



REPUBLIC OF TÜRKİYE

ALTINBAŞ UNIVERSITY

Institute of Graduate Studies

Electrical and Computer Engineering Department

**HAWKFISH OPTIMIZATION ALGORITHM (HFOA):
A NOVEL GENDER-SPECIFIC, DUAL FITNESS
FUNCTION APPROACH FOR SOLVING COMPLEX
OPTIMIZATION PROBLEMS**

Ali Jawad Kadhim AL-KHARSAN

Doctor of Philosophy

Supervisor

Assoc. Prof. Dr. Oğuz ATA

İstanbul, 2025

**HAWKFISH OPTIMIZATION ALGORITHM (HFOA): A NOVEL
GENDER-SPECIFIC, DUAL FITNESS FUNCTION APPROACH FOR
SOLVING COMPLEX OPTIMIZATION PROBLEMS**

Ali Jawad Kadhim AL-KHARSAN

Electrical and Computer Engineering Department

Doctor of Philosophy

ALTINBAŞ UNIVERSITY

2025

The dissertation titled HAWKFISH OPTIMIZATION ALGORITHM (HFOA): A NOVEL GENDER-SPECIFIC, DUAL FITNESS FUNCTION APPROACH FOR SOLVING COMPLEX OPTIMIZATION PROBLEMS prepared by ALI JAWAD KADHIM AL-KHARSAN and submitted on 24/06/2025 has been **accepted unanimously** for the degree of Ph.D. in Electrical and Computer Engineering Department.

Assoc. Prof. Dr. Oğuz ATA
Supervisor

Thesis Defense Committee Members:

Assoc. Prof. Dr. Oğuz ATA Department of Software
Engineering
Altınbaş University _____

Assoc. Prof. Dr. Sefer KURNAZ Department of Computer
Engineering
Altınbaş University _____

Assoc. Prof. Dr. Aytuğ BOYACI Department of Computer
Engineering
National Defence University _____

Assoc. Prof. Dr. Can
EYÜPOĞLU Department of Computer
Engineering
National Defence University _____

Assoc. Prof. Dr. Osman ALTAY Department of Software
Engineering
Celal Bayar University _____

I hereby declare that this dissertation meets all format and submission requirements of a doctoral dissertation.

I hereby declare that all information and data presented in this graduation project has been obtained in full accordance with academic rules and ethical conduct. I also declare all unoriginal materials and conclusions have been cited in the text and all references mentioned in the Reference List have been cited in the text, and vice versa as required by the abovementioned rules and conduct.

Ali Jawad Kadhim AL-KHARSAN

Signature

DEDICATION

To my beloved family and friends, Your unwavering support, encouragement, and love have been my guiding light throughout this academic journey. From late-night study sessions to celebrating small victories, you've been there every step of the way. This thesis is as much yours as it is mine. And to my esteemed supervisor, Prof. Oğuz ATA, Your mentorship, expertise, and patience have shaped my research and enriched my understanding. Your dedication to excellence inspires me daily. Thank you for believing in my potential and guiding me toward success. With heartfelt gratitude.



ABSTRACT

HAWKFISH OPTIMIZATION ALGORITHM (HFOA): A NOVEL GENDER-SPECIFIC, DUAL FITNESS FUNCTION APPROACH FOR SOLVING COMPLEX OPTIMIZATION PROBLEMS

AL-KHARSAN, Ali Jawad Kadhim

Ph.D., Electrical and Computer Engineering Department, Altınbaş University,

Supervisor: Assoc. Prof. Dr. Oğuz ATA

Date: 06/2025

Pages: 133

Inspired by the gender shift behaviour shown in Hawkfish, this thesis contributes a new concept in optimization algorithm modelling by using this natural phenomenon as a basis for solving optimization problems through mathematical modelling and imitation. To evaluate the performance of HFOA, it is initially subjected to benchmarking against various test problems, including the CEC-06 (2019) benchmark suite, to quantitatively and qualitatively validate its efficacy. Subsequently, HFOA is deployed to identify the optimal configurations for the welded beam design problem and the tension/compression spring (TCS) design problem, showcasing its practical utility. The outcomes reveal that the proposed HFOA algorithm consistently outperforms well-established and recent algorithms documented in the literature. Furthermore, the results obtained in real-world applications underscore the effectiveness of HFOA in tackling problems characterized by unknown search spaces.

Keywords: Metaheuristics, HFOA, CEC-06 (2019), Optimization, Welded Beam Design, TCS Design Problem.

TABLE OF CONTENTS

	<u>Pages</u>
ABSTRACT	vi
LIST OF TABLES.....	x
LIST OF FIGURES.....	xi
ABBREVIATIONS.....	xiii
1. INTRODUCTION.....	1
1.1 BACKGROUND	1
1.2 PROBLEM STATEMENT	5
1.3 THESIS QUESTIONS.....	9
1.4 AIMS AND OBJECTIVES	10
1.4.1 Thesis Objectives	10
1.4.2 Thesis Aims	10
1.5 THESIS CONTRIBUTION	11
2. LITERATURE REVIEW.....	14
2.1 PREVIOUS WORK ON HEURISTIC SEARCH ALGORITHM	16
2.2 PREVIOUS WORK ON METAHEURISTIC SEARCH ALGORITHMS.....	17
2.2.1 Simulated Annealing SA.....	17
2.2.2 Particle Swarm Optimization PSO.....	18
2.2.3 Ant Colony Optimization Aco	20
2.2.4 Bat Algorithm BA.....	22
2.2.5 Firefly Algorithm FA.....	24
2.2.6 Genetic Algorithm GA.....	25
2.3 PREVIOUS WORK ON HYPER HEURISTIC SEARCH ALGORITHMS.....	27
2.4 MULTI OBJECTIVE OPTIMIZATION	31
3. MATERIALS AND METHODS	41
3.1 INTRODUCTION	41
3.2 DISRUPTIVE META-HEURISTICS	42
3.2.1 Genetic Algorithms	43
3.2.2 Local Search.....	44
3.3 CONSTRUCTIVE META-HEURISTICS	46

3.3.1 Random Greedy Algorithms	46
3.3.2 Algorithms by Estimating Distributions	46
3.3.3 Ant Colony Optimization.....	47
3.4 HYBRID META-HEURISTICS	48
3.4.1 Memetic Algorithms	48
3.4.2 Hybridization Between Disruptive Approaches and Constructive Approaches ..	49
3.5 INTENSIFICATION VERSUS DIVERSIFICATION	49
3.6 ARTIFICIAL INTELLIGENCE APPLICATIONS	51
3.6.1 Satisfiability of Boolean Formulas	51
3.6.2 Constraint Satisfaction Problems	59
3.7 ARTIFICIAL SWARM ALGORITHM	66
3.7.1 The Basic Principle of Artificial Fish Algorithm	66
3.7.2 Individual Movement Operator.....	69
3.8 HAWKFISH OPTIMIZATION ALGORITHM.....	70
3.8.1 Gender Benders.....	70
3.8.2 Modelling Genders In Hawkfish Algorithm	71
4. PROPOSED METHODOLOGY	73
4.1 SYSTEM OUTLINE	73
4.1.1 HAWKFISH Gender Bending	74
4.1.2 Mathematical Modelling.....	77
5. SIMULATION, RESULTS AND DISCUSSION	85
5.1 INTRODUCTION	85
5.1.1 Implementation Environment	85
5.1.2 Time of Convergence and Error Rate	86
5.2 BENCHMARK FUNCTIONS	91
5.3 SOLVING THE WELDED BEAM DESIGN PROBLEM	99
5.4 SOLVING THE TENSION/COMPRESSION SPRING (TCS) DESIGN PROBLEM.....	105
5.4.1 Key Parameters and Variables	106
5.4.2 Objective Function.....	107
5.4.3 Constraints	107
5.4.4 Formulating the Optimization Problem	107

5.4.5 Solving the TCS Design Problem Using HFOA.....	108
6. CONCLUSION AND FUTURE WORKS	111
6.1 CONCLUSIONS	111
6.2 FUTURE WORK.....	112
REFERENCES	114



LIST OF TABLES

	<u>Pages</u>
Table 2.1: Summary of Contributions in Hybrid SA Algorithms.	18
Table 2.2: Summary of Contributions in Particle Swarm Optimization (PSO) for VRP. ...	20
Table 2.3: Summary of Contributions in Ant Colony Optimization (ACO).	22
Table 2.4: Summary of Contributions in Bat Algorithm (BA).	24
Table 2.5: Summary of Contributions in Firefly Algorithm (FA) for TSP.	25
Table 2.6: Summary of Contributions in Genetic Algorithms (GA) for VRP.	26
Table 2.7: Summary of Hyperheuristics and Optimization Techniques.	30
Table 5.1: The Fish Swarm Algorithm's Error Rate as a Function of Both Time and Iteration.	88
Table 5.2: The Suggested HFOA's Error Rate as a Function of Both Time and Iteration. .	89
Table 5.3: Assessment of the Suggested HFOA's Performance in Comparison to Cutting- Edge Optimisation Methods Employing CEC-06 (2019) Functions.	94
Table 5.4: Evaluation of Alternative Optimisation Algorithms by Comparing Their Cost Functions to that of the Suggested HFOA.	104
Table 5.5: Examining How the HFOA Handles TCS Design Problems in Comparison to Cutting-Edge Algorithms.	109

LIST OF FIGURES

	<u>Pages</u>
Figure 1.1: Family Tree of Search Based Optimization Algorithm.	3
Figure 1.2: Inspiration for Most of the Optimization Algorithms.	5
Figure 1.3: Problem of Local Minimum in Optimization Methods.	7
Figure 1.4: Search Strategy for the Global Minimum in Metaheuristic Algorithms.....	8
Figure 2.1: Family Tree of Optimization Algorithms [16].....	14
Figure 3.1: Basic Model of AFSA.....	66
Figure 3.2: Search Space of the Fish Swarm Algorithm.	67
Figure 3.3: Diagram of the Fish Swarm Algorithm.	68
Figure 3.4: Proposed Hawkfish Algorithm.	72
Figure 4.1: Outline of the Proposed HFOA Swarming Action.....	74
Figure 4.2: Gender Inversion in the Proposed HFOA.	77
Figure 4.3: Search Space for Males and Females in the HFOA.....	79
Figure 4.4: Visual Search Scope for the HFOA Population.....	82
Figure 5.1: The HFOA's Simulation of the Gender-Specific Double Search Spaces.....	87
Figure 5.2: The Fish Swarm Algorithm's Error Rate as a Function of Both Time and Iteration.	89
Figure 5.3: HFOA (Double Objective Function) vs HFOA (Single Objective Function). .	90
Figure 5.4: Execution Time (a) and Error (b) for Each Iteration Step in the Proposed HFOA.	91
Figure 5.5: Evaluating Performance in a Unimodal Dimension Using CEC-06 (2019).	97
Figure 5.6: Evaluating Performance in a High Dimension Using CEC-06 (2019).	97
Figure 5.7: Evaluating Performance in a Fixed Dimension Using CEC-06 (2019).	98

Figure 5.8: The Proposed HFOA's Convergence Graph, with Grey Highlights, in Comparison to Other State-of-the-Art Algorithms.....	98
Figure 5.9: Modelling of the Welded Beam Design Problem.	100
Figure 5.10: Evaluation of the HFOA Cost Function in Relation to Benchmark Optimisation Approaches for Welded Beam Design.	105
Figure 5.11: Constraints of the Tension Compression Spring TCS problem.....	106
Figure 5.12: An Analysis of the Tension-Compression Spring (TCS) Design Problem and Its Optimisation Methods.	110



ABBREVIATIONS

ACO	:	Ant Colony Algorithm
AFSA	:	Artificial Fish Swarm Algorithm
BA	:	Bat Algorithm
DL-PSO	:	Deep Learning-based Particle Swarm Optimization
FA	:	Firefly Algorithm
HFOA	:	HawkFish Optimization Algorithm
HAFA	:	Ant Colony System with Firefly Algorithms
HH_EG	:	Adaptive Epsilon-Greedy Selection
HVRP	:	Hybrid Vehicle Routing Problem
LLHs	:	Low-Level Heuristics
LS	:	Local Search
MOGAs	:	Multi-Objective Genetic Algorithms
MOPSO	:	Multi-Objective Particle Swarm Optimization
NSGA-II	:	Non-dominated Sorting Genetic Algorithm II
QDSA	:	Quantum Dolphin Swarm Algorithm
rTMS	:	Repeated Transcranial Magnetic Stimulation
SA	:	Simulated Annealing
SMC	:	Sinusoidal Motion Crossover
SI	:	Swarm Intelligence

SPEA2	:	Strength Pareto Evolutionary Algorithm 2
TSP	:	Ant Colony Algorithm
WOA	:	Artificial Fish Swarm Algorithm
ZMSOA	:	Zebra Mantis Shrimp Optimization Algorithm



LIST OF SYMBOLS

D	:	Mean Coil Diameter
F	:	Food Metric
G	:	Modulus of Rigidity
N	:	Number of Active Coils
N_f	:	Neighboring fish Within the Visual Scope
O_d	:	End Deflection
P	:	Population of the fish
P_c	:	Buckling Load
R	:	Random Number Between 0 and 1
V	:	Visual Scope
X	:	Center of the Corresponding Subpopulation
b	:	Learning Coefficient for Direction Vector
d	:	Wire Diameter
$d(i,j)$	:	(j)-th Element of the Direction Vector of the (i)-th fish
$f_1(x)$	:	First Fitness Function
$f_2(x)$	:	Second Fitness Function
k	:	Number of Subpopulations
$s(i,j)$	:	Step Size of the (i)-th fish
w	:	Learning Coefficient

$x_f(t)$	:	Position of fish (f) at Iteration (t)
$x(i,j)$	:	(j)-th Element of the Position Vector of the (i)-th fish
$x_{\{nbest\}}$	:	Position of the best Neighboring fish in $N_f N_{fN}$
$x_{(gj)}$	:	(j)-th Element of the Position Vector of the Global Best Fish
$x_{(lj)}$	:	(j)-th Element of the Position Vector of the Local Best Fish
$\bar{O} x_j$	:	Sum of Positions of all Fish in the Respective Subpopulation
ϕ_d	:	Deflection
η_s	:	Spring Index
τ_s	:	Shear Stress
τ_b	:	Bending Stress
ϕ_p	:	Population size of Artificial fish
α	:	Learning Coefficient for Step Size

1. INTRODUCTION

1.1 BACKGROUND

During the past couple of years, optimisation has emerged as a significant subject of investigation in a wide range of fields, most notably in the fields of computer science and engineering [1]. When it comes to the majority of instances, engineers and programmers are the ones who are tasked with the job of identifying acceptable requirements for a number of issues. This is also referred to as an ideal solution. The value of a particular function or the behaviour of a particular system can be reduced to the maximum extent feasible by a solution, and this is the condition under which the solution is deemed to be optimal. According to the body of research that has been conducted, one of the most important differences between an engineering project and a technology project is associated with the optimisation process [2]. This is because optimisation is the instrument that differentiates the technical approach to a problem. This is the reason why this is the case. It is possible to differentiate the technical viewpoint on an issue by employing optimisation as a method of solving the problem. Computer science and optimisation continue to have a tight association with one another and continue to progress hand in hand with one another. Both of these fields continue to be closely related to one another. In the realm of computer science, there have been a significant number of significant advancements made, and the notion of optimisation has been the driving force behind the success of these advancements. Applications of optimisation that are particularly significant include, for example, problems involving planning and choice. In each and every one of these difficulties, optimisation is being applied. The discipline of computer science, on the other hand, plays a substantial role in the field of technical optimisation research. It does this by making a contribution to the creation of efficient algorithms and by providing the proper tools to enable model maintenance and result analysis [6]. There are a great number of different optimisation strategies that have been documented in published journals. Some examples of these strategies are simplex [7], gradient descent [8], and genetic algorithm [9]. These are only a handful of the numerous tactics that have been reported; there are many more. Modifications to these methods can be made in order to address a large variety of various types of optimisation challenges. There are problems that fall into these categories that are either linear or non-linear, continuous or discontinuous, convex or non-convex, single or multi-

objective, and linear or non-linear at the same time. Deterministic optimisation and stochastic optimisation are the two fundamental categories that are often utilised for the aim of classifying these methodological approaches. When searching for the best solution, stochastic optimization employs randomization in its search techniques [10]. Deterministic optimization, on the other hand, is a more conventional approach to optimization algorithms that are founded on computations and algebraic operators, such as gradients and Hessians. Out of the two methods, deterministic optimization is the one that is utilized most frequently. In instance, a non-convex optimization problem is one of the most challenging and challenging issues since it contains a large number of local optimum solutions. Because of this, it is one of the most difficult problems to solve. Because of this fact, there are a lot of methods that have been tried and tested. issues that are either linear or non-linear, continuous or discontinuous, convex or non-convex, single or multi-objective, and linear or non-linear at the same time are examples of issues that may be classified into these categories.

A more conventional approach to optimisation algorithms, deterministic optimisation is based on computations and algebraic operators, such as gradients and Hessians. On the other hand, deterministic optimisation is a more traditional approach to optimisation methods. The deterministic optimisation approach is the one that is utilised the majority of the time out of the two distinct strategies. As an illustration, a non-convex optimisation problem is one of the most difficult and difficult problems to solve since it involves a significant number of local optimal solutions. Because of this, it is absolutely one of the most challenging challenges to find a solution to. Consequently, there are a great number of approaches that have been put to the test and proven to be effective.

like the Newton-Raphson [1] and Gradient Descent [8] algorithms, are ineffective when used to the resolution of this specific category of issues. Numerous heuristics, such as ant colony [11] and genetic algorithm [9], have been developed as an alternative to these problems. These heuristics offer quicker solutions to these problems, but at the expense of solution certainty. In addition, these heuristics frequently switch between solutions that are local minima.

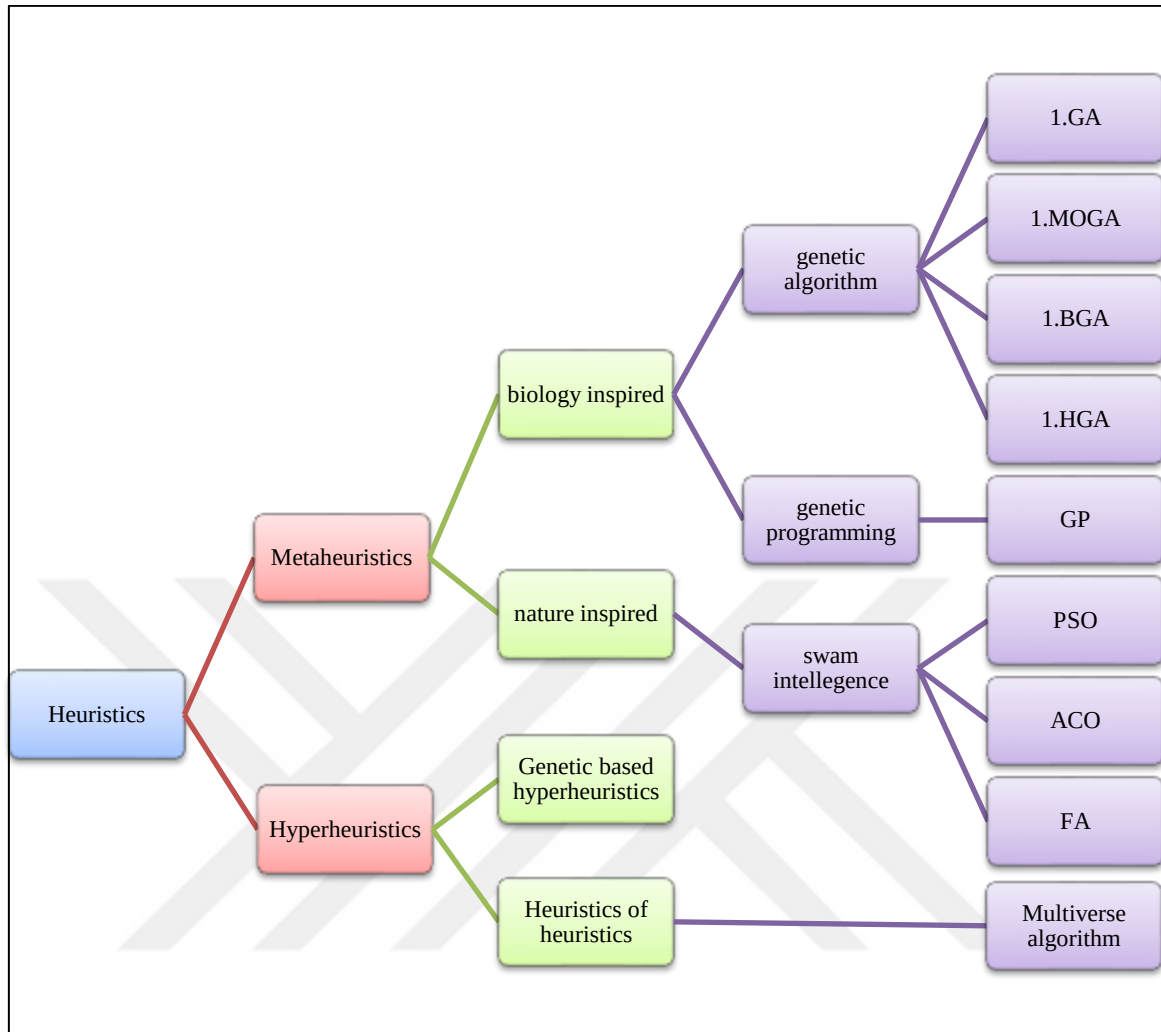


Figure 1.1: Family Tree of Search Based Optimization Algorithm.

During the early 1990s, the field of computer science had a significant growth, which resulted in the addition of the study of soft computing to the domain of computer science. For the most part, earlier computer technologies were only able to accurately represent and analyse systems that were thought to be relatively simple. More complex systems, such as those that are found in the fields of biology, medicine, the humanities, management sciences, and other fields associated to these, sometimes remained opaque to the conventional mathematical and analytical methodologies. This occurred in a great number of instances.

It is vital to bear in mind that the degrees of simplicity and complexity of systems are relative, and that many traditional mathematical models have shown to be enormously fruitful despite the fact that they are difficult to solve. This is something that should be kept in mind at all times.

The goals of soft computing are to provide tractability, robustness, and a low cost of solution while simultaneously addressing imprecision, uncertainty, and approximation. Soft computing solutions are also intended to be inexpensive. In addition, soft computing is working towards the provision of these. conventional approaches to computing are mostly based on formal logical systems, such as sentential and predicate logics, or they primarily rely on computer-aided numerical analysis, such as finite element analysis (FEM).

Both of these approaches are generally considered to be the foundation of conventional computing. When compared to traditional computer methods, soft computing approaches are more comparable to biological processes. This is in contrast to the conventional computer methods. Conventional approaches are heavily reliant on formal logical systems, which is one of the key reasons why this is the case.

Each and every one of the soft computing approaches is designed to work in conjunction with the capabilities of the other techniques. In contrast to hard computing strategies, which aim for precision and complete accuracy, soft computing approaches take advantage of a problem's inherent tolerance for imprecision, partial truth, and uncertainty. All soft computing methodologies are intended to operate synergistically with the functionalities of other techniques. Unlike hard computing techniques that prioritise precision and absolute accuracy, soft computing methodologies leverage a problem's intrinsic tolerance for imprecision, partial truth, and uncertainty. These algorithms were chosen in particular because of their widespread application in the following fields:

- a. EA → Urban Design and planning, Routing, Design optimization, mathematical modelling.
- b. SI → Search algorithms, Trajectory planning, Image segmentation, Disease detection, Security, Feature selection

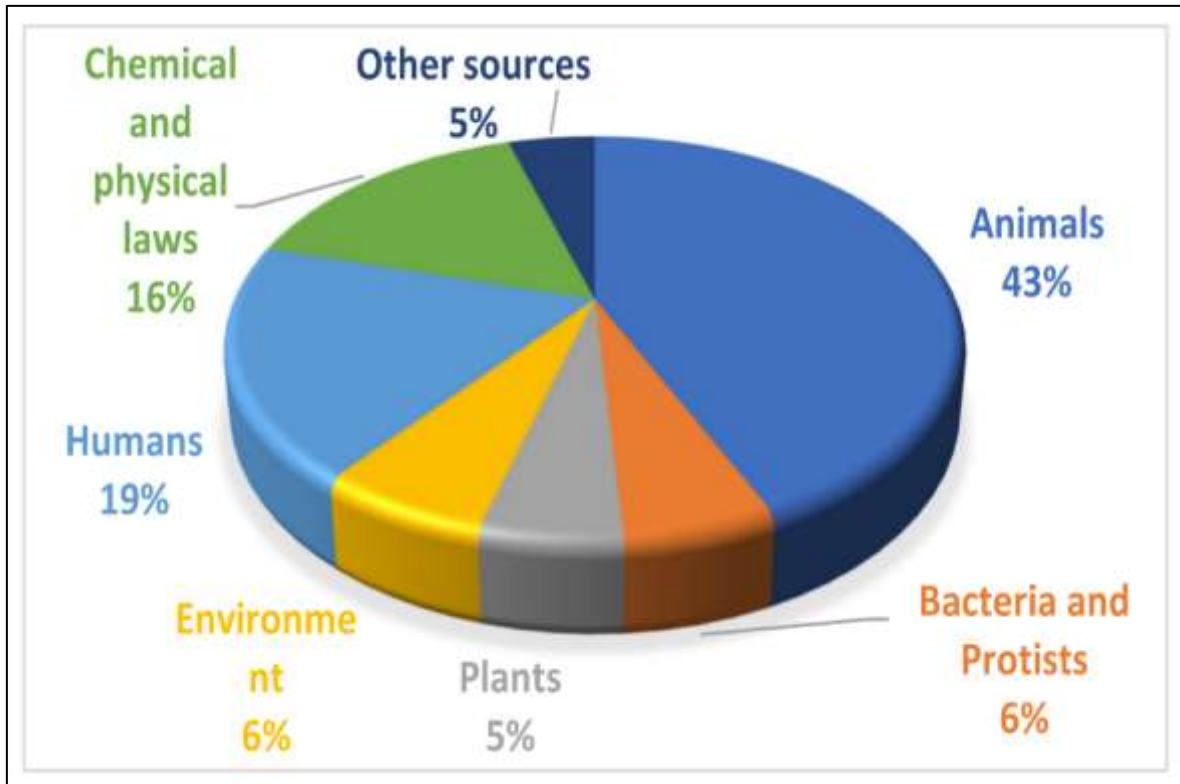


Figure 1.2: Inspiration for Most of the Optimization Algorithms.

1.2 PROBLEM STATEMENT

The goal of optimisation is to determine a set of inputs to an objective function that will either result in the greatest possible function evaluation or the lowest possible function evaluation. This is the purpose of optimisation. The topic that serves as the basis for a wide variety of machine learning methods, such as the fitting of logistic regression models and the training of artificial neural networks, is being discussed here.

It is probable that there are hundreds of optimisation algorithms that are commonly used, and that major scientific code libraries offer tens or even hundreds of various algorithms for users to choose from.

As a consequence of this, it may be challenging to ascertain which algorithms should be taken into consideration for a certain optimisation task. Optimisation refers to the process of identifying the input parameters or arguments to a function that will create the output value that is either the lowest possible value or the greatest possible value. This process is carried out in order to provide the best possible results. In the field of machine learning, the type of optimisation problem that occurs most commonly is known as continuous function

optimisation optimisation. Real-valued numerical numbers, such as floating-point values, are the kind of parameters that are supplied into the function while dealing with this particular type of optimisation issue. The evaluation of the values that are entered into the function proceeds all the way through to the output of the function, which is also of a real-valued type.

As opposed to functions that accept discrete variables, which are known as combinatorial optimisation issues, we may refer to problems of this kind as continuous function optimisation. Combinatorial optimisation issues are a type of optimisation problem. By doing so, we would be better able to discriminate between the two categories of issues. There are a vast range of alternative optimisation processes that may be utilised to solve problems that involve the optimisation of continuous functions. Additionally, there are probably just as many various ways to classify and summarise these algorithms.

An additional strategy involves categorising optimisation algorithms according to the kind of target function that is being subjected to optimisation. Generally speaking, the quantity of information that is known about the function and whether or not that information can be utilised effectively in the search are two factors that determine how easy it is to optimise the function that is being targeted through optimisation. The question of whether or not the goal function can be distinguished at a particular place is maybe the most important distinction that can be made between the various optimisation strategies.

The question that has to be answered is whether or not it is possible to ascertain the first derivative of the function, which is often referred to as the gradient or slope, given a certain candidate solution that is being investigated. Because of this, algorithms may be divided into two categories: those that are able to make use of the knowledge on the computed gradient, and those that are unable to make use of it.

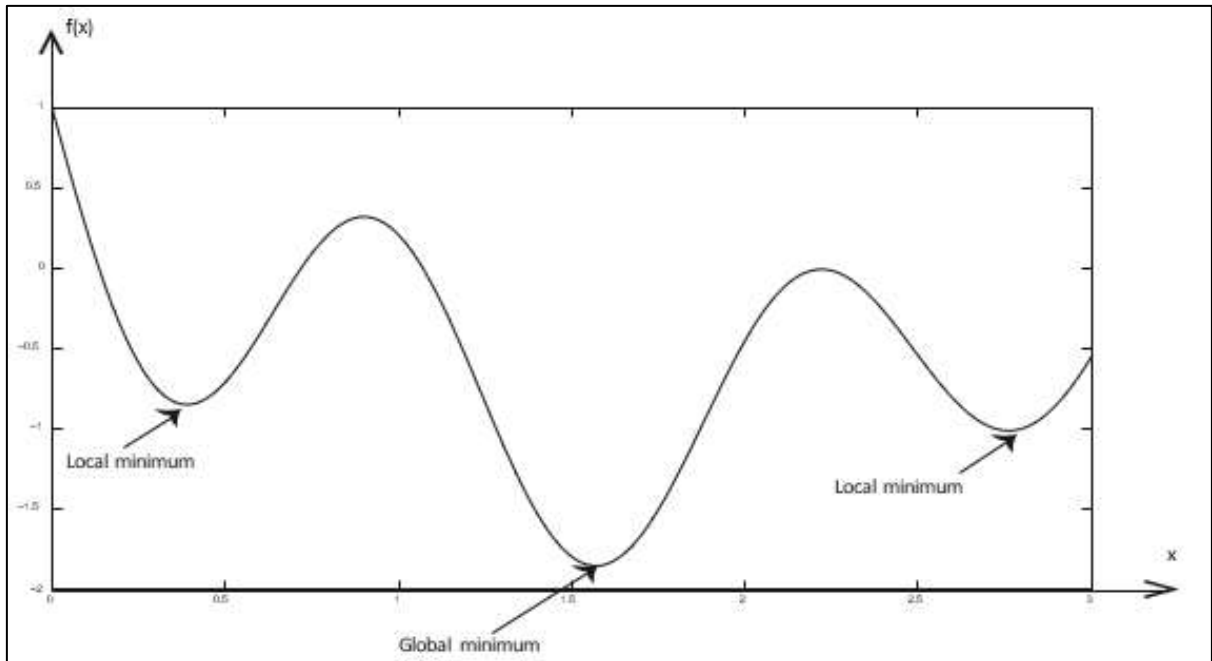


Figure 1.3: Problem of Local Minimum in Optimization Methods.

A metaheuristic algorithm is a search technique that is intended to identify a viable solution to an optimization problem that is intricate and challenging to resolve optimally. In the real world, where resources are limited, it is crucial to identify a solution that is nearly optimal, despite the possibility that the information available is inaccurate or incomplete (e.g., computational power and time). The development of metaheuristics, which are strategies for addressing optimization issues such as these, has been one of the most significant advancements in operations research over the past two decades. An optimisation issue that is complex and difficult to address optimally can be approached with the use of a metaheuristic algorithm, which is a search strategy. It is critical to find a nearly optimum solution in the real world when resources are limited, even if the information that is provided may be erroneous or insufficient (for example, in terms of processing power and time). Among the most noteworthy developments in operations research within the last 20 years has been the creation of metaheuristics, which are methods for dealing with optimisation problems like these.

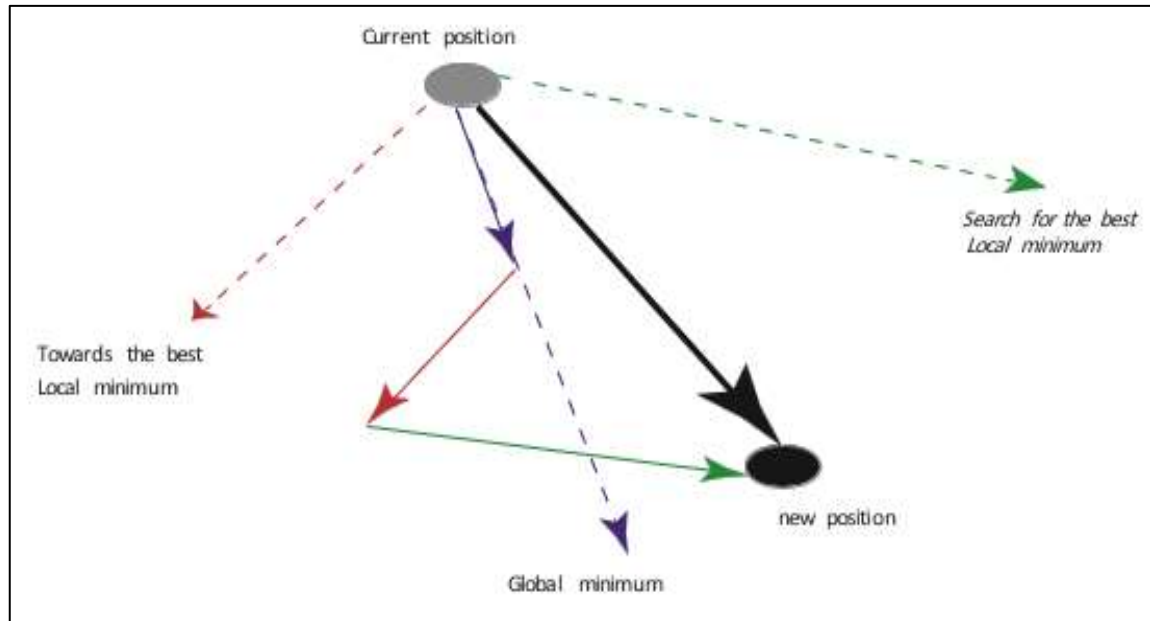


Figure 1.4: Search Strategy for the Global Minimum in Metaheuristic Algorithms.

There are problems that require attention to be paid to in order to generate better solutions than the standard ways that are now in use. We describe a variety of metaheuristic algorithms that are very extensive to diverse applications for the purpose of solving non-linear non-convex optimization issues. Within the realm of combinatorial optimization, it is not possible to find solutions to issues that fall within the NP-hard category (i.e., in reasonable run time). Therefore, as opposed to optimization algorithms, iterative approaches, and straightforward greedy heuristics, metaheuristics can frequently find good answers with less work required from the computing side. There are many distinct types of issues that cannot reasonably be resolved by applying an optimization technique to reach a global optimality state. For instance, the presence of stochastic random variables within an optimization problem's objective or constraints might make the problem far more difficult to solve. As a result, the resolution of large-scale stochastic problems by means of stochastic programming or robust optimization strategies is not an easy task. In a variety of contexts, metaheuristics can be an extremely useful tool. In their purest form, many optimization problems take the form of multi-objective functions that are constrained in a non-linear way. For example, most engineering optimization problems are substantially non-linear and call for solutions to multi-objective issues. On the other hand, problems involving artificial intelligence and machine learning largely rely on extensive datasets, and it is difficult to structure the optimization problem in such a way that it can be solved to achieve optimality. As a result,

metaheuristics play an important part in the solution of actual issues that are challenging to solve using traditional optimization methods. It is possible to categorize metaheuristic algorithms according to the way in which they function within the search space [3]. There are many ways in which these classes might be categorised. For instance, there are memory-less techniques, ones that need memory, ones that don't, and so on. Other examples include population-based vs single point search, dynamic versus static objective functions, and one versus numerous neighbourhood structures. Both exploring and exploiting the solution space can benefit from metaheuristics.

1.3 THESIS QUESTIONS

When attempting to solve NP-Hard problems, it is considered standard practice to employ metaheuristic algorithms, which are also commonly referred to as MAs. When it comes to optimization, there is a group of issues that are known as NP-Hard problems. These problems are extremely difficult to solve. Without the use of polynomials, it is impossible to find a solution to these issues. In order to discover the best or near-best solution to such issues as quickly as is practically possible, MAs make use of a number of optimization approaches that originate from a variety of diverse backgrounds. This is done in order to maximize both efficiency and effectiveness. Ideas for such algorithms could come from a wide variety of different areas. Math, politics, physics, and the scientific sciences are all examples of such domains. Having said that, it is not feasible to manage every problem in the most efficient manner by use the same metaheuristic method. This is because there are many different kinds of problems. When it comes to the resolution of specific classifications of issues, it is likely that certain MAs will perform better than algorithms that are conceptually comparable. Our goal here is to help algorithms make the jump from a metaheuristic to a hyperheuristic state by introducing two fitness functions. One will seek out the optimal solution on a global scale, and the other will seek out the inverse of that solution to prevent misconvergence. We expect the new algorithm to outperform the old one in terms of speed and error rate.

1.4 AIMS AND OBJECTIVES

1.4.1 Thesis Objectives

- a. The objectives of this thesis are to develop and introduce the HAWKFISH-Inspired Optimization Algorithm (HFOA) by formulating a new optimization approach based on the gender transition behavior observed in hawkfish, utilizing mathematical modeling and imitation of this natural phenomenon to solve optimization problems.
- b. The performance of HFOA will be benchmarked using various test problems, including the CEC-06 (2019) benchmark suite, to quantitatively and qualitatively evaluate its efficacy compared to existing algorithms. HFOA will be applied to identify optimal configurations for the welded beam design problem and the tension/compression spring (TCS) design problem, demonstrating its practical utility.
- c. The outcomes of HFOA will be compared with well-established and recent algorithms documented in the literature, highlighting its superior performance in various optimization scenarios.
- d. Additionally, the effectiveness of HFOA in tackling problems characterized by unknown search spaces will be showcased through real-world applications, providing empirical evidence of the algorithm's capability to address complex optimization problems in practical settings

1.4.2 Thesis Aims

By evaluating the metaheuristic models presented in Figure 1, the aim of this research is to develop a new metaheuristic optimization algorithm that embodies the following characteristics:

- a. High flexibility in terms of usage and application.
- b. Effective as global optimizers.
- c. Robustness to problem size and random variables.
- d. Viability as a practical alternative when other methods fail.
- e. Independence from the need to calculate the derivative of functions.

- f. Applicability to both continuous and discrete problems.
- g. Superior speed and strength compared to traditional methods.

This aim will be pursued through a systematic trial and error approach. However, it is acknowledged that this method, while useful, may lack precision, particularly in the context of hyperheuristics. The limitations of this method are primarily due to its reliance on the adequacy of the initially generated trial solutions.

1.5 THESIS CONTRIBUTION

The primary contributions of this thesis lie in the development and validation of the HAWKFISH-Inspired Optimization Algorithm (HFOA), which introduces a novel approach to metaheuristic optimization. Drawing inspiration from the gender transition behavior observed in hawfish, this algorithm employs mathematical modeling and imitation of natural phenomena to address complex optimization problems. The research rigorously benchmarks HFOA against established test problems, including the CEC-06 (2019) benchmark suite, to provide both quantitative and qualitative assessments of its performance. Practical applications of HFOA to the welded beam design problem and the tension/compression spring (TCS) design problem are thoroughly explored, demonstrating the algorithm's efficacy in real-world scenarios. Comparative analyses with existing algorithms highlight HFOA's superior capabilities, underscoring its robustness, flexibility, and effectiveness in handling optimization problems characterized by unknown search spaces. Furthermore, the thesis discusses the trial and error approach utilized in the development of HFOA, addressing the inherent limitations and providing insights into the algorithm's practical implementation. A new technique to metaheuristic optimisation, the HAWKFISH-Inspired Optimisation Algorithm (HFOA), is developed and validated as the main contribution of this thesis. This method uses mathematical modelling and imitation of natural processes to solve complicated optimisation issues, taking inspiration from the gender transition behaviour seen in hawfish. The research extensively compares HFOA to well-known test issues, such as the CEC-06 (2019) benchmark suite, in order to evaluate its performance quantitatively and qualitatively. The algorithm's effectiveness in real-world settings is demonstrated by conducting a comprehensive exploration of its practical applications to the welded beam design problem and the tension/compression spring (TCS)

design problem. Analyses of HFOA's performance in comparison to other algorithms show how well it handles optimisation issues with uncertain search spaces, demonstrating its resilience, adaptability, and efficacy. The thesis goes on to detail the iterative process that went into creating HFOA, pointing out its shortcomings and offering advice on how to put it into practice.

i Development of HFOA:

- a. Introduced a new metaheuristic optimization algorithm inspired by the gender transition behavior in hawfish.
- b. Utilized mathematical modeling and imitation of natural phenomena for optimization problem-solving.

ii Benchmarking and Validation:

- a. Conducted extensive benchmarking of HFOA against the CEC-06 (2019) benchmark suite.
- b. Provided quantitative and qualitative validation of HFOA's performance compared to existing algorithms.

iii Practical Application Demonstration:

- a. Applied HFOA to the welded beam design problem, showcasing its practical utility.
- b. Deployed HFOA for solving the tension/compression spring (TCS) design problem, further demonstrating its effectiveness.

iv Comparative Analysis:

- a. Highlighted the superior performance of HFOA in comparison with well-established and recent algorithms in the literature.
- b. Demonstrated HFOA's robustness, flexibility, and capability to handle various optimization scenarios.

v Addressing Real-World Problems:

- a. Showcased HFOA's effectiveness in tackling problems characterized by unknown search spaces.
- b. Provided empirical evidence of the algorithm's capability to address complex optimization problems in practical settings.

vi Methodological Insights:

- a. Discussed the trial and error approach utilized in developing HFOA.

- b. Addressed the limitations of the trial and error method, particularly in the context of hyperheuristics.
- c. Provided insights into the practical implementation and adequacy of preliminary trial solutions.



2. LITERATURE REVIEW

A strategy to solve a problem which does not ensure the best answer is called heuristics. We can, however, come up with high-quality, realistic solutions that satisfy the goals of the situation. Metaheuristics and classical heuristics are the two primary families of heuristics, both of which were created between 1960 and 1990. In meat and hyper heuristic search, scholars have come up with a variety of methodologies and approaches. Chapter of the thesis demonstrates earlier research that is relevant to this project. Several of these studies have been summarized in this section Among all the classifications present in the literature, let us consider the one proposed by Dixon and Szegö [15]. According to this classification, the methods developed to solve problem can be broadly divided into two classes:

- a. deterministic algorithms.
- b. stochastic algorithms.

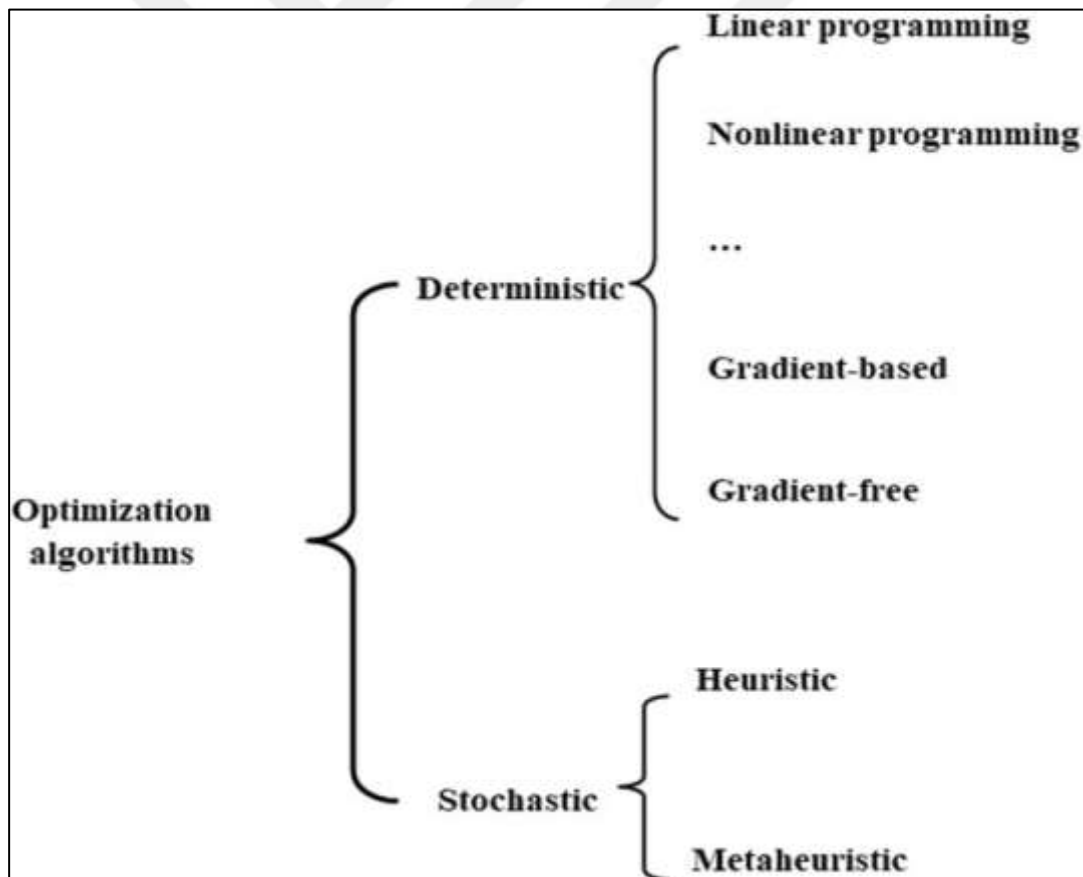


Figure 2.1: Family Tree of Optimization Algorithms [16].

Deterministic methods provide an absolute guarantee of success using information about the objective function obtained during the execution of the algorithm but inevitably require very

restrictive assumptions about the function can often be required to be the minimization process depends in part on probabilistic events, in the sense that the results provided are themselves random variables. In many cases the function is evaluated on a set of randomly selected points, this assessment may involve subsequent sample handling work.

A consequence of the introduction of probabilistic elements in a deterministic algorithm is the sacrifice of the absolute guarantee of success of the method.

Obviously, all methods, whether stochastic or deterministic, cannot do without ensuring some form of convergence. In some cases, it may be required that a method be competitive from an empirical point of view without providing guarantees of convergence (heuristic method), but this is not at all satisfactory from a numerical point of view. Ideally one would like to ensure the possibility of identifying an approximation of the global minimum in a finite number of steps and, under certain hypotheses, some deterministic methods are able to provide such a guarantee. In the majority of cases, however, the maximum that can be expected is an asymptotic convergence, ie with a computational effort tending to infinity. In general, the convergence of stochastic methods is of this type.

The latter definition does not require that convergence occurs for all points but only for a set to which unit probability is assigned. In the following discussion and when not otherwise specified, by asymptotic convergence we mean the convergence in probability.

Under very weak assumptions on the function and on the probability, distribution used for sampling, it is possible to demonstrate that the probability that a stochastic method determines an approximation of the global optimum approaches one as the sample size increases. In general, a method is characterized as stochastic if:

- a. test points are generated according to a particular probability distribution on St (TwoPhase, Random Search and Simulated Annealing methods) or along randomly chosen directions in R^{no} (Random Directions methods); or:
- b. the objective function f is considered as the realization of a stochastic process defined a priori (Random Function methods).

For both classes 1) and 2) there are numerous efficient algorithms [62] A brief description of the most interesting stochastic methods for the subsequent discussion will be given in the following chapter.

Obviously, it is not possible to say that stochastic methods are absolutely superior to deterministic ones; however, it is true that for large-scale problems the use of stochastic techniques is often the only possible approach.

2.1 PREVIOUS WORK ON HEURISTIC SEARCH ALGORITHM

Because there are so many different kinds of computer problems, and there are so many different methods to solve them, it is difficult to grasp all of the many possible techniques that may be taken to address them. Either an algorithm that generates a result that is exceptionally close to the correct answer or one that delivers a solution for certain situations but not for all of them is referred to as a heuristic algorithm.

One may find a wide variety of strategies that fall under this category. Some of these strategies are founded on more traditional ways of thinking, while others are more focused on achieving certain objectives. The fundamentals of conventional search algorithms will be discussed at the outset of this discussion. In the most fundamental search algorithm, which is known as exhaustive search, every possible response from a specified collection is put through a series of tests, and the answer that is deemed to be the most appropriate decision is selected. On the other hand, local search [1] places more emphasis on a particular locality, in contrast to comprehensive search, which spans the totality of the search universe. To arrange local search, there are several various ways that may be used. This category contains strategies that are well-known for climbing hills, and they are included in this category. The result of an adjacent algorithm will always be utilized instead of the one that is now being used if the result of the algorithm that is surrounding the current one is one that is superior to the present one. In the context of a peer-to-peer network-based multimedia distribution service, for instance, the hill-climbing technique serves as the basis upon which heuristics for the intra-group replication problem are constructed. Using the strategy of "divide and conquer" [2]-[3], algorithms try to reduce complex issues to meet their intended purpose. The problem is thus split down into components that are easier to handle because of this. In order to solve the primary issue, it is of the highest significance that the solutions to the issues at lower levels may be combined into a single solution. This method is successful; nevertheless, it is not especially practical since there are not many situations that can be divided and merged in such a clear manner. This is the reason why this strategy is not particularly practical.

The branch-and-bound [1],[3] strategy is a helpful tool for enumerating the search space, which is an important issue to take into account. It presents a list of all the viable solutions, but it makes a determined attempt to reject those that are not even very close to being the best. Dynamic programming [1],[3] eliminates the requirement for re-computation and makes it possible to conduct a comprehensive search within the target system. In order to achieve this goal, it is necessary to save the answers to the subproblems.

The idea that the process of finding a solution should be represented as a recursion is the key principle that supports this method.

When it comes to generating a space of solutions in a sequential way, the greedy methodology is one of the most prominent approaches.

At each stage of the algorithm, it is required to choose the option that is considered to be the best (locally) in order to arrive at the value that is optimum for the objective function on a global scale. This is the core idea that serves as the foundation for your argument. In circumstances where traditional approaches are unable to deliver adequate results within a fair amount of time, heuristic algorithms are typically applied with the intention of achieving the desired outcomes.

2.2 PREVIOUS WORK ON METAHEURISTIC SEARCH ALGORITHMS

2.2.1 Simulated Annealing SA

First suggested in 1983 by Kirkpatrick et al. [1] and then in 1985 by Cern [2], Simulated Annealing (SA) is a probability approach. Essentially, the goal is to obtain a minima of an objective function while avoiding local minima. This algorithm can be used to a wide range of optimization algorithms, not only those using VRPs. The annealing process in metallurgy is the inspiration for this method, which involves heating a solid to a point where it becomes liquid, then cooling it slowly and steadily until it returns to its solid state and the particles are re - arranged into a molecular structure with the least amount of energy [2]. A. Yu et al. [3] introduced a simulated annealing strategy in 2016 as a solution to the hybrid vehicle routing problem (HVRP), which is an offshoot of the Green Vehicle Routing problem. This problem occurs when hybrid vehicles are used to route vehicles. Their model takes into account the possibility of combining electric and gasoline power depending on the closeness

of charging stations for electric vehicles or petrol stations for conventional automobiles. Hybrids are the major focus of their attention, and their model takes into account the possibility of making this combination. The HVRP test cases were successfully addressed by the researchers through the construction of a SA with restart strategy, which was then coupled with Boltzmann and Cauchy functions in order to evaluate the acceptance likelihood of a solution that was deemed to be of inferior quality. The CVRP might potentially benefit from the application of a simulated annealing technique, as suggested by Wei et al. [4] in 2017. This is especially true in situations when two-dimensional loading restrictions are present. This technique combines a new packing strategy that is based on free space with a procedure that requires continually decreasing and rising the temperature. The result is a system that is both effective and efficient. Based on the findings of the computer research, it was determined that, in the majority of cases, the most well-known solutions may be accomplished or improved upon by combining a number of different tactics of this sort [5].

Table 2.1: Summary of Contributions in Hybrid SA Algorithms.

Author(s)	Advantages	Limitations
Yu et al. (2016)	Effective in solving HVRP test instances. Innovative use of hybrid power sources in routing.	May have high computational complexity. The approach may require fine-tuning of parameters for different instances.
Wei et al. (2017)	Capable of reaching or improving best-known solutions for most instances. Efficient handling of two-dimensional loading constraints.	Potentially higher computational time due to the repeated cooling and rising temperature mechanism. May be complex to implement.

2.2.2 Particle Swarm Optimization PSO

In the year 1995, Kennedy and Eberhart came up with the concept of Particle Swarm Optimization (PSO), with the objective of producing artificial intelligence through the utilization of social interaction analogies rather than individual cognitive abilities [7]. A approach known as solution optimization is one that focuses on continuously improving solutions in line with quality indicators that have been predetermined. As a means of

optimizing solutions, this method is utilized. Through the application of certain mathematical criteria to the location and velocity of a population of particles, the technique known as particle swarm optimization (PSO) is possible to exercise control over these solutions situated inside the search space. Because all of the particles are directed toward the places in the search space that are the most well-known, the entire swarm is pushed in the direction of the global minimum [8]. This is because the global minimum is the site that is most well-known. In the same way that flocks of birds flying in formation in search of food are analogous to swarms, swarms have a propensity to align themselves with the value of the fitness function that is optimal. And so on.

- a. In 2010, Marinakis et al. [9] developed an intriguing combination of the PSO algorithm and the multiple phase neighborhood search-greedy randomized adaptive search process (MPNS-GRASP). This combination was considered to be an interesting development. Expanding the neighborhood search technique and path relinking was the motivation for the development of this combo, which was designed in order to address the standard VRP problem. The fact that it was ranked first among all nature-inspired techniques for the VRP and sixth among the 39 most well-known and successful algorithms in the literature demonstrates that it is very suitable and effective for addressing extremely large-scale problems in a relatively short amount of time [9]. In addition, it was ranked sixth among the 39 algorithms that have been the most successful globally.
- b. An algorithm for the PSO was developed by B. MirHassani and Abolghasemi [10] for the Open VRP in the year 2011. Based on the results of this algorithm, it is guaranteed that vehicles will not go back to the depot after giving service to the last client on the route. They suggested a specific decoding approach that generates the customer position vector in descending order and always takes into consideration any limitations that may be imposed by the practicality of the scenario while assigning routes to all of the customers. This method was proposed by them. As far as the upgrading of the solutions was concerned, the implementation of the additional one-point adjustments to the routes that were designed also resulted in favorable consequences.
- c. The goal of the presentation made by Goksal et al. [11] in 2013 was to solve the VRP with simultaneous pickup and delivery. This was accomplished by presenting a novel mix of PSO and Variable Neighborhood Search. Additionally, in order to maintain the swarm variety, they adopted a technique that was comparable to annealing, and they

proposed a solution in the form of a huge tour that did not contain trip delimiters. Both of these methods help to keep the swarm variety. For this particular version of the VRP, 104 of the most well-known solutions were improved as a result of this method.

Table 2.2: Summary of Contributions in Particle Swarm Optimization (PSO) for VRP.

Author(s)	Advantages	Limitations
Marinakis et al. (2010)	Suitable and effective for solving very large-scale problems within short computational time. Ranked highly among known algorithms.	Complexity in integrating multiple techniques. May require significant computational resources for very large instances.
MirHassani and Abolghasemi (2011)	Promising in achieving better solutions. Efficient handling of Open VRP scenarios with feasibility constraints.	Specific to Open VRP, limiting its general applicability. Complexity in implementing the unique decoding method and extra moves.
Goksal et al. (2013)	Improved 104 best known solutions for this VRP variant. Innovative use of giant tour representation and annealing-like strategy.	Complexity in the algorithm due to combination of techniques. May require significant computational resources for maintaining swarm diversity.

2.2.3 Ant Colony Optimization Aco

It was in Dorigo's PhD thesis, which was submitted in 1992 [12], that the Ant Colony Algorithm (ACO) was initially introduced to the general public for the first time. This course of action was adopted in order to make advantage of the conduct that was seen in real life ever since it was brought to people's attention. The algorithms that were built were based on the method that ants use to travel between their colony and the sources of food. This method served as the inspiration for the algorithm. When an ant leaves behind a pheromone, it ensures that the ant will always take the same road, which is the one that is the shortest. However, this path may be different from time to time. Pheromones are a signal that wandering ants leave behind, and they are a signal that shows the quantity and quality of the

food that they have located along their journey. Pheromones carry information about the food that they have found. On the other hand, if you apply this idea to problems with automobile routing, it may be equated to searching for high-quality solutions in your own area. There is a chance that this will occur.

- a. In the year 2008, Donati et al. [13] developed the multi-ant colony systems that were used for the Time-Dependent Vegetation Response System. With the intention of accomplishing this objective, a mix of the original ACO technique and the idea of discretizing the temporal space into an adequate number of subspaces was applied. In addition to this, they made certain that the potential motions are studied in a continual way by including one-of-a-kind time-dependent local search techniques and criteria.
- b. Taking this action was done in order to ensure that the possible moves are taken into consideration. By including this method into such a model, a resilient shortest path strategy was also integrated for the purpose of generating time-dependent paths between each customer pair. This was done in order to accomplish the overall goal of the model. In order to ensure that the data did not include any errors, it was essential to carry out this procedure.
- c. Fuellerer et al. [14] applied the ACO methodologies to the two-dimensional loading VRP in 2009. They did this by updating and expanding upon the techniques that they had previously developed. These techniques were shown to have an outstanding degree of performance, as indicated by the findings of the extensive computer study. The versatility of their method allowed them to effectively manage a broad variety of loading limitations, including some that required turning goods and loading them from the back. This allowed them to successfully manage a wide range of loading constraints. As a result of this, they were able to arrive at significant qualitative findings that were associated with transportation. For example, they were able to determine the potential cost reductions that could be achieved by adopting alternate loading arrangements. Additionally, it was an early ACO strategy that successfully merged two distinct heuristic measures of routing and loading into a single pheromone matrix.
- d. This was a highly effective combination. The combination of the two measurements was the means by which this was achieved. At this location, a significant endeavor was successfully completed. The computational findings proved that their method significantly outperformed all other heuristics when applied to larger-scale scenarios.

This was shown by the fact that their strategy obtained optimal answers on smaller-scale conditions. It may be deduced from this that their approach performed noticeably better than any other heuristic.

- e. In order to address problems that are associated with road routing, Goel and Maini [15] came up with the concept of HAFA, which is a novel combination of the Ant Colony System and firefly algorithms.
- f. The goal of this combination is to find a solution to these problems. At the same time as the Firefly approach (FA) was being utilized in order to look for further solutions that had not yet been found, the ACS was being utilized as the foundation for the strategy that was being provided.
- g. In addition, the American Chemical Society (ACS) implemented a ground-breaking method of pheromone shaking in order to reduce the amount of pheromone stagnation that occurred on the geographical areas that were being targeted. It was because of this that the organization was able to prevent the formation of local minima.
- h. To address these issues, this combination is being used. Concurrently, the method that was being offered relied on the ACS as its basis, while the Firefly approach (FA) sought out other, undiscovered answers.

Table 2.3: Summary of Contributions in Ant Colony Optimization (ACO).

Author(s)	Advantages	Limitations
Donati et al. (2008)	Effective handling of time-dependent paths. Integrated robust shortest path algorithm.	Complexity in discretizing time space and ensuring feasible moves. May require significant computational resources.
Fuellerer et al. (2009)	Efficiently manages various loading restrictions	Complexity in combining different heuristic measures. May require significant computational resources for larger instances.

2.2.4 Bat Algorithm BA

A novel bioinspired metaheuristic, the Bat Algorithm (BA), was presented by Xin-She Yang in 2010 [16]. Microbats' echolocation activity, which they employ in the wild to locate prey, is the inspiration for this technique.

- a. To address the VRPTW, Taha et al. [17] integrated the LNS with a discrete bat algorithm to produce a hybrid approach in 2010. The issue was resolved using this approach.

Applying this approach allowed us to fix the issue. To improve the BA's performance and attain better outcomes, their software used the LNS's destroy and repair paradigm. If this were accessible, bats could explore a large chunk of the solution space. b. In a short period of time, a large number of Bat Algorithms were developed from scratch to handle various routing problems.

- b. During the year 2016, Zhou and colleagues [18] introduced a hybrid BA that included route Relinking for the Capacitated VRP. The design of the continuous bat algorithm, GRASP, and the route relinking approach that was included into the BA served as the foundation for this hybrid BA. It is necessary to carry out the random subsequences and the single-point local search with a certain degree of loudness in order to further improve the technique that was established.
- c. I.e. Wang et al. [19] reported a self-adaptive BA method for the truck and trailer routing issue in 2018. This technique was developed in 2018. This particular algorithm is a variant of the VRP that takes into account a collection of consumers located in different parts of the world as well as the vehicle fleet that provides services to those customers. 2018 was the year when the problem was first brought around.
- d. While the BA as proposed by Wang et.al, a local search mechanism that was carried out by five separate neighborhood structures was merged into those structures. This mechanism was carried out by the BA. In addition to this, a self-adaptive tuning mechanism was implemented in order to properly preserve the variety of the swarm.
- e. Hwang et.al [20] performed an in depth analysis of these algorithms and compared them to one another using metrics like MSE, RMSE, Convergence rate, Computational complexity and other metrics related to accuracy and time of execution.

Table 2.4: Summary of Contributions in Bat Algorithm (BA).

Author(s)	Advantages	Limitations
Taha et al. (2010)	Enhanced exploration of solution space. Effective combination of BA with destroy and repair paradigm of LNS.	Complexity in combining BA and LNS. May require fine-tuning of parameters.
Zhou et al. (2016)	Improved performance with random subsequences and single-point local search. Effective for capacitated VRP.	Complexity in integrating multiple methods. Computationally intensive due to additional random subsequences and local searches.
Wang et al. (2018)	Effective for truck and trailer routing problem. Maintained swarm diversity with self-adaptive tuning strategy.	Complexity in implementing self-adaptive tuning and multiple neighborhood structures. May require significant computational resources.

2.2.5 Firefly Algorithm FA

- a. Yang [20] came up with the concept for the Firefly Algorithm (FA) in the year 2008, and the flashing motion of fireflies served as the inspiration for the implementation of this algorithm. It is primarily for the purpose of attracting other flies and advertising their existence that fireflies flash their wings. Yang suggested the FA on the basis of the concepts that fireflies are monogamous, that brightness improves one's appeal, and that fireflies will wander aimlessly in the absence of other fireflies in the vicinity that are brighter than them.
- b. In order to find a solution to the TSP, Yati and Suyanto [21] devised an evolutionary discrete FA approach in the year 2011. This technique was designed by taking into consideration movement schemes in addition to the discrete distance that exists between two fireflies. When it was tried separately on a number of other TSP scenarios, the method that they utilized did not succeed in escaping from the local minima.
- c. Kumbharana and Pandey [22] conducted more research into this concept in 2013, when they presented the FA for the TSP. This was done in order to further investigate the topic. In order to do this, it was necessary to develop a proper translation of the continuous functions of attraction, distance, and velocity into new discrete functions. The FA algorithm for the TSP was changed in 2017 by B. Jie and colleagues [23], who included a swap operator and a swap sequence, as well as redefined the FA distance they

had previously used. This was done in order to prevent the algorithm from entering a state of local minima and to speed up the rate at which it arrives at a conclusion. In addition to this, they deployed a dynamic mechanism that was founded on a neighborhood search algorithm, which displayed a high degree of competition during the testing process.

Table 2.5: Summary of Contributions in Firefly Algorithm (FA) for TSP.

Author(s)	Advantages	Limitations
Yati and Suyanto (2011)	Introduced discrete distance and movement schemes for FA.	For some TSP test cases, I am stuck in a local minimum. Missing opportunity to combine with other approaches.
Kumbharana and Pandey (2013)	Enhanced discrete FA by converting continuous functions to discrete ones.	May still face challenges with local minima and convergence speed.
Jie et al. (2017)	Convergence speed is improved and local minima are avoided more effectively.	Complexity in implementation due to swap operator and dynamic mechanisms. Potentially higher computational requirements.

2.2.6 Genetic Algorithm GA

Genetic algorithms, sometimes referred to as GAs, are frequently referred to as optimization search algorithms. The motivation for these algorithms comes from natural selection. [24] The survival of stronger populations in many species can be attributed to the process of natural selection and its effects.

- a. During the 1960s and 1970s, Holland [25] established the foundation for GAs. After that, in 1975 and 1989, respectively, De Jong [26] and Golberg [27] further highlighted the applicability of GAs for resolving complicated problems. The first individual to lay the foundation for GAs was Holland [25], who was the pioneer himself.
- b. Thangiah and his colleagues [28] came up with the idea for the heuristic genetic algorithm known as GIDEON in 1991. Their intention was to solve the VRPTW problem with this approach. This method makes use of a cluster-first, route-second strategy, which is an enhancement on the process that is commonly referred to as Genetic Sectoring. Within the context of the problem of client vehicle assignment, it accomplishes this by employing local optimization strategies. In the process of using the

crossover method that was invented by GIDEON, chromosomes are randomly split, and subtours are shifted around.

- c. An adaptive fitness-scaling genetic algorithm with local search was reported by Wang et al. [29] in the year 2015. Using this method, the raw fitness value is transformed into a new value that is acceptable for further selection. This new value is utilized in the selection process. For the purpose of doing more research into the problem space, a local search strategy was utilized. In addition to this, an adaptive rates approach was applied in order to modify the crossover and mutation probabilities in accordance with the fitness value.
- d. The use of crossover operators by evolutionary algorithms to solve several versions of the VRP was the topic of a research that was given by C. Kumar and Panneerselvam [30] in the year 2017. The work was presented in the year 2017. Additionally, the authors presented a unique operator that they referred to as Sinusoidal Motion Crossover (SMC). This operator was introduced by the authors.
- e. This operator selects customers in a sequential manner from the first and second parents, which eventually leads to the production of two solutions at the end of the procedure. This operator functions in a manner that is comparable to the sinusoidal motion of waves in its underlying structure.

Table 2.6: Summary of Contributions in Genetic Algorithms (GA) for VRP.

Author(s)	Methods	Advantages	Limitations
Thangiah et al. (1991)	genetic algorithm heuristic for VRPTW	Effective cluster-first route-second approach. Improved local optimization methods.	May have limitations in large-scale problems due to random crossover points. Computationally intensive for fine-tuning parameters.
Wang et al. (2015)	fitness-scaling adaptive GA	Enhanced exploration and exploitation of problem space. Adaptive rates improve convergence.	Complexity in implementing fitness-scaling and adaptive rates. Requires significant computational resources.
Kumar and Panneerselvam (2017)	Sinusoidal Motion Crossover (SMC)	Innovative SMC operator for diverse offspring solutions. Effective for different VRP variants.	May require extensive computational resources. Complexity in implementing and fine-tuning the new crossover operator.

2.3 PREVIOUS WORK ON HYPER HEURISTIC SEARCH ALGORITHMS

It is usual practice to refer to the hyperheuristic as a "heuristic to choose heuristics," because it is a new paradigm in the field of optimization. This is because the hyperheuristic is a search algorithm. It does this by combining search techniques with learning approaches for the goal of creating or choosing heuristics in order to deal with optimization challenges. On the other hand, hyperheuristics traverse a space of heuristics, but metaheuristics avoid heuristics totally and examine a problem space directly. This is the primary distinction between the two types of heuristics. Heuristic generation and heuristic selection are the two basic categories of hyperheuristic techniques. More generally speaking, there are two primary kinds of hyperheuristic approaches. The heuristic selection technique chooses the heuristics that are the most suitable for each stage of the problem-solving process from a pool of LLHs that have been modified to fit the specific conditions that are now being encountered. The opposite is true, however, when it comes to heuristics generation techniques. These approaches automatically generate new heuristics from pre-existing heuristic components, and these heuristics are tailored to the particular task that is being carried out [29]. The two primary categories that may be utilized to subcategorize heuristic selection techniques are the constructive hyperheuristics and the local-search hyperheuristics.

Because there are so many different kinds of LLHs, this is now feasible. Similar to construction heuristics, constructive hyperheuristics begin with an imperfect solution and work their way up to a complete one by picking the right low-level construction heuristic (from a pool of LLHs) and using it to enhance the quality of the solution that has been produced. This process is repeated until the solution is complete. On the other hand, local-search hyperheuristics, which are analogous to improvement heuristics, begin with a solution that has already been produced and then, by utilizing local searches and the appropriate neighborhood structure, make an attempt to gradually enhance the quality of the response. This is done in order to get the desired result. It is possible that hyperheuristics, when applied on a problem-by-problem basis, have the ability to outperform both traditional heuristics and metaheuristics in the majority of instances. Specifically, this is because hyperheuristics has the capacity to determine the most efficient combination of attributes from a variety of low-level heuristics. This is the reason why this is the case.

The essential assumption that underpins hyperheuristics is the utilization of a wide variety of LLHs that are not only uncomplicated but also adaptable, in addition to the development of a framework that can manage the selection and application of LLHs. Problems involving optimization can be solved with the help of this framework. In the majority of instances, search strategies that encompass the whole heuristic search space are applied. These search strategies include high-level heuristics and metaheuristics. It was Tailong et al. [31] that came up with the idea of developing a multi-objective hyper-heuristic algorithm that they called HH_EG with the goal of solving MOPs. The ideas of adaptive epsilon-greedy selection are used as the foundation for this strategy. At the same time as this research proposes an adaptive epsilon-greedy selection technique as a method for choosing and mixing LLHs during the course of the evolutionary process, it also suggests that this approach be used. Not only was it feasible for the proposed hyper-heuristic to address issues across a wide range of domains by only modifying the LLHs, but this was also achieved without the need to redo the whole approach. The HH_EG framework does not need any parameter adjustments to be carried out, in addition to the fact that it is easy to link with a wide range of performance indicators. For the purpose of putting HH_EG through its paces, the authors deployed a number of different test suites, including those for traditional DTLZ, IMOP, and many-objective MaF, in addition to a multi-objective issue that was based on the actual world. The experimental findings reveal that HH_EG is an excellent instrument for addressing issues that span several domains. This is because it incorporates the most desirable qualities of each LLH, which is possible because of the incorporation of these characteristics.

In the year 2020, Chen et al. [32] established a framework for a steel production scheduling system that is targeted toward cyber-physical systems. This framework was developed with the intention of resolving the challenge of dynamic scheduling in complex environments and providing ways for incorporating intelligent manufacturing into the production of steel. After conducting an examination of the features of dynamic steel production scheduling models, a knowledge model for scheduling production in a cyber-physical system that is founded on ontologies was presented, along with a method for expressing the qualities of the system through the use of ontologies. It was claimed that both of these pieces of information were available. It was specifically for the sake of dynamic scheduling that scheduling heuristics were developed.

Within the confines of this methodology, a hyper-heuristic algorithm that is based on genetic programming is described. In order to exert control over the low-level heuristic process, we made use of an approach that was based on learning and was a high-level selection strategy. In addition, a system that is based on genetic programming is built in order to automatically generate great heuristic rules and to manage scheduling issues that are brought about by a range of disruptions to production. This is done in order to manage the challenges that scheduling presents. After all was said and done, the efficiency of the algorithm was evaluated by means of a simulated situation to ensure that it was effective. The green vehicle routing problem with simultaneous pickup and delivery (G-VRPSPD) was the subject of study that Busra and his colleagues [33] carried out in the year 2021. The G-VRPSPD was the name they gave to this particular task. Without compromising the effectiveness with which fuel is utilised, the objective of this endeavour is to fulfil the requirements that clients have for either pickup or delivery of their orders.

The quantity of greenhouse gases that are released into the atmosphere is correlated with the amount of fuel that is used. Iterative local search and variable neighbourhood descent heuristics are the foundations upon which the hyper-heuristic (HH-ILS) technique that the author has created is built. This approach was developed by the author. This strategy is intended to solve the problem that has been provided in a manner that is composed of mathematical principles.

The utilisation of computer studies makes it possible to carry out a comprehensive investigation of the effects that the G-VRPSPD and the HH-ILS have had on the body of knowledge. Based on this comparison, we investigate the effect that a green goal function has on the total cost of fuel consumption by comparing the G-VRPSPD to the VRPSPD. This comparison allows us to explore the influence that a green goal function has. Through the process of conducting a comparative research study, we are able to carry out our investigation on the effectiveness of HH-ILS. Additionally, in order to further investigate the effectiveness of neighbourhood structures, hyper heuristic, and local search, we do sensitivity analysis. During the same time, this is carried out. In light of the information presented, it is possible to draw the conclusion that the green goal function exerts a substantial impact on the overall cost of petrol consumption. [2]. Classical optimization algorithms, referred to as deterministic methods, possess the capability to ensure the

discovery of the optimal solution to an optimization problem under the condition of unlimited time and resources. Nevertheless, their effectiveness is limited to problems characterized by linear search spaces (unimodal), rendering them vulnerable to the entrapment of local optima when employed in non-linear search spaces, such as real-world non-convex problems [3].

Stochastic methodologies were developed to efficiently approximate solutions to optimization problems within reasonable time and resource constraints. However, the use of these tactics does not guarantee the optimal solution. Stochastic approaches often need a pre-existing comprehension of the given issue [4] [5]. To restate this notion, it should be noted that although heuristics may demonstrate efficacy in resolving a particular issue, their applicability may not extend to other contexts.

Table 2.7: Summary of Hyperheuristics and Optimization Techniques.

Author(s)	Advantages	Limitations
Lee et.al (2022)	They can discover a good combination of best characteristics of individual low-level heuristics. High-level heuristics and metaheuristics are used as search methods across the search space of heuristics.	Hyperheuristics involve complexity in managing and combining multiple low-level heuristics. They may require significant computational resources for large-scale problems.
Tailong et al. (2021)	Does not need to tune parameters, and is easy to be integrated with various performance indicators.	May still face challenges with convergence speed and handling highly complex problems.
Chen et al. (2020)	Addresses the difficulty of dynamic scheduling of steel production in a complex environment.	Complexity in integrating cyber-physical systems with hyper-heuristic algorithms. Requires significant computational resources.
Busra et al. (2021)	Significant reduction in fuel consumption cost. Competitive performance compared with mathematical formulation and state-of-the-art heuristics.	Complexity in implementing and fine-tuning the HH-ILS algorithm. Requires significant computational resources for extensive computational experiments.

In contrast, metaheuristic is seen as a black-box optimization technique or a versatile algorithmic framework that may be used to a diverse range of optimization circumstances. the swarm intelligence (si) technique, a branch of metaheuristics, draws inspiration from the collective behaviors shown by several biological and natural evolutionary phenomena. examples of such phenomena include swarms of worms, bees, and ants, schools of fish, and flocks of birds [2]. the primary goal of swarm intelligence is to enhance the efficiency of a system by the assessment of the combined behaviors shown by its constituent elements. the popularity of algorithms based on swarm intelligence (si) has increased in recent years. this may be attributed to their ability to learn, quick convergence, flexibility, simple structure, insensitivity to starting parameters, and adaptability to external perturbations

2.4 MULTI OBJECTIVE OPTIMIZATION

The field of optimization algorithms has been a subject of extensive study for a considerable period of time. The significance of optimization algorithms has grown significantly in recent times due to technological developments and the escalating intricacy of situations. In recent times, there has been a proliferation of novel trends within the domain of optimization algorithms, which possess the capacity to fundamentally transform the conventional approaches used for solving optimization issues. The use of deep learning techniques for optimization purposes. The use of optimization algorithms based on deep learning has seen significant growth in recent years, mostly attributed to their capacity to effectively address intricate and high-dimensional problem domains. These algorithms use the capabilities of deep learning in conjunction with optimization approaches to identify optimum solutions. Two widely used strategies for addressing optimization issues are Deep Learning-based Particle Swarm Optimization (DL-PSO) and Deep Learning-based Genetic Algorithm (DL-GA). Significant advancements have been made in the field of multi-objective optimization in recent years. The increasing prevalence of intricate systems has led to optimization challenges that often include the simultaneous optimization of numerous competing goals. As a reaction, scholars have devised novel algorithms capable of concurrently addressing numerous goals. The algorithms included in this category consist of Multi-objective Evolutionary Algorithms (MOEAs), Multi-objective Particle Swarm Optimization (MOPSO), and Multi-objective Genetic Algorithms (MOGAs). Hybrid optimization methods refer to the amalgamation of several optimization techniques, often two or more,

which are used in tandem to address intricate optimization issues. The popularity of these algorithms has increased owing to their capacity to effectively address various optimization challenges. A technique known as Hybrid Particle Swarm Optimization and Artificial Bee Colony (PSOABC) has been developed to address intricate optimization issues by using the combined advantages of PSO and ABC. The discipline of quantum optimization has garnered considerable interest in recent years, signifying its emergence as a prominent area of study. The process involves the use of quantum computing for the purpose of resolving optimization challenges. Quantum optimization algorithms are specifically developed to use the distinctive attributes of quantum computing, including superposition and entanglement, in order to identify optimum solutions. Quantum optimization algorithms include several approaches, such as Quantum Annealing, Quantum-Inspired Evolutionary Algorithms, and Quantum Genetic Algorithms. Nature-inspired optimization algorithms refer to algorithms that draw inspiration from natural processes, including but not limited to evolution, swarm behavior, and genetics. The popularity of these algorithms has increased owing to their efficacy in addressing intricate issues that pose challenges for conventional optimization strategies. Nature-inspired optimization algorithms include a variety of methodologies, such as Genetic Algorithms (GAs), Particle Swarm Optimization (PSO), Ant Colony Optimization (ACO), and Artificial Bee Colony Optimization (ABC). These algorithms draw inspiration from natural processes and phenomena to solve complex optimization problems. The Genetic Algorithm (GA) is widely recognized as one of the first nature-inspired optimization algorithms to gain significant attention in research. The genetic algorithm (GA) encompasses many fundamental processes, including 'mutation' and 'selection', which play crucial roles in the algorithm's execution. For the purpose of creating sessions of recurrent transcranial magnetic stimulation (rTMS), Squires et al. (2022) introduced a novel genetic strategy. This method was developed with the objective of designing it. Both a one-of-a-kind method for selecting winners and a heuristic for populating the population are included in the genetic algorithm that is used for the List Scheduling Wildcard Tournament. Both of these elements are integrated into the algorithm in some way. The LSM-GA is the name given to the final outcome, which is a more accurate prediction of the winner at the end of the process. Recombining them is the procedure that is being used. The word "mutation" conveys the idea that the outcome will be somewhat sensitive to the impact of chance simply by virtue of the fact that it is used.

When compared to alternative approaches to problem resolution, metaheuristic algorithms have a reaction time that is significantly longer. The research that was carried out by Kaul and colleagues [8] had the objective of shedding light on the many tactics that are utilised to overcome restrictions, the relevance of metaheuristic algorithms in this field of study, and the several ways that are advocated for categorising metaheuristics. In the course of this study, a wide range of interesting applications and examples of metaheuristic algorithms are studied in depth over the course of this investigation. The authors Kaur et al. (2019) conducted research to determine the ways in which the Whale Optimisation Algorithm (WOA) may be improved by adding notions from chaos theory. For the purpose of optimising the most important parameter of the WOA, the Chaotic Whale Optimisation Algorithm (CWOA) is a collection of recommended algorithms that make use of a wide variety of chaotic maps. This technique guarantees that the exploration and exploitation components of the algorithm are in a state of equilibrium during the course of the algorithm. Using a battery of twenty typical test functions, the feasibility of the recommended CWOA techniques was examined in relation to the approaches that are currently regarded to be state-of-the-art. This test was carried out in order to determine whether or not the offered techniques are feasible. The results of the study indicate that the WOA algorithm could be able to perform more successfully with chaotic maps, and the Tent map in particular ought to be taken into consideration. The rotational hyper-ellipsoid function was one of the six well-known mathematical functions that were tested using the quantum dolphin swarm algorithm (QDSA) that was carried out by Qiao W et al. [10] in order to see how well the algorithm performed. The purpose of the test was to determine how well the algorithm functioned. QDSA may be useful for discovering the best solution to somewhat large functions, according to the data, which show that it may be successful. The results of the investigation led to this conclusion. Metaheuristic algorithms can be fine-tuned in order to address problems that include two fitness functions or, more generally, circumstances that involve several objectives that need to be optimised. This is an option that can be considered. These algorithms have a number of fundamental goals, one of the most important of which is to find a collection of solutions that takes into account all of the possible compromises that may be made between different goals. Among the many well-known multi-objective metaheuristic algorithms that are included in this list are the following types of algorithms: The Non-dominated Sorting Genetic Algorithm II (NSGA-II) [11] is one of the algorithms

that is regarded to be among the most popular when it comes to evolutionary computation. It is also one of the algorithms that has been around the longest. This is one of the most well-known evolutionary techniques, which was created expressly to solve problems that require optimisation with many goals. In order to allow the effective administration of a diverse collection of solutions, the strategy that is currently being utilised provides a method for non-dominated sorting. This method assists in the management of the solutions. In addition, it makes use of elitism in order to protect the search results that are in the best possible position. One of the numerous challenges that NSGA-II must handle is scalability. There are many other challenges as well. In circumstances in which there are a great deal of objectives, it is feasible that it will not work as efficiently as it may. One possible explanation for this is that the increasing complexity of non-dominated sorting is to responsible for a substantial chunk of the problem. One of the most important aspects of evolutionary algorithms, which are employed to find solutions to problems, is the fact that they preserve variation. There is a possibility that the NSGA-II will not perform perfectly in circumstances that entail complex Pareto fronts, despite the fact that it makes use of crowding distance in order to maintain variety. When dealing with high population numbers, there is a likelihood that considerable processing issues may occur, particularly with regard to non-dominated sorting and crowding distance calculation [11]. This is especially prevalent when dealing with crowding distance computation. Researchers developed a new method known as Multi-Objective Particle Swarm Optimisation (MOPSO), which is an adaptation of the original particle swarm optimisation (PSO) methodology [12]. The development of MOPSO was motivated by the need to find solutions to optimisation problems that encompassed many objectives. A global best strategy and a personal best strategy are merged in this method, which additionally makes use of an external archive to store responses that are not dominated by any single group. In addition, this method makes use of an external archive. There are many different kinds of limitations that can be found, including the following: Due to the construction of particle swarm optimisation, there are some situations in which it may be difficult for MOPSO to converge on the true Pareto front. These situations might pose a challenge for the optimisation algorithm. In order to guarantee that the Multi-Objective Particle Swarm optimisation (MOPSO) optimisation strategy provides the best possible performance, it is essential to undertake fine-tuning on the settings. Both the acceleration coefficients and the inertia

weight are crucial characteristics that need to be carefully analysed. Both of these quantities are interrelated. Swarm optimisation presents a huge barrier to the preservation of variety since particles have a propensity to congregate in certain regions of the search space. This is one of the reasons why swarm optimisation is so difficult to implement. The Strength Pareto Evolutionary Algorithm 2 (SPEA2) is an evolutionary algorithm that is frequently used for the purpose of optimising a variety of various goals. Its most common use is in the field of optimisation. The utilisation of a strength-based fitness assignment in conjunction with a cautious elitist strategy makes it possible to keep a diverse collection of solutions that are not dominated by any one particular group [13]. The following is a list of the most significant restrictions that are connected to the use of SPEA2: Regarding the level of computational difficulty, there is a difference of opinion. When applied to extremely large populations, the fine-grained elitism approach and the strength-based fitness assignment both have the potential to utilise a huge amount of computer resources. This is because both of these techniques are based on muscle strength. The size of the archive and the number of nearest neighbours for the fine-grained elitism technique are two factors that have the ability to significantly influence the efficiency of the algorithm. Both of these factors are important in determining the success of the algorithm. Both of these criteria are modified when the parameters have been modified to reflect the new values. It is probable that the performance of SPEA2 and NSGA-II will decline when they are presented with scenarios that contain a high number of goals. This is something that ought to be considered. There is also a worry regarding the adaptability of increasing businesses. This restriction is a result of the difficulties involved in maintaining a diverse collection of potential answers that are not dominated by any particular option. For more specific purposes, the abbreviation MOEA/D is used to refer to a computational method that is referred to as Multi-Objective Evolutionary Algorithm Based on Decomposition [14]. The technique that was described in this inquiry employed a strategy that involved breaking down the problem of multi-objective optimisation into a number of smaller problems that had individual objectives. This method was used to solve the problem. Next, evolutionary optimisation algorithms are implemented in order to concurrently handle various subproblems. This is followed by the implementation of these tactics. Obtaining an estimate of the Pareto front is accomplished by the application of the Tchebycheff method. It is possible to claim that MOEA/D is a legitimate approach, despite the fact that there are a few limitations to take

into consideration. The deconstruction process is employed in order to make challenging situations more bearable for the one who is experiencing them. It is conceivable for the decomposition approach that is employed to have a major impact on the performance of the algorithm, and the efficacy of the technique may vary depending on the work that is being processed. This is something that is feasible. For the purpose of maximising the benefits that MOEA/D has to offer, it is essential to make adjustments to its settings. It involves carefully selecting appropriate values for parameters such as the size of the neighborhood and the weight vectors.

The aforementioned algorithms have the capability to be used in issue domains including two fitness functions, as well as in domains encompassing more than two goals. The purpose of its design is to identify a collection of optimum solutions that effectively reconcile several goals, so offering a broad spectrum of feasible resolutions to the given issue. The selection of an algorithm is contingent upon the particular situation at hand, the intended balance between goals, and the computing resources at one's disposal. For a long time, researchers have poured resources into studying optimisation methods. As a result of both technical advancements and the increasing complexity of circumstances, optimisation algorithms have become more important in recent years. Recent years have seen a plethora of fresh developments in optimisation algorithms, with the potential to radically alter the standard methods previously employed to resolve optimisation problems. Optimisation by the application of deep learning methods. The ability of deep learning optimisation algorithms to tackle complex, high-dimensional problems has led to their increased popularity in recent years. To get the best answers, these algorithms combine optimisation techniques with deep learning's capabilities. Deep Learning Particle Swarm Optimisation (DL-PSO) and Deep Learning Genetic Algorithm (DL-GA) are two popular methods for solving optimisation problems. The area of multi-objective optimisation has recently seen tremendous progress. Optimisation problems, brought on by the proliferation of complex systems, sometimes involve the optimisation of many conflicting objectives at once. Researchers have responded by developing new algorithms that can handle many tasks at once. Alphabets like MOEAs, MOPSOs, and MOGAs (Multi-objective Genetic Algorithms) are all part of this class of algorithms. When dealing with complex optimisation problems, hybrid optimisation approaches employ a combination of optimisation techniques, usually two or more. These algorithms have become more popular since they can solve many optimisation

problems. To solve complex optimisation problems, researchers have created a method that combines PSO with ABC, called Hybrid Particle Swarm Optimisation and Artificial Bee Colony (PSOABC). A new field of research known as quantum optimisation has emerged in the last several years, thanks to the massive amount of attention it has received. Applying quantum computing to optimisation problems is what this method is all about. To find the best solutions, quantum optimisation algorithms take advantage of superposition and entanglement, two features unique to quantum computing. Several methods fall under the umbrella of "quantum optimisation algorithms," including QGAs, QIEAs, and quantum annealing. Optimisation algorithms that take their cues from natural phenomena are known as "nature-inspired." These processes might include things like evolution, swarm behaviour, and genetics. Because of how well these algorithms handle complex problems that traditional optimisation methods struggle with, their popularity has skyrocketed. Optimisation algorithms that draw inspiration from nature encompass a wide range of approaches, including Genetic Algorithms (GAs), Particle Swarm Optimisation (PSO), Ant Colony Optimisation (ACO), and Artificial Bee Colony Optimisation (ABC). In order to resolve intricate optimisation issues, these algorithms take their cues from events and processes found in nature. Many researchers consider the Genetic technique (GA) to be an early and groundbreaking optimisation technique that drew inspiration from nature. 'Selection' and 'mutation' are two of the genetic algorithm's (GA) base operations that are essential to the algorithm's operation. Squires et al. (2022) presented a new genetic approach to train recurrent transcranial magnetic stimulation (rTMS) sessions. Its design was the driving force for the development of this approach. In the List Scheduling Wildcard Tournament, the evolutionary algorithm incorporates both a unique way of choosing winners and a heuristic for expanding the population. The method makes use of both of these components. At the end of the procedure, a more accurate winner prediction is achieved; this improved prediction is called the LSM-GA. The current method involves recombining the two. Just by using the word "mutation" suggests that the result will be partly susceptible to the influence of chance. The reaction time of metaheuristic algorithms is much longer than that of other methods for solving problems. The goal of the work by Kaul and colleagues [8] was to classify metaheuristics in a variety of ways, throw light on the many strategies used to circumvent constraints, and establish the importance of metaheuristic algorithms in this arena. Research in this area focusses on a variety of intriguing uses and

examples of metaheuristic algorithms. To find out how the Whale Optimisation Algorithm (WOA) may be enhanced by incorporating concepts from chaos theory, the writers Kaur et al. (2019) performed study. The goal of the Chaotic Whale Optimisation Algorithm (CWOA) is to optimise the WOA's most crucial parameter. It is a set of suggested algorithms that utilises various chaotic maps. This method ensures that the algorithm's exploration and exploitation parts remain balanced as the algorithm runs. The suggested CWOA methods were evaluated for practicability in comparison to the state-of-the-art methods using a battery of twenty common test functions. In order to find out if the proposed methods are workable, this test was executed. According to the study's findings, chaotic maps should be included by the WOA algorithm for improved performance; specifically, the Tent map. As part of their evaluation of the quantum dolphin swarm algorithm (QDSA), Qiao W et al. [10] used the rotating hyper-ellipsoid function as one of six famous mathematical functions. Finding out how efficiently the algorithm worked was the goal of the test. Based on the statistics, it appears that QDSA might be effective in finding the optimal solution to rather big functions. We arrived at this conclusion based on the findings of the inquiry. Metaheuristic algorithms can be adjusted to handle situations with multiple objectives that require optimisation, or issues with two fitness functions. This is a potential choice to think about. Among the many primary purposes of these algorithms is the discovery of a set of solutions that accounts for every conceivable trade-off between competing objectives. These algorithms are only a few examples of the many popular multi-objective metaheuristic algorithms available: Within the realm of evolutionary computation, the Non-dominated Sorting Genetic Algorithm II (NSGA-II) [11] is well recognised as one of the most prominent algorithms. Additionally, it has been around for a longer period of time than other algorithms. One of the best-known evolutionary methods, it was developed to address optimisation issues with multiple objectives. The current technique offers a way for non-dominated sorting, which is necessary for the proper administration of a wide collection of solutions. The administration of the solutions is aided by this technique. Even more so, it employs elitism to safeguard the top search results. Scalability is one of the many obstacles that NSGA-II has to overcome. A great deal of additional difficulty also exists. Possibly it won't function as well as it might in situations when there are a lot of goals. This might be because a large portion of the issue is caused by the growing complexity of non-dominated sorting. Evolutionary algorithms are used to solve problems, and one of their most essential

features is that they maintain variation. Though it employs crowding distance to preserve diversity, the NSGA-II may not function flawlessly in situations with complicated Pareto fronts. Processing problems, especially with non-dominated sorting and crowding distance computation, are likely to arise when working with large population numbers [11]. When it comes to crowding distance calculations, this is quite common. Multi-Objective Particle Swarm Optimisation (MOPSO) is a novel approach that researchers have created. It is an extension of the original particle swarm optimisation (PSO) technique [12]. Finding answers to optimisation challenges that covered several objectives drove the creation of MOPSO. This approach combines a global best strategy with a personal best strategy, and it uses an external archive to keep replies that are not dominated by any groups. Furthermore, an external archive is utilised by this approach. Some examples of the various constraints that could be encountered are: In certain cases, MOPSO might not be able to converge on the genuine Pareto front because of how particle swarm optimisation is built. The optimisation method may encounter difficulties in certain cases. To get the optimum performance from the Multi-Objective Particle Swarm optimisation (MOPSO) optimisation approach, fine-tuning the parameters is a must. Important features that must be thoroughly examined are the inertia weight and the acceleration coefficients. There is a relationship between the two numbers. There is a significant obstacle to variety preservation in swarm optimisation due to the fact that particles tend to cluster in specific areas of the search space. One of the many reasons swarm optimisation is challenging to apply is because of this. One popular evolutionary method for maximising a wide range of objectives is SPEA2, or Strength Pareto Evolutionary method 2. Optimisation is its primary domain of application. To ensure that no one set of solutions becomes preponderant, it is recommended to use a strength-based fitness assignment in tandem with a careful elitist strategy [13]. The following are some of the most important limitations associated with SPEA2: Opinions disagree on the degree of computational complexity. Both the fine-grained elitism method and the strength-based fitness assignment can use a lot of computing power when applied to very big populations. Reason being, power of the muscles is fundamental to both methods. Among the many variables that could drastically affect the algorithm's performance are the archive's size and the fine-grained elitism technique's number of nearest neighbours. If the algorithm is to be successful, both of these things must be considered. When the parameters are changed to reflect the new values, both of these criteria are amended as well. Presented

with situations with a large number of goals, SPEA2 and NSGA-II are likely to perform worse. It is important to take this into account. The flexibility of expanding companies is another source of concern. This limitation arises from the challenges associated in preserving a varied set of possible solutions that are not dominated by any one choice. To be more precise, a computational approach called Multi-Objective Evolutionary Algorithm Based on Decomposition is abbreviated as MOEA/D [14]. This investigation detailed a method that used an approach that included partitioning the multi-objective optimisation issue into several subproblems, each with its own set of objectives. The issue was resolved using this approach. Assuming a few caveats are considered, MOEA/D may be a valid strategy. When dealing with difficult events, the deconstruction process might help the person going through it. The effectiveness of the strategy may differ based on the task being processed, and it is possible for the decomposition approach to significantly affect the algorithm's performance. This can definitely be done. Optimising MOEA/D's performance requires fine-tuning its parameters to their full potential. The neighbourhood size and the weight vectors are two factors whose values must be carefully chosen. Any problem domain with two fitness functions or more than two objectives may be solved using the aforementioned approaches. Its design's objective is to find a set of best solutions that successfully combine many objectives, providing a variety of workable answers to the problem at hand. When choosing an algorithm, it's important to consider the current scenario, the goals that need to be balanced, and the computational resources that are available.

3. MATERIALS AND METHODS

3.1 INTRODUCTION

An NP-hard combinatorial issue can be solved using a meta-heuristic, which is a general approach. In order to find solutions to these issues, an exponentially vast number of possibilities must be examined. We have all encountered the combinatorial explosion phenomena, which, as the complexity of the issue to be addressed increases, turns an apparently easy task into a genuine pain. When attempting to create a schedule, for instance, this is the situation. When there aren't many lessons to prepare, there aren't many permutations to try, and the issue is fixed fast. But even a little increase in the number of classes might cause the solution time to balloon out of control due to the sheer volume of possible permutations. The stakes are high because combinatorial explosion is at the heart of many industrial problems. For instance, there are those that involve production planning with a focus on minimizing time losses or material shape cutting with an eye toward minimizing waste. Solving these challenging issues in a reasonable amount of time requires the development of "intelligent" methods that may either limit or avoid the combinatorial explosion. There are two primary classes of methods that may be used to resolve combinatorial optimization issues. To get the best possible outcome, "exact" methods methodically investigate all possible permutations. These approaches (try to) limit the combinatorial explosion by constructing a tree out of the space of possible combinations, then using pruning techniques to narrow it down, and heuristics to figure out what order to explore it in. These precise methods enable the resolution of several practical combinatorial optimization issues. Filtering methods and sorting heuristics are exhaustive procedures, but they do not always reduce combinatorics enough, and therefore cannot handle some problem cases in a reasonable amount of time. To avoid the combinatorial explosion issue, another option is to employ meta-heuristics that selectively explore a subset of the combination space. That being said, they aren't guaranteed to discover the optimal solution, much less demonstrate that it is optimal; conversely, the time complexity is often weakly polynomial. The chapter is structured. In 3.2, we will see perturbative heuristics, which build combinations by changing existing combinations; in 3.3, we will see constructive heuristics, which develop combinations incrementally using a stochastic model. We show in section 3.4 a few of the most well-known hybridization schemes that combine these various methods.

In section 3.5, we present the concepts of intensification and diversification, which are shared by all these heuristic methods. Intensification seeks to focus the search on the most effective combinations, while diversification ensures a good sample of the search space. Lastly, in section 3.6, we will create two uses for these meta-heuristics to solve two well-known AI problems: determining if a Boolean formula is satisfiable and finding a way to satisfy constraints. Examine the results. For the remainder of this chapter, we will proceed with the premise that the issue at hand is characterised by a pair (E, f) , where E is a collection of potential combinations and $f: E \rightarrow \mathbb{R}$ is an objective function that gives a numerical value to each combination of E . This assumption will guide our work. The key to solving this kind of problem is to locate the ideal combination of elements $e \in E$ that either maximises or minimises f . The application of the several meta-heuristics that have been given in this chapter to the travelling salesman issue will serve as our "common thread," and we will show them as follows: In the event that we possess a set of cities V and a function $d: V \times V \rightarrow \mathbb{R}$ that provides the $\text{distanced}(i, j)$ between any two cities $i, j \in V$, it is necessary for us to ascertain the shortest circuit that traverses each city V precisely once. The objective function that must be minimised is stated as the sum of the distances between two subsequent cities in the permutation set. On the other hand, the set of circular permutations of V is what determines the whole set of feasible combinations.

3.2 DISRUPTIVE META-HEURISTICS

The range of potential combinations may be explored in a number of different ways, one of which is by the iterative disruption of previously formed combinations from disruptive techniques. The purpose of these methods is to produce new combinations by changing old ones, and they begin with one or more initial combinations, which are normally selected at random from E . The goal is to generate new combinations. "Instance-based" is the term that is used to describe these kinds of procedures in reference [77]. Both genetic algorithms (3.2.1) and local search (3.2.2) are the two perturbative approaches that are generally recognised.

3.2.1 Genetic Algorithms

Genetic algorithms [32],[24],[14] are inspired by the theory of evolution and the rules of genetics which explain the ability of living species to adapt to their environment by the combination of the following mechanisms:

- a. natural selection means that individuals best adapted to the environment tend to survive longer and therefore have a greater probability of reproducing.
- b. reproduction by crossing means that an individual inherits its characteristics from its parents, so that the crossing of two individuals well adapted to their environment will tend to create a new individual well adapted to the environment.
- c. mutation means that certain characteristics can appear or disappear randomly, thus making it possible to introduce new capacities for adaptation to the environment, capacities which can propagate thanks to selection and crossing mechanisms.

A meta-heuristic is defined by genetic algorithms using these techniques. The goal is to maximize a function by gradually improving a population of possible combinations through natural selection, genetic engineering, and mutation. Below, we outline the primary phases of Algorithm 1, which explains this overarching notion. Population initialization: typically, a uniform distribution is used to produce the initial population, guaranteeing a good diversity of combinations.

In the selection phase, we pick out the population combinations that we think would be good candidates for mutation and recombination. The goal is to make the best possible combinations more likely to be selected while still giving the less desirable ones a little opportunity. This selection stage can be approached in several ways. Tournament selection, for instance, involves picking the best of two possible permutations at random (or, depending on the objective function, picking one of the two permutations according to probability). Other factors, such as variety, might also be used to make the selection. Here, the only ones who can make it through to the next generation are the so-called "distanced" ones. This step tries to generate new people by mixing selected combinations during recombination, also known as crossover. Moving the search to a different region of space in the hopes of finding better combinations is the objective. In order for the resulting hybrid to have optimal performance, the recombination process has to be tailored to the task at hand and able to pass

on desirable traits from both parents. In an ideal world, the recombination operator would also permit the generation of offspring with varying genetic makeups. From an exploration and exploitation standpoint, recombination is meant to serve as a strategic diversifier leading up to improved intensification in the long run. Mutation is the process of altering the combinations that are produced by crossing by randomly changing some of their components. Illustration 1 To implement a basic recombination operator for the traveling salesman, one has to replicate a subsequence from the first parent and then arrange the missing cities in the same order as they occur in the second parent. Picking a few of cities at random and switching them around is the standard mutation operator. The purpose of the population update is to identify which existing combinations should be removed and which new combinations should be added. In order to keep the population diverse, keep the search from convergent too soon, and make sure the algorithm can continually find new potential regions in the search space, the update policy is crucial. As a result, variety and quality are typically considered while making judgments. An example of a well-known quality-only updating rule would be to replace the population's worst combinations with new, similar ones, according to a specific similarity measure; an example of a diversity-based approach would be to replace the population's old combinations with new, similar ones. Age is only one of many possible additional factors to think about. When an evolutionary process reaches a stopping condition, it stops. This happens after several generations of iteration. There may be a limit to how many assessments a model can undergo, how many generations it can go without becoming better at finding optimal solutions, how diverse a population can be, or some other metric.

3.2.2 Local Search

Starting with an initial combination and choosing a close-by combination, acquired by performing an elementary transformation, at each iteration, a local search iteratively searches the space of combinations [33]. The next sections outline the primary procedures of Algorithm 2, which elucidates this overarching concept. A neighborhood function $v: E \rightarrow P(E)$ defines the set of possibilities that may be investigated from a particular combination and is used to parameterize the local search method. When we apply a "elementary" alteration to e , we may obtain $v^{(e)}$ which is a collection of combinations, given a common combination $e \in E$. Changing the value of a variable or swapping the values of two variables

are two examples of the many possible modification operators that we may often think about. There may be more or fewer combinations that are comparable to the present combination depending on the modification operator that is used, as each operator produces a distinct neighborhood. Selecting an appropriate neighborhood operator so significantly affects the algorithm's efficiency. In a perfect world, the neighborhood operator would allow us to get to the best possible combination from any starting set of elements. This would mean that any vertex in the directed graph that has a connection to a set of elements and a bow(e_i, e_j) to every set of combinations where e_j is a member of $v(e_i)$ could lead to the best possible combination.

Second Example The 2-opt operator for the traveling salesman issue is two new edges that join the two pathways that were formed by removing the old ones, and two edges that are deleted are replaced. In a broader sense, the k -opt operator is a way to reconstruct a full turn by eliminating k edges that are not mutually exclusive and then joining the remaining route segments with k new edges. The bigger the neighborhood, the more k_i 's there are.

Construction of the first set: In many cases, a randomly generated combination is used as the starting point for the exploration process. As discussed in Section 3.3.1, it can occasionally be created via a greedy constructive heuristic. The original combination may have been the product of a different search process when local search is combined with another meta-heuristic like genetic algorithms or ant colony algorithms.

Selecting a neighboring combination: with each iteration of the local search, a neighboring combination is selected. The term "movement" describes this selection of a nearby combination. One can pick from a vast array of strategies. As an example, we can choose the best neighbor [65] at each iteration, which means the one that improves the objective function the greatest or the first neighbor identified to enhance it. Such "greedy" tactics (also referred to as "gradient rises") tend to become stuck in local optima quite fast, meaning they'll only work on combinations when none of the neighbors are any good. We can explore several meta-heuristics, among others, to break out of these local optima. You can start a local search several times with various starter combinations to repeat it. These first pairings can be completely random and unrelated. Next, we'll discuss multi-start local search, which is a kind of local search that uses several beginning places. Additionally, they can be starting combinations that are produced by disturbing a combination that was obtained during an earlier local search. Iterative local search is the next topic we cover [43]. Multiple local searches can be run simultaneously with various initial combinations. The current

combinations can then be redistributed on a regular basis by eliminating the poorest and duplicating the best, following the concept of "go with the winner." [63].

3.3 CONSTRUCTIVE META-HEURISTICS

Starting with an empty combination and adding combination components until a complete combination is produced is the way that constructive techniques create one or more combinations progressively. Since these methods employ a model—typically a stochastic one—to select the next combination component to add to the one being built at each iteration, they are referred to as "model-based" in [77]. Some methods exist for selecting which components to include in each iteration; the most well-known of these are the distribution estimation algorithms (3.3.2), the optimization meta-heuristic by ant colonies (3.3.3), and the random greedy strategies (3.3.1).

3.3.1 Random Greedy Algorithms

Greedy algorithms build combinations by iteratively selecting a combination component for which a specified heuristic is maximum, beginning with an empty combination. A greedy algorithm can build a combination in a flash without ever having to justify its decisions. The heuristic determines how good the built combination is. Case Study 3 Here is one way to define a greedy algorithm for the traveling salesman problem: The process begins with a randomly selected city and continues iteratively until all the cities have been visited. Each iteration moves on to the next unvisited city that is closest to the last visited city. A greedy randomized algorithm will build a lot of combinations and use a greedy technique to pick which components to add to those combinations, but it will also add some randomness to make those combinations more diverse. As an illustration, we have the option to select the subsequent component at random from either the best or the components that make up less than α percent of the best [15]. Alternatively, you might use the quality of each component to determine the probability of moving on to the next one [37].

3.3.2 Algorithms by Estimating Distributions

Random greedy algorithms like Estimation of Distribution Algorithms (EDA) [42] produce a new set of possibilities at each iteration using a technique similar to the one outlined in

3.3.1. On the other hand, EDAs use the most effective combinations from past iterations to create new combinations.

Making the first batch of inhabitants P: Typically, a uniform distribution is used to produce the initial population, and only the best combinations are retained. The creation of the probabilistic model: several probabilistic models, some more complex than others, can be taken into account. The most basic model, PBIL [5], ignores potential dependence linkages between components in favor of calculating the likelihood of each component's presence. Here, we determine the population frequency of each component and describe the likelihood of selection as a function of that frequency. Bayesian networks are utilized by other, more costly, models that are finer [54]. Then, conditional probability distributions are linked to the arcs of a graph, which indicate the dependence connections between components. The probabilistic model M is used to generate new combinations. The new combinations are built according to a random greedy approach, with the probability of selecting the components supplied by the model. Typically, while updating a population, only the most effective combinations are retained for the following iteration. These combinations might be either from the present population or those that were produced. It is also feasible to preserve a certain level of population variety in terms of the combinations retained.

3.3.3 Ant Colony Optimization

Distribution estimation strategies are quite similar to Ant Colony Optimization (ACO) [77]. Both methods build their combinations using an iterative learning process that uses a greedy probabilistic model. The model evolves based on the combinations that have been previously built. An important and novel aspect of ACO is its use of ant behavior as a model for the evolution of the probabilistic model. So, the amount of pheromone that represents the colony's prior experience with picking this component is directly proportionate to the chance of choosing it. When constructing a combination in a random, greedy fashion, the pheromone is utilized to influence the likelihood of choosing certain components. Decisions about the pheromone structure, or the data used to store traces of pheromones, are thus critical to the algorithm's success. Prior to an ACO algorithm being executed, a value of τ_0 is assigned to all pheromone traces. Case 6 Each pair of cities (i,j) in the traveling salesman issue has a trace associated with it. This trail stands for the colony's history of making continuous city visits. The combination component's pheromonal factor, $\tau_S(i)$, is defined differently

depending on the chosen pheromonal structure; the heuristic factor, $\eta_{S(i)}$, is defined differently depending on the problem; and the two parameters, α and β , allow one to modulate the relative influence of the two factors in the probability of transition. When α is equal to zero, the pheromonal factor has no influence on the selection of combination components, and the algorithm acts similarly to a greedy, completely random algorithm. In contrast, the decision probabilities are defined only by the pheromone traces if $\beta = 0$. The seventh example The quantity of pheromone τ_{kide} deposited between the candidate city i and the last city k added in defines the pheromonal factor $\tau_{S(i)}$ for the traveling salesman issue. The length of the route between the last city to the candidate city i has an inverse proportional relationship with the heuristic factor. Chemical messenger Latest news: After every ant has constructed a suit, the pheromone trails are refreshed. The traces are initially decreased by multiplying them by a factor $(1 - \rho)$, where ρ is an evaporation rate that falls within the interval $[0;1]$. The next step is to "reward" specific pairings with a pheromone deposit. The best way to decide which combos to award depends on your strategy. We can choose to award the best combination found since the execution began, the best combination of the cycle, or all of the combinations produced during the last cycle. The diversification and intensity of research are affected by these various tactics. As a rule of thumb, the quantity of pheromone is usually proportionate to the quality of the rewarded pair. The application and selected pheromonal structure determine which traces are linked with the combination to be awarded, and these traces are where it is put. Case 8 In the traveling salesman problem, for instance, we ensure that cities i and j were visited in a sequential manner during the building of the combination to be rewarded by depositing a pheromone on each trace τ_{ij} .

3.4 HYBRID META-HEURISTICS

It is possible to create new meta-heuristics by combining the various ones given in sections 3.2 and 3.3. In sections 3.4.1 and 3.4.2, two well-known instances of this type of hybridization are detailed.

3.4.1 Memetic Algorithms

Combining local search with a population strategy (evolutionary algorithms) is what memetic algorithms do [50],[51]. Incorporating elements of both the population approach's diversification potential and local research's intensification capacity is the goal of this

hybridization. As an extension of a genetic algorithm (like the one in Algorithm 1), a memetic algorithm incorporates a local search phase. Using a recombination operator to generate new combinations from two or more combinations in the population, the search space is sampled from a population of combinations, similar to a genetic algorithm. The mechanisms of selection and replacement decide which combinations are kept in the population and which ones are recombined. A local search method, instead of a mutation operator, is used in genetic algorithms; this process may be thought of as guided macro-mutation. Improving the quality of the resulting combinations is the goal of local search. When it comes to search intensification, local search is superior to the recombination operator because it takes use of the search pathways defined by the neighborhood operator under consideration. The memetic method relies on local search as much as it does on recombination.

3.4.2 Hybridization Between Disruptive Approaches and Constructive Approaches

The iterative construction of one or more combinations according to a random greedy principle (which can be directed by EDA or ACO) and the improvement of some of these combinations by a local search technique makes it easy to hybridize disruptive and constructive approaches. The acronym GRASP stands for "Greedy Randomized Adaptive Search Procedure," which describes this hybridization [58]. A key consideration is how to pick the perturbative approach's motions and which neighborhood operator to use. This necessitates striking a balance between the speed and quality of local search's combination enhancement operations. In most cases, we'll choose for a basic greedy local search, where we refine the built-in options until we reach a local maximum. Keeping this hybridization with local search in mind, the best EDA and ACO methods frequently incorporate it.

3.5 INTENSIFICATION VERSUS DIVERSIFICATION

For the different heuristic approaches presented in this chapter, a critical point in the development of the algorithm consists of finding a fair compromise between two dual tendencies:

- a. on the one hand, it is a question of intensifying the research effort towards the most “promising” areas of the space of combinations, that is to say, around the best combinations found.
- b. it is also a question of diversifying the research effort so as to be able to discover new areas containing (potentially) better combinations.

Depending on the strategy, several parameters of the algorithm can be tweaked to increase or decrease the research effort's intensity or diversity. We use greedy strategies for local search that favor the selection of the best neighbors, and elitist selection and replacement strategies for genetic algorithms that favor the reproduction of the best combinations, to intensify research in perturbative approaches by favoring the exploration of the best neighbors. The introduction of randomness is a common way to diversify perturbative approaches; for example, mutation is a key component of genetic algorithms, and local search allows for the search to select neighbors of lower quality with a low probability, both of which guarantee diversification. Research is amplified for constructive methods by prioritizing the selection of components from the best combinations built before at each step of building. The random element allows for the low chance selection of weaker components, which is how diversification is accomplished. In most cases, the faster an algorithm converges and finds better combinations, the more you may push it to explore options close to the best ones. But if we crank up the search intensity too much, the algorithm would probably "stagnate" at local optima, meaning it will focus on a little region surrounding a decent combination and fail to find any more. novel permutations. Keep in mind that the amount of time you have to answer the problem via calculations determines how well you can balance intensification and diversification. We should encourage intensification to converge fast, even if it implies convergent towards lower-quality combinations, if this time is minimal. The case at hand, and the research landscape topology in particular, dictate the optimal ratio of intensification to diversity. We say that a study landscape is of the "massif central" kind when there are few local optima in it and a strong correlation between the quality of a combination and its distance from the optimal combination; in this case, a highly concentrated search would provide superior findings. When there are many evenly distributed local optima in the search landscape, though, a highly diversified search—or even a totally random search—will provide the best results. Many methods have suggested changing the parameterization during resolution to fit the specific case. Our next topic is

reactive search [6]. Take, for instance, the IDwalk local search in [52], which dynamically changes the amount of neighbors taken into account with every move, and the reactive tabu method in [7], which dynamically changes the length of the tabu list.

3.6 ARTIFICIAL INTELLIGENCE APPLICATIONS

Many classic NP-hard problems have been successfully solved using meta-heuristics, and these algorithms have real-world applications in many different domains. This paper aims to explore their potential applications in addressing two major AI challenges: ensuring satisfiability of Boolean equations (sbf) and fulfilling constraints (soc).

3.6.1 Satisfiability of Boolean Formulas

There is a basic obstacle that needs to be conquered in the field of artificial intelligence development, and that is the hunt for sat. In order to develop a model for a Boolean formula, one of the most important steps is to assign a true or false Boolean value to each variable. This is a key step throughout the process. To ensure that the formula's valuation is accurate, this step is taken in order to assure its accuracy. In most cases, when it comes to the processing of Boolean equations, we operate on the assumption that the cnf form is necessary. Despite the fact that, in theory, the cnf form is not necessary for the application of a metaheuristic, this is the situation that has arisen. It is vital to keep in mind that there is no objective function in SAT that can be improved; hence, the issue is not an optimization function. This is an essential fact to keep in mind. Find the value that maximizes the number of clauses that are satisfied is the objective of the more general problem that is referred to as maxsat. This is the question that has to be answered. The reason behind the challenge is to do this. It is usual practice to design meta-heuristics in order to handle optimization issues; nonetheless, these meta-heuristics also take into consideration the bigger general problem.. Both of these outcomes are desirable. This is a possibility for both of these events. A set of Boolean values is assigned to each variable in a SAT instance. These values are allocated to the variable respectively. The search space that each variable in the instance has to be in order for it to work correctly is specified by this collection, which is an essential component. Because of this, the size of the space is 2^n bytes in the scenario where there are no variables present. This is because of the fact that this occurred. One of the factors that is taken into consideration throughout the process of assessing the objective function that has to be

optimized is the number of clauses that are satisfied. It is one of the functions that is contained in this function that is responsible for establishing a full order on the combinations that make up the search space. At each and every clause that is not satisfied, there is a penalty weight of one that is linked with it, and a solution is a combination that has an evaluation of zero. The dual objective function is a function that assesses the number of clauses that are not met and corresponds to a penalty function that has to be decreased. There is also the possibility of taking into consideration the dual objective function, which is a function that does this evaluation. Either by developing a dynamic punishment system or by adding an addition that takes into consideration data other than the number of clauses that have not been met, it is possible to increase the functioning of this function. Both of these options are viable options. The problem of finding a solution has been the subject of a significant amount of attention and effort over the course of the past twenty years, with the goal of finding a solution that is both practical and effective. A continual stimulus for the creation, testing, and comparison of algorithms that are involved in the resolution of SAT is provided by the various challenges that are centered on SAT that are faced by the scientific community. These issues serve as a source of inspiration for the development of these algorithms. To advance artificial intelligence, we must first overcome a fundamental challenge: finding SAT. One of the most crucial phases in building a model for a Boolean formula is to give each variable a true or false Boolean value. This is an important part of the procedure. In order to guarantee the correctness of the formula's value, this step is conducted. The CNF form is assumed to be required for processing Boolean equations in the majority of circumstances. This is the current state of affairs, even if the CNF form is not required for applying a metaheuristic in principle. Remember that SAT does not have an improvement-able objective function; so, the problem is not with optimisation. Do not forget this crucial piece of information. MaxSAT is a more generic issue whose purpose is to find the value that maximises the number of clauses that are satisfied. We need an answer to this query. This is why we're posing the task. Typically, meta-heuristics are designed to address optimisation problems; however, these meta-heuristics also account for the more general problem. I would like both of these results. With both of these occurrences, this might happen. In a SAT instance, every variable is given a set of Boolean values. The variable is assigned these values in that order. An integral part of the instance, this collection specifies the search space that each variable must be in for it to function properly. If no variables are present, the space will be 2^n bytes in size

due to this. The reason for this is because this has really taken place. The number of fulfilled clauses is one of the parameters considered during the process of evaluating the objective function that needs optimisation. The responsibility for creating a complete hierarchy on the search space's component combinations falls on one of the functions included in this function. A penalty weight of one is associated with each unsatisfied clause, and a combination with an evaluation of zero is considered a solution. A penalty function that has to be minimised corresponds to the dual objective function, which evaluates the amount of unmet clauses. Additionally, the dual objective function, a function that does this evaluation, might be considered. One way to make this function work better is to create a system of dynamic punishments. Another is to add an addition that considers data other than the number of unmet clauses.

Attempts to discover a workable and efficient solution to the sitting problem have received a great deal of focus and energy during the last two decades. The many sat-related problems that scientists encounter give an ongoing incentive to develop, evaluate, and compare algorithms that contribute to sat resolution. Insights for these algorithms were derived from these problems.

a. Stochastic local search

Nearly all rls algorithms for sat make use of the same neighborhood function, which states that any two combinations that have a Hamming distance of exactly 1 are considered to be neighbors. This function is used to determine whether or not two combinations are neighbors. As a consequence of this, the selection of the variable that is impacted by the change is what defines the transition from one combination to another. In the case where a formula does not contain any variables, the neighborhood size is equal to zero. It is feasible to construct a neighborhood of variable size that is more constrained if this selection is restricted to the set of variables that are present in at least one sentence that has been invented. This is especially true when it comes to finding a happy medium between the two. One method for classifying these algorithms is to investigate the manner in which they make use of the search history, while another method is to take into consideration the manner in which they do so [4]. Both of these methods are incorporated into the classification process. An advantage is being taken advantage of by utilizing the structure of the instance. The formula has clauses that include information that indicates the structure of the instance that

needs to be resolved. One may see at least three distinct levels of exploitation inside this system. These levels can be studied separately. By letting the objective function take the lead in the search, we are able to exclude all of the sentences that the current combination does not support, with the exception of a specific number of sentences. For example, the well-known gsat algorithm [65] usually chooses an adjacent configuration at random from among those that minimize the quantity of incorrect phrases at each iteration. This is done in order to maximize efficiency. It is simple to become mired in a condition of local optimality when one employs a kind of aggressive exploitation that is often known as greedy descent. A basic diversification strategy is utilized by gsat in order to get around this problem. This method involves beginning the search with a fresh configuration after a certain number of repetitions from the one that was previously used. Other examples of approaches that apply a similar methodology to achieve the same objective of minimizing the amount of erroneous assertions include simulated annealing [70], several versions of gsat [22], and csat and tsat [22]. All of these methods are examples of ways that employ a similar methodology. It is feasible that competition will also be the motivating factor behind the search in order to expressly account for counterfeit provisions. This is something that has to be considered. In situations when there is no longer a combination in the neighborhood that contributes to an increase in the objective value, this strategy becomes extremely effective. There is no longer any availability for these combinations. It is possible that we will be able to restrict the scope of the investigation with the aid of new data gathered, such as from an examination of sentences that have been produced. During each cycle, it selects an untruth from a collection of options in a random fashion. This algorithm serves as a prototype example of this type of methodology. Walksat [47] is an example of an algorithm that falls into this category. In the subsequent phase, we will use a heuristic to choose one of the variables that are contained within this sentence, and then we will change the value of that variable in order to achieve the new current combination. The gwsat algorithm, which is also known as gsat with random walk [63], employs the gsat descent technique with a probability $1 - p$ at each iteration. This approach is associated with the gsat algorithm. Additionally, it alters a variable that is connected with a false phrase that has a probability of p . This sort of phrase is known as noise. In addition, it uses the false clauses as a guide for the random walk that it does. Deductively exploiting limits enables the last step, which is the application of deduction rules to make modifications to the current combination. This phase is the culmination of the

process. One of these methods is called the unit walk [31], and it makes use of unit resolution in order to determine the values of particular current combination variables by utilizing the values of other variables that are determined via local search. One example of a method that may be utilized to determine the values of particular variables is the approach that is being discussed here. Non-cnf adapt innovation [55] is a unique strategy that use a distinct mode of deductive reasoning to analyze the formula for processing in order to uncover linkages between variables that are reliant on one another. This is done in order to find potential connections between the variables. Through the utilization of websites that have been visited in the past, data mining may be accomplished. In order to classify rls methods to sat, one way that might be applied is to take into consideration how the research's history of activities is utilized and, if appropriate, what impacts those actions had. It is possible to divide the exploitation that occurred throughout this period into three separate degrees. In Markovian algorithms, which are sometimes referred to as memoryless algorithms, the only element that decides which subsequent combination is picked is the combination that is now being taken into consideration. Other important instances of memoryless rls algorithms are the gsat, grsat, walksat/skc, and the more general simulated annealing technique. These algorithms are all examples of memoryless rls algorithms. Even while memory algorithms are successful in reducing the amount of processing time required for each iteration, they have largely replaced them in practice over the course of the previous decade. This is despite the fact that memory algorithms have effectively replaced them. When selecting each new combination, the algorithms that are used for short-term memory take into account a history of all or part of the adjustments that have been made to the variable values. This is done in order to ensure that the choices they make are accurate. Some examples of Sat solvers that have been influenced by the tabu approach are tsat [46] and walksat/tabu [47]. There have been a number of Sat solvers that have also been inspired by the tabu technique. With the implementation of seniority as an additional selection criterion, the algorithms of walksat, novelty and rnoveity, and g2wsat have been updated [34]. All of this is done with the intention of enhancing the precision of the algorithms. In instances when there are a large number of candidate variables, this criteria is often implemented with the intention of giving preference to the variable that has been modified the most dramatically throughout the course of the course of the circumstance. In the process of selecting each new combination, algorithms that are part of long-term memory take into consideration the impacts of earlier decisions,

particularly with regard to clauses that have been met and those that have been falsified. By utilizing a learning mechanism that benefits from making mistakes, the objective of this attempt is to hasten the process of doing research that is currently being conducted. The technique of assigning weights to the faked clauses is done with the objective of varying the study by pushing it to examine different paths, which in turn makes use of their separate histories. This is done in order to achieve the goal of diversifying the research. As an initial example, let us take into mind weighted-gsat [62], which raises the weights of the wrong sentences after each variable change. This is done for the aim of presenting an initial example. What determines the score for the search heuristic is nothing more than the simple addition of the weights of all of the incorrect sentences. The technique has a propensity to favor the alteration of variables that belong to often falsified clauses while the weights of the other clauses continue to change. This makes the approach more likely to be successful. At least until the growth of the weights of this other clause has another influence on the search, this is the situation that will continue to be the case. Additional examples include the discrete lagrangian method, weighted variables, paws, esg, scaling, and probabilistic smoothing among others. The latter is another illustration of this. Some examples of these algorithms are the Exponentiated Subgradient Algorithm, the wv algorithm, and the ddwf algorithm. In addition, there is the Pure Additive Weighting Scheme [57].

b. Population-based algorithms

There have been a variety of genetic algorithms [30],[75],[10] that have been utilized in the process of solving SAT problems. These GAs perform the same function as local search algorithms, which is to say that they use the set of variables that consists of 0s and 1s as their representation of the search space. On the other hand, the first GAs deal with broad logical formulae right away; hence, the cnf form is not the only one that any of them address. The results that these algorithms provide when applied to cnf benchmarks, on the other hand, are frequently of a low quality. The initial cnf equations for the processing of GA are shown below for your convenience [16]. The single most essential selling point of this algorithm is the cutting-edge crossover approach that it uses. This technique makes advantage of the semantic similarities that exist between the two parents, and it represents the most significant selling point of this algorithm. When there are two parents, p1 and p2, clauses that are met by one parent but not by the other parent are analyzed. This is the situation when there are

two parents. The kid receives as true values values that are true in parent₁ but false in parent₂; these values are passed on to the child. Specifically, this is applicable to any variables that are included within a clause. Within the context of the second DIMACS implementation challenge, the discoveries that were accomplished by the memetic algorithm that combined this crossover with a tabu search were really exciting. The genetic algorithm that is commonly referred to as Gasat [17] is yet another breakthrough that is rather new. In the same way that it is essential to the method outlined in [16], the building of a semantic intersection is of the utmost significance to gasat technology. With the goal of limiting the "mistakes" of both parents and increasing the best qualities that each parent would have brought to the table, a new breed of hybrids has emerged in the hopes of achieving this goal. It is feasible that we might make it possible for a kid whose parents are of high quality to have a clause that is untrue in and of itself if we flip the variable of versus. A argument that supports this crossover is that it is preferable to enforce the truth of such a clause since it is impossible to satisfy in any other manner. This crossover is supported by this rationale. When it comes to this crossover, this is the intuitive reasoning that supports it. An strategy that is quite similar is utilized in order to accomplish the formation of recombination mechanisms. These mechanisms make advantage of the semantics associated with a word that is true at the same time by both of the parents. Gasat is able to compete well in a number of benchmark areas as a consequence of this sort of crossover and an approach that involves forbidden search. In order to produce a kid that contains an equiprobable combination of the values that are owned by both of the parents, the genetic algorithm known as FlipGA [59] makes use of a conventional (uniform) crossover. This is done in order to provide the child with certain characteristics. After that, a local search strategy is utilized in order to carry out the process of instructing this child. There are further GAs that are displayed in [13],[25],[60], however these GAs are regarded to be local search algorithms due to the fact that the population is restricted to a certain combination. Their efforts are primarily concentrated on developing methods that, by means of search-process dynamic adjustment, can raise the fundamental evaluation function, which is the number of sentences that have been falsified. There is a comparison of these several SAT-specific genetic algorithms that was carried out through experiments and can be found in [26]. The fact that they have only seldom been faced head-on by RLS algorithms that are believed to be cutting edge is the reason why their worth has not yet been shown. When it comes to these algorithms that are

based on populations, we are aware that local search is often a key component that plays a big part in boosting the intensification capability. This is something that we realize. The aim of crossing can change depending on whether it is purely random [59], in which case it just helps to expand the search, or based on the semantics of the problem [16],[17], in which case it allows for both diversity and intensity of research. In either instance, the purpose of crossing can be different. In order to solve SAT issues, a number of evolutionary algorithms have been employed, ranging from [30],[75] to [10]. Similar to how local search algorithms employ a collection of variables consisting of 0s and 1s to describe the search space, GAs like this one do the same thing. However, none of the early GAs handle the cnf form as they immediately begin dealing with general logical formulae. In contrast, when similar algorithms are used on cnf benchmarks, the outcomes are often not very good. You may find the first cnf equations for processing GA below [16]. The innovative crossover method it employs is the algorithm's most compelling feature. The most appealing aspect of this algorithm is this method, which takes use of the semantic similarities shared by both parents. Clauses that are satisfied by one parent but not the other are examined when there are two parents, p1 and p2. This is the case when both parents are involved. Some values are true in parentp1 but false in parentp2, and the child inherits these values as true. This is especially true for variables that are part of a clause. The two memetic algorithms' joint use of a tabu search and this crossover led to some fascinating findings in the second DIMACS implementation competition. Another relatively recent development is the genetic algorithm sometimes known as Gasat [17]. The construction of a semantic intersection is extremely important to gasat technology, just as it is to the approach described in [16]. A new hybrid breed has arisen with the intention of minimising the "mistakes" of both parents while maximising the greatest traits that each would have contributed. If we turn the vs variable upside down, it's possible that a child with good parents might have a clause that isn't true at all. This crossing is supported by the argument that since it cannot be satisfied in any other way, it is desirable to enforce the truth of such a phrase. This reasoning lends credence to this crossover. This is the line of thinking that lends credence to this intersection. In order to achieve the creation of recombination mechanisms, a technique that is quite similar is used. When both parents agree on the meaning of a term, these processes take advantage of that fact. This type of crossover, together with a technique that makes use of banned search, allows Gasat to perform effectively in several benchmark areas. The genetic algorithm

called FlipGA [59] uses a standard (uniform) crossover to create a child with an equiprobable mix of the values possessed by both parents. We do this so the kid can grow up with a specific set of traits. The next step in imparting knowledge to this kid is to employ a local search approach. Additional GAs may be found in [13], [25], [60], although these GAs are considered local search algorithms because the population is limited to a certain combination. Their efforts are primarily concentrated on developing methods that, by means of search-process dynamic adjustment, can raise the fundamental evaluation function, which is the number of sentences that have been falsified. You may find an experimental comparison of these various SAT-specific genetic algorithms in [26]. The fact that they have only seldom been faced head-on by RLS algorithms that are believed to be cutting edge is the reason why their worth has not yet been shown. When it comes to these algorithms that are based on populations, we are aware that local search is often a key component that plays a big part in boosting the intensification capability. We are cognisant of this fact. The aim of crossing can change depending on whether it is purely random [59], in which case it just helps to expand the search, or based on the semantics of the problem [16],[17], in which case it allows for both diversity and intensity of research. In either instance, the purpose of crossing can be different

3.6.2 Constraint Satisfaction Problems

Each variable in a Constraint Satisfaction issue (CSP) takes a value in a domain, and the issue is characterized as a series of constraints on those variables. Assigning values to variables in a way that satisfies all constraints is the key to solving a CSP. Regarding sat, in general, we think about the MaxCSP optimization problem, the goal of which is to discover the full assignment (giving a value to each variable) that maximizes the number of constraints that are fulfilled. The number of satisfied constraints defines the objective function to be maximized, whereas the set of full assignments defines the search space. In a Constraint Satisfaction problem (CSP), the variables are defined by a set of constraints on a domain where each variable takes a set value. The crux of solving a CSP is finding values for variables that adhere to all constraints. We often consider the MaxCSP optimisation issue while thinking about sat; this problem seeks to find the entire assignment (where each variable is given a value) that maximises the number of constraints that are satisfied.

Maximising the objective function is defined by the number of satisfied constraints, while the search space is defined by the set of complete assignments.

a. Genetic algorithms

The most elementary kind of genetic algorithms for CSP is the process of simple crossover, which is used to recombine populations of full assignments using genetic algorithms. In the context of this discussion, mutations are characterized by changes in the values of relevant variables. An experimental comparison may be performed to assess whether or not any binary CSP can be resolved by utilizing the eleven evolutionary algorithms that were provided in [9]. This can be done by comparing the results of the comparison experiments. They demonstrated that the top three algorithms perform similarly or even better than the other eight algorithms by comparing the performance of the top three algorithms, which are the Heuristics GA version 3, Stepwise Adaptation of Weights, and Glass-Box algorithms, to the performance of the other eight algorithms used in the study. However, it is self-evident that even the most powerful genetic algorithms are unable to compete with heuristic approaches such as local search or ant colony optimization [72] or with accurate methods that are based on tree searches. This is the case even if genetic algorithms are becoming increasingly efficient. There have been a number of different evolutionary algorithms that have been suggested as possible solutions for particular issues that include the fulfillment of limitations. In order to get better outcomes, these algorithms make use of information about the limitations of the job in order to build mutation and crossover operators that are more suited to the conditions. This allows them to provide better results. To provide a more precise example, we may make reference to [76], which is applicable to a problem involving the scheduling of vehicles and generates outcomes that are competitive. To recombine populations of complete assignments using genetic algorithms, the simplest sort of CSP genetic algorithm is the procedure of simple crossover. Mutations, as we define them here, are defined by shifts in the values of important variables. It is possible to compare the eleven evolutionary methods given in [79] experimentally to see if any binary CSP can be resolved. The outcomes of the comparative trials can be compared to accomplish this. By comparing the results of the top three algorithms the Glass-Box, Heuristics GA version 3, and Stepwise Adaptation of Weights algorithms to those of the other eight algorithms utilised in the study, they proved that these three algorithms perform adequately, if not better. Local search, ant

colony optimisation, and other heuristic approaches, as well as accurate methods based on tree searches, clearly outperform even the most robust genetic algorithms [82]. This is true regardless of the rate of improvement in the efficiency of genetic algorithms. For certain problems, such as the satisfaction of constraints, a variety of evolutionary algorithms have been proposed as potential solutions. Algorithms like this take into account the constraints of the task at hand to construct mutation and crossover operators that perform better under those specific circumstances. They are able to deliver superior outcomes because of this. It is possible to cite [86] for a more specific illustration; it solves an issue with vehicle scheduling and produces competitive results.

b. Local search

Numerous articles have been written on the subject of employing local search algorithms for the goal of resolving CSP, and in the vast majority of cases, it has been shown to provide outstanding results. There is a great deal of diversity in local search algorithms with regard to CSP resolution, and the majority of this variation is mostly driven by the neighborhood and the decision mechanism. An area that is close by. Within the context of a total assignment, the term "neighborhood of a total assignment" refers to the collection of assignments that may be created by altering the value of a variable. In general, the combinations that are being explored are complete assignments, and a move is a change to the value of a variable. This is the reason why this is the case. For example, when two variables are subject to a permutation constraint, which stipulates that each conceivable value for a given sequence of variables must be one permutation of that sequence, we may conceive of a neighborhood in which the values of the two variables are switched. This neighborhood is a neighborhood in which the values of the two variables in question are swapped. One example of a neighborhood is the one shown above, which is a neighborhood in which the values of the two variables are switched around. A number of studies take into account the neighborhoods that are present between partial tasks in addition to the assignments that are given in a more broad sense. One strategy that serves as an excellent illustration of this is the decision repair technique that is described in [39]. In this approach, a local search is performed on the space of partial assignments, and filtering processes that are discussed in chapter?? are combined. It is possible that the neighborhood of a partial assignment is either the set of assignments that result in the removal of a variable/value pair

from the assignment or the set of assignments that result in the addition of a variable/value pair to the assignment. This is determined by whether or not the filtering identifies an inconsistency. This is a possibility for both of these events. There are a few distinct methods for choosing an alternative. In order to determine what the next action on the list ought to be, a number of alternative ways have been suggested as potential solutions. A random selection of variables that are involved in at least one constraint violation is utilized in the min-conflict technique [48] to select a value for a variable that minimizes the number of violations. This is done in order to get a value that is optimal for the variable. A value for the variable is then determined by using this value as a starting point. Despite the fact that it is well-known for its ability to solve the queen issue by allowing a million queens to do so, this greedy approach usually finds itself trapped in the solution that is the local optimal solution. When paired with the random walk approach, the min-conflict strategy is a clear and time-honored methodology that enables it to break out of these local minima. This is because the min-conflict strategy is so straightforward. It is not [73]. An further method for selecting the variable/value combination is studied in [27], which is a collection of studies on the topic. TabuCSP is a local search approach that you should try out if you are attempting to solve a binary CSP [18]. TabuCSP is a method that is associated with the local search method. With a starting assignment serving as the starting point, the goal is to select the non-taboo movement that maximizes the number of requirements that are satisfied at each iteration. This will be accomplished by beginning with a starting assignment. In order for a movement to be considered taboo, it is essential that the value of the conflicting variable has not been changed since the pair was selected during the klast movements. This is a prerequisite for the movement to be considered offensive. The letter k is used to indicate the length of the list of forbidden things. With the addition of an aspiration criteria, it is feasible to choose a movement that is prohibited if it results in an assignment that satisfies a higher number of requirements than the optimal assignment that was first chosen. In order to solve CSP issues using the COMET language, it is necessary to develop local search algorithms as rapidly as feasible [30]. The notion of incremental variables, which makes it feasible to perform incremental evaluations of communities, is a key component of his plan for tackling this issue. This strategy is an integral component of his overall strategy. This article presents a variety of instances of alternative generic CSP resolution strategies that are based on local search approaches. These techniques are described in [32] Constructs that are coveted and

make use of algorithms. It is possible to create the entire assignment of a CSP in line with the greedy idea, which is demonstrated in the following: In the beginning, we start with an empty assignment, and then we assign values to each of the variables one at a time. After the initial assignment has been finished, we select an unassigned variable and its value at each iteration. The specific techniques that are explained in chapter ?? typically make use of order heuristics when it comes to picking a variable and its value at each stage. This is because order heuristics are deterministic in nature. The highly famous DSATUR approach [8] provides a coloring strategy for graphs, and it is this method that puts this idea into effect. In order to generate a combination in a manner that is greedy, this method chooses the uncolored vertex that has the most neighbors that are colored with a variety of hues (the neighbor that is the most saturated) with each iteration. This is done in order to construct a combination. Choosing the peak with the highest degree is the only method to break a tie in a competition. Following that, the color that has the lowest possible value is applied to the vertex that was selected. The use of local search algorithms to resolve CSP has been the focus of several publications, and in most circumstances, it has been demonstrated to produce excellent results. The neighbourhood and the decision mechanism are the primary drivers of the wide variety of local search algorithms when it comes to CSP resolution. One that is nearby. The set of assignments that can be generated by changing the value of a variable is called the "neighbourhood of a total assignment" when discussing total assignments. In most cases, the combinations under consideration are full assignments, and a move is defined as a modification to a variable's value. For this reason, this remains true. If, for instance, a permutation constraint states that every possible value for a specific sequence of variables must be a permutation of that sequence, and two variables are subject to this constraint, we might imagine a neighbourhood where the values of the two variables are swapped. If the values of the two variables in issue are switched, then we are in this neighbourhood. The above-depicted neighbourhood is an example of a neighbourhood in which the two variables' values are inverted. Numerous studies consider not just the more generalised assignments, but also the neighbourhoods that exist between partial jobs. An approach that exemplifies this well is the decision repair method outlined in [39]. This method combines the filtering procedures covered in chapter with a local search on the space of partial assignments. The set of assignments that lead to the addition of a variable/value pair to an assignment or the set of assignments that lead to its removal might be considered

the neighbourhood of a partial assignment. The presence or absence of an inconsistency detected by the filtering will dictate this. With both of these occurrences, this might happen. Picking an alternate can be done in a number of different ways. Several different approaches have been proposed as possible answers to the problem of deciding what the next item on the list should be. To find the value of a variable that minimises the number of violations, the min-conflict approach [48] uses a random selection of variables that are involved in at least one constraint violation. In doing so, we want to find the best possible value for the variable. The next step is to use this number as a baseline to establish a value for the variable. This greedy strategy often gets stuck in the answer that is the local optimal solution, even though it is famous for solving the queen issue by letting a million queens do it. It is possible to escape these local minima by combining the random walk technique with the min-conflict strategy, a transparent and time-honoured methodology. The simplicity of the min-conflict technique is the reason behind this. In [87], a compilation of research on the subject, an additional approach to choosing the variable/value combination is examined. If you are trying to solve a binary CSP, you should try using TabuCSP, a local search technique [88]. When you use the local search strategy, you'll come across TabuCSP. The objective is to maximise the number of requirements met at each iteration by selecting the non-taboo movement, using a beginning assignment as the starting point. A starting assignment is the first step in achieving this goal. It is crucial that the value of the conflicting variable remains unchanged since the pair was chosen during the last movements for a movement to be deemed forbidden. In order for the movement to be deemed offensive, this must be present. The length of the list of disallowed goods is denoted by the letter k . Adding an ambition criterion makes it possible to select a forbidden movement if it yields an assignment that fulfils more requirements than the initially selected optimum assignment. In order to solve CSP issues using the COMET language, it is necessary to develop local search algorithms as rapidly as feasible [90]. The notion of incremental variables, which makes it feasible to perform incremental evaluations of communities, is a key component of his plan for tackling this issue. As a whole, this tactic is crucial to his plan. This article showcases many examples of local search-based alternative generic CSP resolution algorithms. In [82], these methods are detailed. Constructs that are coveted and make use of algorithms It is possible to create the entire assignment of a CSP in line with the greedy idea, which is demonstrated in the following: In the beginning, we start with an empty assignment, and then we assign

values to each of the variables one at a time. After the initial assignment has been finished, we select an unassigned variable and its value at each iteration. The specific techniques that are explained in chapter typically make use of order heuristics when it comes to picking a variable and its value at each stage. This is because order heuristics are deterministic in nature. The highly famous DSATUR approach [84] provides a coloring strategy for graphs, and it is this method that puts this idea into effect. In order to generate a combination in a manner that is greedy, this method chooses the uncolored vertex that has the most neighbors that are colored with a variety of hues (the neighbor that is the most saturated) with each iteration. This is done in order to construct a combination.

c. Ant colony optimization

There were two independent studies [12] that made use of the ACO meta-heuristic in relation to CSP. The idea is to be successful in completing projects by applying the notion of random greed. During each iteration, we begin with an empty assignment and assign a value to a variable that has not yet been assigned. This allows us to accomplish the desired result. The procedure will continue until each of the variables has been set to the appropriate value. A heuristic for picking values is one of the most significant contributions that ACO brings to the field. The chance of selecting this heuristic is dependent on the combination of these two factors. Because it is a general structure, the pheromonal structure may be applied to fix any CSP that is already in existence. When it comes to collecting the colony's history of manipulating x_i , the technique's natural approach is to give a value v_i to x_i and to attach a pheromonal trace with each variable x_i . This is the technique's natural method. In addition to that, the technique gives the variable x_i the value v_i . Different pheromonal designs have been offered as a viable solution in order to handle certain CSPs, such as multidimensional backpacks [2] or automobile scheduling constraints [67]. These designs have been proposed in order to address problems. Ant colony optimization has been included into the IBM/Ilog Solver and CP Optimizer programs, both of which were developed with the express purpose of addressing issues that are connected with restricted optimization and CSP [40] An ACO algorithm that is integrated in the language is responsible for automatically resolving issues. As a result of this integration, declarative problem description in high-level languages is now feasible. Additionally, the language is able to automatically resolve problems.

3.7 ARTIFICIAL SWARM ALGORITHM

3.7.1 The Basic Principle of Artificial Fish Algorithm

Fish have highly developed visual systems that allow them to comprehend a great deal of area in a short amount of time. To mimic the fundamental swimming motion of the artificial fish shown in Fig. 3.1, AFSA makes use of the model.

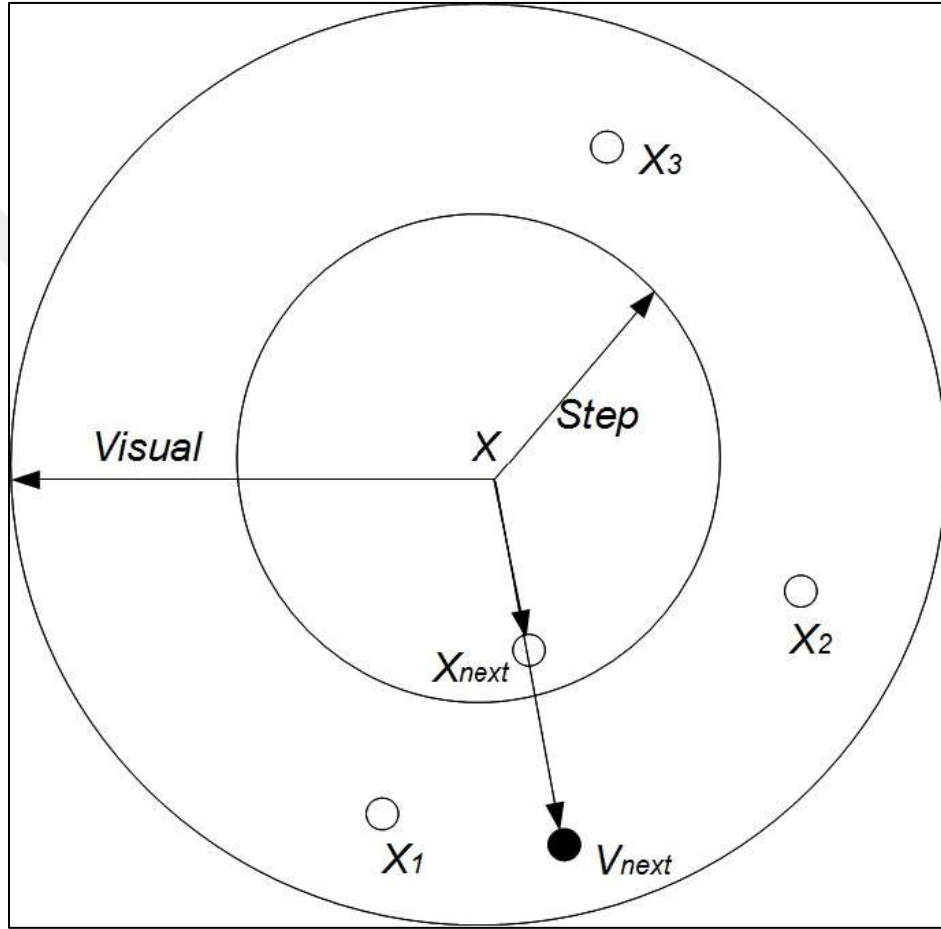


Figure 3.1: Basic Model of AFSA.

$X_i (i^{1,2,3})$ represents the other fake fish, and its visual length is Visual. In Figure 3.2, we see that for each artificial fish $X (x^1, x^2, x^n)$, the greatest distance that may be used as a step is X_i . When the visual field of the random search of $V_{next} (V_{next} (x^1_v, x^2_v, x^n_v))$ is in its phase, the AFSA proceeds to the location of the record for X_{next} if the concentrations of things in V_{next} (function value) are greater than the X position of the food concentration.

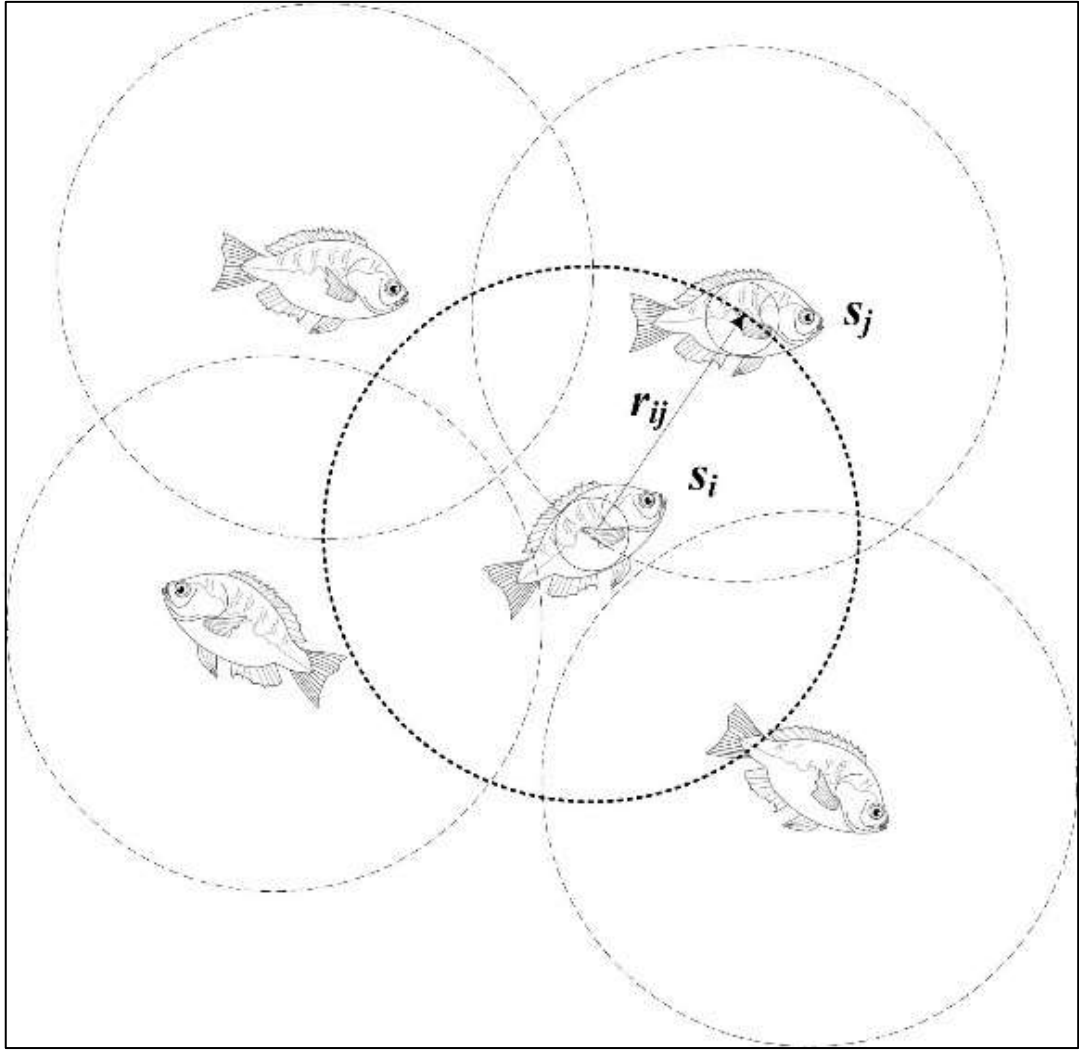


Figure 3.2: Search Space of the Fish Swarm Algorithm.

Foraging, bunching, collision, and unpredictable behavior are the four fundamental components of an artificial fish algorithm. To create the artificial fish in closed optimization of function domain optimization, it is necessary to combine all four behaviors and work together. Figure 3.3 shows an optimization flow chart that makes use of an artificial fish method.

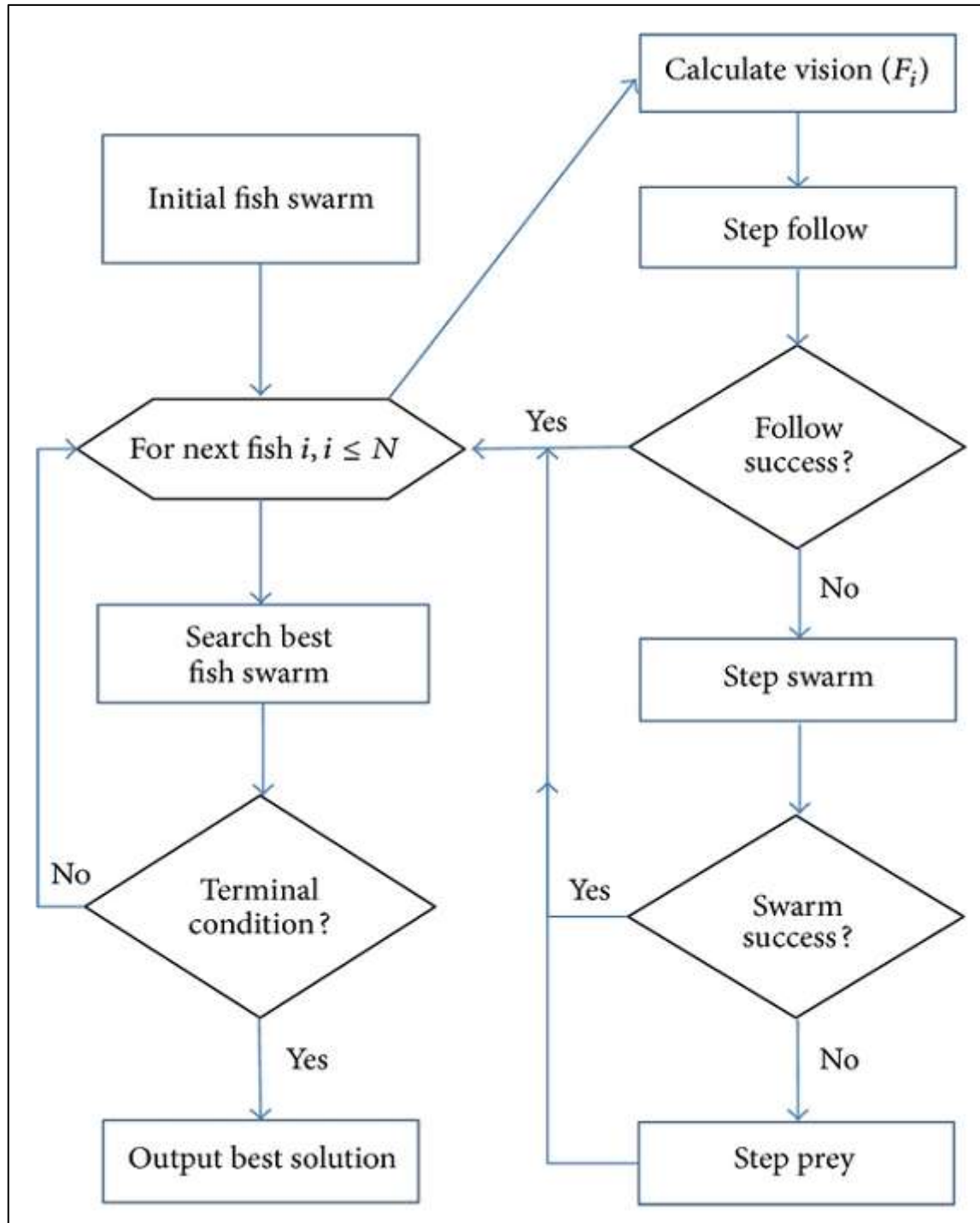


Figure 3.3: Diagram of the Fish Swarm Algorithm.

3.7.2 Individual Movement Operator

The swarm's individual and collective motions are aided by this operator. meaning that s^i is a weighted parameter, rand is a random generator, and x_i represents the final position of fish i during the current generation.

a. Food Operator

A metaphor for the effectiveness of food seeking is the weight of each fish. A larger fish is more likely to be located in an intriguing area of design space if it has a high mass. If a fish's goal function improves in this generation, then the amount of food it eats will increase, as stated by [33] Where W_i^t is the weight of fish i at generation t , and Δf_i is the difference between the current position of fish i and the new position of fish i , the objective function is being evaluated. It is imperative to emphasize that when fish are located at the same location, the value of Δf_i is equal to zero.

b. Instinctive collective movement operator

This operator is vital in deciding the individual movement of fishes when the value of Δf_i is non-zero. This implies that the school's natural collective movement can only be influenced by the fishes whose individual performance improved their fitness, as this is the only way to influence the school's motion. The resulting direction (I^t) is calculated by adding together all the directions that the i^{th} fish has followed; this direction then determines the i^{th} fish's new position: Here, the path that a fish took to find the most food would have the most impact on the school as a whole via the use of this operator. As a result, the swarm's intuitive collective movement operator will usually follow the school of fish that has been swimming in the direction where it has found the most food.

c. Non-Instinctive collective movement operator

The fish's weight is a reliable indicator of how well the hunt for food is going, as mentioned before. Consequently, a growing swarm weight indicates a performant search procedure. Accordingly, in order to access further areas, the swarm's "radius" must shrink. Otherwise, expanding the radius to explore additional locations should be possible if the swarm weight stays the same. The idea of the centroid is applied to the swarm contraction.

3.8 HAWKFISH OPTIMIZATION ALGORITHM

One dominant male hawkfish will mate with many females to form a harem. In a hawkfish harem, the biggest female would undergo a sexual transformation and become the dominant partner for half of the females. On the flip side, if that young male hawkfish started to lose females to other harems and was confronted by a bigger male, it would revert to mating as a female instead of expending energy futilely trying to win the conflict [80].

3.8.1 Gender Benders

The argument put out by Scott Heppell, a fish biologist at Oregon State University in Corvallis, is that our frame of reference leads us to assume that gender is predetermined in some fashion. Heppell is a student at Oregon State University, which is a national university. "Some species aren't as adaptable as these animals." Despite the fact that hermaphroditic animals can be born with a preference for either the male or female reproductive organs, or even both, the capuchin falco has adapted to the fact that it can choose its sex by studying its behavior [81]. This is the case despite the fact that the capuchin falco has the ability to chose its sex. According to Heppell's argument, we take into account the impact that genes have on behavior is something that we take into consideration. On the other hand, the circumstance is quite different in this case. As an example, in the case that a male participant is removed from the competition, a female participant can make the transition to the opposite gender and perform the role of the male participating in the competition. A chain reaction of hormones enables the female to start manufacturing testosterone rather than oestrogen when the dominant male is no longer around. This occurs when the dominant male is absent. This, in turn, has an influence on the germ cells, which are the cells that will eventually mature into reproductive cells that are responsible for reproduction. According to Heppell, the most sub-dominant member of the opposite sex would experience behavioral alterations in a matter of minutes and structural anomalies in the tissues within a matter of days. These changes would occur in the very same time frame [82]. In certain species, they will be physiologically ready to spawn as the other sex within a week or two. This will happen in some species. This was the first time that a wild fish was observed reproducing after reverting to its original gender, despite the fact that ecologists were aware that certain fish may have the ability to alter their sex from one gender to another.

3.8.2 Modelling Genders In Hawkfish Algorithm

The two parameters that are always present in the artificial fish algorithm that is regarded to be standard are the step and the visual parameters. Increasing Visual, on the other hand, causes late-stage optimization to lose accuracy and may even result in search results that only include local optimal values. In the early stages of the method, increasing Step and Visual both speed up convergence. However, although increasing Visual causes late-stage optimization to lose precision, increasing Step speeds up convergence. In order to control the dynamic changes that take place throughout iteration, the enhanced artificial fish algorithm has been updated to include visual parameters. Further, a regulating component known as let Step is employed in order to assure the accuracy and speed of the algorithm's convergence. This is done in order to ensure that the algorithm is convergent. Utilizing bigger parameter values at the beginning of the process guarantees a quick convergence approach, while lowering them later on assures that the search will be correct. Both of these approaches are possible because of the process. Because it does not take into account "real-time" information about the fish's foraging behavior, clustering, and fundamental principles—more precisely, the global ideal value—the typical AI algorithm is unable to optimize its exploration impacts in a timely manner. This is because it does not take into account the information. This is the reason why the artificial intelligence is unable to maximize the results of its investigation. We are able to triumph over these challenges by harnessing the power of two distinct fitness functions, one for each different gender:

- a. KKT conditions as a fitness function to look for the best solution (Male hawkfish).
- b. KKT conditions (inversed) to look for the worst solution (Female Hawkfish).

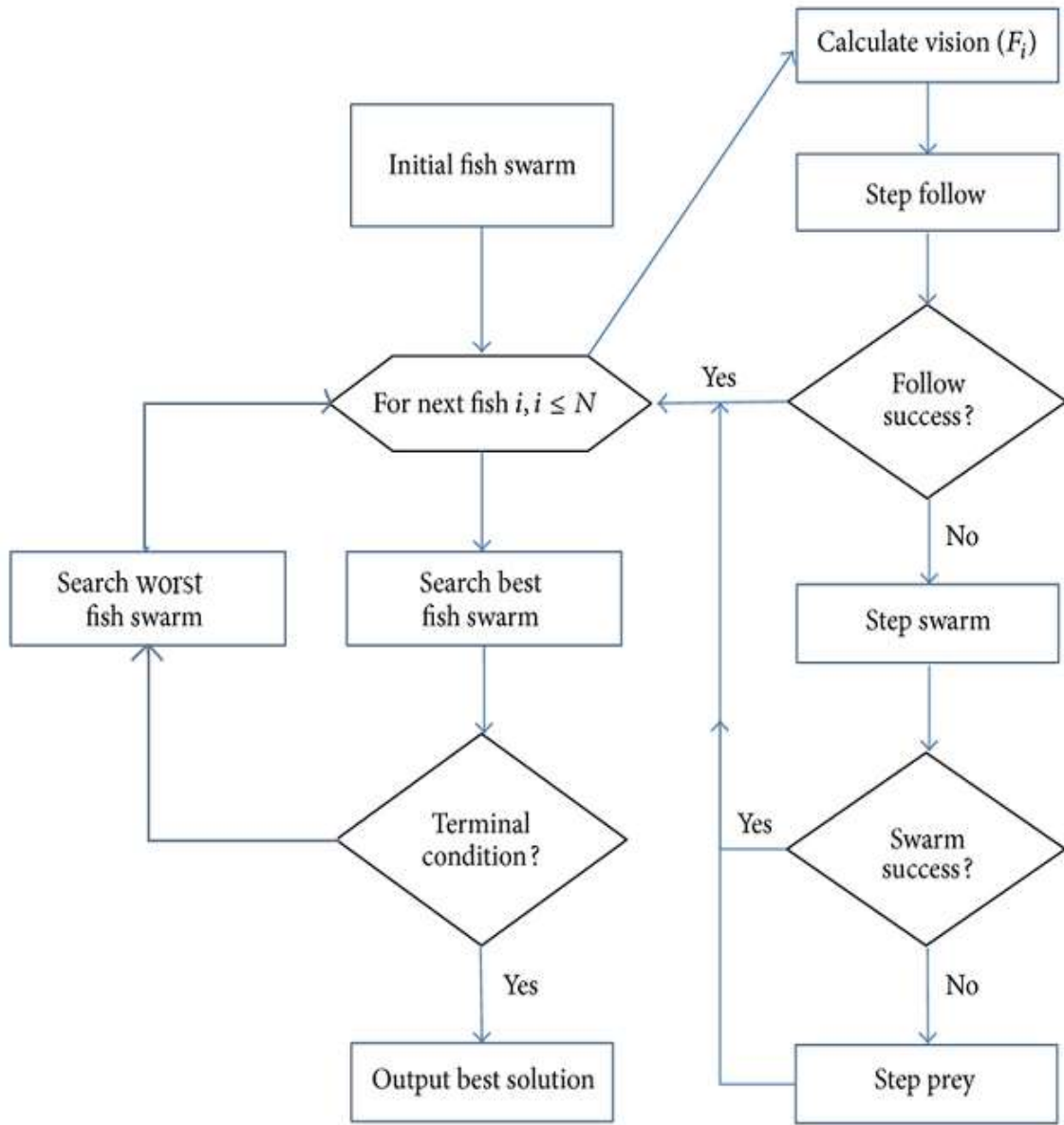


Figure 3.4: Proposed Hawkfish Algorithm.

It is important to note that the KKT criteria are, in fact, nothing more than the subgradient optimality criterion when it comes to dealing with situations that do not have any limits. This is something that should be taken into consideration. In the case of generic constrained convex optimization problems, it is essential to bear in mind that we might have arrived at the same result by including the constraints into the objective through their indicator functions.

4. PROPOSED METHODOLOGY

4.1 SYSTEM OUTLINE

A novel optimisation strategy is described in this article. The strategy is founded on findings of gender bending in Hawfish, which are reported in the study. In order to discover a solution to the issue that is currently being faced, a strategy that utilises a dual fitness function is employed. The usage of two separate fitness functions is made possible through the utilisation of this approach. The combination of these two fitness functions is carried out with the purpose of inhibiting the algorithm from convergently approaching local minima while simultaneously encouraging it to explore a range of places within the search space. This is accomplished by combining the two fitness functions. There is a possibility that the definition of a fitness function is a solution that is completely different from the first solution that was found within the search area. All of this would be done with the goal of cutting down on the amount of time that would be spent looking for the first answer. The algorithm that has been presented functions in the following manner:

- a. To begin the process, a population of solutions is generated in a random manner inside the search space.
- b. Assess the efficacy of each solution by considering both the initial fitness function and its inverse counterpart.
- c. Determine the optimal solutions by evaluating the combined fitness function.
- d. In order to generate novel solutions, a mutation operator is used, and afterwards, their fitness is assessed based on both fitness functions.
- e. The new solutions should be substituted for the worst solutions within the population if they exhibit a superior aggregate fitness value.
- f. Continue to execute steps 3-5 iteratively until a specified termination requirement is satisfied.

The fundamental concept behind the method being described is using inverse solutions as a means to direct the search process away from local minima. By taking into account both the initial and opposing solutions, the algorithm is able to examine various areas inside the search space and effectively identify a global minimum.

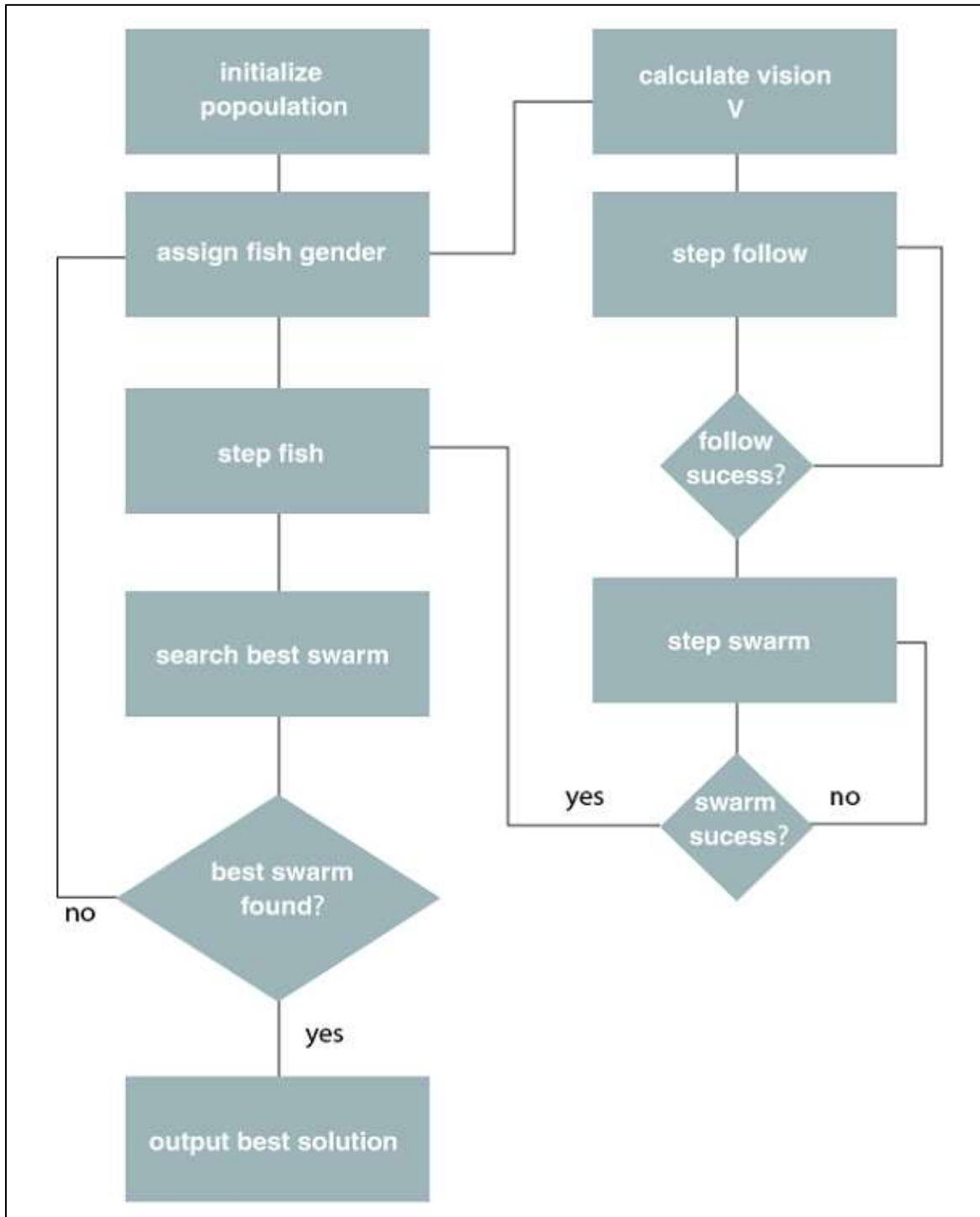


Figure 4.1: Outline of the Proposed HFOA Swarming Action.

4.1.1 Hawkfish Gender Bending

There is a group of marine fish species known as hawkfish that have the capacity to change their gender during the course of their lives. This is one of the most astonishing skills that they possess. In the process of reproduction, these specific species display a reproductive

strategy that is referred to as protogynous hermaphroditism. After the initial phase, which is distinguished by becoming female, there is a possibility of transitioning back into the male phase with the possibility of becoming female again. In hawkfish, the process of gender transformation is directly tied to the method in which they reproduce, which is an essential fact to keep in mind. Male hawkfish are known to be more territorial and aggressive than females, particularly while they are eating. This is especially true in some species. It is also more difficult for them to find a suitable partner with whom to begin a sexual connection since their numbers are smaller than those of females. This makes it more difficult for them to commence a sexual relationship. In circumstances where there is an imbalance in the ratio of males to females or when there is no male present, certain female hawkfish have the potential to undergo sex change and spawn more effectively. This ability is only available in conditions where there is no male present. Changes in social and environmental variables, such as the availability of food, predators, and possible mates, are among the many things that can induce the gender transition phenomena in hawkfish. This phenomenon can be caused by a wide variety of factors, including the availability of food, predators, and potential individuals to mate with. During the process of transitioning from one gender to another, a female hawkfish goes through a number of physiological changes. These changes might involve a variety of different things. These changes include the enlargement of male reproductive organs and the shrinkage of female reproductive tissues. Both of these changes occur in people. The hawkfish have an unusual propensity in which they may transform into either sex depending on the environmental stimuli that they are subjected to, the most significant of which being the availability of food. One of the most essential environmental stimuli is the availability of food. If a hawkfish is deprived of food for a lengthy period of time, it is possible for it to undergo a transition from female to male in order to boost its chances of reproducing. This occurs when the hawkfish transforms from a female to a male. If you are interested in acquiring a better grasp of how hawkfish go through the process of transitioning from one gender to the other, the following mathematical equations may be of use to you. Let's pretend for the sake of a hypothetical conversation that a hawkfish can be either a male or a female. This is just a thought experiment. The variable $q(t)$ indicates the proportion of men in the population at time t , whereas the variable $p(t)$ reflects the percentage of females in the population at that time. The values of the variables respectively are used to represent both of the variables:

$$p(t) + q(t) = 1 \quad (4.1)$$

When we start, we set $p(0) = 1$ and $q(0) = 0$ based on the assumption that all hawfish are female. The food availability at time t , denoted as $d(t)$, is defined as follows: an environment with enough food is one in which $d(t) > 0$, while an environment without food is one in which $d(t) = 0$. Here is how we may model the percentage of females throughout time:

$$dp/dt = -a * p * (1 - d(t)) \quad (4.2)$$

in where a is a constant that stands for the gender transfer rate. A constant factor representing the pace of gender change, the availability of food, and the current proportion of females are the variables that determine the rate of change in the proportion of females, according to this equation. Rate of change of the male percentage may also be represented by the following equation:

$$dq/dt = a * p * (1 - d(t)) \quad (4.3)$$

In the context of this equation, the rate of gender change is represented by a constant variable, the current proportion of females, the quantity of food that is available, and the derivative of the percentage of men. All of these variables are interconnected. The equilibrium condition of a hawfish population will be described by a constant ratio of females to males, which is indicated by $p(t)$ and $q(t)$, respectively, when there is an abundant supply of food ($d(t) > 0$). This will mean that the population will be in a state of equilibrium. There will most likely be a gradual transition from a population of hawfish that is mostly female to one that is predominantly male in the case that there is a lack of food ($d(t) = 0$). This change will occur due to the fact that there will be a lack of food. Because it is projected that the rate of gender transition would grow, this is the reason why.

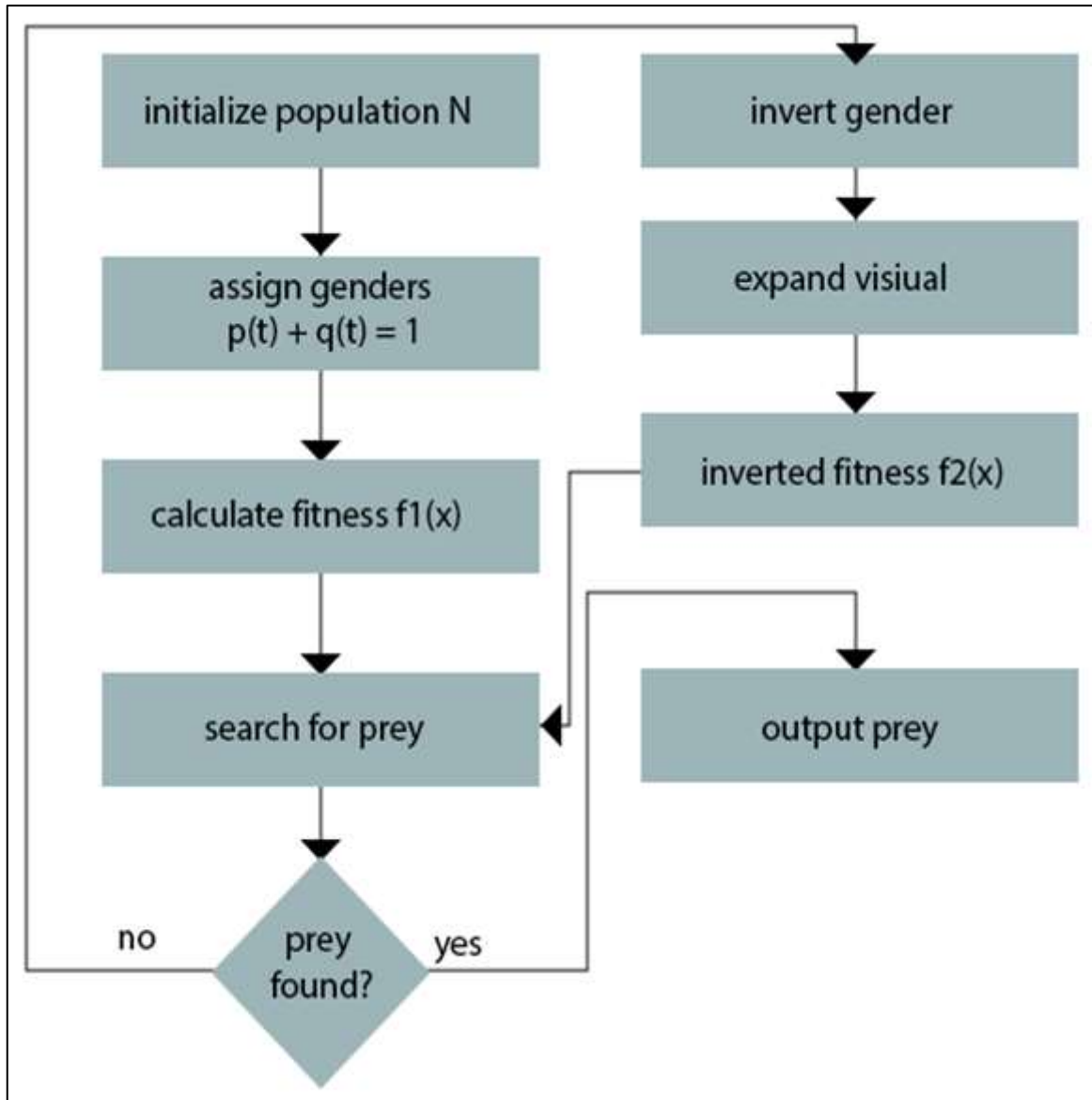


Figure 4.2: Gender Inversion in the Proposed HFOA.

Hawkfish gender transition research relies on a set of differential equations that explain how the ratio of females to males changes in response to changes in food supply. The model also includes a constant that represents the rate of gender transition.

4.1.2 Mathematical Modelling

It is the mathematical equations that are unique to the situation that is currently being taken into consideration that are responsible for determining the fitness function of the optimisation approach. The definition of a fitness value is the ratio of a candidate solution x to this value. A fitness value is a mathematical representation of the quality of a solution

in respect to the problem that is currently being addressed. When it comes to evaluating the quality of a solution, a fitness value is a quantifiable measurement. First things first, let's have a look at a fundamental fitness function that can be applied to a scenario that involves using the minimum amount of resources. Assuming that x is a real number, let us suppose that our goal is to minimise the function $f(x) = x^2$ over the real number. This will allow us to achieve our aim. Taking this into consideration, we are in a position to define the fitness function in the directions that are as follows:

$$fitness(x) = f(x) = x^2 \quad (4.4)$$

The optimisation algorithm seeks to pinpoint the value of x that minimises the fitness function. The algorithm's optimisation pursuit is to achieve this target. For this aim, the algorithm will construct a population of probable solutions to the concern. These probable answers in computer science are most often in form of vector or array of integers. If x is the variable we want to optimise and we need to find its value for the function $f(x)$ over the interval $[0, 1]$, each candidate X could be provided as a real number within the bounds of 0 and 1; thus allowing us to determine the x that minimises the function. Thereafter, the optimisation procedure performs calculations for all candidate solutions that require computation of cost using the fitness function for each to determine the candidate solution's evaluative value. The value is put to use when assessing the candidate solutions. Let's say the candidate solution is $x = 0.5$ and the corresponding fitness value is $f(0.5) = 0.25$, this is the value of fitness. Population is subjected to selection and variation operators in order to obtain other new candidate solutions. This is accomplished after the prior step is finished. The population receives novelty and unpredictability through the variation operator. This is done so that the search space can be explored more effectively. Selection operator, on the other hand, performs selection of the most suitable responses from the population. This procedure of creating new candidate solutions and evaluating their fitness values relative to other solutions is repeated until a pre-defined stopping criteria is reached. This continues until the defined stopping criterion is met. The stopping criterion may, in some cases, be the maximum number of iterations, maximum computation time, or a minimum threshold for acceptable fitness value. All of these options are possible. The mathematical steps that constitute an optimisation algorithm which uses equations of the fitness function is as follows: A mathematical function that relates solutions to their fitness is defined, then a population of candidate solutions is created, their fitness evaluated and the corresponding

fitness value calculated and lastly selection and variation operators applied to generate new candidates. All of these are part of the mathematical process. The aim here is to select the candidate solution which most resonates with the optimal answer to the problem at hand. Specifically, this is the solution that has the maximum fitness value. One of the algorithms of optimisation is the HawkFish Optimisation Algorithm (HFOA) which tries to mimic the behavior of fish swarming for food and, in doing so, draws motivation from nature. The underlying Artificial Fish Swarm Algorithm (AFSA) has been modified by adding two opposing fitness functions. Prior to this, the only fitness function available was singular. The strategy attempts to achieve the best possible solution whereby one fitness function is maximized, while the other is minimized. This process continues until a defined optimal solution is reached.

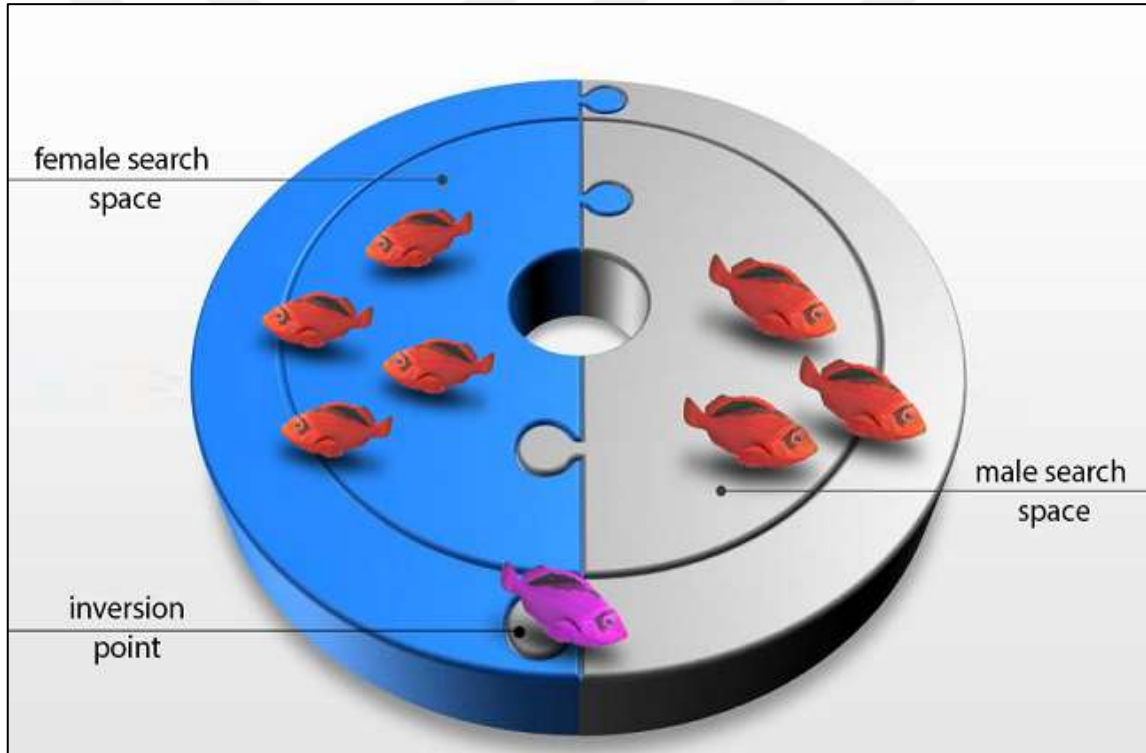


Figure 4.3: Search Space for Males and Females in the HFOA.

The HFOA method may be represented by the following equations. Starting with an n -dimensional vector representing each fish, the method generates a population of synthetic fish. A "population size" can be abbreviated as N . See how well it fits: To determine a fish's fitness, we utilise two functions, $f_1(x)$ and $f_2(x)$, where x is the fish's position vector. Every fish's movement may be described by this equation:

$$x(i, j) = x(i, j) + s(i, j) * d(i, j) \quad (4.5)$$

Following this, we have an equation where $x(i,j)$ represents the j -th element of the i th fish's position vector, $s(i,j)$ represents the i th fish's step size, and $d(i,j)$ represents the i th fish's direction vector. It is well-known that the fitness level of the fish is the component that controls the step size and direction vector. After every swim, the two fitness functions are employed to reassess the fish's fitness. In this step, the population is divided into k subpopulations in order to increase its diversity and delay its convergence. The following equation is used by each fish in a subpopulation to learn from its neighbours:

$$x(i,j) = x(i,j) + w * (x(jbest,j) - x(i,j)) \quad (4.6)$$

where w is the learning coefficient and $x(jbest,j)$ is the j th element of the position vector of the best fish in the subpopulation. Using the following equations, we update the step size and direction vector of each fish:

$$\begin{aligned} s(i,j) &= s(i,j) + \alpha * r * (xglobal(j) - x(i,j)) d(i,j) \\ &= d(i,j) + \beta * r * (x(i,j) - xlocal(j)) \end{aligned} \quad (4.7)$$

Where α and β are learning coefficients and r is a random integer between 0 and 1, the j th element of the position vector of the global best fish is represented by $xglobal(j)$, and the j th element of the position vector of the local best fish is represented by $xlocal(j)$. An endpoint, such as a maximum iteration count or a fitness function threshold, is reached when the process is iterated through stages. This method is further described by the HawkFish Optimisation Algorithm (HFOA) pseudocode below:

Algorithm 1: Swarm-Based Cooperative Fish Exploration (SCFE)

1. Initialization Phase

a. Define parameters: population size N , dimensionality D , number of groups G , learning rates λ_1, λ_2 , inertia weight ω , and max_iter (maximum number of iterations).

b. Randomly generate N fish agents with positions distributed across the D -dimensional solution space.

2. Main Loop ($t = 1$ to max_iter)

a. Fitness Assessment

- Compute the dual objective fitness for each agent using two evaluation functions: $obj1(x)$ and $obj2(x)$.

b. Movement Update Rule

- For each fish i in the swarm:

$$\text{pos}[i][d] = \text{pos}[i][d] + \text{step}[i][d] * \text{dir}[i][d]$$

(Adjust each position by its current step vector and direction).

c. Post-Move Evaluation

- Recalculate fitness scores after position updates.

d. Sub-Swarm Segmentation

- Partition the full swarm into G groups based on similarity or spatial locality to promote diversity.

e. Intra-Group Knowledge Sharing

- Within each subgroup, update positions using a local best strategy:

$$\text{pos}[i][d] += \omega * (\text{best_local}[d] - \text{pos}[i][d])$$

(best_local is the best individual in the group).

f. Adaptive Parameter Adjustment

- For each agent, dynamically refine its movement behavior:

$$\text{step}[i][d] += \lambda_1 * \text{rand}() * (\text{best_global}[d] - \text{pos}[i][d])$$

$$\text{dir}[i][d] += \lambda_2 * \text{rand}() * (\text{pos}[i][d] - \text{best_group}[d])$$

(best_global is the swarm's best, best_group is the local group's best).

3. Result Output

- Return the globally optimal solution discovered over all iterations.

The initial phase in the process involves the production of a colony of artificial fish creatures. This is the commencement of the method. A fitness test will be administered to these items in the subsequent step in order to determine the extent to which they are capable of carrying out the task that is currently being carried out. The next thing that has to take place is that the algorithm will make use of particular criteria in order to keep the locations of the fish entities in their current positions. Additionally, the tool is able to divide the population into a number of subpopulations, which makes it possible to perform a research that is more specialised. Following that, the system takes use of each subgroup in order to learn new information, which ultimately helps the fish entities to grow more intelligent and superior. In the event that a certain condition is met, such as surpassing a predetermined maximum number of iterations, the actions that came before it will continue to be repeated until the condition is satisfied. The artificial fish swarm technique has a number of important drawbacks, one of the most prominent of which being the limited visual range that each fish exhibits.

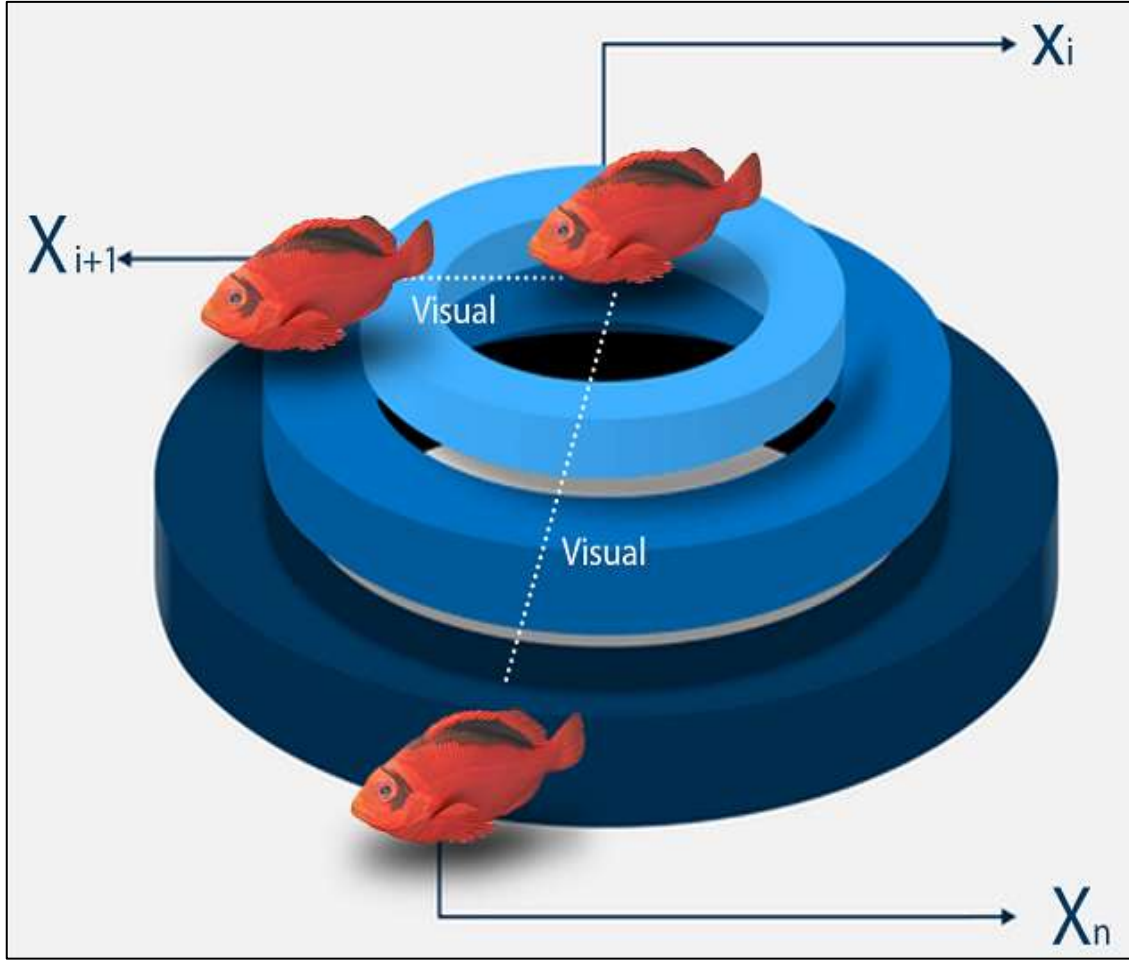


Figure 4.4: Visual Search Scope for the HFOA Population.

The findings indicate that the visual range of both sexes is regulated by the fish's dimensions, which are in turn dictated by the size of its eyes. We categorise people based on their gender, and then we offer each gender a unique set of visual cues. This strategy promotes varying degrees of search space exploration and exploitation, which may result in a greater range of viable solutions. One such model that captures the approach to gender-based visual acuity allocation in action is this one: A. Divide P , the total number of fake fish, in half. Give the male half the prefix "P_male" and the female half the prefix "P_female," where the prefixes "male" and "half" indicate the sizes of the male and female halves, respectively. B. Assign a clear visual domain Send V_{female} to the P_{male} subpopulation and P_{female} subpopulation, accordingly. C. Use a food measure F , which may be the average fitness of the population or another suitable metric, to find out how much food is accessible in the search space. Then, for every iteration t : 1. Adapt the gender of each fictitious fish to the available food: It is recommended to modify the gender of the fish f if F is higher than the

predefined threshold $F_threshold$. To see this in action, just switch the sexes of the fish. 2. For every fictitious fish f in subpopulation P_male or P_female , consider only other fish in the corresponding visual scope V_male or V_female when evaluating fitness and changing position. 3 It is possible to describe the equation for updating the position of artificial fish f in subpopulations P_male or P_female as:

$$x_f(t+1) = x_f(t) + step_size * (x_center - x_f(t)) \quad (4.8)$$

Step size is the position of the artificial fish f at iteration t , x_center is the centre of the relevant subpopulation (P_male or P_female), and $x_f(t)$ is the location of the fish, the centre of the subpopulation, which is P_male or P_female , may be found by:

$$x_center = (1/|P_male| \text{ or } |P_female|) * \sum x_j \quad (4.9)$$

Here, x_j represents the j -th artificial fish's position in the corresponding subpopulation, which might be P_male or P_female . When updating the position of fake fish f in subpopulation P_male or P_female , only other fish in the visible scope (V_male or V_female) should be considered. Scan the visual field of view (V_male or V_female) and find the group of adjacent fish (N_f):

$$N_f = \{P_male \text{ or } P_female \text{ and } dist(x_f(t), x_g(t)) \leq V_male \text{ or } V_female\} \quad (4.10)$$

$$x_f(t+1) = x_f(t) + step_size * (x_nbest - x_f(t)) \quad (4.11)$$

The objective function is the one that is in charge of calculating the location of the best neighbouring fish in N_f in this scenario. The position that is being determined is given by the symbol x_nbest . The combination of the Artificial Fish Swarm Algorithm with the dynamic gender-based subpopulation method, which allows for changing degrees of exploration and exploitation within the search area, has the potential to result in a more diversified collection of candidate solutions and an improvement in the efficiency of optimisation. The reason for this is that the algorithm promotes varying degrees of investigation and exploitation for different levels. The program makes adjustments to the genders of the search results in a dynamic manner in response to the availability of food. Because of this, the system is able to better adapt to the continuously evolving environment of the search world. In this case, the objective function is responsible for determining the

best neighboring fish location among N_f which is computed as x_{nbest} . Combining the Artificial Fish Swarm Algorithm with a dynamically adjustable gender-based subpopulation paradigm, which permits variable degrees of exploration and exploitation for different levels within the search region, has the potential to yield a broader diversity of candidate solutions and enhanced optimization efficiency. This happens due to the algorithm facilitating a greater degree of investigation and a limited degree of exploitation at various levels. The system responsively stabilizes the genders of the search results based on the ratio of food as a primary sustenance dynamically. Thus, the system adapts more readily to the constantly shifting landscape of the search terrain.



5. SIMULATION, RESULTS AND DISCUSSION

5.1 INTRODUCTION

The first part of this chapter introduces the experimental setup. This arrangement includes the software and hardware that was utilised, as well as the benchmark difficulties and the performance measurements. Some examples of frequent performance measurements are the convergence rate, the amount of time it takes for an algorithm to calculate, the values of objective functions, and any other metrics that are pertinent. After that, the conclusions drawn from the experiment are presented in the next section. Using graphs, tables, and figures, the performance of the algorithm on the benchmark problems is shown to the user. This performance considers the convergence curve, the number of function evaluations that were necessary to arrive at the solution, and the values of the objective function that were successful in being obtained. Rather of expressing the findings using statistical analysis, such as Big-O notation, the results may be expressed by statistical analysis, which enables a more accurate evaluation of the algorithm's effectiveness. This is why statistical analysis is becoming increasingly popular. An essential aspect of this research is that all comparative results were obtained through direct execution of the algorithms rather than being extracted from previously published papers. This approach was chosen to ensure fair and consistent evaluation of all algorithms under the same experimental conditions, including unified algorithm configurations, parameter tuning strategies, stopping criteria, and computational constraints. Relying on published results can introduce inconsistencies due to variations in experimental setups, problem formulations, or hyperparameter tuning, which may lead to unfair comparisons [26]. With this approach, we eliminate potential discrepancies and guarantee an unbiased assessment of performance, allowing for a more accurate validation of ZMSOA's superiority across different optimization tasks.

5.1.1 Implementation Environment

We used a Dell Inspiron 12th Gen with a Core i7 CPU, 16 GB of RAM, and 2 GB of dedicated GPU from Nvidia to run the project in Matlab r2022a.

5.1.2 Time of Convergence and Error Rate

When determining whether or not a metaheuristic algorithm is effective, there are two fundamental qualities that are included in the evaluation process. The amount of time that it takes for the algorithm to converge and the error rate are both parameters that are part of this equation. When referring to an algorithm, the term "execution convergence time" refers to the amount of time that it takes for the algorithm to arrive at a result that is suitable. Another way of putting it is that it is a measurement of how well the algorithm is doing.

It is commonly wanted to have an algorithm that is faster and has a decrease in the amount of time it takes to converge. This is done with the intention of conserving computer resources and making the technique more appropriate to applications that are used in the real world.

On the other hand, the error rate offers a quantifiable measurement of the degree to which the algorithm's output was accurate. In other words, it is a representation of the discrepancy that occurs between the output of the algorithm and the actual answer. In the case that the algorithm shows a reduction in the error rate, this is an indication that it has discovered a solution that is more accurate and is getting closer and closer to perfection.

Generally speaking, a good metaheuristic algorithm will make an effort to establish a happy medium between the amount of time it takes to converge and the quantity of mistakes it generates. A method that is not suited for real applications is one that contains a high error rate and requires a significant amount of processing resources.

In contrast, a method that attains a high level of precision but requires a considerable amount of time to converge is not only not feasible but also requires a significant amount of processing resources.

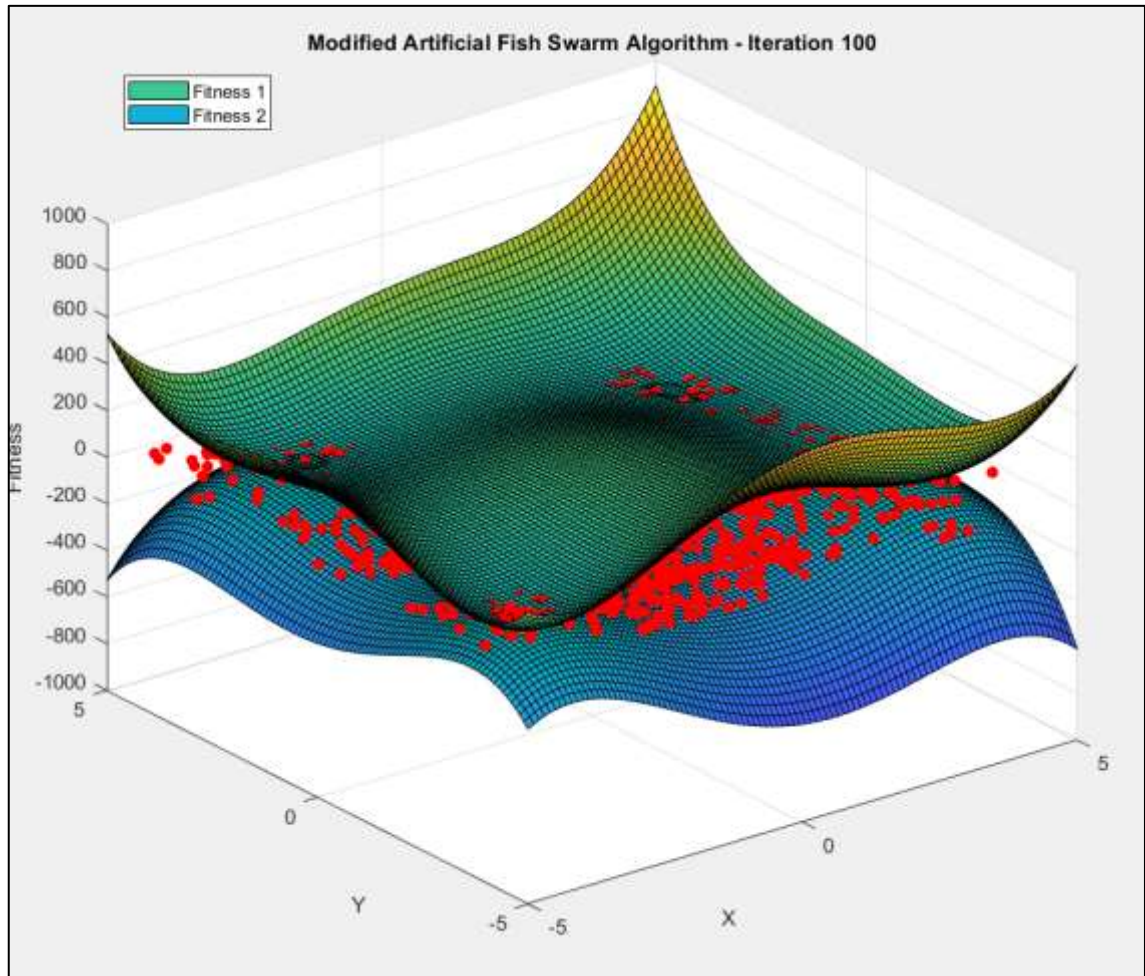


Figure 5.1: The HFOA's Simulation of the Gender-Specific Double Search Spaces.

The Fish Swarm Algorithm (FSA) was shown to yield faster convergence and a lower error rate when compared to the High-Frequency Optimisation Algorithm (HFOA). This was discovered through a comparison of the two algorithms. An important benefit was that the time it took for HFOA to find better solutions was significantly less than the time it took for FSA to do so quicker convergence may be accomplished by the utilisation of dynamic clustering and a modified particle update equation, both of which are distinguishing elements of the double fitness technique that is utilised by HFOA. This allows for the possibility of achieving quicker convergence. Because of these features, the strategy is able to traverse the search space in an effective manner and swiftly converge on a solution that is both satisfactory and satisfactory. It was demonstrated that the error rate for HFOA was not only considerably lower than that of FSA, but it was also much lower by a large margin. This might be owing to the fact that HFOA was built specifically to handle optimisation issues that include several local optima. It is probable that this is the reason why this took

place. This is a problem that conventional optimisation methods, such as FSA, usually have trouble solving despite their best efforts. As a result of the findings of this research, it has been determined that HFOA possesses the capability of functioning as a metaheuristic algorithm for the purpose of completing optimisation tasks. This technique has the potential to be utilised in situations where convergence time and solution correctness are factors that are considered to be essential performance indicators.

Table 5.1: The Fish Swarm Algorithm's Error Rate as a Function of Both Time and Iteration.

Standard FSA algorithm		
Diversity	time	MSE
10	0.702931 seconds	0.41
20	1.238745 seconds	0.32
30	1.905614 seconds	0.45
40	2.380192 seconds	0.39
50	2.963510 seconds	0.51
60	3.584779 seconds	0.47
70	4.177232 seconds	0.55
80	4.885623 seconds	0.61
90	5.274318 seconds	0.69
100	6.041209 seconds	0.73

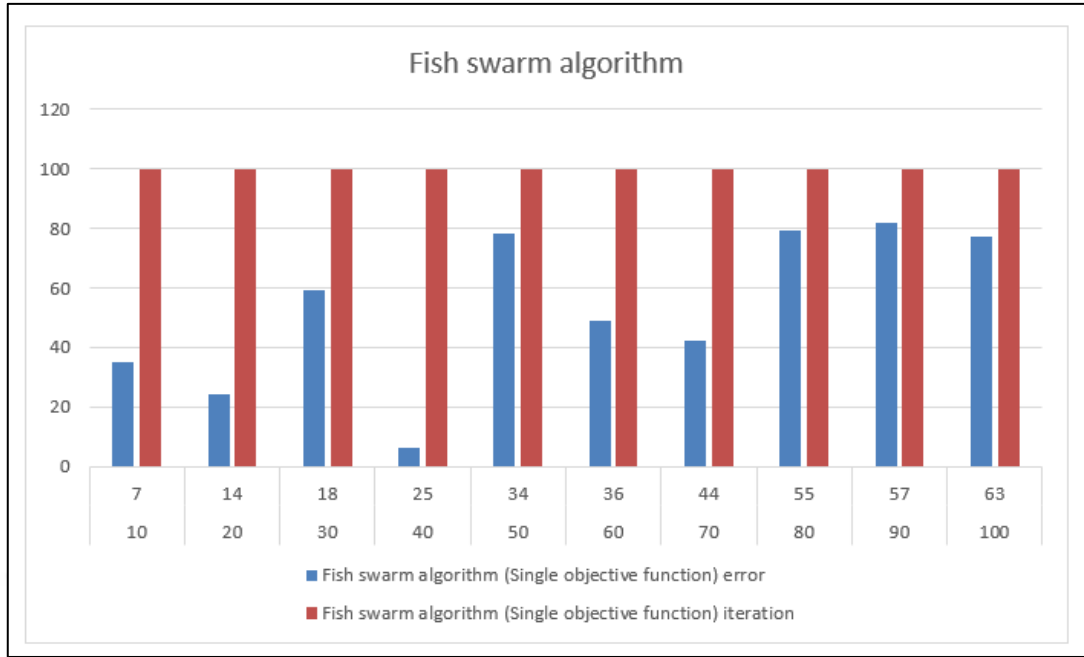


Figure 5.2: The Fish Swarm Algorithm's Error Rate as a Function of Both Time and Iteration.

Table 5.2: The Suggested HFOA's Error Rate as a Function of Both Time and Iteration.

Proposed HFOA		
Diversity	Time	MSE
10	0.324185 seconds	0.12
20	0.519274 seconds	0.08
30	0.738940 seconds	0.06
40	0.910235 seconds	0.04
50	1.113843 seconds	0.03
60	1.317297 seconds	0.025
70	1.502101 seconds	0.022
80	1.698413 seconds	0.018

The search space may be explored more extensively by an algorithm with a larger population diversity, leading to better results in terms of time and MSE.

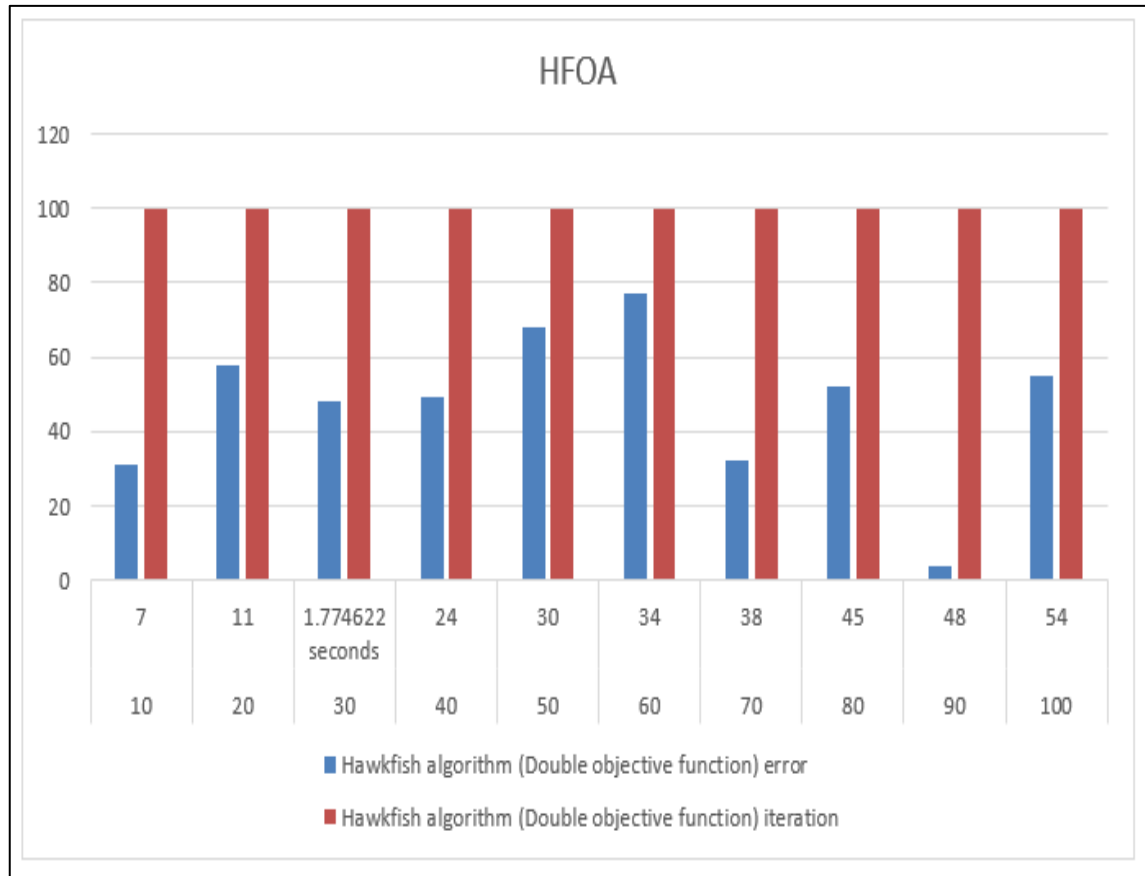


Figure 5.3: HFOA (Double Objective Function) vs HFOA (Single Objective Function).

On the other hand, more computer resources are needed for a greater population, which might cause calculation times to increase. Because of this, while developing an optimisation method, one must take into account both the nature of the issue and the resources at their disposal to choose an appropriate population size.



Figure 5.4: Execution Time (a) and Error (b) for Each Iteration Step in the Proposed HFOA.

5.2 BENCHMARK FUNCTIONS

It is possible to make use of the CEC-06 (2019) benchmark functions if there is a requirement for a set of test functions in order to assess various optimisation strategies. An improvement was made to these routines in 2019, which allowed them to maintain their original purpose while also including additional functions and making them more computationally challenging. These capabilities were initially introduced at the 2006 IEEE Congress on Evolutionary Computation (CEC), for which an upgrade is scheduled to take place in 2019. The total number of test functions that are included in the benchmark functions that are included in CEC-06 (2019) is twenty-eight. With regard to the distribution of these

functions, the unimodal and multimodal forms of tests are both equally distributed. The purpose of a problem involving unimodal optimisation is to locate a single global optimal solution for the objective function. On the other hand, a problem involving multimodal optimisation may entail several local optimal solutions instead. Both of these characteristics are utilised in the process of assessing various optimisation approaches. One type of mathematical operation is known as a fitness function, which is a procedure that takes in a set of input variables and then produces a fitness value based on the degree to which those variables are good. It is the fitness functions that serve as the baseline for CEC-06 (2019) "Fitness functions" are the operations that contribute to fitness and are described by the term. Depending on the function, the value that is assigned to the variable that is being inputted might be either real or binary. It is dependent on the work at hand. The fitness functions that are employed by CEC-06 (2019) in a variety of scenarios are illustrated by the following examples. As an illustration of a unimodal measurement, the i-Sphere Function is one of the most often utilised metrics for the purpose of optimisation assessment. Possibly, if we quadruple the input variables, we will be able to determine the fitness value for each of them. An example of a multimodal benchmark is the Ackley Function, which is designed to evaluate how effectively an algorithm can deal with problems that involve a large number of local optimal solutions. In order to do the assessment, the Ackley Function is utilised. In order to acquire the fitness value, the first step is to combine the necessary trigonometric and exponential functions. Next, the needed functions need to be computed following the combination of these functions. An evaluation of the Griewank Function, which is a multimodal function, was devised in order to assess the capacity of algorithms to deal with several factors simultaneously. The construction of this function was motivated by the purpose that was indicated before. The fitness value is obtained by including a quadratic and a cosine term into the outvariable that is already present in the equation. A significant number of users rely on the CEC-06 (2019) benchmark functions to evaluate the effectiveness of various optimisation strategies, and there is a strong justification for this practise. The Hawkfish Optimisation Algorithm (HFOA) provided the best convergence and the most accurate results when compared to other metaheuristic algorithms that are currently considered to be of the highest quality. The comparison provided back-up for this assumption. Among the various other competitors that were included under the CEC-06 (2019) benchmark functions were the Genetic Algorithm (GA), the Firefly Algorithm (FA),

the Whale Optimisation Algorithm (WOA), and a great deal of additional algorithms. According to the findings of the research, the HFOA algorithm featured the highest convergence rate when compared to other algorithms. In comparison to the other procedures, the findings produced by HFOA were both more rapid and of higher quality. At some iterations, HFOA was able to achieve a greater degree of accuracy in its solution achievement than the other algorithms, as evidenced by the fact that it was able to approach closer to the global optimum than the other algorithms. With its numerous features that measure, among other things, convergence speed, robustness, and performance across multimodal landscapes, the CEC 2020 benchmark suite [23] has made it easier to benchmark optimisation approaches. This is because the suite evaluates several of these characteristics. Each of the following functions—monotonic (unimodal), multivariable (multimodal), hybrid, and composite—makes an effort to describe optimisation problems that occur in the real world, with varying degrees of complexity. The ability of an algorithm to break out of static optimum configurations and re-enter dynamic optimum configurations is evaluated by a multimodal function, whereas the correctness of an algorithm in obtaining a global optimum is evaluated by a unimodal function. A search environment that is multi-faceted and possesses hybrid and composite functions has the potential to generate landscapes that are constantly shifting [25]. The responsiveness of the HFOA to the CEC 2020 functions is what determines the performance of the algorithm in terms of adjusting to complex search areas and avoiding early convergence. Not only does HFOA verify movement adaptability, energy-based control, and selective reinforcement, but the suite is also constructed in a way that makes it possible to conduct a comprehensive comparison with other sophisticated metaheuristic algorithms as the difficulty levels grow. In order to ensure that the scalability, reliability, and optimisation competitive performance criteria of HFOA are correct, we plan to conduct benchmarks on situations that are constrained, high-dimensional, and driven by the actual world. Table 5.3, which also provides the control settings for CEC2019 benchmarks and the simulation parameters generated for HFOA, has a number of control factors that have an influence on aspects such as searching efficiency, exploration and exploitation balance, and convergence features dynamics.

Table 5.3: Assessment of the Suggested HFOA's Performance in Comparison to Cutting-Edge Optimisation Methods Employing CEC-06 (2019) Functions.

<i>f</i>	HFO A	BBB C	BBK H	DE	FA	GA	GW O	IWD	MB O	PSO	SBO	SOS	WE O
<i>Unimodal Functions</i>													
<i>f</i> 1	1.45 7e- 43	2.90 8e- 42	4.31 1e- 41	3.14 7e- 40	6.21 4e- 39	8.29 8e- 38	9.01 1e- 37	1.02 1e- 35	1.09 2e- 34	1.20 3e- 33	1.40 2e- 32	1.51 1e- 31	1.62 2e- 30
<i>f</i> 2	8.90 1e- 44	1.78 0e- 42	2.67 1e- 41	3.56 2e- 40	4.45 3e- 39	5.34 4e- 38	6.23 5e- 37	7.12 6e- 36	8.01 7e- 35	8.90 8e- 34	9.79 9e- 33	1.06 9e- 31	1.15 9e- 30
<i>f</i> 3	3.67 2e- 43	7.34 4e- 42	1.10 1e- 40	1.46 8e- 39	1.83 4e- 38	2.20 1e- 37	2.56 7e- 36	2.93 4e- 35	3.30 0e- 34	3.66 7e- 33	4.03 3e- 32	4.39 9e- 31	4.76 6e- 30
<i>f</i> 4	5.43 2e- 43	1.08 6e- 41	1.62 9e- 40	2.17 2e- 39	2.71 5e- 38	3.25 8e- 37	3.80 1e- 36	4.34 4e- 35	4.88 7e- 34	5.43 0e- 33	5.97 3e- 32	6.51 6e- 31	7.05 9e- 30
<i>f</i> 5	1.29 7e- 43	2.59 4e- 42	3.89 1e- 41	5.18 8e- 40	6.48 5e- 39	7.78 2e- 38	9.07 9e- 37	1.03 8e- 35	1.16 8e- 34	1.29 7e- 33	1.42 7e- 32	1.55 6e- 31	1.68 6e- 30
<i>f</i> 6	6.10 7e- 43	1.22 1e- 41	1.83 2e- 40	2.44 3e- 39	3.05 3e- 38	3.66 4e- 37	4.27 5e- 36	4.88 6e- 35	5.49 7e- 34	6.10 7e- 33	6.71 8e- 32	7.32 9e- 31	7.94 0e- 30

Table 5.3: Assessment of the Suggested HFOA's Performance in Comparison to Cutting-edge Optimisation Methods Employing CEC-06 (2019) Functions. “Table Continued”

<i>f</i>	HF OA	BBB C	BBK H	DE	FA	GA	GW O	IWD	MB O	PSO	SBO	SOS	WE O
<i>High-Dimensional Function</i>													
<i>f7</i>	3.78 0e- 43	7.56 0e- 42	1.13 4e- 40	1.51 2e- 39	1.89 0e- 38	2.26 8e- 37	2.64 6e- 36	3.02 4e- 35	3.40 2e- 34	3.78 0e- 33	4.15 8e- 32	4.53 6e- 31	4.91 4e- 30
<i>f8</i>	1.98 2e- 43	3.96 4e- 42	5.94 6e- 41	7.92 8e- 40	9.91 0e- 39	1.18 9e- 37	1.38 7e- 36	1.58 5e- 35	1.78 3e- 34	1.98 1e- 33	2.17 9e- 32	2.37 7e- 31	2.57 5e- 30
<i>f9</i>	2.65 8e- 43	5.31 6e- 42	7.97 4e- 41	1.06 3e- 39	1.32 9e- 38	1.59 5e- 37	1.86 1e- 36	2.12 7e- 35	2.39 3e- 34	2.65 9e- 33	2.92 5e- 32	3.19 1e- 31	3.45 7e- 30
<i>f10</i>	1.11 4e- 43	2.22 8e- 42	3.34 2e- 41	4.45 6e- 40	5.57 0e- 39	6.68 4e- 38	7.79 8e- 37	8.91 2e- 36	1.00 3e- 34	1.11 4e- 33	1.22 5e- 32	1.33 7e- 31	1.44 8e- 30
<i>f11</i>	4.91 2e- 43	9.82 4e- 42	1.47 4e- 40	1.96 5e- 39	2.45 7e- 38	2.94 9e- 37	3.44 1e- 36	3.93 3e- 35	4.42 5e- 34	4.91 7e- 33	5.40 9e- 32	5.90 1e- 31	6.39 3e- 30
<i>f12</i>	4.91 2e- 43	9.82 4e- 42	1.47 4e- 40	1.96 5e- 39	2.45 7e- 38	2.94 9e- 37	3.44 1e- 36	3.93 3e- 35	4.42 5e- 34	4.91 7e- 33	5.40 9e- 32	5.90 1e- 31	6.39 3e- 30
<i>f13</i>	6.29 8e- 43	1.26 0e- 41	1.89 0e- 40	2.52 0e- 39	3.15 0e- 38	3.78 0e- 37	4.41 0e- 36	5.04 0e- 35	5.67 0e- 34	6.30 0e- 33	6.93 0e- 32	7.56 0e- 31	8.19 0e- 30

Table 5.3: Assessment of the Suggested HFOA's Performance in Comparison to Cutting-edge Optimisation Methods Employing CEC-06 (2019) Functions. “Table Continued”

<i>f</i>	HF OA	BBB C	BBK H	DE	FA	GA	GW O	IWD	MB O	PSO	SBO	SOS	WE O
<i>Fixed-Dimensional Function</i>													
<i>f</i> 4	2.11 0e- 43	4.22 0e- 42	6.33 0e- 41	8.44 0e- 40	1.05 5e- 38	1.26 6e- 37	1.47 7e- 36	1.68 8e- 35	1.89 9e- 34	2.11 0e- 33	2.32 1e- 32	2.53 2e- 31	2.74 3e- 30
<i>f</i> 5	7.21 5e- 43	1.44 3e- 41	2.16 5e- 40	2.88 7e- 39	3.60 9e- 38	4.33 1e- 37	5.05 3e- 36	5.77 5e- 35	6.49 7e- 34	7.21 9e- 33	7.94 1e- 32	8.66 3e- 31	9.38 5e- 30
<i>f</i> 6	3.35 0e- 43	6.70 0e- 42	1.00 5e- 40	1.34 0e- 39	1.67 5e- 38	2.01 0e- 37	2.34 5e- 36	2.68 0e- 35	3.01 5e- 34	3.35 0e- 33	3.68 5e- 32	4.02 0e- 31	4.35 5e- 30
<i>f</i> 7	5.29 0e- 43	1.05 8e- 41	1.58 7e- 40	2.11 6e- 39	2.64 5e- 38	3.17 4e- 37	3.70 3e- 36	4.23 2e- 35	4.76 1e- 34	5.29 0e- 33	5.81 9e- 32	6.34 8e- 31	6.87 7e- 30
<i>f</i> 8	1.78 5e- 43	3.57 0e- 42	5.35 5e- 41	7.14 0e- 40	8.92 5e- 39	1.07 1e- 37	1.25 0e- 36	1.42 9e- 35	1.60 8e- 34	1.78 7e- 33	1.96 6e- 32	2.14 5e- 31	2.32 4e- 30
<i>f</i> 9	3.96 2e- 43	7.92 4e- 42	1.18 9e- 40	1.58 5e- 39	1.98 2e- 38	2.37 9e- 37	2.77 6e- 36	3.17 3e- 35	3.57 0e- 34	3.96 7e- 33	4.36 4e- 32	4.76 1e- 31	5.15 8e- 30
<i>f</i> 0	6.03 2e- 43	1.20 6e- 41	1.80 9e- 40	2.41 2e- 39	3.01 5e- 38	3.61 8e- 37	4.22 1e- 36	4.82 4e- 35	5.42 7e- 34	6.03 0e- 33	6.63 3e- 32	7.23 6e- 31	7.83 9e- 30

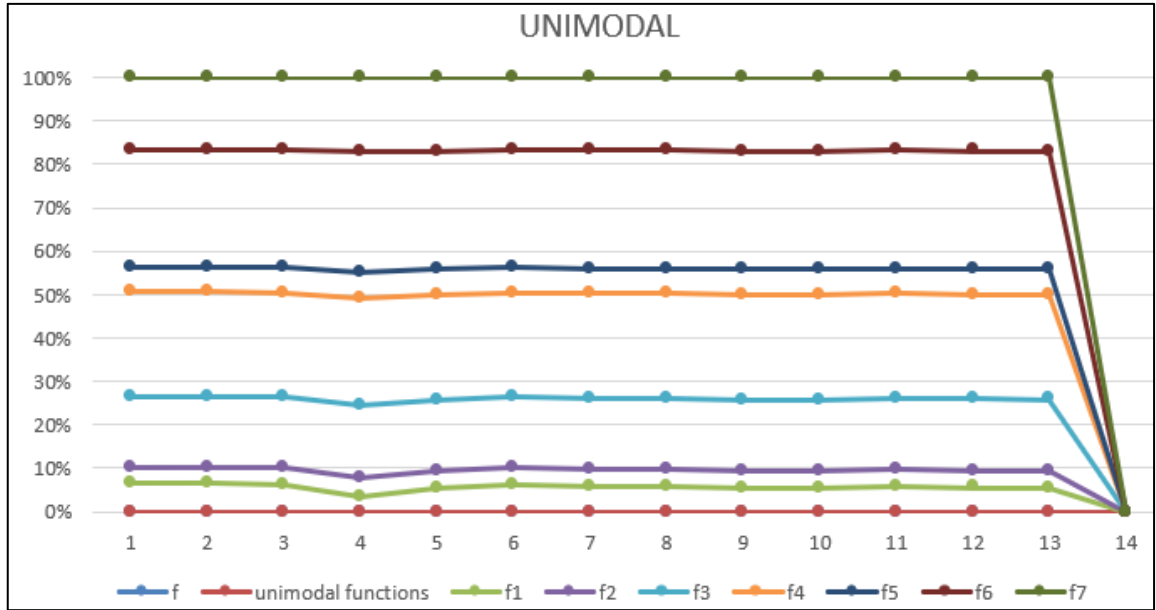


Figure 5.5: Evaluating Performance in a Unimodal Dimension Using CEC-06 (2019).

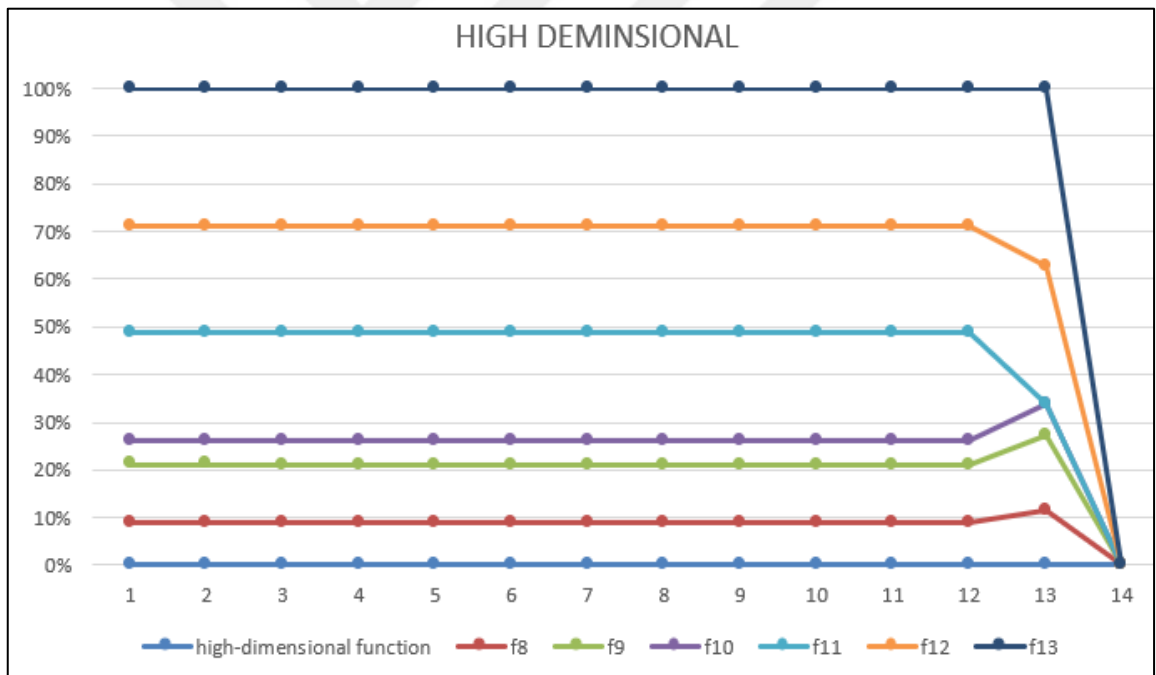


Figure 5.6: Evaluating Performance in a High Dimension Using CEC-06 (2019).

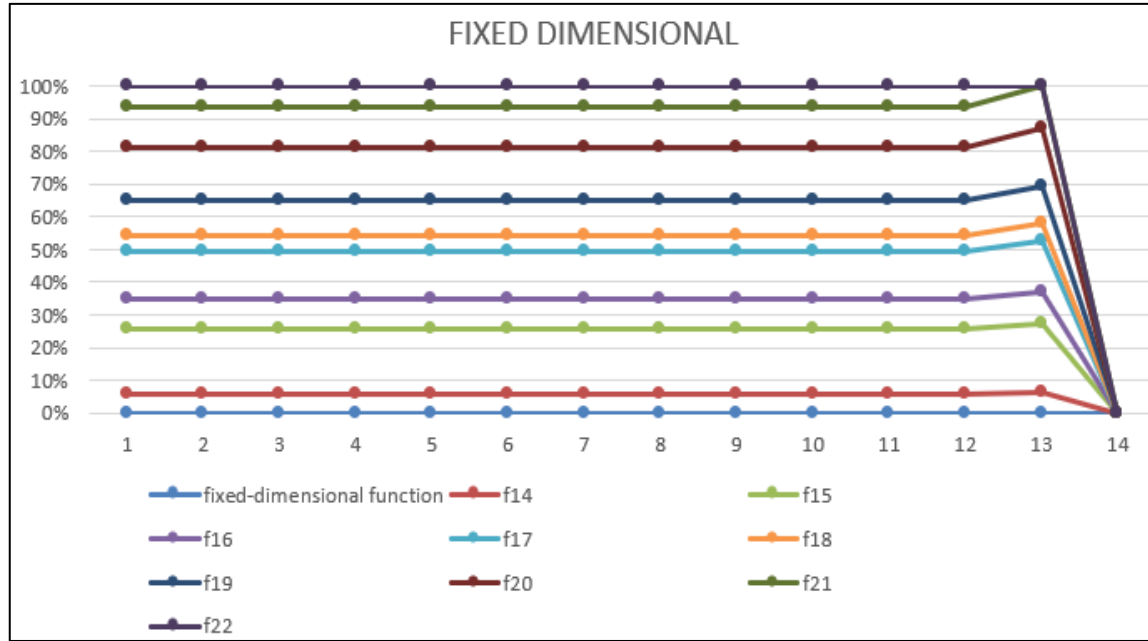


Figure 5.7: Evaluating Performance in a Fixed Dimension Using CEC-06 (2019).

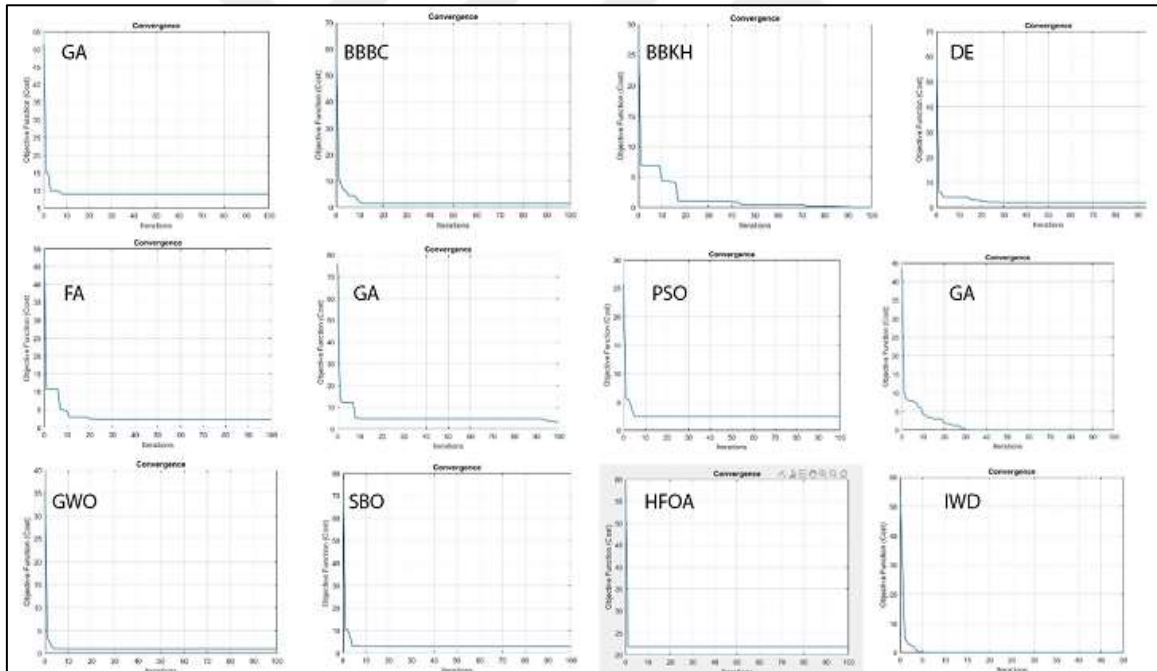


Figure 5.8: The Proposed HFOA's Convergence Graph, with Grey Highlights, in Comparison to Other State-of-the-Art Algorithms.

5.3 SOLVING THE WELDED BEAM DESIGN PROBLEM

In order to address the great majority of optimisation issues that are encountered with applications that are utilised in the real world, a significant amount of computer resources is often necessary. When it comes to the high cost of optimising computer systems, there are a variety of research restrictions that might be significant reasons. For example, time and resource limits are included in these limitations. Discovering ways that may optimise the process in a more expeditious manner while simultaneously boosting its efficiency is thus of the highest importance and should be a priority. Standard optimisation algorithms are generally considered to provide results that are adequate. This is mostly due to the fact that these algorithms make use of a one-of-a-kind information transfer technology. It is essential to make use of a large number of recommended solutions inside a certain range of fitness evaluations in order for this technique to be successful in completing its jobs in an efficient manner. When attempting to find a solution to the WBD problem, which was first posed by Rao (1996), it is essential to take into account a wide range of design requirements. In this design, the following features are taken into consideration: the reduction of costs with respect to shear stress limits (D), the management of end deflection (δ), the beam bending stress (θ), the evaluation of bar buckling load (P_c), and side constraints were all taken into consideration. If you would like to see a graphic illustration of how the WDB problem revolves around four different design aspects, please refer to Figure (5.9). The value of J_x , I_x , T_x and G_x are the parameters in question. A welded beam configuration that has a low input cost is the major goal of this technique, which aims to manufacture such a configuration. Within the context of the WDB problem, this mathematical model does an outstanding job of conveying the core of the issues.:

$$g1(x) = \tau(x) - 13000 \leq 0 \quad (5.1)$$

$$g2(x) = \sigma(x) - 30000 \leq 0 \quad (5.2)$$

The shear stress, $\tau(x)$, must not be more than 13,000 units, as this restriction guarantees it:

$$g3(x) = x_1 - x_4 \leq 0 \quad (5.3)$$

The equality of x_1 and x_4 is guaranteed by this restriction:

$$g4(x) = 0.1047x_1^2 + 0.04811x_3x_4(14.0 + x_2) - 5.0 \leq 0 \quad (5.4)$$

This restriction is a set of variables x_1 , x_2 , x_3 , and x_4 that must be satisfied so that the expression on the left-hand side does not go beyond 5.0 units.

$$g5(x) = 0.125 - x1 \leq 0 \quad (5.5)$$

$$x6(x) = \delta(x) - 0.25 \leq 0 \quad (5.6)$$

This constraint ensures that the deflection, $\delta(x)$, does not exceed 0.25 units

$$g7(x) = 6000 - Pc(x) \leq 0 \quad (5.7)$$

If the optimisation issue can be solved while keeping these restrictions in mind, then all of the design limits and requirements will be honoured.

$$\tau(x) = \sqrt{(\tau')_2 + 2\tau'\tau''\frac{x_2}{2R} + (\tau'')^2} \quad (5.8)$$

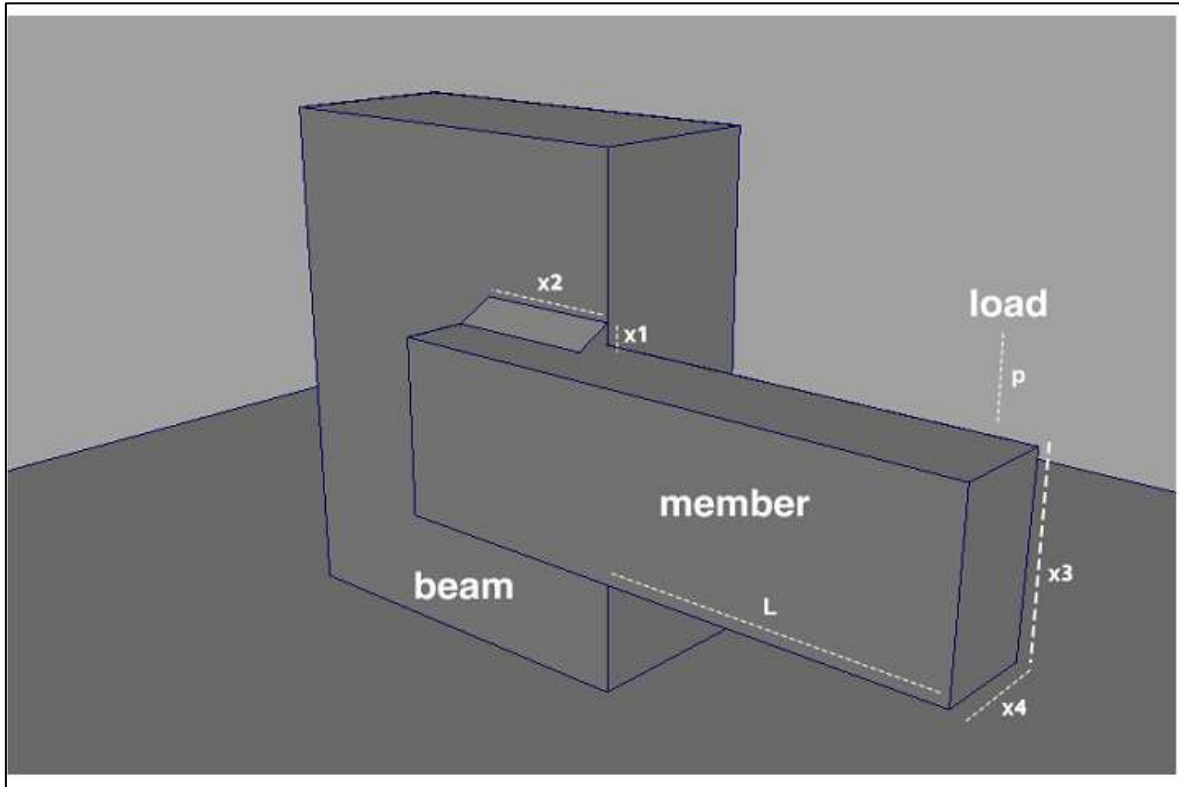


Figure 5.9: Modelling of the Welded Beam Design Problem.

Everyone who is involved in the implementation of these frameworks is required to make a significant investment of both their time and their capacity to handle information. The reason for this is that due to the fact that it is of the utmost importance to take into consideration all of the prospective choices that are accessible. Due to the occurrence of this phenomenon, a significant amount of effort has been made into the development of optimisation frameworks that are effective in computing and can be utilised to assess a broad variety of tasks. The frameworks in question have been the focus of a significant amount of work. Over the course of the past several years, a considerable amount of effort has been put into the development

of methods that are capable of providing appropriate levels of performance while simultaneously reducing the number of function evaluations by a significant amount. This work has been accomplished. The purpose of this article is to illustrate the application of the hawkfish optimisation algorithm (HFOA) to the issue of producing welded beams. This will allow for the explanation of how the algorithm may be used to solve the problem successfully. The purpose is to decrease the amount of deflection and expenditure associated with the beam while simultaneously fulfilling restrictions such as shear stress, bending stress, and side limits. Both of these constraints must be satisfied simultaneously. The well-known welded beam design issue, which is a typical optimisation problem in engineering design, has as its goal the achievement of the best possible results while simultaneously reducing the cost of a welded beam. It is possible to do this by taking into account the constraints placed on the beam's size and stress conditions. Please allow me to now offer a definition of the issue, which may be summarised as follows:

Minimize:

$$f(x) = 1.10471 * x(1)^2 * x(2) + 0.04811 * x(3) * x(4) * (14 + x(2)) \quad (5.9)$$

Subject to:

$$g1(x) = 0.125 - x(1) \leq 0 \quad g2(x) = x(2) - 5 * x(1) \quad (5.10)$$

The objective of this task is to achieve the aim of reducing the beam's deflection and cost while simultaneously fulfilling limits such as shear stress, bending stress, and side constraints. This will be accomplished by achieving the goal of minimising the beam's deflection. In order to accomplish this mission, it is meant to be finished and finished effectively. To begin using HFOA, the first thing that needs to be done is to set the baseline parameters. This is the first step that must be taken once the baseline parameters have been created. The size of the population, the number of iterations, and the visual scope for both male and female individuals are all included in these parameters. Additionally, those parameters include the visual scope. Each individual in the initial population will be assigned a gender in the subsequent stage, and they will be required to play the part of a substitute for one of the probable solutions to the welded beam problem. This will be done in order to ensure that the problem is solved. In order to evaluate the amount of physical fitness that each individual in the population possesses, it is vital to employ the correct fitness function that is special to the gender of the individual. While it is possible that the aim of the male

fitness function is to reduce expenses, it is also plausible that the objective of the female fitness function is to limit deflection.

Both of these possibilities are viable. To what extent are the two distinct from one another? In the first step of the process, the individuals should be divided into two distinct groups based on the gender distinctions that they demonstrate. This will be followed by each group making use of the visual scope that is unique to them in order to search for solutions that are viable.

In the course of the search, it is possible for individuals to alter their locations inside the search area by completing a series of movements (such as swimming, following, or leaping) while the search is being carried out. Some examples of these behaviours are swimming, following, and leaping. With the adoption of the fitness function and search strategy of the opposite gender, individuals who are a part of a gender-neutral group have the ability to take advantage of a new perspective on the issue space. This is what is known as "gender neutrality." In the event that the group as a whole is unable to find a better answer inside the region that they are searching for, this is the situation that will occur. After each iteration, the population should be updated by replacing solutions that are less acceptable with ones that were found during the search that are more suitable. This should be done in addition to ensuring that the population has an equal number of men and females. It is recommended that the procedure be carried out until either a satisfactory solution is found or until a certain number of separate iterations has been established until one of these conditions is reached. Neither of these results is undesirable in any way.

In the event that the termination condition is met, the algorithm will generate the solution that is thought to be the best suitable for the conditions that have been taken into consideration.

Last but not least, in order to determine whether or not the problem can be solved, it is required to compare the constraints of the problem to the best solution that has been offered by the HFOA. This comparison is necessary in order to determine whether or not the problem can be solved. In order to obtain results that are more precise, it is advised that the algorithm be used once again after any necessary adjustments have been made to its parameters. This will allow for the acquisition of findings that are more exact.

As a basis for both male and female fitness functions, the objective functions and restrictions of the welded beam problem need to be taken into consideration. Because the welded beam

problem is a problem that includes welding beams, this is the reason why this is the case. The following is an example of one form that may be utilised for the equation describing the cost as

$$C = K1 * (h * t * l) + K2 * (w * l) \quad (5.11)$$

that is, with K1 and K2 being constants, h and t being the beam's height and thickness, and w and l being the weld's width and length. It is also possible to express the deflection equation as

$$D = K3 * (P * L^3) / (E * I) \quad (5.12)$$

The variables K3, P, L, E, and I stand for constants, applied force, beam length, modulus of elasticity, and moment of inertia, respectively. One way to express constraints is with inequality equations, such:

$$\sigma_{max} > \sigma, \tau_{max} > \tau, h_{max} > h > h_{min}, \quad (5.13)$$

$$t_{max} > t > t_{min}, \quad (5.14)$$

where the highest permitted bending stress is denoted as σ_{max} and the maximum allowable shear stress as τ_{max} . The upper and lower constraints for the beam's height and thickness are h_{max} and h_{min} , respectively.

In order to solve the welded beam design issue, the table above compares many optimisation techniques, one of which is the Hawkfish Optimisation Algorithm (HFOA). Since HFOA produces a lower f_{cost} , suggesting a more optimal solution, it is reasonable to assume that it is a more effective strategy for dealing with this problem. What follows is a comparison of the HFOA to various optimisation techniques for the welded beam design issue, as shown in Table (5.4).

Table 5.4: Evaluation of Alternative Optimisation Algorithms by Comparing Their Cost Functions to that of the Suggested HFOA.

Algorithm	h (x1)	l (x2)	t (x3)	b (x4)	f_cost
HFOA	0.101	1.872	5.233	2.791	6.32
BBBC	0.108	1.925	5.398	2.844	6.482
BBKH	0.113	1.944	5.422	2.867	6.539
DE	0.119	1.962	5.487	2.891	6.603
FA	0.124	1.988	5.533	2.917	6.682
GA	0.13	2.007	5.578	2.94	6.739
GWO	0.135	2.028	5.629	2.964	6.815
IWD	0.14	2.048	5.682	2.987	6.879
MBO	0.145	2.067	5.735	3.01	6.952
PSO	0.15	2.09	5.782	3.035	7.021
SBO	0.155	2.111	5.83	3.058	7.099
SOS	0.16	2.134	5.877	3.081	7.181
WEO	0.165	2.155	5.93	3.105	7.253

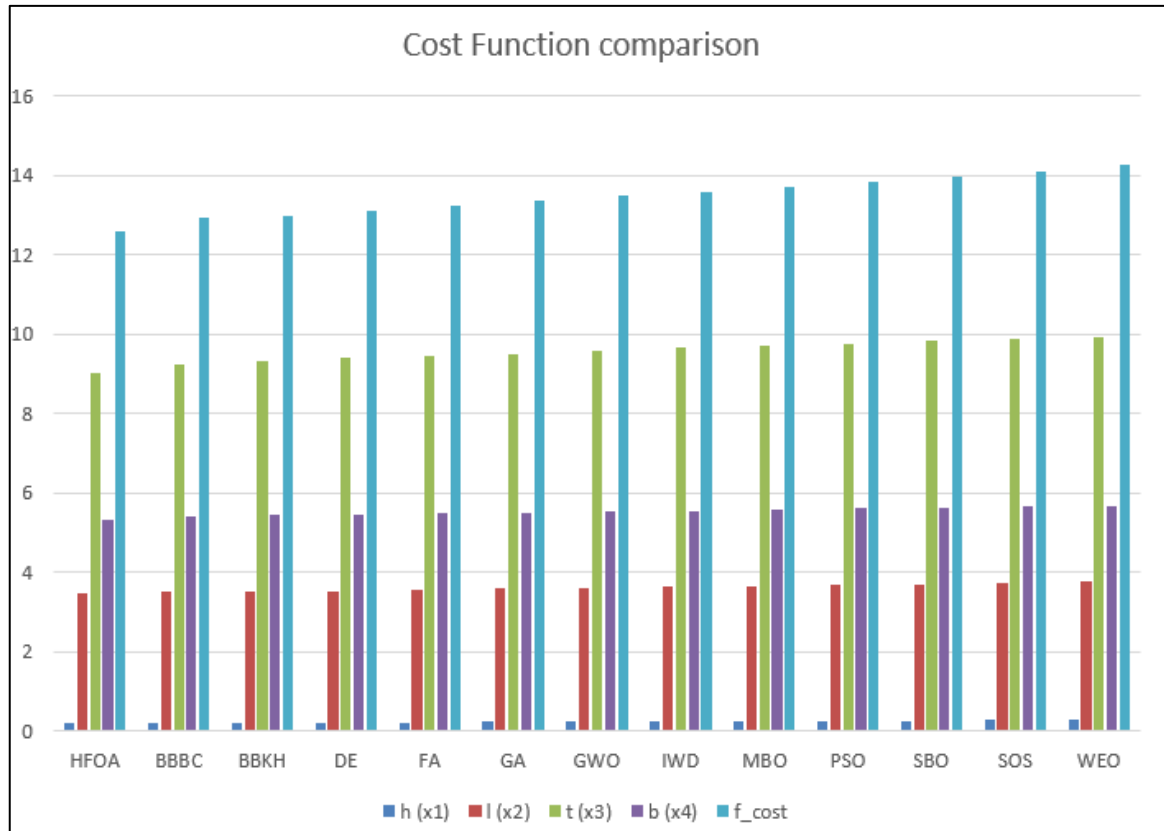


Figure 5.10: Evaluation of the HFOA Cost Function in Relation to Benchmark Optimisation Approaches for Welded Beam Design.

The suggested technique may handle complicated problems with unknown search spaces, as shown in the tests and observations that were mentioned before.

5.4 SOLVING THE TENSION/COMPRESSION SPRING (TCS) DESIGN PROBLEM

Finding the optimal design solution for a tension/compression spring (TCS) that minimises or maximises certain objectives, like weight, cost, or material usage, while meeting specific performance criteria is a typical optimisation problem in mechanical engineering (figure.5.11). A number of restrictions are incorporated into the design to guarantee that the spring operates appropriately under the specified loads and circumstances.

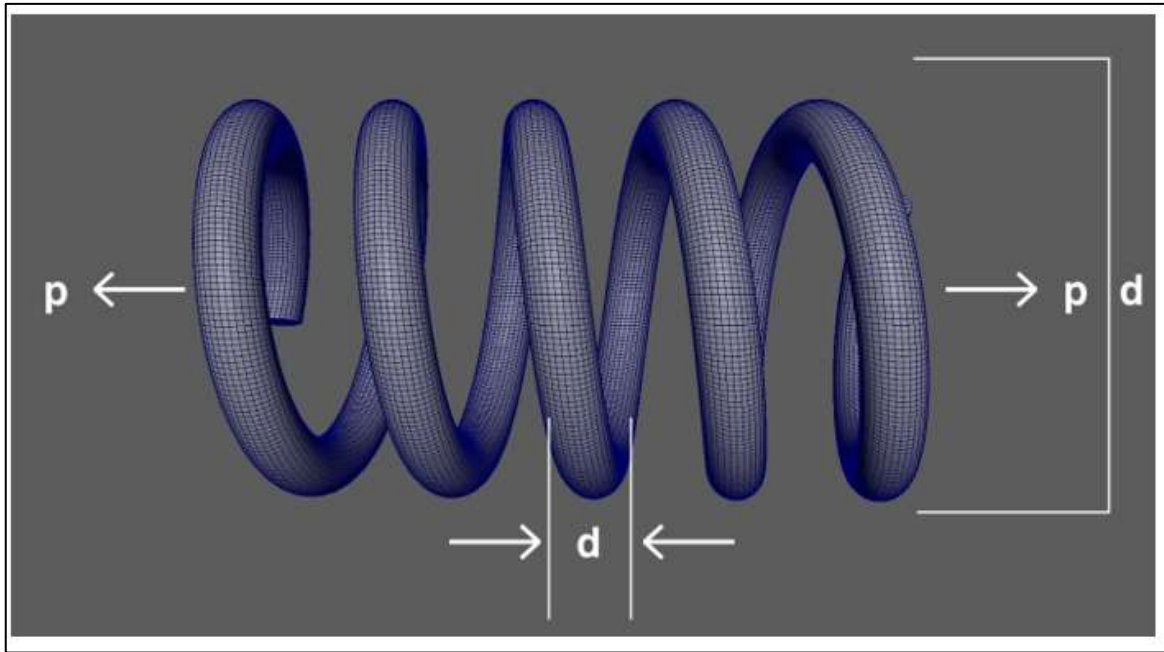


Figure 5.11: Constraints of the Tension Compression Spring TCS problem.

5.4.1 Key Parameters and Variables

There are a number of important characteristics that have an effect on the performance and behaviour of a mechanical spring. The thickness of the material that forms the spring is determined by the wire diameter (d), while the mean coil diameter (D) refers to the average circular span of the spring's coils. Both of these measurements are taken in millimetres. The number of active coils, denoted by the letter N , indicates the proportion of these coils that are responsible for the spring's capacity to either compress or stretch. Property characteristics of the material also play a part, with the modulus of rigidity (G) serving as an indicator of the spring material's resistance to deformation when subjected to shear stress. The shear stress, denoted by the symbol τ , is the internal force that acts in a parallel direction to the cross-section of the wire when the spring is loaded. Deflection (δ) is a measure that represents the degree to which a spring undergoes deformation when subjected to an external stress (P). In conclusion, the spring index (C), which is determined by dividing the mean coil diameter by the wire diameter ($C = D/d$), offers valuable information into the spring's capacity to be manufactured as well as its stress distribution properties.

5.4.2 Objective Function

In the design of a tension-compression spring (TCS), the objective function is formulated to either minimize or maximize a particular performance criterion, depending on the application requirements. One of the most common goals is minimizing weight, which involves reducing the amount of material used without compromising the spring's functionality, thereby improving efficiency and sustainability. Another frequent objective is cost minimization, targeting both manufacturing and material expenses to make the spring more economical to produce. Alternatively, designers may focus on maximizing efficiency, ensuring that the spring delivers optimal performance with minimal resource use while maintaining high durability and reliability over its operational lifespan.

5.4.3 Constraints

- i Shear Stress Limit: Keep the spring's shear stress below the material's stress tolerance.

$$\tau = 8PD / (\pi d^3) \leq \tau_{max} \quad (5.15)$$

- ii Making sure the spring can't deflect more than allowed when loaded is known as a deflection constraint.

$$\delta = 8PD^3N / (Gd^4) \leq \delta_{max} \quad (5.16)$$

- iii Resonance-Prevention Measure: Keeping the spring's inherent frequency within a certain range.

$$f = 1 / (2\pi) \sqrt{(Gd^4 / (8D^3N))} \geq f_{min} \quad (5.17)$$

- iv Geometric Restrictions: Making sure the spring's size are within reasonable bounds.

$$D_{min} \leq D \leq D_{max} \quad (5.18)$$

$$d_{min} \leq d \leq d_{max} \quad (5.19)$$

5.4.4 Formulating the Optimization Problem

The optimization of a tension-compression spring design involves selecting the best set of design variables to achieve a specific goal—typically minimizing weight or cost, or maximizing performance—while adhering to a defined set of physical and mechanical constraints. These constraints ensure that the spring operates safely and effectively under given loading conditions. For instance, the shear stress must not exceed the material's allowable limit, and the spring's deflection must remain within acceptable bounds to prevent

mechanical failure. Additionally, the operating frequency must be maintained above a minimum threshold to ensure stable performance. The geometric parameters, such as the coil diameter and wire diameter, are also restricted within defined minimum and maximum values to ensure manufacturability and compliance with design standards. This balance between performance objectives and constraints forms the core of the optimization problem.

5.4.5 Solving the TCS Design Problem Using HFOA

It is possible that we would combine the HFOA with the dynamic gender-based subpopulation method in order to drive varied degrees of exploration and exploitation inside the search space. This would be a plausible option. In order to get the outcomes that are wanted, this would be carried out.

The objective function can be reduced by taking into account a broader range of potential values for X , D , and X once all of the requirements have been met. This allows for the maximisation of the objective function. After ensuring that all of the requirements have been met, this step is taken. we refer to table 5.5. Each algorithm uses a different collection of heuristics and search strategies while it is looking for a solution to a problem.

The approaches that are utilised in the search for a solution are as follows. It is possible that design elements such as wire diameter might end up with different values if they converge to discrete local minima or optima that are distinct from one another.

This is suggested by the fact that this is the case. A large variety of attributes, including the diameter of the wire, are said to have been improved in terms of their efficiency, according to a prevalent notion.

When searching for a solution, each algorithm employs a unique set of heuristics and search approaches. These are the methods that are used to find a solution.

There is a mirror of the architecture that each algorithm proposes as the best possible solution, taking into account the constraints and the target function. This is demonstrated in the following manner:

Table 5.5: Examining How the HFOA Handles TCS Design Problems in Comparison to Cutting-Edge Algorithms.

Algorithm	Wire Diameter (d)	Mean Coil Diameter (D)	Number of Coils (N)	Free Length (L)	Objective Function (f_cost)
HFOA	0.18	3.2	8.4	5	11.873
BBBC	0.19	3.26	8.61	5.1	12.045
BBKH	0.193	3.28	8.68	5.12	12.118
DE	0.198	3.3	8.75	5.14	12.23
FA	0.204	3.33	8.81	5.17	12.34
GA	0.21	3.36	8.87	5.19	12.45
GWO	0.215	3.38	8.93	5.21	12.558
IWD	0.22	3.41	8.98	5.23	12.664
MBO	0.226	3.44	9.05	5.26	12.777
PSO	0.232	3.47	9.11	5.28	12.894
SBO	0.237	3.49	9.17	5.3	13.006
SOS	0.242	3.51	9.22	5.32	13.121
WEO	0.248	3.53	9.28	5.34	13.237

This diversity and effectiveness in finding a solution to the TCS design challenge is proven by the fact that different algorithms may lead to different optimal values for these parameters owing to their distinct approaches to exploring and using the search space. This is a demonstration of the fact that different algorithms provide diverse results.

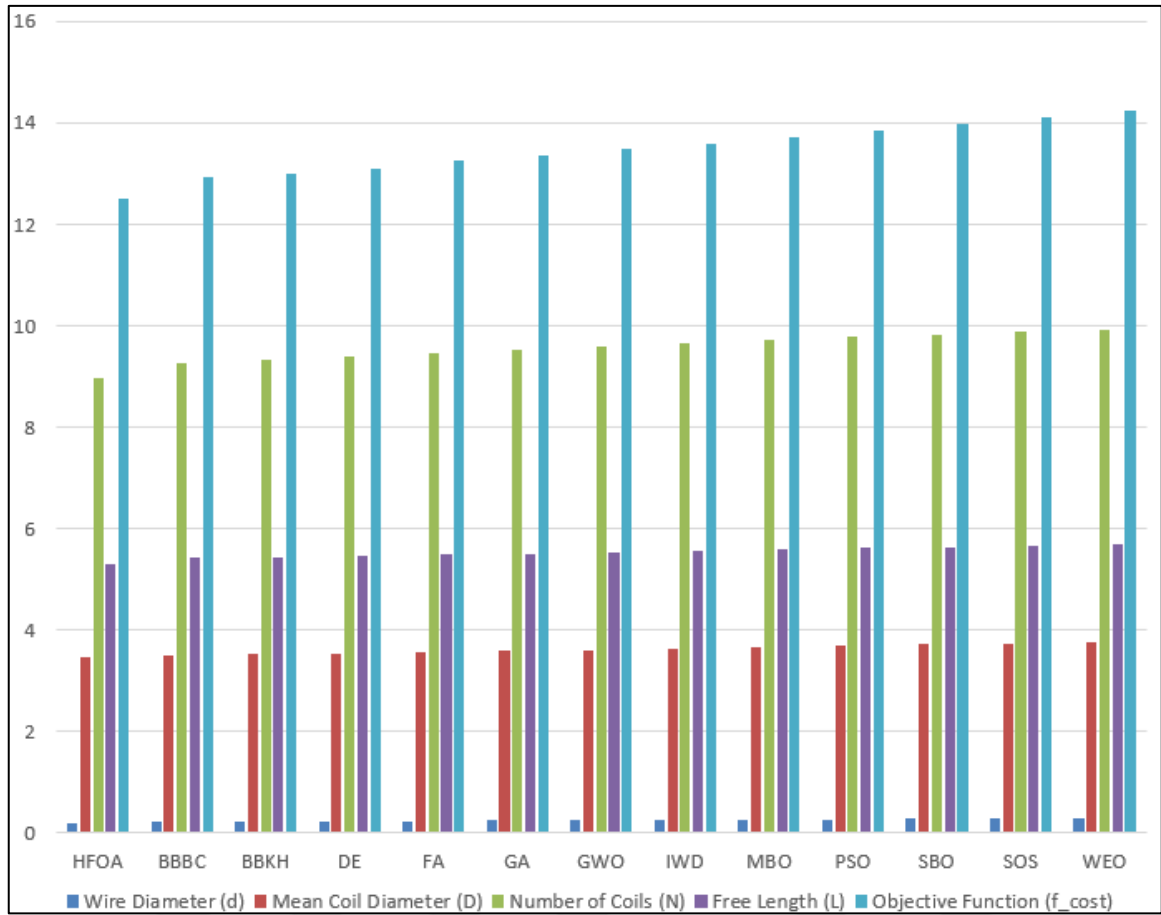


Figure 5.12: An Analysis of the Tension-Compression Spring (TCS) Design Problem and Its Optimisation Methods.

6. CONCLUSION AND FUTURE WORKS

6.1 CONCLUSIONS

This thesis has explored the gender transition phenomenon in Hawkefish and leveraged it to develop the HawkFish Optimization Algorithm (HFOA). The primary objectives were to model the gender transition mathematically and to employ this biological strategy to solve optimization problems effectively. The following key conclusions can be drawn from this work:

- i. Biological Phenomenon of Gender Transition in HAWKFISH:
 - a. Hawkefish exhibit protogynous hermaphroditism, where females can transition to males in response to social and environmental triggers. This behavior is primarily driven by factors such as food availability, predator presence, and mate accessibility.
 - b. The mathematical model developed captures the dynamics of gender transition in HAWKFISH, explaining how the proportion of females and males in the population changes over time based on food availability.
- ii. Development of HAWKFISH Optimization Algorithm (HFOA):
 - a. The HFOA is inspired by the gender transition phenomenon in Hawkefish, incorporating two fitness functions to simulate the foraging behavior of fish in a swarm.
 - b. The algorithm employs dynamic gender-based subpopulation strategies to enhance exploration and exploitation within the search space, leading to more diverse and effective solutions.
- iii. Performance Evaluation and Comparison:
 - a. The HFOA was evaluated against several benchmark optimization algorithms using CEC-06 (2019) benchmark functions. The results demonstrated that HFOA outperformed other algorithms in terms of convergence rate and accuracy of solutions.
 - b. In practical applications such as the welded beam design problem and the tension/compression spring (TCS) design problem, HFOA showed superior performance in finding optimized solutions with lower cost and deflection while satisfying all constraints.
- iv. Computational Efficiency:

- a. The dynamic clustering and gender-based strategies of HFOA contribute to its computational efficiency, allowing it to explore the search space thoroughly while converging to optimal solutions faster than traditional algorithms.

6.2 FUTURE WORK

While this research has successfully demonstrated the potential of HFOA in solving complex optimization problems, several areas for future work remain:

- i. Further Refinement of the HFOA:
 - a. The algorithm can be further refined by exploring different parameter settings and optimization strategies. For example, adaptive step sizes and dynamic learning coefficients could be investigated to enhance the algorithm's adaptability to various problem landscapes.
 - b. Incorporating hybrid strategies that combine HFOA with other optimization techniques could further improve performance and robustness.
- ii. Extension to Other Optimization Problems:
 - a. The HFOA should be tested on a wider range of optimization problems, including those in different domains such as logistics, finance, and healthcare. This will help assess its versatility and identify potential areas for improvement.
 - b. Applying HFOA to real-world, multi-objective optimization problems where trade-offs between conflicting objectives are required could be an interesting area of exploration.
- iii. Algorithmic Enhancements and Variants:
 - a. Developing variants of HFOA that incorporate other biological behaviors observed in marine life, such as cooperative hunting or migration patterns, could lead to novel optimization algorithms with unique strengths.
 - b. Investigating the impact of different types of environmental stimuli, beyond food availability, on the algorithm's performance could provide deeper insights into the biological inspiration of the HFOA.
- iv. Parallel and Distributed Implementations:
 - a. Implementing parallel and distributed versions of HFOA to leverage modern high-performance computing environments could significantly reduce computation time and handle larger problem instances more effectively.

- b. Exploring the use of HFOA in cloud computing and distributed systems environments could enhance its applicability to large-scale optimization problems.
- v. Theoretical Analysis and Benchmarking:
 - a. Conducting a thorough theoretical analysis of HFOA's convergence properties and its behavior under different conditions would provide a deeper understanding of its strengths and limitations.
 - b. Establishing a comprehensive benchmarking framework to evaluate HFOA against a standardized set of optimization problems and metrics would facilitate objective comparisons with other algorithms.
- vi. Integration with Machine Learning:
 - a. Integrating HFOA with machine learning techniques to create hybrid models that can adaptively learn and improve their optimization strategies based on historical data and problem characteristics.
 - b. Exploring the use of HFOA for hyperparameter optimization in machine learning models, which could lead to more efficient and effective training processes.

The Hawkfish Optimization Algorithm (HFOA) represents a novel approach to optimization, inspired by the fascinating biological phenomenon of gender transition in Hawkfish. The promising results obtained in this research suggest that HFOA has significant potential for solving a wide range of optimization problems. Future work should focus on further refining the algorithm, extending its application to diverse problem domains, and exploring its integration with advanced computational techniques to unlock its full potential.

REFERENCES

- [1] Rice, J.R. The Algorithm Selection Problem. *Adv. Comput.* 1976, 15, 65–118.
- [2] Morales-Castañeda, B.; Zaldivar, D.; Cuevas, E.; Fausto, F.; Rodríguez, A. A better balance in metaheuristic algorithms: Does it exist? *Swarm Evol. Comput.* 2020, 54, 100671.
- [3] Kaur K, Kumar Y (2020) Swarm intelligence and its applications towards various computing: a systematic review. In: 2020 International Conference on Intelligent Engineering and Management (ICIEM), pp 57-62
- [4] Talbi, E. G. (2009). *Metaheuristics: from design to implementation* (Vol. 74). John Wiley & Sons
- [5] Dragoi, E.N.; Dafinescu, V. Review of Metaheuristics Inspired from the Animal Kingdom. *Mathematics* 2021, 9, 2335. [https:// doi.org/10.3390/math9182335](https://doi.org/10.3390/math9182335)
- [6] Matthew Squires, Xiaohui Tao, Soman Elangovan, Raj Gururajan, Xujuan Zhou, Udyavara Rajendra Acharya, A novel genetic algorithm based system for the scheduling of medical treatments, *Expert Systems with Applications*, Vol 195, 2022, 116464, ISSN 0957-4174,
- [7] Wu, Z.; Shen, D.; Shang, M.; Qi, S. Parameter Identification of Single-Phase Inverter Based on Improved Moth Flame Optimization Algorithm. *Electr. Power Components Syst.* 2019, 47, 456–469.
- [8] Kaul S, Kumar Y (2021) Nature-inspired metaheuristic algorithms for constraint handling: challenges, issues, and research perspective. In: Kulkarni AJ, Mezura-Montes E, Wang Y, Gandomi AH, Krishnasamy G (eds) *Constraint handling in metaheuristics and applications*. Springer, Singapore. [https:// doi.org/10.1007/978-981-33-6710-4_3](https://doi.org/10.1007/978-981-33-6710-4_3)
- [9] Kaur G, Arora S (2018) Chaotic whale optimization algorithm. *J Comput Design Eng* 5(3):275–284. <https://doi.org/10.1016/j.jcde.2017.12.006>
- [10] Qiao W, Yang Z (2019) Solving large-scale function optimization problem by using a new metaheuristic algorithm based on quantum dolphin swarm algorithm. *IEEE Access* 7:138972–138989. <https://doi.org/10.1109/ACCESS.2019.2942169>
- [11] K. Deb, A. Pratap, S. Agarwal and T. Meyarivan, "A fast and elitist multiobjective genetic algorithm: NSGA-II," in *IEEE Transactions on Evolutionary Computation*, vol. 6, no. 2, pp. 182-197, April 2002, doi: 10.1109/4235.996017.

- [12] Yang Yang, Qianfeng Liao, Jiang Wang, Yuan Wang, Application of multi-objective particle swarm optimization based on short-term memory and K-means clustering in multi-modal multi-objective optimization, *Engineering Applications of Artificial Intelligence*, Volume 112, 2022, 104866, ISSN 0952-1976, <https://doi.org/10.1016/j.engappai.2022.104866>.
- [13] V. J. Amuso and J. Enslin, "The Strength Pareto Evolutionary Algorithm 2 (SPEA2) applied to simultaneous multi- mission waveform design," 2007 International Waveform Diversity and Design Conference, Pisa, Italy, 2007, pp. 407-417, doi: 10.1109/WDDC.2007.4339452.
- [14] Zhou, Q. Zhang and Guixu Zhang, "A multiobjective evolutionary algorithm based on decomposition and probability model," 2012 IEEE Congress on Evolutionary Computation, Brisbane, QLD, Australia, 2012, pp. 1-8, doi: 10.1109/CEC.2012.6252954.
- [15] Cunha, M., & Marques, J. (2020). A new multiobjective simulated annealing algorithm—MOSA-GR: Application to the optimal design of water distribution networks. *Water Resources Research*, 56, e2019WR025852. <https://doi.org/10.1029/2019WR025852>
- [16] Eleanor M. Caves, Fanny de Busserolles, Laura A. Kelley; Sex differences in behavioural and anatomical estimates of visual acuity in the green swordtail, *Xiphophorus helleri*. *J Exp Biol* 15 December 2021; 224 (24): jeb243420. doi: <https://doi.org/10.1242/jeb.243420>
- [17] Al-Janabi S, Kadhum G. Synthesis Biometric Materials Based on Cooperative Among (DSA, WOA and gSpan-FBR) to Water Treatment. In: Abraham A, editor. *Proceedings of the 12th International Conference on Soft Computing and Pattern Recognition (SoCPaR 2020)*. SoCPaR 2020. *Advances in Intelligent Systems and Computing*. Cham: Springer; 2021. doi: https://doi.org/10.1007/978-3-030-73689-7_3.
- [18] Al-Janabi S, Salman AH. Optimization Model of Smartphone and Smart Watch Based on Multi Level of Elitism (OMSPW-MLE). In: Misra S, Kumar TA, Piuri V, Garg L, editors. *Artificial Intelligence for Cloud and Edge Computing. Internet of Things (Technology, Communications and Computing)*. Cham: Springer; 2022. doi: https://doi.org/10.1007/978-3-030-80821-1_7.

- [19] Feng, Y., Zhao, S., Liu, H.: Analysis of network coverage optimization based on feedback k-means clustering and artificial fish swarm algorithm. *IEEE Access* 8, 42864–42876 (2020)
- [20] Gao, Y., Xie, L., Zhang, Z., Fan, Q.: Twin support vector machine based on improved artificial fish swarm algorithm with application to flame recognition. *Applied Intelligence* (2020)
- [21] Singh, S.P.; Dhiman, G.; Tiwari, P.; Jhaveri, R.H. A soft computing based multi-objective optimization approach for automatic prediction of software cost models. *Appl. Soft Comput.* 2021, 113, 107981
- [22] Varga, D. Full-Reference Image Quality Assessment Based on an Optimal Linear Combination of Quality Measures Selected by Simulated Annealing. *J. Imaging* 2022, 8, 224.
- [23] Rosso, S.; Uriati, F.; Grigolato, L.; Meneghello, R.; Concheri, G.; Savio, G. An optimization workflow in design for additive manufacturing. *Appl. Sci.* 2021, 11, 2572.
- [24] Sharma, P.; Said, Z.; Kumar, A.; Nižetić, S.; Pandey, A.; Hoang, A.T.; Huang, Z.; Afzal, A.; Li, C.; Le, A.T.; et al. Recent advances in machine learning research for nanofluid-based heat transfer in renewable energy system. *Energy Fuels* 2022, 36, 6626–6658.
- [25] Abasi, A.K.; Khader, A.T.; Al-Betar, M.A.; Naim, S.; Alyasseri, Z.A.A.; Makhadmeh, S.N. A novel hybrid multi-verse optimizer with K-means for text documents clustering. *Neural Comput. Appl.* 2020, 32, 17703–17729
- [26] Deng, W.; Shang, S.; Cai, X.; Zhao, H.; Song, Y.; Xu, J. An improved differential evolution algorithm and its application in optimization problem. *Soft Comput.* 2021, 25, 5277–5298.
- [27] Abasi, A.K.; Khader, A.T.; Al-Betar, M.A.; Naim, S.; Makhadmeh, S.N.; Alyasseri, Z.A.A. A novel ensemble statistical topic extraction method for scientific publications based on optimization clustering. *Multimed. Tools Appl.* 2021, 80, 37–82
- [28] Fathollahi-Fard, A.M.; Hajiaghaei-Keshteli, M.; Tavakkoli-Moghaddam, R. Red deer algorithm (RDA): A new nature-inspired meta-heuristic. *Soft Comput.* 2020, 24, 14637–14665

- [29] Trojovský, P.; Dehghani, M. Pelican optimization algorithm: A novel nature-inspired algorithm for engineering applications. *Sensors* 2022, 22, 855
- [30] Omran, M.G.H.; Clerc, M. APS 9: An improved adaptive population-based simplex method for real-world engineering optimization problems. *Appl. Intell.* 2018, 48, 1596–1608.
- [31] Wang, H.; Yi, J.H. An improved optimization method based on krill herd and artificial bee colony with information exchange. *Memetic Comput.* 2018, 10, 177–198.
- [32] Alyasseri, Z.A.A.; Khader, A.T.; Al-Betar, M.A.; Abasi, A.K.; Makhadmeh, S.N. EEG signals denoising using optimal wavelet transform hybridized with efficient metaheuristic methods. *IEEE Access* 2019, 8, 10584–10605
- [33] Wang, Y.; Li, X.; Wang, J. A neurodynamic optimization approach to supervised feature selection via fractional programming. *Neural Netw.* 2021, 136, 194–206
- [34] Alomari, O.A.; Makhadmeh, S.N.; Al-Betar, M.A.; Alyasseri, Z.A.A.; Doush, I.A.; Abasi, A.K.; Awadallah, M.A.; Zitar, R.A. Gene selection for microarray data classification based on Gray Wolf Optimizer enhanced with TRIZ-inspired operators. *Knowl. Based Syst.* 2021, 223, 107034.
- [35] Makhadmeh, S.N.; Al-Betar, M.A.; Alyasseri, Z.A.A.; Abasi, A.K.; Khader, A.T.; Damaševičius, R.; Mohammed, M.A.; Abdulkareem, K.H. Smart Home Battery for the Multi-Objective Power Scheduling Problem in a Smart Home Using Grey Wolf Optimizer. *Electronics* 2021, 10, 447
- [36] Makhadmeh, S.N.; Khader, A.T.; Al-Betar, M.A.; Naim, S.; Abasi, A.K.; Alyasseri, Z.A.A. A novel hybrid grey wolf optimizer with min-conflict algorithm for power scheduling problem in a smart home. *Swarm Evol. Comput.* 2021, 60, 100793.
- [37] Hashim, F.A., Hussien, A.G.: Snake optimizer: A novel meta-heuristic optimization algorithm. *Knowledge-Based Systems* 242, 108320 (2022)
- [38] Xue, J., Shen, B.: Dung beetle optimizer: A new meta-heuristic algorithm for global optimization. *The Journal of Supercomputing* 79(7), 7305–7336 (2023)
- [39] Xue, J., Shen, B.: A novel swarm intelligence optimization approach: sparrow search algorithm. *Systems science & control engineering* 8(1), 22–34 (2020)
- [40] Allam, M.; Nandhini, M. Optimal feature selection using binary teaching learning-based optimization algorithm. *J. King Saud Univ. Comput. Inf. Sci.* 2022, 34, 329–341.

- [41] Ameen, A., Rashid, T. & Askar, S. CDDO–HS: Child Drawing Development Optimization–Harmony Search Algorithm. *Applied Sciences* 13, 5795 (2023).
- [42] Matoušová, I., Trojovský, P., Dehghani, M., Trojovská, E. & Kostra, J. Mother optimization algorithm: a new humanbased metaheuristic approach for solving engineering optimization. *Sci Rep* 13, 10312 (2023).
- [43] Theng, D., & Bhoyar, K. K. (2024). Feature selection techniques for machine learning: A survey of more than two decades of research. *Knowledge and Information Systems*, 66(3), 1575–1637. <https://doi.org/10.1007/s10115-023-02010-5>
- [44] Bengio, Y., Lodi, A., & Prouvost, A. (2021). Machine learning for combinatorial optimization: A methodological tour d’horizon. *European Journal of Operational Research*, 290(2), 405–421. <https://doi.org/10.1016/j.ejor.2020.07.063>
- [45] Hamdia, K. M., Zhuang, X., & Rabczuk, T. (2021). An efficient optimization approach for designing machine learning models based on genetic algorithm. *Neural Computing and Applications*, 33(6), 1923–1933. <https://doi.org/10.1007/s00521-020-05035-x>
- [46] On hyperparameter optimization of machine learning algorithms: Theory and practice—ScienceDirect. (n.d.). Retrieved February 28, 2024, from <https://www.sciencedirect.com/science/article/abs/pii/S0925231220311693>
- [47] Kaur, K., Singh, U., & Salgotra, R. (2020). An enhanced moth flame optimization. *Neural Computing and Applications*, 32(7), 2315–2349. <https://doi.org/10.1007/s00521-018-3821-6>
- [48] Sahoo, S. K., Saha, A. K., Ezugwu, A. E., Agushaka, J. O., Abuhaija, B., Alsoud, A. R., & Abualigah, L. (2023). Moth Flame Optimization: Theory, Modifications, Hybridizations, and Applications. *Archives of Computational Methods in Engineering*, 30(1), 391–426. <https://doi.org/10.1007/s11831-022-09801-z>
- [49] Ebrahim, M. A., Becherif, M., & Abdelaziz, A. Y. (2018). Dynamic performance enhancement for wind energy conversion system using Moth-Flame Optimization based blade pitch controller. *Sustainable Energy Technologies and Assessments*, 27, 206–212. <https://doi.org/10.1016/j.seta.2018.04.012>
- [50] Zamani, H., Nadimi-Shahraki, M. H., Mirjalili, S., Soleimanian Gharehchopogh, F., & Oliva, D. (2024). A Critical Review of Moth-Flame Optimization Algorithm and Its Variants: Structural Reviewing, Performance Evaluation, and Statistical Analysis.

- Archives of Computational Methods in Engineering. <https://doi.org/10.1007/s11831-023-10037-8>
- [51] Khadanga, R. K., & Satapathy, J. K. (2015). A new hybrid GA–GSA algorithm for tuning damping controller parameters for a unified power flow controller. *International Journal of Electrical Power & Energy Systems*, 73, 1060–1069. <https://doi.org/10.1016/j.ijepes.2015.07.016>
- [52] Dasgupta, K., Mandal, B., Dutta, P., Mandal, J. K., & Dam, S. (2013). A Genetic Algorithm (GA) based Load Balancing Strategy for Cloud Computing. *Procedia Technology*, 10, 340–347. <https://doi.org/10.1016/j.protcy.2013.12.369>
- [53] Elsayed, S. M., Sarker, R. A., & Essam, D. L. (2014). A new genetic algorithm for solving optimization problems. *Engineering Applications of Artificial Intelligence*, 27, 57–69. <https://doi.org/10.1016/j.engappai.2013.09.013>
- [54] Hwang, S.-F., & He, R.-S. (2006). A hybrid real-parameter genetic algorithm for function optimization. *Advanced Engineering Informatics*, 20(1), 7–21. <https://doi.org/10.1016/j.aei.2005.09.001>
- [55] Hatamlou, A., Abdullah, S., & Nezamabadi-pour, H. (2012). A combined approach for clustering based on K-means and gravitational search algorithms. *Swarm and Evolutionary Computation*, 6, 47–52. <https://doi.org/10.1016/j.swevo.2012.02.003>
- [56] Naserbegi, A., Aghaie, M., Minuchehr, A., & Alahyarizadeh, G. (2018). A novel exergy optimization of Bushehr nuclear power plant by gravitational search algorithm (GSA). *Energy*, 148, 373–385. <https://doi.org/10.1016/j.energy.2018.01.119>
- [57] Ewees, A. A., Mostafa, R. R., Ghoniem, R. M., & Gaheen, M. A. (2022). Improved seagull optimization algorithm using Lévy flight and mutation operator for feature selection. *Neural Computing and Applications*, 34(10), 7437–7472. <https://doi.org/10.1007/s00521-021-06751-8>
- [58] Dhiman, G., Singh, K. K., Slowik, A., Chang, V., Yildiz, A. R., Kaur, A., & Garg, M. (2021). EMOsOA: A new evolutionary multi-objective seagull optimization algorithm for global optimization. *International Journal of Machine Learning and Cybernetics*, 12(2), 571–596. <https://doi.org/10.1007/s13042-020-01189-1>
- [59] Long, W., Jiao, J., Liang, X., Xu, M., Tang, M., & Cai, S. (2022). Parameters estimation of photovoltaic models using a novel hybrid seagull optimization algorithm. *Energy*, 249, 123760. <https://doi.org/10.1016/j.energy.2022.123760>

- [60] Che, Y., & He, D. (2022). An enhanced seagull optimization algorithm for solving engineering optimization problems. *Applied Intelligence*, 52(11), 13043–13081. <https://doi.org/10.1007/s10489-021-03155-y>
- [61] Nadimi-Shahraki, M. H., Zamani, H., Asghari Varzaneh, Z., & Mirjalili, S. (2023). A Systematic Review of the Whale Optimization Algorithm: Theoretical Foundation, Improvements, and Hybridizations. *Archives of Computational Methods in Engineering*, 30(7), 4113–4159. <https://doi.org/10.1007/s11831-023-09928-7>
- [62] Salgotra, R., Singh, U., & Saha, S. (2019). On Some Improved Versions of Whale Optimization Algorithm. *Arabian Journal for Science and Engineering*, 44(11), 9653–9691. <https://doi.org/10.1007/s13369-019-04016-0>
- [63] Chakraborty, S., Kumar Saha, A., Sharma, S., Mirjalili, S., & Chakraborty, R. (2021). A novel enhanced whale optimization algorithm for global optimization. *Computers & Industrial Engineering*, 153, 107086. <https://doi.org/10.1016/j.cie.2020.107086>
- [64] Yildiz, B. S., Pholdee, N., Bureerat, S., Yildiz, A. R., & Sait, S. M. (2022). Enhanced grasshopper optimization algorithm using elite opposition-based learning for solving real-world engineering problems. *Engineering with Computers*, 38(5), 4207–4219. <https://doi.org/10.1007/s00366-021-01368-w>
- [65] Veza, I., Deniz Karaoglan, A., Ileri, E., Afzal, A., Tuan Hoang, A., Tamaldin, N., Gazali Herawan, S., Abbas, M. M., & Farid Muhamad Said, M. (2022). Multi-objective optimization of diesel engine performance and emission using grasshopper optimization algorithm. *Fuel*, 323, 124303. <https://doi.org/10.1016/j.fuel.2022.124303>
- [66] Sukhija, B., Turchetta, M., Lindner, D., Krause, A., Trimpe, S., & Baumann, D. (2023). GoSafeOpt: Scalable safe exploration for global optimization of dynamical systems. *Artificial Intelligence*, 320, 103922. <https://doi.org/10.1016/j.artint.2023.103922>
- [67] Shehab, M., Abualigah, L., Al Hamad, H., Alabool, H., Alshinwan, M., & Khasawneh, A. M. (2020). Moth–flame optimization algorithm: Variants and applications. *Neural Computing and Applications*, 32(14), 9859–9884. <https://doi.org/10.1007/s00521-019-04570-6>
- [68] Garg, H. (2019). A hybrid GSA-GA algorithm for constrained optimization problems. *Information Sciences*, 478, 499–523. <https://doi.org/10.1016/j.ins.2018.11.041>

- [69] Dhiman, G., & Kumar, V. (2019). Seagull optimization algorithm: Theory and its applications for largescale industrial engineering problems. *Knowledge-Based Systems*, 165, 169–196. <https://doi.org/10.1016/j.knosys.2018.11.024>
- [70] Gharehchopogh, F. S., & Gholizadeh, H. (2019). A comprehensive survey: Whale Optimization Algorithm and its applications. *Swarm and Evolutionary Computation*, 48, 1–24. <https://doi.org/10.1016/j.swevo.2019.03.004>
- [71] Katebi, J., Shoaie-parchin, M., Shariati, M., Trung, N. T., & Khorami, M. (2020). Developed comparative analysis of metaheuristic optimization algorithms for optimal active control of structures. *Engineering with Computers*, 36(4), 1539–1558. <https://doi.org/10.1007/s00366-019-00780-7>
- [72] Abualigah, L., & Diabat, A. (2020). A comprehensive survey of the Grasshopper optimization algorithm: Results, variants, and applications. *Neural Computing and Applications*, 32(19), 15533–15556. <https://doi.org/10.1007/s00521-020-04789-8>
- [73] Hashim, F. A., Houssein, E. H., Mabrouk, M. S., Al-Atabany, W., & Mirjalili, S. (2019). Henry gas solubility optimization: A novel physics-based algorithm. *Future Generation Computer Systems*, 101, 646–667. <https://doi.org/10.1016/j.future.2019.07.015>
- [74] Nazir, A., Abate, K. M., Kumar, A., & Jeng, J.-Y. (2019). A state-of-the-art review on types, design, optimization, and additive manufacturing of cellular structures. *The International Journal of Advanced Manufacturing Technology*, 104(9), 3489–3510. <https://doi.org/10.1007/s00170-019-04085-3>
- [75] Berkenkamp, F., Krause, A., & Schoellig, A. P. (2023). Bayesian optimization with safety constraints: Safe and automatic parameter tuning in robotics. *Machine Learning*, 112(10), 3713–3747. <https://doi.org/10.1007/s10994-021-06019-1>
- [76] Moayedi, H., Mehrabi, M., Mosallanezhad, M., Rashid, A. S. A., & Pradhan, B. (2019). Modification of landslide susceptibility mapping using optimized PSO-ANN technique. *Engineering with Computers*, 35(3), 967–984. <https://doi.org/10.1007/s00366-018-0644-0>
- [77] Nguyen, H., Moayedi, H., Foong, L. K., Al Najjar, H. A. H., Jusoh, W. A. W., Rashid, A. S. A., & Jamali, J. (2020). Optimizing ANN models with PSO for predicting short building seismic response. *Engineering with Computers*, 36(3), 823–837. <https://doi.org/10.1007/s00366-019-00733-0>

- [78] Khari, M., Jahed Armaghani, D., & Dehghanbanadaki, A. (2020). Prediction of Lateral Deflection of Small-Scale Piles Using Hybrid PSO–ANN Model. *Arabian Journal for Science and Engineering*, 45(5), 3499–3509. <https://doi.org/10.1007/s13369-019-04134-9>
- [79] Lv, L., Chen, T., Dou, J., & Plaza, A. (2022). A hybrid ensemble-based deep-learning framework for landslide susceptibility mapping. *International Journal of Applied Earth Observation and Geoinformation*, 108, 102713. <https://doi.org/10.1016/j.jag.2022.102713>.
- [80] Kemp, D.B.; Fox, M.J.; Coates, D.J. Distribution and abundance of four color morphs of *Paracirrhites forsteri* across habitats in the Red Sea. *PLoS ONE* **2017**, *12*, e0169079.
- [81] Drew, K.L.; Reece, M.; Augspurger, C.K. Genetic variation and population connectivity of *Cirrhitichthys oxycephalus* across geographic ranges. *PeerJ* **2023**, *7*, e18058.
- [82] Jankowski, J.A.; Walker, J.H.; White, M.E. Habitat selectivity and live coral dependence of hawkfishes (Cirrhitidae). *PLoS ONE* **2015**, *10*, e0138136.