Computers, Materials & Continua
DOI:10.32604/cmc.2021.018395
Article

Tech Science Press

Energy Optimization in Multi-UAV-Assisted Edge Data Collection System

Bin Xu[1,2,3], Lu Zhang[1], Zipeng Xu[1], Yichuan Liu[1], Jinming Chai[1], Sichong Qin[4] and Yanfei Sun[1,*]

[1]Nanjing University of Posts and Telecommunications, Nanjing, 210003, China
[2]Nanjing Pharmaceutical Co., Ltd., Nanjing, 210012, China
[3]Jiangsu Key Laboratory of Data Science and Smart Software, Jinling Institute of Technology, Nanjing, 211169, China
[4]Central Washington University, Ellensburg, 98926, United States
[*]Corresponding Author: Yanfei Sun. Email: sunyanfei@njupt.edu.cn
Received: 07 March 2021; Accepted: 17 April 2021

Abstract: In the IoT (Internet of Things) system, the introduction of UAV (Unmanned Aerial Vehicle) as a new data collection platform can solve the problem that IoT devices are unable to transmit data over long distances due to the limitation of their battery energy. However, the unreasonable distribution of UAVs will still lead to the problem of the high total energy consumption of the system. In this work, to deal with the problem, a deployment model of a mobile edge computing (MEC) system based on multi-UAV is proposed. The goal of the model is to minimize the energy consumption of the system in the process of data transmission by optimizing the deployment of UAVs. The DEVIPSK (differential evolution algorithm with variable population size based on a mutation strategy pool initialized by K-Means) is proposed to solve the model. In DEVIPSK, the population is initialized by K-Means to obtain better initial positions of UAVs. Besides, considering the limitation of the fixed mutation strategy in the traditional evolutionary algorithm, a mutation strategy pool is used to update the positions of UAVs. The experimental results show the superiority of the DEVIPSK and provide guidance for the deployment of UAVs in the field of edge data collection in the IoT system.

Keywords: UAV; mobile edge computing; differential evolution algorithm; K-Means; edge data collection

1 Introduction

The Mobile Edge Computing (MEC) Model [1] was proposed to solve such problems as delay, high pressure on network bandwidth, high energy consumption, and insufficient caching capacity [2] caused by centralized data processing in cloud computing mode. In recent years, some studies on UAV (Unmanned Aerial Vehicle) in mobile edge computing environment have been carried out in academia. Kim et al. [3] proposed an optimal task-UAV-edge server matching algorithm that minimizes energy consumption and processing time. Du et al. [4] regarded a UAV as an edge cloud. They proposed an effective iterative algorithm to minimize the energy consumption of the UAV on the premise of meeting the quality of service (QoS) requirements of the IoT devices and the computing resources available to the UAV. A UAV communication system based on mobile

edge computing was studied [5]. Under the condition of resource constraints, the path and task allocation of the UAV, and the computing speed of the CPU are jointly optimized to minimize the energy consumption of the whole system. Yang et al. [6] designed a multi-UAV deployment scheme for MEC enhanced IoT architecture, which provides computing offloading services for ground IoT devices with limited local processing capacity. Then they proposed a multi-UAV deployment mechanism based on a differential evolution algorithm to balance the load of UAVs. Mozaffari et al. [7] considered a UAV-enabled MEC system and established a game model to achieve lower energy consumption. Alzenad et al. [8] researched the optimal deployment of UAVs to cover the maximum number of ground users with minimum transmission power, and then studied deployment operation optimization under different QoS requirements [9]. Liu et al. [10] proposed a cooperative MEC network architecture that supports UAVs, in which UAV can help other UAVs to perform computing tasks. On this basis, a collaborative computing offloading scheme considering the interference suppression of UAVs to equipment was proposed. It should be noted that the positions and number of UAVs are not optimized or only the positions of UAVs are optimized in these papers. A variable population differential evolution algorithm was proposed [11], in which the positions and the number of UAVs are considered. In the whole optimization process, since the number of UAVs is a dynamic variable, a new coding mechanism [11] was introduced in the paper. So that each individual of the population is on behalf of a UAV, and the whole population represents the whole UAV group. That is, the size of the population represents the number of UAVs. Our paper makes some improvements to its model and algorithm on the basis of [11].

In the scenario of IoT, terminal IoT devices are usually unable to transmit over long distances because of their own energy limitations. Therefore, as a new data collection platform, the UAV was introduced into the IoT scene. To use UAVs in a highly efficient way, we need to optimize the deployment of UAVs to provide reliable and energy-saving data acquisition solutions for IoT devices. In this paper, a differential evolution algorithm with variable population size based on a mutation strategy pool initialized by K-Means [12] is proposed (DEVIPSK). Meanwhile, the number and the positions of UAVs are optimized.

The main contributions of this paper are as follows:

(1) A differential evolution algorithm with variable population size based on a mutation strategy pool initialized by K-Means is proposed to optimize the deployment of UAVs. It adopts the K-Means clustering algorithm to initialize the population. In addition, a control mechanism is also designed to dynamically change the parameter scaling factor F and crossover probability CR, to balance local search and global search. It also adopts a mutation strategy pool to help find the optimal mutation strategy.
(2) The results show that, compared with other algorithms on 8 different instances, DEVIPSK performs better. The experimental results indicate that DEVIPSK reduces effectively the energy consumption of the entire system while solving problems for its green characteristics.

The rest of this paper is arranged as follows. The second chapter details the system model. The third chapter introduces the algorithm and its innovation. The fourth chapter carries on the simulation contrast experiment and data analysis. The last chapter summarizes the full paper.

2 System Model

An IoT system, as shown in Fig. 1, contains a set of IoT devices deployed in a certain area, represented as $N = \{1, 2, \ldots, n\}$, and some fixed-location base stations. Because there may be a

long transmission distance or the base station used as an auxiliary calculation is too far away in the process of data transmission, a group of rotary-wing UAVs is used to collect data from these ground-based IoT devices to MEC servers. Here, we hypothesis that the set of the number of UAVs can be expressed as $K = \{1, 2, \ldots, k\}$.

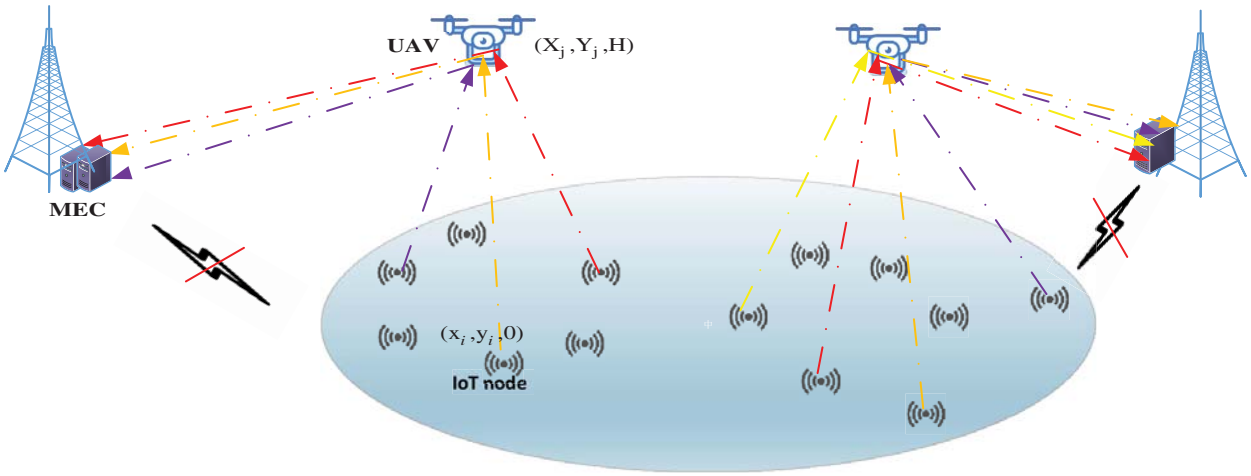

Figure 1: Scene diagram of multi-UAV-assisted Internet of Things

The distance between the IoT device i and the UAV j is expressed as Eq. (1):

$$d_{i,j} = \sqrt{(X_j - x_i)^2 + (Y_j - y_i)^2 + H^2} \quad \forall i \in N, j \in K \tag{1}$$

where $(x_i, y_i, 0)$ is the position of the IoT device i, (X_j, Y_j, and H) is the position of the UAV j. In this model, the height of the UAV is set to a fixed value H. The connection between the IoT device i and the UAV j is expressed as a binary variable $a_{i,j}$. Specifically, if there is a data exchange between the UAV and the IoT device, then $a_{i,j}$ equals 1; otherwise $a_{i,j}$ equals 0. To decrease energy, each IoT device always transmits data to the UAV closest to itself. Therefore, $a_{i,j}$ is set as Eq. (2):

$$a_{i,j} = \begin{cases} 1 & if \ j = \arg\min \ d_{i,j}, j \in K \\ 0 & otherwise. \end{cases} \tag{2}$$

Moreover, the constraint of $a_{i,j}$ is Eq. (3):

$$\sum_{j=1}^{k} a_{i,j} = 1, \quad \forall i \in N. \tag{3}$$

Each IoT device can only send data to one UAV. Moreover, taking into account the system bandwidth limitations, each UAV can receive data from up to M IoT devices at the same time. It can be expressed as Eq. (4):

$$\sum_{i=1}^{n} a_{i,j} \leq M, \quad \forall j \in K. \tag{4}$$

To ensure that all the IoT devices can be served, Eq. (5) should be satisfied:

$$\sum_{i=1}^{n} \sum_{j=1}^{k} a_{i,j} = n. \tag{5}$$

Each IoT device has a probability of establishing a line-of-sight relationship for a specific UAV. This probability depends on the environment of the IoT and the UAV, and the degree of elevation between the two. The probability of line-of-sight relationship [13] is Eq. (6):

$$P_{LoS} = \frac{1}{1 + \psi \exp(-\beta[\theta - \psi])}. \tag{6}$$

Among them, ψ and β are the constants depending on environment types and carrier frequency. θ is elevation angle, $\theta = \frac{180}{\Pi} \cdot \sin^{-1}\left(\frac{H}{d_{i,j}}\right)$, where $d_{i,j}$ is the distance between the IoT device i and the UAV j. H is the UAV's height.

From Eq. (6), it can be seen that the probability of establishing a line-of-sight relationship will be increased by increasing the degree of elevation or the height of the UAV. Reference [14] indicates that only when P_{LoS} is greater than a threshold, the two can be connected, so $P_{LoS}(\theta) \geq \varepsilon$ (ε is close to 1). Eq. (7) is derived from $\theta \geq P_{LoS}^{-1}(\varepsilon)$:

$$d_{i,j} \leq \frac{H}{\sin(P_{LoS}^{-1}(\varepsilon))}. \tag{7}$$

From Eq. (7), it can be concluded that the maximum distance between a UAV and an IoT device is $\frac{H}{\sin(P_{LoS}^{-1}(\varepsilon))}$.

In this model, the channel gain between the IoT device i and the UAV j [15] is expressed as Eq. (8):

$$h_{i,j} = h_0 d_{i,j}^{-2} = \frac{h_0}{(X_j - x_i)^2 + (Y_j - y_i)^2 + H^2}, \quad \forall i \in N, j \in K. \tag{8}$$

According to Eq. (8), the shorter the distance between the IoT device i and the UAV j, the greater its channel gain. This coincides with Eq. (2) in which each IoT device always chooses the nearest UAV to send data.

The rate at which the IoT device sends data to the UAV is Eq. (9):

$$r_{i,j} = B\log_2\left(1 + \frac{p_i h_{i,j}}{\sigma^2}\right) = B\log_2\left(1 + \frac{p_i h_0}{\sigma^2((X_j - x_i)^2 + (Y_j - y_i)^2 + H^2)}\right), \quad \forall i \in N, j \in K. \tag{9}$$

Among them, p_i is the transmission energy; h_0 indicates the channel power gain at reference distance $d_0 = 1m$; σ^2 is the noise power of white Gaussian noise and B is the system bandwidth. While the IoT device sends data volume D_i to the UAV, the time of transmitting is given by Eq. (10):

$$T_{i,j} = \frac{D_i}{r_{i,j}}, \quad \forall i \in N, j \in K. \tag{10}$$

Energy consumption is calculated using Eq. (11):

$$E_{i,j} = p_i T_{i,j} = \frac{p_i D_i}{r_{i,j}}, \quad \forall i \in N, j \in K. \tag{11}$$

Thus, energy consumption for all IoT devices is Eq. (12):

$$E_{iot} = \sum_{i=1}^{n} \sum_{j=1}^{k} a_{i,j} E_{i,j}. \tag{12}$$

The hover time of the UAV j is Eq. (13):

$$T_j^h = \max_{i \in N} \{a_{i,j} T_{i,j}\}, \quad \forall j \in K. \tag{13}$$

Further, the hover energy consumption of the UAV j is Eq. (14):

$$E_j^h = p^h T_j^h, \quad \forall j \in K, \tag{14}$$

where p^h represents the hover power of the UAV.

The distance between two UAVs should meet certain conditions so that there will be no collision [16]. The distance between the UAV $j1$ and the UAV $j2$ should be greater than d_{min}, as shown in Eq. (15):

$$d_{j1,j2} = \sqrt{(X_{j1} - X_{j2})^2 + (Y_{j1} - Y_{j2})^2} \geq d_{\min}. \tag{15}$$

In the model in this paper, the UAV serves as the data transfer station, and the data of the IoT device is finally sent to the edge server for processing. The volume of data received by the UAV j is shown in Eq. (16):

$$D_j = \sum_{i=1}^{n} a_{i,j} D_{i,j}, \quad \forall j \in K. \tag{16}$$

The time T_j of the UAV j sending data to the edge server is expressed as Eq. (17):

$$T_j = \frac{D_j}{r_j}, \quad \forall j \in K. \tag{17}$$

According to Eq. (9), r_j represents the rate at which the UAV j transmits data to the edge server.

The transmission energy of the UAV j is Eq. (18):

$$E_j = p_u T_j + p^h T_j, \quad \forall j \in K, \tag{18}$$

where p_u represents the UAV's transmission power.

The total energy consumption of the UAVs is Eq. (19):

$$E_{uav} = \sum_{j=1}^{k} (E_j^h + E_j). \tag{19}$$

The energy consumption of the whole system consists of the energy consumption of the UAVs and the energy consumption of the IoT devices. Thus, the problem can be expressed as Eq. (20):

$$\min_{\{X_j, Y_j\}, k} \quad (E_{uav} + \phi E_{iot}).$$

$$\text{s.t.} \quad C1: a_{i,j} \in \{0, 1\}, \quad \forall i \in N, j \in K$$

$$C2: \sum_{j=1}^{k} a_{i,j} = 1, \quad \forall i \in N$$

$$C3: \sum_{i=1}^{n} a_{i,j} \le M, \quad \forall j \in K$$

$$C4: \sum_{i=1}^{n} \sum_{j=1}^{k} a_{i,j} = n \tag{20}$$

$$C5: d_{i,j} \le \frac{h_j}{\sin(P_{LoS}^{-1}(\varepsilon))}$$

$$C6: d_{j1,j2} \ge d_{\min}$$

$$C7: X_{\min} \le X_j \le X_{\max}, \quad \forall j \in K$$

$$C8: Y_{\min} \le Y_j \le Y_{\max}, \quad \forall j \in K$$

$$C9: k_{\min} \le k \le k_{\max},$$

where $\varphi \ge 0$ is the weight of energy consumption of the IoT devices; $d_{i,j}$ represents the distance between the IoT device i and the UAV j; $d_{j1,j2}$ represents the distance between the two UAVs, and d_{min} is the shortest distance. X_{max} and X_{min} are the maximum and minimum values of X_j; Y_{max} and Y_{min} are the maximum and minimum values of Y_j; k_{max} and k_{min} are the maximum and minimum values of k; k_{min} and k_{max} are $\frac{n}{M}$ and n respectively; n represents the size of IoT devices; M represents the maximum size of IoT devices that a UAV can serve.

3 Method

DEVIPSK is proposed to optimize the deployment of UAVs, in which K-Means is used for population initialization, and a mutation strategy pool is proposed to help find the optimal mutation strategy. A control mechanism is also designed to dynamically change the parameter scaling factor F and crossover probability CR, to balance local search and global search.

The overall frame of the deployment optimization algorithm for UAVs is shown in Tab. 1.

Table 1: DEVIPSK framework for UAVs' deployment

Input:
the number of UAVs and IoTPosition
Output:
energy consumption of the system
1: $FE_s = 0$; // FE_s represents the number of times the fitness function is executed;
2: Initialize the population P according to Tab. 2;
3: **While** $FE_s \leq \max\ FE_s$ do
4: $Q = \Phi$;
5: **For** $i = 1$ to $|P|$ **do**
6: Execute the mutation and crossover operation of the differential evolution algorithm proposed in this paper to each individual x_i^G in P to generate the next generation x_i^{G+1};
7: $Q = Q \cup x_i^{G+1}$;
8: **End for**
9: Update the population P according to the method of adaptive population size in [11];
10: **End while**

The population P is initialized according to Tab. 2. In the whole process of evolution, the UAV deployment optimization algorithm is based on the mutation strategy pool. The mutation and crossover operations of dynamic parameters generate the progeny population Q. Then the adaptive population size strategy in [11] is used to update the population P. In this adaptive mechanism, a population update includes inserting, replacing, and deleting at most one individual, so as to ensure that the change of population size will not be too large. By comparing the fitness value of the population after inserting, replacing or deleting an individual, the population with the highest fitness value replaces the current population P. If the fitness value of the three has not improved, and the fitness value of the population after deleting the individual is equal to the fitness value of the current population P, then the current population will be replaced with the population after deleting the individual. For the mutation and crossover operation of the differential evolution algorithm, the method based on the mutation strategy pool and binomial crossover is adopted.

3.1 Population Initialization

Generally speaking, a UAV can serve multiple IoT devices. When the number of UAVs is initialized, it does not have to be directly set to the maximum value. The initial value k of UAVs is set to a value between $\left[\frac{n}{M}, n\right]$ to make the algorithm converge faster.

Table 2: The process of population initialization

Input:
the number of UAVs and IoTPosition
Output:
the initial positions of UAVs
1: The initial population P is generated by the K-Means algorithm;
2: $FE_s = FE_s + 1$;
3: **While**
Population P does not meet the constraints of the model and $FE_s \leq \max FE_s$ **do**
4: Re-generate the positions of the UAVs according to the K-Means algorithm;
5: The positions of the UAVs are considered to be the population P;
6: $FE_s = FE_s + 1$;
7: **End while**
8: Return P

We solve this problem by setting the number of UAVs to a cluster value. The K-Means is used to initialize the positions of the UAVs. The center of each cluster is the initial position of the UAV. As can be seen from Eq. (8), the shorter the distance between the IoT device and the UAV, the greater the channel gain.

The effect diagram executed with the K-Means algorithm is shown in Fig. 2, where the dots of different colors represent the positions of the IoT devices. The asterisk indicates the centroid of each cluster, which is the initial position of the UAVs.

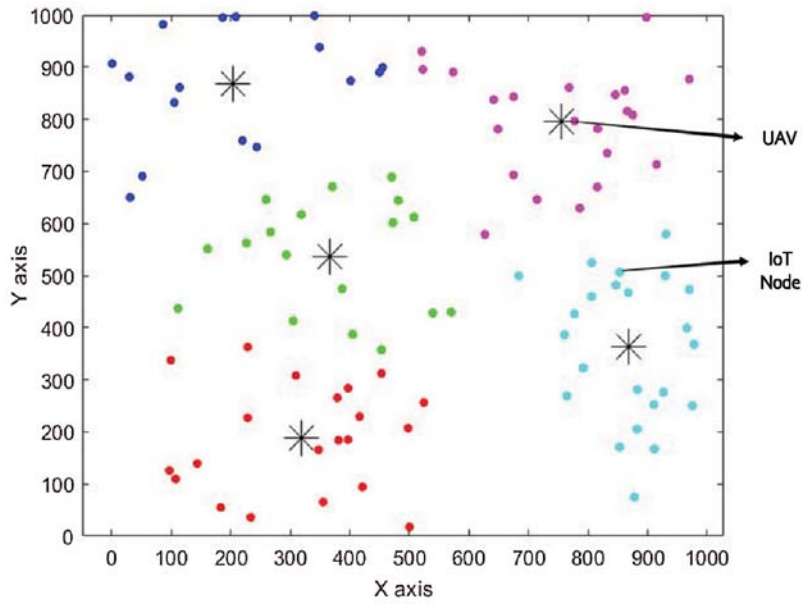

Figure 2: The distribution map of UAVs in the IoT devices through K-Means

The process of population initialization is shown in Tab. 2.

In the process of initialization, the original data can be divided into k non-intersecting clusters by using the clustering algorithm. That is, the position of the UAV is the center of each cluster, making the UAV the closest to each IoT device in the cluster. Then check whether the population P satisfies all the constraints of the model. If so, a feasible initial population P is successfully generated. If not, repeat the operation.

3.2 Update Population Based on Variation Pool Strategy and Dynamically Changing Parameters

In this paper, a method of mutation strategy candidate pool is proposed, which includes several effective mutation strategies with different characteristics. In the process of evolution, for each individual in the current population, a probability will be selected according to previous experience to select a strategy from the candidate pool to perform the mutation operation. We have studied several effective mutation strategies commonly used in DE literature, and selected the following three strategies as a candidate pool for mutation strategies.

(1) The "DE/rand/1" strategy converges slowly, but has a strong search ability. So it is suitable to solve multimodal problems, rather than relying on the optimal solution currently found.
(2) The "DE/rand/2" strategy based on two difference vectors may produce better disturbance than the strategy based on one difference vector. Under the background of particle swarm optimization, Storn [17] proved that all statistical distributions based on two difference vectors are better perturbation modes.
(3) "DE/current-to-rand/1" is a rotation invariant strategy, and it is mainly applied to solving multi-objective optimization problems.

It is found that a control mechanism can be used to dynamically change the parameter scaling factor F and crossover probability CR, to make the evolution algorithm perform better. The whole process is described in detail below.

According to the study by Brest et al. [18], the calculation method of scaling factor F is defined as Eq. (21), and the calculation method of crossover probability CR is defined as Eq. (22):

$$F_i^{G+1} = \begin{cases} F_l + rand_1 \cdot F_u, & if\ \ rand_2 < \tau_1 \\ F_i^G, & otherwise \end{cases}, \tag{21}$$

$$CR_i^{G+1} = \begin{cases} rand_3, & if\ \ rand_4 < \tau_2 \\ CR_i^G, & otherwise \end{cases}, \tag{22}$$

where $rand_j$, $j \in \{1, 2, 3, 4\}$ is the random number between [0, 1]. τ_1 and τ_2 are the probability to dynamically adjust F and CR. F_l and F_u are the range of F, F_i^{G+1} and CR_i^{G+1} are determined before the mutation operation. τ_1, τ_2, F_l, F_u are all preset values that are fixed throughout the operation. For different function problems, the appropriate control parameters are different. In our method, the change of $F$and CR is controlled by τ_1 and τ_2, so better control parameters will be used in the next iteration.

The algorithm flowchart of the whole system is shown in Fig. 3.

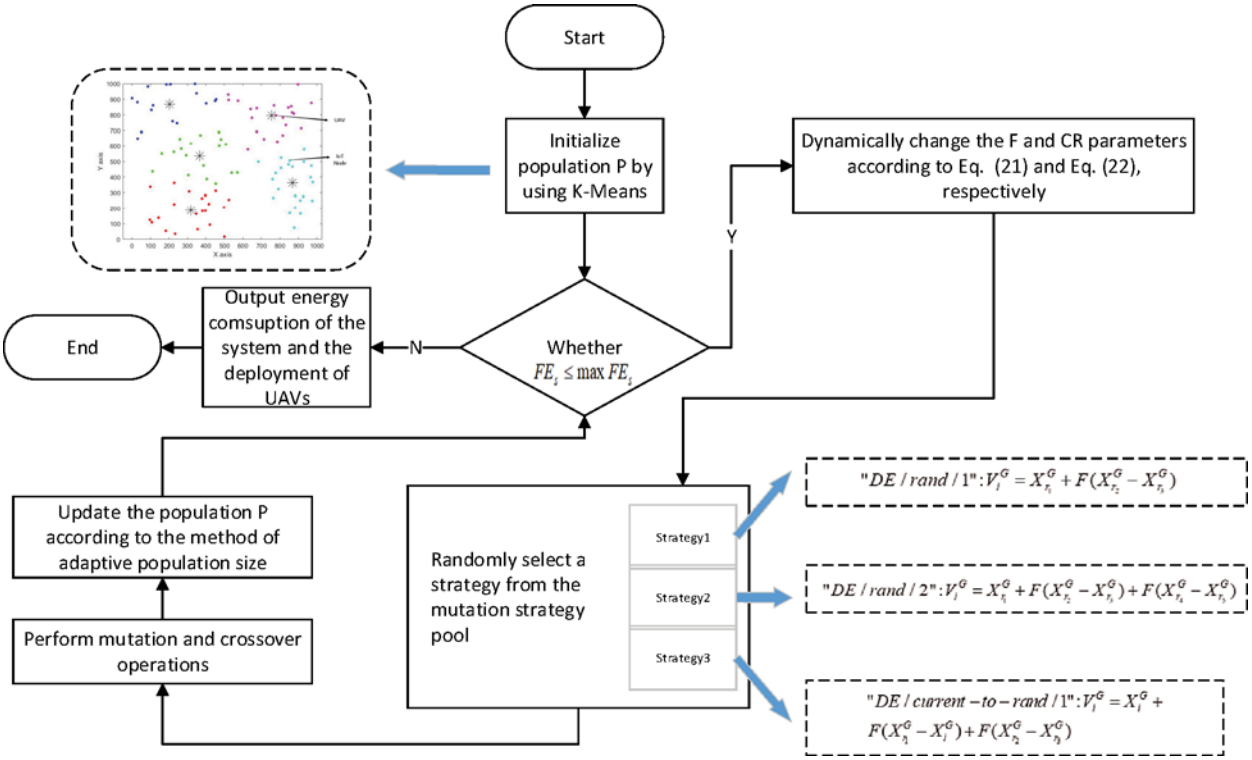

Figure 3: The framework of DEVIPSK

4 Simulation and Analysis

In the experiments, all the IoT devices are randomly deployed in a square area, and the size s is 1000 m $\times$ 1000 m; the height of the UAV H is 200 m; the position of the edge server $(x_0, y_0, 0)$ is (2000, 2000, 0); the data volume D_i $(i \in N)$ sent to the UAV by the ith IoT device is randomly distributed in the range of [1, 103] MB. A total of 8 examples are used to test the performance of our proposed algorithm, in which the number of IoT devices is n = {50, 100, 200, 300, 400, 500, 600, 700}. The specific parameters of the IoT scenario are shown in Tab. 3.

Table 3: The specific parameters of the IoT scenario

s	H	$(x_0, y_0, 0)$	D_i $(i \in N)$	M	P_i $(i \in N)$	h_0	σ^2
1000 m $\times$ 1000 m	200 m	(2000, 2000, 0)	[1, 103] MB	30	0.1 W	-30 dB	-250 dBm
B	p^h	p_u	φ	F	CR	F_l	F_u
1 MHz	1000 W	50 W	10000	0.5	0.9	0.1	0.9
τ_1	τ_2	ε	ψ	β	$d_{\min}$	n	
0.1	0.1	0.95	5	9	5	{50, 100, 200, 300, 400, 500, 600, 700}	

To test the statistical significance of the proposed algorithm and other algorithms, the Wilcoxon test is performed when the significant level is 0.05. In this paper, "+" and "−" are used to indicate how good or bad the DEVIPK algorithm is compared with other algorithms respectively.

4.1 The First Group of Experiments

For the sake of verifying the significance of the initial population by K-Means in DEVIPSK, DEVIPSK and DEVIPSK-K are compared under the same conditions in the first group of experiments. DEVIPSK-K does not use K-Means when initializing the population P. That is to say, DEVIPSK adds K-Means to initialize the population P on the basis of DEVIPSK-K.

As shown in Fig. 4, we can clearly see that in each different example, DEVIPSK generally keeps better results in the whole evolution process compared with DEVIPSK-K. Since the position of the UAV is the center of each cluster, it can make the UAV have the closest distance to each IoT device in this cluster compared with randomly generating the position of the UAV. According to Eq. (8), the smaller the distance between the two devices, the greater the channel gain. When the channel gain increases, the performance will also improve.

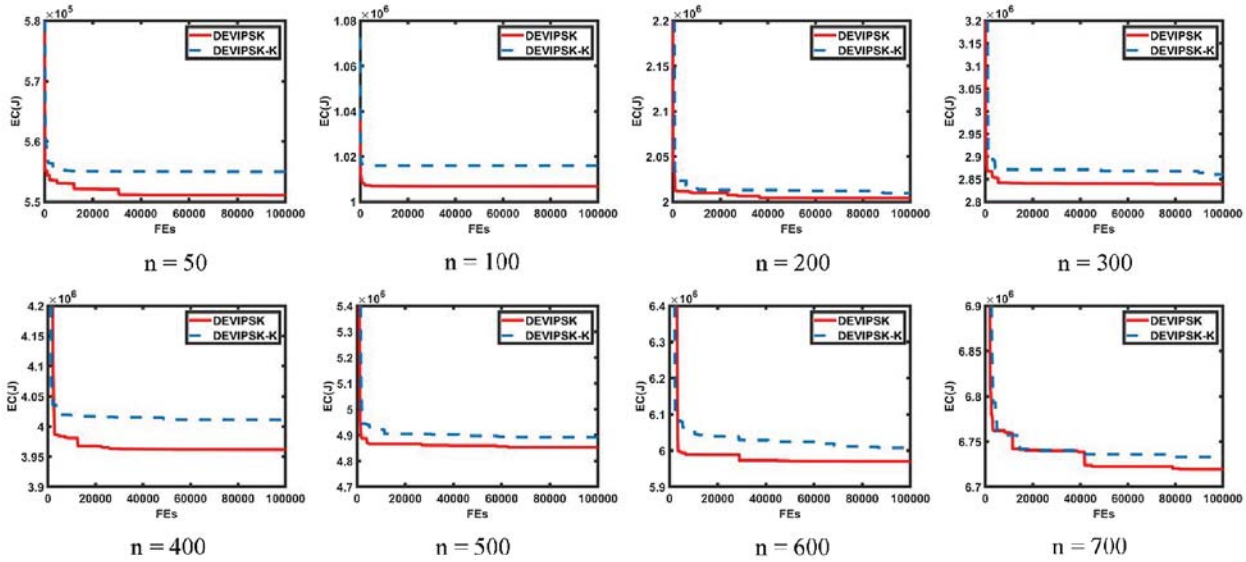

Figure 4: Comparison of EC(J) obtained by DEVIPSK and DEVIPSK-K when n is 50, 100, 200, 300, 400, 500, 600, 700, respectively

4.2 The Second Group of Experiments

In the second group of experiments, DEVIPSK was tested as a new variant of the differential evolution algorithm. We compared DEVIPSK with DEVIPSK-DE and DEVIPSK-jDE.

(1) DEVIPSK-DE [19] is the traditional differential evolution algorithm adopting the mutation and crossover operation of "DE/rand/1".
(2) DEVIPSK-jDE [18] is a differential evolution algorithm with dynamically changing parameters, which can dynamically change mutation factor F and crossover factor CR, to make the algorithm perform better.
(3) DEVIPSK adds a mutation strategy pool on the basis of DEVIPSK-jDE.

Tab. 4 shows the statistical test results. It represents the average and standard deviation of the energy of the whole system for 20 runs, and the percentage of performance improvement is shown in square brackets. Specifically, when n = 200, the most improvement was made with an increase of

4.78%. Compared with the differential evolution algorithm with dynamically changing parameters, in 6 out of 8 different examples, DEVIPSK obtained better performance. When n = 50, the most improvement was made, with an increase of 2.41%. But the differential evolution algorithm with dynamically changing parameters had better performance in the cases of n = 200 and n = 400.

Table 4: Experimental results of DEVIPSK, DEVIPSK-DE and DEVIPSK-jDE

n	DEVIPSK Mean (std dev)	DEVIPSK-DE Mean (std dev)	DEVIPSK-jDE Mean (std dev)
50	6.0146e+05 (4.2420e+04)	6.0834e+05 (4.6960e+04) [1.13%]	6.1629e+05 (4.4435e+04) [2.41%]
100	1.0618e+06 (2.3313e+04)	1.0634e+06 (2.0665e+04) [0.15%]	1.0630e+06 (2.2035e+04) [0.11%]
200	2.0928e+06 (3.8239e+04)	2.1978e+06 (3.2236e+04) [4.78%]	2.0882e+06 (5.6798e+04)
300	2.9450e+06 (6.1066e+04)	2.9484e+06 (7.8447e+04) [0.12%]	2.9543e+06 (9.5914e+04) [0.31%]
400	4.1247e+06 (8.1511e+04)	4.1328e+06 (1.0926e+05) [0.20%]	4.0098e+06 (6.9493e+04)
500	5.0140e+06 (1.6405e+05)	5.0519e+06 (1.2104e+05) [0.75%]	5.0329e+06 (1.1107e+05) [0.38%]
600	5.9650e+06 (1.0496e+06)	6.1902e+06 (2.0118e+05) [3.63%]	6.0701e+06 (1.4047e+05) [1.73%]
700	6.9494e+06 (2.3387e+05)	6.9751e+06 (2.0105e+05) [0.37%]	6.9789e+06 (1.7005e+05) [0.42%]
+		8	6
–		0	2

4.3 The Third Group of Experiments

DEVIPS [11], DEEM [20] and DEVPISM are compared with DEVIPSK in the third group experiments. The purpose is to reflect the superiority of DEVIPSK in energy consumption for the UAV model. The following is an introduction to three comparing algorithms.

(1) The coding mechanism adopted by DEVIPS is similar to that in this paper, but the number of k_{max} of UAVs is randomly generated when the population P is initialized. The mutation strategy pool and dynamic parameter mechanism are not used in DEVIPS.
(2) The coding mechanism adopted by DEEM is similar to that of this paper, but the number of UAVs needs to be preset and does not change in the whole running process of the algorithm. The mutation strategy pool and dynamic parameter mechanism are not used in DEEM.
(3) DEVPISM adopts the same coding mechanism as DEVIPSK and adopts the K-Means to initialize the population. The mutation strategy pool and dynamic parameter mechanism are used in the DEVPISM. But when updating the population, it performs replacement and removal to generate offspring population.

Fig. 5 shows that in each different example, the convergence rate of DEVIPSK and DEVPISM is better than that of DEVIPS and DEEM in the early stage, and keeps better results in the whole process of evolution. The position of the UAV is the center of each cluster. So compared with randomly generating the position of the UAV, the initialization of the K-Means can make the UAV have the closest distance to each IoT device in this cluster. According to Eq. (8), the smaller the distance between the two devices, the greater the channel gain, so the performance will be better.

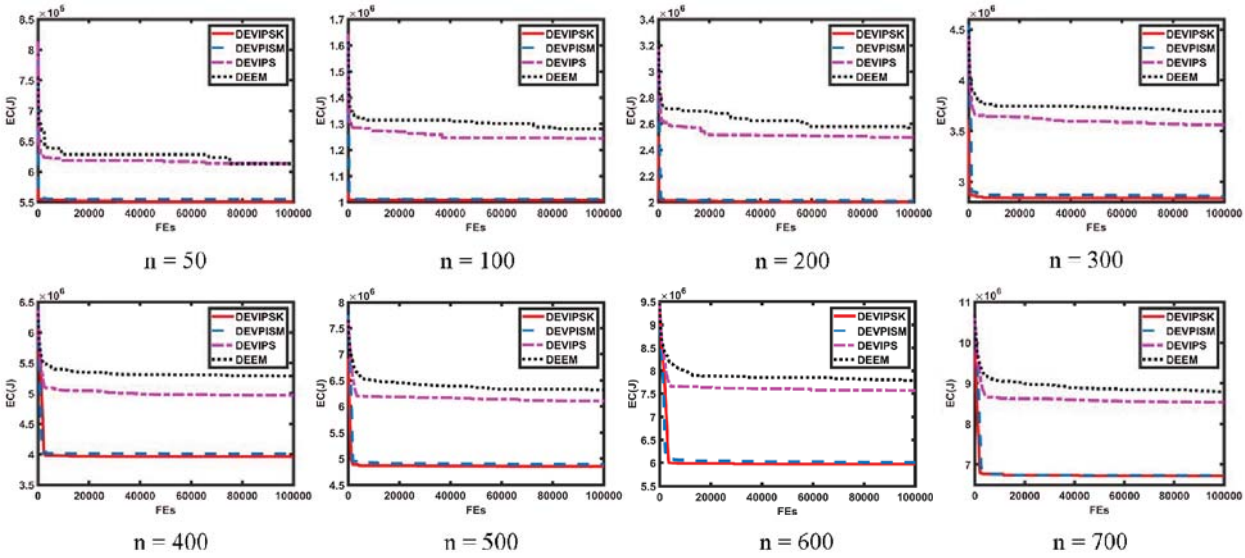

Figure 5: Comparison of EC(J) obtained by DEVIPSK, DEVPISM, DEVIPS, and DEEM when n is 50, 100, 200, 300, 400, 500, 600, 700, respectively

Since DEVIPSK adopts an adaptive mechanism to update the population, the energy consumption of DEVPISM and DEVIPSK in the process of evolution is very similar. In this adaptive mechanism, a population update includes inserting, replacing and deleting at most one individual. This can ensure that the change of population size will not be too large, so it will not have a great impact on the overall performance.

The statistical test results of different algorithms are shown in Tab. 5. It represents the average value and standard deviation of the energy consumption of the entire system for 20 runs, and the percentage of performance improvement in square brackets.

It can be seen that the DEVIPSK has good performance on each different instance. Compared to DEEM, DEVIPSK has the biggest performance improvement. When n = 600, DEEM consumed 9.4260e + 06 J energy, whereas DEVIPSK consumed 5.9650e + 06 J energy, which is 36.72% less. DEEM only randomly generates the positions of UAVs during initialization, and the number of UAVs remains the same throughout the process. Compared to DEVIPS, when n = 600, the improvement of DEVIPSK is the most, increasing 21.13%. Since DEVIPS only randomly generates the position of the UAV during initialization, it is effective to introduce the K-Means clustering algorithm when initializing the population. Compared to DEVIPSM, DEVIPSK also improves the performance of the whole system to a certain extent, but the improvement is the least. When n = 600, the performance improved the most, with a promotion of 4.40%.

Table 5: Experimental results of four different evolution algorithms

n	DEVIPSK Mean (std dev)	DEVPISM Mean (std dev)	DEEM Mean (std dev)	DEVIPS Mean (std dev)
50	6.0146e+05 (4.2420e+04)	6.0236e+05 (4.6960e+04) [0.14%]	8.0033e+05 (4.3557e+04) [24.85%]	6.1658e+05 (3.2300e+03) [2.45%]
100	1.0618e+06 (2.3313e+04)	1.0671e+06 (2.3350e+04) [0.50%]	1.5894e+06 (7.2984e+04) [33.19%]	1.2525e+06 (6.7254e+03) [15.23%]
200	2.0928e+06 (3.8239e+04)	2.1162e+06 (4.9636e+04) [1.11%]	3.1233e+06 (1.2304e+05) [32.99%]	2.5045e+06 (7.1329e+03) [16.44%]
300	2.9450e+06 (6.1066e+04)	2.9933e+06 (6.7190e+04) [1.61%]	4.4666e+06 (1.8742e+05) [34.07%]	3.5809e+06 (1.4834e+04) [17.76%]
400	4.1247e+06 (8.1511e+04)	4.1592e+06 (4.5523e+04) [0.83%]	6.2710e+06 (2.4091e+05) [34.23%]	5.0016e+06 (1.8278e+04) [17.53%]
500	5.0140e+06 (1.6405e+05)	5.1540e+06 (1.2085e+05) [2.72%]	7.6493e+06 (3.1705e+05) [34.45%]	6.1248e+06 (1.8445e+04) [18.17%]
600	5.9650e+06 (1.0496e+06)	6.2395e+06 (1.5306e+05) [4.40%]	9.4260e+06 (4.1082e+05) [36.72%]	7.5628e+06 (2.0203e+04) [21.13%]
700	6.9494e+06 (2.3387e+05)	7.0390e+06 (1.6720e+05) [1.27%]	1.0629e+07 (4.1422e+05) [34.62%]	8.5702e+6 (3.5668E+4) [18.91%]
+		8	8	8
–		0	0	0

5 Conclusion

Considering that the terminal IoT devices are usually unable to transmit over long distance due to the limitation of its battery energy, we try to solve this problem by combining multi-UAV deployment with mobile edge computing. Besides, a differential evolution algorithm with variable population size based on a mutation strategy pool initialized by K-Means is proposed to optimize the positions and number of UAVs. In a series of different examples, each instance has a different number of IoT devices. Through the comparison between DEVIPSK and other algorithms, the experimental results show the effectiveness of the algorithm. In the future, we will study the problem of task scheduling in a mobile edge computing system.

Funding Statement: This paper was supported in part by Project funded by China Postdoctoral Science Foundation under Grant 2020M671552, in part by Jiangsu Planned Projects for Postdoctoral Research Funds under Grant 2019K233, in part by NUPTSF (NY220060), in part by the Opening

Project of Jiangsu Key Laboratory of Data Science and Smart Software (No. 2020DS301), in part by Natural Science Foundation of Jiangsu Province of China under Grant BK20191381.

Conflicts of Interest: The authors declare that they have no conflicts of interest to report regarding the present study.

References

[1] R. Amin, M. Hussain, M. Alhameed, S. M. Raza, F. Jeribi *et al.*, "Edge-computing with graph computation: A novel mechanism to handle network intrusion and address spoofing in SDN," *Computers, Materials & Continua,* vol. 65*,* no. 3*,* pp. 1869–1890, 2020.

[2] X. L. Wei, J. W. Liu, Y. G. Wang, C. G. Tang and Y. Y. Hu, "Wireless edge caching based on content similarity in dynamic environments," *Journal of Systems Architecture,* vol. 115*,* pp. 102000–102007, 2021.

[3] K. Kim and C. S. Hong, "Optimal task-UAV-egde matching for computation offloading in UAV assisted mobile edge computing," in *2019 20th Asia-Pacific Network Operations and Management Symp.*, Matsue, Japan, pp. 1–4, 2019.

[4] Y. Du, K. Wang, K. Yang and G. Zhang, "Energy-efficient resource allocation in UAV based MEC system for IoT devices," in *2018 IEEE Global Communications Conf.*, Abu Dhabi, United Arab Emirates, pp. 1–6, 2018.

[5] L. Fan, W. Yan, X. Chen, Z. Chen and Q. Shi, "An energy efficient design for UAV communication with mobile edge computing," *China Communications,* vol. 16*,* no. 1*,* pp. 26–36, 2019.

[6] M. Yang, H. Yao, X. Zhang, J. Wang and Y. Liu, "Multi-UAV deployment for MEC enhanced IoT networks," in *2020 IEEE/CIC Int. Conf. on Communications in China*, Chongqing, China, pp. 436–441, 2020.

[7] M. Mozaffari, W. Saad, M. Bennis and M. Debbah, "Mobile unmanned aerial vehicles (UAVs) for energy-efficient internet of things communications," *IEEE Transactions on Wireless Communications,* vol. 16*,* no. 11*,* pp. 7574–7589, 2017.

[8] M. Alzenad, A. El-Keyi, F. Lagum and H. Yanikomeroglu, "3D placement of an unmanned aerial vehicle base station (UAV-bS) for energy-efficient maximal coverage," *IEEE Wireless Communication Letters,* vol. 6*,* no. 4*,* pp. 434–437, 2017.

[9] M. Alzenad, A. El-Keyi and H. Yanikomeroglu, "3-D placement of an unmanned aerial vehicle base station for maximum coverage of users with different QoS requirements," *IEEE Wireless Communications Letters,* vol. 7*,* no. 1*,* pp. 38–41, 2018.

[10] 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. 12229–12239, 2020.

[11] P. Huang, Y. Wang, K. Wang and K. Yang, "Differential evolution with a variable population size for deployment optimization in a UAV-assisted IoT data collection system," *IEEE Transactions on Emerging Topics in Computational Intelligence,* vol. 4*,* no. 3*,* pp. 324–335, 2020.

[12] S. Wu, Y. Liu, J. Wang and Q. Li, "Sentiment analysis method based on kmeans and online transfer learning" *Computers, Materials & Continua,* vol. 60*,* no. 3*,* pp. 1207–1222, 2019.

[13] A. Hourani, S. Kandeepan and A. Jamalipour, "Modeling air-to-ground path loss for low altitude platforms in urban environments," in *2014 IEEE Global Communications Conf.*, Austin, TX, USA, pp. 2898–2904, 2014.

[14] M. Mozaffari, W. Saad, M. Bennis and M. Debbah, "Mobile internet of things: can UAVs provide an energy-efficient mobile architecture," in *2016 IEEE Global Communications Conf.*, Washington, DC, pp. 1–6, 2016.

[15] Q. Wu, Y. Zeng, and R. Zhang, "Joint trajectory and communication design for multi-UAV enabled wireless networks," *IEEE Transactions on Wireless Communications,* vol. 17*,* no. 3*,* pp. 2109–2121, 2018.

[16] Y. Wang, Z. Ru, K. Wang and P. Huang, "Joint deployment and task scheduling optimization for large-scale mobile users in multi-UAV-enabled mobile edge computing," *IEEE Transactions on Cybernetics,* vol. 50*,* no. 9*,* pp. 3984–3997, 2020.

[17] R. Storn, "On the usage of differential evolution for function optimization," in *Biennial Conf. of the North American Fuzzy Information Processing Society*, Berkeley, CA, USA, pp. 519–523, 1996.

[18] J. Brest, S. Greiner, B. Boskovic and V. Zumer, "Self-adapting control parameters in differential evolution: A comparative study on numerical benchmark problems," *IEEE Transactions on Evolutionary Computation,* vol. 10*,* no. 6*,* pp. 646–657, 2006.

[19] R. Storn and K. Price, "Differential evolution-a simple and efficient heuristic for global optimization over continuous spaces," *Journal of Global Optimization,* vol. 11*,* no. 4*,* pp. 341–359, 1997.

[20] Y. Wang, H. Liu, H. Huang and P. N. Suganthan, "Differential evolution with a new encoding mechanism for optimizing wind farm layout," *IEEE Transaction on Industrial Informatics,* vol. 14*,* no. 3*,* pp. 1040–1054, 2018.