Hybrid Quantum-Behaved Particle Swarm Optimization for Mobile-Edge Computation Offloading in Internet of Things

Shijie Dai[1], Minghui Liwang[1], Yang Liu[1], Zhibin Gao[1(✉)], Lianfen Huang[1], and Xiaojiang Du[2(✉)]

[1] School of Information Science and Engineering, Xiamen University, Xiamen 361005, China
xmu_dsj@sina.com, minghuilw@stu.xmu.edu.cn, liuyangxmuce@126.com, {gaozhibin,lfhuang}@xmu.edu.cn
[2] Department of Computer and Information Sciences, Temple University, Philadelphia, PA 19122, USA
dux@temple.edu

Abstract. Mobile edge computing (MEC) is a technology that transfers resource to the edge of network, which spares more attention to giving users easier access to network and computation resources. Due to the large amount of data computation needed for devices in Internet of Things, MEC technology is applied to improve computing efficiency. Though MEC can be applied to the Internet of Things, it needs further consideration on how to efficiently and reasonably allocate computing resources, and how to minimize the computing time of all users. This paper proposes a computing resources allocation scheme based on hybrid quantum-behaved particle swarm optimization. Simulation experiments with the network environment based on the Internet of Things is carried out. The results show that this algorithm can accelerate the whole computing process and reduce the number of iterations.

Keywords: Internet of Things · Mobile edge computing · Offloading

1 Introduction

With the rapid development of information technology and information industry, the Internet of Things (IoT), supported by big data mining, cloud computing and machine learning, has been applied to various fields. And the appearances of newly-developing technology, cloud computing, bid data, AR and other technologies, promote the industry IoT upgrading. A recent study by NCTA in United States assumes that about 5.01 million Internet of Things will be connected to the Internet by 2020 [1].

The future era will rely on Internet technology to achieve the intelligent life, covering the fields in home security, environmental testing, energy, car networking, industrial intelligence manufacturing and other. IoT transforms itself from simple mode of things to the intelligent mode. IoT typically involve a large number of smart sensors that sense information from the environment and share it with the cloud service for processing. IoT application services can be divided into two types: one is a post hoc

L. Zhu and S. Zhong (Eds.): MSN 2017, CCIS 747, pp. 350–364, 2018.
https://doi.org/10.1007/978-981-10-8890-2_26

analysis, which collects data through the IoT terminal. And it upload to the cloud through the IoT private network or public network. Then the information will combine with bid data to be filtered and analyzed in the cloud. This application is often one-way. In other words, to capture and analysis do not need feedback data transmission. The other one is a real-time feedback type, which is making data acquisition and analysis not only through the IoT terminal, but also through the reverse real-time feedback. Such applications have higher requirements for latency and reliability.

Currently the IoT architecture is still cloud-centric architecture. Its main feature is that the communication exists between terminal and the cloud, and the main type of application services is a post hoc analysis. With the development of IoT, real-time feedback application requirements will increase more and more. And the IoT current architecture, cloud-centric architecture, is clearly unable to meet the needs of such applications. The main problems caused by generate data from IoT are: (1) IoT application processing time may be limited by the network delay in offloading data to the cloud. (2) the generation and upload of a large amount of IoT data causes network congestion resulting in further network delay.

To address the network problems designed in the Internet of Things and similar applications, researchers have proposed to bring computing closer to data generators and consumers. One suggestion is the fog computation [2], which enables the device to run the cloud application on its native architecture. The purpose of the fog is to perform low latency calculations/aggregations on the data while routing it to the central cloud for a large amount of calculation [3, 4]. On the other hand, the edge-centric computing cloud (mobile edge computing) [5] is inspired by projects such as SETI @ Home, Folding @ Home [6, 7], and the integration of voluntary human resources are proposed, such as desktop PC, Tablet, smartphone, nanometer data center as cloud. Since the resources in the Edge cloud are usually located near the hop of the Internet of Things sensors, processing the data at the edges can significantly reduce network latency [8, 9]. In addition, several papers (e.g., [10–20]) have studied the related wireless and IoT issues.

The essence of fog calculation/MEC is “near-service” and “segmental intelligence”. The traditional IoT uses a three-tier architecture, about the perceived layer, the network layer, and the platform layer. As shown in the figure, through the Internet of Things gateway equipped with MEC service platform, IoT gateway not only has the router function, but also obtain the abilities in storage and computing capacity in some actual application scenarios. Gateway nodes have the full ability to achieve the edge of intelligent networking. However, due to the wireless network resources provided by the nodes and the computation resource provided by the MEC server is limited, the problem of wireless and computation resource allocation problem is appeared. Therefore, how to choose access devices to provide computation offloading service under limited resources is a decision-making process.

In this paper, the MEC calculation offloading decision algorithm is studied. Firstly, the scene of wireless communication system with MEC calculation offloading technology under microcell is introduced. Then the content of the scene was mathematical modeling. It sums up the corresponding mathematical expression and establishes the problem model. Then the quantum-behaved particle swarm optimization algorithm is introduced to optimize the MEC calculation offloading decision, and the water injection algorithm is mixed into the above optimization algorithm. Based on this, a quantum-behaved particle swarm optimization algorithm with water injection algorithm is proposed to solve the above problems. And the algorithm is simulated. The simulation results show that the result of the algorithm approximates the optimal solution, which can effectively reduce the task's completion time.

The remainder of the paper is organized as follows. In Sect. 2, we describe the Edge-computation architecture and we propose our approach for deploying compute application tasks on the Edge-computation in efficient manner. In Sect. 3 we evaluate the effectiveness of our HQPSO algorithm by some simulation. Section 4 concludes the paper.

2 System Model

2.1 Systems Background

As shown in the Fig. 1, the IoT devices connect to the MEC server through a wireless network access node(AN). The AN is responsible for scheduling the time-frequency resource when the device communicates with the AN and is responsible for forwarding the data and related information of the device's calculation task to the Edge-Computation server. And the Edge-Computation server is responsible for scheduling the corresponding computing resources and storage resources for the device tasks to be evaluated. And it helps the devices to completes the calculation task and returns the calculated result to the AN. The AN then returns the result to the device via the wireless access network.

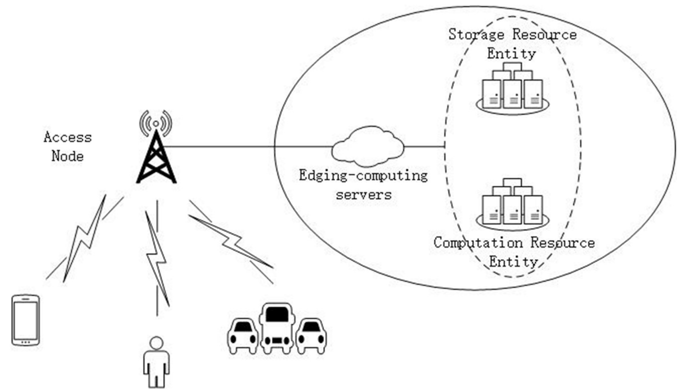

Fig. 1. The architecture of mobile edge computation offloading

Based on the above scenario, we model the radio access network as an OFDM system. And the minimum scheduling unit is a subcarrier (each subcarrier width is 15 kHz). When the device has a calculating task to upload, the AN schedules some subcarrier resources for the device. Edge-computation server, the cloud computing virtualization management server, manages many computing resources and storage resources. And in our simulation, We simplify the Edge-computation server. At a random point, multiple device's calculation task requests are uploaded to the AN. AN and Edge-computation server make joint decision, which is to assign all or many tasks to some computing resource. The order of assigning is following the principle named first-come-first-served (FCFS).

In our paper, we propose an algorithm to optimal the time for completing all calculation task by reasonably assigning computing resource.

2.2 System Parameters and Model

As mentioned in the previous section, we consider that the AN has a total of N subcarriers scheduled to M devices. The CPU of the computing resource, which is single-core single-threaded, execute by the principle of the first-come-first-served (FCFS).

Each device has only one calculation task to perform. They could choose to calculate locally or offloading to MEC for calculation. Some arguments are introduced in Table 1.

Table 1. The arguments of system

Arguments	Significance
f	The frequency of the CPU
$C = (1, 2, \ldots, N)$	The set of scheduled carriers
$U = \{1, 2, \ldots, M\}$	The device
B	The bandwidth of each carrier
$task_m = \{I_m, T_{max}^m\}, m \epsilon U$	The m-th device's calculation task
I_m	The number of m-th device's data bits
T_{max}^m	The maximum completion time allowed by the m-th calculation task
f_L^m	The CPU frequency of each device
P_{max}	The maximum transmit power
κ	The number of CPU clock cycles required to process each bit of data
R_m	The m-th device can upload rate

The time of computational tasks performed locally is defined as:

$$T_C^m = \frac{KI_m}{f_m} \tag{1}$$

Define $\exists! m : \eta(m, n) = 1, \forall m \epsilon U, \forall n \epsilon C$, which indicates whether the n-th carrier is assigned to the m-th device or not. $\eta(m, n) = 1$ indicates that the n-th carrier is assigned to the m-th device, and vice versa.

Define $\eta(m, n) = \{0, 1\}, \forall m \epsilon U, \forall n \epsilon C$, represents the power value on the n-th carrier of m-th device.

The total power of the device is limited, so we define:

$$P_m = \sum_{n=1}^{N} \eta(m,n)P(m,n) \le P_{max}, \forall m \epsilon U \tag{2}$$

which constrains the power of m-th device on all subcarriers. It is meant that the power of m-th device on all subcarriers shall not exceed the maximum transmit power of the device.

By the Shannon formula, the m-th device can upload the total rate is as:

$$R_m = B\sum_{n=1}^{N} \eta(m,n)\log(1+P(m,n)H(m,n)), \\ \forall m \epsilon U \tag{3}$$

Where $H(m,n)$ represents the channel gain for noise on the n-th carrier of device m.

Offloading calculation time includes the task's upload time, queuing time, execution time and download time.

Therefore, the device to calculate the task data upload time is:

$$T_U^m = \frac{I_m}{R_m} \tag{4}$$

The calculation execution time in MEC is:

$$T_C^m = \frac{KI_m}{f_m} \tag{5}$$

Since the schedule for assigning CPU of MEC is FCFS, the queuing time of the device is determined by its upload time and the execution time of the MEC calculation. Here, it is assumed that the MEC server provides the sufficient memory to store the upload data. Since the number of carriers per device is not necessarily same and the channel quality is different, the time for each device uploading and calculate the task data is different. According to the upload time is not of uniform size, we could get the order of the task reaching the MEC. Therefore, the tasks should be executed orderly.

Define $A = \{a_i | a_i = 1, 2, \ldots, M, i \neq j, i, j \epsilon U\}$, which represents each tasks' order of arrival, and the order of each task executing is a unique value.

The work of the preceding article got the task's upload time, arrival order and computing resources for the task calculation time. We can define the task's queuing time:

$$Q_{wait}^{a_i} = \begin{cases} 0, & a_i = 1 \\ max\left(0, T_U^j + T_C^j + Q_{wait}^{a_j} - T_U^i, a_j = a_i - 1, a_i \neq 1\right) \end{cases} \tag{6}$$

If the task is the first one arriving, then his waiting time must be equal to 0. If the task is not the first to arrive, the second formula is calculated. If the value is equal to 0, then the previous task has been completed, before the arrival of this task. Otherwise, the waiting time should be the above time difference.

After the task is executed by the MEC, the calculation result is returned to the device. The amount of result's data is very small, and the time to download it to the device is considered negligible.

Therefore, the completion time of the task for the device performing the calculation offloading is:

$$T_{MEC}^{m} = T_{U}^{m} + Q_{wait}^{a_i} + T_{C}^{m} \tag{7}$$

At this point, the minimum time to complete the task is calculated from follow equation set:

$$\min_{\eta, P, K_1, K_2} \sum_{k_1=1}^{k_1 \epsilon K_1} T_{L}^{k_1} + \sum_{k_2=1}^{k_2 \epsilon K_2} T_{MEC}^{k_2} \tag{8}$$

subject to

$$\exists! m : \eta(m, n) = 1, \forall m \epsilon U, \forall n \epsilon C \tag{9}$$

$$\eta(m, n) = \{0, 1\}, \forall m \epsilon U, \forall n \epsilon C \tag{10}$$

$$\sum_{m}^{M} \sum_{n}^{N} \eta(m, n) = N, \forall m \epsilon U, \forall n \epsilon C \tag{11}$$

$$P(m, n) \geq 0, \forall m \epsilon U, \forall n \epsilon C \tag{12}$$

$$P_m = \sum_{n=1}^{N} \eta(m, n) P(m, n) \leq P_{max}, \forall m \epsilon U \tag{13}$$

$$R_m = B \sum_{n=1}^{N} \eta(m, n) \log(1 + P(m, n) H(m, n)), \forall m \epsilon U \tag{14}$$

$$K_1 \cup K_2 = U, K_1 \cap K_2 = \varnothing \tag{15}$$

$$T_{MEC}^{k_2} = T_{U}^{k_2} + Q_{wait}^{a_i} + T_{C}^{k_2} \leq T_{L}^{k_2}, \forall k_2 \epsilon K_2 \tag{16}$$

2.3 Hybrid Quantum-Behaved Particle Swarm Optimization

The model with the smallest completion time established by (7)–(16) is a mixed integer nonlinear programming. The complexity of the problem is very high because its constraints contain integer terms and the nonlinearity of the objective function. And it is difficult to find the optimal solution of the objective function. Therefore, a feasible method is proposed based on the evolutionary algorithm.

First, we decompose the optimization problem established by (7)–(16) into sub-problems 1 and sub-problem 2. The mathematical model of Sub-problem 1 consists of (16)–(18).

The restriction condition is that the total number of subcarriers is limited and whether the condition for the device offloading calculation satisfies that the total time of the MEC calculation is less than the local calculation time or not. The optimization goal is to maximize the time saved.

$$\max_{\eta,K_2} \sum_{k_2=1}^{k_2 \epsilon K_2} T_L^{k_2} - T_{MEC}^{k_2} \tag{16}$$

$$subject\ to$$

$$\sum_{m}^{M} \sum_{n}^{N} \eta(m,n) = N, \forall m \epsilon U, \forall n \epsilon C \tag{17}$$

$$T_{MEC}^{k_2} = T_U^{k_2} + Q_{wait}^{a_i} + T_C^{k_2} \le T_L^{k_2}, \forall k_2 \epsilon K_2 \tag{18}$$

The mathematical model of sub-problem 2 consists of (19)–(24), which mainly solves the calculation of subcarrier and power allocation and the total uploading rate.

$$\max_{\eta,P_2} R_m \tag{19}$$

$$subject\ to$$

$$\exists! m : \eta(m,n) = 1, \forall m \epsilon U, \forall n \epsilon C \tag{20}$$

$$\eta(m,n) = \{0,1\}, \forall m \epsilon U, \forall n \epsilon C \tag{21}$$

$$P(m,n) \ge 0, \forall m \epsilon U, \forall n \epsilon C \tag{22}$$

$$P_m = \sum_{n=1}^{N} \eta(m,n) P(m,n) \le P_{max}, \forall m \epsilon U \tag{23}$$

$$R_m = B \sum_{n=1}^{N} \eta(m,n) \log(1 + P(m,n) H(m,n)), \quad \forall m \epsilon U \tag{24}$$

It is not difficult to find that the device waiting time in sub-problem 1 needs to be solved by the device upload rate obtained from sub-problem 2. Therefore, solving sub-problem 1 needs to solve sub-problem 2 first, and then solve sub-problem 1 to get the solution of modeling problem. However, it is necessary to determine the allocation of the K_2 carriers in sub-problem 1 to solve sub-problem 2. For the above analysis, we propose a quantum behavior particle swarm optimization algorithm that combines the water-filling algorithm to solve the sub-problem 1 and the sub-problem 2, so that the optimization problem established by (7)–(15) is also solved.

First, we apply the quantum-behaved particle swarm algorithm to solve sub-problem 1 and initialize K particles to represent the initialization of M devices in sub-problem 1. The k-th particles are initialized as $X_k = \left(X_k^1, X_k^2, \ldots, X_k^m, \ldots, X_k^M\right)$. Unlike the previous section, where X_k^m represents only the subcarrier allocation of the m-th device, but no more parameters on the allocation of power on the subcarriers are involved. Thus, the dimensions of the solution are reduced.

$$X_k^m = (\eta(m,1), \ldots, \eta(m,n), \ldots, \eta(m,N)) \tag{25}$$

Similarly, $\eta(m,n)$ is 0 or 1, indicating whether the n-th carrier is allocated to the m-th device or not.

When the particle swarm is initialized, it means that the initial subcarrier allocation has been determined. In this case, sub-problem 2 can be simplified as shown in Eqs. (26)–(28). By solving sub-problem 2, the maximum uplink transmission rate of the device can be calculated. Where C_m is the set of subcarriers in each column of X_k^m equal to 1, and J is the total number of elements equal to 1 in row vector X_k^m, which is the total number of subcarriers allocated to device m.

$$\max_P R_m = B\sum_{j=1}^{J} \log(1 + P(m,j)H(m,j)), \quad \forall j \epsilon C_m \tag{26}$$

$$\textit{subject to}$$

$$P(m,j) \geq 0, \forall j \epsilon C_m \tag{27}$$

$$\sum_{j=1}^{J} P(m,j) \leq P_{max} \tag{28}$$

At this point, solving sub-problem 2 can be understood as the maximum uplink transmission rate of the device when the total power of the device equipment transmitted is invariable.

To solve sub-problem 2, we establish the Lagrangian equation for (26)–(28) according to the Karush-Kuhn-Tucker (KKT) condition as (28):

$$\begin{aligned} L(P,\mu) &= B\sum_{j=1}^{J} \log(1 + P(m,j)H(m,j)) \\ &-\mu\left(\sum_{j=1}^{J} P(m,j) - P_{max}\right) \end{aligned} \tag{29}$$

Where μ *i*s the Lagrangian multiplier, which is a constant.

Next, to solve the optimal transmission power of the device m on each carrier, the transmission power of the m-th device in the Eq. (29) is subjected to calculate its partial derivative:

$$\begin{cases} \frac{\partial L(P,u)}{\partial P(m,1)} = 0 \\ \frac{\partial L(P,u)}{\partial P(m,2)} = 0 \\ \cdot \\ \cdot \\ \frac{\partial L(P,u)}{\partial P(m,J)} = 0 \end{cases} \tag{30}$$

Solve Eq. (30), and we could get the equation J shown equation.

$$\mathrm{P(m,j)} = \max_{\forall \mathrm{j} \epsilon \mathrm{C_m}} \left\{ 0, \frac{\mathrm{B}}{\ln 2 \cdot \mu} - \mathrm{H(m,j)}^{-1} \right\}, \tag{31}$$

From the Eq. (31) we can see that if the sub-carrier's gain-to-noise ratio is bigger, which means that the better channel quality of the subcarrier, the assigned transmit power of the subcarrier should be larger. The idea of power allocation above is the principle of water-filling algorithm.

In the following, we use the water-filling algorithm proposed in [22] to solve the sub-problem 2 that the subcarrier transmit power is allocated under the condition that the total transmit power is limited to maximize the total rate of the device. Compared with the classical binary search water-filling algorithm, the water-level algorithm reduces the complexity of the algorithm because it does not solve the Lagrangian factor directly by iterative method, but by using the iterative factors for the inconvenient adjustment of the iterative adjustment until all the power distribution is completed in power limited and initial water-filling line assumed conditions.

Then, back to the analysis of sub-problem 1, after get the device's upload rate, we can calculate the time of task uploading and calculating. According to the upload time, we could get the order of the tasks reach the AN for getting the execution order and the waiting time of tasks.

In Sub-problem 1, the goal of solving the problem is to maximize the difference between the completion time of the offloading task calculation task and the completion time of the local task calculation. Previously, by initializing the M particles representing the subcarrier assignment of K devices in sub-problem 1, there is a certain degree of randomness. Because the device's subcarrier allocation determines whether the device can perform the offloading calculation. And we can get all the possible distribution and device uninstall calculation.

Due to adopting the method of split solution, we need to modify penalty coefficient function of the fitness function. In the sub-problem 2, the water-filling algorithm has been applied to solve the power-related constraints, so the penalty function is the function (32) term.

$$\sum_{n=1}^{N}\left(\sum_{m=1}^{M}\eta(m,n)-1\right)^2 \tag{32}$$

Thus, the final fitness function is shown in (33), where α is the penalty factor.

$$f(X)=\sum_{k_2=1}^{k_2\epsilon K_2}(T_L^{k_2}-T_{MEC}^{k_2})-\alpha\cdot\sum_{n=1}^{N}\left(\sum_{m=1}^{M}\eta(m,n)-1\right)^2 \tag{33}$$

In addition, the device needs the data size of the computing task, the calculation rate of their own equipment and other information reported to the AN. these all are a priori information. In the premise of obtaining these information, the AN solve the decision problem to get the final subcarrier allocation results through the proposed algorithm, and reply to the device. And the device will decide to choose the offloading calculation or local calculation according to the results of the answer.

3 Simulation Setting and Result

3.1 Simulation Parameter Setting

Table 1 shows the simulation parameters of the MEC calculation unloading decision algorithm based on the hybrid quantum-behaved particle swarm optimization algorithm proposed in the mobile communication macro cellular network.

The simulation scene is a hexagonal area (500 * 500 m), and the user's cellular terminal equipment evenly distributed in the network. The simulation scene is shown in Fig. 2. The small red dots represent devices, and the blue point represents the macro base station, the devices' location in the figure is randomly generated. To ensure the

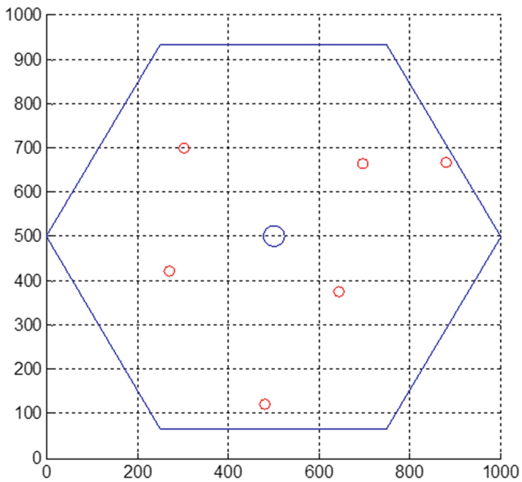

Fig. 2. The paper's simulation scene

generalization of the simulated devices' position, we compare the average of the solution obtained by solving the algorithm with 100 devices randomly generated and the average of the 100 optimal solutions Values. The path loss mainly considers large-scale fading. The channel model refers to the LTE COST231-HaTa propagation model proposed by 3GPP [23]. And the MEC calculation speed reference to [21]. And the rest of the simulation parameters are shown in Table 2. Through repeated experiments, the value of the penalty factor α of the penalty function in the fitness function is set to be 0.05, can get the better simulation result.

Table 2. Simulation parameters

Parameters	Values
Area of simulation Scene	Regular hexagon (length of side: 600 m)
Working frequency	2.4 GHz
MEC rate of computing	8×10^8 cycle/s
The amount of task data	(5–20) KB
Value of κ	200
Subcarrier bandwidth	15 kHz
BS coverage	500 m
Maximum transmit power	24 dBm
Thermal noise power density	−174 dBm/Hz
Number of iterations	300

3.2 Simulation Result

In the simulation, we compare the performance of the proposed algorithm with the average calculation completion time of the optimal solution introduced. Figure 3 shows the average time of completing the calculation and obtaining the optimal solution by the offloading decision algorithm in different situations with the different number of subcarriers. As is shown in the figure, with the increasing of the total number of subcarriers, the average time at which the user completes the computational task is reduced. In addition, it can be seen from the figure that the performance of proposed HQPSO algorithm is close to the one of optimal solution. In the hybrid quantum-behaved particle swarm optimization algorithm, the allocation of subcarriers is randomly initialized firstly, and the penalty function is introduced to modify the fitness function to solves the optimization target problem. This not only considers the sub-carrier resources that may be wasted due to the long offload time, but also the situation in which the user's task is queued after offloading. It effectively allocate all available subcarriers to the devices who uploaded the calculation. It can be seen from Fig. 3 that the solution result based on the hybrid quantum-behaved particle swarm optimization algorithm is very small and the difference is less than 5%.

Figure 4 shows the number of different users, by selecting some devices to offloading the calculation, we can optimize the calculation of the completion of the time value. It can be seen from the figure that the performance of the random allocation algorithm is very poor because it does not consider the devices' situation. However, the

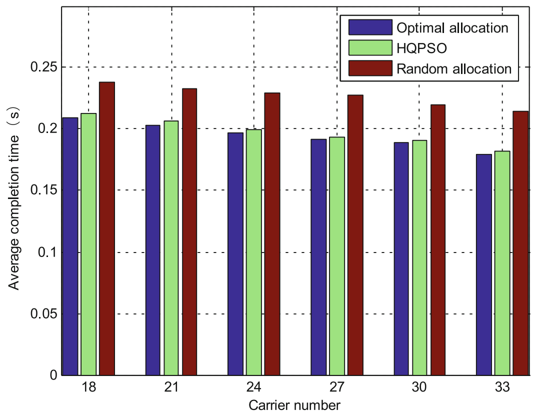

Fig. 3. The average time for computational task

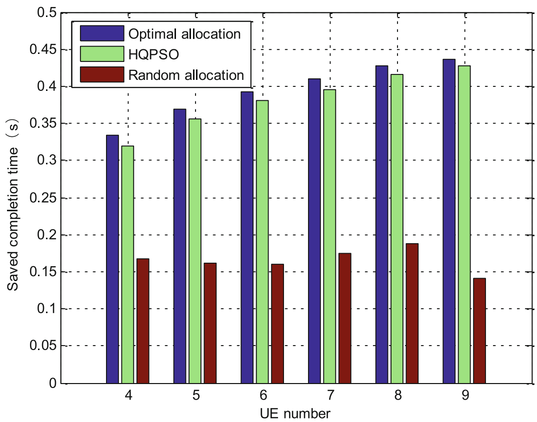

Fig. 4. The total time saved by each algorithm

results of the HQPSO algorithm we proposed and the optimal solution increase with the increase in the number of devices. And the calculation time is also increasing. This is because that the greater the number of selectable devices, the greater the likelihood of a better allocation scheme in the case of allocating the same number of subcarriers. It can be seen from Fig. 3 that the results of the proposed HQPSO algorithm are still close to those of the optimal solution.

Figure 4 shows the reduction in the completion time of the unloading calculation with the total transmission power of the different devices. As is seen from the figure, with the increase of the total transmission power of the device terminal equipment, the optimization of the equipment offloading computation time is increased, which is meant that the user can complete the task calculation in a shorter time. This is mainly due to the increase in allocable power resulting in an increase in user upload rates. So, the calculation time for each task is decreasing

Figure 5 is the calculated time reduction for HQPSO in the case of 6 devices assigned 30 carriers. The result is averaged by 100 iterations of randomly generated device simulations. The optimal solution value is the value of the fitness function obtained when the optimal subcarrier is allocated. The final value of HQPSO is the global optimal value of the fitness function obtained by iterative convergence. It can be observed from the figure that the proposed algorithm can quickly reach the convergence of the iteration. In addition, the accuracy of the algorithm is relatively high, it usually could be achieved with the optimal value of less than 10% within 50 iterations. This is because we have incorporated the water-filling algorithm into the iterative solution of the quantum-behaved particle swarm optimization, which reduces the dimension of the feasible solution, accelerates the speed of the iterative convergence and improves the accuracy of the solution.

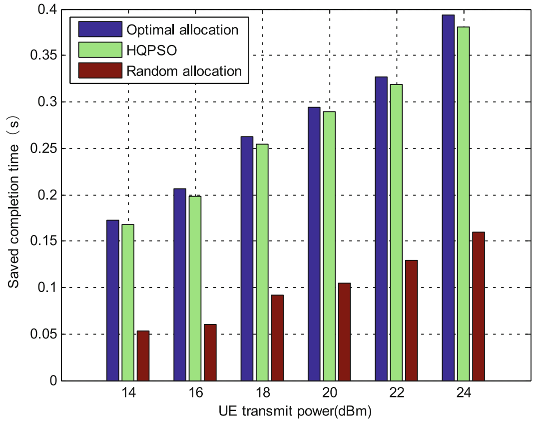

Fig. 5. The calculated time reduction for HQPSO in the case of 6 devices assigned 30 carriers

4 Conclusion

This paper first introduces the research progress of Internet of Things (IoT) and mobile edge computing. And the mathematical modeling is carried out on this basis, and the corresponding optimization problem model is put forward. Then, to reduce the complexity of iterative computation, this paper proposes a hybrid quantum behavior particle swarm optimization algorithm to solve the optimization problem. Finally, we verify the performance of the proposed algorithm, and analyze the performance of the proposed algorithm. And experimental results prove that this algorithm can accelerate the whole computing process and lessening the iterations.

Acknowledgment. The work presented in this paper was partially supported by the 2015 National Natural Science Foundation of China (Grant number 61401381).

References

1. Cremer, D., Bang, N., Simkin, L.: The integrity challenge of the Internet-of-Things (IoT): on understanding its dark side. J. Mark. Manage. **33**(1–2), 145–158 (2017)
2. Bonomi, F., Milito, R., Natarajan, P., Zhu, J.: Fog computing: a platform for internet of things and analytics. In: Bessis, N., Dobre, C. (eds.) Big Data and Internet of Things: A Roadmap for Smart Environments. SCI, vol. 546, pp. 169–186. Springer, Cham (2014). https://doi.org/10.1007/978-3-319-05029-4_7
3. Yannuzzi, M., Milito, R., Serral-Gracià, R., et al.: Key ingredients in an IoT recipe: fog computing, cloud computing, and more fog computing. In: 2014 IEEE 19th International Workshop on Computer Aided Modeling and Design of Communication Links and Networks (CAMAD), pp. 325–329. IEEE (2014)
4. Hong, K., Lillethun, D., Ramachandran, U., et al.: Mobile fog: a programming model for large-scale applications on the internet of things. In: Proceedings of the Second ACM SIGCOMM Workshop on Mobile Cloud Computing, pp. 15–20. ACM (2013)
5. Lopez, P.G., Montresor, A., Epema, D., et al.: Edge-centric computing: vision and challenges. ACM SIGCOMM Comput. Commun. Rev. **45**(5), 37–42 (2015)
6. Korpela, E., Werthimer, E., Anderson, D., et al.: SETI@HOME—massively distributed computing for SETI. Comput. Sci. Eng. **3**(1), 78–83 (2001)
7. Beberg, A.L., Ensign, D.L., Jayachandran, G., et al.: Folding@home: lessons from eight years of volunteer distributed computing. In: IEEE International Symposium on Parallel & Distributed Processing, IPDPS 2009, pp. 1–8. IEEE (2009)
8. Islam, S., Grégoire, J.C.: Giving users an edge: a flexible cloud model and its application for multimedia. Future Gener. Comput. Syst. **28**(6), 823–832 (2012)
9. Chandra, A., Weissman, J., Heintz, B.: Decentralized edge clouds. IEEE Internet Comput. **17**(5), 70–73 (2013)
10. Yao, X., Han, X., Du, X., Zhou, X.: A lightweight multicast authentication mechanism for small scale IoT applications. IEEE Sens. J. **13**(10), 3693–3701 (2013)
11. Du, X., Wu, D., Liu, W., Fang, Y.: Multi-class routing and medium access control for heterogeneous mobile ad hoc networks. IEEE Trans. Veh. Technol. **55**(1), 278–285 (2006)
12. Hei, X., Du, X.: Biometric-based two-level secure access control for implantable medical devices during emergency. In: Proceedings of IEEE INFOCOM 2011, Shanghai, China, April 2011
13. Du, X., Xiao, Y., Chen, H.H., Wu, Q.: Secure cell relay routing protocol for sensor networks. Wirel. Commun. Mobile Comput. **6**(3), 375–391 (2006)
14. Xiao, Y., Rayi, V., Sun, B., Du, X., Hu, F., Galloway, M.: A survey of key management schemes in wireless sensor networks. J. Comput. Commun. **30**(11–12), 2314–2341 (2007)
15. Du, X., Xiao, Y., Guizani, M., Chen, H.H.: An effective key management scheme for heterogeneous sensor networks. Ad Hoc Netw. **5**(1), 24–34 (2007)
16. Hei, X., Du, X., Wu, J., Hu, F.: Defending resource depletion attacks on implantable medical devices. In: Proceedings of IEEE GLOBECOM 2010, Miami, Florida, USA, December 2010
17. Du, X., Guizani, M., Xiao, Y., Chen, H.H.: A routing-driven elliptic curve cryptography based key management scheme for heterogeneous sensor networks. IEEE Trans. Wireless Commun. **8**(3), 1223–1229 (2009)
18. Xiao, Y., Du, X., Zhang, J., Guizani, S.: Internet protocol television (IPTV): the killer application for the next generation internet. IEEE Commun. Mag. **45**(11), 126–134 (2007)
19. Du, X., Chen, H.H.: Security in wireless sensor networks. IEEE Wirel. Commun. Mag. **15**(4), 60–66 (2008)

20. Du, X., Guizani, M., Xiao, Y., Chen, H.H.: Secure and efficient time synchronization in heterogeneous sensor networks. IEEE Trans. Veh. Technol. **57**(4), 2387–2394 (2008)
21. Munoz, O., Pascual-Iserte, A., Vidal, J.: Optimization of radio and computational resources for energy efficiency in latency-constrained application offloading. IEEE Trans. Veh. Technol. **64**(10), 4738–4755 (2015)
22. Liu, Y., Tan, X., et al.: Research on spectrum allocation algorithms based on game theory in cognitive radio networks, Harbin Institute of Technology
23. TR25 G. 996, 3GPP SCM channel models, 3GPP TR25.996, vol. v6.1 (2003)