Hindawi
Wireless Communications and Mobile Computing
Volume 2022, Article ID 1917172, 11 pages
https://doi.org/10.1155/2022/1917172

Research Article

Mathematical Model and Genetic Algorithm in Computer Programming Optimization and Network Topology Structure

Yonghong An

College of Continuing Education, Hulunbuir University, Hailar, 021008 Inner Mongolia, China

Correspondence should be addressed to Yonghong An; hkoury19850@student.napavalley.edu

Received 21 April 2022; Revised 11 June 2022; Accepted 26 July 2022; Published 12 August 2022

Academic Editor: Chia-Huei Wu

Recent emphasis has been focused on the industrial Internet of Things (Ind-IoT) in the context of the Fourth Industrial Revolution (Industry 4.0). IoT devices are used in the Ind-IoT to increase manufacturing productivity. The problem, however, is that these instruments will create enormous volumes of data records that must be handled efficiently. Cloud computing (CC) is frequently cited as a viable option for providing effective support for Ind-IoT applications. However, the high-latency and unstable connectivity problem between the cloud and Ind-IoT endpoints continues to plague Ind-IoT operations. Fog computing (FC), which extends computation and storage to the edge of the network, is a possible answer to these problems. Cloud-fog integrated Internet of Things (CFI-Ind-IoT) is discussed in this study as an approach to integrating FC with cloud-based industrial Internet of Things (Ind-IoT). A constrained multiparent crossover genetic algorithm (CMPC-GA) for optimization of the load adjusting challenge in the distributed cloud-fog network is proposed in order to attain ultralow response latency in the CFI-Ind-IoT system. Furthermore, we develop a duty reallocation and retransmission method in order to lower the average delivery latency of the CFI-Ind-IoT architecture due to the unreliable scenario. Effectiveness measurements show that the CMPC-GA technique can deliver ultralow latency functionality in a typical Ind-IoT.

1. Introduction

The Ind-IoT devices create enormous amounts of data that need to be analyzed. Even in the industry 4.0 revolution, there is an increasing need for real-time data processing for industrial IoT systems, such as smart factories. It is, therefore, critical to have a strong data center in the Ind-IoT [1].

Indeed, CC is often regarded as a critical enabler for satisfying the objectives of Industrial IoT systems. Cloud-based IoT networks, on the other hand, continue to encounter a number of unresolved issues. Transmission delay is tremendous since cloud data centers are constantly placed remotely. Intelligent services create a great amount of data, which puts a strain on cloud servers (CSs), and any malfunction in the network might cause a widespread communication issue [2].

In the cloud-based Ind-IoT, FC appears to be a potential way to overcome the aforesaid difficulties. To minimize latency, FC processes workloads locally on fog nodes (FNs) located near the terminals, which enables a new breed of IoT apps and services that demand low latency, mobility, and geo-distribution. Therefore, in the context of Industry 4.0 [3], we propose the cloud-fog-based Ind-IoT (CFI-IoT) network architecture, which utilizes the present edge devices to establish an FC layer in the CFI-IoT architecture to fulfill the needs of latency-sensitive Ind-IoT operations [4].

FC and enhanced CC have received a lot of attention recently. A comparison of the FC's low latency and power-saving characteristics with that of regular CC is shown in. A revolutionary C-RAN design with mobile CC was proposed in [5] by writers who concentrated on the problem of service allocation inside the Combined Fog-Cloud architecture. To learn more about how much power Ind-IoT consumes and how it may be put to use in the real world, check out the works of [6]. For indoor wireless coverage, [7] investigates unexpected spectrum access methods. Reference [8] proposes a management strategy for multimodal sensor networks that is both efficient and successful. In order to reduce service latency even further, these studies did not examine how to link many fog terminals and the cloud platform

[7]. To further reduce latency, we suggest in this work that numerous FNs and the CS be linked together to conduct computationally expensive real-time operations in a distributed way [9–11]. Load balancing is a significant tool for reducing latency in distributed computing [12]. As a result, we are looking at how several FNs and a CS balance load. A load balancing technique that uses genetic algorithms [13] for constrained optimization (CMPC-GA) is also being proposed for consideration.

The FN, which lacks resources, is particularly vulnerable to attack. In addition, faulty network connections might be caused by shaky wireless transmission lines [14]. It is impossible for a single FN to handle the current burden [15]. This is why we suggest a system for reallocating uncompleted subtasks on failure nodes and retransmitting the new sequence of operations to normal nodes in order to ensure that the task may be completed on time [16]. When an FN fails, the simulation results suggest that the redistribution and retransmission technique can minimize average service latency [17].

The rest of this paper's reminder will go like this: the CFI-Ind-IoT architecture is shown in Section 2. Using the CFI-Ind-IoT latency mathematical model, we suggest utilizing the CMPC-GA technique in Section 3 to address the issue. Section 4 displays the results of the evaluations of employee performance. Finally, in Section 5, we summarize this study.

2. CFI-IND-IoT Structure

Low latency is essential for our CFI-Ind-IoT architecture (shown in Figure 1); hence, we have included FC in the cloud.

The cloud platform layer, the FC layer, and the infrastructure layer make up the architecture. As the foundation of an architect's design lies in its sensor network, industrial machinery, conveyor networks, and intelligent industrial robots and manipulators, the infrastructure layer is comprised of these elements and many more. The infrastructure's job is to carry out certain production processes, such as data collection, production, and transportation.

There are several edge network devices (such as gateways and routers) with relatively low computation and storage capacity in the FC layer, which play an essential role in the design of the CFI-Ind-IoT. The gates are designated as the FNs in this design. FNs connect with each other wirelessly in order to speed up the implementation and development of the fog network.

According to the infrastructure layer needs, FNs interface with the cloud to get relevant information and service, store the product information supplied via them, and upload the useful product information and updates to accomplish global data sharing. Furthermore, the FC layer may analyze a wide range of information (such as product data, user requirements, and measurement data) to decrease latency.

Moreover, we suggest distributing computing in a cluster of numerous FNs and CSs to achieve ultralow latency by balancing the load of each FN and cloud. Distributed computing paradigms that use job redistribution and retransmission mechanisms in the cloud-fog network are an excellent way to increase the architecture's real-time performance in the FN and cloud.

Servers in the high-performance cloud layer are suitable for protecting Ind-IoT data as well as exchanging global information and performing simple data mining activities. The cloud is also viewed as a processing core in our design in order to boost the architecture's computing capability.

3. Interactive Load Balancing Strategy in CFI-Ind-IoT

To address the issue of latency in the CFI-Ind-IoT, we recommend using the CMPC-GA method, which we describe in this section.

3.1. CFI-Ind-IoT Latency Mathematical Model. The path planning procedure for an industrial robot using laser navigation, for example, involves a slew of data processing [18]. We examine the CFI-Ind-IoT architecture's data processing. First, the robot continuously records the course it takes. In order to execute rapidly distributed computing, the acquired real-time route data is sent to the localized fog devices and cloud.

Finally, the robot will get the path planning findings and use them to carry out its actions. A weighted node WUG $= (\mathrm{FN}, \mathrm{ES})$ can be used to represent the network architecture seen in Figure 2.

$$\mathrm{FN} = \{\mathrm{fn}_1, \mathrm{fn}_2, \cdots, \mathrm{fn}_\mathrm{k}, \mathrm{CS}\}, \tag{1}$$

$$\mathrm{ES} = \left\{\mathrm{es}_{\mathrm{fn}_1,\mathrm{fn}_2}, \cdots, \mathrm{es}_{\mathrm{fn}_{\mathrm{k}\text{-}1},\mathrm{fn}_\mathrm{k}}\right\}. \tag{2}$$

Vectors fn_k and CS indicate the FNs and CSs, respectively, in the set of vertices FN showed in Figure 2. A CS has been added to the CFI-Ind-IoT design in order to increase its computational power. Each FN's computational power is indicated by CS. $\mathrm{es}_{\mathrm{fn}_i,\mathrm{fn}_j}$ is a wireless transmission connection between the nodes fn_i and fn_j represented by the edge set; each edge's weight (i.e., $\mathrm{es}_{\mathrm{fn}_i,\mathrm{fn}_j}$) represents the delay in transmission among fn_i and fn_j regarded to be a "master node" in the CFI-Industrial-IoT network since it receives requests from industrial endpoints for ultralow latency services first. Sub is then passed forward to the master node by these endpoints. It is possible to break down the computer effort into smaller subtasks, such as sub_i. For these duties, all the FNs are involved, including master fn_i and all of the slaves such as fn_j. As the last step, the results of computation were communicated back to industrial endpoints through the master node. As a result, CF-Ind-service IoT's latency t may be represented as [1]:

$$t = \max\left\{\frac{\mathrm{Sub}_i}{\mathrm{CS}_{\mathrm{fn}_i}} + \mathrm{Wg}_{\mathrm{fn}_i,\mathrm{fn}_j} h_{\mathrm{fn}_i,\mathrm{fn}_j}, \frac{\mathrm{Sub}_{\mathrm{cs}}}{\mathrm{CS}_{\mathrm{cs}_i}} + \mathrm{Wg}_{\mathrm{fn}_i,\mathrm{cs}} h_{\mathrm{fn}_i,cs}\right\} i, \quad j = 1, 2, \cdots, k, \tag{3}$$

where $\mathrm{Wg}_{\mathrm{fn}_i,\mathrm{fn}_j}$ is the transmission latency among fn_i and fn_j and Sub_i is the computation delay of the specific task

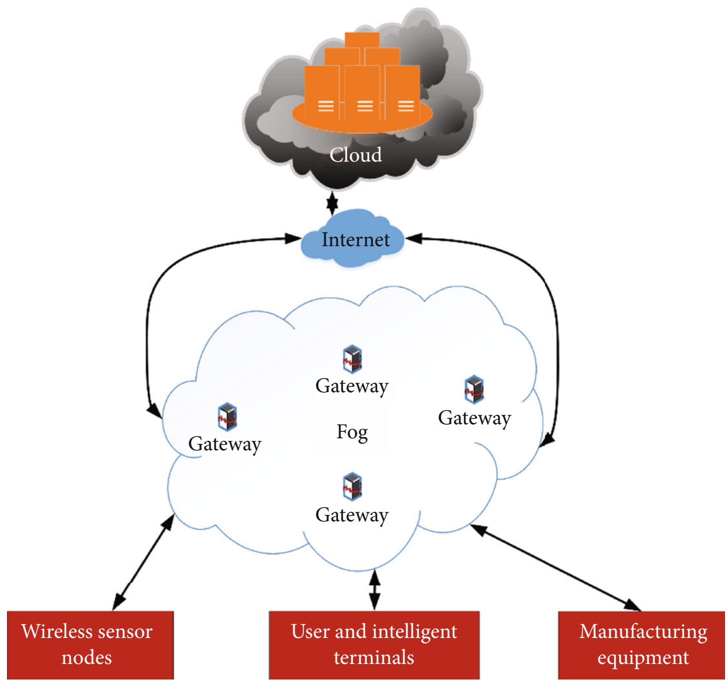

Figure 1: The CFI-Ind-IoT architecture.

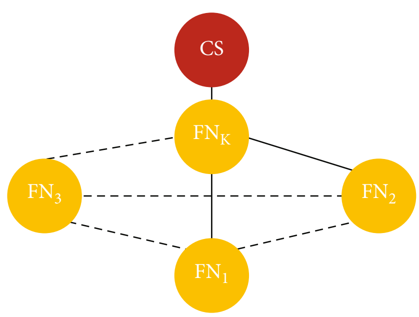

Figure 2: A graph with weighted undirected.

Sub_i on the fn_i. When it comes to the subtask assignment connection between fn_i and fn_j, it is denoted by the value of $h_{\mathrm{fn}_i,\mathrm{fn}_j}$: 1 indicates that the relationship exists, while zero indicates that it does not. This is a measure of how long it takes the $\mathrm{CS}_{\mathrm{cs}_i}$ to process data from the subtask $\mathrm{Sub}_{\mathrm{cs}}$, and it indicates the time it takes for the master fn_j to communicate with the cloud CS. The wireless communication delay between the CFI-Ind-IoT nodes, $\mathrm{Wg}_{\mathrm{fn}_i,\mathrm{fn}_j}$ and $\mathrm{Wg}_{\mathrm{fn}_i,\mathrm{cs}}$, under the ARQ protocol, may be defined as follows [1]:

$$\mathrm{Wg}_{\mathrm{fn}_i,\mathrm{fn}_j} = \frac{\mathrm{Sub}_i}{\mathrm{DTR}_i} \times \frac{1+\mathrm{PE}_i}{1-\mathrm{PE}_i}, \tag{4}$$

$$\mathrm{Wg}_{\mathrm{fn}_i,\mathrm{cs}} = \frac{\mathrm{Sub}_c}{\mathrm{DTR}_c} \times \frac{1+\mathrm{PE}_c}{1-\mathrm{PE}_c}. \tag{5}$$

There are two subtasks that are sent to the node fn_i and CS through Sub_i and Sub_c. To achieve ultralow latency, we should identify an optimum task allocation technique, namely, find a set of ideal subtasks $\{\mathrm{Sub}_1, \mathrm{Sub}_2, \cdots, \mathrm{Sub}_c\}$. Finally, the CFI-Ind-load IoT's balancing problem may be expressed as an optimization problem:

$$\begin{aligned} \min\max \Bigg\{ & \frac{\mathrm{Sub}_i}{\mathrm{CS}_{\mathrm{fn}_i}} + \left(\frac{\mathrm{Sub}_i}{\mathrm{DTR}_i} \times \frac{1+\mathrm{PE}_i}{1-\mathrm{PE}_i}\right) h_{\mathrm{fn}_i,\mathrm{fn}_j}, \frac{\mathrm{Sub}_{\mathrm{cs}}}{\mathrm{CS}_{\mathrm{cs}_i}} \\ & + \left(\frac{\mathrm{Sub}_c}{\mathrm{DTR}_c} \times \frac{1+\mathrm{PE}_c}{1-\mathrm{PE}_c}\right) h_{\mathrm{fn}_i,\mathrm{cs}} \Bigg\} i, \quad j = 1, 2, \cdots, k, \end{aligned} \tag{6}$$

$$\begin{aligned} h_{\mathrm{fn}_i,\mathrm{fn}_j} &= \begin{cases} 0, & \mathrm{Sub}_i = 0, \\ 1, & \mathrm{Sub}_i \neq 0, \end{cases} \\ h_{\mathrm{fn}_i,\mathrm{cs}} &= \begin{cases} 0, & \mathrm{Sub}_c = 0, \\ 1, & \mathrm{Sub}_c \neq 0. \end{cases} \end{aligned} \tag{7}$$

3.2. Mathematical Model for the Average Service Delay in the Event of System Failure of FNs in Ind-IoT. Uncertainty is evident in the Ind-IoT, such as FNs being damaged or wireless connectivity failing. The FN damages likelihood and the wireless connection outage probability are jointly referred to as the prediction error of a single FN in this research. For each FN, we assume that it has a failure chance of pi. In the event that several FNs fail, the job will be impossible to finish unless a mechanism is implemented. To reduce latency when certain FNs fail, we suggest reallocating and retransmitting the appropriate subtasks to the regular FNs and cloud server for distributed computing. The reassignment and retransmission method could ensure timely and accurate completion of the jobs. CF-average Ind-IoT's service latency might be mathematically described as Equation (8) if the FNs in the system fail according to the method presented [1].

$$t_{\text{ave}}=\sum\prod_{\text{fn}_n\in\text{FN-FN}'}\text{PF}_n\prod_{\text{fn}_i\in\text{FN}'}(1-\text{PF}_i)\times\left(\min\max\left\{\frac{\text{Sub}_i}{\text{CS}_{\text{fn}_i}}+\left(\frac{\text{Sub}_i}{\text{DTR}_i}\times\frac{1+\text{PE}_i}{1-\text{PE}_i}\right)h_{\text{fn}_i,\text{fn}_j},\frac{\text{Sub}_{\text{cs}}}{\text{CS}_{\text{cs}_i}}+\left(\frac{\text{Sub}_c}{\text{DTR}_c}\times\frac{1+\text{PE}_c}{1-\text{PE}_c}\right)h_{\text{fn}_i,\text{cs}}\right\}+\min\max\left\{\frac{\text{Sub}_i'}{\text{CS}_{\text{fn}_i}}+\left(\frac{\text{Sub}_i'}{\text{DTR}_i}\times\frac{1+\text{PE}_i}{1-\text{PE}_i}\right)h_{\text{fn}_i,\text{fn}_j},\frac{\text{Sub}_{\text{cs}}'}{\text{CS}_{\text{cs}_i}}+\left(\frac{\text{Sub}_c'}{\text{DTR}_c}\times\frac{1+\text{PE}_c}{1-\text{PE}_c}\right)h'_{\text{fn}_i,\text{cs}}\right\}\right), \tag{8}$$

where

$$h_{\text{fn}_i,\text{fn}_j}=\begin{cases}0, & \text{Sub}_i=0,\\ 1, & \text{Sub}_i\neq 0,\end{cases} \tag{9}$$

$$h_{\text{fn}_i,\text{cs}}=\begin{cases}0, & \text{Sub}_c=0,\\ 1, & \text{Sub}_c\neq 0,\end{cases} \tag{10}$$

$$\text{fn}_n\in\text{FN-FN}', \tag{11}$$

$$0\le\text{Sub}_i,\text{Sub}_c,\text{Sub}_i{}',\text{Sub}_c{}'\le\text{Sub}, \tag{12}$$

$$\sum\text{Sub}_i+\text{Sub}_c+\sum\text{Sub}_i{}'+\text{Sub}_c{}'=\text{Sub}, \tag{13}$$

$$h'_{\text{fn}_i,\text{fn}_j}=\begin{cases}0, & \text{Sub}_i'=0,\\ 1, & \text{Sub}_i'\neq 0,\end{cases} \tag{14}$$

$$h'_{\text{fn}_i,\text{cs}}=\begin{cases}0, & \text{Sub}_c'=0,\\ 1, & \text{Sub}_c'\neq 0.\end{cases} \tag{15}$$

3.3. GA-Based Load Balancing Algorithm. As a solution to Equation (6)'s load balancing and Equation (8)'s latency minimization, we use a limited CMPC-GA. An optimization issue can be represented by the X_i variables as an array of alternative solutions, each of which would be given a random real number as an initialization parameter in the CMPC-GA. This process is repeated over and over again until an ideal individual is discovered. The CMPC-GA may be used to solve optimization problems with restrictions, such as inequality and equality, such that the constrained optimization issues can be transformed into unconstrained optimization problems. The CMPC-GA is described in the following paragraphs. The CMPC-fitness GA's function differs from the typical real-coded GA in the following ways:

$$\text{fit}(x)=\begin{cases}t(x) & x\in\text{FR},\\ t(x)+\text{PF}\sum_{j=1}^{k+2}t_j(x)+\psi(x,\text{gen}) & x\in\text{IFR}.\end{cases} \tag{16}$$

In the search space S, the feasible zone is FR, whereas the infeasible region is IFR. It is important to notice that for each successive constraint, PF represents the penalized factor, $t_j(x)$ represents the infeasible people's constraint violation value for the jth limitation, and $\psi(x,\text{gen})$ represents an extra heuristic value for each successive generation's infeasible individuals. In which, $t_j(x)$ and $\psi(x,\text{gen})$ may be written as

$$t_j(x)=\begin{cases}\max(-x(j),0) & 1\le j\le k+1,\\ \left|\sum_{i=1}^{k+1}\text{Sub}-x(i)\right| & j=k+2,\end{cases} \tag{17}$$

$$\psi(x,\text{gen})=\text{Worst}(\text{gen})-\min_{x\in\text{IFR}}\left\{t(x)+\text{PF}\sum_{j=1}^{k+2}t_j(x)\right\}, \tag{18}$$

where

$$\text{Worst}(\text{gen})=\max\left\{\max_{x\in FR}\{t(x)\},\text{Worst}(\text{gen}-1)\right\}. \tag{19}$$

The performance index of the gth iteration viable individuals is represented by $t(x)$ in Equation (17). Using g generations of evolution, Worst(g) identifies the possible person with the best fitness. It is necessary to initially randomly initialize the search space S with real numbers in order to construct the one-dimensional real array of $k+1$ genes that makes up each chromosome, or individual x_i in the population, for the CMPC-GA algorithm. To assess the population, the fitness value of each member would be computed using Equation (16). As a further step, the genetic algorithms are used to bring the original population up to date again. And here are the specific genetic operators: selection: high-fitness people are selected from the existing population to be preserved for future generations. Using a two-tournament selection approach is employed in this study.

In a GA, the crossover is a key technique for passing on the original excellent genes to the next generation of children. When using the CMPC-GA, two new offspring are formed by the linear integration of the two-parent individuals. The arithmetic crossover is employed in this case. The following may be said about the relationship between a child and their parents. The CMPC-local GA's search capacity is determined by the mutation operation, which increases the population's variety. We use a nonuniform mutation operator in our article. Based on the load-balancing problem in Equation (6), the CMPC-GA algorithm is illustrated in the following subsection.

3.4. The CMPC-GA Algorithm. Charles Darwin's theory of natural evolution inspired Holland to create the GA in 1975 [19]. As with evolution, this algorithm selects only the most powered to propagate. Indefinitely, the population of solutions can be altered using this way. Individuals in the current population are chosen at random to be the

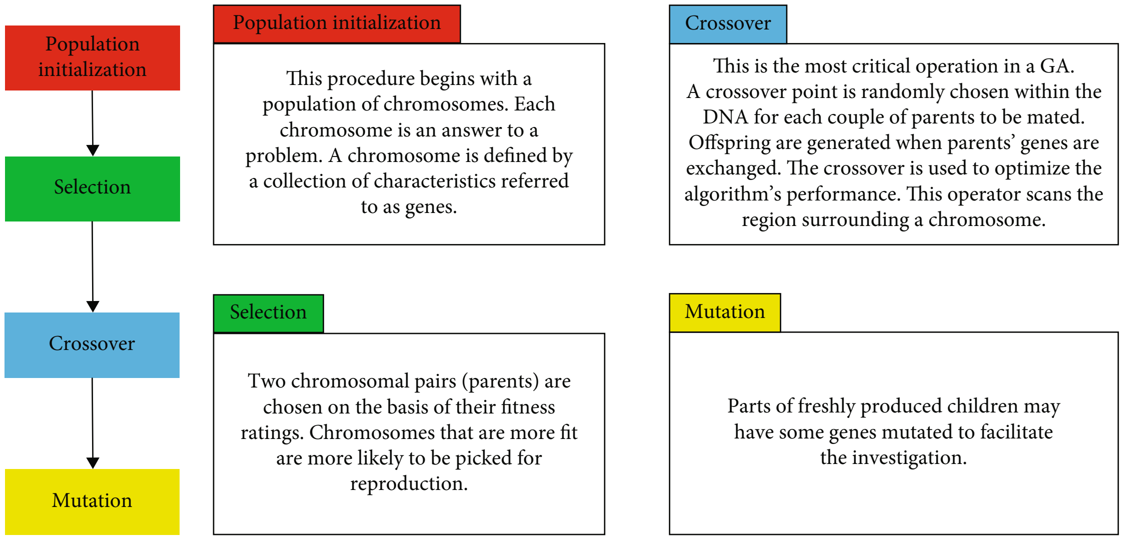

Figure 3: Four phases of a typical GA.

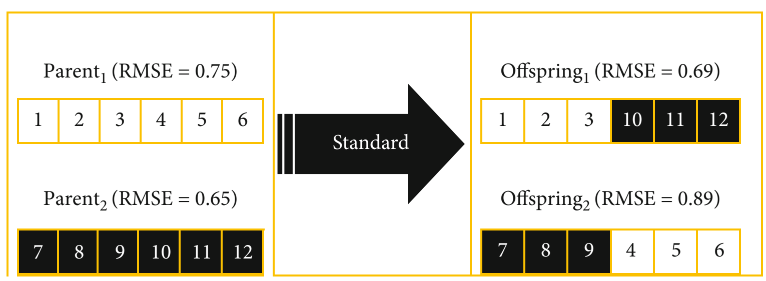

Figure 4: An example of a single-point crossover operator of standard GA.

parents of the next generation's children by GA. It takes several generations for the population to "evolve" towards an optimal answer. Figure 3 depicts the four phases of a typical GA.

Typical GA flaws include an inability to exploit and a propensity to become locked in local minimums. Although it may take longer for GA to converge, it does so more quickly when chromosomes move quickly and abruptly in the correct direction. Figure 4 depicts a conventional GA's single-point crossover. As can be seen, the resulting chromosomes are not much different from their parents when only two chromosomes are combined in a single-point crossover.

Traditional crossover has the drawback of only seeing in the direction of one or both of its parents. It is feasible to improve the number of viable solutions in GA by using a more optimal solution search. In order to improve algorithm exploitation and exploration, this paper presents a new MC operator. The updated MPCGA's MC is seen in Figures 5 and 6. It is clear that the RMSE is similar to the goal function, and the lower it is, the more precise the response will be.

Because of this, the offspring produced by using several best parents in MC-GA shows less resemblance to any one of their parents, indicating that they are more diverse (improves exploration). As a result, the algorithm may be more effectively used since the offspring inherit all of their genes from a variety of parents.

4. Experimental Results and Discussion

The CFI-Ind-IoT-based CMPC-GA retransmission and redistribution method is the subject of this section's performance assessments. According to [20], the CFI-Ind-IoT has four FNs configured for low latency. For the mobile communication parameters, we selected 802.11 ac protocol parameters. To create a realistic network simulation, a heterogeneous fog network with varying computer capacities is needed. Table 1 includes some relevant parameters. Also, Table 2 represents the CMPC-parameter GA's settings.

Our simulation has comprehensive service demand assignment information from intelligent manufacturing machinery. Task loads range from 0 MB to 400 MB, depending on the simulation configuration. The CMPC-GA technique allows us to obtain the job allocation ratio in advance, which reduces service delay. The workflow scheduling ratio allows for easy retrieval of the work allocation results when they become available. As a result, CMPC-running GA's time may be ignored. MATLAB is used to generate all of the simulation results, and each value is the average of a large number of tests.

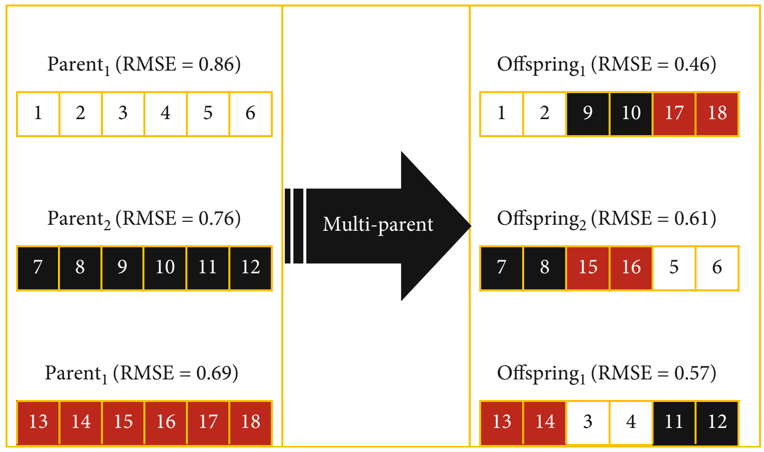

Figure 5: A typical MC (3 parents).

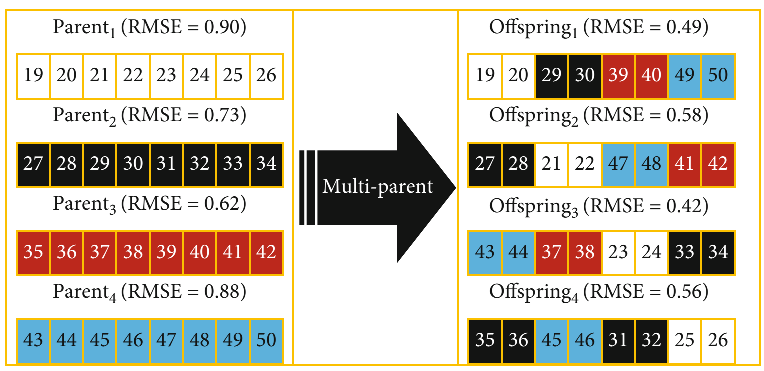

Figure 6: A typical MC (5 parents).

Table 1: Relevant parameters.

Parameter type	FN1	FN2	FN3	FN4	C
CFN_i/CCS (Gbps)	0.28	0.23	0.18	0.13	8
Transfer rate (Mbps)	190	190	180	195	18
Packet error rate	0.0201	0.0185	0.0198	0.0201	0.0082

Table 2: The CMPC-parameter GA's settings.

Parameters	Values
Population size	50
Crossover probability	0.8
Mutation probability	0.4
Worst (0)	96
Maximum number of generations	100

4.1. Ind-IoT Performance Comparison of Three Architectures. CFI-Ind-IoT, based on the CMPC-GA load-balancing algorithm, is tested against the cloud-based and fog-based architectures in Ind-IoT to verify the low latency efficiency of CFI-Ind-IoT. Latency comparison findings are depicted in Figure 7. It is becoming increasingly difficult for the cloud-based structure to keep up with the rising demands of the activity, resulting in a large increase in latency.

As a result of the cloud's distance from intelligent manufacturing endpoints and its restricted bandwidth, the fog-based structure offers lower latency than cloud infrastructure. CFI-Ind-service IoT's latency is shown in Figure 7 to be lower than that of fog-based architectures when the load is high. That is because the CFI-Ind-processing IoT's capability is enhanced by the CS's computational capacity. Compared to cloud-based and fog-based architectures, the latency effectiveness of CFI-Ind-IoT has improved by 93.98 percent and 36.67 percent when the job load is 400 Mb. Consequently, a low latency service may be provided by the CFI-Ind-IoT, which is ideal for Ind-IoT.

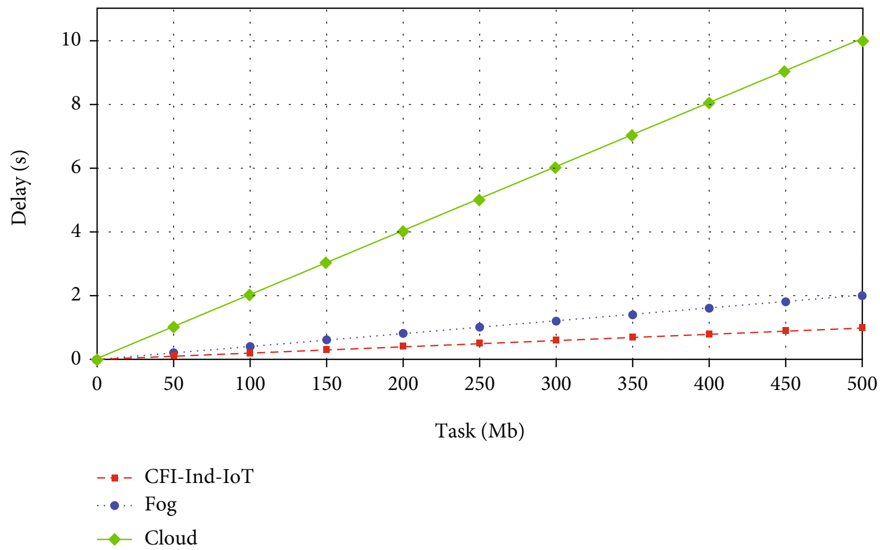

Figure 7: Comparison of the mentioned structures.

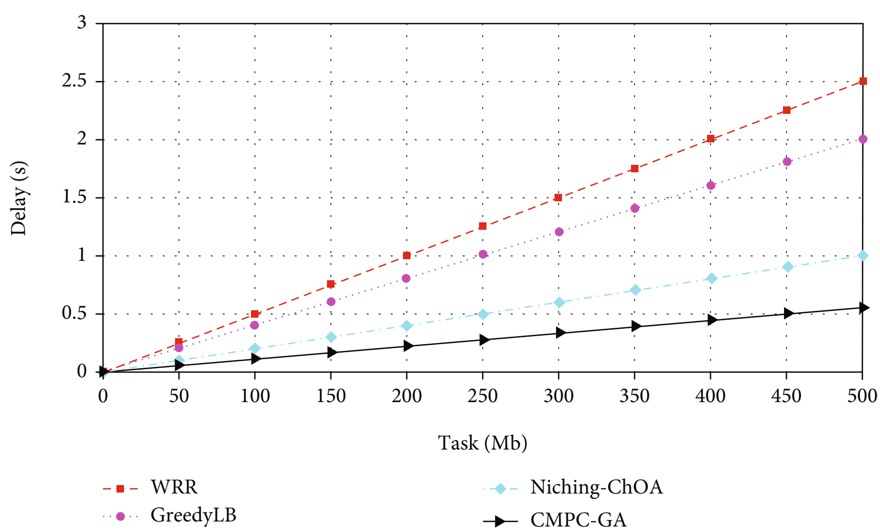

Figure 8: The comparison of latency effectiveness for various algorithms.

4.2. The Performance of Various Load Balancing Techniques in terms of Latency. Comparisons between CMPC-GA, WRR [21], Niching-ChOA [22], and Greedy scheduling algorithms [23] are made in the next section to demonstrate the great efficiency of CMPC-GA in minimizing the latency of CFI-Industry-IoT. In Figure 8, the simulation results are depicted.

There was a noticeable difference in latency between CMPC-GA and the other algorithms. To ensure global searching, CMPC-GA uses the crossover operation, whereas mutation is used for local searching. CMPC-global GA's load-balancing method reduces service latency because of this. When balancing the load, the WRR and GreedyLB algorithms do not take transmission latency into account; hence, the Niching-ChOA method may end up in the local optimum, resulting in increased latency. When the task size is 400 MB, the CMPC-GA method outperforms WRR, GreedyLB, and Niching-ChOA in terms of latency by 71.3%, 72.8%, and 42.8%. Consequently, it is shown that reducing latency in the CFI-Ind-IoT structure is made easier by using the CMPC-GA load balancing method [24, 25].

4.3. Failure of FNs Results in an Average Service Latency. Each FN is given a failure probability in the simulation. A failure example in which the CFI-Ind-IoT adopts task reallocation and retransmission, as well as another failure instance in which the CFI-Ind-IoT simply adopts the retransmission technique, is used to compare average service delay in order to evaluate the effectiveness of the CFI-Ind-IoT. CFI-Ind-IoT service latency is also compared in four different

Table 3: Fog nodes' failure probabilities for various conditions.

Environment	FN1	FN2	FN3	FN4	CS
Best	0	0	0	0	0
Good	0.002	0.14	0.021	0.009	0
General	0.29	0.28	0.39	0.29	0
Poor	0.5	0.59	0.71	0.74	0

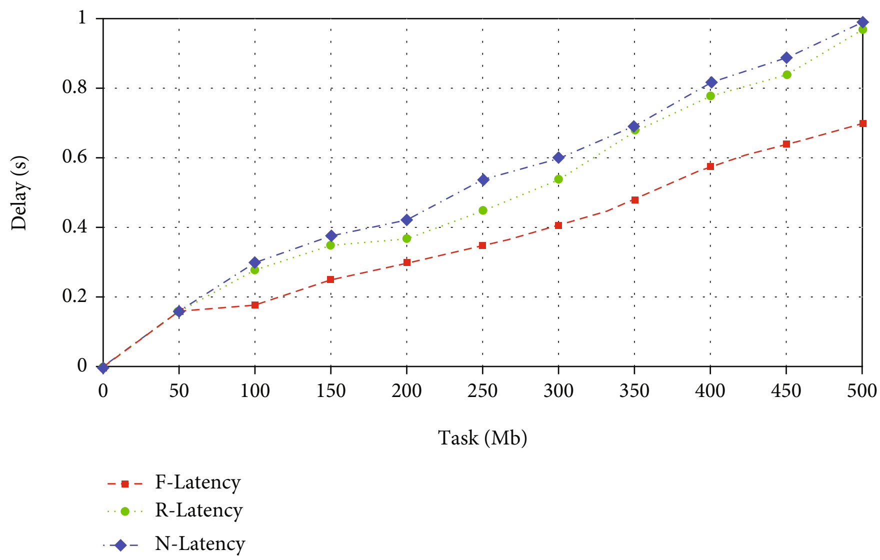

Figure 9: The comparison of service latency.

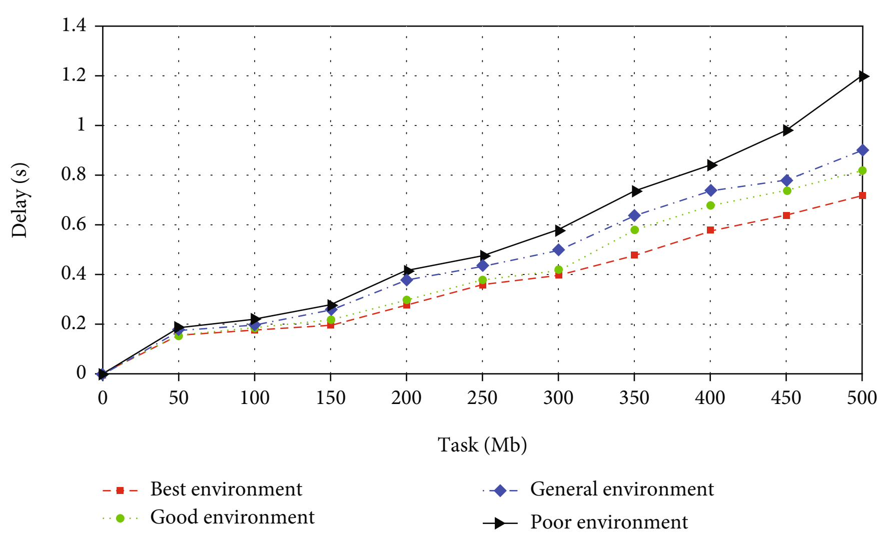

Figure 10: The comparison of service latency for various environments.

production situations where the failure probability of the fog nodes is varied to study the impact of the FNs on the service latency [26]. Table 3 shows the FNs' failure probability in four different contexts. Figures 9 and 10 illustrate the results of the computer simulations.

There are some FNs that fail when CFI-Ind-IoT adopts task reallocation and retransmission mechanisms (R_ Latency), and there are some FNs that fail when CFI-Ind-IoT only employs the retransmission mechanism (F_Latency) where the unimplemented subtasks on the failure FNs are

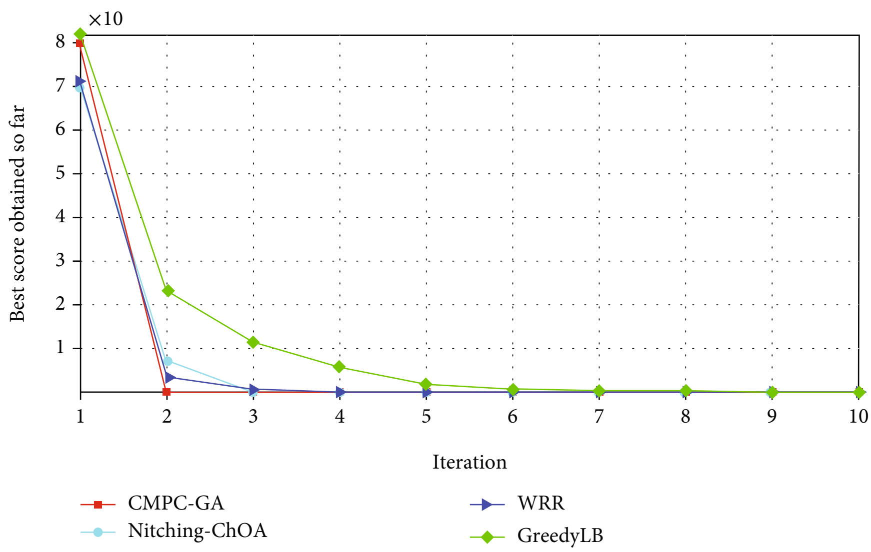

Figure 11: The comparison of convergence curves for population size = 20.

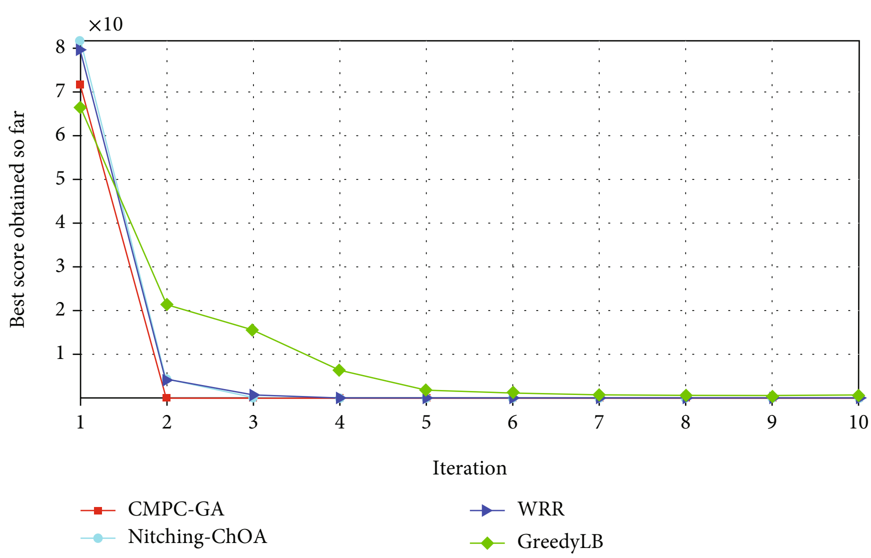

Figure 12: The comparison of convergence curves for population size = 40.

only resent to one normal node to analyze. In the event of a failure, the failure probability of the FNs is based on Table 3's good environment. We can see from Figure 5 that the work may be performed in a specific amount of time, rather than being unable to be done when FNs fail [27]. The latency in the failure scenario (R_Latency and F_Latency) is longer than the duration in the normal situation as the number of tasks increases (N_Latency).

To avoid increasing delay, it is necessary to reprocess activities that were left unfinished on the failed nodes on the regular nodes instead. The F_Latency is lower than the R_Latency, and the gap between the latency of the two rises as the job size increases. When the job is huge, it may be possible to lower the processing delay by distributing the unfinished work from the failed FNs to all of the normal nodes [28]. The R_Latency decreases by 9.02 percent when the processing load is 400 Mb, compared to the F_Latency. With this job reallocation and transmission mechanism, low latency service may be provided in case of FN malfunction in Ind-IoT using CFI-Ind.

In CFI-Ind-IoT, the average service delay is influenced by the failure probability of the FNs. The average service delay is compared among four types of production settings with varying failure rates. Table 3 displays the FNs' failure probability in four different contexts. In all four types of contexts, it is evident that the operation latency grows as the number of tasks increases. When the process demand is fixed, we can see that the average service delay rises as the risk of failure increases.

An average service delay in the CFI-Ind-IoT of 400 Mb was found to be lowered by 32.38 percent, 19.23 percent, and 14.21 percent compared to the latency in negative,

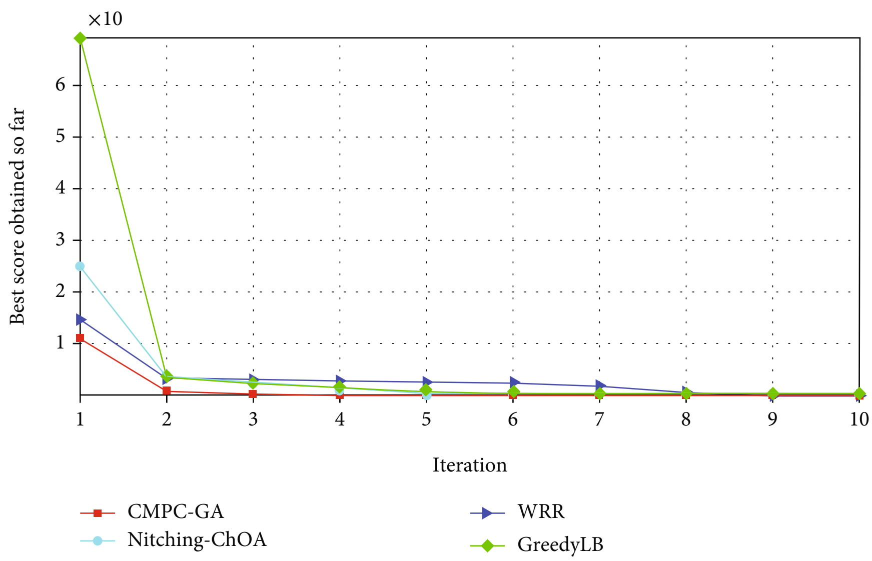

Figure 13: The comparison of convergence curves for population size = 60.

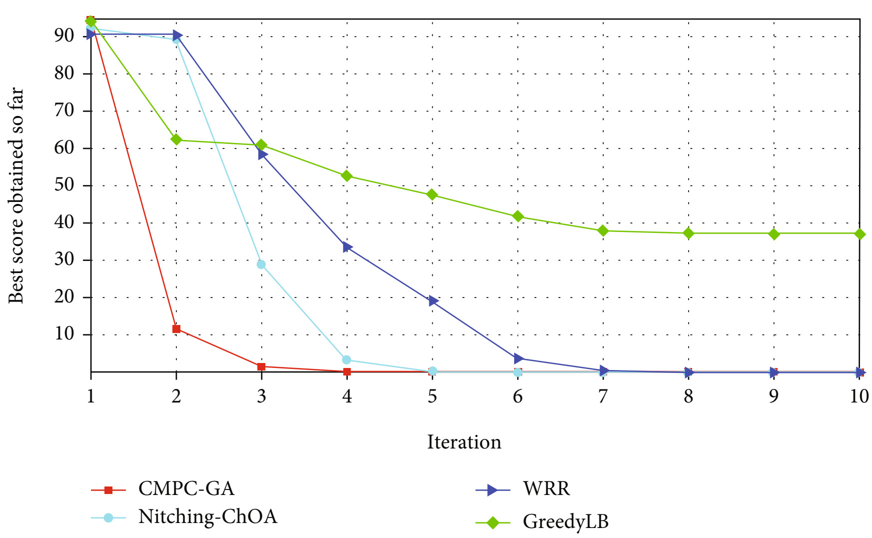

Figure 14: The comparison of convergence curves for population size = 80.

general, and excellent environments. The substantial failure rate of the FNs is expected to result in a big average service delay. It should be noted that the convergence curves for various population size are presented in Figures 11–14.

5. Conclusion

CFI-Ind-IoT network architecture is built by integrating FC with cloud-based Ind-IoT architecture and introducing cloud applications to the network. The CFI-Ind-IoT latency mathematical model is first established. In order to reduce latency, we implemented the CMPC-GA algorithm. For the CFI-Ind-IoT, we developed a demand redistribution and retransmission technique to decrease the average service delay in the event of FN failure. In the Ind-IoT, the simulation results suggest that the CFI-Ind-IoT based on the CMPC-GA architecture, as well as our proposed job reallocation and retransmission mechanism, may achieve ultralow latency performance. Research in the future will focus mostly on the modeling and improvement of cloud-fog network dependability. Using novel metaheuristic algorithms, including ChOA and DLFChOA, can be considered as another research direction.

Data Availability

Data sharing is not applicable to this article as no new data were created or analyzed in this study.

Conflicts of Interest

The author states that this article has no conflict of interest.

Acknowledgments

This work was supported by the Research Program of Science and Technology at Universities of Inner Mongolia Autonomous Region (Grant Nos. NJZY19231 and NJZZ21052).

References

[1] C. Shi, Z. Ren, K. Yang et al., "Ultra-low latency cloud-fog computing for industrial Internet of Things," in *In 2018 IEEE Wireless Communications and Networking Conference (WCNC)*, pp. 1–6, IEEE, Barcelona, Spain, 2018.

[2] L. Qian, Z. Luo, Y. Du, and L. Guo, "Cloud computing: an overview," in *In IEEE international conference on cloud computing*, vol. 5931, pp. 626–631, Springer, Berlin, Heidelberg, 2009.

[3] H. Lasi, P. Fettke, H.-G. Kemper, T. Feld, and M. Hoffmann, "Industry 4.0," *Business and Information Systems Engineering*, vol. 6, no. 4, pp. 239–242, 2014.

[4] M. Forcan and M. Maksimović, "Cloud-fog-based approach for smart grid monitoring," *Simulation Modelling Practice and Theory*, vol. 101, article 101988, 2020.

[5] K. Wang, K. Yang, and C. S. Magurawalage, "Joint energy minimization and resource allocation in C-RAN with mobile cloud," *IEEE Transactions on Cloud Computing*, vol. 6, no. 3, pp. 760–770, 2016.

[6] E. E. Tsiropoulou, G. Mitsis, and S. Papavassiliou, "Interest-aware energy collection & resource management in machine to machine communications," *Ad Hoc Networks*, vol. 68, pp. 48–57, 2018.

[7] M. Khishe and M. R. Mosavi, "Chimp optimization algorithm," *Expert Systems with Applications*, vol. 149, article 113338, 2020.

[8] J. Aranda, D. Mendez, and H. Carrillo, "Multimodal wireless sensor networks for monitoring applications: a review," *Journal of Circuits, Systems and Computers*, vol. 29, no. 2, article 2030003, 2020.

[9] M. Khishe, M. Nezhadshahbodaghi, M. R. Mosavi, and D. Martín, "A weighted chimp optimization algorithm," *IEEE Access*, vol. 9, pp. 158508–158539, 2021.

[10] J. Wang, M. Khishe, M. Kaveh, and H. Mohammadi, "Binary chimp optimization algorithm (BChOA): a new binary metaheuristic for solving optimization problems," *Cognitive Computation*, vol. 13, no. 5, pp. 1297–1316, 2021.

[11] Y. Yang, Y. Wu, H. Yuan, M. Khishe, and M. Mohammadi, "Nodes clustering and multi-hop routing protocol optimization using hybrid chimp optimization and hunger games search algorithms for sustainable energy efficient underwater wireless sensor networks," *Sustainable Computing: Informatics and Systems*, vol. 35, article 100731, 2022.

[12] A. Saffari, M. Khishe, and S.-H. Zahiri, "Fuzzy-ChOA: an improved chimp optimization algorithm for marine mammal classification using artificial neural network," *Analog Integrated Circuits and Signal Processing*, vol. 111, no. 3, pp. 1–15, 2022.

[13] M. Azhdari, A. Mahmoodzadeh, and M. Khishe, "Digital image watermarking using the combination of genetic algorithm and spread spectrum method in the field of discrete cosine transform," *Iranian Journal of Marine Science and Technology*, vol. 25, no. 1, pp. 14–33, 2021.

[14] A. Mahmoodzadeh, M. Mohammadi, K. M Gharrib Noori et al., "Presenting the best prediction model of water inflow into drill and blast tunnels among several machine learning techniques," *Automation in Construction*, vol. 127, p. 103719, 2021.

[15] A. Mahmoodzadeh, M. Mohammadi, H. H. Ibrahim et al., "Machine learning forecasting models of disc cutters life of tunnel boring machine," *Automation in Construction*, vol. 128, article 103779, 2021.

[16] W. Kaidi, M. Khishe, and M. Mohammadi, "Dynamic Levy flight chimp optimization," *Knowledge-Based Systems*, vol. 235, article 107625, 2022.

[17] A. Farhad, D.-H. Kim, and J.-Y. Pyun, "R-ARM: retransmission-assisted resource management in LoRaWAN for the Internet of Things," *IEEE Internet of Things Journal*, vol. 9, no. 10, pp. 7347–7361, 2021.

[18] P. Khazaei Poul, M. Aghababaee, and M. Khishe, "Improve the quality of images UAVs and small satellites using super-resolution techniques," *Iranian Journal of Marine Science and Technology*, vol. 21, no. 83, pp. 20–29, 2017.

[19] J. H. Holland, "Genetic algorithms," *Scientific American*, vol. 267, no. 1, pp. 66–72, 1992.

[20] T.-C. Chiu, W.-H. Chung, A.-C. Pang, Y.-J. Yu, and P.-H. Yen, "Ultra-low latency service provision in 5G fog-radio access networks," in *In 2016 IEEE 27th Annual International Symposium on Personal, Indoor, and Mobile Radio Communications (PIMRC)*, vol. 21no. 83, pp. 20–29, Valencia, Spain, 2016.

[21] B. Radojević and M. Žagar, "Analysis of issues with load balancing algorithms in hosted (cloud) environments," in *In 2011 Proceedings of the 34th international convention MIPRO*, pp. 416–420, IEEE, Opatija, Croatia, 2011.

[22] S.-P. Gong, M. Khishe, and M. Mohammadi, "Niching chimp optimization for constraint multimodal engineering optimization problems," *Expert Systems with Applications*, vol. 198, article 116887, 2022.

[23] B. Sahoo, D. Kumar, and S. K. Jena, "Performance analysis of greedy load balancing algorithms in heterogeneous distributed computing system," in *In 2014 International Conference on High Performance Computing and Applications (ICHPCA)*, pp. 1–7, Bhubaneswar, India, 2014.

[24] R. G. N. Ngassam, L. Ung, R. Ologeanu-Taddei et al., "An action design research to facilitate the adoption of personal health records," *Journal of Organizational and End User Computing (forthcoming)*, vol. 34, no. 4, pp. 1–18, 2022.

[25] X. Wang, H. Schneider, and K. R. Walsh, "A predictive analytics approach to building a decision support system for improving graduation rates at a four-year college," *Journal of Organizational and End User Computing (JOEUC)*, vol. 32, no. 4, pp. 43–62, 2020.

[26] D. R. Kanniga, M. Gurusamy, and P. Vijayakumar, "An efficient cloud data center allocation to the source of requests," *Journal of Organizational and End User Computing (JOEUC)*, vol. 32, no. 3, pp. 23–36, 2020.

[27] A. Karim, V. Chang, and A. Firdaus, "Android botnets," *Journal of Organizational and End User Computing (JOEUC)*, vol. 32, no. 3, pp. 50–67, 2020.

[28] S. F. Verkijika, "Assessing the role of simplicity in the continuous use of mobile apps," *Journal of Organizational and End User Computing (JOEUC)*, vol. 32, no. 4, pp. 26–42, 2020.