9-1-2022

# HNIO: A Hybrid Nature-Inspired Optimization Algorithm for Energy Minimization in UAV-Assisted Mobile Edge Computing

Yang Chen
*College of Computer Science and Technology, Nanjing University of Aeronautics and Astronautics, Nanjing, China*

Dechang Pi
*College of Computer Science and Technology, Nanjing University of Aeronautics and Astronautics, Nanjing, China*

Shengxiang Yang
*School of Computer Science and Informatics, De Montfort University, Leicester, U.K*

Yue Xu
*College of Computer Science and Technology, Nanjing University of Aeronautics and Astronautics, Nanjing, China*

# HNIO: A Hybrid Nature-Inspired Optimization Algorithm for Energy Minimization in UAV-assisted Mobile Edge Computing

Yang Chen, Dechang Pi, Shengxiang Yang, *Senior Member, IEEE*, Yue Xu, Junfu Chen, Ali Wagdy Mohamed

***Abstract*—Mobile edge computing (MEC) is an emerging computing paradigm that decreases the computing time and extends the lifespan of user equipments (UEs). In MEC, the computational tasks are offloaded from UEs to the base station (BS) at the edge of the network for processing. However, MEC cannot cope with environments where there are no BS or where communication facilities have been destroyed. In this paper, we study the problem of minimizing the energy consumption of UAV equipped with MEC servers as a mobile base station to serve users. The problem involves user offloading decision, UAV location and allocation with computational resources, and is a hybrid optimization problem with continuous and discrete variables. To address this problem, we propose a hybrid nature-inspired optimization algorithm (HNIO) and its version for discrete optimization, where HNIO incorporates mutation and population diversity detection mechanisms to boost its global optimization capability, and we design a probabilistic selection-based coding strategy for the discrete optimization version. The experimental study is conducted based on ten cases with different numbers of UEs. Comparing HNIO with several other state-of-the-art optimization algorithms, it is concluded from the Friedman and Wilcoxons test of the experimental results that HNIO shows better precision and stability in nine out of the ten cases with higher number of UEs.**

***Index Terms*—Mobile edge computing, UAV, Nature-inspired algorithms, Computational task offloading, Discrete optimization, Convergence analysis.**

## I. Introduction

The process of industrial informatization is developing rapidly and people have come to the 5G era. Various types of mobile equipments have gained popularity among people, which helps to gradually form an intelligent society where everything is interconnected. The dazzling variety of services and applications such as online gaming, live video streaming, augmented reality, etc. generate a large amount of data that need to be processed on time. These services are sensitive to time delays and traditional cloud computing methods cannot fully meet their demands. Mobile edge computing (MEC) is a promising computing paradigm that provides users with the required service computing capabilities at the edge of the wireless network. Compared with cloud computing, MEC can effectively reduce the congestion of data transmission and save the energy consumption of UEs to extend their life [1]–[3]. However, MEC has its shortcomings. Zhang *et al.* pointed out that the existing MEC service facilities are not efficient in the lack of wireless network scenarios, such as disaster response or remote environments [4]. In addition, MEC services are limited by fixed location, which is not flexible enough to meet the needs of mobile users.

As a mobile platform with high flexibility, unmanned aerial vehicle (UAV) has been focused on in recent years both in military and life. Xu *et al.* investigated the UAV in wireless communication to provide Hertzian connectivity, considering the molecular absorption effect in the THz-enabled UAV channel gain model [5]. Ho *et al.* studied the energy efficiency of communication between UAVs and ground terminals from a control perspective [6]. Aggarwal *et al.* sufficiently studied the path planning problem of UAV, based on which the coverage and connectivity of UAV network communication are discussed and analyzed [7]. In [8], blockchain technology is used for UAV communication security to ensure data collection and transmission. Moreover, Aggarwal *et al.* presented in detail blockchain-envisioned UAV communication using 6G networks and provide a viable solution. [9]. Straffelini *et al.* combined UAV with motion structure-based photogrammetry to detect potential waterlogging problems on farms [10]. Chhikara *et al.* applied UAV to air quality index prediction. These studies provide a certain foundation for UAV in flight and communication work [11].

Therefore, in order to address the environment with limited MEC infrastructure, this paper employs a UAV equipped with a MEC server as a BS to serve users. The system model of UAV-assisted MEC consists of the local computing model, UAV computational model, and UAV hovering model. From the perspective of minimizing system energy consumption, the optimization variables of the system are UAV deployment location, computational resource allocation, and user offloading decision. The problem is a large-scale hybrid optimization problem, which belongs to the category of non-convex optimization. The hidden enumeration, branch delimitation, and dynamic planning methods are time-consuming.

This work was funded in part by Postgraduate Research & Practice Innovation Program of Jiangsu Province (KYCX21_0226), in part by the Science and Technology Innovation 2030-Key Project of "New Generation Artificial Intelligence" under Grant 2021ZD0113103 and in part by China Scholarship Council Public Graduate Program at High Level Universities (202006830128). (*Corresponding author: Dechang Pi.*)

Yang Chen, Dechang Pi, Yue Xu, Junfu Chen are with the College of Computer Science and Technology, Nanjing University of Aeronautics and Astronautics, Nanjing, China (e-mail: shawn.cy; dc.pi; ayue; cjf@nuaa.edu.cn).

ShengxiangYang is with the School of Computer Science and Informatics, De Montfort University, Leicester LE1 9BH, UK(e-mail: syang@dmu.ac.uk).

Ali Wagdy Mohamed is with the Operations Research Department, Faculty of Graduate Studies for Statistical Research, Cairo University, also with the Department of Mathematics and Actuarial Science School of Sciences Engineering, The American University in Cairo, Egypt (e-mail: aliwagdy@gmail.com).

Nature-inspired algorithms are stochastic algorithms based on population search, which can effectively cope with non-convex optimization [12], [13]. There are two main categories: evolutionary algorithms and swarm intelligence algorithms. Evolutionary algorithms are a general term for a class of algorithms that simulate the evolution of living organisms. Swarm intelligence is a general term for a class of artificial intelligence models that simulates the foraging and migration behaviors of organisms living in swarms in nature, and one of the most representative is the particle swarm algorithm [14], [15].

The problem with nature-inspired algorithms is that there is a loss of diversity in the process of population update leading to premature convergence and failure to obtain a satisfactory solution. To address this problem, we propose a hybrid nature-inspired optimization algorithm (HNIO) that differs from traditional algorithms and incorporates mutation operations and a mechanism for population diversity detection, which can effectively avoid the premature convergence and thus improve the quality of the solution. Since the user offloading decision in the UAV-assisted MEC model is discrete, we propose a discrete version of the HNIO.

### A. Related Work

Research on UAV-assisted MEC has been conducted in recent years. Researchers have discussed it from different perspectives. UAV-assisted MEC will greatly enhance the advantages of MEC. Zeng *et al.*. stated that UAV can provide services to mobile users with wide coverage, reliable line-of-site (LOS) and with additional computing capabilities [16]. MEC service to mobile users is required to offload computational tasks from the UEs to the MEC's server for processing, which mainly involves the offloading computational tasks and computational resources allocation [17]. Tranet *et al.* averred that UEs can transmit more data and offload more computational tasks over the LOS channel when working with a UAV equipped with a MEC server [18]. Meanwhile, since UAV can be deployed flexibly, UEs can save energy when transferring data to the UAV. Nguyen *et al.* discussed the problems of MEC and the challenges of UAV-assisted MEC [19]. Ye *et al.* investigated the flight speed scheduling problem in UAV-assisted MEC systems [20]. Liu *et al.* proposed an IoT collaborative MEC network based on UAV, in which UAV act as a type of MEC server to provide computing services not only for local devices but also for BS with small coverage in the vicinity [21]. Hu *et al.* studied the maximum latency minimization problem between the UAV and the users, aiming to optimize the UAV trajectory, the ratio of offloading tasks, and the user scheduling variables, where energy consumption is the major constraint [22]. Zhang *et al.* studied to improve the average user delay in UAV-assisted MEC networks where UAVs act as computational nodes and relay nodes [23]. Zhou *et al.* studied maximizing UAV data transfer rate in two computation offloading modes: Partial computation offloading mode and Binary computation offloading mode [24]. The starting points of the above studies are different and both mention the problem involving user resource allocation and task offloading which is a non-deterministic polynomial problem. In this paper, we conduct a study with the objective of minimizing the system energy consumption of UAV-assisted MEC and propose a nature-inspired algorithm for continuous-space and discrete-space optimization.

Slowik *et al.* inductively discussed the nature-inspired algorithms and their applications in industry [25], [26]. In the literature [15], Han *et al.* designed a nonlinear regression function to adjust the weight coefficients of PSO based on the fitness of the population, named adaptive particle swarm optimization (APSO), which was applied to a self-organizing radial basis function neural network to improve the accuracy and resolution. With the continuous exploration, the researcher's ideas are not limited to the simulation of biological behavior. Karaboga *et al.* proposed the artificial bee colony algorithm (ABC) by simulating the behavior of a bee colony foraging work [27]. Santana *et al.* proposed an improved novel binary bee colony algorithm (NBABC) for discrete optimization problems, which successfully solved the OneMax problem, Knapsack problem, and Feature Selection [28]. Geem *et al.* proposed harmony search by simulating music performance (HS) [29]. Pan *et al.* introduced the adaptive mechanism to HS and proposed self-adaptive HS (SGHS) for continuous function optimization problems [30]. Salman*et al.* proposed an adaptive probabilistic harmony search (APHS) to deal with combinatorial optimization problems [31]. Rao *et al.* proposed a teaching-learning based optimization (TLBO) by simulating classroom learning [32], and an algorithm called JAYA [33]. Shukla *et al.* proposed an adaptive teaching learning-based optimization (ATLBO) that employs a chaotic initialization strategy as well as an adaptive weighting approach, which demonstrates superior performance over TLBO in continuous function optimization with gene selection problem [34]. Li *et al.* improved JAYA for the flexible job shop scheduling problem with transportation and setup times [35]. Yang *et al.* proposed a flower pollination algorithm (FPA) by simulating the pollination of flowering plants [36]. Chen *et al.* studied the performance of FPA and proposed a cloud mutation-based FPA version (CMFPA) [37]. In addition, two other improved were applied to chaotic system identification and UAV path planning [38] [39]. This kind of algorithms are currently studied to improve the precision and stability of the solution for solving practical application problems.

### B. Contributions and Organization

A UAV equipped with a MEC server is promising to provide services to users in the target area in environments where wireless communication infrastructures are not available or limited. In this paper, we propose a hybrid nature-inspired algorithm with the goal of optimizing the UAV deployment location, with computational resource allocation and user offloading decisions to minimize the overall energy consumption.

The main contributions of this work are summarized as follows:

1) We propose a hybrid nature-inspired algorithm as well as its discrete version. In contrast to traditional algorithms such as PSO, this algorithm combines multiple strategies to

avoid premature convergence, such as employing a solution-based probability to control different update strategies for the population. In addition, individuals are selected for mutation operations based on probabilities. Finally, we detect the diversity of the population and construct new populations for searching again in the objective space when diversity is lost.

2) The convergence of HNIO is theoretically analyzed by constructing a Markov model. We define HNIO's population state space and the state space of individuals. First, we verify whether the population state sequence generated by HNIO is a homogeneous Markov chain, and then verify whether the population's state set can be transferred to the population's optimal set as iterations proceed.

3) The energy minimization problem of UAV-assisted MEC is a large-scale hybrid optimization problem. The proposed HNIO is used for the optimization of UAV location and computational resource allocation, and its discrete version is used to optimize user offloading decisions. Analyzed by experimental results and non-parametric tests, the optimization performance of HNIO is better than the other six state-of-the-art algorithms in nine scenarios.

The remainder of this paper is organized as follows: Section II presents the problem formulation. In Section III, we study the proposed algorithm and analyze the convergence. Experimental results are given in Section IV. Section V concludes this article.

## II. System Model and Problem Formulation

In this section, we present the UAV-assisted MEC system model for UEs and the minimizing energy consumption for the system. Fig.1 presents a brief diagram to explain the system, where UE1, UE2 and UE4 offload the tasks to the UAV equipped with the MEC server, while UE3 and UE5 perform local computation without offloading the task.The solid and dashed lines in the diagram indicate the upload and download links respectively when the user computes tasks that need to be offloaded.

Considering a total of $N$ UEs in a region, the set of UEs is denoted as $\mathcal{N} = \{1, 2, \cdots, N\}$. Although some of these UEs can perform smaller computational tasks independently, the UAV platform must be involved if more computational tasks arise simultaneously. Therefore, the UAV assisted MEC system model, can be specifically divided into the local computational model and UAV computational model. The UAV computational model should include the energy consumption on the hovering of the UAV. We summarize the key symbols of the system model in Table I.

### A. Local computational model

The local computational model is that the UEs use their own allocated computational resources to complete the computational task, and the time consumed is calculated as follows [2], [3]:

$$T_n^l = C_n(f_n)^{-1}, \forall n \in \mathcal{N} \tag{1}$$

where $C_n$ represents the total number of CPU cycles to complete the $n$-th UE task and $f_n$ denotes the computing resources of UE $n$.

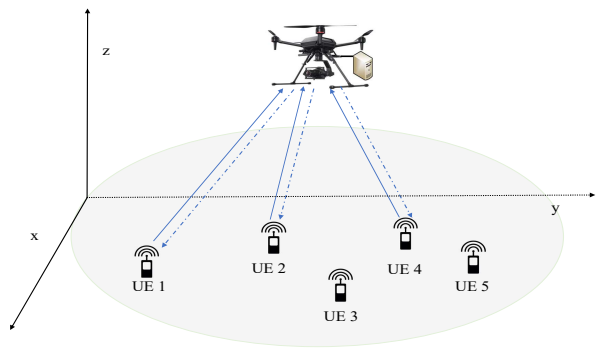



Fig. 1. UAV-assisted MEC system. The diagram contains five UEs, where UE1, UE2 and UE4 need to offload computing tasks to the UAV for processing, and UE3 and UE5 perform local computation. The solid and dashed lines indicate the communication channels for uploading and downloading.

TABLE I
The Symbols of System Model

| Symbols | Definitions |
|---|---|
| $N$ | The number of all UEs |
| $\mathcal{N}$ | Set of $N$ UEs |
| $B$ | System Bandwidth (Hz) |
| $D_n$ | The size of input data of UE $n$ |
| $C_n$ | The number of CPU cycles required for computing UE $n$ |
| $\eta_1$, $\eta_2$ | The effective switched capacitance of different chip architecture |
| $f_n$ | Computing resources allocated to UE $n$ |
| $p_n$ | The transfer power of UE $n$ |
| $h_0$ | The channel power gain at the reference distance |
| $\beta$ | The noise spectral density |
| $f_{\max}$ | The maximum allowable computing resources of UEs |
| $h_0$ | Channel power gain at the reference distance |
| $P_h$ | UAV hovering power |
| $T_h$ | UAV hovering time |
| $T_n^l$ | Time consumption for UE $n$ on local computing |
| $E_n^l$ | Energy consumption for UE $n$ on local computing |
| $L_n$ | The distance from UE $n$ to UAV |
| $R_n$ | The uplink data rate of UE $n$ |
| $T_n^u$ | Time consumption for UE $n$ on UAV |
| $E_n^u$ | Energy consumption for UE $n$ on UAV |
| $E^h$ | The energy consumption of UAV hovering |

Consequently, the energy consumption of the local model processing task is calculated as follows

$$E_n^l = \eta_1 C_n (f_n)^2, \forall n \in \mathcal{N} \tag{2}$$

where $\eta_1$ denotes the effective conversion capacity that depends on the chip architecture [2].

### B. UAV computational Model

The UAV provides computing services to UEs within the coverage area that need to take into account the location of the UAV deployment.

The distance from UE $n$ to the UAV is $L_n$. $\{x_n, y_n, z_n\}$ is the location of $n$-th UE and $\{x_u, y_u, z_u\}$ is the location of UAV.

$$L_n = \sqrt{(x_n - x_u)^2 + (y_n - y_u)^2 + (z_n - z_u)^2}, \forall n \in \mathcal{N} \tag{3}$$

For UE $n$ , the uplink data rate $R_n$ is calculated as (4) [3].

$$R_n = B\log_2(1 + \frac{p_n \cdot h_0}{B \cdot \beta \cdot L_n^2}) \tag{4}$$

where $B$ is system bandwidth, $p_n$ is transfer power from the UEs offload task to the UAV, $\beta$ is the noise spectral density, $h_0$ is the channel power gain at the reference distance.

The time spent on processing UEs under the UAV computational model is the sum of the task computation time and transmission time, which is denoted as $T_n^u$ and is calculated as follows:

$$T_n^u = C_n(f_n)^{-1} + D_n(R_n)^{-1} \tag{5}$$

Therefore, the energy consumption of UAV to handle UEs tasks is expressed as:

$$E_n^u = \eta_2 C_n(f_n)^2 + p_n D_n(R_n)^{-1} \tag{6}$$

where $\eta_2$ is effective switched capacitance depending on the chip architecture, $p_n$ is the transfer power of UE $n$.

The energy consumption of the UAV while hovering is denoted as $E^h$ , which is calculated as follows:

$$E^h = P_h \cdot T_h \tag{7}$$

where $P_h$ denotes the UAV hovering power, $T_h$ is the hovering time.

### C. Description of the constraints

Each UE can only be performed in one mode, so we set the decision variable $k = \{ 0, 1 \}$ . When $a_{n,k} = 1$, $k = 1$ denotes that the UE $n$ offloads the computing task to be processed on the UAV. $k = 0$ indicates that the UE $n$ processes the task using the local computational model [17].

$$a_{n,k} = \begin{cases} 1, & \text{UEs task is processed} \\ 0, & \text{otherwise} \end{cases} \tag{8}$$

We require all UEs to be processed in the objective area, which ensures that every task is counted.

$$\sum\nolimits_{n=0}^{N} a_{n,k} = N, \forall n \in \mathcal{N}, k \in \{0,1\} \tag{9}$$

The computational consumption of the UEs is finite, so excessive tasks computation must be handled on offload to the UAV, i.e., each UEs has to meet the following constraints:

$$\sum\nolimits_{n=1}^{N} a_{n,k} f_n \leq f_{\max}, \forall k = 0 \tag{10}$$

Regardless of whether the task is handled using a local computational model and or a UAV computational model, the completion time must be within the hover time of the UAV. That is, the constraints to be satisfied are as follows:

$$T_n^l \leq T_h, T_n^u \leq T_h, \forall n \in \mathcal{N} \tag{11}$$

Thus, the overall energy consumption of the UAV-assisted MEC as an objective function for the fixed hover time of the UAV is as follows:

$$\mathcal{F} = \min \Big( \sum\nolimits_{n=1}^{N} \left( a_{n,k=0}(E_n^l) + a_{n,k=1}(E_n^u) \right) + E_h \Big) \quad \text{, s.t.(8),(9),(10),(11)} \tag{12}$$

## III. Proposed algorithm

### A. Motivation

The energy minimization problem of the UAV-assisted MEC system is a large-scale non-convex optimization problem involving continuous space and discrete space. The application of nature-inspired algorithms to tackle this problem does not require knowledge of the specific gradient information of its objective function. It can be solved directly by the designed objective function. However, the nature-inspired algorithm suffers from the premature convergence problem, which affects the precision of the final solution of the algorithm. Literature [25] points out that traditional nature-inspired algorithms often give a satisfactory sub-optimal solution in engineering applications, but how to design methods to enhance the global optimization capability of the algorithm is crucial in engineering optimization. In order to improve the quality of the solution for solving the objective problem, we need to propose an algorithm with stronger optimization capability applicable to search in both continuous and discrete spaces.

Nature-inspired algorithms can be divided into two categories. In algorithms with evolution as the main idea, population individuals update new solutions by crossover and mutation. In contrast, swarm intelligence algorithms tend to guide the population evolution using the best individuals of the current population. These types of algorithms all have different mathematical models designed for population updating and their main problem is premature convergence with iterations. The FPA proposed in the [36] combines the ideas of two types of algorithms to guide the population and is controlled by a stochastic parameter with two different update strategies. One of the update strategies is to converge in the population toward the current optimal individual, and the other is to update all individuals of the population by a difference vector. The parameter controlling the switch between these two strategies is dependent on human empirical, which severely affects the performance of the algorithm [38].

An affective population diversity evaluation helps to improve the search efficiency and can help to determine whether the algorithm is premature convergence the iterative process. The solutions of individuals in the population are similar because they are all obtained after evaluating the same objective function. We consider the application of clustering algorithms to monitor changes in population diversity. The change in the population's clustering center as well as the global best solution is determined to detect whether the population falls into a local optimum. The population can be regenerated when they lose diversity but do not reach the maximum iteration number.

Therefore, we propose a hybrid nature-inspired algorithm that is designed to control different update strategies of the population based on the probability of individual solutions. To further overcome premature convergence, a mutation operation is employed as a mechanism to determine population diversity.

### B. Algorithm design and implementation

FPA has two update strategies , where global exploration is performed as shown in Eq.13 and the other is local exploitation

as in Eq.14. Essentially, the global exploration is the updating process of individuals all with the participation of the best individuals in the population, defined as follows:

$$X_i = X_i + l(g^* - X_i) \tag{13}$$

where $X_i$ denotes an individual in the population, $g^*$ is the current best individual and $l$ denotes random numbers obeying the Lévy distribution [40].

Another update strategy is basic differential evolution equation [36], defined as follows:

$$X_i = X_i + r(X_j - X_k) \tag{14}$$

where $r$ is a random number between 0 and 1, $X_j$ and $X_k$ are two individuals different from $X_i$.

We employ the solution-based probabilistic selection method in [27] to guide the switch of the two update strategies. The total number of individuals in the population is $NP$. The probability of selecting the $i$-th individual is calculated as follows:

$$P_{fit_i} = \frac{Fit_i}{\sum_{i=1}^{NP} Fit_i} \tag{15}$$

where $Fit_i$ denotes the fitness of the $i$-th individual.

For the minimization problem, the lesser the solution an individual obtains, the better that individual is. Some of the worse individuals should get information from the dominant individuals in the population to update their positions. Therefore, for the $i$-th individual, we execute Eq.(13) when a random number is less than $P_{fit_i}$, otherwise we execute Eq.(14). This improves the inconvenience of setting parameters manually.

The subjective form of the loss of population diversity is expressed by the fact that all individual positions stagnate at the same position at some point in the iteration and do not change. All individuals are unable to obtain further better solutions. For this, we aim to evaluate population diversity using a clustering algorithm. The clustering algorithm is used to divide the population into three classes based on their solutions, and the best individual in each class is defined as the class center. Therefore, the population is considered to lose its diversity once the solutions of the three class centers are identical during the iterative process, at which a new population is generated using the generalized opposition-based learning. Generalized opposition-based learning implementation is as follows:

$$\tilde{X}_d = r \cdot (Ub_d + Lb_d) - X_d \tag{16}$$

where $\tilde{X}_d$ is new individual that is generated by $X_d$ in $d$-dimensions, $r$ is a random number between 0 and 1, $Ub_d$ and $Lb_d$ denote the upper and lower bounds in dimension $d$ respectively.

In addition to detecting the diversity of populations, we employed a mutation operation on individuals based on the probability of the solution. The strategy of randomly assigning positions to individuals in the search space has been used in the literature [39] and [41], but that strategy was unfocused. Gaussian mutation [42] is an effective way to improve the quality of individuals. We perform gaussian mutation on individuals with random probability less than $P_{fit_i}$ to reassign position on the basis of the current population optimal position. For individual $X_i$, the mutation method is calculated as follows:

$$X = g^* \, (1+normrnd(0,1)) \tag{17}$$

where $g^*$ denotes the best individual, $normrnd(0,1)$ indicates the generation of random numbers with a mean of 0 and a variance of 1.

Therefore, we present the pseudo-code of HNIO as Algorithm 1: Algorithm HNIO. The input parameters are the population size: $NP$ and the maximum iteration number: $MaxIter$. Line 1 is the population initialization. Line 2 is the evaluation of the solution for each individual in the population. Lines 4 to 16 are the process of population update, where lines 13 to 15 are the process of individual mutation. Line 17 is the clustering of the population according to the solution. Line 18 is the recording of the change in the best solution. Lines 19 to 21 are the generation of new populations.

**Algorithm 1 :** Algorithm HNIO

**Input** : Population size ($NP$), Maximum iteration number ($MaxIter$).

1: Population initialization.
2: **while** $(Iter < MaxIter)$ **do**
3: Evaluating the solution of the population according to the objective function.
4: **for** i=1:*NP* **do**
5: Calculating the selection probability $P_{fit_i}$.
6: Record the solution for each individual.
7: Preserving the best optimal position $g^*$.
8: **if** $rand < P_{fit_i}$ **then**
9: $X_i = X_i + r(X_j - X_k)$
10: **else**
11: $X_i = X_i + L(g^* - X_i)$
12: **end if**
13: **if** $rand < P_{fit_i}$ **then**
14: $X_i = g^* \, (1+normrnd(0,1))$
15: **end if**
16: **end for**
17: The population is clustered into three classes according to the individual solutions, and the solutions at the center of these three classes were recorded.
18: Recording the number of changes in the best solution starting from the second function evaluation.
19: **if** The clustering centers are the same and the number of times the best solution has not changed consecutively is half of the number of times the maximum function is evaluated **then**
20: Generating new populations by Eq.(16).
21: **end if**
22: **end while**

**Output** : $g^*$

### C. Time complexity analysis

Time complexity is a measure of the efficiency of an algorithm and evaluates how efficiently the algorithm performs in the worst case. We analyze the time complexity of HNIO in

this subsection. In HNIO, the time complexity of population initialization is $\mathcal{O}(NP)$. The frequency of execution of each step is 1 in the population updating process of the HNIO. Therefore, the time complexity from the 4-th line to the 16-th line is $\mathcal{O}(NP)$. The time complexity of clustering is $\mathcal{O}(NP)$ in line 17 as well as regenerating $NP$ individuals in line 20. Therefore, the time complexity of line 4 to 21 is expressed as $\mathcal{O}(NP)$. We record the time complexity of the function evaluation as $\mathcal{O}(\mathcal{F})$. The time complexity of the evaluation of $NP$ individuals in line 3 is $\mathcal{O}(\mathcal{F} \cdot NP)$. Therefore, the time complexity of HNIO within the while loop is $\mathcal{O}(NP)$+$\mathcal{O}(\mathcal{F} \cdot NP \cdot MaxIter)$=$\mathcal{O}(\mathcal{F} \cdot NP \cdot MaxIter)$. We can observe that the time spent on HNIO is related to the parameters $NP$ and $MaxIter$ in addition to the objective function.

### D. Convergence analysis

*Definition* 1: The state of individuals in a population and the state space.

The state of all individuals in HNIO is constituted by the position of the individual in the objective space. Marked as $x$, $x \notin \mathcal{A}$, $\mathcal{A}$ denotes the feasible space. The set of all possible states of an individual constitutes the state space of that individual, marked as $\mathcal{X}$=$\{x|x \in \mathcal{A}\}$

*Definition* 2: The state of the population and the state space of the population.

The state of the population is composed of the states of all the individuals and is labeled $s = (x_1, x_2, \cdots, x_n)$. $x_i$ indicates the state of the $i$-th individual in HNIO. All possible states of the population constitute the state space of the population, which is labeled $S$=$\{s = (x_1, x_2, \cdots x_n)|x_i \in \mathcal{X}\}$.

*Definition* 3: State transfer for an individual.

For $\forall x_i \in \mathcal{X}$ , $\forall x_j \in \mathcal{X}$. An individual is transferred from $x_i$ to $x_j$ in one step, labeled as $\mathcal{T}_x x_i = x_j$.

**Theorem** 1: The transfer probability $P(\mathcal{T}_x x_1 = x_2)$ of an individual state from $x_1$ to $x_2$ in one-step is determined by $P_a$, $P_b$, $P_c$ in HNIO.

*Proof.* Population can be considered as a set of point sets in a hyperspace, and population evolution can be considered as a transformation between point sets in that hyperspace. HNIO employs the Eq.(13), Eq.(14) and (Eq.17) to update the population, and their transfer probabilities are obtained as Eq.(18) to Eq.(20), respectively.

$$P_a(\mathcal{T}_x x_1{=}x_2) = 1/|g^* - x_1| \cdot p_d(x_1 \to x_2), \quad x_2 \in [x_1, x_1 + g^* - x_1] \tag{18}$$

$$P_b(\mathcal{T}_x x_1{=}x_2) = 1/(|x_3 - x_4|) \cdot p_d(x_1 \to x_2), \quad x_2 \in [x_1, (x_3 - x_4)] \tag{19}$$

$$P_c(\mathcal{T}_x x_1 = \mathrm{x}_2) = 1/(g^*(1{+}normrnd(0,1))) \ , \quad x_2 \in [Lb, Ub] \tag{20}$$

$$p_d\left(x_1 \to x_2\right) = \begin{cases} 1 & F\text{itness}\left(x_1\right) < F\text{itness}\left(x_2\right) \\ 0 & F\text{itness}\left(x_1\right) \geq F\text{itness}\left(x_2\right) \end{cases} \tag{21}$$

Eq.(18), Eq.(19) and Eq.(20) are the probability of transferring the individual state from $x_1$ to $x_2$ when updating the population using Eq.(13), Eq.(14) and Eq.(17), respectively. HNIO adopts a greedy retention strategy, as shown in Eq.(21), where $P_d$ affects the calculation of transfer probability.

*Definition* 4: State transfer for a population.

The probability of a population state $i$ to $j$ is the probability of every individual moving from state $i$ to $j$. For $\forall \mathrm{s}_i = (x_{i1}, x_{i2}, \cdots x_{in}) \in S$ and $\forall s_j = (x_{j1}, x_{j2}, \cdots x_{jn}) \in S$, the probability of a one-step population state transfer from $s_i$ to $s_j$ is shown in Eq.(22).

$$P(\mathcal{T}_s s_i{=}s_j) = P(\mathcal{T}_x x_{i1}{=}x_{j1}) P(\mathcal{T}_x x_{i2}{=}x_{j2}) \cdots P(\mathcal{T}_x x_{in}{=}x_{jn}) \tag{22}$$

**Theorem**2: The population state sequence $\{s(t) : t >= 0\}$ for the HNIO is the homogeneous Markov chain.

*Proof.* According to Definition 4, $\forall s(t-1), s(t) \in S$, the $P(\mathcal{T}_s(s(t-1)) = s(t))$ depends on $P(\mathcal{T}_{\mathrm{x}} x(t-1) = x(t))$ that is the probability of state transfer from $t-1$ to $t$ for each individual. Form Theorem 1, the probability of an individual state transfer in HNIO is jointly determined by $P_a$, $P_b$, $P_c$. The state of all individuals at the $t$ moment is determined by their $t-1$ moment state. The state of all individuals determines the state of the population. Therefore, the population state of HNIO is only related to the state of the $t-1$ moment. So the population's state sequence $\{s(t) : t >= 0\}$ has Markov property.

Furthermore, HNIO retains the solution using a greedy strategy. If the solution does not improve at moment $t$ then the individual state is potentially identical at moments $t-1$ and $t-2$. Therefore $P(\mathcal{T}_x x(t-1) = x(t))$ is not correlated with moment $t-1$, but only with the state at moment $t-1$. So the population state sequence $\{s(t) : t >= 0\}$ is a homogeneous Markov chain.

Solis *et al.* give the convergence criterion for stochastic algorithms, which is that the algorithm needs to satisfy the following two conditions [43]. $\langle \Omega, f \rangle$ is an objective problem, where $\Omega$ denotes the search space and $f$ is the objective function. For an algorithm $A$, $x_t$ is the solution at the $t$ iteration, the solution of next iteration is $x_{t+1} = A(x_t, \zeta)$ where $\zeta$ denotes the candidate solutions searched by $S$ before the $t$-th iteration.

*Condition*1: If $f(A(x,\zeta)) \leq f(x)$ && $\zeta \in \Omega$, $f(A(x,\zeta)) \leq f(\zeta)$.

*Condition*2: If $\forall \mathcal{B} \in \Omega$,$v(\mathcal{B}) > 0$, then $\prod_{t=0}^{\infty} (1 - \mu_t(\mathcal{B})) = 0$.

$v(\mathcal{B})$ is the Lebesguece measure on set $\mathcal{B}$. $\mu_t(\mathcal{B})$ represents the probability measure of the solution obtained in the t-th generation on $\mathcal{B}$.

**Theorem** 3: $f$ is measurable. Algorithm $A$ satisfies *Condition* 1 and 2. $\{x_t\}_{t=0}^{\infty}$ represents the sequence of solutions produced by iterations from $t$ to $\infty$.

If $\lim_{t \to \infty} P(x_t \in \mathcal{R}_\varepsilon)$=1, then $A$ is convergence. $\mathcal{R}_\varepsilon$ is the set of global optimal position [43].

*Definition* 5: Population optimal state set $M$. Assuming that $g^*$ is the optimal position. Therefore, the optimal set of

states for the population is $M$=$\{s=(x_1,x_2\cdots,x_n)|f(x_k)=f(g^*),k\in 1\cdots n,s\in S\}$.

**Theorem** 4: State sequence of the population is $\{s(t):t>=0\}$. The optimal state set $M$ of the population is a closed set on the state space $S$.

*Proof.* Defining $\mathcal{L}$ is step. $\forall s_i\in M,\forall s_j\notin M$, the transition probability from state $s_i$ via $\mathcal{L}$+1 step to state $s_j$ is Eq.(23).

$$P_{s_i,s_j}^{\mathcal{L}+1}=\sum_{s_{r_1}}\sum_{s_{r_2}}\cdots\sum_{s_r}P(\mathcal{T}_s(s_i)=s_{r_1})P(\mathcal{T}_s(s_{r_1})=s_{r_2})\cdots P(\mathcal{T}_s(s_l)=s_j) \tag{23}$$

For $P(\mathcal{T}_s(s_{r_{c-1}})=s_{r_c}),\exists s_{r_{c-1}}\in M s_{r_c}\notin M, c\in 1,2\cdots\mathcal{L}$. We can obtain Eq.24.

$$P(\mathcal{T}_s(s_{r_{c-1}})=s_{r_c})\text{=}P(\mathcal{T}_x x_{r_{c-1}1}\text{=}x_{r_c1})P(\mathcal{T}_x x_{r_{c-1}2}\text{=}x_{r_c2})\cdots P(\mathcal{T}_x x_{r_{c-1}n}\text{=}x_{r_cn}) \tag{24}$$

$s_{r_{c-1}}\in M$, $s_{r_c}\notin M$, $f(x_{r_{c-1},k})<f(x_{r_c,k})$, $k\in 1,2,\cdots n$, so $f(x_{r_{c-1}})=f(g^*)=\inf(f(\tau)),\tau\in\Omega$.
According to $P(\mathcal{T}_x(x_{r_{c-1},k})=x_{r_c,k})$=0, $P_{s_i,s_j}^{\mathcal{L}+1}=0$ and the Theorem 1, it can be shown that $M$ is a closed set on $S$.

**Theorem** 5: There is no non-empty closed set $O$ in the state space $S$ of the population satisfying $O\bigcap M=\varnothing$

*Proof.* Suppose there is a non-empty closed set $O\in S$, $O\bigcap M=\varnothing$, then $s_i$=$(x_{i1},x_{i2}\cdots x_{in})$is a state on the $O$. $s_j$=$(g^*,g^*\cdots g^*)$is a state on $M$ for the entire population. Hence from Eq.23, Eq.23 and Teheorem1 we know that: Because of $f(x_{r_{c-1},k})<f(x_{r_c,k}),k\in 1,2,\cdots n$, so we have $P(\mathcal{T}_x(x_{r_{c-1}k})=x_{r_ck})>0$ and $P_{s_i,s_j}^{\mathcal{L}+1}>0$. This demonstrates that a state on $O$ can be transferred to a state of $M$ by the $\mathcal{L}$+1 step. Therefore, $O$ is not a closed set under $S$, which is an incorrect assumption. That is, there is no non-empty closed set $O$ satisfying $O\bigcap M=\varnothing$ in $S$.

**Theorem** 6: The literature [41] provides a proof that if there exists non-empty closed set $Q$ in a Markov chain and no other non-empty closed set $W$ that makes $Q\bigcap W=\varnothing$. Then it follows that $i\in Q$ , $\lim_{t\to\infty}P(x_t=i)=\pi_i$, $i\notin Q$, $\lim_{t\to\infty}P(x_n=i)=0$.

**Theorem** 7: HNIO is a global convergence algorithm

*Proof.* HNIO is run with a greedy strategy for the best solution retention method, regardless of which update equation is utilized. Each individual in the population, as well as the optimal position of the population, is preserved. *Condition* 1 is satisfied.

According to Theorem 4, Theorem 5 and Theorem 6, we can conclude that the population's state sequence $\{s(t):t>=0\}$ will necessarily enter the population's optimal set $M$ when the population iteration of HNIO tends to infinity.

Therefore, the probability measure that HNIO cannot obtain an optimal solution is 0 when the iteration satisfies $t\to\infty$. This satisfies *Condition* 2. Consequently, it follows from Theorem 3 that HNIO is a global convergence algorithm.

### *E. Discrete HNIO based on probabilistic coding*

The energy consumption optimization problem for UAV-assisted MEC is a joint continuous and discrete optimization problem. HNIO can be applied to UAV location selection and computational resource allocation, but cannot directly deal with offloading decision optimization. In this subsection, we give the version of the framework based on HNIO to handle the discrete problem. Since all individuals have only 0 or 1 decision variables, the two different update strategies of HNIO both guide individual updates according to the difference of different individuals in the population. To effectively fit the 0-1 variable problem, we use the designed elite probability selection scheme to guide individual mutations. The elite probability selection scheme is to calculate the percentage of 0-1 variables for the top ranked individuals among them after sorting the $NP$ individuals in the increasing order of the solution [34]. As shown in Fig.2, we select three dominant individuals to determine the selection probability. Since our goal is energy consumption minimization, $X_1$ is necessarily superior to $X_{NP}$. $P_f$ is defined as the selection probability, and the decision variable for experimental individual $X_i$ is easily set to 1 for larger values of $P_f$.

The pseudo-code for the 0-1 discrete problem of HNIO is shown in Algorithm 2: Discrete HNIO. The discrete HNIO differs from HNIO in the adjustment of the main update method. Line 9-Line 15 are the main update strategies of the discrete version of the algorithm. For individual $X_i$, we judge whether different individuals $X_j$ and $g^*$ are consistent with $X_i$ variables respectively, and change the variables if they are not. Line 17 to 21 are mutation operation for individuals according to the probability $P_f$.

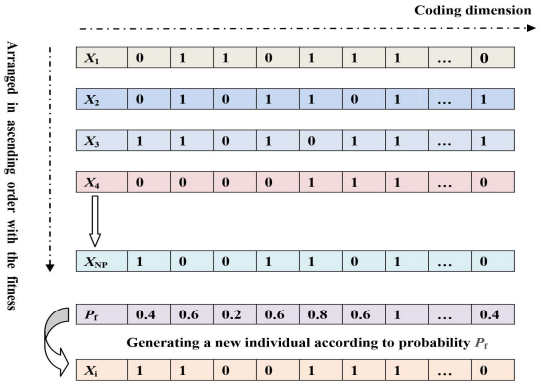



Fig. 2. Elite probability selection. The figure briefly describes the calculation of the probability $P_f$. $X_1$ represents the individual with the best solution and $X_{NP}$ is the individual with the worst solution. According to the dimensional accumulation of the five individuals in the figure, the individuals with a decision variable of 1 are divided by the total number of individuals to obtain $P_f$.

## IV. Experimental Study

### *A. Experimental settings*

In this section, we perform simulation experiments. The software for the experiment is Matlab R2020a, and the hardware is a PC with CPU: i5-11300H and Memory: 16GB. The coverage area of UEs is within the range of 1000m*1000m. Ten cases with different numbers of UEs are tested. The number of UEs is from 100 to 1000. The parameters of the

**Algorithm 2 :** Discrete HNIO

**Input** : Population size ($NP$ ), Maximum iteration number ($MaxIter$).

Population initialization.
**while** $(Iter < MaxIter)$ **do**
  Evaluating the solution of the population according to the objective function.
  **for** i=1:*NP* **do**
    Calculating the selection probability $P_{fit_i}$.
    Record the solution for each individual.
    Preserving the best optimal position $g^*$.
    **if** $rand < P_{fit_i}$ **then**
      **if** $|X_i - X_j|$=1 **then**
        $X_i = 1 - X_i$
      **end if**
    **else**
      **if** $|g^* - X_j|$=1 **then**
        $X_i = 1 - X_i$
      **end if**
    **end if**
    **if** $rand < P_f$ **then**
      $X_i = 1$
    **else**
      $X_i$=0
    **end if**
  **end for**
  The population is clustered into three classes according to the individual solutions, and the solutions at the center of these three classes were recorded.
  Recording the number of changes in the best solution starting from the second function evaluation.
  **if** The clustering centers are the same and the number of times the best solution has not changed consecutively is half of the number of times the maximum function is evaluated **then**
    Generating new populations by Eq.(16).
  **end if**
**end while**

**Output** : $g^*$

UAV-assisted MEC system are set as follows: $B$=10MHz, $\beta$=-174dBm/Hz, $p_n$=0.5W, $\eta_1$, $\eta_2$=$10^{-27}$, $h_0$=$10^{-4}$, $p_h = 10W$, $f_{\max}$=$2 \times 10^9$ Cycles/s.

The comparison algorithms we used to compare the performance of HNIO are: SGHS [30], APHS [31], CMFPA [37], APSO [15], NBABC [28], ATLBO [34]. The descriptions of the relevant parameter settings for the comparison algorithms are shown in Table II. In order to compare the algorithm optimization results fairly, we set the same parameters for all algorithms as: the maximum iterations number $MaxIter$ = 500, the population size $NP$=40. Since the population initialization of all algorithms has a random property, which will have a certain impact on the final results. Therefore, we have run each algorithm 31 times independently in order to avoid this randomness of one experiment. The experimental results are tested and ranked by Friedman test [44]. In addition, the optimization results of all algorithms are performed using Wilcoxon signed-rank tests [45] and comparing their average error that is calculated as the difference between the average solution and the smallest solution of these algorithms.

TABLE II
PARAMETER SETTING OF THE COMPARISON ALGORITHM

| Algorithms | Parameter Description |
|---|---|
| SGHS | Transfer probability is 0.2. The scaling of the ambiguity and uncertainty measures are 2 and 0.1, respectively. |
| APHS | Harmony memory considering rate is 0.9. Pitch adjusting rate is 0.3. The sample size is 5 for adjusting the probability distribution. The probabilistic increment is 0.2. |
| CMFPA | Transfer probability is 0.2. The scaling of the ambiguity and uncertainty measures are 2 and 0.1, respectively. |
| APSO | The acceleration constants of APSO are1.49. The parameters in the nonlinear regression function are set as follows: 2.1. The parameter used to improve the global search capability of the particles is set to 2. |
| NBABC | Trials limit is 50. Flips limit is 0.1. |
| ATLBO | Bifurcation coefficient is 4. The maximum weight is 0.9 and the minimum weight is 0.4. |

### *B. Parameter calibration for HNIO*

In performing the energy optimization of UAV-assisted MEC systems, the proposed HNIO is affected by the maximum iterations number and the population size, both of which need to be set manually. The upper limit of the maximum iterations number selected generally depends on the convergence of the optimal solution obtained by the algorithm, which is set to 500 in accordance with experiments.

Since the population size $NP$ affects the time complexity of the algorithm, we adjust its value by experimental simulation. Fig.3 gives the box plot of the optimization results for different $NP$, which are tested in a scenario where the population size of HNIO ranges from 10-100 at UEs of 1000. We can observe that the distribution of the final optimization results tends to be smooth after $NP$=40 under the condition of increasing $NP$ in sequence. Considering that the increase of $NP$ affects the running time of the algorithm, $NP$ is set to 40 in this paper.

### *C. Impact of different strategies on HNIO*

The HNIO we designed for optimizing the energy consumption problem of UAV-assisted MEC has two strategies to overcome the premature convergence of the algorithm. This subsection tests the performance impact of the different strategies on the algorithm. We name the algorithm when it does not have population diversity detection and mutation strategies as HNIO-s0, the algorithm with only mutation strategies as HNIO-s1, and the algorithm with only population diversity detection as HNIO-s2; the complete algorithm with both strategies as HNIO. The box plots of the experimental results of the algorithms under different strategies in the scenario with the number of UEs of 1000 are given in Fig. 4. We can obtain that the significance difference between the results of HNIO-s2 and HNIO-s0 is not significant, and there is a slight performance improvement, which indicates that generating new

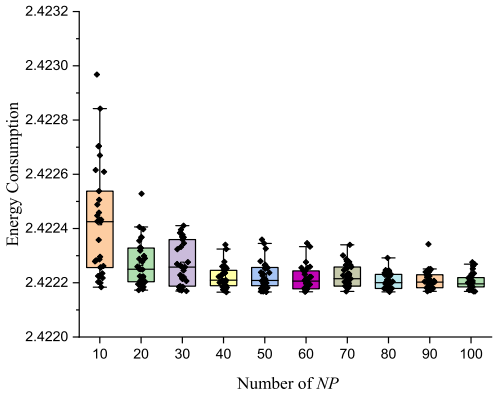



Fig. 3. Box plot for different number of $NP$ at UEs of 1000. The figure shows the distribution of optimization results under multiple independent runs of HNIO for population sizes from 10-100.

populations using single population diversity detection has no significant impact on the algorithm performance. However, HNIO-s1 has a significant difference compared to HNIO-s0, indicating that the mutation strategy can substantially improve the performance of the algorithm. HNIO is increased population diversity detection compared to HNIO-s1, which further improves the search performance of the algorithm. Therefore, the mutation strategy can effectively increase the population diversity to improve the optimization precision of the algorithm, and on the basis of having the mutation strategy in addition to the population diversity detection can further increase the optimization capability of the algorithm.

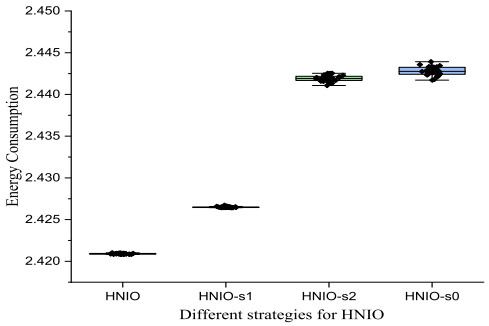



Fig. 4. Box plot of different strategies for HNIO at UEs of 1000. In the figure, HNIO, HNIO-s0, HNIO-s1 and HNIO-s2, indicate that the algorithm has mutation and population diversity detection strategies, does not have both strategies, has only mutation strategy, and has only population diversity detection strategy, respectively.

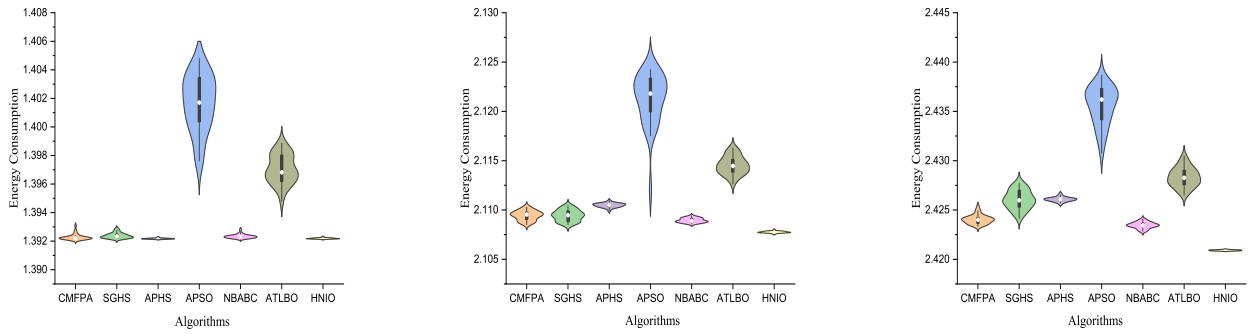
### *D. Results and Discussion*

We perform experiments to verify the energy minimization results of the UAV-assisted MEC system with HNIO and the other six comparison algorithms for the number of UEs ranging from 100 to 1000. Due to the large number of all experimental images, we only show here the images under the number of UEs $N$ of 100, 500, and 1000, where Fig.5 shows the violin plot of the optimization results of HNIO and the comparison algorithms after 31 independent runs, and Fig.6 gives the convergence curves of the seven algorithms under the maximum iterations number. To show the figure more clearly, we perform all Log10 transformations on all data in the figure. We can observe from Fig.5 that the distribution of optimization results of HNIO in the scenario of $N$=100 is obviously the most concentrated with APHS, better than APSO and ATLBO, and slightly more concentrated than CMFPA, SGHS and NBABC. In the $N$=500 scenario, the distribution of optimization results of HNIO does not overlap with the other comparison algorithms, indicating that the optimization results obtained in each of the multiple independent runs are better than those of the other comparison algorithms. HNIO still maintains a significant advantage in the scenario with $N$=1000, and its stable optimization performance can be derived from the data density. As the number of UEs increases and the optimization dimension of the problem becomes larger, the HNIO displays better advantages. The best convergence curves of these algorithms are shown in Fig.6, from which we can observe that HNIO has the fastest convergence speed in the $N$=100 scenario, but the final convergence accuracy is not obvious in comparison with SGHS, NBABC, CMFPA and APHS. In the other two scenarios, we can observe from 6(b) and 6(c) that HNIO has the fastest convergence speed spending fewer iterations to get better solutions compared to the other six comparison algorithms.

TABLE III
RANKING OF HNIO WITH COMPARISON ALGORITHMS IN DIFFERENT UES EXPERIMENTS BY FRIEDMAN TEST.

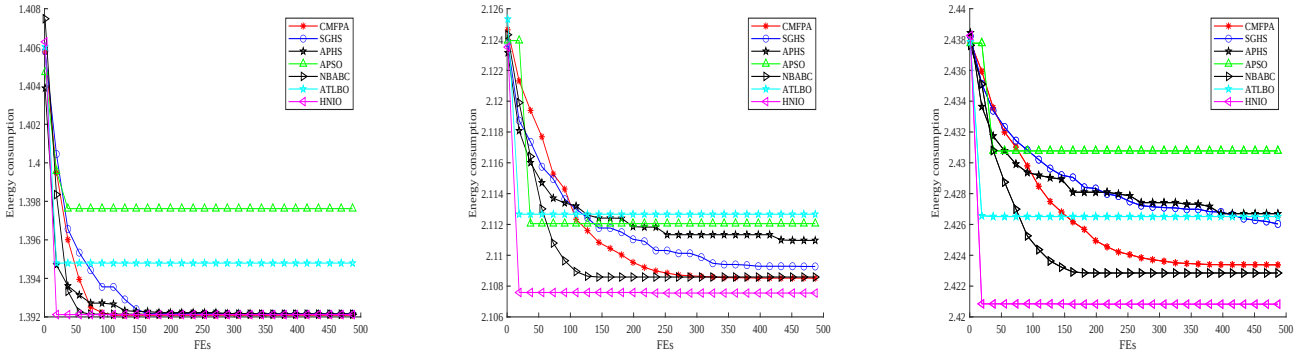
| $N$ | CMFPA | SGHS | APHS | APSO | NBABC | ATLBO | HNIO |
|---|---|---|---|---|---|---|---|
| 100 | 2.94 | 4.10 | **1.90** | 6.90 | 3.74 | 6.10 | 2.32 |
| 200 | 3.76 | 2.02 | 4.16 | 6.97 | 3.69 | 6.03 | **1.37** |
| 300 | 3.81 | 2.61 | 4.81 | 6.97 | 2.77 | 6.03 | **1.00** |
| 400 | 3.87 | 2.10 | 4.84 | 7.00 | 3.19 | 6.00 | **1.00** |
| 500 | 3.27 | 3.26 | 4.97 | 6.97 | 2.50 | 6.03 | **1.00** |
| 600 | 3.48 | 3.52 | 4.87 | 7.00 | 2.13 | 6.00 | **1.00** |
| 700 | 3.16 | 3.74 | 4.84 | 6.97 | 2.26 | 6.03 | **1.00** |
| 800 | 2.90 | 4.03 | 4.77 | 6.97 | 2.29 | 6.03 | **1.00** |
| 900 | 2.97 | 4.19 | 4.68 | 7.00 | 2.16 | 6.00 | **1.00** |
| 1000 | 2.87 | 4.42 | 4.58 | 7.00 | 2.16 | 5.97 | **1.00** |

To further validate the performance of HNIO, we employ Friedman test to rank-order the optimization results of these algorithms in different scenarios of the number of UEs. Table III shows the rank-mean ranking of HNIO with other comparison algorithms in ten scenarios, and we bold the rank-mean of the best algorithm. We can obtain that APHS is the best among the seven algorithms in scenario $N$=100, while HNIO is with the best performance in the other nine scenarios. Wilcoxon signed-rank tests results of HNIO and the comparison algorithms are given in Table IV, as well as the mean error which is the difference between the optimization results of each algorithm and the algorithm obtaining the best optimization result. We can notice that APHS is the best among the seven algorithms at $N$=100. However, as the number of UEs increases, APHS shows progressively worse optimization results. In scenarios with larger UEs, APHS falls into premature convergence unable to update the excellent solution further. HNIO overcomes this problem and achieves




(a) $N$=100. (b) $N$=500. (c) $N$=1000.

Fig. 5. Comparison of the violin plots of the experimental results of HNIO with the comparison algorithms at the number of UEs $N$=100, 500 and 1000. The plots show the results of multiple independent running of these algorithms, from which the overall distribution of optimization results of these algorithms can be observed visually.




(a) $N$=100. (b) $N$=500. (c) $N$=1000.

Fig. 6. The best convergence curves of HNIO are compared with the comparison algorithms at the number of UES $N$=100, 500 and 1000. The figure shows the convergence status of these algorithms, from which the convergence precision and speed can be observed.

the best optimization results for the other nine UEs scenarios except $N$=100. Therefore, summarizing the experimental results, we can conclude that HNIO has a fast convergence speed as well as a strong optimization capability, and can still demonstrate a superior optimization effect as the size of UEs increases.

## V. Conclusions

In this paper, a hybrid nature-inspired algorithm is proposed with the goal of minimizing the energy consumption of a UAV-assisted MEC system. The energy consumption minimization problem of UAV-assisted MEC system concerns the joint optimization of computational resource allocation, UAV location and offloading decision, and the variables involve continuous space and discrete space. Our proposed HNIO incorporates multiple advantageous strategies, e.g., the population individuals are updated with different strategies. The individual updating process is adjusted by solutions-based probability and each individual will be operated with random probability of mutation. In addition, the clustering algorithm is used to determine whether the population falls in premature convergence using the cluster center as a reference during the iterative process. If the population stagnates then a new population is generated with the local best position of the current population using generalized opposition learning. By constructing a Markov model, we establish that HNIO converges towards the optimal position during continuous iterations.

The experimental study tests the optimization capability of HNIO with ten UEs of different sizes and compares it with six other state-of-the-art algorithms. The experimental results illustrate that HNIO is weaker than APHS in terms of optimization precision when the number of UEs is 100, but as the size of UEs increases, HNIO has a stable and excellent optimization effect and can effectively reduce the overall energy consumption compared to the six compared algorithms.

Our future research work focuses on the following directions. The optimization capability of HNIO will be further improved. HNIO will be applied to the energy consumption problem of a multi-UAV cooperative terrestrial base station-assisted MEC. In addition, the current studies are limited to a single objective, and our subsequent work will consider the design of a multi-objective UAV-assisted MEC system from the perspectives of user data rate, system energy consumption, and latency. HNIO will be extended as a multi-objective optimization algorithm to solve the problem.

TABLE IV
THE ERROR OF HNIO WITH CONTENDER ALGORITHM AND THE WILCOXON SIGNED-RANK TESTS RESULTS ON DIFFERENT NUMBER OF UES AT THE SIGNIFICANCE LEVEL OF 0.5.

| $N$ | CMFPA<br>Mean(Wilcoxon's test) | SGHS<br>Mean(Wilcoxon's test) | APHS<br>Mean(Wilcoxon's test) | APSO<br>Mean(Wilcoxon's test) | NBABC<br>Mean(Wilcoxon's test) | ATLBO<br>Mean(Wilcoxon's test) | HNIO<br>Mean |
|---|---|---|---|---|---|---|---|
| 100 | 6.17E-03(+) | 1.18E-02(+) | 0.00E+00(–) | 5.46E-01(+) | 8.88E-03(+) | 2.79E-01(+) | 8.22E-04 |
| 200 | 6.35E-02(+) | 1.63E-02(+) | 6.96E-02(+) | 1.47E+00(+) | 6.66E-02(+) | 8.25E-01(+) | 0.00E+00 |
| 300 | 1.79E-01(+) | 1.05E-01(+) | 2.21E-01(+) | 2.12E+00(+) | 1.22E-01(+) | 1.19E+00(+) | 0.00E+00 |
| 400 | 3.37E-01(+) | 1.32E-01(+) | 4.45E-01(+) | 3.02E+00(+) | 2.44E-01(+) | 1.53E+00(+) | 0.00E+00 |
| 500 | 4.96E-01(+) | 4.95E-01(+) | 8.06E-01(+) | 4.05E+00(+) | 3.52E-01(+) | 2.01E+00(+) | 0.00E+00 |
| 600 | 8.04E-01(+) | 8.21E-01(+) | 1.15E+00(+) | 4.84E+00(+) | 5.14E-01(+) | 2.49E+00(+) | 0.00E+00 |
| 700 | 9.13E-01(+) | 1.21E+00(+) | 1.52E+00(+) | 5.60E+00(+) | 7.30E-01(+) | 3.12E+00(+) | 0.00E+00 |
| 800 | 1.23E+00(+) | 1.77E+00(+) | 2.03E+00(+) | 6.47E+00(+) | 1.01E+00(+) | 3.45E+00(+) | 0.00E+00 |
| 900 | 1.68E+00(+) | 2.33E+00(+) | 2.51E+00(+) | 8.26E+00(+) | 1.23E+00(+) | 3.88E+00(+) | 0.00E+00 |
| 1000 | 1.91E+00(+) | 3.14E+00(+) | 3.16E+00(+) | 9.05E+00(+) | 1.55E+00(+) | 4.53E+00(+) | 0.00E+00 |

## REFERENCES


[1] N. K. Shah, "Big data and cloud computing: Pitfalls and advantages in data management," in *2015 2nd International Conference on Computing for Sustainable Global Development (INDIACom)*. IEEE, 2015, pp. 643–648.

[2] G. Zhang, W. Zhang, Y. Cao, D. Li, and L. Wang, "Energy-delay tradeoff for dynamic offloading in mobile-edge computing system with energy harvesting devices," *IEEE Transactions on Industrial Informatics*, vol. 14, no. 10, pp. 4642–4655, 2018.

[3] Y. Zhang, J. He, and S. Guo, "Energy-efficient dynamic task offloading for energy harvesting mobile cloud computing," in *2018 IEEE international conference on networking, architecture and storage (NAS)*. IEEE, 2018, pp. 1–4.

[4] J. Zhang, L. Zhou, Q. Tang, E. C.-H. Ngai, X. Hu, H. Zhao, and J. Wei, "Stochastic computation offloading and trajectory scheduling for uav-assisted mobile edge computing," *IEEE Internet of Things Journal*, vol. 6, no. 2, pp. 3688–3699, 2018.

[5] L. Xu, M. Chen, M. Chen, Z. Yang, C. Chaccour, W. Saad, and C. S. Hong, "Joint location, bandwidth and power optimization for thz-enabled uav communications," *IEEE Communications Letters*, vol. 25, no. 6, pp. 1984–1988, 2021.

[6] T. M. Ho, K.-K. Nguyen, and M. Cheriet, "Uav control for wireless service provisioning in critical demand areas: A deep reinforcement learning approach," *IEEE Transactions on Vehicular Technology*, vol. 70, no. 7, pp. 7138–7152, 2021.

[7] S. Aggarwal and N. Kumar, "Path planning techniques for unmanned aerial vehicles: A review, solutions, and challenges," *Computer Communications*, vol. 149, pp. 270–299, 2020.

[8] S. Aggarwal, M. Shojafar, N. Kumar, and M. Conti, "A new secure data dissemination model in internet of drones," in *ICC 2019-2019 IEEE International Conference on Communications (ICC)*. IEEE, 2019, pp. 1–6.

[9] S. Aggarwal, N. Kumar, and S. Tanwar, "Blockchain-envisioned uav communication using 6g networks: Open issues, use cases, and future directions," *IEEE Internet of Things Journal*, vol. 8, no. 7, pp. 5416–5441, 2020.

[10] E. Straffelini, S. Cucchiaro, and P. Tarolli, "Mapping potential surface ponding in agriculture using uav-sfm," *Earth Surface Processes and Landforms*, vol. 46, no. 10, pp. 1926–1940, 2021.

[11] P. Chhikara, R. Tekchandani, N. Kumar, S. Tanwar, and J. J. Rodrigues, "Federated learning for air quality index prediction using uav swarm networks," in *2021 IEEE Global Communications Conference (GLOBECOM)*. IEEE, 2021, pp. 1–6.

[12] S. Das and P. N. Suganthan, "Differential evolution: A survey of the state-of-the-art," *IEEE transactions on evolutionary computation*, vol. 15, no. 1, pp. 4–31, 2010.

[13] G. R. Harik, F. G. Lobo, and D. E. Goldberg, "The compact genetic algorithm," *IEEE transactions on evolutionary computation*, vol. 3, no. 4, pp. 287–297, 1999.

[14] J. Kennedy and R. Eberhart, "Particle swarm optimization," in *Proceedings of ICNN'95-international conference on neural networks*, vol. 4. IEEE, 1995, pp. 1942–1948.

[15] H.-G. Han, W. Lu, Y. Hou, and J.-F. Qiao, "An adaptive-pso-based self-organizing rbf neural network," *IEEE transactions on neural networks and learning systems*, vol. 29, no. 1, pp. 104–117, 2018.

[16] Y. Zeng, R. Zhang, and T. J. Lim, "Wireless communications with unmanned aerial vehicles: Opportunities and challenges," *IEEE Communications Magazine*, vol. 54, no. 5, pp. 36–42, 2016.

[17] B. Shang and L. Liu, "Mobile-edge computing in the sky: Energy optimization for air–ground integrated networks," *IEEE Internet of Things Journal*, vol. 7, no. 8, pp. 7443–7456, 2020.

[18] T. X. Tran and D. Pompili, "Joint task offloading and resource allocation for multi-server mobile-edge computing networks," *IEEE Transactions on Vehicular Technology*, vol. 68, no. 1, pp. 856–868, 2018.

[19] V. Nguyen, T. T. Khanh, P. Van Nam, N. T. Thu, C. S. Hong, and E.-N. Huh, "Towards flying mobile edge computing," in *2020 International Conference on Information Networking (ICOIN)*. IEEE, 2020, pp. 723–725.

[20] W. Ye, J. Luo, F. Shan, W. Wu, and M. Yang, "Offspeeding: Optimal energy-efficient flight speed scheduling for uav-assisted edge computing," *Computer Networks*, vol. 183, p. 107577, 2020.

[21] Y. Liu, S. Xie, and Y. Zhang, "Cooperative offloading and resource management for uav-enabled mobile edge computing in power iot system," *IEEE Transactions on Vehicular Technology*, vol. 69, no. 10, pp. 12 229–12 239, 2020.

[22] Q. Hu, Y. Cai, G. Yu, Z. Qin, M. Zhao, and G. Y. Li, "Joint offloading and trajectory design for uav-enabled mobile edge computing systems," *IEEE Internet of Things Journal*, vol. 6, no. 2, pp. 1879–1892, 2018.

[23] L. Zhang and N. Ansari, "Latency-aware iot service provisioning in uav-aided mobile-edge computing networks," *IEEE Internet of Things Journal*, vol. 7, no. 10, pp. 10 573–10 580, 2020.

[24] F. Zhou, Y. Wu, R. Q. Hu, and Y. Qian, "Computation rate maximization in uav-enabled wireless-powered mobile-edge computing systems," *IEEE Journal on Selected Areas in Communications*, vol. 36, no. 9, pp. 1927–1941, 2018.

[25] A. Slowik and H. Kwasnicka, "Nature inspired methods and their industry applicationsłswarm intelligence algorithms," *IEEE Transactions on Industrial Informatics*, vol. 14, no. 3, pp. 1004–1015, 2017.

[26] ——, "Evolutionary algorithms and their applications to engineering problems," *Neural Computing and Applications*, pp. 1–17, 2020.

[27] D. Karaboga, B. Gorkemli, C. Ozturk, and N. Karaboga, "A comprehensive survey: artificial bee colony (abc) algorithm and applications," *Artificial Intelligence Review*, vol. 42, no. 1, pp. 21–57, 2014.

[28] C. J. Santana Jr, M. Macedo, H. Siqueira, A. Gokhale, and C. J. Bastos-Filho, "A novel binary artificial bee colony algorithm," *Future Generation Computer Systems*, vol. 98, pp. 180–196, 2019.

[29] R. V. Rao, V. J. Savsani, and D. Vakharia, "Teaching–learning-based optimization: a novel method for constrained mechanical design optimization problems," *Computer-Aided Design*, vol. 43, no. 3, pp. 303–315, 2011.

[30] R. Rao, "Jaya: A simple and new optimization algorithm for solving constrained and unconstrained optimization problems," *International Journal of Industrial Engineering Computations*, vol. 7, no. 1, pp. 19–34, 2016.

[31] A. A. Salman, M. G. Omran, and I. Ahmad, "Adaptive probabilistic harmony search for binary optimization problems," *Memetic Computing*, vol. 7, no. 4, pp. 291–316, 2015.

[32] Z. W. Geem, J. H. Kim, and G. V. Loganathan, "A new heuristic optimization algorithm: harmony search," *simulation*, vol. 76, no. 2, pp. 60–68, 2001.

[33] Q.-K. Pan, P. N. Suganthan, M. F. Tasgetiren, and J. J. Liang, "A self-adaptive global best harmony search algorithm for continuous optimization problems," *Applied Mathematics and Computation*, vol. 216, no. 3, pp. 830–848, 2010.

[34] A. K. Shukla, P. Singh, and M. Vardhan, "An adaptive inertia weight teaching-learning-based optimization algorithm and its applications," *Applied Mathematical Modelling*, vol. 77, pp. 309–326, 2020.

[35] J.-q. Li, J.-w. Deng, C.-y. Li, Y.-y. Han, J. Tian, B. Zhang, and C.-g. Wang, "An improved jaya algorithm for solving the flexible job shop scheduling problem with transportation and setup times," *Knowledge-Based Systems*, vol. 200, p. 106032, 2020.

[36] X.-S. Yang, "Flower pollination algorithm for global optimization," in *International conference on unconventional computing and natural computation*. Springer, 2012, pp. 240–249.

[37] Y. Chen and D. Pi, "An innovative flower pollination algorithm for continuous optimization problem," *Applied Mathematical Modelling*, vol. 83, pp. 237–265, 2020.

[38] Y. Chen, D. Pi, and B. Wang, "Enhanced global flower pollination algorithm for parameter identification of chaotic and hyper-chaotic system," *Nonlinear Dynamics*, vol. 97, no. 2, pp. 1343–1358, 2019.

[39] Y. Chen, D. Pi, and Y. Xu, "Neighborhood global learning based flower pollination algorithm and its application to unmanned aerial vehicle path planning," *Expert Systems with Applications*, vol. 170, p. 114505, 2021.

[40] I. Pavlyukevich, "Lévy flights, non-local search and simulated annealing," *journal of computational physics*, vol. 226, no. 2, pp. 1830–1844, 2007.

[41] W. Zhang and Y. Liang, "Definition and properties of markov chain," in *Mathematics of operations research*. Xi'an Jiaotong University, 2003, pp. 67–87.

[42] R. Salgotra, U. Singh, G. Singh, S. Singh, and A. H. Gandomi, "Application of mutation operators to salp swarm algorithm," *Expert Systems with Applications*, vol. 169, p. 114368, 2021.

[43] F. J. Solis and R. J.-B. Wets, "Minimization by random search techniques," *Mathematics of operations research*, vol. 6, no. 1, pp. 19–30, 1981.

[44] O. B. FADEYI, "Robustness and comparative statistical power of the repeated measures anova and friedman test with real data," Ph.D. dissertation, Wayne State University, Detroit, 2021.

[45] F. Wilcoxon, "Individual comparisons by ranking methods," in *Breakthroughs in statistics*. Springer, 1992, pp. 196–202.


**Yang Chen** is currently pursuing the Ph.D. degree with the College of Computer Science and Technology, Nanjing University of Aeronautics and Astronautics.
His research interests include swarm intelligence optimization, UAV path planning, mobile edge computing

**Dechang Pi** was born in 1971. He received the B.Eng. and M.Eng. degree from the Nanjing University of Aeronautics and Astronautics, Nanjing, China, in 1994 and 1997, respectively, his Ph.D. degree in mechatronic engineering from Nanjing University of Aeronautics and Astronautics, in 2002. Now he is a professor and Ph.D. supervisor in Nanjing University of Aeronautics and Astronautics. He has authored or co-authored some academic papers which have been published in international journal, such as IEEE Transactions on Automation Science and Engineering, IEEE Sensors Journal, IEEE System Journal, Knowledge-Based Systems, Expert Systems with Applications, Journal, Computer & Operation Research, Computers & Industrial Engineering. His research interests are data mining, big data analysis in manufacturing, and intelligent optimization methods. He presided over 30 research projects of the National Natural Science Foundation of China, the National 863 Program, the National Technical Foundation, the Civil Aerospace Foundation, and the Aviation Science Foundation.

**Shengxiang Yang** (M'00-SM'14) received the Ph.D. degree from Northeastern University, Shenyang, China, in 1999. He is currently a Professor of Computational Intelligence and Deputy Director of the Institute of Artificial Intelligence, School of Computer Science and Informatics, De Montfort University, Leicester, U.K. He has over 360 publications with an H-index of 63 according to Google Scholar. His current research interests include evolutionary computation, swarm intelligence, artificial neural networks, data mining and data stream mining, and relevant real-world applications. Prof. Yang serves as an Associate Editor/Editorial Board Member of a number of international journals, such as the IEEE TRANSACTIONS ON EVOLUTIONARY COMPUTATION, IEEE TRANSACTIONS ON CYBERNETICS, Information Sciences, Enterprise Information Systems, and CAAI Transactions on Intelligence Technology.

**Yue Xu** received the B.Eng. degree in software engineering from Xidian University, Xian, China, in 2016. She is currently working toward the Ph.D. degree in computer science and technology from the Nanjing University of Aeronautics and Astronautics, Nanjing, China. Her current research interests include reliability design, intelligent optimization methods, and deep reinforcement learning.

**Junfu Chen** received the bachelors degree in air traffic management from the Civil Aviation University of China, Tianjin, China. He is currently pursuing the degree in computer science and technology with the Nanjing University of Aeronautics and Astronautics, Nanjing, China. His current research interests include developing data-driven tools for anomaly detection and Bayesian deep learning models.

**Ali Wagdy Mohamed** received the B.Sc.,M.Sc., and Ph.D. degrees from Cairo University,Egypt, in 2000, 2004, and 2010, respectively. He is currently an Associate Professor with the Operations Research Department, Faculty of Graduate Studies for Statistical Research, Cairo University. He is also an Associate Professor with the Department of Mathematics and Actuarial Science School of Sciences Engineering, The American University in Cairo. He has recognized among the top 2 scientists according to Stanford University report 2019. He has published more than 55 articles in reputed and high impact journals, such as Information Sciences, Swarm and Evolutionary Computation, Computers and Industrial Engineering, Intelligent Manufacturing, Soft Computing, and International Journal of Machine Learning and Cybernetics. His research interests include mathematical and statistical modeling, stochastic and deterministic optimization, swarm intelligence and evolutionary computation, real world problems, such as industrial, transportation, manufacturing, education, and capital investment problems. He has been awarded the Publons Peer Review Awards 2018, for placing in the top 1 of reviewers worldwide in assorted field. He is also an Associate Editor of Swarm and Evolutionary Computation Journal (Elsevier). He is also an Editor of more than ten journals, such as Information Sciences, Applied Mathematics, Engineering, System Science, and Operations Research.