Emergence, Complexity and Computation ECC

Abdul Hanif Abdul Halim
Swagatam Das
Idris Ismail

Into a Deeper Understanding of Evolutionary Computing: Exploration, Exploitation, and Parameter Control

Volume 1

Springer

Emergence, Complexity and Computation

Volume 50

Abdul Hanif Abdul Halim · Swagatam Das ·
Idris Ismail

Into a Deeper Understanding of Evolutionary Computing: Exploration, Exploitation, and Parameter Control

Volume 1

Springer

Abdul Hanif Abdul Halim
Department of Electrical and Electronics Engineering
Universiti Teknologi PETRONAS
Seri Iskandar, Perak, Malaysia

Swagatam Das
Electronics and Communication Sciences Unit
Indian Statistical Institute
Kolkata, West Bengal, India

Idris Ismail
Department of Electrical and Electronics Engineering
Universiti Teknologi PETRONAS
Seri Iskandar, Perak, Malaysia

ISSN 2194-7287 ISSN 2194-7295 (electronic)
Emergence, Complexity and Computation
ISBN 978-3-031-74012-1 ISBN 978-3-031-74013-8 (eBook)
https://doi.org/10.1007/978-3-031-74013-8

Foreword

Evolutionary algorithms and other population-based metaheuristics have become increasingly popular for solving hard optimization problems for the last 40 years. However, in spite of their apparent simplicity, their proper use faces several challenges, including the initialization of their parameters and their proper tuning and control, as well as finding the proper balance between exploitation and exploration during the search process. This book presents a comprehensive overview of these and other relevant topics related to metaheuristic improvement, with the aim of achieving a better effectiveness and efficiency in their use.

The book is divided in two volumes. The first volume consists of three chapters that focus on: algorithm initialization, the tradeoff between exploration and exploitation, and using distributions and functions to guide the optimization process. Next, I will provide a brief description of each chapter.

In Chap. 1, the authors discuss the importance of the initialization of parameters for the performance of metaheuristic algorithms. The authors consider two types of initialization methods: (1) those oriented toward uniformity of the distribution of solutions and (2) those that bias the initial solutions toward promising regions of the search space. On the one hand, the uniformity of initial solutions aims to ensure that the individuals are scattered evenly. This task can be achieved using computationally efficient methods (e.g., random initialization, probability distribution initialization, and sub-space-based initialization). On the other hand, biasing the initial solutions toward promising regions of the search space may increase the convergence rate as well as solution accuracy. The approaches adopted in this case are sub-divided into four categories: (1) location-based initialization, (2) hybrid initialization, (3) initialization methods related to machine learning and mathematical programming, and (4) initialization based on a specific application. In spite of their advantages, as the authors properly indicate, these approaches may involve the use of considerable computational resources since they require a pre-optimization process.

Chapter 2 discusses one of the oldest and most difficult problems faced when using evolutionary algorithms (and other similar population-based metaheuristics): how to produce a proper balance between exploration and exploitation during the search.

The chapter discusses aspects related to producing good tradeoffs between exploration and exploitation, including randomization, memory allocation, and individual update mechanisms, as well as diversity, fitness landscape analysis, and hybridization. Randomization is a key aspect of metaheuristics and the authors indicate that there are normally two options to be applied in an algorithm: (1) modifying the magnitude of randomization, which affects the trajectory of individual solutions in different paths or (2) performing randomized moves which determine the probability condition that leads to different paths or moves of a portion of the population (such portion is defined using a probability factor). In either case, randomization is intended to improve exploration in a broader portion of the search space as well as to explore in an exploited area that could potentially lead to a promising solution.

The second section of Chap. 2 discusses memory allocation and individual update mechanisms. The authors indicate that memory usage allows algorithms to be proactive during exploration and exploitation and serves as an intelligent tool to avoid non-promising areas of the search space. The chapter discusses short-term, medium-term, and long-term memory, analyzing how each of them has a different impact on exploration and exploitation. Then, the integration of memory allocation into the individual update mechanism is discussed. This integration aims to guide the search toward better-memorized solutions in order to avoid getting trapped in local optima.

Chapter 2 also discusses diversity measures and fitness landscape analysis as additional mechanisms that can improve convergence and provide a better balance between exploration and exploitation. This chapter discusses several types of diversity measures including fitness sharing, niching, crowding, and entropy. Additionally, different features of fitness landscapes that can be considered during the optimization process are briefly discussed, including basins, ruggedness, neutrality, gradient, deception, and evolvability. The use of fitness landscape analysis for designing robust novel algorithms and to assess performance is also discussed.

The final section of Chap. 2 focuses on hybridization, which consists of the combination of different methods or algorithms into a single metaheuristic. Two types of hybrids are analyzed: collaborative and integrative. Collaborative methods combine different algorithms and run in sequence or in parallel in order to solve the problem, whereas integrative approaches combine the concepts from two or more algorithms in order to form a single integrated algorithm.

Chapter 3 discusses probability distributions and the way in which they help to locate individuals in a broader section of the search space, as a means to prevent them to get trapped in local optima. Several probability distributions are discussed, including Lévy, Cauchy, beta, extreme value, generalized extreme value, generalized Pareto, gamma, inverse gamma, half-normal, inverse Gaussian, logistic, and chaotic map distribution. The authors also discuss how the exploration and exploitation can be dynamically shaped using functions related to the iterations rate (e.g., logarithmic, trigonometric, sigmoid, exponential, spiral, or combinations of them). Finally, the authors also indicate that the degree of exploration can be adjusted using random perturbations (e.g., following a spiral path).

Volume Two consists of two main chapters. Chapter 4 discusses factors that belong to the location, size, and capacity for optimization. Such factors include search space reduction, population models, opposition-based learning, and algorithm selection methods. The authors argue that these factors are further enhancements to those discussed in Chap. 3 for balancing the exploration and exploitation mechanisms and they provide detailed discussions of each of them.

Chapter 5 focuses on parameter adaptation and discusses two topics: (1) parameter tuning in which the values and algorithm settings are adjusted prior to the execution of the algorithm, and (2) parameter control, in which the parameters values and algorithm settings are adjusted during the algorithm's execution. A number of techniques for parameter control are discussed in this chapter, including response surface methods, Taguchi method, meta-optimization, and machine learning methods.

Overall, this book provides a very comprehensive study on techniques aimed to improve the performance of population-based metaheuristics and it constitutes a valuable source for students, researchers, and practitioners interested in using metaheuristics for optimization. Because of that, I highly recommend its use both as a reference and as a textbook in graduate courses.

July 2024

Carlos A. Coello Coello
Editor-in-Chief, IEEE Transactions on
Evolutionary Computation
Department of Computer Science
CINVESTAV-IPN
Mexico City, Mexico

Preface

The developments of metaheuristic algorithms date back to the 1960s, and since then, metaheuristic algorithms have grown to extend, especially since the year 2000. These significant increases show that there is a growing number of researchers who have contributed to various studies, including engineering, medical, logistics, networks, artificial intelligence, and many more (Velasco et al. 2024). Most of the metaheuristic algorithms are inspired by various sources of nature (Yang 2017, Lones 2020), including the theory of evolution, creatures, sciences, and humans (Rajwar et al. 2023). Driven by the success of solving numerous problems, including NP-hard problems (non-deterministic polynomial-time hardness problems) (Tomar et al. 2023), more and more such algorithms are being proposed by numerous researchers and algorithm developers in an exponential trend. The motivation for establishing new algorithms is in hot pursuit of new journals and conference publications, as this chapter was written at the time.

Generally, metaheuristic algorithms are mostly prominent and can overcome shortfalls in many deterministic methods, specifically in nonlinear problem space or black-box problems (Kvasov and Mukhametzhanov 2018; Stojanović et al. 2017). In addition, deterministic algorithms may easily fall into local optima and be ineffective for solving multi-modal, high-dimensional, and highly constrained engineering optimization problems (Zhang et al. 2018, Rajwar et al. 2023). There are mainly three aspects that differing metaheuristic algorithms with respect to deterministic counterparts (Khajehzadeh et al. 2011):

- Search agents based on the population of individuals that reduce the likelihood of local optima. The deterministic approach searches from one point to another based on the deterministic rule.
- Metaheuristics uses fitness information of individuals instead of function derivatives or other related knowledge.
- Metaheuristics searches are probabilistic and drive the search iteratively at random with fitness dependence. This led the search toward regions that are likely to have better fitness. Such a search paradigm is referred to as stochastic.

Metaheuristic optimization algorithms are practically proficient yet disreputably hard to analyze and to understand. The stochastic nature lies in the metaheuristic properties that generally search the optimized solution based on the trial and error attempt, suggesting the fundamental characteristic of exploration and exploitation searches. The stochastic mechanism of the search procedure made the algorithm effective in finding a correct solution, thus catalyzing the intensive development of such an algorithm at an increasing rate (Dokeroglu et al. 2019).

As mentioned, most metaheuristic algorithms are inspired by nature. The algorithm was initially introduced with the idea from evolution theory in the 1980s and followed by numerous inspirations later on. Among the most currently inspired nature are the swarming behaviors of creatures, followed by physics-based and human life. To date, numerous metaheuristic algorithms are introduced with the ability to solve continuous and discrete problems in many applications (Greiner et al. 2018).

Amidst various nature-inspiring methods, there is still room for improvement for better algorithm performance with the aim of reproducibility of solutions and faster convergence. The most common terminology used in the optimization approach of metaheuristic algorithms is exploration, exploitation, or intensification and diversification, respectively. The terms are used extensively and interchangeably in the literature, especially in the vast majority of articles of leading journals related to metaheuristics (Blum and Roli 2003), indicating that the terms are significant to the field of metaheuristics optimization (Rajwar et al. 2023).

In essence, exploration refers to the algorithm's ability to explore in the search space in order to find the better solution. The exploration method has more influence in ensuring global optimum results for any optimization problem, and it is merely related to the search process with randomized characteristics. The main feature of exploration is a higher probability of finding the global solution and escaping from the local optima. However, the drawback of exploration is slower convergence and expensive computational efforts since many newly explored solutions may be far from the global optimum. On the contrary, exploitation is an approach to finding the next solution based on the information acquired from the current problem of interest with the aim of getting toward a better solution. The benefit of this method is a higher convergence rate, whereas the main drawback is a higher probability of being trapped in local optima. An example of algorithm search with exploration and exploitation mechanisms is demonstrated in Fig. 2. In this example, the Tree Physiology Optimization (TPO) algorithm (Halim and Ismail 2017) attempts to find the global optimum of the Bird function (Jamil and Yang 2013) that is composed of multiple local optima (f_{Local}) and two global optima ($f_{Global} = -106.7645$).The three-dimensional plot of the function is depicted in Fig. 1.

The algorithm is executed with a total of 100 function evaluations (FEV). The exploration and exploitation search of the algorithm's individuals is shown in Fig. 2. The figure depicts a contour plot of the test function with smaller dotted points representing the population individuals and the star points as the best candidates found so far during the search. The top figure indicates the location of individuals in the early FEV, followed by the location of individuals at the 50th FEV in the middle figure, and finally, the location of individuals at the 100th FEV in the bottom figure.

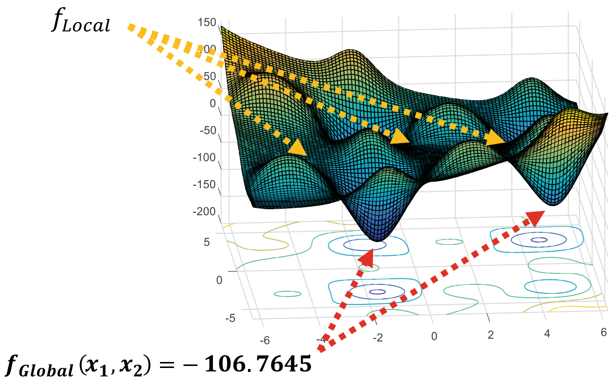


Fig. 1 Bird function with several local optima and two global optima

Note that in the initial and the middle stages, the individuals are explored throughout the search space and finally exploited toward the best solution in the last stage of the optimization.

Both strategies need to be balanced in order to achieve a good performance of optimization algorithm (Salleh et al. 2018). The tradeoff between exploration and exploitation is vital that affects the accuracy of solution and convergence speed of the algorithm. Thus, an effective tradeoff reduces the computational cost and improves the efficiency of optimization (Chen et al. 2009). To date, the question of better tradeoffs between exploration and exploitation is still open and being studied by many researchers. Moreover, there is no algorithm that can be proven to be the most effective compared to others on all possible problem instances. Some algorithms may perform better on a specific problem, while the other algorithms may able to find feasible solutions to other problems; this is known as the *no-free-lunch* (NFL) theorem (Wolpert and Macready 1997). It implies that the efficacy of an optimization algorithm is inherently tied to the specific characteristics of the problem being addressed. Therefore, a fine balance between exploration and exploitation is necessary as it improves any algorithm's efficiency and effectiveness toward finding the best solution within tolerable speed. Various methods of balancing the exploration and exploitation search mechanisms are being proposed in copious literature. Further and detailed details on the concept and fundamental principles of exploration, as well as exploitation mechanisms, are discussed in Chap. 2.

This book is dedicated to providing and reviewing the state-of-the-art modes of optimizing the exploration and exploitation tradeoffs for metaheuristic algorithms and offers a better understanding of the improvement potentials that may enhance the optimization with higher solution quality and faster convergence. For this reason, various concepts of the tradeoff's improvements are discussed and divided into chapters as summarized in Fig. 3. The discussion on the exploration and exploitation concepts in this book is generally organized according to the phases of the algorithm, which are the (1) initialization and (2) optimization phase, as shown in the

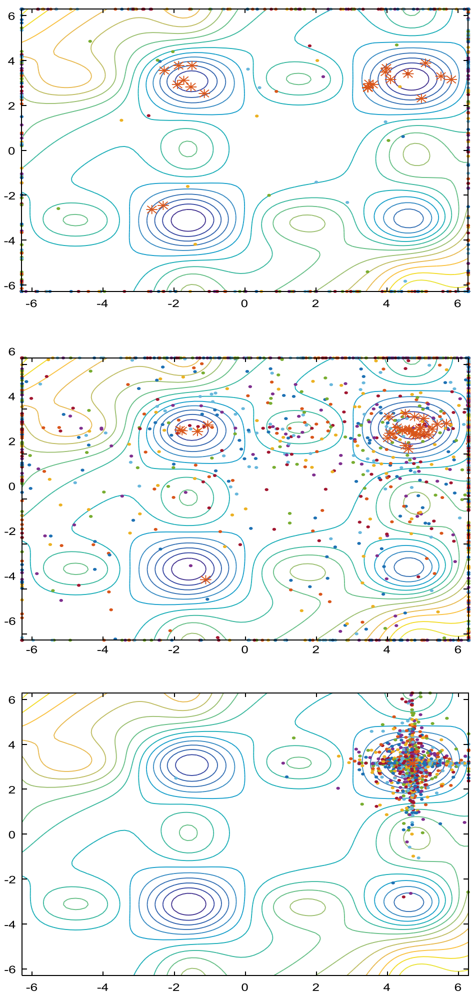


Fig. 2 Exploration and exploitation mechanism of TPO

figure with the dashed-line section. The initialization phase is referred to the beginning stage of the algorithm, whereas the optimization phase belongs to the rest of the optimization procedures after the initialization process ends until the optimization is terminated.

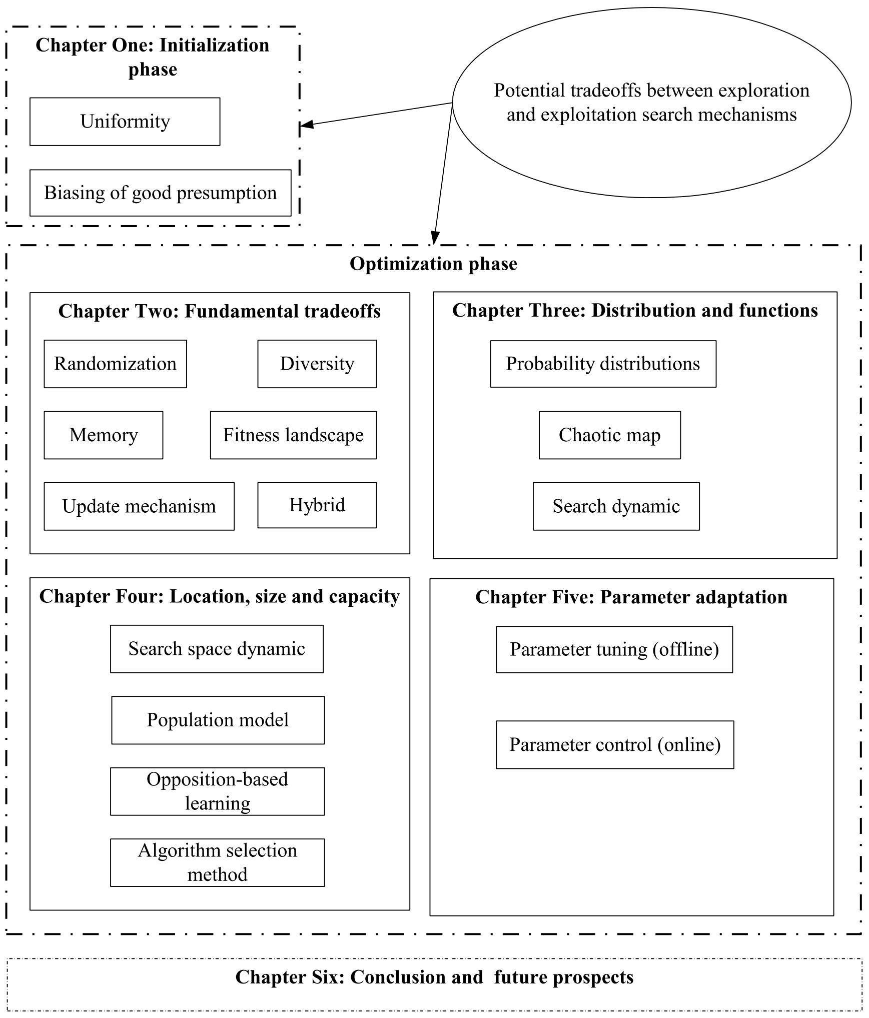

Chapter One: Initialization phase
Uniformity
Biasing of good presumption
Potential tradeoffs between exploration and exploitation search mechanisms
Optimization phase
Chapter Two: Fundamental tradeoffs
Randomization
Diversity
Memory
Fitness landscape
Update mechanism
Hybrid
Chapter Three: Distribution and functions
Probability distributions
Chaotic map
Search dynamic
Chapter Four: Location, size and capacity
Search space dynamic
Population model
Opposition-based learning
Algorithm selection method
Chapter Five: Parameter adaptation
Parameter tuning (offline)
Parameter control (online)
Chapter Six: Conclusion and future prospects

Fig. 3 Book overview

Chapter 1 discusses the initialization phase of metaheuristic algorithms and development options under the scope of exploration and exploitation mechanisms with the objective of improving or preparing the candidate solution of the algorithm before entering into the later stage of optimization search (optimization phase). The chapter focuses on two main principles: improving the uniformity of individual distributions throughout the search space and biasing the candidate solutions toward a presumed near-optimal solution. The rest of the chapter discussed various techniques that are applicable for balancing the exploration and exploitation mechanisms and are primarily concerned with the optimization phase. Nonetheless, some of the methods are also applicable in the initialization phase. Chapter 2 introduces the concept of exploration and exploitation mechanisms in more detail and further presents the essential methods for the tradeoffs, including randomization, memory features, individual update mechanism that incorporates memory, diversity maintenance for better convergence-divergence of the search agents, fitness landscape analysis, and hybrid methods.

Chapter 3 highlights potential improvements related to distributions and function driven. The distributions essentially serve as a medium for the individuals to wander throughout the search space during the optimization. Thus, proper implementation of distribution functions may improve the algorithm's effectiveness and efficiency in finding the optimum solution. Various types of distributions as discussed in the chapter are applicable to explore and exploit the search space. In addition, a unique type of distribution due to its ergodic and random characteristic; denoted as chaotic map is discussed in detail with varieties of maps that can be integrated in the optimization search process. The chapter also discusses the implementation of function dynamics that potentially drives the candidate solutions and proficient to intensify or diversify the optimization search process.

Chapter 4 discusses further options of attaining good tradeoffs using elite methods that belong to the location, size, and capacity of the algorithm during the optimization search. This includes the search dynamics that focuses on the optimization search space, population model concerning on the size of the candidate solutions, opposition-based learning that attempts to accelerate the search via opposite or relative location of individuals, and algorithm selection approach that promotes the use of multiple algorithms for problem instances.

Chapter 5 brings forward another unique and smart enhancement strategies that potentially further improve the algorithm's capability on balancing the tradeoffs between exploration and exploitation mechanisms. Due to the NFL theorem (Wolpert and Macready 1997) as well as the notion of algorithm's sensitivity to the value or setting of their parameters (Birattari 2009), proper fine-tuning of the parameters with respect to the problem instances and optimization state typically improves the optimization performance. This chapter investigates in-depth method of algorithm's parameter improvement that is based on offline- and online tuning (also termed as parameter tuning and parameter control respectively) with numerous examples from copious literatures. Among the highlighted categories for this area includes the statistical methods, hyperheuristics, machine learning as well as various adaptive and self-adaptive methods.

Chapter 6 summarizes the chapters of the book and discusses the future potential prospects that researchers in the field of metaheuristic algorithms can venture for further improvement of optimization process. The book project is divided into two volumes with Introduction, Chaps. 1–3 in the first volume, whereas Chaps. 4 and 5 to the Conclusion Chapter is discussed in the second volume.

It is hoped that this book project be an ideal textbook for vast levels of applicants from undergraduate, graduate courses, researchers to professionals or practitioners that interested in algorithm improvement or involved in algorithm development. As some of the tradeoff's concepts such as the parameter tuning and control are the forefront of the current extensive research and are far from being satisfactorily solved for multiple optimization problems, this book hopefully may serve as source of motivations and benchmark toward a better improvement of existing and future algorithms.

Seri Iskandar, Malaysia — Abdul Hanif Abdul Halim
Kolkata, India — Swagatam Das
Seri Iskandar, Malaysia — Idris Ismail

References

1. Birattari, M.: Statement of the tuning problem. In: Tuning Metaheuristics. Studies in Computational Intelligence **197**, Springer, Berlin, Heidelberg, 69–83 (2009)
2. Blum, C., Roli, A.: Metaheuristics in combinatorial optimization: Overview and conceptural comparison, ACM Comput. Surv. **35**(3), 268–308 (2003) https://doi.org/10.1145/937503.937505
3. Chen, J., Xin, B., Peng, Z., Dou, L., Zhang J.: Optimal contraction theorem for exploration-exploitation tradeoff in search and optimization. IEEE Trans. Syst. Man. Cybernetics-Part A: Sys. Hum. **39**(3), 680–691 (2009). https://doi.org/10.1109/TSMCA.2009.2012436
4. Dokeroglu, T., Sevinc, E., Kucukyilmaz, T., Cosar, A.: A survey on new generation metaheuristic algorithms. Com. Ind. Eng. **137**, 1–29 (2019) https://doi.org/10.1016/j.cie.2019.106040
5. Greiner, D., Periaux, J., Quagliarella, D., Magalhaes-Mendes, J., Galvan, N.: Evolutionary Algorithms and Metaheuristics: Applications in Engineering Design and Optimization, Mathematical Problems in Engineering, 2793762, 1–4 (2018) https://doi.org/10.1155/2018/2793762
6. Halim, A.H., Ismail, I.: Tree physiology optimization in benchmark function and travelling salesman problem. J. Intell. Syst. **28**(5), 849–871 (2017) https://doi.org/10.1515/jisys-2017-0156
7. Jamil, M., Yang, X.-S.: A literature survey of benchmark functions for global optimization problems. Int. J. Math. Modell. Num. Opt. **4**(2), 150–194 (2013) http://dx.doi.org/10.1504/IJMMNO.2013.055204
8. Khajehzadeh, M., Taha, M.R., El-Shafie, A., Eslami, M.: A survey on meta-heuristic global optimization algorithm. Res. J. App. Sci. Eng. Tech. **3**(6), 569–578 (2011)
9. Kvasov, D.E., Mukhametzhanov, M.S.: Metaheuristic vs. deterministic global optimization algorithms: The univariate case. App. Math. Comp. **318**, 245–259 (2018) https://doi.org/10.1016/j.amc.2017.05.014
10. Lones, M.A.: Mitigating metaphors: a comprehensible guide to recent nature-inspired algorithm. SN Comput. Sci. **1**(49), (2020) https://doi.org/10.1007/s42979-019-0050-8
11. Rajwar, K., Deep, K., Das, S.: An exhaustive review of the metaheuristic algorithms for search and optimization: taxonomy, applications, and open challenges. Artif. Intell. Rev. **56**, 13187–13257 (2023) https://doi.org/10.1007/s10462-023-10470-y
12. Salleh, M.N.M., Hussain, K., Cheng, S., Shi, Y., Muhammad, A., Ullah, G., Naseem, R.: Exploration and Exploitation Measurement in Swarm-Based Metaheuristic Algorithms: An Empirical Analysis. In: Ghazali, R., Deris, M., Nawi, N., Abawajy, J. (eds) Recent Advances

on Soft Computing and Data Mining. SCDM 2018. Advances in Intelligent Systems and Computing, **700**, Springer, Cham (2018) https://doi.org/10.1007/978-3-319-72550-5_3

13. Stojanović, I., Brajević, I., Stanimirović, P.S., Kazakovtsev, L.A., Zdravev, Z.: Application of Heuristic and Metaheuristic Algorithms in Solving Constrained Weber Problem with Feasible Region Bounded by Arcs, Mathematical Problems in Engineering, 1–13 (2017) https://doi.org/10.1155/2017/8306732
14. Tomar, V., Bansal, M., Singh, P.: Metaheuristic algorithms for optimizaion: a brief review. Eng. Proc. **59**(1), 238 (2023) https://doi.org/10.3390/engproc2023059238
15. Velasco, L., Guerrero, H., Hospitaler, A.: A Literature Review and Critical Analysis of Metaheuristics Recently Developed. Arch. Computat. Meth. Eng. **31**, 125–146 (2024) https://doi.org/10.1007/s11831-023-09975-0
16. Wolpert, D.H., Macready, W.G., No free lunch theorems for optimization. IEEE Trans. Evolut. Comput. **1**(1), 67–82 (1997) https://doi.org/10.1109/4235.585893
17. Yang, X.-S.: Social Algorithms. In Meyers, R.A. (eds.) Encyclopedia of Complexity and Systems Science. Springer (2017) https://doi.org/10.1007/978-3-642-27737-5_678-1
18. Zhang, J., Xiao, M., Gao, L., Pan, Q.: Queuing search algorithm: a novel metaheuristic algorithm for solving engineering optimization problems. App. Math. Model. **63**, 464–490 (2018) https://doi.org/10.1016/j.apm.2018.06.036

Acknowledgments

First and foremost, for the first author, all praises are due to Allah, the God almighty, for the ultimate showers of blessings throughout the research work and to complete this book project successfully.

The first author wishes to express his sincere and deep gratitude to his respectful co-authors AP Dr. Idris Ismail from Universiti Teknologi PETRONAS, Malaysia, and Prof. Swagatam Das from Indian Statistical Institute, India, for invaluable guiding, mentoring, and supervising during the flow of this book project. Their visions, dynamisms, and motivations have deeply inspired him with vast methodologies related to metaheuristics and evolutionary algorithms, as well as research ideas that are significant to state-of-the-art applications.

The first author is also extremely grateful to his loving mother, Nuzhat Jahan Wasi Ahmad, for spiritual motivation and to his late father, Abdul Halim Abdul Satar, who went to meet the Lord on March 2023. May God accept all of his deeds, take good care of him, and have mercy upon him as he brought me up—Ameen.

The first author conveys his special admiration also to siblings and family as well as family in-laws who supported me throughout the journey. Last but not least, He would like to convey the profound appreciation to his dear wife, Aidahanie Baharuddin, and his sons, Abdul Hazim Abdul Hanif and Abdul Hariz Abdul Hanif, for the unlimited encouragement, sacrifices, and inside-out motivation during this ambitious period.

Contents

Chapter 1
Algorithm Initialization: Categories and Assessment

This chapter dedicates the essential principle and improvement options for metaheuristic algorithms during the initialization of the optimization search. All metaheuristic algorithms must follow the initialization procedure before the primary optimization search process, as highlighted in Fig. 1.2. As also understood among algorithm developers and researchers, good tradeoffs between exploration and exploitation mechanisms are fundamental in metaheuristic algorithms to ensure a good optimization process concerning solution quality and faster convergence. The initialization phase is one of the main factors contributing to a better exploration–exploitation balancing mechanism. This chapter discusses the initialization phase of metaheuristic algorithms and development options proposed in numerous pieces of literature to improve the algorithm later in the optimization search. The outline of this chapter is summarized in Fig. 1.1.

The chapter begins with the fundamentals of initialization in metaheuristic algorithms with the idea of better initial solutions that may lead to superior algorithm performance in solving problems, especially with high complexity and dimensions. Based on the overall methods of algorithm initialization, this chapter then divides the main category of initialization into two perspectives: the uniformity of solution distribution and biasing towards a good presumption of initial solutions. Each category is discussed separately in Sects. 1.2 and 1.3, respectively. Several sub-categories are further addressed in each section of the initialization method. The chapter ends with concluding remarks based on the findings from both main methodologies and their respective advantages and disadvantages.

A. H. Abdul Halim et al., *Into a Deeper Understanding of Evolutionary Computing: Exploration, Exploitation, and Parameter Control*, Emergence, Complexity and Computation 50, https://doi.org/10.1007/978-3-031-74013-8_1

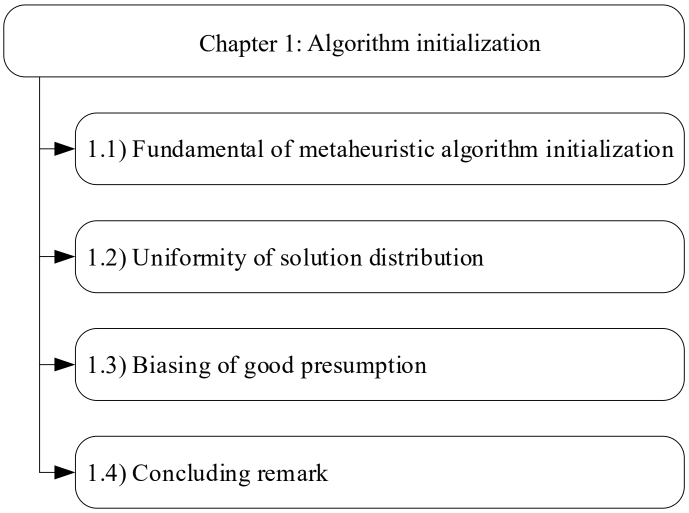

Chapter 1: Algorithm initialization
1.1) Fundamental of metaheuristic algorithm initialization
1.2) Uniformity of solution distribution
1.3) Biasing of good presumption
1.4) Concluding remark

Fig. 1.1 Summarized outline of this chapter

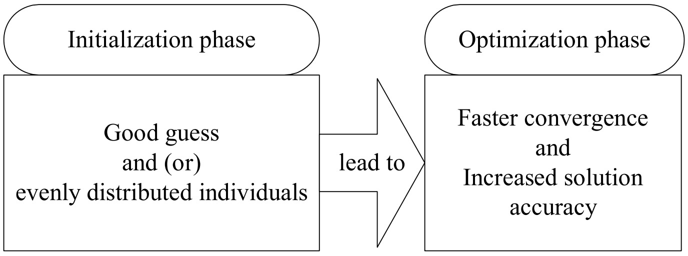


Fig. 1.2 Good characteristics of the initial population

1.1 Fundamental of Metaheuristic Algorithm Initialization

The initialization of parameters for the metaheuristic algorithms, especially for the population-based algorithms, plays a vital role in the overall algorithm performance. The method of metaheuristic algorithm initialization has a profound impact on its exploration and exploitation abilities. Improper diversification of initial solutions may lead to premature convergence as the search progress is trapped in the local optima. Alternatively, initialization with low quality solutions may consume a larger number of iterations to converge. In this regard, the diversity and good quality of the initial solutions need to be preserved. In ideal circumstances, the initial solution is usually independent of the final optimal solutions converged by the algorithm. This may apply only to problems such as convex optimization or linear programming. Nonetheless, this dependency may be tricky in other cases, especially for real-world problems that are nonlinear and complex. In addition, the degrees of dependency can differ by different algorithms and problems (Kondamadugula et al. 2016; Li et al. 2020) even within the same problem domain but with different types of instances such as truss-structure optimization (Contreras-Bejarano and Villaba-Morales 2024). In essence, the algorithm initialization is related to the number of individuals or candidates, sometimes termed search agents of the population. Due to the nature of the Markov process, the locations of current individuals are highly associated with the previous position, formulated as follows:

$$x_j(i + 1) = x_j(i) + \Delta x_j(i), \tag{1.1}$$

where $x_j(i)$ is the state of jth individual in the ith step or previous step and Δ is the step function that varies by different introduced algorithms. Therefore, as simplified in Fig. 1.2, individuals with either good educated guess or equivalently scattered throughout the search space may enhance the search process in the next step of the algorithm iterations especially in the population-based algorithms towards better convergence rate and optimum solution.

Initialization of individuals that are not uniformly scattered or with locations further from the global optimum definitely led to higher algorithm computation as well as the potential of premature convergence as the algorithm is not able to locate the optimum solution within a prescribed number of iterations. In the initialization phase, the location of each search agent by default (in most of metaheuristic algorithms) is usually defined randomly within a defined lower and upper boundary of problem space with $X = LB + rand(UB - LB)$, where LB and UB are the lower and upper bound, respectively, and $rand(x)$ as the randomization move that creates a random value around the x individuals. Besides this default random localization of individuals, other initialization methods are introduced in the literature that potentially improve the sound characteristics of the initial population.

The main motive for proper initialization is due to the unknown optimum solution before the optimization search and the stochastic characteristic of the algorithm. Moreover, improper population size and search iterations with respect to the search dimension may bias the search, potentially leading to sub-optimal solutions. Consider test function problems with either multimodal or deceptive landscape. The population-based algorithm may easily be trapped in the local optimum merely due to the vast area of local optima, as shown in Fig. 1.3a. In the deceptive landscape from Fig. 1.3b, however, the challenge is slightly different, in which the problem consists of the deceptive attractor as the local optimum and one global optimum (Whitley 1991; Deb and Goldberg 1994). Improper population initialization may attract the search candidates to the deceptive area just after a few iterations, which is far from the global optimum. Therefore, the population initialization needs to be properly rationalized to ensure that the individuals cover all possible search space or to estimate the better optimal location to converge towards the true optimum solution.

Due to this reason, many works attempted some efforts on the improvement of initial stage with better diversity of the initial population (exploration) or concentrated in the promising region (exploitation) in order to obtain guaranteed optimal

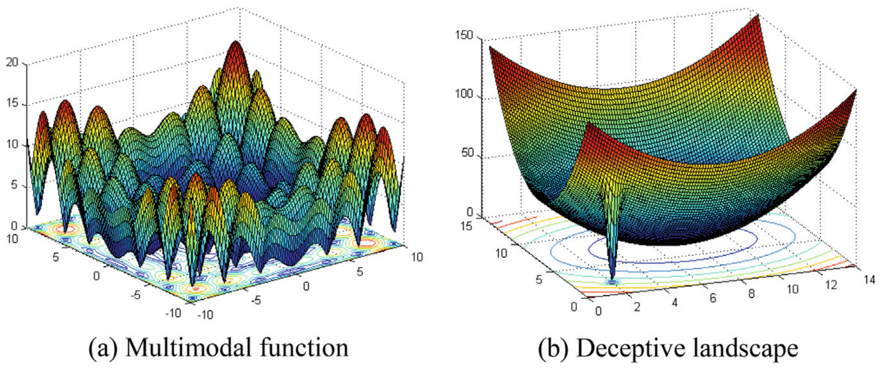


Fig. 1.3 Example of test functions

solutions. There are also several published review papers related to population initialization methods in metaheuristic algorithms, such as Zhang et al. (2023), Sarhani et al. (2023), Agushaka and Ezugwu (2022a), Hasanzadeh and Keynia (2021b) and Kazimipour et al. (2014). Based on these review papers, the initialization methods can be categorized into several types. Kazimipour et al. (2014) differentiate the initialization methods into three major perspectives, including randomness, compositionality, and generality. Each type is defined based on the manner in which the population of individuals is initialized. The randomness is related to the randomized initialization method. Depending on the nature of random initialization, this branch is further divided into two types: deterministic that derived the randomness based on deterministic rule and stochastic that is derived from some stochastic features. Under deterministic randomness, another two sub-branches are defined: low discrepancy randomization and uniform experimental design (UED). The randomness is supplementary and divided into random number generator and stochastic for the stochastic complements. The second major perspective, the compositionality is sub-categorized into non-composite (which is based only on one step method) and composite (which is based on two or more steps for the initialization purpose). The third major perspective, the generality, is clustered into generic and application specific. The generic method is referred to as the initialization technique that can be directly applied to all kinds of optimization problems, whereas the application-specific method refers to the initialization method special only for a specific problem and is not applicable to all optimization problems.

Hasanzadeh and Keynia (2021b) also characterized the initialization categories similar to the previous review but with further elaborations of each specific method, such as the type of number generators, chaotic maps, opposition-based learning, and many more. Another recent review paper (Agushaka and Ezugwu 2022a) categorizes the initialization methods into eight variants, including pseudo-random generator, quasi-random generator, probability distributions, hybridization with other metaheuristics, chaos, ad-hoc knowledge, Lévy flights, and others.

The metaheuristic algorithms can also be initialized under the machine learning approaches. In essence, Machine learning (ML) is a branch of artificial intelligence that uses algorithmic and statistical approaches to give computers the ability to learn from data and improve their performance in solving tasks without being explicitly programmed for each one (Bishop 2006). There are numerous learning methods under the ML umbrella. Proper ML techniques can be applied in the initialization phase of metaheuristic algorithms with the objective of maintaining the diversity of initial solutions as well as with good quality. The ML contribution in the initialization of metaheuristic algorithms can be divided into three main strategies (Karimi-Mamaghan et al. 2022), which are discussed below.

(1) Complete generation: The ML approaches can be implemented to construct complete initial solutions and replace the default individual generation strategies (such as pseudo-random generation) from an empty solution. Among the ML approaches under this category are the Reinforcement Learning (RL) or precisely the Q-learning method, Artificial Neural Network (ANN), and Opposition-based Learning (OBL).
(2) Partial generation: ML approaches can be used to generate partial initial solutions via the prior knowledge of good solutions from past experience or data. The remaining solutions can then be generated using other non-ML initialization methods. In this case, ML extracts knowledge from previous good solutions and integrates it into the new initial solutions. Among the ML techniques under this category is the Case-Based Reasoning (CBR) and Association Rule (AR).
(3) Decomposition: This category can be carried out in the data or search spaces. In data space decomposition, the ML approach can be used to decompose the data or population into several sub-spaces to generate the initial solutions with less computational cost. In the search space decomposition, the ML approach can be used to diversify the initial solutions over the search space by prioritizing the region of the search space that potentially leads to the optimum solution. Among the examples of applicable ML techniques is the Multi-Armed-Bandit (MAB), where each arm corresponds to the search space region; thus, each arm learns and reveals which section of the search space is worth exploring.

This chapter compiles and summarizes the initialization approaches introduced in the literature, taking into consideration the review of the above papers. Unlike the categorization presented by Kazimipour et al. (2014) as well as Hasanzadeh and Keynia (2021b), this section divides the initialization methods into another perspective of view driven by the objective of the initialization as highlighted in Fig. 1.2, which are either oriented towards (1) uniformity of distribution or (2) biasing of reasonable presumption. Depending on the nature of categorization, the method under the ML perspective is also integrated into both of the groups. Each of the categories is discussed briefly in the following subsections. Figure 1.4 summarizes the main branch and sub-categories of the initialization methods. It is also to be noted that not all initialization methods are applicable in each metaheuristic algorithm. The sensitivity of algorithms is problem-dependent, hinting that the option of an initialization

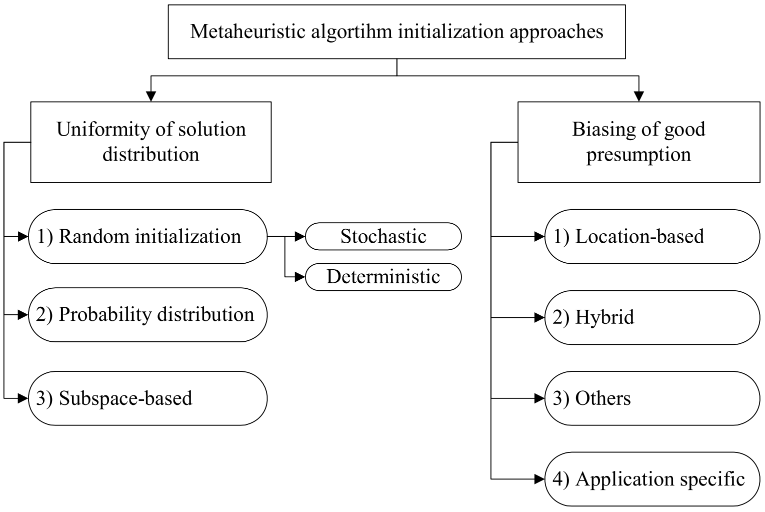

Metaheuristic algortihm initialization approaches
Uniformity of solution distribution
1) Random initialization
Stochastic
Deterministic
2) Probability distribution
3) Subspace-based
Biasing of good presumption
1) Location-based
2) Hybrid
3) Others
4) Application specific

Fig. 1.4 Some of the main initialization approaches

approach potentially substantially influences the overall performance for particular problems (Garcia et al. 2023; Agushaka et al. 2023).

1.2 Uniformity of Solution Distribution

As mentioned in the first section of this chapter, most of the standard and traditional metaheuristic algorithms are initialized with random distribution to ensure that the individuals are scattered in the search space such that some or one of them may drive the search towards the optimum path as the optimization phase started. The uniformity of solution is referred to as the degree of evenness of the individual scattered in the search space. The more even the scatter is, the more uniform the distribution. Apart from the default random initialization, there are several methods proposed in the literature that tried to improve the uniformity of the distribution. This section compiled three main categories related to uniformity including random initialization, probability distribution initialization, and sub-space-based initialization. Each of them is discussed with examples from the literature.

1.2.1 Random Initialization

A random number generator is defined as a sequence of numbers that portray the properties of complete unpredictability, incompressibility, and irregularity (Ergün and Özoğuz 2010). According to Kazimipour et al. (2014) and Hasanzadeh and Keynia (2021b), random initialization can be further divided into stochastic and deterministic depending on the initial seed generation for the random generator.

Stochastic methods

Stochastic random initialization is defined as a random sequence that generates different populations with different initial seeds. The corresponding initial seed mainly causes the randomness and is usually provided by an external source. The stochastic random can be further divided into the pseudo-random number generator and the chaotic number generator. Each of these sub-groups is discussed briefly in the following sub-topic.

Pseudo-random initialization

The pseudo-random initialization is the most applied method in numerous meta-heuristic algorithms. This method (or also denoted as the Monte Carlo method) generates uniform random sequences based on the uniform probability distribution in order to specify the initial location of individuals of the population. The pseudo-random generators can be evaluated based on several indicators such as the cycle time, which is defined as the smallest integer that the pseudo-random generator repeats the previously defined numbers and equi-distribution, which is also referred as the points that have equal frequency. Another factor of evaluation is the empirical comparison of diffusion capacity between the old and the new random generator (Panneton et al. 2006).

There are various types of pseudo-random generators, such as KISS (Marsaglia and Zaman 1993), Mersenne Twister algorithm (Matsumoto and Nishimura 1998), Lagged Fibonacci generator (Knuth 1998), and WELL algorithm (Panneton et al. 2006). These pseudo-random generators need to pass several stringent statistical tests, such as DieHard, TestU01, and NIST, to be acknowledged as random number generators (Lorek et al. 2020). An example for Mersenne Twister algorithm, the algorithm is asserted to attain longer period of $2^{19937} - 1$ and larger distribution properties of 632-dimension of equal distribution, which revealed faster and better randomness than the other pseudo-random generators and passed the DieHard test as shown in the findings from Matsumoto and Nishimura (1998). This Mersenne Twister algorithm is a 64-bit algorithm; there is also a 32-bit Mersenne Twister algorithm applicable as a pseudo-random generator. Both generators are the same in terms of flow and basic logic but mainly differ by the number of integers for generating numbers: 32-bit uses unsigned 32-bit with respect to unsigned 64-bit integers for 64-bit Mersenne Twister algorithm. The advantage of the later pseudo-random generator is better statistical properties with a higher degree of uniform distribution and higher quality of random sequence; however, it requires more memory and computational resources due to its large state size (Gulić and Žuškin 2023).

An example of an application using the Mersenne Twister generator in algorithm initialization can be referred to Sacco and Rios-Coelho (2019). The paper combined this random generator with opposition-based learning in Differential Evolution (DE) (Storn and Price 1997) initialization. Figure 1.5 demonstrates the distribution of the

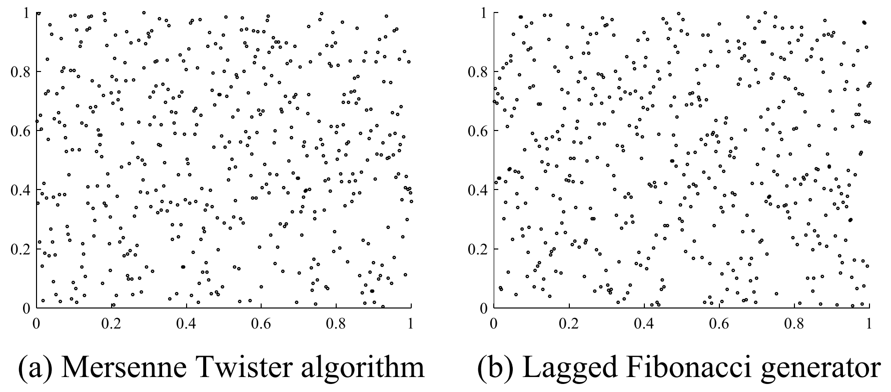


Fig. 1.5 Pseudo-random generator

Mersenne Twister algorithm and Lagged Fibonacci generator over 500 individuals. As shown in the figure, the distribution is randomly scattered, however some clusters of points are observed, and some areas are empty.

Despite its randomness and the major attractions of metaheuristic algorithm developer, this method did not efficiently cover the entire search space with the high inconsistency of random sequences, as also indicated in Fig. 1.5. The inconsistency of random sequence is referred as the random distributed solutions are not really random throughout the search space, which then lead to convergence of the algorithm towards local optimum after a number of iterations (Niederreiter 1992). After a few iterations, the individuals generated by the pseudo-random will be clustered rather than scattered around the search space. This indicates the shortcomings of a pseudo-random generator for population initialization.

In an improvement attempt to use the random generator for population initialization, Pan et al. (2014) proposed an adaptive randomness, which modifies the random initialization by controlling the individuals of the initial population. The method aims to ensure that the individuals are not too closely located to the other individuals and are distributed uniformly throughout the search space. For this purpose, the method is divided into two sets of individuals: partial initial population (PP) and trial individuals (ST). PP is the partial population from the total number of P individuals in the population with a with $PP \subseteq P$. The trial individuals are composed of a set of k individuals with $ST \cap PP = \emptyset$ and each trial individual is randomly chosen from the search space that is not yet been added into PP. The Euclidean distance of each generated trial individual is then compared with the individuals in PP, and the trial individual with the farthest distance is added into PP. This process is repeated until the number of individuals in PP is equal to the total number of individuals in the P population. The author applied the proposed method in the DE algorithm and compared it with other DE-initialized variants, such as opposition-based and standard pseudo-random generators, for solving 34 benchmark test functions. The proposed method outperformed most of the test functions with a better success rate and final solution. In the scalability test, the proposed method showed similar performance as the dimensionality D is increased from $0.5D$ to $2D$. However, the performance deteriorates in some functions as the dimensionality increases.

Other example attempted to improve the randomization is using higher quality of pseudo-random generator that exhibits larger period of repeating number of sequences. Gulić and Žuškin (2023) integrated two pseudo-random generators, namely 64-bit and 32-bit of SIMD-oriented Fast Mersenne Twister (SFMT) in hybrid of Genetic Algorithm (GA) (Holland 1972) and African Buffalo Optimization (Odili and Kahar 2016) for solving container relocation problems.

Another attempt to develop better uniformity from a pseudo-random generator is based on a distance matrix from Khajeh et al. (2019). In this approach, each individual point is assigned within a defined distance such that each of them is separated evenly. The paper devises five steps that modify the randomly generated points into better uniform distributed individuals as follows:

(1) Generate a matrix $m \times n$ randomly according to the domain of variables with m as the number of individuals and n as the number of variables.
(2) Compute the distance of each individual and saved as a matrix of $m \times m$ representing the ith row and jth column correspond to the ith and jth distance of individual $d_{i,j}$ with the following distance formulation:

$$d_{i,j} = \sqrt{\sum_{k=1}^{n} \left(x_{i,k} - x_{j,k}\right)^2}. \tag{1.2}$$

(3) The radius of coverage r_{cov} based on the size of the search space S_{tot}. given in the equations below with UB_k and LB_k as the upper and lower bound of k^{th} variable.

$$r_{cov} = \sqrt[n]{\frac{S_{tot}}{m}}. \tag{1.3}$$

$$S_{tot} = \prod_{k=1}^{n} UB_k - LB_k \tag{1.4}$$

(4) The location of individuals with a distance less than the coverage radius is eliminated.
(5) New individuals are then generated and added to the matrix with distance greater than the radius coverage.

The method is summarized as illustrated in Fig. 1.6. At the beginning of the step, the individuals are randomly scattered throughout the search space (Fig. 1.6 left). Then, the individuals that lie within the defined radius are eliminated in Fig. 1.6 (mid), and new individuals that are located greater than the defined radius are re-initialized (Fig. 1.6 right). The last figure indicated more uniformly distributed individuals compared to the beginning state of the population.

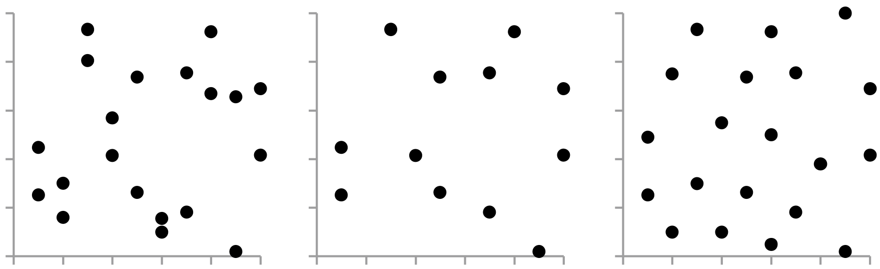


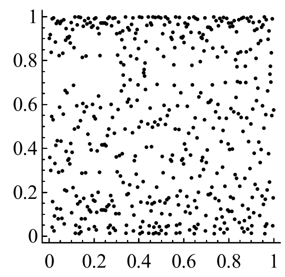
Fig. 1.6 Illustration of uniform distribution concept adopted from Khajeh et al. (2019)

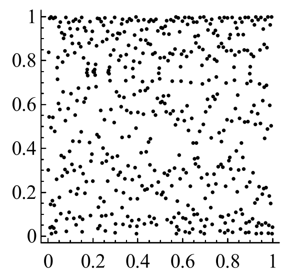
The proposed method from Khajeh et al. (2019) is demonstrated in Particle Swarm Optimization (PSO) (Kennedy and Eberhart 1995) population initialization and compared with standard PSO of pseudo-random initialization in solving benchmark test functions. Their results indicate a better convergence rate towards the global optimum for different population sizes.

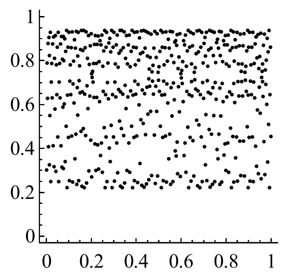
Chaotic function

The chaotic function is composed of ergodic properties that are sensitive to the initial conditions and become chaos in the later stage depending on the specified value of the control parameter. This chaotic characteristic introduces unpredictable randomization patterns in the exploration process, which serves as an advantage for searching effectively in the search space. The chaotic function is usually denoted as a chaotic map with a function or multiple functions resembling a sequence map from the first step towards the n number of steps. The chaotic map has a similar randomness property with better statistical and dynamical properties (Gandomi and Yang 2014). In the improvement of the exploration and exploitation mechanisms of metaheuristic algorithms, the chaotic maps are applicable in the initialization phase and optimization search process. Since the objective of integrating chaos in the initialization phase is to enhance the uniformity of the individuals, a proper function that is able to distribute the initial solutions regularly is crucial. The selection of chaotic maps depends on the sequences that the map characterizes. This feature is determined by the initial seed (initial value of chaotic sequence) and its specific control parameter. Different initial seed and control parameters resulted in variations of sequences due to the ergodic behavior of the chaotic function. Examples in Fig. 1.7 illustrate the difference of selection for initial seed (x_0) and control parameter (μ) of logistic map with 500 individuals. The first two charts in Fig. 1.7a and b demonstrate the differences of individual locations by different selection of x_0. The second and third charts compare the individual distributions by different μ but with the same x_0, showing also different pattern of distribution and scale. The lower chart illustrates the bifurcation diagram of logistic map generated with 10,000 numbers of iterations within $2.6 \leq \mu \leq 4$.

As a brief background, the bifurcation diagram is a nonlinear dynamical system used to demonstrate the asymptotic evolution of the map or oscillator represented


(a) $x_0 = 0.9$ with $\mu = 3.99$


(b) $x_0 = 0.3$ with $\mu = 3.99$


(c) $x_0 = 0.3$ with $\mu = 3.75$

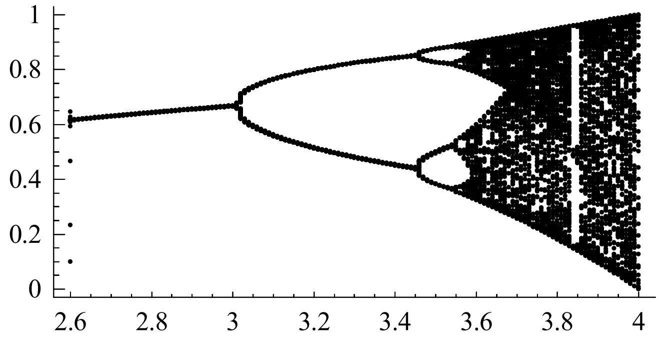

(d) Bifurcation diagram with control parameter $2.6 \leq \mu \leq 4$

Fig. 1.7 Individual sequences and bifurcation diagram of logistic map

with a sample of steady-state values of one variable x_i on the vertical axis with respect to the control parar on the horizontal axis. The dynamic behavior of the variable usually changes from fixed point, periodic to quasi-periodic (represented with path of lines), chaotic and may also to hyperchaotic (represented with random-like pattern). Detailed description of the bifurcation diagram can be referred to in Chap. 3.

It is, therefore, necessary to select good values of the initial seed and control parameter in order to obtain better uniformity of the individuals. There are various chaotic maps that are applicable in the population initialization. Some of them are listed in the Table 1.1. In this table, the individual sequences map is selected with the initial seed, x_0 equals 0.7 and defined control parameters specific for each chaotic map for 500 individuals. The table also lists the dimensionality (**D**) of the function either in one- or two-dimensions. The individual sequence for each chaotic map is illustrated in Figs. 1.8 and 1.9.

It is to note that some of these chaotic maps are normalized within [0, 1] in order to locate the individuals in this specified range. Some of these chaotic maps distribute the individuals within [−1, 1] such as cosine map, square map, and Hénon map. Others such as the Chirikov map and the Ikeda map scatter the individual points within other range of values.