Design and Application of Improved Genetic Algorithm for Optimizing the Location of Computer Network Nodes

Chunlei Zhong[1*], Gang Yang[2]
[1]Huai'an Bioengineering Branch Institute, Jiangsu Union Technical Institute, Huai'an, 223200, China
[2]College of Teacher Education, Wenzhou University, Wenzhou, 325035, China
E-mail: hm_spring@163.com
*Corresponding author

Keywords: genetic algorithm, computer network, network nodes, improved genetic algorithm, average error

Received: September 24, 2024

The rapid development of computer technology has made network stability and node positioning accuracy important challenges in optimizing computer network design. This study proposes an optimization method based on the Improved Genetic Algorithm (IGA) to improve the positioning accuracy and stability of network nodes. Firstly, by combining the characteristics of the centroid algorithm and the Approximate Point in Triangulation Test (APIT) algorithm, preliminary optimization of node positions is carried out. Subsequently, an IGA is utilized for further optimization, dynamically adjusting the crossover probability and mutation probability to balance global and local search capabilities and avoid the algorithm falling into local optima. The experimental results showed that IGA achieved significant performance improvement in node localization. Compared with the centroid algorithm, the maximum error of IGA has been reduced by 19% and the overall average error has been reduced by 8.8%. Compared with APIT, IGA has reduced the maximum error by 7% and the overall average error by 3.8%. Regarding fitness values, IGA exhibited faster convergence speed, achieving optimal results with only 75 iterations, surpassing traditional genetic algorithms and APIT algorithms. The node coverage rate reached 98.6%, far higher than the 85.3% of the centroid algorithm and 90.5% of the APIT algorithm. These results demonstrate that IGA has higher accuracy, stability, and computational efficiency in complex network environments, providing an efficient and reliable solution for optimizing the design of computer network nodes.

Povzetek: Predlagan je izboljšan genetski algoritem (IGA) za optimizacijo lokacij vozlišč v računalniških omrežjih, ki z dinamičnim prilagajanjem verjetnosti križanja in mutacije poveča točnost, stabilnost in učinkovitost algoritma.

1 Introduction

With the continuous progress of modern technology, computer networks play an increasingly important role in modern society. They connect various devices and systems, making the transmission and sharing of information more efficient and convenient. To meet the needs of users for high-quality network services, improving network performance and optimizing network design have become increasingly important. Traditional optimization algorithms frequently encounter issues of low efficiency and a propensity to fall into local optima when addressing large-scale network design problems. Therefore, it is necessary to introduce new optimization algorithms to solve these problems [1-2]. In recent years, researchers have made significant advancements in applying enhanced genetic algorithms to optimize computer networks. These enhancements include the introduction of new operators, optimization of algorithm parameters, and adjustments to the algorithms themselves. As a result, genetic algorithms are now more efficient and accurate when utilized for network design optimization. At the same time, researchers combine genetic algorithms with other optimization algorithms to create multiple hybrid optimization algorithms, which enhances network design performance and effectiveness [3-4]. The objective of the research is to achieve an optimized design of computer networks and to improve network performance indicators, including latency, throughput, resource utilization, and cost, through an Improved Genetic Algorithm (IGA). The research aims to solve the problems of slow convergence speed, susceptibility to local optima, and difficulty in dynamic adjustment of traditional network optimization methods in complex network environments. This study designs a computer network optimization technique based on a genetic algorithm as the core and introduces multiple techniques to improve performance. A fitness function based on network performance indicators is constructed to quantify the network optimization objectives. This technology adjusts the crossover and mutation probabilities adaptively by comparing individual fitness

and population average fitness, balancing global and local search capabilities.

2 Related work

The 5G era is coming and the network technology is developing rapidly. Numerous data have brought enormous challenges to the stability and reliability of computer networks. The reliability of computer networks is a major indicator of computer comprehensive performance. Computer networks are large and complex, and they are also easily affected by many adverse factors. This leads to instability in the system, which exposes the entire computer network to significant risks. To ensure the stability and ongoing optimization of computer networks, computer network optimization design has become a prevalent point of discussion in computer research. Through their study of cloud computing, Fan et al. [5] presented a novel mathematical model for virtual network embedding in optical data center networks. This model reduced Network Topology (NT) complexity during optical fiber transmission. They used a comprehensive system of node awareness and path evaluation to derive algorithms with priority locations. The algorithm obtained by this model could reduce the latency of virtual network requests by 20% and improve the request rate by 13%. Rajendran and Venkataraman [6] proposed a new neural network algorithm to analyze network traffic built on the application and analysis of big data in network security. They used this method to conduct statistics on the worst data and abnormal activity sent by the network and conducted experiments with the data. Compared with traditional algorithms for neural networks, the optimized algorithm showed a notable enhancement in distinguishing between false alarms and actual detection, which significantly improved the security and stability of the network. Xiaokaiti et al. [7] raised an efficient data transmission strategy for the detection algorithm of computer network communities. They first combined NT attributes with social attributes when dividing communities and then selected the optimal relay node for network transmission based on the number of channels. This algorithm had high merit in data delivery efficiency and routing overhead in computer networks. Alsaqour et al. [8] put forward a location-assisted routing algorithm grounded on genetic algorithms to optimize the efficiency of MANET routing protocols. Firstly, through algorithm optimization, node information was added to the route and these nodes were grouped. These nodes were then sent to their destinations to adaptively update the node location. The results showed that the optimized algorithm could achieve a delivery rate of over 99% for small network overhead packets. Bu [9] developed a load-balancing scheduling algorithm for Internet of Things (IoT) clusters using a combination of Particle Swarm Optimization and Genetic Algorithm (PSOGA). The purpose of this algorithm was to address the persistent challenge faced by IoT networks due to high-volume business data traffic causing downtime. They first used the CPU, RAM, and network bandwidth to measure the server node information, then adjusted the appropriate function value, and used the IGA to obtain the optimal solution. The results showed that the optimized algorithm could reduce latency and error rates by 5%, while also reducing server overload and downtime. Network coding could integrate coding capabilities with network multi-path propagation, bolster the capacity of computer networks, and facilitate more intricate security solutions. To address the susceptibility of network coding to attacks, Wu et al. [10] developed a comprehensive unicast secure transmission scheme based on Random Linear Network (RLN). The matrix was randomly generated from the received nodes and the resulting vector was sent back to the source node via the link to form a new matrix. This approach effectively thwarted network eavesdropping attacks. The comparative analysis between the research and the advanced methods is shown in Table 1.

Table 1: Comparative analysis of research and the advanced methods

Reference	Technical Method	Advantages	Disadvantages	Comparison with IGA
Fan et al. [5]	Virtual network embedding with node awareness and path evaluation.	Reduces latency by 20% and improves request rate by 13%.	Limited applicability; does not optimize node positioning.	IGA reduces error by 8.8%, with broader applicability.
Rajendran et al. [6]	Enhanced neural network algorithm for malicious traffic detection.	Improves security and reduces false alarms.	High computational cost; lacks node optimization.	IGA achieves 2.41% error, with higher efficiency.
Xiaokaiti et al. [7]	Community detection algorithm to optimize data transmission.	Improves transmission efficiency and reduces routing overhead.	Dependent on BT; limited precision.	IGA reduces error by 3.8%, offering better stability.
Alsaqour et al. [8]	Genetic algorithm for optimizing mobile ad	Achieves 99% small packet delivery rate with	Suitable for small networks; struggles	IGA reduces error to 2.46%, with

	hoc network routing.	low overhead.	with large-scale networks.	wider applicability.
Bu [9]	PSO and GA combined for load balancing.	Reduces load and downtime by 5%.	Focuses on load balancing; lacks positioning accuracy.	IGA improves accuracy by 8.8%, offering a comprehensive solution.
Wu et al. [10]	Secure transmission using random linear network coding.	Enhances security and prevents eavesdropping.	Does not optimize node positioning or transmission efficiency.	IGA achieves 5.2% error, with better precision and stability.

Previous research has found that related work mainly focuses on specific aspects of computer network optimization, including security enhancement, data transmission efficiency, and load balancing. However, they have shortcomings in addressing the accuracy of network node localization and overall stability under different network conditions. Existing algorithms such as the centroid algorithm and Approximate Point in Triangulation Test (APIT) have significant drawbacks, including limited accuracy and sensitivity to node density. Traditional algorithms, such as genetic algorithms and MANET routing protocols perform well in specific network types, but perform poorly in large-scale or dynamic environments. Using genetic algorithms to assist routing protocols can improve network overhead and delivery rates. This fully optimizes the genetic algorithm and enhances network delivery. To optimize computer network nodes for better environmental conditions, this study uses centroid and APIT algorithms, which provide better conditions for computer network optimization. Then, based on node optimization, an IGA is used to construct a network design optimization model. Through optimizing the traditional ant colony algorithm, the efficiency of network nodes in computer network optimization design is enhanced. This paper aims to increase the stability and reliability of computer network optimization design.

3 Construction of computer network optimization design model based on genetic algorithm

3.1 Optimization of node location based on centroid algorithm and APIT algorithm

From the perspective of topology, a computer network is composed of several network nodes and communication links connecting these network nodes. This indicates that the positioning of network nodes is indispensable in computer network data transmission. The centroid algorithm is the most typical node localization algorithm among commonly used localization algorithms. The algorithm has four advantages: low storage energy consumption, simple algorithm principle, low computing energy consumption, and low communication energy consumption.

Before using this algorithm for localization, it is first necessary to determine whether the location node that the sensor needs to determine is located within the region. At the same time, nodes requiring location determination will continually emit various

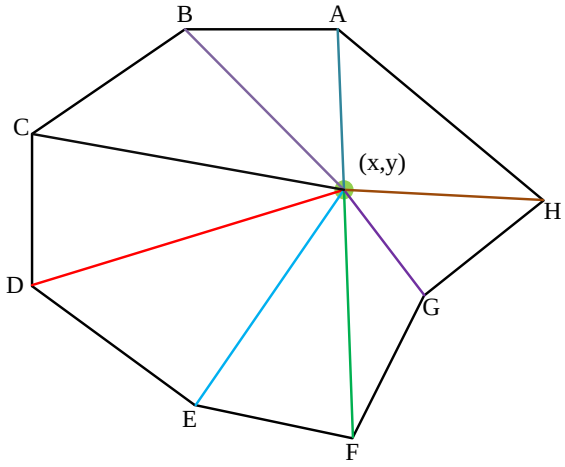


Figure 1: Schematic diagram of centroid algorithm positioning

communication signals to the surrounding environment. To determine whether the unknown node is in the monitoring area, it is essential to verify the strength of the signal obtained at the beacon node. The strength can reflect the unknown node location [11]. The principle of the centroid algorithm is obtained built on the algorithm for the centroid. In any irregular polygon, there must be a center of mass inside it. Usually, the coordinates of each vertex are accumulated, and then the average value is calculated to determine its specific coordinates. The specific location can be represented by Formula (1), and its algorithm diagram is shown in Figure 1.

$$(x,y)=(\frac{1}{n}\sum_{i=1}^{n}x_i,\frac{1}{n}\sum_{i=1}^{n}y_i) \tag{1}$$

In Formula (1), n represents a n-sided shape. (x,y) means the coordinate of the vertex. The centroid of this n-sided shape can be obtained by calculating the formula.

If the polygon is situated within the solved region and has matching coordinates, then the centroid coordinates of the octagon can be computed using Formula (2).

$$(x,y)=(\frac{x_1+x_2+\cdots+x_8}{8},\frac{y_1+y_2+\cdots+y_8}{8}) \tag{2}$$

The $(x_1,y_2)\cdots(x_8,y_8)$ in Formula (2) represents the coordinates of eight vertices. To use centroid positioning algorithms for positioning, it is essential to rely on the smoothness of the entire network structure and the specific distribution of positioning nodes within the network. If an error occurs in the coordinates calculated by unknown nodes, it will bias towards areas with densely distributed beacon nodes, potentially resulting in significant errors with the centroid algorithm. Therefore, the algorithm's calculation accuracy is typically not high, and the positioning accuracy may be low. However, the centroid algorithm only needs to broadcast once to locate all unknown nodes. In many applications that do not require high positioning accuracy, the centroid algorithm is still the most suitable method.

APIT is an improved algorithm for the centroid algorithm. It requires a completely random selection of many known coordinate nodes, and the coordinate nodes are grouped every three. In accordance with these nodes, the triangles drawn on the graph will be completely randomly distributed throughout the entire region. There will be some overlap between these triangles to calculate the coordinates of unknown nodes. The specific operation steps are: First, multiple coordinate nodes around unknown nodes are identified, and three known location nodes are randomly

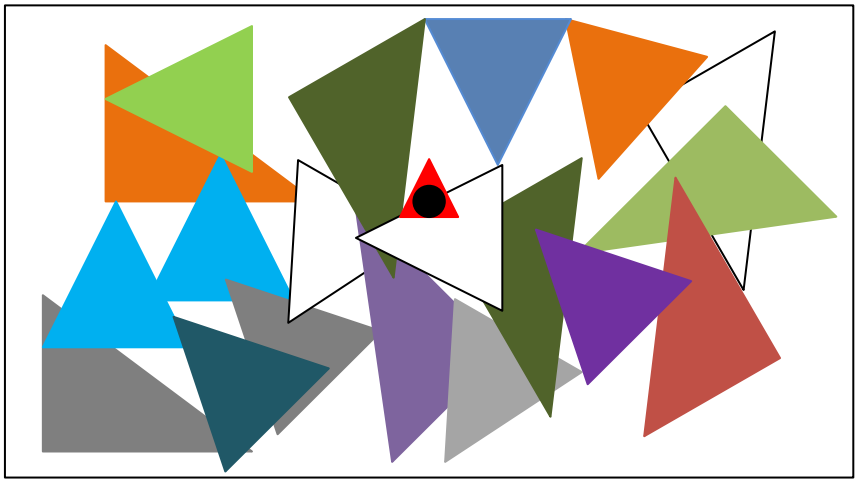


Figure 2: Schematic diagram of APIT algorithm positioning

selected each time. Then, the approximate location of the signals received by these known location nodes is determined. If there are m beacon nodes, the paper will randomly select and match them, and use the combination of three random position points to form C_n^3 triangle. It shows that some triangular regions can contain unknown nodes, while others do not. These specific points, which contain triangular regions of unknown nodes, are connected to each other. Finally, the recorded location algorithm is utilized to calculate the specific location of unknown nodes. The incorporation of a greater number of unknown nodes into the algorithm results in enhanced accuracy in location estimation [12-13]. However, this is accompanied by an increased computational burden. In such a scenario, choosing a subset of vertices to create a polygon based on the real circumstances, as illustrated in Figure 2, can be beneficial. In Figure 2, the node positioning accuracy of the APIT is significantly greater than that of the centroid positioning algorithm.

Due to the relatively large impact of node density on APIT, when the beacon node density is relatively large, APIT can achieve relatively ideal positioning accuracy.

APIT also has good performance in irregular wireless signal propagation models and irrational circular propagation models. However, APIT also has a significant disadvantage. When connecting triangles, it may mistake points located outside the triangle for points inside the triangle. The probability of generating this situation can reach a maximum of 13% through research [14], which will have a significant impact on positioning accuracy. The algorithm must divide a large number of triangular regions to identify the locations of unknown nodes and necessitates multiple beacon nodes. As a result, the algorithm performs numerous calculations, which elevates the likelihood of encountering errors.

3.2 Construction of improved genetic algorithm model

The research and analysis of the centroid and APIT algorithms in node location optimization have revealed shortcomings in both algorithms concerning their calculation and location processes. Additionally, the use of genetic algorithms for node localization requires extra constraints, which may lead to increased computational time and reduced efficiency, resulting in premature convergence [15]. To obtain better positioning optimization results, an IGA model is studied and constructed. The flow chart of the model is shown in Figure 3, and the blue box in the figure shows the improved steps.

Compared with Traditional Genetic Algorithms (TGA), the paper has improved the node localization of genetic algorithms and constructed a matrix. The specific construction of the matrix is shown in Formula (3).

$$S(m)=\begin{bmatrix} s_1 \\ s_2 \\ \vdots \\ s_m \end{bmatrix}=\begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & \vdots & \vdots \\ a_{m1} & a_{m2} & \cdots & a_{mn} \end{bmatrix} \tag{3}$$

The m in Formula (3) represents a total of m chromosomes in the genetic algorithm. n means that each chromosome has n elements. s_1, s_2 $\cdots$ s_m are each chromosome. In TGA, determining the value range of genes is a commonly used method to generate an initial population waiting for calculation. If a certain number of initial populations are randomly generated within this value range, there may be situations where the distribution is too random. This result is basically not helpful for improving the algorithm efficiency [16]. If one wants to obtain a global optimal solution, the distribution of the initial population in the solution space should be as uniform and dispersed as possible. The schematic diagram of random initial population generation within the overlapping range of communication areas of different anchor nodes is shown in Figure 4.

IGA performs improved optimization in TGA such as parameter setting, population initialization, appropriate function values, selection operations, and crossover operations. The specific key parameter settings are to set the number of populations to be 40, the crossover probability p_{c1} to be 0.6, p_{c2} to be 0.4, the mutation probability p_{m1} to be 0.08, p_{m2} to be 0.06, and the maximum iterations to be 100. Population initialization can determine the initial population range according to Formula (4), and generate an initial population randomly within this range.

$$\begin{cases} \max\limits_{i=1,2,\cdots,n}(x_i-d_i)\le x\le \min\limits_{i=1,2,\cdots,n}(x_i+d_i) \\ \max\limits_{i=1,2,\cdots,n}(y_i-d_i)\le y\le \min\limits_{i=1,2,\cdots,n}(y_i+d_i) \end{cases} \tag{4}$$

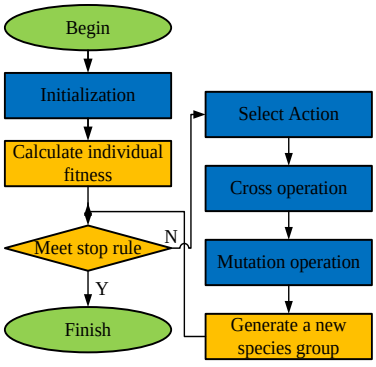


Figure 3: The improvement process of genetic algorithms

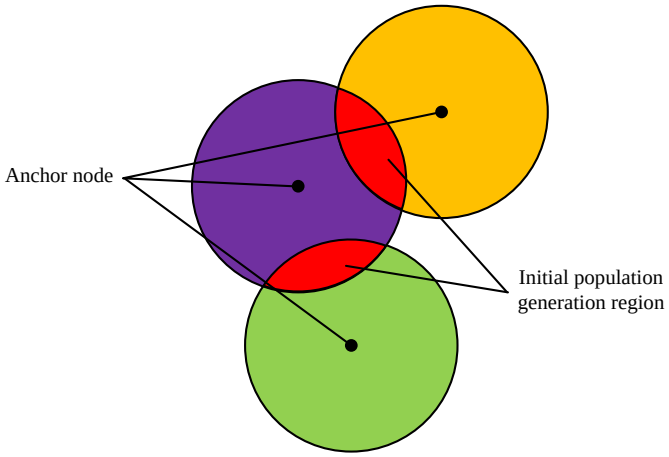


Figure 4: Schematic diagram of initial population generation area

In Formula (4), d_i means the distance between the node of unknown i and the anchor. The fitness function value calculation assumes a total of $(M+N)$ nodes in the wireless sensor network to be located, where the number of known nodes is *M*. The unknown node number is *N*. Through a certain distance measurement method, if each unknown node knows the distance between all known nodes within the communication radius and itself, the calculated node position can be obtained through the least square method [17]. Assuming that the coordinate of a node at a known location is $(x_1,y_1),(x_2,y_2),\ldots,(x_M,y_M)$, the coordinate of an unknown node is (x, y), and the distance from the node at a known location is $d_1, d_2, d_3, \cdots, d_M$. The equation set shown in Formula (5) can be established.

$$\begin{cases} (x-x_1)^2+(y-y_1)^2=d_1^2 \\ (x-x_2)^2+(y-y_2)^2=d_2^2 \\ (x-x_3)^2+(y-y_3)^2=d_3^2 \\ \vdots \\ (x-x_M)^2+(y-y_M)^2=d_M^2 \end{cases} \tag{5}$$

From Formula (5), the fitness function for genetic algorithms can be defined as Formula (6), and the fitness function of the initial population can be calculated by using it.

$$f(x,y)=\frac{1}{M}\sum_{i=1}^{M}\left|\sqrt{(x-x_i)2+(y-y_i)2}-d_i\right| \tag{6}$$

In Formula (6), (x, y) is the unknown location node location. (x_i, y_i) represents a known location node location. d_i refers to the distance from an unknown location to a known location (x_i, y_i). The use of absolute error instead of squared error in the fitness function avoids the calculation of squared error, reduces complex multiplication operations, and lowers computational load. During the iteration process, absolute error is more robust to outliers (i.e. has less impact on data with larger deviations) and also enables the algorithm to approach the global optimal solution faster, accelerating the convergence speed of the algorithm. The selection operation is to perform a unified comparison of each individual based on the fitness value calculated from the fitness function. After the comparison is completed, the two individuals with the highest fitness remain unchanged and proceed to the next round of operation. The individuals with the lowest fitness are directly eliminated, and the remaining objects will normally undergo crossover and mutation operations. Special individuals with high fitness values are set with judgment values to distinguish and limit their reproduction. After completing the full iterative process, the fitness value of each individual should be appropriately amplified [18-19]. The crossover probability is used to control the probability of individuals (chromosomes) performing crossover operations. By calculating an individual's fitness value, it can be determined whether the individual should participate in crossover operations. The goal of crossover operation is to generate offspring with higher fitness by recombining the genetic information of the parent individual, gradually approaching the optimal solution. When performing a crossover operation, if $F_g \geq F_{avg}$, the crossover probability is calculated according to Formula (7).

$$P_c = p_{c_1}\frac{F_g - F_{\arg}}{F_{gb}-F_{avg}} \tag{7}$$

In Formula (7), $pc_1 \in (0,1)$. F_g is the value of the individual's fitness function. F_{gb} represents the optimal individual adaptation function value. F_{avg} is the average of the adaptive function. The calculation process includes normalizing the fitness difference and converting the normalized fitness difference into actual cross probability. If $F_g < F_{avg}$, the crossover probability is calculated according to Formula (8).

$$P_c = P_{c2} \tag{8}$$

In Formula (8), $P_{c2} \in (0,1)$. After pairing the chromosomes in the population, the crossover operation is performed based on the calculated crossover probability. A random number between [0-1] is randomly

generated for each chromosome. The objective of the treatment of poorly adapted individuals is to provide those with lower fitness a certain opportunity to participate in crossover, increase population diversity, and circumvent premature convergence to local optimal solutions. If the corresponding value is less than the crossover probability, the chromosome is ready to perform the next operation. The chromosomes for the next step are sequentially crossed in pairs. For each pair of crossed chromosomes, the location of the crossing is determined by random numbers and the crossover operation is performed. During the mutation operation, if $F_g \geq F_{avg}$, the mutation probability is calculated by Formula (9).

$$P_m = p_{m1} \frac{F_g - F_{avg}}{F_{gb} - F_{avg}} \tag{9}$$

In Formula (9), $p_{m1} \in (0.01, 0.10)$. F_g represents the adaptive degree function value of individual g. F_{gb} is the value of the optimal individual adaptation function. F_{avg} refers to the average value of the adaptive function. If $F_g < F_{avg}$, the probability of variation is calculated by Formula (10).

$$p_m = P_{m2} \tag{10}$$

In Formula (10), $P_{m2} \in (0.01, 0.10)$. The first step is to randomly generate a number between 0 and 1 for each chromosome in the population. If the generated value is less than the mutation probability, it will undergo mutation operation. The position corresponding to the required mutation is determined by generating a random number value, and the next step is to invert the value to complete the relevant mutation operation. Monotonic gene locus detection analyzes the whole population and identifies any monotonic gene loci present. If these are detected, targeted adjustments can be made by generating random numbers. The termination operation should be determined and the loop termination should be evaluated based on the number of iterations. The final step is to output the optimal solution and test the average positioning error of the algorithm as a performance parameter, as shown in Formula (11).

$$error = \frac{100}{N \times R} \sum_{i=1}^{N} \sqrt{(x_{i1} - x_{i2})^2 + (y_{i1} - y_{i2})^2}\% \tag{11}$$

N in Formula (11) is the sum of nodes. (x_{i1}, y_{i1}) and $(x_{i2}, y_{i2}) i = (1, 2, 3, \cdots, M)$ represent the actual and calculated coordinates of the unknown node i. R is the maximum communication distance of the node. In the process of transforming IGA theory into practical applications, the rigor of mathematical analysis is reflected in the precise modeling and dynamic adjustment of fitness functions, crossover probabilities, and mutation probabilities. The dynamic allocation of crossover probability is achieved by comparing individual fitness with group average fitness and optimal fitness. This allows individuals with higher fitness to have a higher probability of crossover, thereby accelerating the spread of excellent genes, while preserving a small number of crossover opportunities for individuals with lower fitness and maintaining population diversity. This normalization mechanism based on fitness differences effectively balances local search and global search, avoids premature convergence of the algorithm, and improves solution accuracy and efficiency. Furthermore, the implementation of random number generation techniques and probability judgment processes enables the transformation of theoretical models into practical operations, thereby ensuring the randomness and controllability of crossover and mutation. This approach facilitates the robustness and convergence of the algorithm in complex optimization problems, thus achieving efficient integration of theory and practice. When using IGA for computer network optimization design, the network optimization problem is first modeled as a fitness function to measure network performance indicators. Then, through iterative evolution through selection, crossover, and mutation operations, the crossover and mutation probabilities are dynamically adjusted to optimize the network structure and parameter configuration, thereby achieving efficient and accurate network optimization design.

4 Performance analysis of computer network optimization design model based on genetic algorithms

4.1 Performance analysis of node location based on centroid location algorithm and APIT algorithm

To verify the actual positioning effects of the centroid algorithm, APIT, and IGA, simulation experiments are conducted on three algorithms in MATLAB. The reason for choosing APIT and centroid algorithm as benchmarks for research is their effectiveness and wide application in network optimization. The APIT algorithm performs well in localization problems and is suitable for evaluating the accuracy and reliability of network nodes, serving as a benchmark for network performance optimization in research. The centroid algorithm is known for its simplicity, ease of use, and fast convergence, making it suitable for solving optimization problems in basic network structures. The selection of these two algorithms perfectly covers different types of network optimization requirements. Through comparison, the advantages of IGA in solving complex optimization problems can be clearly demonstrated. Using MATLAB version R2021a, the hardware specifications are as follows: Intel Core i7-9700K processor, 32 GB DDR4 RAM, 512 GB solid-state drive, and Windows 10 Professional 64 bit operating system. The algorithm sets the population size

to 100, the number of iterations to 500, the crossover probability to 0.8, and the mutation probability to 0.05. The elite strategy is to retain the top 10% of excellent individuals. To ensure the statistical validity of the test scenario, multiple sets of experiments are designed and optimized for network topologies of different sizes and complexities. Each experiment should be repeated at least 30 times to obtain stable average performance indicators and standard deviations, ensuring the reliability of the results. In the collected performance indicators, statistical analysis is used to evaluate the significant differences in algorithms under different configurations, thereby determining the efficacy of the optimization effects. When comparing, a null hypothesis and an alternative hypothesis are set. When calculating the P-value, it represents the probability of obtaining the current or more extreme result under the null hypothesis. The t-test is used to compare the results of IGA and benchmark algorithms. If the P-value is less than 0.05, the difference is considered statistically significant. The experiment generates 20 anchor nodes and 80 unknown nodes in the 100×100 area. After generating this region, the node is predicted by running the corresponding algorithm. Figure 5 shows the original node distribution diagram.

The green circle in Figure 5 represents an anchor node, while the blue pentagon represents an unknown node. Figure 6 shows the positioning results of three algorithms. The positioning results of centroid algorithm, APIT, and IGA are shown in Figure 6(a), Figure 6(b) and Figure 6(c), respectively.

In Figure 6, the predicted value of IGA has a higher coincidence rate with unknown nodes, reaching 94.36% ($P<0.05$). The predicted value of the centroid algorithm has the lowest coincidence rate with unknown nodes, which is 86.25%. The coincidence rate between the predicted value of APIT and the unknown node is 89.67%. The coincidence rate of the IGA is 8.16% higher than that of the centroid algorithm and 4.69% higher than that of the APIT algorithm ($P<0.05$). This means IGA has a high positioning computing ability. The positioning errors of the centroid algorithm, APIT, and IGA are listed in Figure 7.

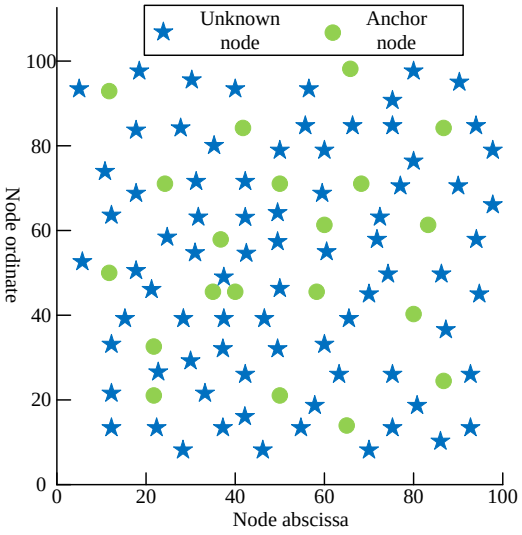
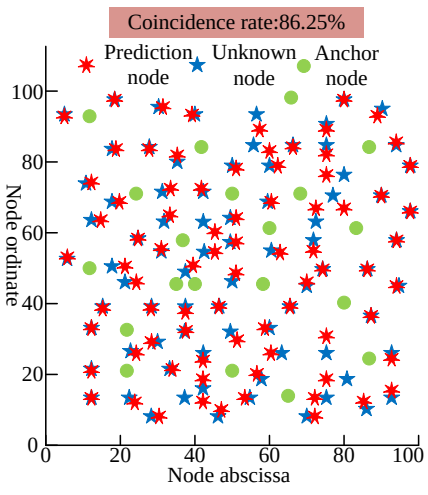
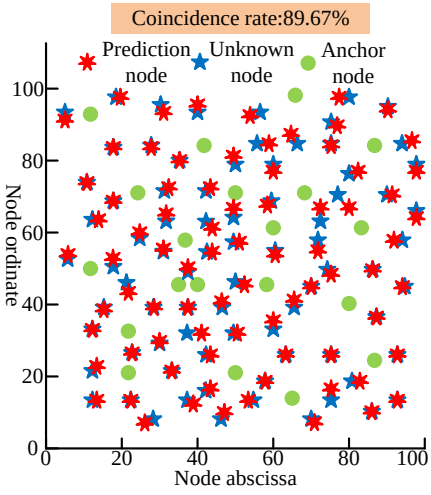


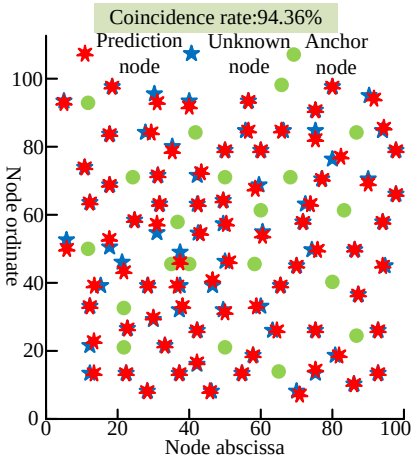
Figure 5: Original node distribution diagram



(a)Centroid algorithm positioning results



(b)ATIP positioning results



(c)Improved genetic algorithm location results

Figure 6: Location results of three algorithms

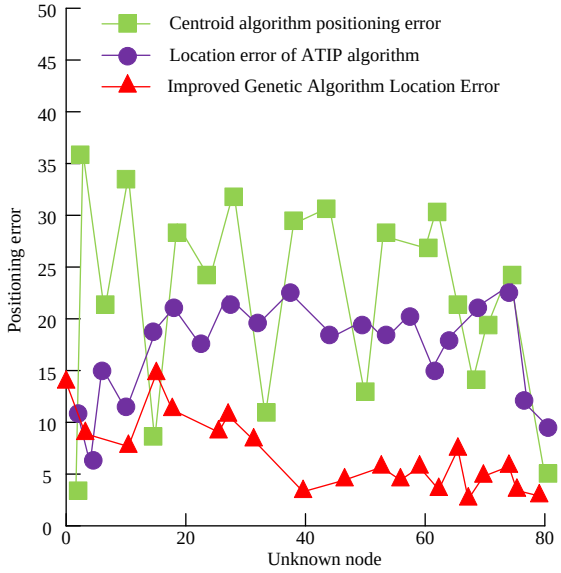


Figure 7: Positioning error of three algorithms

The centroid algorithm in Figure 7 achieves a maximum error rate of 32% during positioning, and the overall node positioning average error rate is 14% (P<0.05). The maximum error rate of APIT in positioning is 20%, but the average value of the overall error has decreased to 9% (P<0.05). This indicates that compared to the centroid positioning algorithm, the predicted coordinates error calculated by the APIT positioning algorithm is significantly reduced, with better positioning results. The maximum error rate of IGA during positioning is 13%, and the average value of its overall error is 5.2% (P<0.05). The maximum error of IGA is 19% lower than the centroid algorithm, and the overall average error is 8.8% lower (P<0.05). Compared to APIT, the maximum error of IGA is 7% lower, and the average overall error is 3.8% lower (P<0.05). APIT improves positioning accuracy through random triangle coverage, but relies on high-density anchor nodes and is prone to misidentifying external points of the triangle as internal points. The centroid algorithm is computationally simple and suitable for scenarios that do not require high accuracy. However, its accuracy is low and it is easily affected by uneven distribution of node density, resulting in positioning bias towards areas with dense anchor nodes and significant errors. IGA introduces a method of dynamically adjusting crossover probability and mutation probability during the evolution process. It dynamically adjusts based on individual fitness and population average fitness, avoiding premature convergence of the algorithm and ensuring the search for the global optimal solution. IGA performs local fine optimization, improving the accuracy and stability of the algorithm. The comparison of the three shows that IGA has excellent node positioning capabilities in wireless sensor networks.

4.2 IGA-based application analysis of computer network optimization design

There are differences in the performance of IGA and other node localization algorithms for wireless sensor networks under different iterations. Hence, under the same parameter conditions, this study gradually changes the iterations and simulates them using TGA, centroid positioning algorithms, APIT, and IGA, respectively. Thus, the iteration number

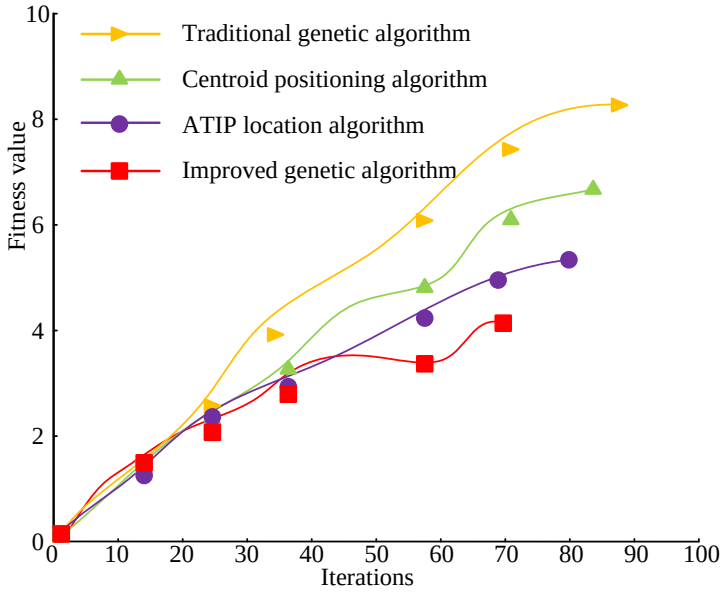


Figure 8: Changes in fitness values of the four methods

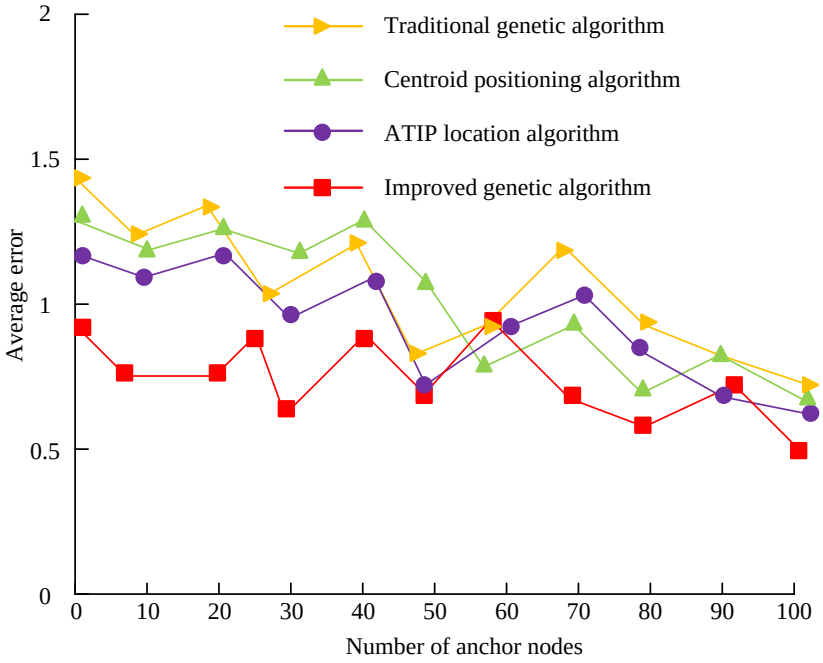


Figure 9: Relationship between the number of anchor nodes and average error

and its corresponding fitness value are obtained. As the number of iterations gradually increases, Figure 8 illustrates the corresponding changes in fitness between the IGA and other algorithms for positioning wireless sensor network nodes.

Figure 8 shows that the fitness values of the four localization algorithms are all less than 10. The fitness values of IGA, TGA, centroid algorithm, and APIT are 4.26, 8.15, 6.42, and 5.31, respectively. The iteration numbers are 69, 86, 83, and 79, respectively. The IGA's fitness value is the lowest, 3.89 lower than that of TGA,

2.03 lower than centroid algorithm, and 1.05 lower than APIT. This shows that IGA has better adaptability in node localization, and verifies the superiority of this algorithm. The anchor node numbers and the average error value's relationship of the four algorithms is shown in Figure 9.

In Figure 9, as the anchor node amount increases, the average error of the four positioning algorithms is gradually decreasing. The average errors of TGA, centroid algorithm, APIT, and IGA are 1.12, 1.03, 0.95, and 0.68, respectively ($P<0.05$). The average error of IGA is significantly lower than that of TGA. Meanwhile, as the number of anchor nodes increases gradually in proportion to all nodes, the average error of various algorithms undergoes slight changes over time, as shown by the curve. As the anchor node amount keeps the same, the average error of IGA is the lowest among all four algorithms. This fully demonstrates the advantages of IGA. Figure 10 displays the chart that pertains to the communication radius of a particular node and the average error value of the four algorithms.

In Figure 10, the communication radius of nodes is increasing, while the average error of the four node positioning algorithms is gradually decreasing. The average errors of TGA, centroid algorithm, APIT, and IGA are 5.75%, 4.52%, 3.87%, and 2.46%, respectively ($P<0.05$). When the communication radius is the same, the IGA's average error is always at the lowest value. This verifies the superiority of IGA when the communication radius of nodes changes. Figure 11 is the relevant of the specific network connectivity and the average error value of the four algorithms.

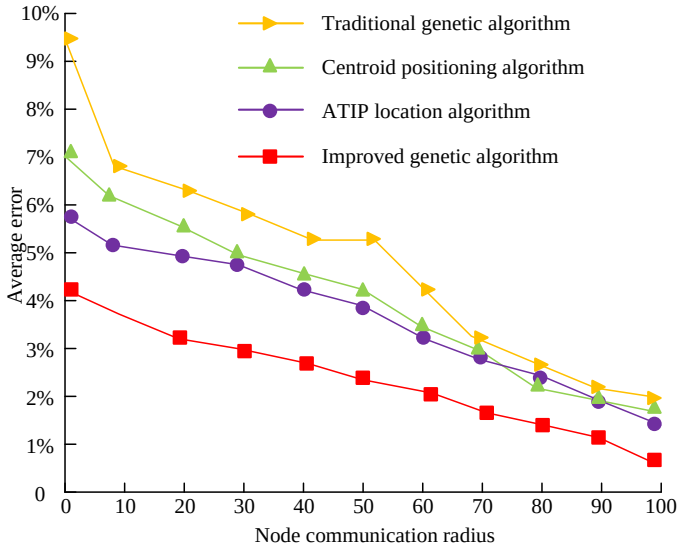


Figure 10: Relationship between node communication radius and average error

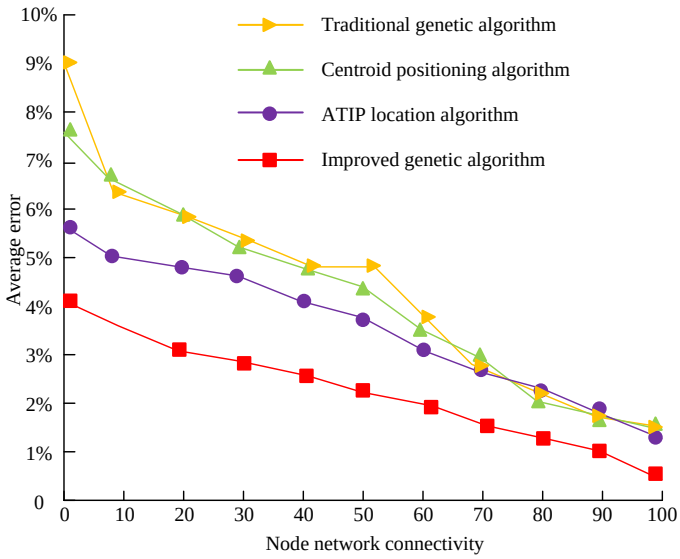
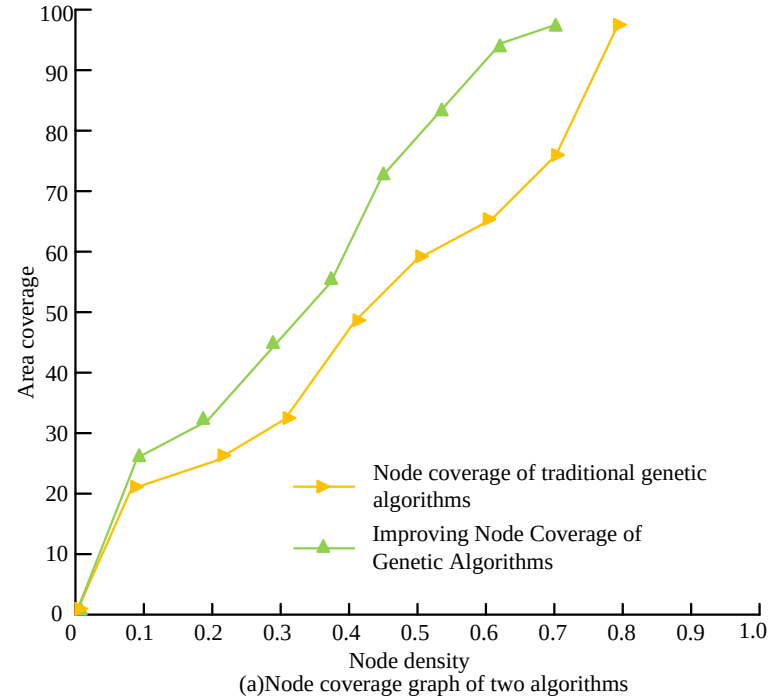


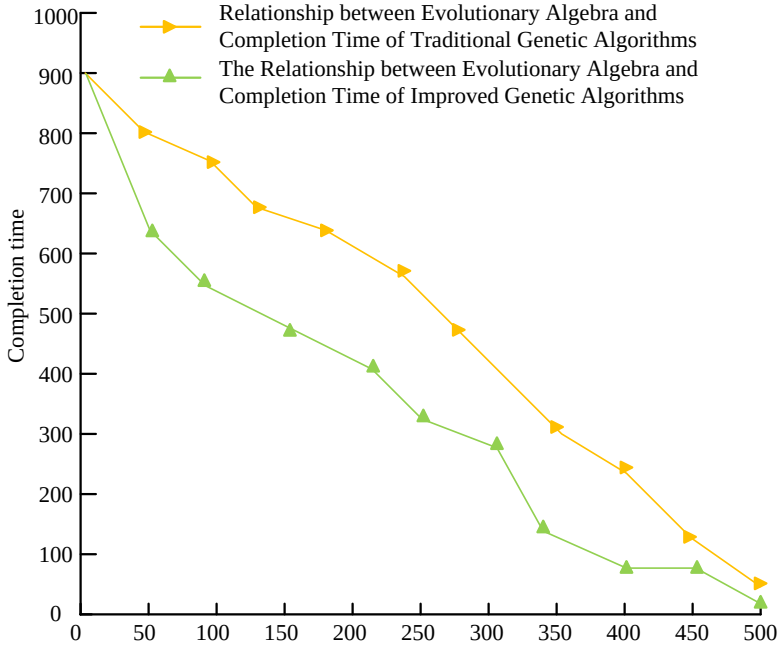
Figure 11: Relationship between network connectivity and average error value

The average errors of TGA, centroid algorithm, APIT, and IGA in Figure 11 are 5.41%, 4.49%, 3.71%, and 2.41%, respectively ($P<0.05$). This indicates that regardless of changes in network connectivity, IGA's positioning ability is always higher than the other three algorithms. Figure 12 is the graph about connection between node coverage, evolutionary algebra, and completion time of TGA and IGA.

The node coverage of the two algorithms in Figure 12(a) shows: in the same node density, the IGA has a higher regional coverage. Figure 12(b) displays the relationship between the number of evolutions and completion time of both algorithms over



(a)Node coverage graph of two algorithms



(b)Evolution algebra and completion time graph of two algorithms

Figure 12: The relationship graph between node coverage, evolutionary generations, and completion time of two algorithms

successive iterations of the genetic algorithm. As the iterations progress, the number of evolutions gradually increases while the time required to complete all iterations decreases. However, with the same evolutionary algebra, IGA takes less time. This shows the superiority and stability of the IGA. To further analyze the adaptability and superiority of the research method, further application analysis is conducted in a large-scale wireless sensor network node positioning scenario with a side length of 500m in a region. The total number of sensor nodes in the region is 500, including 100 anchor nodes and 400 unknown nodes. The results of the large-scale application analysis are shown in Table 2.

Table 2: Results of application analysis in major scenarios

Metrics	IGA	APIT Algorithm	Centroid Algorithm
Average Positioning Error (%)	2.45	5.62	7.89
Positioning Time (s)	12.3	18.4	9.6
Number of Iterations	75	120	60
Convergence Speed	Fast (0.5 fitness variation)	Medium (0.8 fitness variation)	Slow (1.2 fitness variation)
Node Coverage Rate (%)	98.6	90.5	85.3

As shown in Table 2, the average positioning error of IGA is 2.45%, significantly lower than the 5.62% of APIT algorithm and the 7.89% of centroid algorithm ($P<0.05$). This indicates that IGA can effectively improve the accuracy of node localization in large-scale scenarios and is suitable for complex and high-precision network environments. The positioning time of IGA is 12.3 seconds, which is between 18.4 seconds of APIT and 9.6 seconds of centroid algorithm ($P<0.05$). Although the computational complexity is slightly higher than that of the centroid algorithm, IGA improves efficiency by optimizing the evolution process, enabling it to maintain fast computational speed while achieving high-precision positioning. After 75 iterations, IGA achieves convergence, which is faster than the APIT algorithm's 120 iterations ($P<0.05$), demonstrating the advantages of IGA's dynamic parameter adjustment and elite strategy in the search process. Although the centroid algorithm has fewer iterations, its accuracy is significantly insufficient ($P<0.05$). The node coverage rate of IGA reaches 98.6%, which is much higher than the 90.5% of APIT algorithm and the 85.3% of centroid algorithm ($P<0.05$). This indicates that IGA has better coverage performance in large-scale networks and can optimize node positioning layout more comprehensively.

4.3 Discussion

This study has designed an IGA that effectively improves the accuracy and stability of node localization in wireless sensor networks through techniques such as dynamic parameter adjustment, fitness function optimization, and elite strategy. Compared with traditional centroid algorithms and APIT algorithms, IGA exhibits significant advantages in key performance indicators. Specifically, the average positioning error of IGA was 2.45%, which was much lower than APIT's 5.62% and centroid algorithm's 7.89%, indicating that IGA has significant advantages in node positioning accuracy. At the same time, IGA had a faster convergence speed, requiring only 75 iterations to reach the optimal solution, with a stable fitness value change (0.5). APIT and centroid algorithms required 120 and 60 iterations, respectively, and had slower convergence speeds. In addition, IGA achieved a node coverage rate of 98.6%, significantly higher than APIT (90.5%) and centroid algorithm (85.3%), demonstrating its applicability and advantages in large-scale complex network environments. The reason why IGA outperforms traditional methods in terms of positioning error and fitness values is mainly due to several key technological innovations. Firstly, the dynamic parameter adjustment mechanism can dynamically adjust the crossover probability and mutation probability based on the fitness value, thereby balancing global and local search and preventing the algorithm from getting stuck in local optimal solutions. This is consistent with the ideas of Yu et al. [20]. Secondly, fitness function optimization reduces computational complexity and enhances robustness to outliers by introducing absolute error instead of traditional square error, enabling the algorithm to approach the global optimal solution more quickly. In addition, the elite strategy ensures the retention of high fitness individuals and reduces the loss of high-quality solutions. The uniform distribution of the initial population within the communication area improves search efficiency and reduces ineffective calculations caused by random initialization. The results obtained are consistent with Singh et al.'s study [21]. These improvements effectively address common pitfalls of TGAs, such as local optima and premature convergence, enabling IGA to exhibit higher stability and accuracy in complex dynamic network environments. The core innovation of IGA lies in combining the local improvement of TGAs with global search, which is suitable for non-standard situations such as uneven node

distribution, limited number of anchor nodes, and complex conditions such as changes in communication radius. In practical applications, IGA demonstrates good stability and adaptability by flexibly adjusting parameters and optimizing search space. In previous studies, TGAs often faced local optimal traps, leading to premature convergence of the algorithm. The study aims to enhance population diversity, reduce the interference of outliers on the search process, and accelerate the convergence of the global optimal solution by providing low fitness individuals with moderate opportunities for crossover and mutation. The research provides a more stable, accurate, and efficient solution for node localization and optimization in complex network environments.

5 Conclusion

The high-speed development of computer network technology has caused tremendous changes in people's production and life. Currently, computer network optimization still has the problem of low positioning accuracy of network nodes. To solve the related problems, this study constructed an IGA model and applied it to computer network optimization. Experimental results showed that IGA has significantly improved location coverage and average location error compared to centroid algorithm and APIT. The coincidence rate of the improved algorithm was 8.16% higher than centroid algorithm's and 4.69% higher than that of the APIT algorithm. The maximum error of IGA was 19% lower than the centroid algorithm, and the overall average error was 8.8% lower. Compared to APIT, the maximum error of IGA was 7% lower, and the average overall error was 3.8% lower. Under the same parameters, TGA, centroid algorithms, APIT algorithms, and IGA were used to compare the performance of network nodes in computer networks. Experimental data were obtained: the fitness value of IGA, the amount of anchor nodes and the average error, the communication radius and the average error, and the network connectivity and the average error were 4.26, 0.68, 2.46, and 2.41, respectively. IGA had a significant improvement over the calculated values corresponding to the three algorithms, which proves the accuracy and stability of the improved genetic positioning algorithm.

6 Abbreviated List

NT: Network Topology
PSOGA: Particle swarm optimization and genetic algorithm
RLN: Random linear network
IGA: Genetic Algorithm
TGA: Traditional Genetic Algorithm
APIT: Approximate Point In Triangulation Test

References

[1] F. Wang, X. Lai, and N. Shi, "A multi-objective optimization for green supply chain network design," Decision Support Systems, vol. 51, no. 2, pp. 262-269. https://doi.org/10.1016/j.dss.2010.11.020

[2] Q. Liu, Z. Guo, and J. Wang, "A one-layer recurrent neural network for constrained pseudoconvex optimization and its application for dynamic portfolio optimization," Neural Networks, vol. 26, pp. 99-109, 2012. https://doi.org/10.1016/j.neunet.2011.09.001

[3] J. L. Ribeiro Filho, P. C. Treleaven, and C. Alippi, "Genetic-algorithm programming environments," Computer, vol. 27, no. 6, pp. 28-43, 1994. https://doi.org/10.1109/2.294850

[4] C. D. Lin, C. M. Anderson-Cook, M. S. Hamada, L. M. Moore, and R. R. Sitter, "Using genetic algorithms to design experiments: a review," Quality and Reliability Engineering International, vol. 31, no. 2, pp. 155-167, 2015. https://doi.org/10.1002/qre.1591

[5] W. B. Fan, F. Xiao, X. B. Chen, L. Cui, and S. Yu, "Efficient virtual network embedding of cloud-based data center networks into optical networks," IEEE Transactions on Parallel and Distributed Systems, vol. 32, no. 11, pp. 2793-2808, 2021. https://doi.org/10.1109/TPDS.2021.3075296

[6] B. Rajendran, and S. Venkataraman, "Detection of malicious network traffic using enhanced neural network algorithm in big data," International Journal of Advanced Intelligence Paradigms, vol. 19, no. 3-4, pp. 370-379, 2021. https://doi.org/10.1504/ijaip.2021.116366

[7] A. Xiaokaiti, Y. Qian, and J. Wu, "Efficient data transmission for community detection algorithm based on node similarity in opportunistic social networks," Complexity, vol. 2021, pp. 1-18, 2021. https://doi.org/10.1155/2021/9928771

[8] R. Alsaqour, S. Kamal, M. Abdelhaq, Y. Zan, and D. Jerou, "Genetic algorithm routing protocol for mobile ad hoc network," CMC Computers Materials Continua, vol. 68, no. 1, pp. 941-960, 2021. https://doi.org/10.32604/cmc.2021.015921

[9] B. Bu, "Mult-task equilibrium scheduling of internet of things a rough set genetic algorithm," Computer Communications, vol. 184, pp. 42-55, 2022. https://doi.org/10.1016/j.comcom.2021.11.027

[10] R. Y. Wu, J. M. Ma, Z. X. Tang, X. H. Li, and K. K. R. Choo, "A generic secure transmission scheme based on random linear network coding," IEEE ACM Transactions on Networking, vol. 30, no. 2, pp. 855-866, 2021. https://doi.org/10.1109/TNET.2021.3124890

[11] W. C. Chang, and I. H. R. Jiang, "iClaire: A fast and general layout pattern classification algorithm with

clip shifting and centroid recreation," IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, vol. 39, no. 8, pp. 1662-1673, 2019. https://doi.org/10.1109/TCAD.2019.2917849

[12] T. Ganesan, and P. Rajarajeswari, "Efficient sensor node connectivity and target coverage using genetic algorithm with Daubechies 4 lifting wavelet transform," International Journal of Communication Networks and Distributed Systems, vol. 28, no. 3, pp. 337-364, 2022. https://doi.org/10.1504/ijcnds.2022.122170

[13] S. T. Shishavan, and F. S. Gharehchopogh, "An improved cuckoo search optimization algorithm with genetic algorithm for community detection in complex networks," Multimedia Tools and Applications, vol. 81, no. 18, pp. 25205-25231, 2022. https://doi.org/10.1007/s11042-022-12409-x

[14] B. Nahavandi, M. Homayounfar, A. Daneshvar, and S. Mohammad, "Hierarchical structure modelling in uncertain emergency location-routing problem using combined genetic algorithm and simulated annealing," International Journal of Computer Applications in Technology, vol. 68, no. 2, pp. 150-163, 2022. https://doi.org/10.1504/ijcat.2022.123466

[15] Z. Sabir, M. R. Ali, and R. Sadat, "Gudermannian neural networks using the optimization procedures of genetic algorithm and active set approach for the three-species food chain nonlinear model," Journal of Ambient Intelligence and Humanized Computing, vol. 14, no. 7, pp. 8913-8922, 2023. https://doi.org/10.1007/s12652-021-03638-3

[16] C. Zhao, W. X. Zhu, G. Qiao, and F. Zhou, "Optimisation method with node selection and centroid algorithm in underwater received signal strength localization," IET Radar, Sonar Navigation, vol. 14, no. 11, pp. 1681-1689, 2020. https://doi.org/10.1049/iet-rsn.2020.0178

[17] Y. Zou, "Coupled neural networks and genetic algorithms application in the field of mine fire extinguishing," Informatica, vol. 48, no. 16, 2024. https://doi.org/10.31449/inf.v48i16.6317

[18] Y. M. Wu, Z. Li, C. X. Sun, Z. B. Wang, D. S. Wang, and Z. W. Yu, "Measurement and control of system resilience recovery by path planning based on improved genetic algorithm," Measurement and Control, vol. 54, no. 7-8, pp. 1157-1173, 2021. https://doi.org/10.1177/00202940211016094

[19] Y. Zhou, "Structural damage identification of large-span spatial grid structures based on genetic algorithm," Informatica, vol. 48, no. 17, 2024. https://doi.org/10.31449/inf.v48i17.6428

[20] M. Yu, and S. Chai, "Adaptive iterative learning control for discrete time nonlinear systems with multiple iteration varying high order internal models," International Journal of Robust and Nonlinear Control, vol. 31, no. 15, pp. 7390-7408, 2021. https://doi.org/10.1002/rnc.5690

[21] G. Singh, V. K. Tewari, R. R. Potdar, and S. Kumar, "Modeling and optimization using artificial neural network and genetic algorithm of self‐propelled machine reach envelope," Journal of Field Robotics, vol. 41, no. 7, pp. 2373-2383, 2024. https://doi.org/10.1002/rob.22255