

Article

Hybridization and Optimization of Bio and Nature-Inspired Metaheuristic Techniques of Beacon Nodes Scheduling for Localization in Underwater IoT Networks

Umar Draz ¹, Tariq Ali ^{2,*}, Sana Yasin ^{3,*}, Muhammad Hasanain Chaudary ⁴, Muhammad Ayaz ² ,
El-Hadi M. Aggoune ² and Isha Yasin ³

¹ Department of Computer Science, University of Sahiwal, Sahiwal 57000, Pakistan; umardraz@uosahiwal.edu.pk

² Artificial Intelligence and Sensing Technologies (AIST) Research Center, University of Tabuk, Tabuk 71491, Saudi Arabia

³ Department of Computer Science, University of Okara, Okara 56300, Pakistan; ishayasin42@gmail.com

⁴ Department of Computer Science, COMSATS University Islamabad, Lahore Campus, Lahore 45550, Pakistan; mhchaudary@cuilahore.edu.pk

* Correspondence: teshaq@ut.edu.sa (T.A.); sana.yasin@uo.edu.pk (S.Y.)

Abstract: This research introduces a hybrid approach combining bio- and nature-inspired metaheuristic algorithms to enhance scheduling efficiency and minimize energy consumption in Underwater Acoustic Sensor Networks (UASNs). Five hybridized algorithms are designed to efficiently schedule nodes, reducing energy costs compared to existing methods, and addressing the challenge of unscheduled nodes within the communication network. The hybridization techniques such as Elephant Herding Optimization (EHO) with Genetic Algorithm (GA), Firefly Algorithm (FA), Levy Firefly Algorithm (LFA), Bacterial Foraging Algorithm (BFA), and Binary Particle Swarm Optimization (BPSO) are used for optimization. To implement these optimization techniques, the Scheduled Routing Algorithm for Localization (SRAL) is introduced, aiming to enhance node scheduling and localization. This framework is crucial for improving data delivery, optimizing Route REQuest (RREQ) and Routing Overhead (RO), while minimizing Average End-to-End (AE2E) delays and localization errors. The challenges of node localization, RREQ reconstruction at the beacon level, and increased RO, along with End-to-End delays and unreliable data forwarding, have a significant impact on overall communication in underwater environments. The proposed framework, along with the hybridized metaheuristic algorithms, show great potential in improving node localization, optimizing scheduling, reducing energy costs, and enhancing reliable data delivery in the Internet of Underwater Things (IIoT)-based network.

Keywords: bio-inspired; genetic algorithm; beacon node localization; energy reduction; metaheuristic algorithms

MSC: 68W50; 68T20



Citation: Draz, U.; Ali, T.; Yasin, S.; Chaudary, M.H.; Ayaz, M.; Aggoune, E.-H.M.; Yasin, I. Hybridization and Optimization of Bio and Nature-Inspired Metaheuristic Techniques of Beacon Nodes Scheduling for Localization in Underwater IoT Networks. *Mathematics* **2024**, *12*, 3447. <https://doi.org/10.3390/math12223447>

Academic Editors: Igor Litvinchev, Tetyana Romanova and Socorro Rangel

Received: 27 September 2024

Revised: 30 October 2024

Accepted: 31 October 2024

Published: 5 November 2024



Copyright: © 2024 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

1. Introduction

Underwater environment is complex in nature and exploration is not an easy job. The complexity further increases due to its high dynamic and harsh condition. In such environments, the traditional algorithms are not suitable for data routing with an appropriate set of beacon nodes, because these algorithms are only based on a partial optimization approach [1]. Metaheuristic optimization is a method of solving complex optimization problems by using metaheuristic algorithms. These algorithms are inspired by natural processes and are used to find approximate solutions to problems that are difficult or impossible to solve exactly. They work by mimicking the behavior of natural systems, such as the movement of particles, the evolution of populations, or the search behavior of animals,

to find the optimal solution to a problem [2]. Metaheuristic optimization algorithms [3] are generally divided into two categories: population-based and individual-based. Population-based algorithms, such as Genetic Algorithms (GA), Firefly Algorithm (FA), Levy Firefly Algorithm (LFA), Bacterial Foraging Algorithm (BFA), etc., work by maintaining a population of candidate solutions and repeatedly applying genetic operators such as selection, crossover, and mutation to find the optimal solution [2–5]. Individual-based algorithms such as Binary Particle Swarm optimization (BPSO), Ant Colony Optimization (ACO), etc., which simulate annealing and particle swarm optimization, work by iteratively modifying a single candidate solution to find the optimal solution. Metaheuristic optimization algorithms have a wide range of applications including engineering, physics, and computer science [5]. They have been used to solve problems such as scheduling, logistics, object optimization, and ranking of nodes in wireless networks. One of the key advantages of metaheuristic optimization algorithms is their ability to find approximate solutions to complex problems, even in the presence of noise and uncertainty. They are also relatively easy to implement and can often be applied to problems with many variables or a complex and non-linear structure. However, bio-inspired approaches in computer science have gained wider adoption within the research community compared to other fields. Additionally, solutions to bio-related issues and computational challenges are increasingly reliable and effective, enhancing user satisfaction and computational performance [6]. In this study, metaheuristic algorithms are used in Underwater Acoustic Sensor Networks (UASNs) to optimize and further hybridized with Elephant Herding Optimization (EHO) [7] for various aspects of the network, such as scheduling/ranking of nodes, path routing, sensor deployment, and so on. The specific algorithms used include Genetic Algorithm (GA), Bacterial Foraging Algorithm (BFA), Firefly Algorithm (FA) and its latest version Levy Firefly Algorithm (LFA), and Binary Particle Swarm Optimization (BPSO), all of which are hybridized with Elephant Herding Optimization (EHO). Moreover, GA, BFA, FA, LFA, and BPSO are applied with hybridized EHO [7] for the Scheduled Routing Mechanism (SRM) to design Scheduling Routing Algorithm for Localization (SRAL). To reduce delays and an unnecessary long routing path, these hybrid schemes help by optimizing the route from source to destination. Moreover, EHO can potentially be used for scheduling and routing mechanisms, as it is a general optimization algorithm that can be applied to a wide range of problems. However, the performance of EHO for scheduling and routing problems specifically would depend on the specific problem and its fitness function and its error value. Scheduling and routing problems typically involve finding the optimal order or sequence of tasks or routes based on certain constraints and objectives [8]. These problems can be formulated as optimization problems, and EHO can be used as an optimization algorithm to find an optimal or near-optimal solution. All of the above techniques have latency issues when only used for optimization [9] but the optimization with hybridization of metaheuristic approaches such as EHO with GA, FA, LFA, BFA, and BPSO have enough potential on those areas where the large-scale network needs to be tackled, such as an underwater environment.

All of these algorithms are hybridized with EHO for particular tasks so that EHO-BFS is used for selection population update, EHO-BFA emphasizes the selection of the best, EHO-GA is used for the selection of flight (routing) path, BFS-FA is used for select positions, BFA-LFA is used to calculate directions (best selection), and BFA-BPSO provides scheduled positions updated for beacon node localization. After that, the architecture of scheduled beacon node broadcasting and fitness functions of metaheuristic techniques are analyzed to identify the impact of scheduled/unscheduled nodes before going to extensive simulation setup with state-of-the-art benchmark techniques. The proposed model, SRAL, is finally compared with other competing schemes such as Range-based Directional Beacon Forwarding (RDBF) [10], Energy Efficient Probabilistic Depth Based Routing (EEPDBR) [11], Using Directional Beacon (UDB) routing [12], Three-Dimensional Underwater Localization (3DUL) [13], Underwater Localization (UL) [14], and Range-free scheme based on Mobile Beacon (RSMB) [15].

The rest of this paper is structured as follows: Section 2 deals with related work while Sections 3 and 4 discuss the system and proposed model. In Sections 5–10, the hybridization of nature- and bio-inspired techniques is presented. Moreover, the beacon node broadcasting is presented in Section 11 while simulation results are described in Section 12. Finally, the conclusion is presented in Section 13.

Contributions

The core contributions of the proposed communication framework are as follows:

- The proposed framework integrates multiple bio-inspired metaheuristic algorithms, such as GA, FA, LFA, BFA, and BPSO, with optimization and hybridization of EHO. This hybridization optimizes the scheduling of underwater nodes, resulting in significant energy savings and improved network efficiency.
- The framework is designed to enhance the Packet Delivery Ratio (PDR) by addressing the limitations of existing routing protocols such as RDBF, EEPDBR, UDB, 3DUL, UL, and RSMB, respectively. Through the strategic use of scheduled beacon nodes and optimized routing processes, the proposed scheme outperforms the prior protocols.
- Results from extensive simulations indicate that the scheme performs well even with an increased number of nodes (up to 700), maintaining stable performance with minimal localization errors and optimized resource utilization. These results validate the scheme's effectiveness in managing network traffic, improving communication efficiency, and ensuring reliable data delivery in large-scale underwater networks.

These contributions collectively address the core challenges of underwater communication, providing a highly efficient and reliable solution for managing the unique characteristics of UASNs. The proposed scheme demonstrates potential for real-world applications, particularly in scenarios where energy efficiency, accurate localization, and optimized routing for data transmission are critical.

2. Related Work

Many traditional techniques are proposed to handle the issue of scheduling the beacon nodes to some extent such as [10–15] but in fact no one takes the inspiration from Bio-Inspired Algorithms (BIAs) particularly in the field of UASNs [6]. Energy-efficient, low-latency scheduling has been addressed in the past, but they only aim to optimize traffic in one direction (broadcast or converge cast). For tree-based networks, a prior study describes the low-latency two-way beacon scheduling (LTBS) issue [16], where beacon nodes provide broadcast and converge cast traffics. This scheme does not cover the localization error and ignores the latency issue with adopting the two-way scheduling. Many techniques like proportion of line-of-sight (LoS) beacon scheduling based on angle of information/arrival, Time Difference of Arrival (TDoA) based scheduling, etc. [17] worked well in WSNs but failed underwater due to compatibility issues. For node scheduling, underwater metaheuristic techniques are used to optimize the scheduling by introducing shiftable nodes. The major reasons are due to dynamic topology, local work optima, link quality, and support for multi-path options to reach the destination within the shortest time.

Sufficient research focuses on ensuring the successful operation of UASNs in challenging conditions, including application, transportation, network, medium access, synchronization, and localization. This takes into account the difficulties caused by mobility and energy limitations [18]. A conventional oceanographic technology known as “acoustic based localization” uses either the Short Base Line (SBL) or Long Base Line acoustic-based procedures (LBL) [19,20]. Due to the high complexity of the procedures, semi-centralized and distributed localization strategies for UASNs must be developed. Article [21] reviewed various range-based projection and localization phases in 3D-UASNs using Angle of Arrival (AoA) conversion to 2D-UASNs optimization-based good localization ratio and coverage while consuming less energy. Although using a wide array of beacon nodes increases the computational complexity, it also reduces the energy consumption, but because there are fewer unknown nodes, this approach may be used to locate more mobile nodes with less

energy and less time expense than using a narrow array of beacon nodes. SOSNET (Smart Ocean Underwater Sensor Network) [22] introduced adaptive node clustering techniques that employ a moth flame optimizer (MFO) [23] for selecting the nearest optimal cluster. MFO is basically a bio-inspired technique that detects the movement of moths toward light (it is the same as firefly tendency to move towards light). SOSNET-FMO [22,23] was compared with other algorithms, including Grey Wolf Optimization (GWO) [24] and Ant Colony Optimization (ACO) [25]. Transmission range, node density, and grid size are the comparison parameters that show the MFO outperformed the other schemes. To enhance the reliability of data transmission through a hybrid metaheuristic algorithm, GSLS was performed in [26]. This algorithm integrates a Global Search Algorithm (GSA) and a Local Search Algorithm (LSA). The GSA, inspired by the principles of Genetic Algorithms, and the LSA, derived from scattered search techniques, function concurrently to effectively explore the problem space. This approach not only reduces runtime but also improves the quality of the path taken. An innovative multi-objective optimization (MOO) agent that integrates Particle Swarm Grey Wolf Optimization (PSGWO) with inverse fuzzy ranking is proposed in [27]. The enhanced PSGWO model was then employed to create a population- and multi-criteria-based soft computing algorithm known as fuzzy PSGWO. Additionally, the results indicate lower average values for the fitness function and demonstrate the computational efficiency of the proposed algorithm.

In general, the GA [28] is a bio-inspired optimization technique that mimics the process of natural selection. It uses operators such as selection, crossover, and mutation to evolve a population of solutions towards an optimal solution. Hybridization of GA with other algorithms can enhance convergence speed and solution quality by combining exploration (global search) and exploitation (local refinement) capabilities. In the same way, FA [29] is inspired by the flashing behavior of fireflies, where brighter fireflies attract others. The brightness is associated with the fitness function, and the movement of less bright fireflies toward brighter ones represents the search for optimal solutions. Similarly, LFA [30] is a variation in the Firefly Algorithm that incorporates the Levy flight distribution, allowing for more random, long-distance movements in the search space. This enhances the exploration ability of FA, making it less likely to become trapped in local optima, especially in complex optimization problems. Meanwhile, the BFA [31] is inspired by the foraging behavior of *E. coli* bacteria, where bacteria “swim” or “tumble” in search of nutrients (optimal solutions). It involves chemotaxis, reproduction, and elimination-dispersal processes. Moreover, BPSO [32] is a variant of the Particle Swarm Optimization (BPSO) adapted for binary search spaces. It is used for discrete optimization problems where variables take binary values. Hybridizing BPSO with other metaheuristics like EHO enhances the exploration of the solution space while maintaining robustness in finding optimal or near-optimal solutions. EHO is a nature-inspired optimization algorithm based on the social behavior of elephant herds. It uses clan updating and separating operators to optimize solutions. When hybridized with algorithms like [28–32], the EHO [7] can enhance the exploration–exploitation balance and improve global search capabilities, leading to faster and more accurate convergence.

It is worth noting that each of these algorithms can be hybridized with Elephant Herding Optimization (EHO) to combine the strengths of various approaches, leveraging the global search capabilities of one method with the local refinement potential of another for more effective optimization.

3. System Model

This section provides a solution to the identified problem in which beacon-based broadcasting of node localization with scheduled underwater nodes is addressed, although many schemes already exist for beacon-based broadcasting for node localization underwater without a scheduling mechanism [21–27]. The scenario of beacon broadcasting is when a beacon node finds another volunteer node in the communication zone for localization to include the action of data packet forwarding. There are multiple objectives, including the

minimization of nodes broadcasting, accommodating a set of minimum neighboring nodes with the maximum set of localized nodes, and priority-based scheduled beacon broadcasting, for beacon node localization. An effective method is provided to manage beacon nodes for optimal broadcasting, with its objective function framed as a multi-objective scheduling problem, where beacon node broadcasting is considered for two instances: (i) adaptive beacon broadcasting and (ii) opportunistic beacon broadcasting modes. The mechanism of Beacon Node Localization (BNL) [33] is also incorporated because it influences more power scheduling with the help of metaheuristic Bio-Inspired Algorithms (BIAs) during critical condition of unlocalized nodes. The BNL and scheduled BNL also help in reducing the cost of nodes and energy during the broadcasting of nodes in a large scale of area such as UASNs.

The sonobuoys are assumed to locate and satisfy the initial node position that helps to determine the other node locations through beacon nodes. These beacon nodes are deployed usually on seabed (middle of the ocean) and set accordingly with respect to area of interest. Beacon nodes include two-way motion which is diagonal (3D space) and vertical to accommodate the number of nodes for its location [21]. Many responsibilities such as running the beacon chain, synchronizing the beacon chain with peer nodes, and keeping track of the nodes chain state are in line with beacon node localization [33]. The underwater network is connected to the base station, which is further linked with sonobuoys. The sonobuoys send classified messages to beacon nodes. Beacon nodes are either adaptive or opportunistic for further broadcasting for scheduling (according to the defined fitness function) which is taken from [34,35]. Under the concept of opportunistic broadcasting and routing underwater, the proposed technique first locates the nearby node. The nodes that are located first as the result of beacon-based broadcasting are called a ‘primary node’. The primary nodes then locate further nodes associated as ‘secondary node’. Figure 1 presents a quick description of the system model.

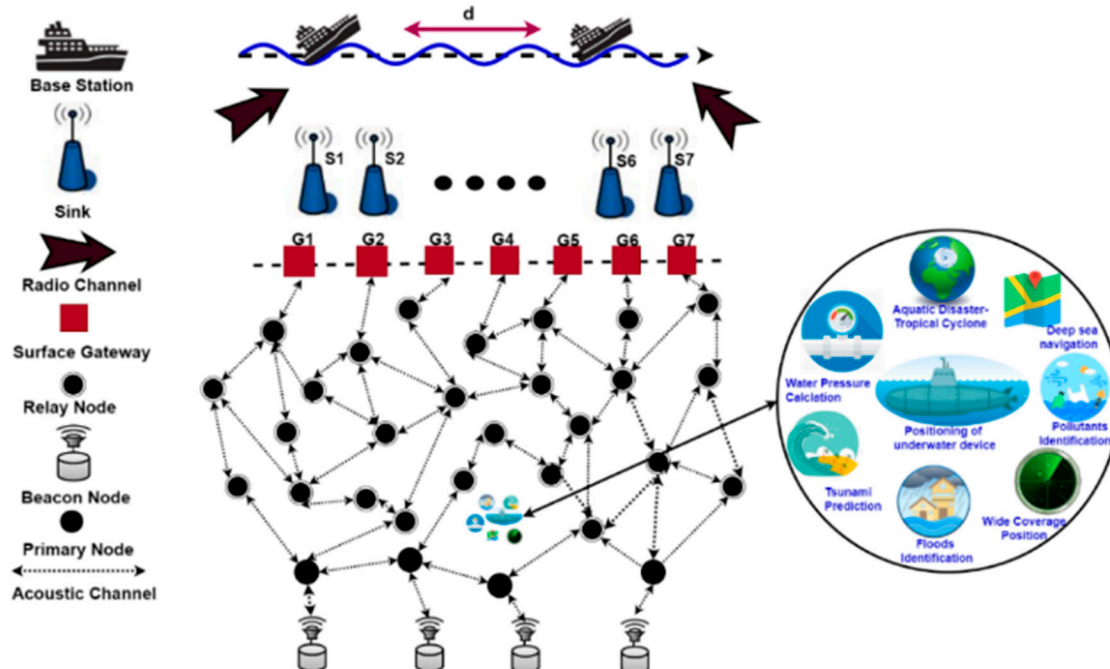


Figure 1. Conceptual overview of system model.

4. Proposed Model

Existing work regarding scheduled beacon node broadcasting in underwater network provide the location using adaptive and opportunistic methods formulated under different schemes such as Time of Arrival (ToA), Time Difference of Arrival (TDoA), and receiving signal strength indicators without introducing the scheduled mechanism [17]. Additionally,

many existing techniques offer linear algorithm control parameters, such as centroid and range-based or range-free algorithms, to modify and expand the criteria of beacon-based broadcasting. In GA, FA, LFA, BPSO, BFS, BFA [28–32] algorithms also give ideas to use a metaheuristic approach in TWNS and provide motivation for the application of UASNs. To address these challenges, a range-free metaheuristic-based beacon node broadcasting algorithm is proposed with the following key parameters, beacon frequency, transmission power, and beacon interval, optimized in UASN algorithms to ensure network performance while conserving energy. Algorithms can incorporate adaptive scheduling to adjust beacon behavior based on node status, balancing accuracy, and energy efficiency. The proposed method does not require the adjustment of algorithm-specific control parameters but the control parameters like population size of control, fitness function, lifetime of the nodes, and impact of scheduled beacon nodes need to be modified when and where required. The scheduling of the beacon nodes problem, defined as a multi-objective scheduling problem, is solved by the metaheuristic-based method. The goal of the multi-objective scheduling issue is to optimize the data packet forwarding ratio while decreasing localization error, false identification of beacon nodes, and shortest scheduled route for data forwarding. Many metaheuristic-based approaches, like BFA, GA, FA, LFA, and BPSO [28–32] with their hybridization of EHO, are implemented for computation analysis in order to calculate the performance of the suggested method. The formulation of beacon-based broadcasting and its parameter modification are discussed in detail in the next section.

4.1. Formulation of Objective Function

A Beacon-based Scheduled Routing Algorithm for Localization (SRAL) (either adaptive or opportunistic) requires the following information for optimal results: Node deployment order (O^{ND}) which refers to the sequence in which nodes are deployed in a network. Number of beacon nodes (B^{unit}) which is the total number of beacon nodes in a network. Broadcasting communication scenario (S^{cost}) which refers to the conditions under which broadcasting is conducted in a network. The criteria of beacon broadcasting (CB) are the factors that determine when a beacon node will broadcast its information. The neighboring list of primary nodes (LN^P) is a list of all the primary nodes that are in close proximity to a given primary node.

- Node deployment order (O^{ND});
- Number of beacon nodes (B^{unit});
- Broadcasting communication scenario (S^{cost});
- Criteria of beacon broadcasting (CB);
- Neighboring list of primary nodes (LN^P).

For optimal node scheduling problems, the metaheuristic algorithms have their own characteristics, therefore the objective functions are split into two cases. The primary goal is to optimize energy efficiency, cost-effectiveness, and network overhead while maximizing trust, security, and the lifetime of the network with the help of simulation models in which nodes are randomly distributed as shown in Equation (1) as follows:

$$\text{Obj}^s(N) = \text{Min} \sum_{n=1}^N (\sum (F_f^c(N.E) \times B^{unit}(N.E) + S^{cost}(N.E) + \sum_{n=1}^{|CB|} (UID(N.E) \times E^{UID}(N.E) + \sum_{n=1}^{|CN|} (UID^{LN}(N.E) \times E^{UID}(P)) + \sum_{n=1}^{|CN|} (UN^{LN}(N.E) \times E^{UD}(P) + \sum_{n=1}^{|ND|} (ND(N.E) \times E^{ND}(N.E)) + \sum_{n=1}^{|BNB|} (BNB^{B^{unit}+S+CB+E+LN^P+UD+ND}(N.E) \times E^{BNB}(N.E))) \quad (1)$$

where $|UID|$, $|UD|$, $|B^{unit}|$, $|LN^P|$, and $|ND|$ are the coordinates of UID (Un-identified Nodes ID), UD (Identified Nodes), B^{unit} , LN^P , and ND, respectively.

$$B^{unit}(N.E) = |x| = \begin{cases} 1, & \text{If the } n\text{th node dispatch } LN^p \text{ is turn ON} \\ 0, & \text{If the } n\text{th node dispatch } LN^p \text{ is turn OFF} \end{cases} \quad (2)$$

where ‘E’ is taken as each associated node energy at any given time ‘t’; $B^{unit}(N.E)$ is the nth beacon neighboring node while F_f^c is total fitness function cost; S^{cost} is the scenario cost, and CB is the denoted criteria parameter of beacon broadcasting that are usually common in all types of data scenarios which are taken from [36]. UID is the unidentified node while UD is the identified; ND is the node deployment order (layer-by-layer scenario is adopted); and NL^P is the neighboring list of nodes which are primary. In order to decide the number of sub-units of BNB Equation (3) works, in such a way that, the $E(N^{B^{unit}})$, $E(N^{ND})$, $E(N^{UID})$, $E(N^{UD})$, and $E(N^{NL^P})$ are the cost evaluation of B^{unit} , ND, UID, UD, and NL^P , respectively. The beacon node generation function is given in Equation (3) as follows:

$$F_f^c(N.E) = F_a(N) + F_b(N) + E^{BNB}(N) + F_f^{BND}(N) \times E(F_f^{BND}(N)t)^2 \quad (3)$$

The startup function cost is given in Equation (4) that helps to initiate the process of broadcasting as follows:

$$S^{cost}(N.E) = \{S^{cost}, \text{ if } B^{unit}(N.E) - B^{unit}(N, t-1) = 1, 0 \text{ otherwise}\} \quad (4)$$

where S^{cost} is the initial startup cost. In case II (opportunistic beacon broadcasting), the maximum node broadcast is obtained from the localization revenue by sub-structuring the production cost of the scenario, and consequently the operation and energy costs are minimized. Thus, the objective function is split into Obj_1^G and Obj_2^G as demonstrated in Equations (5) and (6).

$$Obj_1^G(N) = \min \sum_{t=1}^T (G(t) \times E^{loc}(t) + \sum_{f=1}^{F_f^c} (F_f^c(N.E)) \times B^{unit}(N.E) + S^{cost}(N.E) + \sum_{n=1}^{|CB|} (UID(N.E) \times E^{UID}(N.E)) + \sum_{n=1}^{|LN|} (UID^{LN}(N.E) \times E^{UID}(P)) + \sum_{n=1}^{|UD|} (UN^{LN}(N.E)) + E^{UD}(P) + \sum_{n=1}^{|ND|} (ND(N.E)) \times E^{ND}(N.E) + \sum_{n=1}^{BNB} (BNB(N.B) \times E^{BNB}(N.E))) \quad (5)$$

where, given t, $G_{all}(t)$ is the total generation cost of beacon node operation and $E^{Loc}(t)$ denotes the total cost of localization in $G_{all}(t)$. Meanwhile, all other related parameters must be greater than zero as $E^{Loc}(t) > 0$.

$$Obj_2^G(t) = \max \sum_{t=1}^T (-G_{del}(t) \times E^{loc}(t) + \sum_{f=1}^{F_f^c} (F_f^c(n, t) \times B^{unit}(n, t) + S^{cost}(n, t) + \sum_{n=1}^{|CB|} (UID(n, t) \times E^{UID}(n, t)) + \sum_{n=1}^{|LN|} (UID^{LN}(n, t) \times E^{UID}(P)) + \sum_{n=1}^{|UD|} (UN^{LN}(n, t)) + E^{UD}(P) + \sum_{n=1}^{|ND|} (ND(n, t)) \times E^{ND}(n, t) + \sum_{n=1}^{BNB} (BNB(n, t) \times E^{BNB}(n, t)))) \quad (6)$$

where Obj_1^G and Obj_2^G are objective functions of case II. At any given time, ‘t’ $G_{all}(t)$ is the total generator cost of allocated nodes as beacon nodes. The negative sign of $G_{del}(t)$, the deallocating cost, should be less than the $G_{all}(t)$ cost. The generation cost at any time, ‘t’, to the beacon node must be less than zero as $P_{all}(t) < 0$.

4.2. Formulation of Multi-Objective Scheduling Problem

The diversion of the objective function from adaptive and opportunistic modes is formulated for a multi-objective problem by keeping the capacity of beacon node broadcasting and data packet exchange, which is discussed in Equation (7).

$$E_{beacon}^{Pmin}(t) \leq E_{beacon}^P(t) \leq E_{beacon}^{Pmax}(t) \quad (7)$$

where $E_{beacon}^{Pmin}(t)$ and $E_{beacon}^{Pmax}(t)$ are the maximum and minimum position exchange $E_{beacon}^P(t)$. The value of $E_{beacon}^P(t)$ can be as large as possible at any given time, ‘t’. The net position exchange, $E_{beacon}^P(t)$, with generation of allocated and deallocated cost of beacon node is computed as in Equation (8) to Equation (11).

$$E_{beacon}^P(t) = G_{all}(t) - G_{del}(t) \quad (8)$$

$$E_{min}^{Ff}(t) \leq E^{Ff}(t) \leq E_{max}^{Ff}(t) \quad (9)$$

where $E_{min}^{Ff}(t)$ and $E_{max}^{Ff}(t)$ are maximum and minimum function $F^{Ff}(t)$.

$E^{Ff}(t) = \{E^{Ff}(t) > 0, \text{ if not allocated,}$

$E^{Ff}(t) < 0, \text{ if allocated,}$

$$E^{Ff}(t) = 0, \text{ if empty} \quad (10)$$

$$E_{min}^{SON}(t) \leq E^{SON}(t) \leq E_{max}^{SON}(t) \quad (11)$$

where $E_{min}^{SON}(t)$ and $E_{max}^{SON}(t)$ are the minimum and maximum state of node (SoN) $E^{SON}(t)$ which is further computed from Equation (12).

$$E^{SON}(t+1) = E^{SON}(t) - B^{unit}(t) \frac{E^{Ff}(t) \times E^{SON}(t)}{\delta^{SON}} \quad (12)$$

where at any given time, 't', $E^{Ff}(t)$ is the total function efficiency and δ^{Ff} is the related positioning exchange capacity as described in Equation (13).

$$\sum_{t=1}^T E^{UB}(t) + \sum_{t=1}^{E^{Ff}}(n, t) = E_{load}^{forecast}(t) - E^{Bunit}(t) - E^{LNP}(t) - E^{Gall}(t) - E^{Gdel}(t) \quad (13)$$

where at any given time 't' $E_{load}^{forecast}(t)$ is the forecast load demand. Equation (13) presents the demand and supply of beacon node to balance equation.

The opportunistic based allocated demand from UD and unallocated node from UID area are discussed. SRAL adoption and its memory with On\Off station of dispatched LN^P as per Equation (2) and determine which LN^P is ON.

If the LN^P for beacon nodes are dispatched, then it is turned On. If the allocated node is less than the deallocated node, the F_f (fitness function) [37] starts working to explore other criteria options for CB. Otherwise, it again starts to continue allocating more nodes as required. This study considered SON of beacon if the allocated or deallocated nodes are empty. This process continues until the F_f does not stop to explore other nodes. For the B^{unit} operation, it directly depends upon the neighboring list of primary nodes. For the F_f^c , it depends upon the sub-structuring of all related beacon nodes to meet the allocated criteria. In our scenario, two cases are considered as the F_f^c is directly applied to all neighboring nodes or only applied when a node is sent through a proper channel as described in Obj_1^G and Obj_2^G , respectively. The symbols used in equations and their acronyms are listed in Table 1.

Table 1. Acronyms and their meanings.

Acronyms	Meanings
(O ND)	Node Deployment Order
(B ^{unit})	Number of Beacon Nodes
(S ^{cost})	Broadcasting Communication Scenario
(CB)	Criteria of Beacon Broadcasting
(LN ^P)	Neighboring List of Primary Nodes
(ND)	Node Deployment
E(B ^{unit})	Cost Analysis Is Counts As
E ^{UID}	Cost of Un-Identified Nodes ID
F_f^c	Cost of Fitness Function
BNB	Beacon Node Broadcasting
S ^{cost}	Initial Startup Cost

Table 1. Cont.

Acronyms	Meanings
E^{loc}	Energy For Localization
$G_{\text{all}}(t)$	Total Generation Cost of Beacon Node Operation
$E^{\text{Loc}}(t)$	The Total Cost of Localization In Result of $G_{\text{all}}(T)$.
$G_{\text{del}}(t)$	Total De-Allocation Cost f Beacon Node Operation

5. Nature- and Bio-Inspired Metaheuristic Algorithm Optimization for Scheduled Routing Mechanism

The fact that nature presents a variety of incredibly amazing, dynamic, resilient, sophisticated, and intriguing phenomena makes it a tremendous source of inspiration for figuring out how to simplify complex issues in computer science. The greatest and most ideal solutions, inspired by nature, are always found while monitoring and balancing exploitation and investigation. The basic idea behind bio/nature-inspired computing systems is that they are metaheuristic in nature, which is why a lot of studies have been focused in this field. It is important to expand the horizons of the research of bio/nature-inspired algorithms for exploring new areas of applications with greater prospects in computing is still young and the outcomes are spectacular. In underwater communication networks, multiple optimization algorithms are combined to improve the performance of the routing path. The specific algorithms that can be used in the hybrid approach include GA, BFA, FA, LFA, and BPSO with optimization of EHO for scheduled routing. In this research, five hybrid schemes are proposed. Modern GA, BFA, FA, LFA, and BPSO are applied with hybridized EHO for the Scheduled Routing Mechanism (SRM) to design SRAL. To reduce delays and unnecessary long routing paths, these hybrid schemes help with the optimization of the route from source to destination. Moreover, EHO can potentially be used for scheduling and routing mechanisms, as it is a general optimization algorithm that can be applied to a wide range of problems. However, the performance of EHO for scheduling and routing problems specifically would depend on the specific problem and the implementation of the algorithm. Scheduling and routing problems typically involve finding the optimal order or sequence of tasks or routes based on certain constraints and objectives. These problems can be formulated as optimization problems, and EHO can be used as an optimization algorithm to find an optimal or near-optimal solution. It is also worth noting that there are other optimization algorithms that are specifically designed for scheduling and routing problems, such as Ant Colony Optimization (ACO) and Tabu Search (TS) [38]. These algorithms have been well studied for their performance on a variety of scheduling and routing problems. The quick short description and output function of each scheme are discussed below. GA is a widely used optimization algorithm that is inspired by the process of natural selection. In the context of SRM, GA can be used to generate a diverse set of routing schedules and find the optimal schedule that minimizes the energy consumption while satisfying the constraints of the network. The BFA is a bio-inspired optimization algorithm that is inspired by the foraging behavior of bacteria. It has been proposed as a solution to the problem of scheduling and routing in underwater sensor networks. The algorithm mimics the foraging behavior of bacteria by dividing the network into several clusters, each led by a 'leader' node that is responsible for coordinating the activities of the other nodes in the cluster. The leader nodes then communicate with each other to exchange information about the network's state and to determine the best routes for data transmission. The BFA has been shown to be effective in reducing energy consumption and increasing network lifetime in underwater sensor networks by using the natural behavior of bacteria as inspiration to solve routing and scheduling problems. FA is a population-based optimization algorithm inspired by the flashing behavior of fireflies. FA uses the attraction and brightness of the fireflies to guide the search process. In the context

of SRM, FA can be used to find the optimal schedule that minimizes energy consumption while satisfying the constraints of the network.

The latest version of FA is Levy Firefly Algorithm (LFA) which uses light intensity as a measure of firefly attractiveness to update the traditional function of FA. In the case of scheduling and routing in UASNs, a BFA can be used to optimize the selection of the beacon nodes, schedule data transmissions, and adapt to changing network conditions. The algorithm can be integrated into the beacon routing protocol to optimize the selection of the beacon nodes, which are responsible for transmitting the network's control information, and to optimize the schedule of data transmissions. BPSO is a variant of the Particle Swarm Optimization (PSO) algorithm that is specifically designed to handle binary variables. In the context of SRM, BPSO can be used to find the optimal schedule that minimizes energy consumption while satisfying the constraints of the network and the hop-by-hop addressing based routing mechanism [39].

In a hybrid approach, the strengths of multiple optimization algorithms are combined to improve the performance of the SRM. For example, GA can be used to provide a diverse population of solutions, while BFA can be used for its ability to handle binary variables. Similarly, FA and LFA can be used for their ability to handle multi-modal optimization problems and EHO and BPSO can be used for their ability to handle optimization problems with discrete and binary variables, respectively.

It is important to note that hybridization is not a magic solution, and the performance of hybrid algorithms depends on the problem at hand and the way the algorithms are combined. It is also important to note that an appropriate selection method chooses the best algorithm for the current iteration. The simulation is conducted based on inline scheduled algorithms for underwater nodes. The SRM evaluated the output of the five hybrid algorithms and selected the best one for underwater node broadcasting for forwarding the data packets. It is assumed that all hybrid algorithms underwater node time intervals are equivalent. The simulation was conducted for 700 nodes by using different parameters.

6. Hybridization of Elephant Herding Optimization

Elephants are very sociable creatures, and they prefer to herd together and adhere to herding customs. Elephants that live in herds in the form of clans are collectively governed by a female known as the matriarch, who is typically the leader of all elephants living in the herd at the time. The matriarch oversees upholding the clan's herding customs while the males live apart and use low-frequency vibration to stay connected to their primary family group, the female and the calves dwell in the main family group. The group taps with their feet to create the vibrations. When they reach adulthood, the male calves are compelled to join the other males and are removed from the main family group. As a result, the two aspects of elephant herding, creating clans and separating the males from the main family group, are crucial. To implement the suggested scheme like SRM, the two phases of the elephant herding rules are combined with GA, FA, LFA, BFS, BFA, and BPSO in this study. First, a problem's population is divided into sub-populations (clans), and then the worst elements, typically nodes that are not scheduled, are either eliminated from the clans or the best elements are chosen. The best element (matriarch) in a clan influences changes to the clan's elements (location of underwater nodes). Equation (14) is used by the clan's elephant, designated "E", to update its location.

$$P_{new,clani,E} = P_{clani,E} + \beta \times (P_{clanbest,E} - P_{clani,E}) \times w \quad (14)$$

where $P_{new,clani,E}$ is the updated position of the elephant from the clan, 'i'. $P_{clani,E}$ is the old position of the elephant, E. The best position of the matriarch which is the fitness in clans is $P_{clanbest,E}$ while β is the influence of the matriarch with [0, 1]. 'w' is used for uniform distribution with range of [0, 1]. In the simulation, the random function is used to generate ' β ' and 'w'. The BFA is a metaheuristic optimization algorithm that is inspired by the foraging behavior of bacteria hybridized with EHO. The algorithm simulates the behavior of bacteria searching for food in a specific area. In the BFA, each bacteria represents a

solution to the optimization problem and the flock of bacteria represents the population of solutions. The bacteria move in the search space following the foraging behavior of bacteria. The time complexity of the BFA depends on several factors, for example, the size of the search space, the number of bacteria in the flock, and the number of iterations. In general, the time complexity of the BFS algorithm is 'O (n*T)' where 'n' is the number of variables and 'T' is the number of iterations. If the elephant 'E' from the clan is best, then the matriarch is updated as per Equation (15). Furthermore, the hybridization of EHO with the BFA is discussed in Algorithms 1 and 2, respectively.

$$P_{new,clani,E} = \gamma \times P_{clancentre,E} \quad (15)$$

When ' γ ' is the factor of [0, 1] to determine the influence of the central elephant of the clan $P_{new,clani,E}$.

Algorithm 1. Hybridization of EHO-BFS: Scheduling Population Update

```

1. for  $E_{beacon}^{P_{min}}(t) = 1 : P^{th}$  class (for all bacteria (underwater nodes) do
2.   for  $E_{beacon}^{S_{min}}(t) = 1 : S^{th}$  class (all bacteria (underwater nodes) in the nest) do
3.     Update  $E_{beacon}^P$  and  $E_{beacon}^S$  and generate
4.      $E_{beacon}^{P_{max}}(t)$ ,  $E_{beacon}^{S_{max}}(t)$  using Equations (7) and (8)
5.     if  $E_{beacon}^{P_{max}}(t) \geq E_{beacon}^{S_{max}}(t)$  then
6.       Update  $E_{beacon}^P(t)$  and fix the  $E_{min}^{F_f}(t)$  and  $E_{max}^{F_f}(t)$  Equation (9)
7.     end if
8.   end for
9. end for
```

Algorithm 2. Hybridization of EHO-BFS: Selection Best

```

1. for  $E_{beacon}^P(t)$  and  $E_{beacon}^S(t) = 1 : P^{th}, S^{th}$  group of birds
2.   (nodes) do
3.     Find the best pair using Equation (10)
4.     if  $E_{min}^{F_f}(t)$ ,  $E_{max}^{F_f}(t) > 0$ 
5.       Update and choose the pair using Equation (11)
6.     end if
7.   end for
```

The divide and conquer principle can improve the performance of an algorithm by reducing its runtime. To optimize the execution of the hybrid SRM for underwater nodes, the EHO algorithm can be combined with other metaheuristic techniques like GA, FA, BFA, and BPSO. Updating the clans about matriarchs can also be incorporated as part of the EHO process to further reduce runtime and complexity while still obtaining an optimal solution from the population. To incorporate the necessary components of SRM-EHO, the other hybrid algorithm is further broken into detailed sections in the next section.

7. Hybridization of EHO-BFS: Genetic Algorithm

GA is an algorithm inspired by natural selection in biology [28]. The genetic properties of an element are changed through the evolutionary processes of crossover and mutation. The population is generated at random, and elements are chosen using a fitness function, with the best element serving as the 'parents' for genetic crossover and mutation to produce the next population. The early stages of GA are used in the proposed hybrid SRM-GA to establish and evaluate the population for element positioning. The initial stage of EHO receives input from the examined population. The population of size 'n' produces the 'w' number of clans. Each clan's population is updated and mutated accordingly. Each clan's element is updated under the guidance of the matriarch. Furthermore, the hybridization of BFS with GA as BFS-GA is discussed in Algorithms 3 and 4, respectively. If an element

is superior to the matriarch, it eliminates the matriarch population. The second phase involves eliminating the least suitable parts, leaving just the finest ones. On the updated population, GA's crossover and mutation operations are put into practice. This procedure continues until the right (best) answer is found. The effect of the GA stages on the updated EHO population optimizes the time complexity and offers a productive remedy. The time complexity of the proposed hybrid GA depends on the total population 'n' and the number of clans 'w' (sub-population). In the proposed system, the complexity of SRC-GA is $\theta(n \times w)$, where $n \neq w$ and the size of the population $n > \text{size of clan } d$.

Algorithm 3. Hybrid of BFS-GA: Selection of Flight (Routing) Path

```

1. Initialize the population 'n'
2.   for i = 1 : n do
3.     Evaluate the population
4.   end for
5. Initialize the flight path  $X^{T,iter+1} \rightarrow rd.\lambda$ 
6. Sub – population of size  $X^{S,iter}$  of d, size =  $\frac{n}{X}$ 
7. While Max Generate do
8.   Generate the new population using Algorithm 1
9.   for i = 1 : w, For number of flight paths of actreai do
10.    for j = 1 : d, for all bactreia in follar, do
11.      Evaluate  $j^{th}$  bacteria in flight path
12.    end for
13.  end for
14.  Clf = Converger evaluated folts
15.  Evaluate Fitness of population (Uf)
16.  end while
17. LBP = Produce list of best position
18. CLBP = Calculate cost of list best position (scheduled position)
19. Update best position Algorithm 2 from Uf
20. Conv = Converge (Un) and (LBP)
21. for k = 1 : Inter Nodes; For all interraptible do
22.   for m = 1 : position; for all positions for a day
23.    for  $k^{th}$  Inter Node do
24.      Cross Over(Conv[m], rand(CLBP))
25.    end for
26.  end for
27. end for
  
```

Algorithm 4. Hybrid of BFS-FA: Selection of Positions

```

1.
   Initialize the objective function  $Q_1, Q_2, Q_3$ , and  $Q_4$  with its parametres and modified parametrs
2. Calculate the objective function cost  $f(x) = (Q_1, Q_2, Q_3, \dots, Q_n^T \text{ w.r.t time}$ 
3. Generate initial population of firefly  $X_i$  ( $i = 1, 2, \dots, n$ )
4.  $I_i$  for the  $X_i$  is determined by  $f(x_i)$ 
5. Calculate the  $\beta_i$  from  $I_i$  is determined by  $f(x_i)$ 
6. Coefficient of light absorption is defined ( $\beta$ )
7. Divide the firefly population of f equal sub population of (f folks, clans)
8. Update ( $UC_c$ ) folks, clans using Algorithm 1 and converge the updated clans of firefly
9. LBP = Find the list of best postion using Algorithm 2
10. while  $f < \text{Max Generation}$  do
11.  for  $i = 1 : n$ , all n fireflies ( $UC_c$ )do
12.    for  $j = 1 : i$  all 'n' fireflies do
13.      if ( $I_i > I_j$ ) move i towards j then
14.        Distance 'r' varies the attractiveness
15.      Find the new solution and update the light intensity  $f(x_i)$ 
  
```

Algorithm 4. *Cont.*

```

16.           end if
19.            $Idx = \text{Find current beat and store in an index}$ 
20.            $CIdx = \text{Calculate the cost for}(Idx)$ 
21.            $Mbp = \text{Mega beat position (BPs) and } (Idx)$ 
22.           for  $k = 1 : \text{length}(N_{\text{beacon}} > \text{Avg}_{p,s})$  do
23.             for  $N_{\text{beacon}}$  nodes greater than the average distance
24.               primary and secondary nodes rating  $\text{Avg}_{p,s}$ 
25.               for  $m = 1 : \text{BT for length of (BT)}$ 
26.                 for  $k^{\text{th}}$  firefly in a folks/clans do
27.                   do while
28.                     Select location from  $(Mbp)(\text{priority})$  minors
29.                     cost  $(CIdx)$ 
30.                   end for
31.                 end for
32.             end while

```

8. Hybridization of EHO-FA: Firefly Algorithm

FA is a flashing pattern-influenced algorithm that draws inspiration from nature. Flashes are used for a variety of things, mostly to attract new partners and for defense. High, brilliant fireflies warn off attackers and promote dominance. Distance has an inverse relationship with luminous intensity, which makes it changeable. The population is updated in the proposed hybrid algorithm utilizing the EHO and GA sub-steps, where population input is performed by Algorithms 1 and 2 in FA. In Algorithm 4, the steps for hybridized FA are described. FA generates the initial population, which EHO updates in two phases. Each sub-population of the population is referred to as a ‘clan’. Each clan has an ideal member who is known as the matriarch. A portion of the ‘w’ clans is supplied to the current FA. The worst aspect in each clan is eliminated in the second installment of EHO and GA, leaving just the greatest element, after it, the related best places are combined. The original population is separated into sub-populations throughout Algorithm 4’s phases and updated using Algorithm 1. Using Algorithm 2, the best sub-population component is chosen. The pieces that do not fit are all removed. The best elements are kept with their costs in a list, which is often a routing table. The list is combined with the top element produced by Algorithm 2, and a new population is created. The shiftable nodes in the final phase of EHO-FA have a lower fitness function rating than the submerged nodes, including beacon nodes. Every node is replaced with the least expensive location (already determined). The time complexity of EHO-FA depends on BP which is found using Algorithm 2 and the total updated population ‘n’. However, $BP \neq n$ and the time complexity of proposed hybridized FA is $\theta(BP \times n)$. Furthermore, the hybridization of BFS-FA is discussed in Algorithms 4 and 5, respectively.

Algorithm 5. Hybrid of BFA-FA: Calculate Direction (Best Positions)

```

1. Initialize all ‘b’ population
2. for  $i = 1 : b$  do
3.   Fitness evaluation of each Bacterium  $b_i$ 
4. end for
5. while Max Generation do
6.   for  $j = 1 : b$  do
7.     Discard and introduce new for unfit
8.   end for
9.   for  $k = 1 : b$  do
10.    Calculate fitness and tumble in random direction  $[-1, 1]$ 
11.    Calculating new direction (positions)
12.    Evaluate the fitness for new position

```

Algorithm 5. *Cont.*

```

13. if Last Fitness < Current then
14.   Replace Last Fitness with Current position
15.   F/N
16.   random direction (new position)
17. end if
18.   Evaluate fitness for new position selected best position
19. end for
20. LbP = Return list of best position
21. CbP = Calculate cost of list best position (Clans)
22. Update each clan using Algorithm 1
23. FbP = Find best position using Algorithm 2
24. LFbP = Converge (LbP) and (FbP)
25.   for k = 1 : length (Nbeacon > Avgp,s) do
26.     for Nbeacon greater than average energy rating Avgbeacon
27.       for b = 1 : BT : for length of (BT) for kth for, do
28.         Select position from (LFbP) prefer (priority) minimum at (CbP)
29.       end for
30.     end for
31. end while

```

9. Hybridization of EHO-BFA: Bacterial Foraging Algorithm

The *E. coli* bacterium species' behavior when looking for food served as the model for the BFA. The bacteria look for food in the surroundings while it is foraging. The neighbors are informed when plentiful food is found. The educated bacteria swarm towards the direction of food. The following swarm behaviors are seen throughout this process.

The meal is covered in bacteria;

- Chemotaxis motion help a bacteria identify food in their environment (tumble and swimming);
- The bacterium with insufficient nourishment perishes and vanishes;
- Bacteria that successfully proliferate and live reach the food. Additionally, they entice others by creating chemicals.

In the quest for new food sources, the germs disperse once more. The fitness of randomly chosen bacteria is assessed once the population has been initiated in the EHO-BFA. The BFA process is then used, which involves removing unhealthy bacteria, conducting a tumble, and swimming toward the food supply. With every instance of tumbles or swimming, the bacteria move either closer to the food or farther away from it. Every step in the water or during a tumble updates the population. The procedures are carried out for the entire population, returning a list of the top bacteria. The population on the list is brand new and is broken up into 'w' sub-populations or clans. Algorithm 1 is used to update the sub-populations. The upgrade is carried out with the help of the matriarch (local best). When another ingredient proves to be more suitable than the matriarch, the matriarch is updated. Each clan is updated, and Algorithm 2 only preserves the best components while eliminating the remainder of the undesirable components from the population. Each clan's result is combined to create a new list, and the cost of the items on the new list is determined. The nodes that have the closer position towards the sink are placed into less expensive time slots. As a result, nodes are placed in the optimal locations. The EHO-BFA stages are listed in Algorithm 5. For the number of nodes with an energy status larger than the average energy of all shiftable nodes, the temporal complexity of the proposed EHO-BFA is $(\text{length}(\text{NBeacons} > \text{AvgPr}))$. Hence, the complexity of EHO-BFA is $O(\text{length}(\text{NBeacons} > \text{AvgPr}) \times \text{BP})$.

Due to their conceptual similarities, BFA and FA's scheduling habits are comparable. Due to the allure of the dominating components, scheduling behavior in the proposed EHO-BFA, and EHO-FA is comparable. It is possible to use a hybrid of BFA and FA for scheduling and routing mechanisms in underwater environments, as both algorithms are general-purpose optimization algorithms that can be applied to a wide range of problems.

Scheduling and routing problems in underwater environments typically involve finding the optimal order or sequence of tasks or routes based on certain constraints and objectives. These problems can be formulated as optimization problems, and a hybrid of BFA and FA can be used as an optimization algorithm to find an optimal or near-optimal solution.

10. Hybridization of EHO-BPSO: Binary Particle Swarm Optimization

PSO has the capacity to be upgraded for a particular circumstance to offer effective answers. The PSO-based binary discrete search space is referred to as BPSO [32]. By requiring the velocity to be either 0 or 1 rather than between 0 and 1, the logical properties of the PSO and BPSO are not altered. The population in the proposed EHO-BPSO is initialized. Each particle's fitness is assessed. By comparing each particle with the one that is now fitted, the population is updated. This provides a list of the most recent (best) particles found using BPSO and each particle's price is computed for the list.

Using Algorithm 1, the list is divided into 'w' sub-populations or clans and updated for each of them. The update is carried out while being influenced by the 'matriarch' (best in the clan). The particle transforms into a matriarch if its value exceeds that of the matriarch. Using Algorithm 2, the best components are chosen from each clan. All clans' offspring are combined. Furthermore, the hybridization of EHO with BFA-BPSO is discussed in Algorithm 6.

A cost-effective solution is offered by this hybridized algorithm. The runtime is unaffected because the update and population evaluation are carried out independently. However, the time complexity of the proposed algorithm is influenced by the number of beacon nodes, which possess higher energy levels compared to the average energy across all available beacon nodes ($N_{\text{Beacons}} > \text{AvgPr}$) and their BT. The time complexity of proposed EHO-BPSO is $O(\text{length}(N_{\text{Beacons}} > \text{AvgPr}) \times \text{BT})$.

Algorithm 6. Hybrid of BFA-BPSO: Scheduled Positions Updated

START

1. Initialize all 'T' particles with their dimensions
 2. while Max Generation do
 3. for $i = 1 : T$ do
 4. Evaluate fitness values for each particle T_i
 5. if T_i particle is better than fitness value then
 6. Set current particle as BEST
 7. end if
 8. end for
 9. for $i = 1 : m$ select global BEST from all switch do
 10. Update variety of particle (between 0 – 1)
 11. Update position of particles
 12. end for
 13. end while
 14. RLu_p = Return list of updated position of particles
 15. CLu_p = Calculate cost of (RLu_p)
 16. Divide list (RLu_p) into 'n' equal sub population (clans)
 17. Update each clan using Algorithm 1
 18. Find Best position using Algorithm 2
 19. BP = Find best position using Algorithm 2
 20. LBP = Find best position using Algorithm 3
 21. UC_c = Find best updated scheduling UC_c using Algorithm 4
 22. for $k = 1 : \text{length}(N_{\text{scheduled}} > \text{Avg}_{BT})$ underwater nodes
 23. N_{sced} with best scheduled positions and rating
 24. Avg_{BT} do
 25. for $m = 1 : BT$; for length of Burst Time (BT)
 26. for k^{th} for a day do
 27. Select position for C_p minimum scheduled
-

Algorithm 6. Cont.

```

28.      cost (Lup)
29.    end for
30.  end for
31. END

```

11. Scheduled Beacon-Based Node Broadcasting

The proposed architecture for Beacon Node Broadcasting is presented in Figure 2. It has been demonstrated that the SRM-BNB method, which is a metaheuristic intelligence-based approach, offers the best scheduling for limited situations and unscheduled nodes. It is recognized that certain control parameters for optimization-based algorithms might affect them. Every algorithm has its own set of parameters, and the performance is governed by the correct setting of the control parameters. By avoiding a poorer option, the employment of the Bacteria Foraging Search (BFS) and its BFA with the BNB algorithm, which is influenced by metaheuristic, ensures that the solution to be determined does not depart from the optimum solution. It also has a single phase, making it simple to implement.

Let 'h(x)' be the objective function that is minimized at any iteration 't'. Suppose that there are 'n' decision variables (i.e., $j = 1, 2, \dots, n$) in the j^{th} variable, $V_{j,t,l}$ value is obtained for the l^{th} individual performing the l^{th} iteration. For our case, the modified value is obtained with Equation (16) as follows:

$$h(x) = V'_{j,t,l} + V_{j,t,l} + V_{1,t,l}(V_{j,\text{ideal},t} - |V_{j,t,l}|) - V_{2,j,l}(V_{j,\text{worst},t} - |V_{j,t,l}|) \quad (16)$$

where $V'_{j,t,l}$ is the updated value of $V_{j,t,l}$ and $V_{1,j,l}$. $V_{2,j,l}$ is in the range [0, 1]. The term $V_{1,j,l}(V_{j,\text{ideal},t} - |V_{j,t,l}|)$ indicates the tendency of the solution to move closer to the ideal solution. The term $V_{2,j,l}(V_{j,\text{worst},t} - |V_{j,t,l}|)$ indicates the tendency of the solution to avoid the worst solution. $V_{j,l,t}$ is accepted for a better function value at the end of the iteration that is used as the input to the subsequent iteration. The several functions described above are considered to determine how the suggested model works. Here, the value of 'x' that minimizes the multi-objective function is sought for. The stages that make up the suggested model are as follows:

Step 1: Obtain the beacon node generating individual dispatchable B^{unit} , the G_{all} and G_{dall} , load, while 'P' is taken as 30 individuals' solution to exchange the position, and 100 iterations are used as the stopping criteria.

Step 2: Use the upper and lower limits to generate the starting population at random within the range value that corresponds to h(x). A counter is set to the maximum iteration after the population generation.

Step 3: Run the first beacon node dispatch without the BNB, then use Equations (1) and (5), respectively, to obtain the objective function that has to be reduced for the adoptive and opportunistic nodes.

Step 4: At the conclusion of each iteration, update the population based on the fitness function comparison. In the event that the answer makes the goal constraint clear, be sure to perform the local search. Implement a strategy to keep and add variety to the population. Additionally, if the new solution provides a higher objective value, a global solution is obtained.

Step 5: Convert the adoptive population to a binary representation that is indicated by the on/off value of a dispatchable B^{unit} .

Step 6: For each iteration, run the BNB method for each person and calculate their related objective value.

Step 7: The energy scenario and turn on the dispatchable standby node if it is empty. Otherwise, turn off the dispatchable node.

Step 8: A control signal is transmitted to another dispatchable node and memory, and the routing table is modified by BNB if the load demand of UID at a specific time 't' is high.

```

graph TD
    Begin([Begin]) --> GetStatus[Get Status of Active Beacons]
    GetStatus --> BNB_BFA[BNB-BFA]
    BNB_BFA --> Is_LN_P{Is LN^P}
    Is_LN_P -- Y --> CB_ON[CB ON]
    CB_ON --> Is_Load_Threshold{IS Load < threshold}
    Is_Load_Threshold -- Y --> Dispatch[Dispatch]
    Dispatch --> End1([End])
    Is_Load_Threshold -- N --> Operate_B_Unit[Operate B_Unit]
    Operate_B_Unit --> Is_LN_P
    
    Is_LN_P -- N --> Is_O_N_D{IS O^N_D}
    Is_O_N_D --> BNB_BFA
    
    BNB_BFA --> UID_ID_Nodes{UID > ID Nodes}
    UID_ID_Nodes -- Y --> calculates_Fr[calculates F_r]
    calculates_Fr --> Is_CB{IS CB}
    Is_CB -- Y --> Discard1[Discard]
    Discard1 --> BNB_BFA
    
    Is_CB -- N --> Is_Load_LN_P{IS Load LN^P}
    Is_Load_LN_P -- Y --> CB_OFF[CB OFF]
    CB_OFF --> Is_Empty_B_Unit{Is Empty B_Unit}
    Is_Empty_B_Unit -- Y --> Nodes_Not_Schedule[Nodes not Schedule]
    Nodes_Not_Schedule --> S_cost_B_Unit{S^cost < B_Unit}
    S_cost_B_Unit -- Y --> CB_ON2[CB ON]
    CB_ON2 --> End2([End])
    S_cost_B_Unit -- N --> Dispatch_Nodes[Dispatch Nodes]
    Dispatch_Nodes --> CB_ON3[CB ON]
    CB_ON3 --> End3([End])
    
    Is_Load_LN_P -- N --> Track_UID_Nodes[Track UID Nodes]
    Track_UID_Nodes --> Is_B_Unit_LN_P{IS B_Unit < LN^P}
    Is_B_Unit_LN_P -- Y --> S_cost1[S^cost]
    S_cost1 --> Is_Empty1{Is Empty}
    Is_Empty1 -- Y --> Again_apply_CB[Again apply the CB]
    Again_apply_CB --> Is_Empty1
    Is_Empty1 -- N --> Schedule_nodes[Schedule the nodes]
    Schedule_nodes --> S_cost_B_Unit2{S^cost < B_Unit}
    S_cost_B_Unit2 -- Y --> CB_ON4[CB ON]
    CB_ON4 --> End4([End])
    S_cost_B_Unit2 -- N --> Discard_Nodes[Discard Nodes]
    Discard_Nodes --> S_cost_B_Unit2
    
    Is_B_Unit_LN_P -- N --> Track_PV[Track PV]
    Track_PV --> S_cost_B_Unit3{S^cost < B_Unit}
    S_cost_B_Unit3 -- Y --> CB_ON5[CB = ON]
    CB_ON5 --> Is_Empty2{Is Empty}
    Is_Empty2 -- Y --> Find_other_set[Find other set of nodes]
    Find_other_set --> End5([End])
    Is_Empty2 -- N --> Dispatch_Nodes2[Dispatch Nodes]
    Dispatch_Nodes2 --> End6([End])
    
    S_cost_B_Unit3 -- N --> Nodes_Active{O^N_D Active}
    Nodes_Active --> Is_Empty3{Is Empty}
    Is_Empty3 -- Y --> Check_Fitness[Check Fitness Function (F_f)]
    Check_Fitness --> Dispatch_Nodes3[Dispatch Nodes (If any)]
    Dispatch_Nodes3 --> End7([End])
    Is_Empty3 -- N --> Re_Schedule_Nodes[Re-Schedule Nodes]
    Re_Schedule_Nodes --> End8([End])
    
    Is_Load_LN_P --> Is_Load_LN_P
  
```

The flowchart illustrates the BNB-BFA algorithm. It begins with 'Begin', leading to 'Get Status of Active Beacons', which then leads to the 'BNB-BFA' block. From 'BNB-BFA', the algorithm checks 'Is LN^P'. If 'Yes', it turns 'CB ON' and checks 'IS Load < threshold'. If 'Yes', it 'Dispatch' and ends. If 'No', it 'Operate B_{Unit}'. If 'Is LN^P' is 'No', it checks 'IS O^N_D'. If 'Yes', it 'calculates F_r' and checks 'IS CB'. If 'Yes', it 'Discard' and loops back to 'BNB-BFA'. If 'No', it checks 'IS Load LN^P'. If 'Yes', it turns 'CB OFF' and checks 'Is Empty B_{Unit}'. If 'Yes', it 'Nodes not Schedule', then checks 'S^{cost} < B_{Unit}'. If 'Yes', it turns 'CB ON' and ends. If 'No', it 'Dispatch Nodes', then 'CB ON' and ends. If 'Is Load LN^P' is 'No', it 'Track UID Nodes', then checks 'IS B_{Unit} < LN^P'. If 'Yes', it calculates 'S^{cost}' and checks 'Is Empty'. If 'Yes', it 'Again apply the CB' and loops back. If 'No', it 'Schedule the nodes', then checks 'S^{cost} < B_{Unit}'. If 'Yes', it turns 'CB ON' and ends. If 'No', it 'Discard Nodes' and loops back. If 'IS B_{Unit} < LN^P' is 'No', it 'Track PV', then checks 'S^{cost} < B_{Unit}'. If 'Yes', it sets 'CB = ON' and checks 'Is Empty'. If 'Yes', it 'Find other set of nodes' and ends. If 'No', it 'Dispatch Nodes' and ends. If 'S^{cost} < B_{Unit}' is 'No', it checks 'O^N_D Active'. If 'Yes', it checks 'Is Empty'. If 'Yes', it 'Check Fitness Function (F_f)', then 'Dispatch Nodes (If any)' and ends. If 'No', it 'Re-Schedule Nodes' and ends. If 'O^N_D Active' is 'No', it 'Is Empty B_{Unit}'. If 'Yes', it 'Operate B_{Unit}' and ends. If 'No', it 'Is Empty' and ends.

11.1. Comparative Analysis of Metaheuristic Approaches for Fitness Function

By evaluating the fitness of candidate solutions, the optimization algorithm is able to generate new solutions that are progressively better in terms of the specific problem being solved. A well-designed fitness function can help the optimization algorithm converge faster and find higher quality solutions. Therefore, the separation and minimization of UIDn are formulated in Equation (17) as follows:

where UIDnB is the total un-identified beacon nodes that are calculated using Equation (18) and Equation (19), respectively.

where E_{cost}^{total} represents the aggregated total cost of the network during the process of selecting and separating out the reliable beacon nodes (BRel). ‘Un’ represents the aggregated

value of the total unknown nodes inside the network. Meanwhile, $\gamma = 1, 0$ and selecting and unselecting B^* (trusted beacon node) from 'B' beacon nodes.

This feature aids in the discovery of the best explanation. From 'Ff', choose the fitness criteria via list F_{pop}^B to avoid the unnecessary selection and saturation of beacon nodes in Equation (20) to Equation (25).

$$F_f = \min (F_{pop}^B) \quad (20)$$

Meanwhile, F_{pop}^B is calculated using Equation (5) as follows:

$$F_{pop}^B = \begin{cases} B_{negh}^{leg} \leq BC_{beac}^{negh} \leq B_{l1} \text{ if } \hat{I}_N = 0 \\ B_{l2} \leq E(B_{negh}^{leg}) < B_{l3} \text{ if } \hat{I}_N = 1 \end{cases} \quad (21)$$

$$B_{l1} = \text{Sum} (BC_{beac} \times BC_{beac}^{negh}) - \text{std} (E(BC_{beac}^{negh})) \quad (22)$$

$$B_{l2} = \text{std} E(BC_{beac}^{negh}) + \eta \times \min (E(BC_{beac}^{negh})) \quad (23)$$

$$B_{l3} = \text{mean}(E(BC_{beac}^{negh})) \quad (24)$$

$$Loc_B = \frac{Loc_B^{negh}(B_{l1}) - \min(List_{Loc_B}^{int})(B_{l2})}{\max Loc_B^{negh}(B_{l3}) - \min(List_{Loc_B}^{int})(B_{l2})} \quad (25)$$

The ' η ' value is determined to minimize costs by selecting beacon-enabled neighboring nodes as a by-product of the interaction between the beacon node and its neighboring nodes. The reason for taking $\eta = 2$ and the location of the beacon are considered as per set $\forall l \in \{2, 3, 4, \dots\}$ for every value of the beacon node and its corresponding location. Intentionally, we start from $\eta = 2$ because the first beacon node must be selected and considered as an anchor node to start the process; therefore, $\eta = 2$ for the localized BC and the limited criteria B_{l1} , B_{l2} , and B_{l3} , respectively. While the limitation of the total beacon limit (B_{l1}) and its related standard deviation (B_{l2}) with its mean value of (B_{l3}) can be calculated using Equation (20) to Equation (25), respectively. Therefore, the distribution of localization can be carried out using Equation (25), where Loc_B (localization of beacon nodes) is calculated from Equation (25). Meanwhile, all analyses of fitness functions are observed in Figures 3–5, respectively.

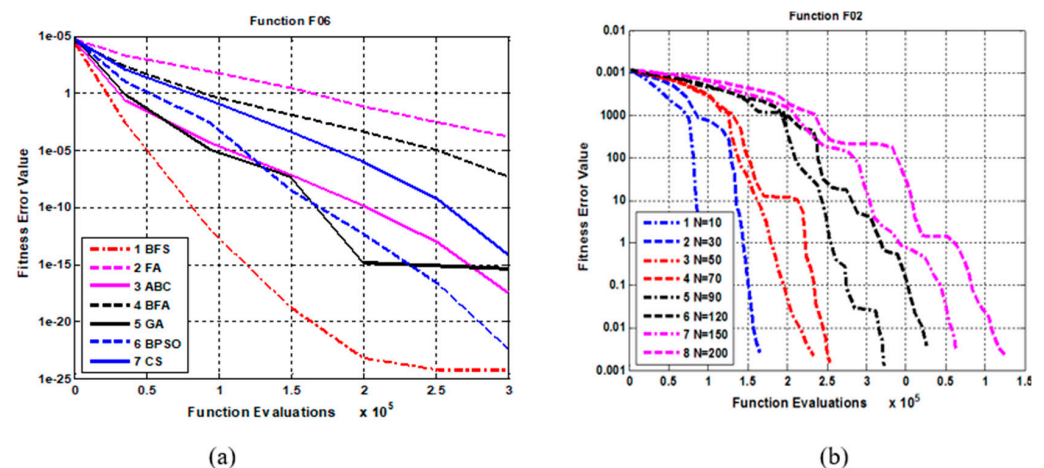


Figure 3. (a) Comparative analysis of fitness function and fitness error value for 100 iterations; (b) results for 200 iterations.

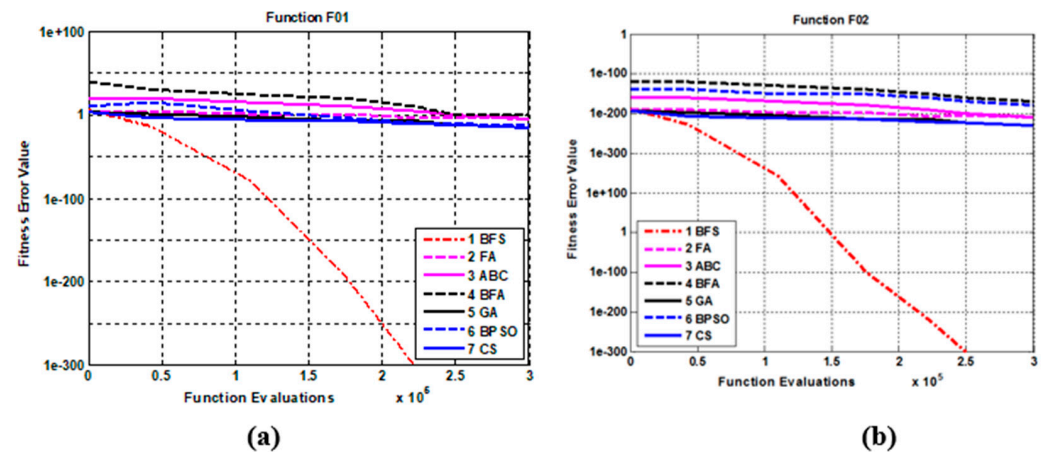


Figure 4. (a) Comparative analysis of fitness function and fitness error value for 200 iterations; (b) results for 300 iterations.

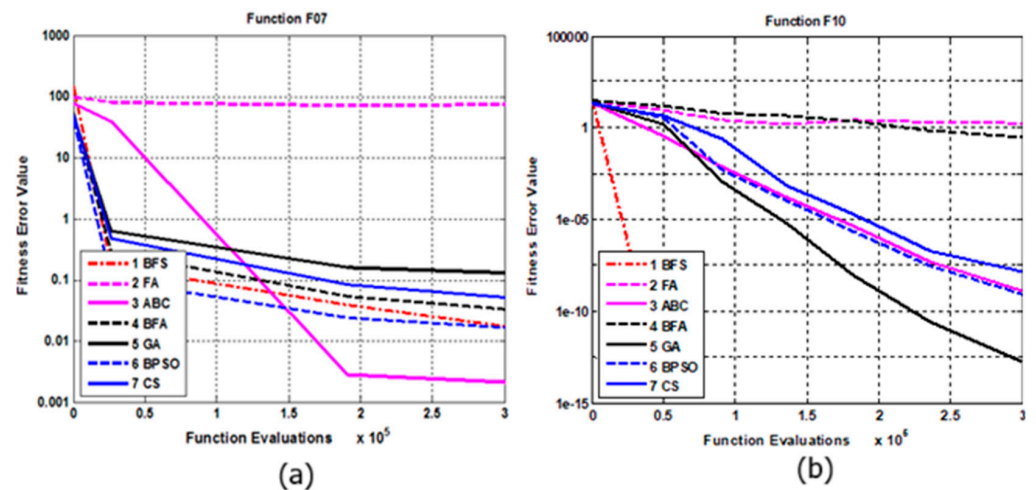


Figure 5. (a) Comparative analysis of fitness function and fitness error value for 300 iterations; (b) results for 400 iterations.

11.2. Performance Evaluation of Metaheuristic Techniques

Population-based metaheuristic algorithms are suitable for global searches because they can explore the entire search space and make use of local resources. However, traditional optimization algorithms such as dynamic programming, backtracking, and branch-and-bound may not be effective in solving complex and non-linear optimization problems. Exhaustive search is not feasible in these cases due to the exponential growth of the search space. The BFA is modeled after the behavior of bacteria defending their territory by following directional cues to reduce the search area to the target location with respect to their fitness function and error values.

The fitness error value indicates how far a candidate solution is from the optimal solution in the search space. It is the difference between the computed fitness value and the best-known or ideal fitness value. In bio-inspired or metaheuristic algorithms, the fitness function serves as an objective criterion to measure the “quality” of a solution. The fitness error value represents the difference between the current solution’s fitness value and the best or optimal solution’s fitness value. From Figure 3a,b, the function evaluation and fitness error value are compared with the approximate of between 0 (min.) and 1 (max) coordinate.

The comparison across the algorithms (GA, BFS, FA, ABC, CS, BPSO, BFA) in plot (a) reveals that BFS is the most efficient algorithm in terms of quickly reaching the lowest fitness error value. The other algorithms, FA and ABC, converge slower, indicating less

optimal performance for this function. The different parameter configurations (e.g., $N = 10$, $N = 30$) influence the convergence rate of the fitness error values. Smaller configurations (e.g., $N = 10$) converge faster initially, but larger configurations like $N = 150$ or $N = 200$ take longer to improve the solution.

Similarly, beacon nodes perform a similar function by broadcasting their signals in a specific direction to reduce the search area for non-beacon nodes. Both BFS and BNB by SRAL aim to improve the efficiency of the search process by reducing the search area and focusing on the target location. Additionally, the BFS algorithm and beacon broadcasting can be adjusted and modified based on the accumulated historical experience. Many algorithms modeled after natural processes, including BFS, FA, GSA, CS, ABC, DE, and PSO, have been proposed to address these limitations. This work aims to rigorously evaluate each algorithm, with a focus on the BFS and its ability to reach global optimum conditions. BFS is modeled after the behavior of bacteria defending their territory by following directional cues to reduce the search area to the target location. The underwater nodes also serve to increase local exploration capability which is important for maintaining a balance between exploration and exploitation, just as bacteria search abilities are improved depending on their historical performance. Then, as demonstrated in Figure 3a,b, BFS performed similarly to the employed underwater search abilities and outperformed FA, LFA, GA, BFA, and BPSO.

Figure 4a shows simulations to demonstrate that the scheduling behavior of a node with a BFS (Bacterial Foraging Search) in which the BFA is working similarly when utilizing a foraging strategy with a 60-minute Simulation Time Interval (STI). Nodes with scheduled locations and equal energy status are favored to change positions more slowly. For BFA, GA, FA, LFA, and BPSO, the cost of every node shift or exchange is controllable, yet the fitness error values continue to rise. It has been observed that the route for scheduled nodes has an error of fitness function, which is the reason the BFA is usually chosen and considered the best fit for the scheduling problems. The average rate of BFS is usually between 2 and 2.5 unit until the number of nodes reaches 100. Due to the spiral strategy of the path routing (to search for food), the BFA also helps to optimize the nodes among all metaheuristic approaches. The BFA outperformed against the STI 60 minutes to reduce the fitness error from Figures 4b and 5a, respectively. Furthermore, the analysis is also performed when the number of BFS nodes are 200 which is shown in Figure 5b.

12. Simulation and Results

This section illustrates the effectiveness of the proposed system model through two case studies: adaptive and opportunistic algorithm modes. The proposed multi-objective algorithm is simulated in NS-3 along with the support of the AquaSim framework. The results of the proposed method are analyzed and compared with the baseline schemes, e.g., RSMB, EEPDBR, UDB, 3DUL, and UL, respectively. The rationale for choosing these schemes lies in their similar nature of localization, which facilitates the scheduling and appropriate positioning of nodes. The extensive experimental setup connects 700 nodes that start from 50 until it reaches its limits, like 700 nodes.

In many UASN simulation models, nodes are randomly distributed within a 3D space (with defined boundaries for the X, Y, and Z coordinates). This approach is often used for simplicity and generalization, as it does not require any prior knowledge of the underwater environment. A 3D environment with $1600 \times 1000 \times 1000$ dimensional area is used with acoustic link 1500 m/s. All of the other important parameters like data packet size, data rates, transmission range, initial energy, and beacon message size are listed in Table 2.

Table 2. Simulation parametric values.

Simulation Parameters	Values
Number of nodes	700
Network range	$1600 \times 1000 \times 1000$
Initial energy	100 j
Bandwidth	2 Kh
Acoustic network speed/link	1500 m/s
Transmission range	200 m
Receiving	0.8 W
Data packet size	50 B
Beacon message size	52 Bits
Data rates	10 kbps
Network simulator with AquaSim framework	NS-3 (v3.35)

12.1. Protocol Analysis

This section discusses the routing schemes, Range-based Directional Beacon Forwarding (RDBF), Energy Efficient Probabilistic Depth Based Routing (EEPDBR), Using Directional Beacon (UDB) routing, Three-Dimensional Underwater Localization (3DUL), Underwater Localization (UL), and Range-free scheme based on Mobile Beacon (RSMB) [10–15], compared against the proposed protocol with various performance metrics. Deep and shallow water significantly impact node density, which increases with depth and distance. The mobility speed of the nodes also affects the performance of node density; therefore, we fix the mobility speed of the nodes, for example, 5 to 25 m/s of RREQ packets, for all compatible techniques.

12.2. Analysis of Normalized Number of RREQ and RO

In Figure 6a, for route discovery, the performance metrics directly affect the performance of the optimization method. With increasing forwarding, the RREQ is directly meant to increase the node density. For example, RREQ propagation is not restricted by the routing scheme; it may be propagated to those areas that are far from the destination nodes. Therefore, to re-ensure the node's availability and its accurate position, the flow of RREQ increased.. On average, RREQ will decrease by 10% for the proposed scheme compared to a 2% increase for EEPDBR, along with an increase of 6% UDB, 4.5% 3DUL, 4.1% UL, and 3.9% RSMB schemes, respectively. Under some constraints, it will further decrease by more than 10% compared with EEPDBR. Consequently, from the result, we determined that both SRAL and RDBF can significantly reduce the normalized numbers of RREQ against the different density of nodes. Moreover, the traditional gossip protocol shows that for the normalized number of RREQ packets, the information with varying numbers of nodes from all observing techniques like RDBF, EEPDBR, UDB, UL, 3DUL, and RSMB.

To evaluate the performance of the ROs in Figure 6b, the combination of RREQ, RREP, and RERR was included. Both have two extra overhead trends when compared with the proposed scheme. On average, the RO is reduced when the control number of RREQ is maintained for all schemes like 6.2% RDBF, 4.1% EEPDBR, 4.3% UDB, 4.0% 3DUL, 2.5% UL, and 2.9% RSMB, respectively.

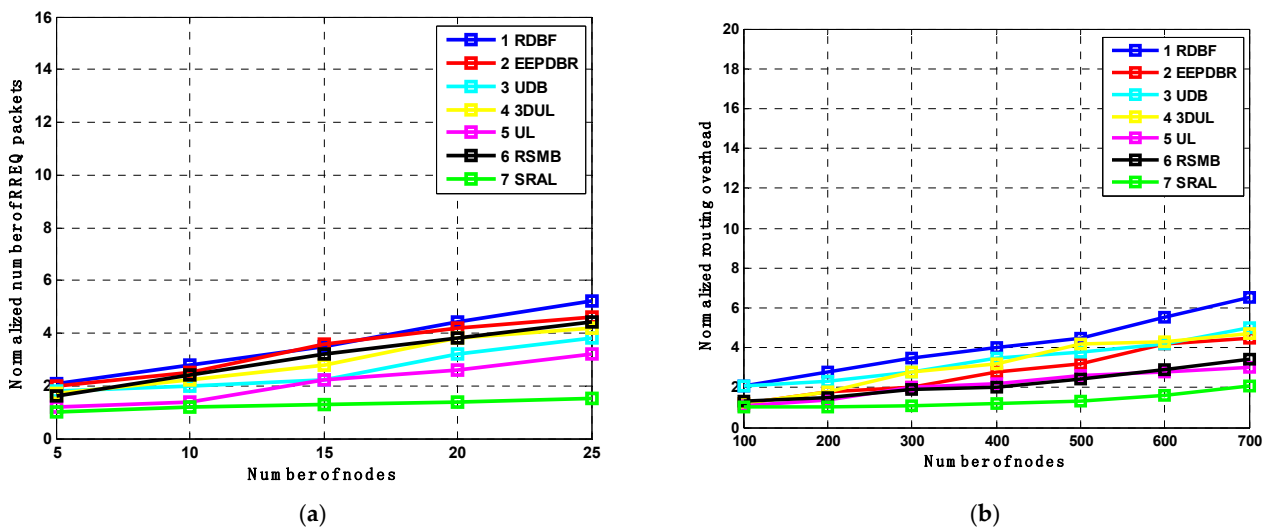


Figure 6. Performance analysis in terms of (a) RREQ, and (b) routing overhead, for various algorithms in different scenarios.

12.3. Analysis of PDR and AE2E

The Packet Delivery Ratio (PDR) for RDBF, EEPDBR, and the proposed SRAL scheme is shown in Figure 7a. It is evident that EEPDBR and SRAL exhibit a bend in their curves when the number of nodes is around 75. At this point, RDBF achieves a PDR of 89.2%, but as node density increases, the PDR drops to 88%, showing a declining trend. This decrease in PDR is characteristic of RDBF, reflecting the negative effect of higher node density. In this analysis, we did not account for the impact of dead nodes in our implementation.

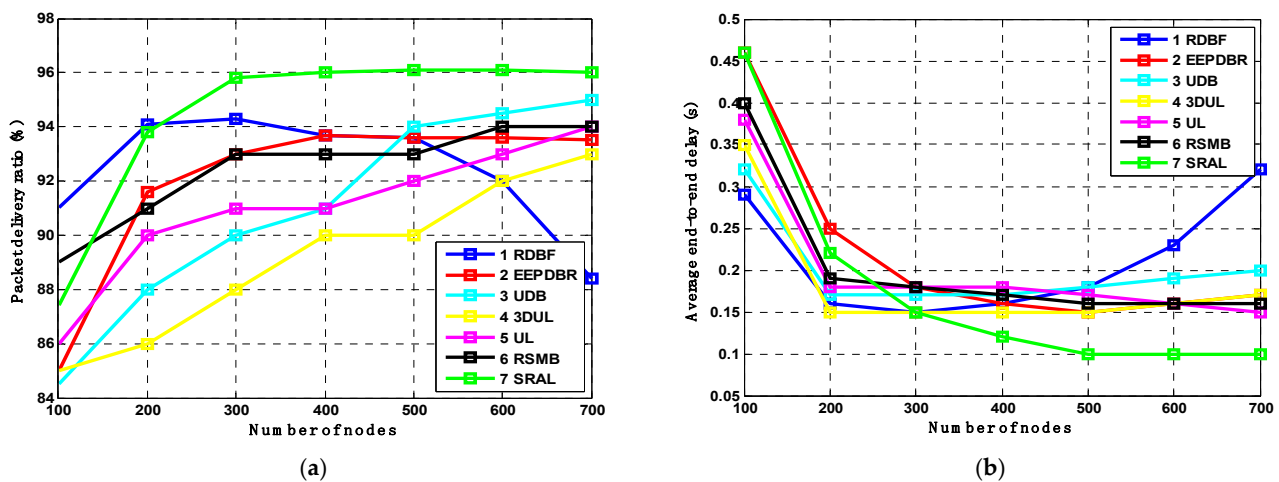


Figure 7. Performance analysis in terms of (a) PDR, and (b) Average End-to-End delay, for various algorithms in different scenarios.

When the number of nodes exceeds 125, EEPDBR shows a more stable performance, maintaining a PDR of around 93.7% before gradually decreasing to approximately 92.5%, a much smaller decline compared to RDBF's 4.1% decrease. Similarly, other benchmarks exhibit high PDRs, including UDB at 94.7%, 3DUL at 93.1%, UL at 94%, and RSMB at 94.1%. The proposed blockchain-based localization scheme achieves an even higher PDR of 96%. SRAL, in particular, maintains a nearly linear PDR performance starting from around 100 nodes (approximately 65 nodes per square kilometer) and up to 200 nodes, achieving a PDR of 96%, which surpasses that of RDBF, UDB, 3DUL, UL, and EEPDBR.

Figure 7b illustrates the Average End-to-End (AE2E) delay for CBR packets recorded at the destination. The AE2E delay follows an inverse trend to the PDR, showing that when

node distribution is sparse, SRAL does not outperform RDBF or EEPDBR. However, as the number of nodes increases (around 76 nodes per square kilometer), SRAL demonstrates a slight improvement, with delays 0.3% lower than RDBF and 0.45%, 0.23%, 0.39%, 0.40%, and 0.41% lower than EEPDBR, UDB, 3DUL, UL, and RSMB, respectively. The presence of void holes in denser networks leads to increased RREQs, RREPs, and RERRs, contributing to a sudden spike in delay, particularly for RDBF and EEPDBR, where delays exceed 3 s.

The results of PDR, normalized RREQ, RO, and AE2E delay, show that SRAL performs best in dense networks, especially when the network size increased around 370 to 480 nodes over large areas, as validated by extensive simulations.

12.4. Analysis of AE2E Delay, RREQ, RO, and PDR with Maximum Speed

This section examines the routing schemes, including RDBF, EEPDBR, RSMB, 3DUL, UL, and UDB [10–15], comparing them against the proposed protocol under varying maximum node speeds from Figure 8a–d. Both deep and shallow water environments play a critical role in node density, which increases with depth and distance. Node mobility speed significantly impacts node density performance, so for this study, the mobility speed was fixed between 5 and 25 m/s for RREQ packets across all techniques.

Figure 8a illustrates the Average End-to-End (AE2E) delay for CBR packets recorded at the destination. The AE2E delay follows an inverse trend to the PDR, showing that when node distribution is sparse, SRAL does not outperform RDBF or EEPDBR.

The Packet Delivery Ratio (PDR) for RDBF, EEPDBR, and the proposed SRAL scheme is shown in Figure 8b. It is evident that EEPDBR and SRAL exhibit a bend in their curves when the number of nodes is around 98. At this point, RDBF achieves a PDR of 89.2%, but as node density increases, its PDR drops to 88%, showing a declining trend.

The maximum speed analysis is conducted across different schemes, observing similar reactions in node mobility and the corresponding RREQ and RO ratios in Figure 8c,d. Each scheme exhibited varying behaviors; for example, RDBF and EEPDBR achieved control ratios of 21% and 27%, respectively, while UDB reached 19.6%, UL reached 15%, 3DUL reached 17%, and RSMB reached 22%. The proposed scheme, however, maintained a significantly lower control ratio of only 15%. Despite variations in node mobility speed, all routing schemes successfully managed RREQ, RO, and enhanced PDR in a structured manner, as shown in Figure 8c. However, protocols EEPDBR and SRAL effectively limit this range and show a decreasing trend compared to RDBF, UDB, 3DUL, UL, and RSMB. EEPDBR and SRAL exhibit reductions of 76.2% and 79.5%, respectively, compared to RDBF and especially UL. Under the same network conditions, these protocols show only a 2–4% increase. The normalized RO, as shown in Figure 8d, which includes RREQ, RREP, WFA, and RERR packets, demonstrates a significant reduction in RO for both EEPDBR and SRAL. As discussed in Figure 8c, the normalized number of RREQ for these protocols is reduced, leading to more than a 50% reduction in RO compared to RDBF and a 45% reduction compared to EEPDBR. Under the same network conditions, RDBF continues to increase with the rising maximum speed of up to 25 m/s, while EEPDBR and CGDBLR maintain close performance, likely due to their restricted RREQ range, contributing to the control of RO across RDBF, UDB, 3DUL, UL, and RSMB.

Figure 8c,d analyzes the normalized RREQ and RO at varying maximum speeds, demonstrating that all schemes exhibit a sufficient capability to manage and control localization errors to a certain extent. The core concept of SRAL is to optimize the routing path while limiting the number of RREQs. By reducing RREQ packet propagation, SRAL minimizes RREQ and RO. Under the same conditions, RDBF and EEPDBR exhibit significantly higher RREQ values, with RDBF at 12 and EEPDBR at 3.5, while SRAL maintains an average of around 2 across speed intervals of 5, 10, 15, 20, and 25 m/s. Conversely, RDBF shows a continuous increase in both normalized RREQ and RO values, which include packets of RREQs, RREPs, and RERRs. The same trend is observed for PDR and AE2E delay analysis for SRAL (proposed scheme) and the baseline schemes in Figure 8c,d, respectively.

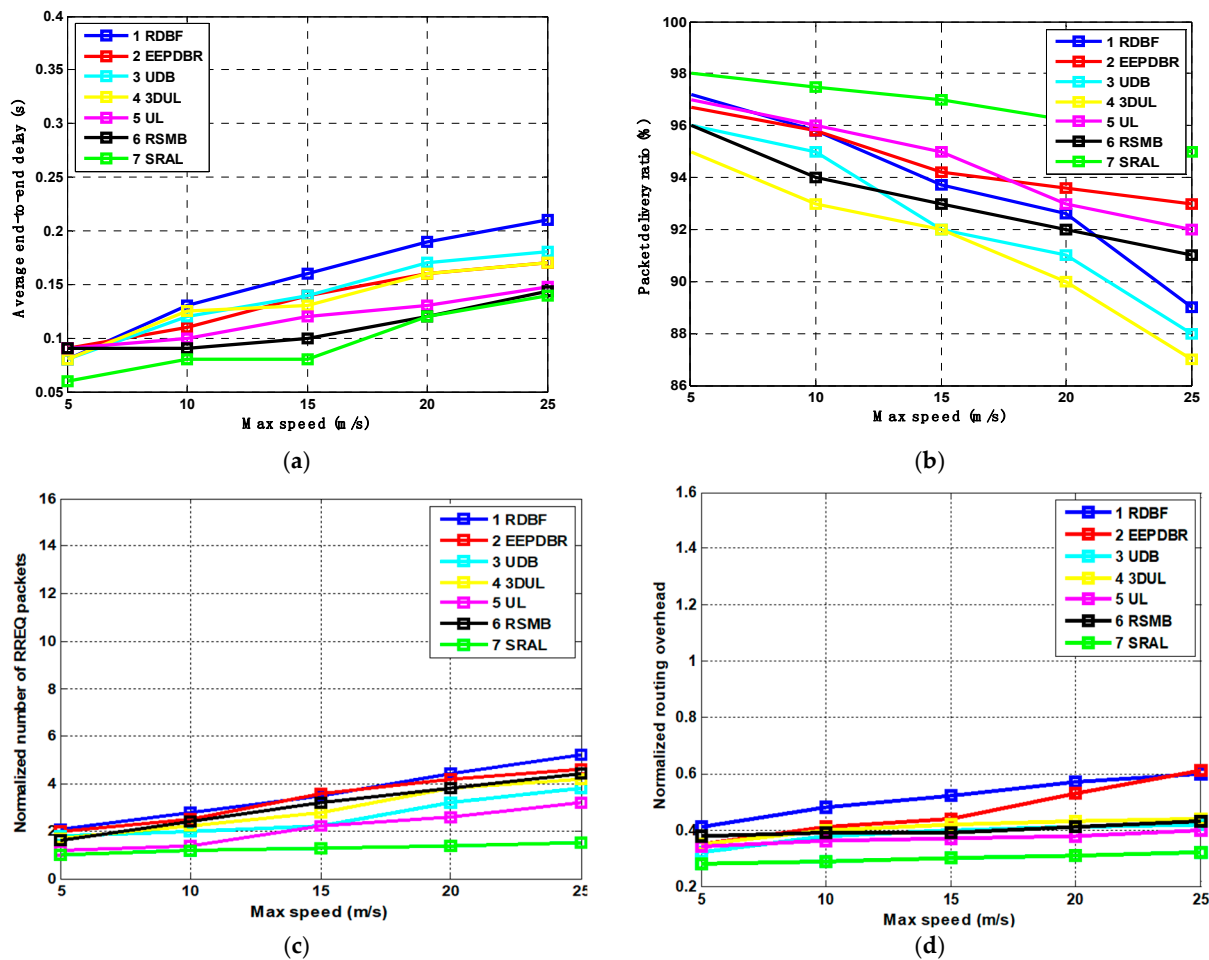


Figure 8. Comparative analysis with maximum speed of nodes for (a) AE2E delay; (b) Packet Delivery Ratio; (c) number of RREQ; and (d) number of RO.

12.5. Analysis of Localization Error and Energy Reduction

In this scenario, the speed of the nodes plays a crucial role in affecting network topology. For this experiment, the number of nodes is fixed, starting at 500 and increasing to 700, while the maximum speed varies from 5 m/s to 25 m/s. The performance evaluation of metrics, for example, normalized Route REQuest (RREQ), Routing Overhead (RO), Packet Delivery Ratio (PDR), and Average End-to-End (AE2E) delays are detailed in Table 3.

Figure 9a illustrates the localization error for all studied protocols, showing that as the network stability decreases with an increase in node speed, the PDR also declines. Despite this, the proposed SRAL protocol successfully restricts the maximum number of RREQs, ensuring that increased node density does not negatively impact performance. Furthermore, SRAL improves PDR through enhanced routing path stability. By reducing the number of RREQs, the likelihood of packet collisions decreases, ultimately enhancing PDR and minimizing localization error. Notably, SRAL achieves approximately 97% accuracy in reducing localization error, outperforming EEPDBR and RDBF, which show 2% higher localization errors as node density increases up to 500 nodes. In contrast, RDBF experiences a 21% reduction in PDR, while EEPDBR shows a 6% drop in PDR as network density and speed approach their limits. At a node count of around 400, which represents a mid-sized network, the localization errors for RDBF, EEPDBR, UDB, 3DUL, UL, and RSMB are 21%, 27%, 19.6%, 17%, 15%, and 22.2%, respectively. In contrast, the proposed SRAL only increases by 15% with the error continuing to decrease as the network size grows.

Table 3. Comparative analysis of different baseline schemes with proposed approach under different parameters.

Benchmark Schemes	Performance Parameters							
	AE2E	RO	RREQ	PDR	Localization Error	Energy Consumption	RREQ and RO Reliability	Metaheuristic Enabled
RDBF	0.3%	6.2%	6.5%	89.2%	21%	88,000 J	No	No
EEPDBR	0.45%	4.1%	4.2%	93.7%	27%	83,000 J	No	No
UDB	0.23%	6.1%	6.1%	94.9%	19.6%	81,000 J	No	No
3DUL	0.37%	4.5%	4.5%	93.0%	17%	87,000 J	No	No
UL	0.39%	4.1%	4.1%	93.8%	15%	90,000 J	No	No
RSMB	0.4%	3.9%	3.9%	94.5%	20%	81,000 J	No	No
Proposed	0.2%	2.1%	2.2%	95.5%	15%	72,000 J	Yes	Yes

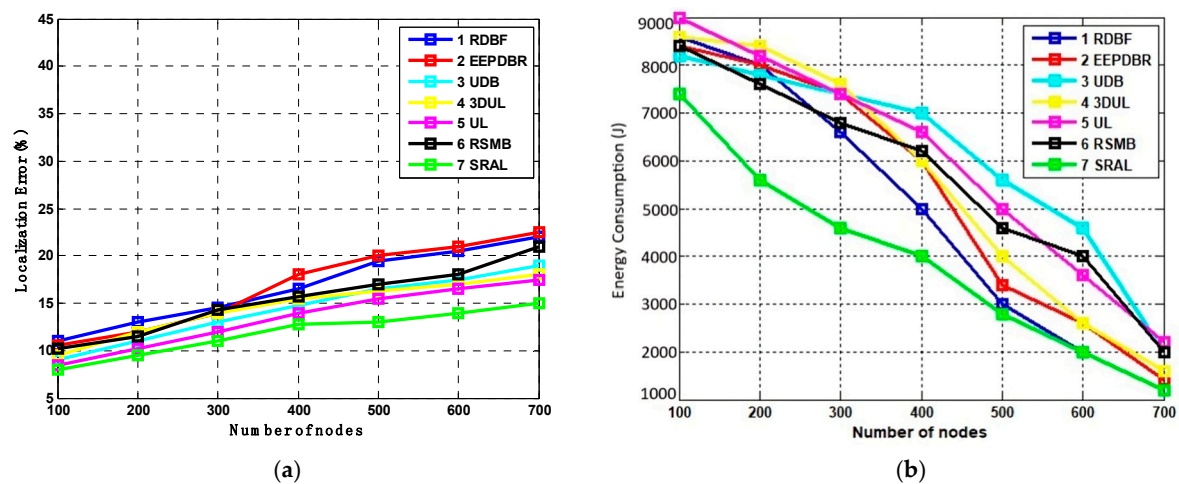
**Figure 9.** Performance analysis for (a) localization error and (b) energy consumption.

Figure 9b depicts the energy reduction (measured in Joule) for all benchmark techniques. The results show that all techniques are compatible with metaheuristic approaches when performed with hybridization. Among all baseline schemes, SRAL consumed the least energy at 72,000 J, compared to other techniques like RDBF, which consumed 90,000 J.

12.6. Impact Analysis of Scheduled and Unscheduled Nodes

Figure 10 presents a detailed comparative analysis of the impact of node scheduling on three critical performance metrics in a network: Packet Delivery Ratio (PDR), End-to-End delay, and energy consumption. The analysis is carried out for both scheduled and unscheduled nodes using five optimization algorithms: Bacterial Foraging Algorithm (BFA), Firefly Algorithm (FA), Levy Firefly Algorithm (LFA), Genetic Algorithm (GA), and Binary Particle Swarm Optimization (BPSO).

In Figure 10a, the graph illustrates the Packet Delivery Ratio (PDR) as a function of the number of nodes in the network. As the number of nodes increases from 100 to 700, the PDR improves for both scheduled and unscheduled nodes. However, it is evident that the scheduled nodes consistently exhibit a higher PDR compared to unscheduled nodes. This implies that node scheduling plays a significant role in optimizing the PDR, which directly impacts the reliability of the network. The PDR for scheduled nodes reaches around 95.5% as the number of nodes increases, while unscheduled nodes only achieve around 80%.

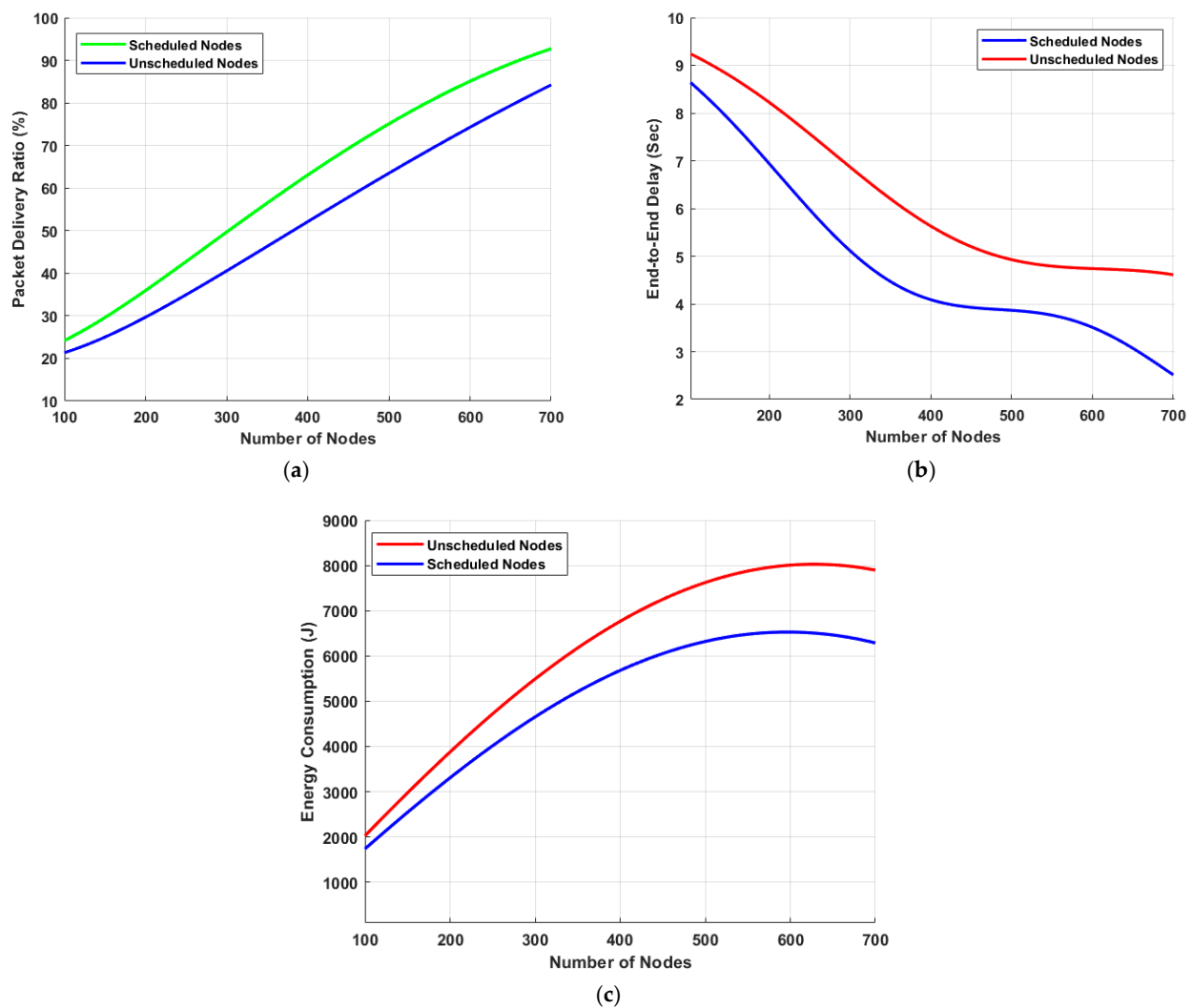


Figure 10. Impact analysis of (a) Packet Delivery Ratio; (b) End-to-End delay; and (c) energy consumption for scheduled and unscheduled nodes in network.

Figure 10b shows the effect of node scheduling on End-to-End delay. Here, we observe that scheduled nodes significantly outperform unscheduled nodes in terms of reducing delay. As the number of nodes increases, both scheduled and unscheduled nodes experience a reduction in delay. However, scheduled nodes demonstrate a much steeper decline in delay, reaching below 3 s when there are 700 nodes. In contrast, unscheduled nodes still experience higher delays, approximately 5 s, even as the number of nodes increases. This highlights the efficiency of node scheduling techniques like BFA, FA, GA, and BPSO in minimizing the latency within the network.

Lastly, Figure 10c presents the energy consumption trends for scheduled and unscheduled nodes. As the number of nodes increases, energy consumption initially rises for both scheduled and unscheduled nodes due to higher network traffic and processing requirements. However, the energy consumption for scheduled nodes is consistently lower than for unscheduled nodes. At the peak, the energy consumption for unscheduled nodes reaches nearly 8000 Joules, while scheduled nodes consume around 7000 Joules. This reduction in energy consumption through scheduling is essential for prolonging the network's operational lifetime, particularly in energy-constrained environments such as UASNs [40].

Overall, the figure demonstrates the effectiveness of scheduling techniques such as BFA, FA, LFA, GA, and BPSO in optimizing network performance. Scheduled nodes outperform unscheduled nodes across all three-performance metrics, PDR, End-to-End

delay, and energy consumption, highlighting the importance of intelligent scheduling mechanisms in enhancing network efficiency and reliability.

13. Conclusions

In this study, we proposed a hybridization of bio/nature-inspired metaheuristic algorithms to optimize node scheduling and reduce energy costs in UASNs. The integration of five hybridized algorithms such as GA, FA, LFA, BPSO, and BFA with EHO was shown to effectively improve the scheduling of nodes, leading to energy efficiency compared to existing methods. The challenges of node localization, RREQ reconstruction at the beacon level, and increased RO, along with End-to-End delays and unreliable data forwarding, have a significant impact on overall communication in underwater environments. By introducing the Scheduled Routing Algorithm for Localization (SRAL), we successfully optimized the localization and routing process, addressing the key challenges of node localization, RREQ reconstruction, and the reduction in RO, AE2E delay and energy reduction. The proposed scheme provides significant improvements in data delivery, communication reliability, and resource utilization in underwater environments. The results demonstrate that this hybridized approach holds great potential for future applications in UASNs, contributing to more efficient and secure network operations. The proposed framework, along with the hybridized metaheuristic algorithms, show great potential in improving node localization, optimizing scheduling, reducing energy costs, and enhancing reliable data delivery in the Internet of Underwater Things (IoUT)-based network. Future research should focus on real-world implementation and further validation of the proposed scheme framework across varying underwater scenarios.

Author Contributions: Conceptualization was carried out by U.D. and T.A.; methodology by U.D.; software development by S.Y.; validation was performed by S.Y., T.A., M.H.C. and S.Y.; formal analysis was led by S.Y. and T.A.; investigation by M.A.; resources were provided by E.-H.M.A.; data curation by I.Y.; original draft preparation by U.D.; review and editing were completed by M.A.; visualization was handled by I.Y.; supervision was provided by E.-H.M.A.; project administration by M.H.C.; and funding acquisition by E.-H.M.A. All authors have read and agreed to the published version of the manuscript.

Funding: This work is supported by a research grant from the Research, Development, and Innovation Authority (RDIA), Saudi Arabia, grant no.13010-Tabuk-2023-UT-R-3-1-SE.

Data Availability Statement: The data, models, or codes that support the findings of this study are available from the corresponding author upon request.

Acknowledgments: This work is supported by a research grant from the Research, Development, and Innovation Authority (RDIA), Saudi Arabia, grant no.13010-Tabuk-2023-UT-R-3-1-SE.

Conflicts of Interest: The authors declare no conflicts of interest.

References

- Kim, T.H.; Ramos, C.; Mohammed, S. Smart city and IoT. *Future Gener. Comput. Syst.* **2017**, *76*, 159–162. [\[CrossRef\]](#)
- Yang, X.S. Metaheuristic optimization: Algorithm analysis and open problems. In *International Symposium on Experimental Algorithms*; Springer: Berlin/Heidelberg, Germany, 2011; pp. 21–32.
- Nassef, A.M.; Abdelkareem, M.A.; Maghrabie, H.M.; Baroutaji, A. Review of metaheuristic optimization algorithms for power systems problems. *Sustainability* **2023**, *15*, 9434. [\[CrossRef\]](#)
- Yang, X.S. Optimization and metaheuristic algorithms in engineering. *Metaheuristics Water Geotech. Transp. Eng.* **2013**, *1*, 23.
- Abdel-Basset, M.; Abdel-Fatah, L.; Sangaiah, A.K. Metaheuristic algorithms: A comprehensive review. In *Computational Intelligence for Multimedia Big Data on the Cloud with Engineering Applications*; Academic Press: Cambridge, MA, USA, 2018; pp. 185–231.
- Husnain, G.; Anwar, S.; Sikander, G.; Ali, A.; Lim, S. A bio-inspired cluster optimization schema for efficient routing in vehicular ad hoc networks (VANETs). *Energies* **2023**, *16*, 1456. [\[CrossRef\]](#)
- Li, J.; Lei, H.; Alavi, A.H.; Wang, G.G. Elephant herding optimization: Variants, hybrids, and applications. *Mathematics* **2020**, *8*, 1415. [\[CrossRef\]](#)
- MahmoudZadeh, S.; Powers, D.; Sammut, K.; Lammas, A.; Yazdani, A.M. Optimal route planning with prioritized task scheduling for AUV missions. In *Proceedings of the 2015 IEEE International Symposium on Robotics and Intelligent Sensors (IRIS)*, Langkawi, Malaysia, 18–20 October 2015; IEEE: Piscataway, NJ, USA, 2015; pp. 7–14.

9. Silva, M.M.; Subramanian, A.; Vidal, T.; Ochi, L.S. A simple and effective metaheuristic for the minimum latency problem. *Eur. J. Oper. Res.* **2012**, *221*, 513–520. [\[CrossRef\]](#)
10. Qureshi, T.N.; Javaid, N. Enhanced adaptive geographic opportunistic routing with interference avoidance assisted with mobile sinks for underwater wireless sensor network. In Proceedings of the 2018 International Conference on Frontiers of Information Technology (FIT), Islamabad, Pakistan, 17–19 December 2018; IEEE: Piscataway, NJ, USA, 2018; pp. 367–372.
11. Zhang, M.; Cai, W. Energy-efficient depth based probabilistic routing within 2-hop neighborhood for underwater sensor networks. *IEEE Sens. Lett.* **2020**, *4*, 1–4. [\[CrossRef\]](#)
12. Luo, H.; Zhao, Y.; Guo, Z.; Liu, S.; Chen, P.; Ni, L.M. UDB: Using directional beacons for localization in underwater sensor networks. In Proceedings of the 2008 14th IEEE International Conference on Parallel and Distributed Systems, Melbourne, VIC, Australia, 8–10 December 2008; IEEE: Piscataway, NJ, USA, 2018; pp. 551–558.
13. Isik, M.T.; Akan, O.B. A three-dimensional localization algorithm for underwater acoustic sensor networks. *IEEE Trans. Wirel. Commun.* **2009**, *8*, 4457–4463. [\[CrossRef\]](#)
14. Luo, J.; Yang, Y.; Wang, Z.; Chen, Y. Localization algorithm for underwater sensor network: A review. *IEEE Internet Things J.* **2021**, *8*, 13126–13144. [\[CrossRef\]](#)
15. Nain, M.; Goyal, N.; Dhurandher, S.K.; Dave, M.; Verma, A.K.; Malik, A. A survey on node localization technologies in UWSNs: Potential solutions, recent advancements, and future directions. *Int. J. Commun. Syst.* **2024**, *37*, e5915. [\[CrossRef\]](#)
16. Leão, E.; Vasconcelos, V.; Portugal, P.; Montez, C.; Moraes, R. A hybrid beacon scheduling scheme to allow the periodic reconfiguration of large-scale cluster-tree wsns. In Proceedings of the 2018 IEEE 16th International Conference on Industrial Informatics (INDIN), Porto, Portugal, 18–20 July 2018; IEEE: Piscataway, NJ, USA, 2018; pp. 169–174.
17. Wu, P.; Su, S.; Zuo, Z.; Guo, X.; Sun, B.; Wen, X. Time difference of arrival (TDoA) localization combining weighted least squares and firefly algorithm. *Sensors* **2019**, *19*, 2554. [\[CrossRef\]](#) [\[PubMed\]](#)
18. Li, X.; Deng, Z.D.; Rauchenstein, L.T.; Carlson, T.J. Contributed Review: Source-localization algorithms and applications using time of arrival and time difference of arrival measurements. *Rev. Sci. Instrum.* **2016**, *87*, 041502. [\[CrossRef\]](#) [\[PubMed\]](#)
19. Weiss, A.; Arikan, T.; Vishnu, H.; Deane, G.B.; Singer, A.C.; Wornell, G.W. A semi-blind method for localization of underwater acoustic sources. *IEEE Trans. Signal Process.* **2022**, *70*, 3090–3106. [\[CrossRef\]](#)
20. Tuna, G.; Gungor, V.C. A survey on deployment techniques, localization algorithms, and research challenges for underwater acoustic sensor networks. *Int. J. Commun. Syst.* **2017**, *30*, e3350. [\[CrossRef\]](#)
21. Su, X.; Ullah, I.; Liu, X.; Choi, D. A review of underwater localization techniques, algorithms, and challenges. *J. Sens.* **2020**, *2020*, 6403161. [\[CrossRef\]](#)
22. Durrani, M.Y.; Tariq, R.; Aadil, F.; Maqsood, M.; Nam, Y.; Muhammad, K. Adaptive node clustering technique for smart ocean under water sensor network (SOSNET). *Sensors* **2019**, *19*, 1145. [\[CrossRef\]](#)
23. Sahoo, S.K.; Saha, A.K.; Ezugwu, A.E.; Agushaka, J.O.; Abuhaija, B.; Alsoud, A.R.; Abualigah, L. Moth flame optimization: Theory, modifications, hybridizations, and applications. *Arch. Comput. Methods Eng.* **2023**, *30*, 391–426. [\[CrossRef\]](#)
24. Faris, H.; Aljarah, I.; Al-Betar, M.A.; Mirjalili, S. Grey wolf optimizer: A review of recent variants and applications. *Neural Comput. Appl.* **2018**, *30*, 413–435. [\[CrossRef\]](#)
25. Dorigo, M.; Stützle, T. *Ant Colony Optimization: Overview and Recent Advances*; Springer International Publishing: Cham, Switzerland, 2019; pp. 311–351.
26. Saemi, B.; Goodarzian, F. Energy-efficient routing protocol for underwater wireless sensor networks using a hybrid metaheuristic algorithm. *Eng. Appl. Artif. Intell.* **2024**, *133*, 108132. [\[CrossRef\]](#)
27. Ghorpade, S.; Zennaro, M.; Chaudhari, B.S. Towards green computing: Intelligent bio-inspired agent for IoT-enabled wireless sensor networks. *Int. J. Sens. Netw.* **2021**, *35*, 121–131. [\[CrossRef\]](#)
28. Johnvictor, A.C.; Durgamahanthi, V.; Pariti Venkata, R.M.; Jethi, N. Critical review of bio-inspired optimization techniques. *Wiley Interdiscip. Rev. Comput. Stat.* **2022**, *14*, e1528. [\[CrossRef\]](#)
29. Wang, H.; Wang, W.; Sun, H.; Rahnamayan, S. Firefly algorithm with random attraction. *Int. J. Bio-Inspired Comput.* **2016**, *8*, 33–41. [\[CrossRef\]](#)
30. Wu, J.; Wang, Y.G.; Burrage, K.; Tian, Y.C.; Lawson, B.; Ding, Z. An improved firefly algorithm for global continuous optimization problems. *Expert Syst. Appl.* **2020**, *149*, 113340. [\[CrossRef\]](#)
31. Xing, B.; Gao, W.J.; Xing, B.; Gao, W.J. Bacteria inspired algorithms. In *Innovative Computational Intelligence: A Rough Guide to 134 Clever Algorithms*; Intelligent Systems Reference Library; Springer: Cham, Switzerland, 2014; Volume 62, pp. 21–38.
32. Nezamabadi-pour, H.; Rostami-Shahrabaki, M.; Maghfoori-Farsangi, M. Binary particle swarm optimization: Challenges and new solutions. *CSI J. Comput. Sci. Eng.* **2008**, *6*, 21–32.
33. Draz, U.; Chaudary, M.H.; Ali, T.; Sohail, A.; Irfan, M.; Nowakowski, G. Trust-Based Beacon Node Localization Algorithm for Underwater Networks by Exploiting Nature Inspired Meta-Heuristic Strategies. *Electronics* **2022**, *11*, 4131. [\[CrossRef\]](#)
34. Draz, U.; Yasin, S.; Irfan, M.; Ali, T.; Ali, A.; Glowacz, A.; Brumercik, F.; Glowacz, W. TANVEER: Tri-Angular Nearest Vector-Based Energy Efficient Routing for IoT-Enabled Acoustic Sensor and Actor Networks (I-ASANs). *Sensors* **2021**, *21*, 3578. [\[CrossRef\]](#)
35. Draz, U.; Yasin, S.; Ali, T.; Ali, A.; Bin Faheem, Z.; Zhang, N.; Jamal, M.H.; Suh, D.-Y. ROBINA: Rotational Orbit-Based Inter-Node Adjustment for Acoustic Routing Path in the Internet of Underwater Things (IoUTs). *Sensors* **2021**, *21*, 5968. [\[CrossRef\]](#)
36. Hwang, K.I.; Nam, S.W. Analysis and enhancement of IEEE 802.15. 4e DSME beacon scheduling model. *J. Appl. Math.* **2014**, *2014*, 934610. [\[CrossRef\]](#)

37. Smith, J.E. Estimating meme fitness in adaptive memetic algorithms for combinatorial problems. *Evol. Comput.* **2012**, *20*, 165–188. [[CrossRef](#)]
38. Draz, U.; Ali, A.; Bilal, M.; Ali, T.; Iftikhar, M.A.; Jolfaei, A.; Suh, D.Y. Energy efficient proactive routing scheme for enabling reliable communication in underwater internet of things. *IEEE Trans. Netw. Sci. Eng.* **2021**, *8*, 2934–2945. [[CrossRef](#)]
39. Abbas, M.Z.; Bakar, K.A.; Ayaz, M.; Mohamed, M.H.; Tariq, M. Hop-by-hop dynamic addressing based routing protocol for monitoring of long range underwater pipeline. *KSII Trans. Internet Inf. Syst. (TIIS)* **2017**, *11*, 731–763.
40. Kumar, N.; Kumar, V.; Ali, T.; Ayaz, M. Prolong network lifetime in the wireless sensor networks: An improved approach. *Arab. J. Sci. Eng.* **2021**, *46*, 3631–3651. [[CrossRef](#)]

Disclaimer/Publisher’s Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.