225 – 330.
3. Chu PC, Beasley JE. A genetic algorithm for the multidimensional knapsack problem. Journal of heuristics. 1998;4: 63 – 86.
4. Applegate D, Cook W. A computational study of the job-shop scheduling problem. ORSA Journal on computing. 1991;3: 149 – 156.
5. Fleurent C, Ferland JA. Genetic and hybrid algorithms for graph coloring. Annals of operations research. 1996;63: 437 – 461.
6. Pugh W. The omega test: a fast and practical integer programming algorithm for dependence analysis. Proceedings of the 1991 acm/ieee conference on supercomputing. 1991. pp. 4 – 13.
7. Braekers K, Ramaekers K, Van Nieuwenhuyse I. The vehicle routing problem: State of the art classification and review. Computers & industrial engineering. 2016;99: 300 – 313.
8. Woeginger GJ. Exact algorithms for np-hard problems: A survey. Combinatorial optimization—eureka, you shrink! papers dedicated to jack edmonds 5th international workshop aussois, france, march 5 – 9, 2001 revised papers. Springer; 2003. pp. 185 – 207.
9. Chen Y, Li H, Jin K, Song Q. Wind farm layout optimization using genetic algorithm with different hub height wind turbines. Energy Conversion and Management. 2013;70: 56 – 65.
10. Yeh W-C. New parameter-free simplified swarm optimization for artificial neural network training and its application in the prediction of time series. IEEE Transactions on Neural Networks and Learning Systems. 2013;24: 661 – 665.
11. Narendra, Fukunaga. A branch and bound algorithm for feature subset selection. IEEE Transactions on Computers. 1977;C-26: 917 – 922.
12. Zhang Z, Schwartz S, Wagner L, Miller W. A greedy algorithm for aligning dna sequences. Journal of Computational biology. 2000;7: 203 – 214.
13. Aarts EH, van Laarhoven PJ, Lenstra JK, Ulder NL. A computational study of local search algorithms for job shop scheduling. ORSA Journal on Computing. 1994;6: 118 – 125.
14. Barnhart C, Johnson EL, Nemhauser GL, Savelsbergh MW, Vance PH. Branch-and-price: Column generation for solving huge integer programs. Operations research. 1998;46: 316 – 329.
15. Fisher ML. The lagrangian relaxation method for solving integer programming problems. Management science. 1981;27: 1 – 18.
16. Rahmaniani R, Crainic TG, Gendreau M, Rei W. The benders decomposition algorithm: A literature review. European Journal of Operational Research. 2017;259: 801 – 817.
17. Mazyavkina N, Sviridov S, Ivanov S, Burnaev E. Reinforcement learning for combinatorial optimization: A survey. Computers & Operations Research. 2021;134: 105400.
18. Liu F, Zeng G. Study of genetic algorithm with reinforcement learning to solve the tsp. Expert Systems with Applications. 2009;36: 6995 – 7001.
19. Nazari M, Oroojlooy A, Snyder L, Takác M. Reinforcement learning for solving the vehicle routing problem. Advances in neural information processing systems. 2018.
20. Abdel- Basset M, Abdel- Fatah L, Sangaiah AK. Metaheuristic algorithms: A comprehensive review. Computational intelligence for multimedia big data on the cloud with engineering applications. 2018; 185 – 231.
21. Holland JH. Genetic algorithms. Scientific american. 1992;267: 66 – 73.
22. Tang D. Spherical evolution for solving continuous optimization problems. Applied Soft Computing. 2019;81: 105499.
23. Das S, Suganthan PN. Differential evolution: A survey of the state-of-the-art. IEEE Transactions on Evolutionary Computation. 2011;15: 4 – 31.
24. Awad NH, Ali MZ, Suganthan PN, Reynolds RG. An ensemble sinusoidal parameter adaptation incorporated with l-shade for solving cec2014 benchmark problems. 2016 ieee congress on evolutionary computation (cec). 2016. pp. 2958 – 2965.
25. Dorigo M, Birattari M, Stutzle T. Ant colony optimization. IEEE Computational Intelligence Magazine. 2006;1: 28 – 39.
26. Kennedy J, Eberhart R. Particle swarm optimization. Proceedings of icnn’95 - international conference on neural networks. 1995. pp. 1942 – 1948.
27. Bertsimas D, Tsitsiklis J. Simulated annealing. Statistical science. 1993;8: 10 – 15.
28. Hansen N, Müller SD, Koumoutsakos P. Reducing the time complexity of the derandomized evolution strategy with covariance matrix adaptation (cma-es). Evolutionary Computation. 2003;11: 1 – 18.
29. Zhang C, Li H, Yang Y, Zhang B, Zhu H, Gao S. Improved differential evolutionary algorithm based on adaptive scaling factor. International conference on industrial, engineering and other applications of applied intelligent systems. Springer; 2023. pp. 171 – 176.
30. Roberge V, Tarbouchi M, Labonte G. Comparison of parallel genetic algorithm and particle swarm optimization for real-time uav path planning. IEEE Transactions on Industrial Informatics. 2013;9: 132 – 141.
31. Hugo V. Are most metaheuristic algorithms different metaphors for the same method? Available: https://cs.stackexchange.com/q/74915
32. Huang C, Li Y, Yao X. A survey of automatic parameter tuning methods for metaheuristics. IEEE Transactions on Evolutionary Computation. 2020;24: 201 – 216.
33. Li H, Yang H, Zhang B, Zhang H, Gao S. Swarm exploration mechanism-based distributed water wave optimization. International Journal of Computational Intelligence Systems. 2023;16: 1 – 26.
34. Dong S, Yang Y, Xu W, Zhu Q, Zhang Y, Li X. L-cjade: Differential evolution algorithm based on a linearly decreasing population structure. 2023 8th international conference on computer and communication systems (icccs). 2023. pp. 875 – 879.
35. Nagata Y, Bräysy O. Edge assembly-based memetic algorithm for the capacitated vehicle routing problem. Networks: An International Journal. 2009;54: 205 – 215.
36. Neri F, Cotta C. Memetic algorithms and memetic computing optimization: A literature review. Swarm and Evolutionary Computation. 2012;2: 1 – 14.
37. Deb K. Multi-objective optimisation using evolutionary algorithms: an introduction. Multi-objective evolutionary optimisation for product design and manufacturing. Springer; 2011. pp. 3 – 34.
38. He J, Yao X. Drift analysis and average time complexity of evolutionary algorithms. Artificial intelligence. 2001;127: 57 – 85.
39. Akimoto Y, Nagata Y, Ono I, Kobayashi S. Theoretical foundation for cma-es from information geometry perspective. Algorithmica. 2012;64: 698 – 716.
40. Nagata Y. High-order entropy-based population diversity measures in the traveling salesman problem. Evolutionary computation. 2020;28: 595 – 619.
41. Doerr B, Qu Z. Runtime analysis for the nsga-ii: Provable speed-ups from crossover. Proceedings of the aaai conference on artificial intelligence. 2023. pp. 12399 – 12407.
42. Yu X, Lu Y. Reinforcement learning-based multi-objective differential evolution for wind farm layout optimization. Energy. 2023; 129300.
43. Glover F. Tabu search—part i. ORSA Journal on computing. 1989;1: 190 – 206.
44. Beyer H-G, Schwefel H-P. Evolution strategies – a comprehensive introduction. Natural computing. 2002;1: 3 – 52.
45. Yang H, Gao S, Wang R-L, Todo Y. A ladder spherical evolution search algorithm. IEICE Transactions on Information and Systems. 2021;104: 461 – 464.
46. Yang H, Yu Y, Cheng J, Lei Z, Cai Z, Zhang Z, et al. An intelligent metaphor-free spatial information sampling algorithm for balancing exploitation and exploration. Knowledge-Based Systems. 2022;250: 109081.
47. Črepinšek M, Liu S-H, Mernik M. Exploration and exploitation in evolutionary algorithms: A survey. ACM computing surveys (CSUR). 2013;45: 1 – 33.
48. Yang H, Gao S, Lei Z, Li J, Yu Y, Wang Y. An improved spherical evolution with enhanced exploration capabilities to address wind farm layout optimization problem. Engineering Applications of Artificial Intelligence. 2023;123: 106198.
49. Gao S, Wang Y, Wang J, Cheng J. Understanding differential evolution: A poisson law derived from population interaction network. Journal of Computational Science. 2017;21: 140 – 149.
50. Lynn N, Ali MZ, Suganthan PN. Population topologies for particle swarm optimization and differential evolution. Swarm and evolutionary computation. 2018;39: 24 – 35.
51. Wang Y, Gao S, Yu Y, Xu Z. The discovery of population interaction with a power law distribution in brain storm optimization. Memetic Computing. 2019;11: 65 – 87.
52. Zhang Y, Yang Y, Li X, Yuan Z, Todo Y, Yang H. A dendritic neuron model optimized by meta-heuristics with a power-law-distributed population interaction network for financial time-series forecasting. Mathematics. 2023;11: 1251.
53. Li X, Li J, Yang H, Wang Y, Gao S. Population interaction network in representative differential evolution algorithms: Power-law outperforms poisson distribution. Physica A: Statistical Mechanics and its Applications. 2022;603: 127764.