sensors

MDPI

Article

Clustered Routing Using Chaotic Genetic Algorithm with Grey Wolf Optimization to Enhance Energy Efficiency in Sensor Networks

Halimjon Khujamatov [1], Mohaideen Pitchai [2], Alibek Shamsiev [3], Abdinabi Mukhamadiyev [1,*] and Jinsoo Cho [1,*]

[1] Department of Computer Engineering, Gachon University, Seognam-daero, Sujeong-gu, Seongnam-si 1342, Gyeonggi-do, Republic of Korea; kh.khujamatov@gmail.com
[2] Department of Computer Science and Engineering, National Engineering College, Kovilpatti 627011, Tamilnadu, India; kmpcse@nec.edu.in
[3] Department of Data Communication Networks and Systems, Tashkent University of Information Technologies Named after Muhammad al-Khwarizmi, Tashkent 100200, Uzbekistan; a.shamsiyev@tuit.uz
* Correspondence: mukhamadiyev@gachon.ac.kr (A.M.); jscho@gachon.ac.kr (J.C.)

Abstract: As an alternative to flat architectures, clustering architectures are designed to minimize the total energy consumption of sensor networks. Nonetheless, sensor nodes experience increased energy consumption during data transmission, leading to a rapid depletion of energy levels as data are routed towards the base station. Although numerous strategies have been developed to address these challenges and enhance the energy efficiency of networks, the formulation of a clustering-based routing algorithm that achieves both high energy efficiency and increased packet transmission rate for large-scale sensor networks remains an NP-hard problem. Accordingly, the proposed work formulated an energy-efficient clustering mechanism using a chaotic genetic algorithm, and subsequently developed an energy-saving routing system using a bio-inspired grey wolf optimizer algorithm. The proposed chaotic genetic algorithm–grey wolf optimization (CGA-GWO) method is designed to minimize overall energy consumption by selecting energy-aware cluster heads and creating an optimal routing path to reach the base station. The simulation results demonstrate the enhanced functionality of the proposed system when associated with three more relevant systems, considering metrics such as the number of live nodes, average remaining energy level, packet delivery ratio, and overhead associated with cluster formation and routing.

Keywords: chaotic genetic algorithm; clustering; energy efficiency; grey wolf optimizer; routing; sensor networks

check for updates

Citation: Khujamatov, H.; Pitchai, M.; Shamsiev, A.; Mukhamadiyev, A.; Cho, J. Clustered Routing Using Chaotic Genetic Algorithm with Grey Wolf Optimization to Enhance Energy Efficiency in Sensor Networks. *Sensors* **2024**, *24*, 4406. https://doi.org/10.3390/s24134406

Academic Editors: Paolo Bellavista and Kit Yan Chan

Received: 14 May 2024
Revised: 5 July 2024
Accepted: 5 July 2024
Published: 7 July 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

Wireless sensor networks (WSNs) are networks composed of several nodes strategically deployed in the environment, where sensors are tasked with sensing and transmitting information to the sinks and base stations. However, these sensors have limited battery power and are very difficult to recharge following deployment [1]. Therefore, the present study's objective was to reduce the energy exhaustion of sensors and thereby extend the overall lifetime of the sensor network. To minimize network energy consumption, nodes are grouped into clusters [2], with each cluster containing a designated cluster head collecting and aggregating information from its members. Subsequently, the cluster head forwards the collected information to the sink using either a single- or multi-hop approach. The selection of cluster heads is a challenging task, and various techniques have been employed to achieve an optimal selection of cluster heads [3,4].

Because the direct transmittance of data to the base station demands higher energy expenditure from the cluster heads, clustered sensor networks require a routing protocol that prioritizes minimizing energy consumption while selecting the optimal path from the

clusters to the base station [5]. However, the construction of a clustering-based routing algorithm with high energy efficiency and increased packet transmission rate for large-scale sensor networks is an NP-hard problem [6]. In other words, the process of selecting optimal cluster heads and establishing efficient routing paths for large-scale sensor networks entails high time complexity. This issue can be efficiently solved by metaheuristic algorithms [7,8] and swarm intelligence [9,10].

This paper introduces an energy-saving clustering mechanism utilizing a chaotic genetic algorithm (CGA) coupled with the construction of an energy-efficient routing system using swarm intelligent grey wolf optimization (GWO). The proposed system, named 'chaotic genetic algorithm–grey wolf optimization (CGA-GWO)' is designed to minimize overall energy consumption by selecting energy-aware cluster heads and devising an optimal routing path to reach the base station. The key contributions and novelty of this study are summarized as follows:

- Chaotic systems, including logistic and tent maps, are employed to generate the initial population and govern the crossover and mutation processes of the genetic algorithm for the selection of cluster heads. The fitness function is specifically designed to identify chromosomes with higher residual energy and designate them as cluster heads. To accelerate genetic algorithm convergence, optimal cluster heads are selected by the elitist selection method in place of the roulette selection method.
- To determine an energy-efficient routing path to the base station, the GWO method was selected for its ability to yield optimal solutions across multiple iterations. The efficacy of these solutions was evaluated using a fitness function, where a higher fitness value signifies a reduction in overall distance, fewer hops, and minimized energy consumption along the routing path.

The remainder of this paper is organized as follows: Section 2 presents an investigation of pertinent mechanisms concerning energy-efficient cluster-based routing, along with essential background information on CGA and GWO; Section 3 provides a detailed explanation of the proposed CGA-GWO algorithm; Section 4 delves into a discussion of simulation results; finally, Section 5 concludes the paper.

2. Related Work

This section outlines recent developments in the study of energy-aware cluster routing within sensor networks. In recent years, a trend has emerged towards incorporating hybrid approaches, including swarm-based metaheuristic optimization, aimed at lowering energy depletion and thereby extending network lifetime. Ajmi et al. [11] proposed an energy-saving clustering approach known as the chicken-swarm-based genetic algorithm, which incorporates chicken-swarm-based optimization, cluster head selection, multi-weight clustering, a genetic algorithm, and cluster communication. Although this approach is notably effective in terms of energy efficacy, end-to-end latency, ratio of delivered packets, and network throughput, it might not be suitable for large-scale applications due to the possibility of communication delays.

Jia et al. [12] proposed a clustering algorithm that utilizes ant colony optimization through adaptive chaos. To disrupt the pheromone along the route, the algorithm employs chaotic mapping while enhancing the transition probability using an adaptive technique. Using this approach, the local pheromone is updated and ants are sent out to adjust the present pheromone levels using a chaos factor. Nonetheless, this algorithm does not distribute transmission tasks effectively in scenarios involving a relationship between the cluster head and sink node. Wang et al. [13] presented a cluster routing protocol that incorporates a self-organizing map neural network and firefly algorithm. Using the ant colony optimization algorithm, intercluster routing is used to select the next hop node by considering factors such as energy, distance, and node angle. The updating process of pheromones relies on the geometric coefficient of variation, and the routing path is improved by linking energy and distance.

Majeed et al. [14] introduced a fuzzy-based genetic algorithm designed for cluster formation and cluster head selection, achieving fuzzy rule optimization and output value modification for membership in fuzzy logic. In addition, the ant colony process was employed to estimate the shortest path from each cluster head to the base node. A limitation of this approach lies in its focus on short routing, which leads to a suboptimal load balance in the network. Ram et al. [15] introduced an effective approach for selecting cluster heads using the K-genetic algorithm. In this method, sensors are assembled into clusters based on their locations using k-means clustering, and a genetic algorithm is implemented to determine the most suitable cluster head. A trust-based firefly algorithm is applied to ensure secure and optimal routing.

Agrawal [16] proposed a method for selecting cluster heads through the application of grey wolf swarm intelligence. The conventional GWO algorithm was tailored to the specific objective of selecting cluster heads in sensor networks, where the cluster formation function is determined by various factors including the balancing feature of the cluster heads, residual energy, average distance of each cluster, and sink distance. Joseph et al. [17] proposed an enhanced chaotic GWO algorithm to enhance energy awareness in sensor networks, focusing on the selection of optimal cluster heads and routing paths. This approach aims to increase the percentage of packets delivered inside the sensor network while reducing node energy consumption.

Patra et al. [18] introduced an energy-efficient clustering technique that employs a genetic algorithm. For shortest-path routing, they utilized ad hoc multipath routing by adapting grey wolf intelligence. Their approach identifies an optimal multipath route from paths generated during on-demand routing, where gray wolf intelligence is used to anticipate the optimal path. Singh et al. [19] introduced an energy-saving routing algorithm that employs fuzzy GWO, aiming to minimize power consumption and achieve a balance in power usage among nodes in the sensor network. Additionally, the algorithm facilitates the reliable collection of link data by designated nodes and enhances throughput by mitigating traffic resulting from buffer occupancy in the nodes.

Gunjan et al. [20] proposed a protocol based on a metaheuristic approach for clustering and routing in sensor networks, where a heuristic search is deployed to select cluster heads by considering three distinct fitness factors: the residual energy of each cluster head, the distance between each cluster head and the sink, and intercluster isolation. To direct data towards the base station, three additional fitness functions are applied targeting the residual energy of the next hops, distance to the next hop node from each cluster head, and total hop count. Liu et al. [21] introduced the LEACH protocol based on a genetic algorithm comprising three phases for each round: preparation, setup, and data transmission. During the preparation phase, all nodes transmit messages indicating their candidacy statuses as cluster heads and share their locations with the base station. Using a genetic algorithm, the base station determines the optimal probability of each node becoming a cluster head. Subsequently, in the setup phase, the base station broadcasts a message containing the probability of all nodes forming clusters. The LEACH protocol was adopted for the setup and data transmission phases of each round of the protocol.

In all aforementioned studies, the genetic algorithm was used to encode clustering and routing processes within a single chromosome, resulting in the global estimation of energy. In addition, previous hops were not considered as parameters in the fitness functions for network load balancing. At present, there is a notable focus on leveraging the combined power of the chaos genetic algorithm (CGA) and grey wolf optimization (GWO) to enhance energy efficiency in sensor networks. This combination is gaining traction as a promising strategy to optimize energy consumption, extend network lifetime, and improve overall performance in sensor network deployments. The chaos genetic algorithm introduces chaotic dynamics into the genetic algorithm framework, promoting exploration and preventing premature convergence. This chaotic exploration enables CGA to effectively search for optimal solutions in complex and dynamic environments, making it well suited for addressing the challenges inherent in energy optimization in sensor networks.

On the other hand, grey wolf optimization mimics the social hierarchy and hunting behavior of grey wolves to optimize solutions in a distributed manner. GWO's hierarchical search strategy allows for efficient exploration of the solution space, enabling the identification of energy-efficient configurations and resource allocation strategies within sensor networks. The synergy between CGA and GWO has led to significant strides in enhancing energy efficiency within sensor networks. This progress is evident across several avenues, notably through the development of energy-aware clustering mechanisms, routing protocols, and energy-harvesting optimization techniques. In contrast, the presently proposed CGA-GWO algorithm encodes clustering and routing in different chromosomes while using energy consumption and previous hops as parameters to evaluate the fitness function.

2.1. Chaotic Genetic Algorithm

The genetic algorithm, much like other evolutionary algorithms, is founded upon randomness. Consequently, local convergence and high tolerance for results owing to randomness represent primary drawbacks of the genetic algorithm [22]. To enhance the performance of the conventional genetic algorithm, chaotic maps are employed in place of conventional random functions to generate random values. As a nonlinear phenomenon in nature, chaos can be used to mirror system complexity. Furthermore, the role of randomness introduced by chaotic dynamics resembles that of random variables. Optimization methods leverage chaotic systems to generate data with random values [23].

Chaotic systems, such as Hennon and logistic maps, can substitute for randomness in the initial population, as well as the mutation, and crossover processes of the genetic algorithm. As the initial population generates random solutions, the randomness in the crossover creates new offspring. The randomness inherent to the mutations alters some genes in the offspring [24]. Thus, the significant attributes of chaotic systems—namely pseudo-randomness and determinism—serve as compelling reasons for employing them in place of random processes within the chaotic genetic algorithm. This choice helps circumvent local convergence issues, thereby enhancing the traditional genetic algorithm's performance [25].

2.2. Grey Wolf Optimization

Mirjalili et al. [26] introduced the GWO algorithm inspired by the stalking behavior of grey wolves in nature. Specifically, gray wolves can be classified as alpha, beta, delta, or omega. The GWO algorithm involves three steps: surrounding prey, hunting prey, and attacking prey.

2.2.1. Surrounding Prey

The surrounding behavior of grey wolves is mathematically represented by Equations (1) and (2):

$$\vec{D} = \left| \vec{V}.\vec{P_p}(t) - \vec{P}(t) \right| \quad (1)$$

$$\vec{P}(t+1) = \vec{P_p}(t) - \vec{U}.\vec{D} \quad (2)$$

where 't' denotes the round number, $\vec{U}$ and $\vec{V}$ are vectors, $\vec{P_p}$ denotes the location of the prey, $\vec{P}$ is the location of the wolf, and $\vec{D}$ denotes the gap between the wolf and prey. The coefficient vectors $\vec{U}$ and $\vec{V}$ are computed using Equation (3):

$$\vec{U} = 2\,\vec{a}\,.\vec{r_1} - \vec{a} \quad (3)$$

$$\vec{V} = 2\,.\vec{r_2} \quad (4)$$

where $\vec{r_1}$ and $\vec{r_2}$ are vectors with values in a range of [0, 1], and the value of $\vec{a}$ increases linearly from 0 to 2 throughout the iterations. The grey wolf updates its position by modifying $\vec{U}$ and $\vec{V}$ along with random vectors $\vec{r_1}$ and $\vec{r_2}$ and which aid the wolf in approaching the prey.

2.2.2. Hunting Prey

The hunting strategy of the GWO is orchestrated by optimal solutions provided by the alpha, beta, and delta wolves, enabling the prediction of prey locations. The locations of omega wolves are dynamically modified according to the coordinates of the alpha, beta, and delta wolves. The following equation can be used to represent this process:

$$\vec{Y}_\alpha = |\vec{N_1}.\vec{P}_\alpha - \vec{P}|, \vec{Y}_\beta = |\vec{N_2}.\vec{P}_\beta - \vec{P}|, \vec{Y}_\delta = |\vec{N_3}.\vec{P}_\delta - \vec{P}| \quad (5)$$

where $\vec{P}$ denotes the wolf's current position; $\vec{Y}_\alpha$, $\vec{Y}_\beta$, and $\vec{Y}_\delta$ denote the current locations of alpha, beta, and delta wolves, respectively; $\vec{D}_\alpha$, $\vec{D}_\beta$, and $\vec{D}_\delta$ represent the updated locations of alpha, beta, and delta wolves, respectively; and $\vec{N}_1$, $\vec{N}_2$, and $\vec{N}_3$ are coefficient vectors. The absolute positions of wolves $\vec{P}_1$, $\vec{P}_2$, and $\vec{P}_3$ are calculated using the following equations:

$$\vec{P}_1 = \vec{P}_\alpha - \vec{R}_1.(\vec{Y}_\alpha), \vec{P}_2 = \vec{P}_\beta - \vec{R}_2.(\vec{Y}_\beta), \vec{P}_3 = \vec{P}_\delta - \vec{R}_3.(\vec{Y}_\delta) \quad (6)$$

$$\vec{P}(t+1) = \frac{\vec{P}_1 + \vec{P}_2 + \vec{P}_3}{3} \quad (7)$$

where $\vec{R}_1$, $\vec{R}_2$, and $\vec{R}_3$ represent random vectors, and t specifies the current iteration.

2.2.3. Attacking Prey

To execute an attack, the parameter $\vec{a}$ is reduced and the values of vector $\vec{U}$ are concurrently decreased by a within the interval of $[-2a, 2a]$. The parameter a gradually diminished from 2 to 0 in successive iterations. If the random values of $\vec{U}$ fall within the interval of $[-1, 1]$, the subsequent location of the searching agent will be the point lying between its present position and the location of the prey. If $|U| < 1$, the wolf is compelled to initiate an attack on the prey. Conversely, if $|U| > 1$, the wolves separate from each other in their pursuit of locating the prey. In the proposed algorithm, cluster heads are represented by grey wolves, with the base station serving as the prey.

3. Proposed Method

The hierarchical architecture of a WSN is depicted in Figure 1. Sensor nodes are arranged into discrete clusters, where the head of each cluster gathers information from all member nodes. The information collected by the cluster head is transmitted to the base station in a multi-hop manner. This type of hierarchical architecture is known as the clustering architecture of a WSN. The proposed method used two algorithms for routing and clustering, respectively. GWO is used to identify the best route to reach the base station from each cluster head, and a chaotic genetic algorithm is employed for clustering. Because the sensor nodes operate on batteries, it is crucial to optimize both energy usage and sensor lifetime when designing the clustering and routing processes.

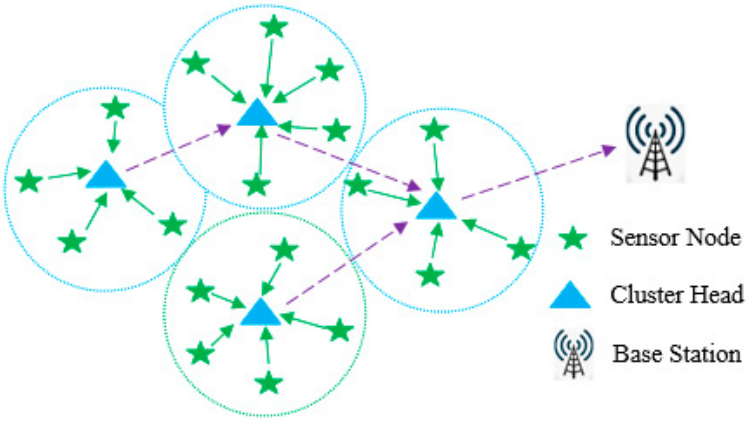


Figure 1. Cluster-based WSN.

3.1. Clustering Using Chaotic Genetic Algorithm

The proposed method generates clusters using various phases of a genetic algorithm. Chaotic dynamics are used in the initial population, crossover, and mutation phases of the algorithm. The initial population process uses chaotic systems, such as logistic maps, whereas the crossover and mutation processes use tent maps. Figure 2 illustrates the operation of the chaotic genetic algorithm. First, solutions are randomly initialized to generate an initial population, with each solution composed of one or more chromosomes consisting of a string or collection of characters. After the initial population is generated, higher-performing individuals are selected using the fitness function. Next, a crossover operation is performed on two individuals at random such that their parents can create two new individuals as offspring.

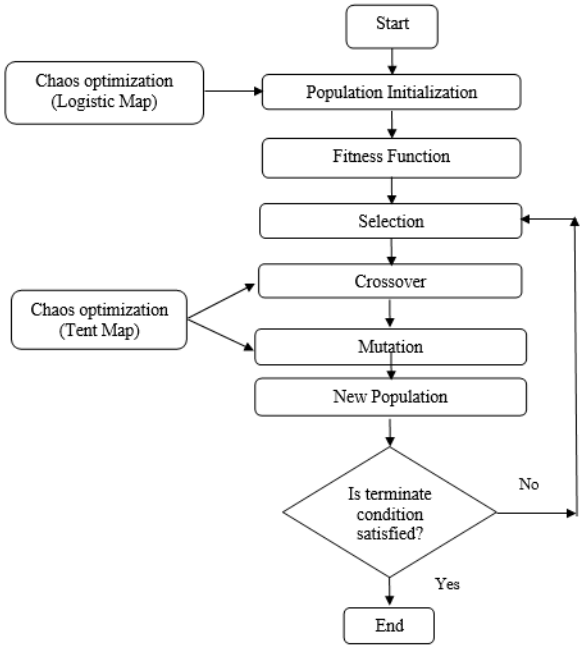


Figure 2. Chaotic genetic algorithm.

A mutation procedure is performed to create a new generation, with the fitness levels of the parents and new children evaluated using the fitness function. Parents are formed through selection, and the next generation is created through mutation. Towards the end, the algorithm mixes the new population with half of the individuals who perform well. If the new population does not satisfy the termination criterion, the algorithm is passed to the next generation; otherwise, it terminates. The objective of embedding chaotic features

into a genetic algorithm is to improve search capabilities and introduce randomness into the optimization process.

3.1.1. Chromosome Representation

With sensor nodes represented as genes, chromosomes are used to represent a random selection of nodes from the network. Basic sensor nodes like Mica2/MicaZ motes were utilized in this study. These nodes are known for their low power consumption and straightforward sensing capabilities. They are typically deployed in large quantities to provide extensive coverage across an area. As a solution to identifying suitable cluster heads, real-number coding is used to generate chromosomes. Each chromosome has a length of $n - 1$, where n represents the total number of sensors as well as the base station. Each node in the network is identified using a real number, with the base station identified as 1 (ID(sink) = 1), each cluster head node identified as $ID(ch_i) = I + 1$, and each cluster member node identified as $ID(cm_i) = i + 1 + n_{CH}$. The size of each cluster head is indicated by n_{CH}. For example, in the WSN depicted in Figure 3, the identities of cluster heads 2, 3, and 4 are 1, 2, and 3, respectively, while that of member node 6 is 2. Figure 4 shows the process of chromosome coding for cluster head selection in the network depicted in Figure 3. The network has 14 nodes, with nodes 2, 3, and 4 being cluster heads and the remaining nodes being cluster members.

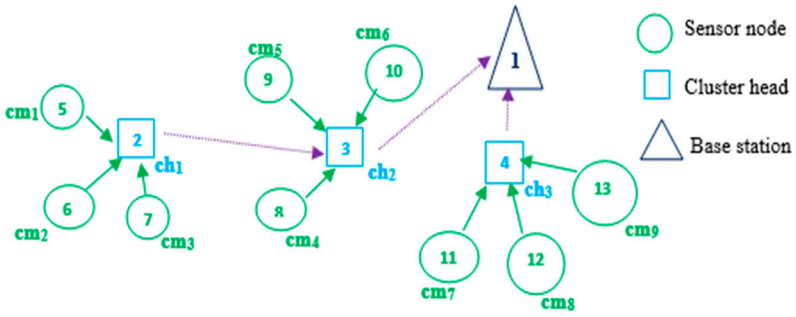


Figure 3. WSN clustering.

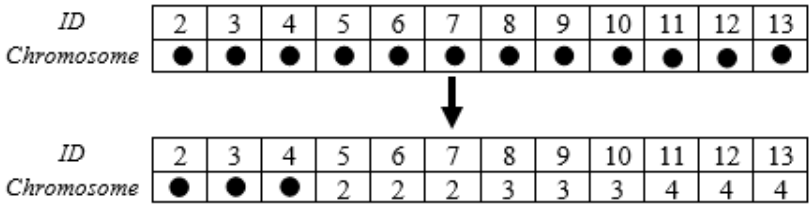


Figure 4. Real-number coding of chromosome for cluster head selection.

3.1.2. Population Initialization

There are q randomly produced chromosomes in the initial population P_{init}, where each chromosome corresponds to a potential solution. The initial population is denoted as

$$P_{init} = \{chr_{m1}, chr_{m2}, \ldots\ldots, chr_{mi}, \ldots\ldots, chr_{mn}\} \text{ and } chr_{mi} \epsilon [a_{mi}, b_{mi}] \quad (8)$$

where n represents the dimension of variables chr_{mi}, m = 1, 2, 3, ...N, and i = 1, 2, 3 .....n.

The local convergence and high tolerance of a genetic algorithm result from randomness. Hence, the rule of randomness given by chaotic dynamics replaces the random process in the population initialization. Chaos offers pseudo-randomness and ergodicity characteristics in a random process, thereby avoiding the premature convergence of results. Several chaotic maps—including the Chebyshev, Gauss, logistic, piecewise, singer, and tent maps—are used to generate the initial population. The tent map is used during the crossover and mutation processes, whereas the logistic map is used to create the initial

population. As expressed by Equation (9), the logistic map yields the chaotic variables, where *k* represents the iteration number and r represents the control parameter.

$$x_i^{k+1} = rx_i^k\left(1 - x_i^k\right) \text{ where } x_i^0 \epsilon [0,1] \text{ and } i = 1, 2 \ldots\ldots n,\ k = 0, 1, 2 \ldots\ldots \quad (9)$$

The value of r lies within [0, 4], where r = 4 corresponds to a completely chaotic system. After the value of r is determined, the chaotic-type initial value is determined using Equation (10):

$$x_{mi}^0 = (chr_{mi} - a_{mi})/(b_{mi} - a_{mi}) \quad (10)$$

The chaotic variables $x_i^0, x_i^1, x_i^2, \ldots\ldots\ldots, x_i^k$ are generated according to Equation (9) and subsequently mapped from the chaos space to the solution space using Equation (11).

$$chr_{mi} = a_{mi} + (b_{mi} - a_{mi})x_{mi}^k \text{ where } m = 1, 2, \ldots N \text{ and } i = 1, 2 \ldots\ldots\ldots n. \quad (11)$$

Thus, the initial population is created based on the locations of the n values in the chaotic environment following the aforementioned calculations.

3.1.3. Fitness Function

Chromosome quality in a genetic algorithm is assessed using a fitness function. Because the cluster heads send gathered information to the base station via several hops, the maximum energy remaining in each cluster head node corresponds to chromosome fitness. As the energy depletion of cluster heads during periods of high load determines the network's energy efficiency, reducing the energy consumption of the cluster heads is crucial for extending the network lifetime. The remaining energy of the nodes is estimated using a first-order radio (FOR) model. Among a set of cluster head candidates, choose the one which satisfies the criterion. Through this way of selecting cluster heads, the load is balanced among the nodes in the network. The fitness value of each chromosome (Chr_i) is given by

$$fit(Chr_i) = E_{res}(i) \quad (12)$$

The i^{th} node's residual energy ($E_{res}(i)$) is calculated using Equation (13):

$$E_{res}(i) = E_{init}(i) - E_{cons}(i) \quad (13)$$

where $E_{init}(i)$ is the initial energy of i^{th} node, and $E_{cons}(i)$ represents the energy consumed by the nodes. The energy consumption ($E_{cons}(i)$) of i^{th} node is calculated using Equation (14):

$$E_{cons}(i) = E_{rx}(i) - E_{tx}(i) \quad (14)$$

where $E_{rx}(i)$ and $E_{tx}(i)$ are the reception and transmission energies of i^{th} node respectively. The reception energy ($E_{rx}(i)$) of i^{th} node is calculated using Equation (15):

$$E_{rx}(i) = pl \times E_{elec}(i) \quad (15)$$

where pl is the length of the packet to be received and $E_{elec}(i)$ is the electrical energy consumption of i^{th} node. The transmission energy of the nodes is calculated using Equation (16) for the free space model and Equation (17) for the multispace model.

$$E_{tx}(i) = pl \times E_{elec}(i) + pl \times \varepsilon_{fs} \times d^2, d < d_0 \quad (16)$$

$$E_{tx}(i) = pl \times E_{elec}(i) + pl \times \varepsilon_{mp} \times d^4, d > d_0 \quad (17)$$

where $d_0 = \frac{\varepsilon_{fs}}{\varepsilon_{mp}}$. Here d is the distance between the nodes, ε_{fs} is the free space energy and, ε_{mp} is the multispace energy.

Thus, chromosomes that consume lower amounts of energy or have maximum residual energy are selected as cluster heads. Chromosomes with a high residual energy are

associated with higher fitness values; hence, the fitness function is directly proportional to the residual energy of each node, as shown in Equation (18). Therefore, the solution with maximum fitness value after the given number of iterations is considered as the best solution.

$$\mathrm{fit}(i) \propto E_{res}(i) \tag{18}$$

After selecting the cluster heads, they broadcast a message called CH_DEC to all other nodes in the network. This message contains the ID and location of the sender cluster head within the deployment area. Upon receiving this broadcast, every node in the network prepares a candidate list based on the information provided. This list includes the ID of each cluster head and a corresponding cost value. The cost value is determined by factors such as the distance between the node and the cluster head, the maximum transmission range of the network, and the initial and residual energy levels of the node. It can be defined as

$$c(i) = (E_{res}(i)/E_{init}(i)) + (D_{ij}/D_{max}) \tag{19}$$

where D_{ij} is the distance between i^{th} node to j^{th} cluster head and D_{max} is the maximum transmission range. The nodes join with cluster head based on the cost value. Cluster heads are selected based on having the lowest cost value, which is determined by factors like distance, network range, and energy levels. Additionally, a balancing factor is employed to control the cluster size, considering the total number of nodes and the number of selected cluster heads. Nodes then send a request message, JOIN_MESG, to the identified cluster head. Upon receiving this request, the cluster head evaluates whether it can include the node in its cluster. If it can, it responds with a JOIN_ACN message. Otherwise, it sends a JOIN_RJ message. If a node receives a JOIN_RJ message, it should seek another cluster head with the lowest cost value from the candidate list. This process continues until the node successfully joins a cluster. This way, clusters are formed based on the calculated costs.

3.1.4. Selection

Chromosomes with the highest fitness values are selected as parent chromosomes for the crossover and mutation operators. The proposed algorithm selects high-quality chromosomes to pass on from elite individuals directly to the next generation using elitist selection. The elitist selection approach improves convergence speed over the roulette wheel selection method by directly reproducing elite individuals in each successive generation.

3.1.5. Crossover and Mutation

The tent-map chaotic system is used for crossover and mutation operations. Chaotic behavior can influence the mixing of genetic material in a manner that is less predictable than traditional crossover methods. Here, the two child chromosomes $x_{c1} = (x_{c1}^0, x_{c1}^1, x_{c1}^2, \ldots\ldots\ldots, x_{c1}^k)$ and $x_{c2} = (x_{c2}^0, x_{c2}^1, x_{c2}^2, \ldots\ldots\ldots, x_{c2}^k)$ are generated from a pair of parent chromosomes $x_{p1} = (x_{p1}^0, x_{p1}^1, x_{p1}^2, \ldots\ldots\ldots, x_{p1}^k)$ and $x_{p2} = (x_{p2}^0, x_{p2}^1, x_{p2}^2, \ldots\ldots\ldots, x_{p2}^k)$ as follows:

$$x_{c1}^i = \frac{1}{2}[(1-\alpha)x_{p1}^i + (1+\alpha)x_{p2}^i] \tag{20}$$

$$x_{c2}^i = \frac{1}{2}[(1+\alpha)x_{p1}^i + (1-\alpha)x_{p2}^i] \tag{21}$$

where α is generated as

$$\alpha = \begin{cases} (2u)^{\frac{1}{(\eta_c+1)}}, & \text{if } u \leq 0.5 \\ \left(\frac{1}{2(1-u)}\right)^{\frac{1}{(\eta_c+1)}} & \text{otherwise} \end{cases} \tag{22}$$

Here, u is a random integer between 0 and 1. The distribution index of the crossover operator is denoted by η_c. The value of u is generated using the tent map, with the k^{th} iteration of u calculated as follows:

$$u^{k+1} = \begin{cases} ru^k & \text{if } u^k < \frac{1}{2} \\ r(1-u^k) & \text{if } u^k > \frac{1}{2} \end{cases} \text{ where } u^0 \epsilon [0,1] \text{ and } k = 0, 1, 2 \ldots\ldots \quad (23)$$

where r is the control parameter satisfying $1 < r < 2$, thereby ensuring chaotic behavior. For solution x_i, the mutation operation is expressed as follows:

$$x_i^* = x_i + (x_i^u - x_i^l)\delta_i \quad (24)$$

where x_i^u and x_i^l are the upper and lower bounds of x_i, respectively, and

$$\delta_i = \begin{cases} (2u_i)^{\frac{1}{(\eta_m+1)}} & \text{if } u_i < 0.5 \\ 1 - (2(1-u_i)))^{\frac{1}{(\eta_m+1)}} & \text{otherwise} \end{cases} \quad (25)$$

where 'u_i' is a random integer between 0 and 1. The distribution index of the mutation operator is denoted by η_m. The value of 'u_i' is generated using tent map and the k^{th} iteration of u_i is calculated as follows.

$$u_i^{k+1} = \begin{cases} ru_i^k & \text{if } u_i^k < \frac{1}{2} \\ r(1-u_i^k) & \text{if } u_i^k > \frac{1}{2} \end{cases} \text{ where } u_i^0 \epsilon [0,1] \text{ and } i = 1, 2 \ldots\ldots n, \; k = 0, 1, 2 \ldots\ldots \quad (26)$$

where 'r' is the control parameter and $1 < r < 2$, the system exhibits chaotic behavior. The generated child chromosome is finalized as a cluster head for the given network. Thus, cluster heads are selected using a chaotic genetic algorithm that considers the residual energy of the nodes during each network round.

3.2. Routing Using Grey Wolf Optimization

In GWO, alpha, beta, delta, and omega wolves represent different levels of the family structure, mimicking the social environment and hunting tactics of real-world grey wolves. These wolves work together to optimize the fitness environment with the goal of determining the most suitable path for a particular problem. There are three phases of routing using GWO:

3.2.1. Initialization of Wolves

A population of grey wolves is initialized, where each wolf corresponds to a routing path for the problem and the positions of wolves on the routing path represent potential solutions. The solution must establish a route to reach the base station through the network's subsequent cluster heads. Because the wolves' positions are randomly generated upon initialization, a random number is assigned to each cluster head. Letting $P_i = \{CH_1, CH_2, \ldots\ldots\ldots\ldots CH_n)$ be the i^{th} solution in the population, CH_i represents i^{th} cluster head and n denotes the number of clusters. The population of grey wolves CH_i (i = 1, 2, …, n) is randomly initiated and each cluster head is initialized with a random number CH_i = rand(0,1). CH_j, which represents the next succeeding cluster head of CH_i in the routing path to reach the base station, is determined according to the following equation:

$$CH_j = \text{Choose}(\text{Neighbors}(CH_i), k) \quad (27)$$

where the Choose function returns the k^{th} cluster head from the neighbors of CH_i, and the Neighbors function determines the potential neighboring cluster heads. The value of k is determined as

$$k = CH_i \times |\text{Neighbors}(CH_i)| \quad (28)$$

3.2.2. Routing Fitness Evaluation

The energy usage, length, and hop count of the routing path are key parameters that determine the fitness value for routing. Based on these considerations, a fitness function was formulated to determine the most efficient route for arriving at the base station.

The energy usage EP_i of the path from the cluster head CH_i to the base station is calculated as follows:

$$\mathrm{EP_i} = \sum_{\mathrm{i=1}}^{\mathrm{m}} \frac{1}{\mathrm{E_{nexthop(CH_i)}}} \tag{29}$$

where $\mathrm{E_{nexthop(CH_i)}}$ is the residual energy of the subsequent hop from the cluster head $\mathrm{CH_i}$.

Equation (30) defines the route length required to reach the base station $\mathrm{DP_i}$ from the cluster head $\mathrm{CH_i}$ as follows:

$$\mathrm{DP_i} = \sum_{\mathrm{i=1}}^{\mathrm{m}} \mathrm{d(CH_i, nexthop(CH_i))} \tag{30}$$

where $\mathrm{nexthop(CH_i)}$ denotes the cluster head identified as the next succeeding hop for $\mathrm{CH_i}$. The distance between $\mathrm{CH_i}$ and $\mathrm{CH_j}$ is represented by the function $\mathrm{d\left(CH_i, CH_j\right)}$.

Because additional cluster head hops in a given path result in higher energy consumption, the total number of hops between the base station $\mathrm{HP_i}$ and cluster head $\mathrm{CH_i}$ is defined as follows:

$$\mathrm{HP_i} = \sum_{\mathrm{i=1}}^{\mathrm{m}} \mathrm{nexthopCount(CH_i)} \tag{31}$$

where $\mathrm{nexthopCount(CH_i)}$ denotes the hop length between the base station and cluster head.

The selection of a routing path considers variables including path length, number of hops, and energy consumption. Specifically, the routing path with the highest fitness value is associated with a shorter total distance, fewer hops, and lower energy consumption per hop. Thus, the routing fitness value RFV is evaluated as

$$\mathrm{RFV} = (\alpha \times \mathrm{EP}) + (\beta \times \mathrm{DP}) + (\gamma \times \mathrm{HP}) \tag{32}$$

where α, β, and γ are weighing factors satisfying $\alpha + \beta + \gamma = 1$.

3.2.3. Wolf Position Updating

The positions of beta, delta, and alpha wolves are updated according to Equations (5)–(7), representing the first-, second-, and third-best solutions, respectively. These updated positions are determined by averaging the current locations of the wolves. To standardize each wolf's most recent position within the interval [0, 1], values falling outside that interval are adjusted as follows:

- If $\mathrm{pos(CH_i)} \leq 0$ then $\mathrm{pos(CH_i)} = \max(\mathrm{r1, r2, r3})$;
- If $\mathrm{pos(CH_i)} \geq 1$ then $\mathrm{pos(CH_i)} = 1$.

4. Simulation and Results

The proposed chaotic genetic algorithm was evaluated using a MATLAB simulation under the assertion that the chaotic dynamics used in the algorithm are sufficient to achieve energy-efficient clustering. However, evaluating the efficacy of GWO-based routing involves assessing key parameters—including network lifetime, energy efficiency, throughput, and routing overhead—using a network simulation tool (Network Simulator 2.35). Although MATLAB is used for simulating the clustering process, the performance graphs were created using NS2.35. In NS2.35, various network scenarios are generated and obtain trace files containing crucial simulation events data, such as packet transmissions and receptions, among other metrics. These trace files were then formatted appropriately to ensure compatibility with Xgraph, a visualization tool within NS2.35. Xgraph offers a wide array of options for customizing plot appearance, including line styles, colors, and axis labels. Moreover, it allows for the plotting of multiple datasets on the same graph, facilitating easy comparison between different scenarios. After generating the plots, the vi-

sualized data are meticulously analyzed to derive meaningful insights into the performance of network simulation.

Table 1 presents the simulation configuration and parameters, along with their respective values or methodologies. The 100 × 100 m^2 network area size offers a realistic yet computationally manageable environment for sensor network simulation, accommodating diverse topologies. With 50–250 nodes, scalability across networks of varying sizes can be explored, from smaller setups where efficiency is less critical to larger ones. Placing the base station at (90,90) simplifies evaluation by centralizing data aggregation and transmission efficiency. A 2 m transmission range ensures ample connectivity without excessive interference or energy use. Initial node energy at 3 J provides a realistic starting point. Transmission and receiving energies (0.6 J and 0.2 J) are chosen based on theoretical estimates. A packet length of 4000 bits aligns with typical sensor network data sizes. A population size of 20 and 100 generations balances computational efficiency with thorough solution exploration. The elitist selection method maintains the best-performing solutions, aiding convergence towards optimal outcomes.

Table 1. Parameter settings for simulation.

Parameter/Scheme	**Value/Method**
Network area size	100 × 100 m^2
Number of nodes	50–250
Location of base station	(90, 90)
Transmission range	2 m
Initial energy of each node	3 J
Transmission energy	0.6 J
Receiving energy	0.2 J
E_{elec}	50 nJ/bit
ε_{fs}	15 pJ/bit/m^2
ε_{mp}	0.0015 pJ/bit/m^4
Packet length (*pl*)	4000 bits
Population size	20
Number of generations	100
Selection method	Elitist selection method

The efficacy of the proposed CGA-GWO algorithm was evaluated through a comparative experiment with existing schemes, including the low-energy adaptive clustering hierarchy genetic algorithm (LEACH-GA) [21], fuzzy GWO [19], and genetic-algorithm-based unequal clustering and routing (GA-UCR) [20]. Performance metrics including the active node count, remaining network energy, packet transmission ratio, clustering overhead, and routing overhead were measured to evaluate each method's efficacy. The performance measures are explained as follows:

Number of alive nodes: This metric measures the number of nodes in the network whose reserved energy has not yet depleted.
Average remaining energy: This metric indicates the average amount of energy still available among sensor nodes active in the network.
Percentage of packets received: This metric represents the ratio of packets successfully received by a node to the total number of packets it attempted to send within the network.
Clustering overhead: This metric refers to the additional control and communication costs associated with the establishment and maintenance of clusters within the network.

Routing overhead: This metric refers to the additional communication and computational costs incurred by routing protocols in managing and optimizing data transmission within the network.

The classic LEACH protocol presents several shortcomings that render it less suitable for comparison with the proposed CGA-GWO algorithm. These drawbacks include unequal cluster sizes, inefficiency with network size, limited support for multi-hop communication, and static network assumptions. Due to these inherent limitations, this research work does not include LEACH classic in the comparative analysis. Furthermore, according to the literature, LEACH-GA exhibits significant enhancements over classic LEACH in critical performance areas such as network lifetime, energy efficiency, throughput, and routing overhead. Moreover, existing research [20,27,28] indicates that alternative algorithms such as LEACH-GA, Fuzzy-GWO, and GA-UCR offer notable improvements over LEACH classic in key performance metrics such as network lifetime, energy efficiency, throughput, and routing overhead. These algorithms have demonstrated enhanced effectiveness in managing energy consumption, extending network lifetime, optimizing data throughput, and reducing routing overhead when compared to the traditional LEACH approach.

Figure 5 depicts a plot of the number of live nodes in a 200-node network with respect to the number of rounds. According to the simulation results, the CGA-GWO algorithm exhibited more live nodes, surpassing GA-UCR, fuzzy GWO, and LEACH-GA by 15%, 30%, and 50%, respectively. In LEACH-GA, nodes are designated as cluster heads based on an optimized probability, leading to increased energy consumption. Moreover, the dead node count increased when the base station moved beyond the sensor field. In contrast, fuzzy GWO selects cluster heads according to residual energy, specifically targeting nodes with values that surpass the average remaining energy of all cluster heads. This involves the calculation of a fitness function that considers increased energy consumption, resulting in a higher number of dead nodes within the network. In GA-UCR, a genetic is applied for both cluster head selection and routing, combining the two schemes into a single chromosome. Consequently, energy consumption is a parameter in the fitness function calculation. As compared to other related algorithms, CGA-GWO shows a higher number of alive nodes as the number of iterations increases, as illustrated in Figure 5. This improvement is attributed to the integration of chaotic features in the genetic algorithm and the strategic selection of nodes as cluster heads. CGA-GWO enhances search efficiency and increases the number of alive nodes by integrating chaotic elements into its genetic algorithm framework.

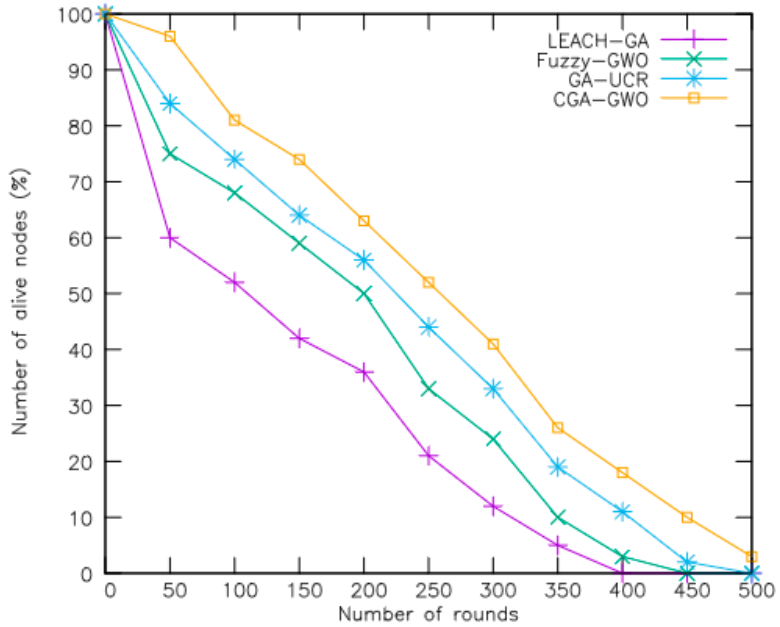


Figure 5. Number of live nodes in each network with respect to number of rounds.

Figure 6 presents a plot of average remaining energy in a 200-node network over an increasing number of rounds. Here, CGA-GWO showed approximately 17%, 22%, and 30% more residual energy than GA-UCR, fuzzy-GWO, and LEACH-GA, respectively, by selecting nodes with the highest fitness values as cluster heads. The network can sustain operation for up to 700 rounds when the average remaining energy of the proposed algorithm CGA-GWO reaches 1 J by the 500th round. Specifically, the proposed algorithm determines suitable nodes for every generation of the population for energy-efficient routing. In addition, CGA-GWO selects chromosomes that consume less energy—i.e., have the maximum residual energy. These chromosomes are associated with the highest fitness values, establishing a direct proportionality between the fitness function and residual energy. Thus, the selection of cluster heads is facilitated by a chaotic genetic algorithm that relies on the residual energy of nodes in the network. This approach enhances the efficiency of cluster head selection and contributes to the overall energy optimization in the network.

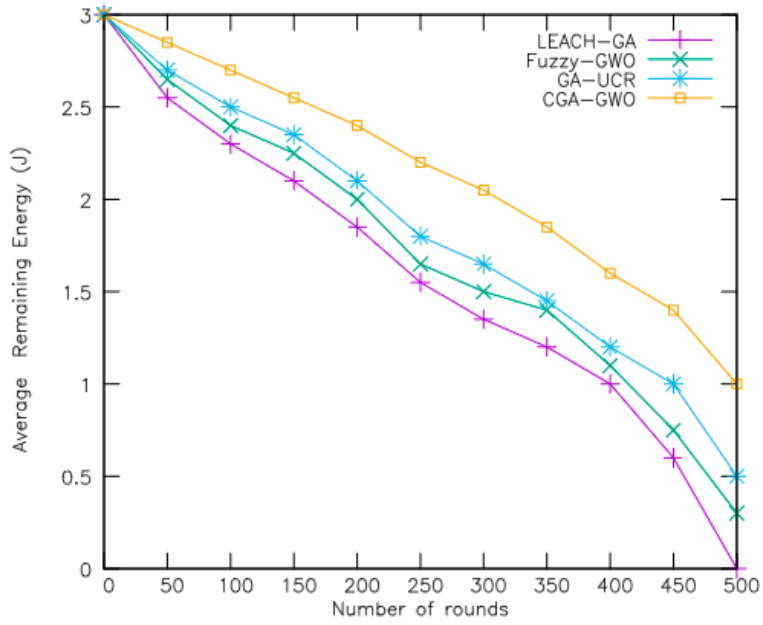


Figure 6. Average remaining energy with respect to number of rounds.

Figure 7 shows the percentage of packets received by considering the total number of nodes in the network. For each algorithm, the percentage of packets received increased along with the number of nodes. Nonetheless, CGA-GWO outperformed LEACH-GA, fuzzy-GWO, and GA-UCR by 38%, 27%, and 14%, respectively, in terms of the percentage of packets received. The proposed CGA-GWO algorithm achieves a packet reception rate of 63%, which surpasses that of other related schemes. These results can be attributed to the fact that the GA-UCR algorithm neglects the existence of energy holes or hotspots formed during the routing process. In contrast, CGA-GWO maintains an increased number of delivered packets owing to its fitness function, which effectively minimizes packet loss during data transmission. To further reduce packet loss, CGA-GWO constructs an efficient path that does not contain dead nodes. Conversely, LEACH-GA and fuzzy GWO have high packet drop ratios stemming from inappropriate cluster head selection.

In Figure 8, the clustering overhead of the CGA-GWO is compared with that of the three baselines by varying the number of nodes. CGA-GWO exhibited overhead reductions of 41%, 34%, and 11% compared with LEACH-GA, Fuzzy-GWO, and GA-UCR, respectively, as the proposed clustering fitness function selects high-residual energy nodes as cluster heads. Furthermore, the incorporation of chaotic features reduces the involvement of heavily loaded sensor nodes in the clustering process. In addition, the CGA-GWO algorithm employs an elitist selection method to select high-quality chromosomes, ensuring the transmission of elite individuals to subsequent generations. This ensures

that higher-quality genes participate in each successive generation's clustering process. In contrast, fuzzy GWO incurs a higher clustering overhead owing to repeated transmissions, whereas GA-UCR uses an intercluster separation fitness function to select cluster heads, resulting in increased overhead owing to the sizes of clusters. Ultimately, CGA-GWO employs a chaotic genetic algorithm to identify suitable cluster heads, minimizing the overhead associated with the clustering process.

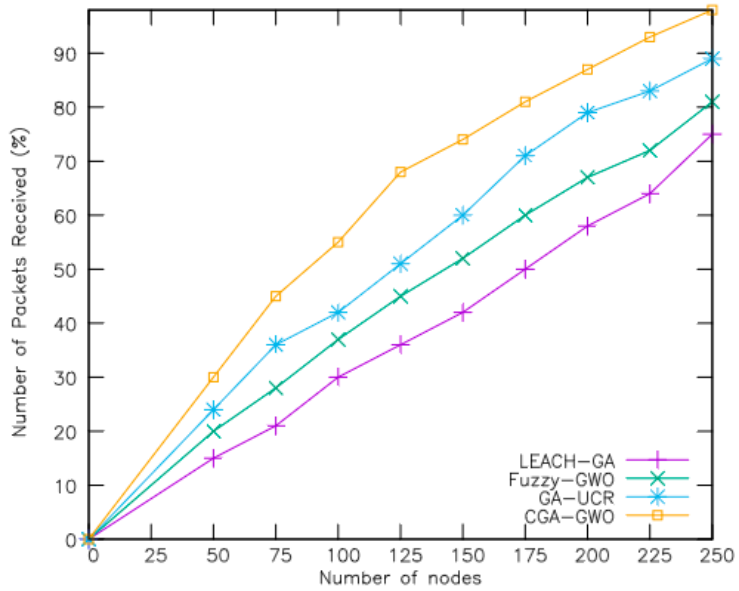


Figure 7. Number of packets received by increasing nodes.

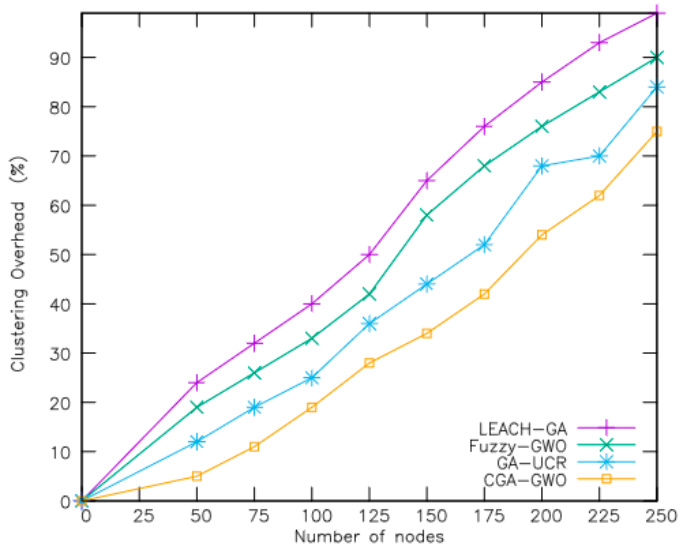


Figure 8. Clustering overhead.

In Figure 9, the routing overheads of the different schemes are plotted for varying numbers of nodes, with CGA-GWO exhibiting a 47% reduction in overhead compared with the baselines. This is attributed to the consideration of factors such as the number of hops per path and residual energy per hop when selecting a routing path. By considering these factors, the fitness function facilitates the identification of an efficient route to the base station. The optimization aims to reduce the hop count along the path, resulting in a more efficient routing path compared with that determined by the baselines. In addition, the wolves' arrangement and directional movements play crucial roles in determining the optimal routing solution. In GA-UCR, the classic genetic algorithm is employed to navigate

data toward the base station given the NP-hard nature of the problem. In fuzzy GWO, nodes along the routing path remove the corresponding packets from their queues, leading to increased packet retransmissions and, consequently, a higher routing overhead.

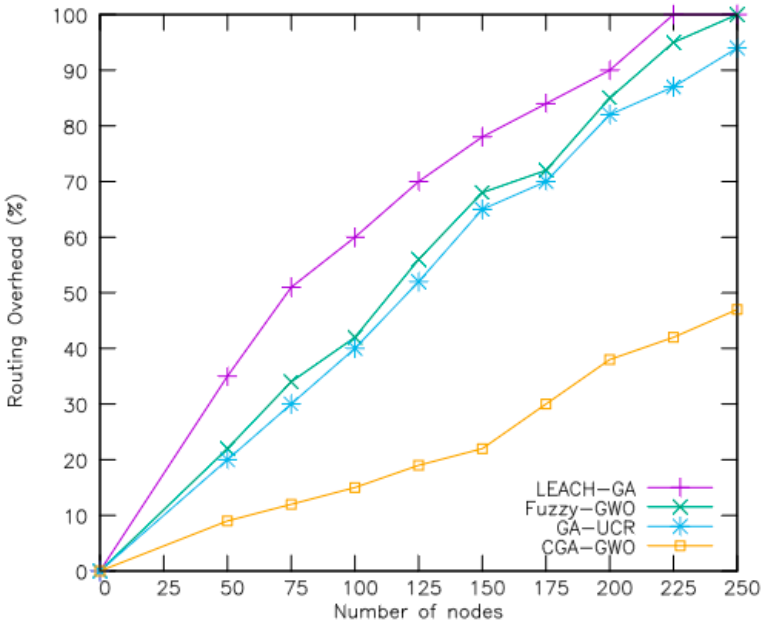


Figure 9. Routing overhead.

The fitness function is specifically designed to incorporate these factors, aiding in the identification of efficient routes to the base station. This optimization strategy aims to minimize the hop count along the path, resulting in a superior routing path compared to other algorithms. Additionally, the arrangement of wolves and their directional movement within CGA-GWO's genetic algorithm framework play a crucial role in determining the optimal solution for routing within the population. In contrast, GA-UCR utilizes a classical genetic algorithm to navigate data toward the base station, acknowledging the NP-Hard nature of the problem. Meanwhile, Fuzzy-GWO experiences increased routing overhead due to nodes along the routing path removing corresponding packets from their queues, leading to heightened packet retransmissions. In summary, CGA-GWO's comprehensive approach to routing optimization, considering factors such as hop count and residual energy, coupled with its efficient genetic algorithm framework, contributes to its significant reduction in routing overhead compared to other schemes.

5. Conclusions

In this study, a chaotic genetic algorithm was combined with the GWO technique to obtain an energy-efficient solution for cluster-based optimal routing. The innovation of the proposed algorithm lies in the utilization of chaotic dynamics to select high-quality chromosomes for the clustering process and determine the routing path, with the GWO mechanism implemented to minimize packet retransmission. In addition, the residual energy levels of the nodes in each network round are considered using the chaotic genetic algorithm during the selection of cluster heads. Furthermore, the optimal routing solution is determined by the wolves' positions and trajectories. Extensive simulation experiments were conducted to assess the functionality of the proposed CGA-GWO algorithm using both MATLAB and NS2 simulators. The simulation outcomes indicate that CGA-GWO enhances network performance across a wide range of metrics.

Author Contributions: Conceptualization, H.K. and M.P.; methodology, H.K. and A.S.; software, H.K., A.S. and M.P.; validation, H.K., M.P. and A.S.; formal analysis, H.K.; investigation, H.K. and M.P.; resources, M.P., A.M. and A.S.; data curation, H.K. and A.S.; writing—original draft preparation, H.K., M.P. and A.S.; writing—review and editing, J.C. and A.M.; visualization, H.K.; supervision, J.C. and A.M.; project administration, J.C. and A.M.; funding acquisition, J.C. All authors have read and agreed to the published version of the manuscript.

Funding: This work was partly supported by Institute of Information & communications Technology Planning & Evaluation(IITP) grant funded by the Korea government(MSIT) (No. RS-2023-00229801, Development of digital communication, indoor/outdoor traffic guide, and non-face-to-face helper service for the elderly/disabled based on XR glasses).

Data Availability Statement: The original contributions presented in the study are included in the article, further inquiries can be directed to the corresponding authors.

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

References

1. Ketshabetswe, L.K.; Zungeru, A.M.; Mangwala, M.; Chuma, J.M.; Sigweni, B. Communication protocols for wireless sensor networks: A survey and comparison. *Heliyon* **2019**, *5*, e01591. [CrossRef] [PubMed]
2. Yuste-Delgado, A.J.; Cuevas-Martinez, J.C.; Trivino-Cabrera, A. EUDFC—Enhanced unequal distributed Type-2 fuzzy clustering algorithm. *IEEE Sens.* **2019**, *19*, 4705–4716. [CrossRef]
3. Gheisari, M.; Abbasi, A.A.; Sayari, Z.; Rizvi, Q.; Asheralieva, A.; Banu, S.; Awaysheh, F.M.; Shah, S.B.H.; Raza, K.A. A survey on clustering algorithms in wireless sensor networks: Challenges, research, and trends. In Proceedings of the 2020 International Computer Symposium (ICS), Tainan, Taiwan, 17–19 December 2020; pp. 294–299. [CrossRef]
4. Pal Singh, S.; Bhanot, K.; Sharma, S. Critical analysis of clustering algorithms for wireless sensor networks. In *Proceedings of Fifth International Conference on Soft Computing for Problem Solving, Proceedings of the SocProS 2015, Roorkee, India, 18–20 December 2015*; Advances in Intelligent Systems and Computing; Springer: Singapore, 2016; Volume 436, pp. 783–793.
5. Kaur, S.; Mahajan, R. Hybrid meta-heuristic optimization based energy efficient protocol for wireless sensor networks. *Egypt. Inf. J.* **2018**, *19*, 145–150. [CrossRef]
6. Zeng, B.; Dong, Y.; Li, X.; Gao, L. IHSCR: Energy-efficient clustering and routing for wireless sensor networks based on harmony search algorithm. *Int. J. Distrib. Sens. Netw.* **2017**, *13*, 1550147717741103. [CrossRef]
7. Yang, X.S. *Nature-Inspired Metaheuristic Algorithms*, 2nd ed.; Luniver Press: Bristol, UK, 2010.
8. Cuevas, E.; Rodríguez, A.; Alejo-Reyes, A.; Del-Valle-Soto, C. Metaheuristic algorithms for wireless sensor networks. In *Recent Metaheuristic Computation Schemes in Engineering*; Studies in Computational Intelligence; Springer: Cham, Switzerland, 2021; Volume 948, pp. 193–235. [CrossRef]
9. Yang, X.S.; Karamanoglu, M. 1-Swarm intelligence and bio-inspired computation: An overview. In *Swarm Intelligence and Bio-Inspired Computation*; Elsevier: Amsterdam, The Netherlands, 2013; pp. 3–23.
10. Benmansour, F.L.; Labraoui, N. A Comprehensive review on swarm intelligence-based routing protocols in wireless multimedia sensor networks. *Int. J. Wirel. Inf. Netw.* **2021**, *28*, 175–198. [CrossRef]
11. Ajmi, N.; Helali, A.; Lorenz, P.; Mghaieth, R. 'MWCSGA-Multi Weight Chicken Swarm Based Genetic Algorithm for Energy Efficient Clustered Wireless Sensor Network. *Sensors* **2021**, *21*, 791. [CrossRef] [PubMed]
12. Jia, W.; Liu, M.; Zhou, J. Adaptive chaotic ant colony optimization for energy optimization in smart sensor networks. *J. Sens.* **2021**, *2021*, 5051863. [CrossRef]
13. Wang, Z.; Ding, H.; Li, B.; Bao, L.; Yang, Z.; Liu, Q. Energy efficient cluster based routing protocol for WSN using firefly algorithm and ant colony optimization. *Wirel. Pers. Commun.* **2022**, *125*, 2167–2200. [CrossRef]
14. Majeed, D.M.; Rabee, H.W.; Ma, Z. Improving energy consumption using fuzzy-GA clustering and ACO routing in WSN. In Proceedings of the 2020 3rd International Conference on Artificial Intelligence and Big Data (ICAIBD), Chengdu, China, 28–31 May 2020; pp. 293–298. [CrossRef]
15. Ram, S.; Nageswara Rao, K.; Basha, S.J.; Reddy, S. Cluster head and optimal path selection using K-GA and T-FA algorithms for wireless sensor networks. In Proceedings of the 2020 4th International Conference on Electronics, Communication and Aerospace Technology (ICECA), Coimbatore, India, 5–7 November 2020; pp. 798–805.
16. Agrawal, D.; Wasim Qureshi, M.H.W.; Pincha, P.; Srivastava, P.; Agarwal, S.; Tiwari, V.; Pandey, S. GWO-C: Grey wolf optimizer-based clustering scheme for WSNs. *Int. J. Commun.* **2020**, *33*, e4344. [CrossRef]
17. Joseph, L.M.I.L.; Deepika, G.J.; Dinesh, P.S.; Vijayashaarathi, S.; Samanvita, N. Modified chaotic grey wolf optimization algorithm for energy aware in WSN. In Proceedings of the 2023 International Conference on Evolutionary Algorithms and Soft Computing Techniques (EASCT), Bengaluru, India, 20–21 October 2023; pp. 1–5. [CrossRef]
18. Patra, B.K.; Mishra, S.; Patra, S.K. Genetic algorithm-based energy-efficient clustering with adaptive grey wolf optimization-based multipath routing in wireless sensor network to increase network life time. In *Intelligent Systems, Proceedings of the ICMIB 2021, Sarang, India, 18–20 December 2021*; Lecture Notes in Networks and Systems; Springer: Singapore, 2022; Volume 431, pp. 499–512.

19. Jainendra Singh, D.; Zaheeruddin, S.B.; Kumararaja, V.; Jegathesh Amalraj, S. Energy-efficient clustering and routing algorithm using hybrid fuzzy with grey wolf optimization in wireless sensor networks. *Sec. Commun. Netw.* **2022**, *2022*, 9846601.
20. Gunjan, S.; Sharma, A.K.; Verma, K. GA-UCR: Genetic algorithm based unequal clustering and routing protocol for wireless sensor networks. *Wirel. Pers. Commun.* **2023**, *128*, 537–558. [CrossRef]
21. Liu, J.-L.; Ravishankar, C.V. LEACH-GA: Genetic algorithm-based energy-efficient adaptive clustering protocol for wireless sensor networks. *Int. J. Mach. Learn. Comput.* **2011**, *1*, 79–85.
22. Javidi, M.; Hosseinpourfard, R. Chaos genetic algorithm instead genetic algorithm. *Int. Arab. J. Inf. Technol.* **2015**, *12*, 163–168.
23. Yadav, A.; Pal, V.; Jha, K. A Chaotic Genetic Algorithm for Wireless Sensor Networks. In *Proceedings of First International Conference on Smart System, Innovations and Computing, Proceedings of the SSIC 2017, Jaipur, India, 14–16 April 2017*; Smart Innovation, Systems and Technologies; Springer: Singapore, 2018; Volume 79, pp. 799–808.
24. Wang, C.; Liu, X.; Hu, H.; Han, Y.; Yao, M. Energy-efficient and load-balanced clustering routing protocol for wireless sensor networks using a chaotic genetic algorithm. *IEEE Access* **2020**, *8*, 158082–158096. [CrossRef]
25. Banerjee, P.S.; Mandal, S.N.; De, D.; Maiti, B. CGARP: Chaos genetic algorithm-based relay node placement for multifaceted heterogeneous wireless sensor networks. *Innov. Syst. Softw. Eng.* **2022**, *2022*, 1–16. [CrossRef]
26. Mirjalili, S.; Mirjalili, S.M.; Lewis, A. Grey wolf optimizer. *Adv. Eng. Softw.* **2014**, *69*, 46–61. [CrossRef]
27. Sivakumar, P.; Radhika, M. Performance Analysis of LEACH-GA over LEACH and LEACH-C in WSN. *Procedia Comput. Sci.* **2018**, *125*, 248–256. [CrossRef]
28. Lata, S.; Mehfuz, S.; Urooj, S.; Alrowais, F.M. Fuzzy Clustering Algorithm for Enhancing Reliability and Network Lifetime of Wireless Sensor Networks. *IEEE Access* **2020**, *8*, 66013–66024. [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.