*Academic Journal of Nawroz University (AJNU), Vol.12, No.1, 2023*



https://doi.org/10.25007/ajnu.v12n1a1660

# A NEW PERSPECTIVE OF METAHEURISTIC ALGORITHMS

Pawan Shivan Othman[1], Rasheed Rebar Ihsan[2] and Reving Masoud Abdulhakeem[3]

[1,3]Department of Computer Science, Nawroz University, Duhok, KRG -Iraq

[2]Department of Computer Engineering and Communication, Nawroz University, Duhok, KRG -Iraq

## ABSTRACT

Optimization is the art of modeling in order to produce the optimal outcome under the given conditions. The objective of optimization is to maximize or decrease the consequences that best satisfy technological and management procedures. In view of the findings, this paper provides a brief survey of methods for examining the optimization problem space, illustrates the mechanics of metaheuristic and developmental calculations, and defines their connection to constructing optimization problems. In addition to covering the encoding of metaheuristic and developmental calculations and the management of constraints, this paper also delves into the periods of introductory or provisional arrangements, the iterative determination of arrangements, and the assessment of the execution of metaheuristic and developmental calculations. All meta-heuristic and developmental calculations are shown to share a single calculation with their respective phases highlighted.

KEY WORDS: Optimization, Metaheuristics, Local search, Evolutionary Algorithms, New Solutions.

## 1. NTRODUCTION

This In metaheuristic algorithms, the term meta- signifies "beyond" or "higher level." In general, they outperform simple heuristics. All metaheuristic algorithms employ a compromise between local search and global exploration. Frequently, the diversity of answers is achieved through randomness. Despite the prevalence of metaheuristics, the literature does not have an agreement meaning of heuristics and metaheuristics. Some researchers confuse the terms 'heuristics' and 'metaheuristics.

However, current agreement has it that any stochastic algorithm that uses randomization and global exploration is a metaheuristic. Transitioning from local search to global search can be accomplished by randomization [5].

As a result, nonlinear modeling and global optimization can benefit from the use of virtually any meta heuristic technique. Both metaheuristic and developmental calculations are general-purpose methods applicable to a wide variety of problems. In computing, the term "algorithm" is used to refer to a specific sequence of steps taken to solve a problem [2]. The iterative processes or steps in the calculations are completed once an expression is reached, as illustrated in Figure1.

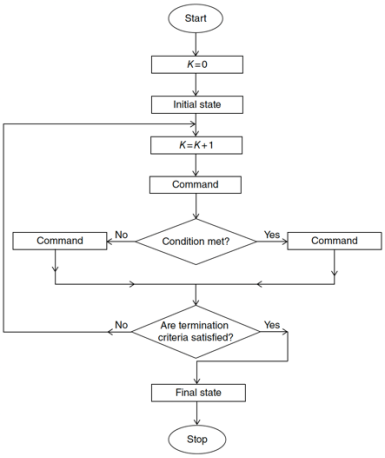


Fig. 1. General diagram of a simple meta heuristics, the variable K represents the counter of iterations.

The search strategy is necessity to address numerous problems with their respective decision spaces has controlled to the creation of algorithms encouraged by natural events or emulating human intellect.

Some critics have contended that some newly produced algorithms, despite being inspired by novel events, are in fact repeats of algorithms that have been invented previously (e.g.: S rensen, 2013). Other researchers, however, have proven distinctions between newly designed and older algorithms (e.g. Saka et al., 2016). All meta-heuristic and evolutionary algorithms share properties with the generic method presented in this investigation.

However, there are substantial variations between the two, such as the invention of new and original ideas and the selection of new solutions. Each algorithm examines the decision space in a unique manner, and their efficacy in solving certain issues differs. Single-modal decision spaces can be efficiently searched by algorithms employing strong selection pressure in their selection step and emphasizing close search for the best solutions discovered.

Nevertheless, their efficiency falls when locating multimodal decision spaces with several local optimal due to the possibility of becoming entrapped in the local optimal [3].

The algorithms, on the other hand, use a low-pressure selection strategy and randomly explore the decision space, thereby conducting deep searches throughout the whole choice space and lowering the danger of near-trapping. optima, enhancing their problem-solving efficiency in multimodal decision environments [1].

To handle problems with unary decision spaces where there is no risk of getting trapped around the local optimal point, these algorithms conduct a huge number of calculations [12]. In addition, the majority of meta-heuristic and evolutionary algorithms have performance-tuning parameters. The selection of these criteria impacts their search technique. Other algorithms automatically set their parameters. For these reasons, it is necessary for users to understand the operational principles of each algorithm, since they must select a suitable algorithm to address specific optimization problems.

## 2. CLASSIFICATION OF METAHEURISTICS

A classification of metaheuristics Metaheuristic algorithms seek to identify the optimal (feasible) answer to an optimization issue among all conceivable alternatives. In order to achieve this, they analyze probable solutions and apply a series of operations to them in order to discover other, superior answers. Metaheuristics act on a representation or encoding of a solution, an object that can be kept in computer memory and is easily manipulable by the metaheuristic's different operators. On the basis of how solutions are altered, three major categories of metaheuristics can be separated. Local search metaheuristics make incremental modifications to a single solution. Constructive metaheuristics assemble solutions from their constituent components. Population-based metaheuristics repeatedly combine existing solutions to create new ones. Nevertheless, these groups are not mutually exclusive, and numerous metaheuristic algorithms integrate concepts from multiple classes. These techniques are known as hybrid metaheuristics [7].

### 2.1 Local search

Number Local search techniques (which terminate at a local optimum) and associated meta heuristic strategies (which alter and direct local procedures to explore the solution space more completely) have been the subject of extensive scientific research over the past decade. For more than two decades, there have been two dominant "meta models" for heuristic techniques: those based on "single stream" trajectories and those based on "multiple stream" trajectories, the latter of which seeks to produce new solutions from a collection (or population) of solutions [11].

The contrast is fundamentally identical to that between serial and parallel algorithms, with the exception that population-based methods can also be employed in a serial fashion, as in the case of a serial simulation of a parallel approach. As expected, there are some overlaps between the best processes of these two categories. Traditional population-based methods, however, have frequently been designed from a narrower perspective that excludes strategies typically applied with single stream methods [10] . Consequently, hybrid techniques are frequently used to describe more contemporary systems that combine elements of both methods.

### 2.2 Constructive metaheuristics

Constructive techniques develop one or more combinations incrementally, beginning with an empty combination and progressively adding components until the combination is full. These approaches are referred regarded as model-based in [Zlochin et al, 2004] because they employ a model (typically stochastic) to select the next component to be added to the partial combination at each iteration. GRASP, an acronym for greedy randomized adaptive search process (Feo and Resende, 1995), mitigates the greediness of a constructive metaheuristic through the use of randomization [4].

A greedy algorithm constructs a combination slowly, beginning with an empty combination and gradually completing it by adding components. At each stage, the component to be added is selected in a greedy manner, i.e., by selecting the component that maximizes a problem-dependent heuristic function that locally

evaluates the benefit of adding the component in terms of the objective function. Typically, a greedy algorithm has a very low time complexity since it never returns to a previous decision. The final combination's quality is determined by the heuristic [3].

2.3 Population-based metaheuristics

The population-based method can finish the search process in parallel using numerous initial points (Beheshti, Mariyam, & Shamsuddin, 2013). The advantage of the population-based algorithm is that it effectively provides the search space for exploration. This approach is appropriate for global searches since it is capable of both global exploration and local exploitation. Ant colony optimization (ACO), particle swarm optimization (PSO), genetic algorithm (GA), evolution strategies (ES), intelligent water drop algorithm, and artificial bee colony are examples of population-based algorithms) [12].

2.4 "Hybrid" metaheuristics

The concept of hybrid metaheuristics has only lately been generally accepted, despite the fact that the idea of merging different metaheuristic approaches and algorithms dates back to the 1980s. Today, there is general acceptance on the benefits of mixing components from several search methodologies, and the trend of designing hybrid algorithms is ubiquitous in the disciplines of operations research and artificial intelligence [6].

## 3. TAXANOMY OF METAHEURISTIC AND EVOLUTIONARY ALGORITHMS

Categorization of Heuristics and Shortcuts When presented with an optimization problem, metaheuristic algorithms attempt to find the best possible solution by comparing and contrasting all of the possible ones. They accomplish this by examining likely solutions and performing a series of operations on them in an effort to unearth new, superior answers. Metaheuristics perform their tasks by manipulating a representation or encoding of a solution, an item that can be stored in computer memory and is amenable to the various operators that make up the metaheuristic. Three broad classes of metaheuristics can be distinguished according on the methods they use to modify solutions. Metaheuristics that perform local searches iteratively improve upon an existing answer. Solutions are pieced together using constructive metaheuristics. Metaheuristics based on sampling a population iteratively combine and provide new solutions. Still, the categories are not exclusive, and many metaheuristic algorithms combine ideas from several of them. Hybrid metaheuristics are the name given to these methods [11].

3.1 Initial State

A large number of variables is used as a starting point by both metaheuristics and evolutionary algorithms. This starting point could be manually entered, randomly generated, or calculated using algorithms and equations. This initial state might be specified, randomly generated, or derived deterministically using equations and algorithms.

3.2 Iterations

Computation executes actions iteratively during the search for a structure. The emphasis of developmental or metaheuristic computations begins with one or more introductions to the optimization problem. The subsequent sequence of activities produces an unused solution (s). A focus concludes when a feasible modern layout is produced. The unused created solution(s) are considered as an introduction solution(s) for the next computation cycle.

3.3 Final State

After satisfying the selected end criterion, the calculation ceases and reports the leading or most recently developed solution(s) to an optimization problem. The end criterion is specified in three unique forms:

(1) the number of cycles,

(2) the improvement edge of the approval of arrangement between continuous cycles, and

(3) the calculation time for optimization. The primary measure specifies a specified maximum number of cycles that the calculation may execute. The instant measure establishes a limit for advancing the arrangement between successive stages. The third criterion terminates the calculation after a specified runtime and writes the leading arrangement available at that time.

3.4 Initial Data (Information)

Two forms of initial input are distinguished:

First one the knowledge on the optimization problem, which is required for reenactment;

And the other are the parameters of the calculation, which are required for its execution and may need to be calibrated.

The second type of preliminary information is required to calibrate the layout computation in order to solve an optimization problem. Almost all metaheuristic and

developmental computations need adjusting parameters. To realize its effective implementation, it is necessary to select the calculation's parameters properly. The terminating criterion, for instance, is a user-specific algorithmic parameter. If the terminating measure is not chosen appropriately, the calculation may not focus on the global answer. On the other side, the calculation may take an abnormally lengthy time to complete [10].

### 3.5 Decision Variables

Choice factors are those whose values are computed at the conclusion of the computation, and whose values are detailed as an arrangement of an optimization problem upon reaching the stopping point. Metaheuristic and developmental computations begin with the initialization of choice factors and the recalculation of their values during calculation execution [ 1].

### 3.6 State Variables

The state factors have a relationship with the selection factors. In actuality, the values of the state factors vary based on the selection criteria.

### 3.7 Constraints

Obligations limit the space of possible solutions to an optimization problem and are addressed in metaheuristic and evolutionary computations. In actuality, these factors affect the attractiveness of every possible arrangement. After assessing the objective task and state factors associated with each arrangement, the constraints are calculated and indicate the criteria that must be met for every conceivable arrangement to be possible. If the prerequisites are met, the arrangement is acknowledged and referred to as a feasible arrangement; otherwise, the arrangement is discarded or changed.

### 3.8 Fitness Function

The value of an objective work is not always the level of desirable quality chosen for a composition. For example, the computation may scan a changed shape of the goal work by the expansion of penances that keep a strategic distance from the violation of constraints; in this case, the altered work is referred to as the fitness function. The fitness function is then used to evaluate the desirability of potential configurations.

### 3.9 Selection of Solutions in Each Iteration

The term "selection" is used to refer to the process of picking one or more solutions from a pool of possible ones during an algorithmic calculation. While many meta-simulation and evolution algorithms make use of previously found solutions to develop new ones, some do not. Many already-existing solutions are disregarded by the selection operators. Selecting solutions from the existing set might be random or deterministic depending on the algorithm being used.

Some algorithms use all previously found solutions to develop new solutions, but only some of these solutions are ultimately accepted. When doing the search, only candidates with a decent level of accomplishments will be evaluated.

Some of the newly developed solutions can be chosen at random or according to some specified criteria. In most cases, the algorithm's current answers (the decision variables) are used as the basis for such a conclusion. This means that in random selection processes, better solutions have a better chance of being chosen. The best solution (or solutions) in a set is often chosen using deterministic selection processes.

One of the most important factors in arriving at the best solution is the choosing of existing solutions to generate new solutions during algorithm iterations. Selection pressure is essential in meta-simulation and evolution methods.

The high-selection-pressure selection strategy is more likely to pick the best answers and ignore the bad ones at each search step. In contrast, a selection method with very little selection pressure will treat all solutions with matching values equally and select based on nothing more than whether or not they happen to be the best fit. difference.

In figure 2 shows a set of solutions to a made-up maximizing issue, with each solution's fit value and the probability of being chosen as an example under high or low selective pressure displayed in a descending order [1].

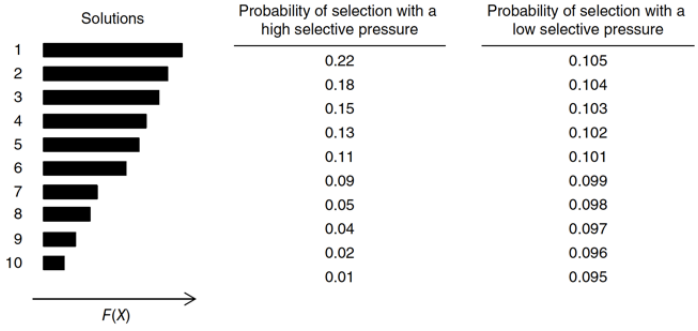

Fig. 2. Selection possibility for a set of hypothetical maximizing problem solutions 1–10.

In contrast, selective pressure boosts the odds that the best (largest) solutions will be chosen. Keep in mind that the homogeneous distribution gives each option an equal chance of being picked on each iteration. When no selective pressure is applied, the selection mechanism picks solutions at random from a given distribution [1].

In contradiction of definite selection and uniform distribution, other selection methods either allow the user to modify the selection pressure or automatically adjust the selection pressure based on the criteria being used. An important distinction between meta-heuristics and evolutionary algorithms is how they choose among potential solutions. There are algorithms that skip the selection step entirely, while others might be quite detailed in this regard. Many popular methods of choosing include the Boltzmann distribution, a roulette wheel, and a tournament.

### 3.10 Generating New Solutions

At each iteration, meta-empirical and evolutionary algorithms develop new solutions based on the current result. The algorithm completes each iteration by producing new solutions. Each algorithm generates novel answers that are distinct from those generated by other algorithms. To develop new solutions, however, all algorithms rely on preexisting ones. In reality, new answers are frequently similar to an earlier solution, a combination of two or more previous solutions, or randomly generated solutions whose adoption contributes to the search process. determined by comparing to past solutions. This article describes the techniques employed by the most prominent evolutionary and meta-empirical algorithms to develop new solutions iteratively.

### 3.11 The Best Solution in Each Algorithmic Iteration

For some algorithms, the best possible outcome is flagged after each cycle. It is the practice of some algorithms to carry on the best solution from one iteration unmodified to the next iteration, in search of a better one, at which time the best solution is replaced with the new one. The ideal answer from each iteration is saved and given more weight by other algorithms like HBMO when coming up with fresh solutions. Each algorithm has its own unique name for the best response of each iteration, such as "base point" in pattern search (PS) or "queen" in the HBMO algorithm.

### 3.12 Termination Criteria

The algorithm generates new solutions at the conclusion of each iteration. Each solution's fit function is calculated, and either the algorithm continues to the next iteration or stops. There were three main measures of success: total reps, the rate at which fitness function improved between sets, and total exercise time. With the initial criterion in place, the algorithm will run for a certain amount of time. For instance, the algorithm can be programmed to run for no more than 106 iterations.

The fundamental drawback of this criterion is that the analyst is in the dark as to how many iterations are necessary. As a result, the method may be terminated too soon if the present solution is suboptimal, or it may obtain a near-optimal solution too rapidly and then keep reproducing this solution without further improvement, resulting in an excessive computing burden.

When the difference between solutions from two or more successive iterations is less than a user-specified threshold, the method is terminated according to the second criterion. An issue with this strategy is that it may only produce a solution that is optimal in a very narrow context. Some empirical and evolutionary algorithms, if allowed to keep searching after hitting a threshold between multiple consecutive iterations, will use randomness or other methods to eliminate local solutions.

In contrast to other evaluation criteria, which take into account both the number of iterations and the rate at which the solution improves, the maximum runtime criterion simply terminates the algorithm after a predetermined amount of time has elapsed and reports the best solution obtained up to that point. In general, it is difficult to predict how long it will take to arrive to a near-optimal solution, which is the same problem that applies to this criterion and to finding the maximum number of iterations.

## 4. CONCLUSION

This work provides an introduction to meta-simulation and evolution algorithms by elaborating on the many techniques utilized to explore the choice space. In describing the characteristics of a variety of technical optimization problems, emphasis was placed on the mechanisms of meta-heuristic and evolutionary

algorithms. This paper also introduces coding meta-empirical and evolutionary algorithms, dealing with limitations, and selecting solutions, among other subjects. A general algorithm comprising the most prevalent characteristics of meta-simulation and evolution algorithms has been described. This broad method will serve as a good comparison standard for other algorithms. This study concludes by reviewing the performance evaluation techniques for meta-simulation and evolution algorithms.

**REFERENCES**

Saka, M. P., Hasançebi, O. Ğ. U. Z. H. A. N., & Geem, Z. W. (2016). Metaheuristics in structural optimization and discussions on harmony search algorithm. *Swarm and Evolutionary Computation*, *28*, 88-97.

Sörensen, K. (2015). Metaheuristics—the metaphor exposed. *International Transactions in Operational Research*, *22*(1), 3-18.

Dokeroglu, T., Sevinc, E., Kucukyilmaz, T., & Cosar, A. (2019). A survey on new generation metaheuristic algorithms. *Computers & Industrial Engineering*, *137*, 106040.

Kondamadugula, S., & Naidu, S. R. (2016, October). Accelerated evolutionary algorithms with parameterimportance based population initialization for variation-aware analog yield optimization. In *2016 IEEE 59th International Midwest Symposium on Circuits and Systems (MWSCAS)* (pp. 1-4). IEEE

Battiti, R., Brunato, M., & Mascia, F. (2008). *Reactive search and intelligent optimization* (Vol. 45). Springer Science & Business Media.

Hinterding, R., Michalewicz, Z., & Eiben, A. E. (1997, April). Adaptation in evolutionary computation: A survey. In *Proceedings of 1997 Ieee International Conference on Evolutionary Computation (Icec'97)* (pp. 65-69). IEEE.

Eiben, A. E., & Smith, J. E. (2003). Gray coding. In *Introduction to Evolutionary Computing* (pp. 265-265). Springer, Berlin, Heidelberg.

Gent, I. P., & Walsh, T. (1993, July). Towards an understanding of hill-climbing procedures for SAT. In *AAAI* (Vol. 93, No. Citeseer, pp. 28-33)

Reeves, C. R. (Ed.). (1993). *Modern heuristic techniques for combinatorial problems*. John Wiley & Sons, Inc..

Larrañaga, P., & Lozano, J. A. (Eds.). (2001). *Estimation of distribution algorithms: A new tool for evolutionary computation* (Vol. 2). Springer Science & Business Media.

Marchiori, E., & Rossi, C. (1999). A flipping genetic algorithm for hard 3-SAT problems.