Hindawi
Mathematical Problems in Engineering
Volume 2020, Article ID 5391479, 12 pages
https://doi.org/10.1155/2020/5391479

Research Article

An Adaptive Task Scheduling Method for Networked UAV Combat Cloud System Based on Virtual Machine and Task Migration

Bo Li,[1] **Shiyang Liang**,[1] **Linyu Tian**,[1] **Daqing Chen**,[2] **and Ming Zhang**[1]

[1] *School of Electronics and Information, Northwestern Polytechnical University, Xi'an 710072, China*
[2] *School of Engineering, London South Bank University, London SE1 0AA, UK*

Correspondence should be addressed to Bo Li; libo803@nwpu.edu.cn

Received 21 October 2019; Revised 16 January 2020; Accepted 11 February 2020; Published 28 April 2020

Academic Editor: Haiyan Lu

This paper presents a systematic work aiming to improve the efficiency of task processing in a networked UAV combat cloud system. The work consists of three major aspects: (1) an architecture of UAV combat cloud systems—such a system provides the necessary resource pool for powerful computing and storage facilities and defines the attributes of the entities in the resource pool in detail; (2) an online adaptive task redistribution and scheduling algorithm—the algorithm involves task migration being performed on virtual machines on the cloud system and aims to address the problems caused by static task scheduling approaches; and (3) an online virtual machine and task migration algorithm—the algorithm considers collectively the priority type and quantity of the tasks to be migrated on virtual machines along with time constraints to determine the migration of virtual machine or task and optimize resource usages. Experimental simulation results have demonstrated that the proposed system and the relevant algorithms can significantly improve the efficiency of task schedule.

1. Introduction

In the modern information warfare, UAVs (unmanned aerial vehicles) are playing an increasingly important role, and now the major trends in the development of UAV systems are intelligence, integration, and networking [1]. Cloud computing [2] has become one of the disruptive technologies that is reshaping IT infrastructures, network services, and various applications. The Internet that is powered by cloud computing technology is rapidly developing. One of the prominent features of this network is its super computational power and massive data storage capabilities [3, 4]. In order to take advantages of cloud computing to maximize UAVs' effectiveness in cooperative mission [5, 6], a new research was conducted on integrating cloud computing with a UAV system [7]. Such a cloud-based system can connect combat units scattered in an environment through a network to form a cloud resource pool. This cloud resource pool stores all essential resources required on the battlefield, such as the number of weapons, the position of airplane, and the target damage, and therefore provides resources to the distant UAVs or missiles in the form of services. However, to make such a system functional effectively in a real environment such as in a combat environment, there are some crucial issues that need to be addressed. Among them, reasonable tasks scheduling [8] and the improvement of the efficiency of combat cloud task processing are considered prominent in the research.

After considering the combat units in a battlefield, this paper designs a networked UAV combat cloud system, aiming at providing a feasible method for a combination of cloud computing and UAV cooperative combat. In order to resolve the problems mentioned above, this paper first proposes an online adaptive task redistribution mechanism to alleviate the scramble for resources caused by difference in task priority. Then, considering the diversity of UAV types and tasks in actual combat, this paper analyzes combat tasks and proposes a virtual machine and task migration algorithm based on the fine-grained task types, aiming to avoid excessive migration of virtual machines, improve the task processing efficiency, and make the system more suitable for real-world combat environment. Finally, a set of experimental simulations have been undertaken which demonstrate that the proposed method is effective in improving the performance and efficiency of cloud combat system.

The reminder of this paper is organized as follows. Section 2 presents a literature review of related works in a comparative way to emphasize the major issues of the existing methods. Section 3 introduces and discusses the structure and resource entities of the proposed UAV combat cloud system. Section 4 explores the concept of task redistribution with examples, and further in Section 5, the proposed virtual machine and the task migration methods are presented with detailed procedure explained. The analysis experiments and the relevant results obtained are given in Section 6, and finally, the concluding remarks are summarized in Section 7.

2. Related Works

In the relevant research field about UAV cloud systems, literature [9] proposed a mixed QoS (quality of service) reassurance and energy-balanced architecture for UAV cloud system from a view of control theory. This UAV cloud system architecture intended to address the limitations of high complexity of UAV cluster and low intelligence of individual UAVs. Literature [10] introduced a new cloud-supported UAV application framework and a prototype system to implement the framework. The proposed method can be used to solve problems such as time delay in real-time data processing of UAV cloud. However, it only considers the use of UAVs for the perception of natural disasters and does not involve battlefield.

In order to improve the resource utilization and task processing efficiency of cloud computing platform, and to realize appropriate resource scheduling, many task scheduling algorithms and virtual machine migration schemes have been proposed in the field of cloud computing [11–15]. Literature [11] analyzed the impact of service-level agreement (SLA) default rate and power consumption of virtual machine migration on data center energy consumption and adopted virtual machine energy-saving scheduling algorithm based on energy consumption perception to reasonably deploy and migrate virtual machines. In [12], an improved particle swarm optimization algorithm was proposed to solve the task scheduling problem. Liu [13] added greedy algorithm to the resource threshold management strategy of physical machine so as to optimize the virtual machine selection algorithm and virtual machine placement algorithm. Li et al. [14] used a dual-threshold strategy, virtual machine selection strategy based on minimum migration cost, and probabilistic selection strategy of target physical node to reduce energy consumption. Chen et al. [15] proposed an online scheduling algorithm with uncertainty after considering the uncertainty of a scheduling environment, which ultimately improves the performance of cloud service platform. In terms of optimizing the migration timing and avoiding excessive migration of virtual machines, a lot of new research studies have been done based on some earlier methods [16]. The work discussed in [17] can dynamically adjust the virtual machine migration threshold through statistical analysis of physical machine historical load data. Yan et al. [18] modeled the selection of threshold as a Markov decision process, calculated the optimal threshold, and adjusted the threshold according to system performance.

However, the existing approach tends to be a single virtual machine migration without taking into account the types, characteristics, and resource requirements of the tasks handled by the virtual machines, which often leads to excessive migration and results in slower information processing and higher energy costs. Therefore, this paper proposes a virtual machine and task migration algorithm. By analyzing the types, characteristics, and combat missions of UAVs, the algorithm can reasonably determine the trigger time of virtual machine migration and task migration and select the target physical machine according to resource matching degree so as to finally reduce the probability of excessive migration of virtual machine. As such the algorithm can not only improve the efficiency of task processing but also make resource allocation more consistent with resource requirements and task characteristics.

3. Structure Design of Networked UAV Combat Cloud System

This section introduces the structure of the networked UAV combat cloud system designed in this paper and the attributes of resource entities that make up the cloud system.

3.1. System Structure. A networked UAV combat cloud system can realize resource sharing, information exchange, task assignment, and coordination on different platforms and effectively improve the capability of situational awareness, battlefield information acquisition, and interconnection. However, with the increased diversity of UAVs in terms of types and functionality, it will become increasingly difficult for a combat cloud network to meet the needs of information communication, data exchange, and coordinated attack among UAVs, resulting in a decline in the overall combat effectiveness. Therefore, in order to enable the combat cloud system to efficiently and flexibly handle task requests and scheduling [19, 20], this paper proposes a structure of a networked UAV combat cloud system as shown in Figure 1.

The cloud system consists of multiple UAV combat units in a battlefield environment. Each combat unit is treated as a node and eventually forms a network through data links. Virtually, all available resources can be put together into a huge resource pool, including physical entity resources, environmental resources, situational resources, and other resources. This resource pool will be managed and distributed uniformly to solve problems on the cloud network, such as snatching resources and virtual machine over-migration [21].

In Figure 1, communication type refers to the UAV with high communication capability, compute type refers to the UAV with powerful computing ability, data type refers to the

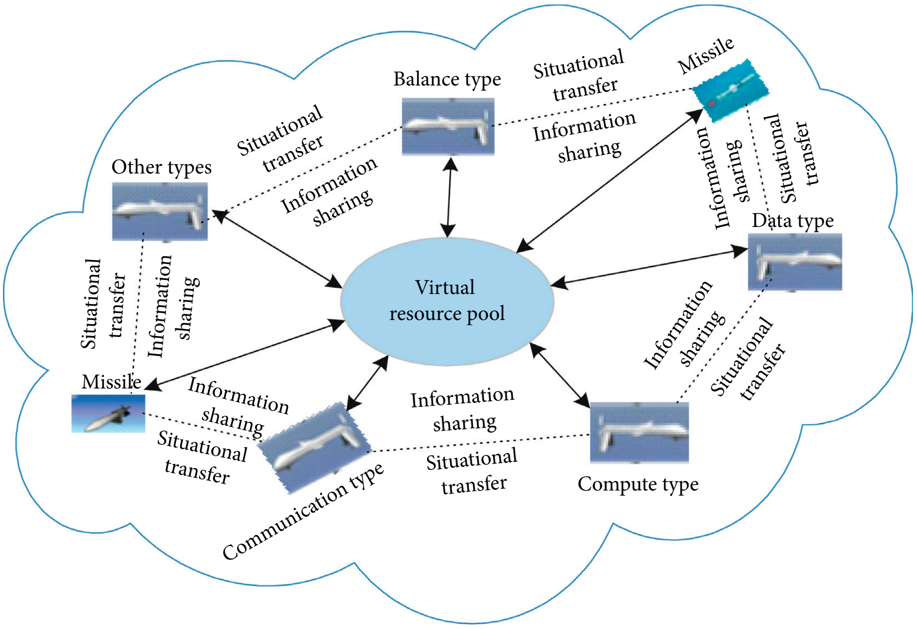


Figure 1: Structure of the proposed networked UAV combat cloud system.

UAV with super data processing capability, and balance type refers to the UAV with balanced ability.

3.2. Resource Entity Attributes. The following is an introduction of the attributes of the various resource entities that make up the cloud system.

(1) Physical machine (H): a processing unit in a data center. It is a basic physical entity for information processing of the entire cloud system.

(2) Virtual machine (VM): virtual computing unit. Based on the entity resources of the physical machine, it can provide complete hardware functions through virtualization technology.

(3) Data center: a resource collection that consists of various resources in the cloud system. It is responsible for controlling the physical and virtual machines, dynamically generating or destroying virtual machines based on tasks submitted by users, and deploying virtual machines on appropriate physical machines.

4. Adaptive Scheduling Strategies Based on Task Redistribution

This section first illustrates the time delay caused by resource grabbing between different priority tasks and then proposes an online adaptive scheduling strategy to make up for this deficiency.

4.1. Problem Description. An ideal task allocation algorithm can allocate different combat tasks to suitable virtual machines, which can balance the load of resources and improve system utilization. However, it is not guaranteed that each task can be completed within the allotted time. When there are a large number of high-priority tasks, they will be usually prioritized to access and use resources, while other low-priority tasks are put on hold. Therefore, such a single static task assignment algorithm [22–24] can cause a large number of low-priority tasks being denied accessing and using resources, resulting in extra wait time. Apparently, in a real battlefield environment, this can be a critical issue and can lead to deadly consequences.

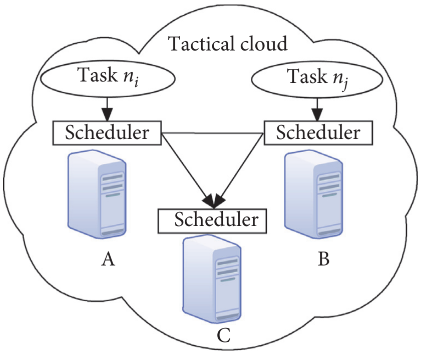
The following briefly describes the time delay caused by different priority tasks to access and use resources when scheduling using the static task allocation method.

Assume that a combat system has two priority tasks in total: Type A and Type B. Type A has a higher priority than Type B. The estimated time $\tau_{i,j}$ to complete task n_i in the virtual machine VM_j may be expressed as

$$\tau_{i,j} = \mathrm{ERAT}_{i,j} + \frac{S_I}{\mathrm{bw}} + \mathrm{ETM}_{i,j}, \quad (1)$$

where $\mathrm{ERAT}_{i,j}$ represents the earliest available time of resources, S_I is the size of the hard disk image for the task, bw denotes network bandwidth, S_I/bw is the task transmission time, and $\mathrm{ETM}_{i,j}$ represents the processing time of task n_i in virtual machine vm_j.

As shown in Figure 2, assume that a tactical cloud contains three combat nodes. The scheduler of node A needs to allocate a task n_i of Type B to the combat cloud. According to equation (1) and Figure 2, the scheduler of node A will transmit task n_i to node C because it has the shortest task completion time τ. After this, the scheduler of node B needs to submit a task n_j of Type A to the combat cloud, and the task requires system to reserve resources for n_j. If node C happens to have a minimum time of τ for the task, node B will transmit this task to node C. Assume that two tasks are allocated to the same virtual machine; task n_j will have priority to use the resources if task n_i is still incomplete. In this case, task n_i will be forced to suspend until task n_j is



Node	Task n_i			Task n_j		
	ERAT	S_I/bw	ETM$_i$	ERAT	S_I/bw	ETM$_j$
A	2	1	10	8	1	4
B	3	1	8	8	1	3
C	4	1	6	8	1	2

Figure 2: Task processing diagram of tactical cloud.

completed. Imagine that if there were many tasks of Type A requesting resources, then tasks of type B would have a long wait in the queue.

4.2. Online Adaptive Task Redistribution Mechanism. In order to alleviate the system delay caused by conflicting resources demands from tasks with different priorities and to overcome the drawbacks of the static task allocation algorithm, an online adaptive processing mechanism is proposed as discussed below.

Essentially, the mechanism will repeatedly evaluate the remaining available static resources and the completion of each task within a predefined time interval. In each evaluation, the scheduler on each combat node will recalculate the expected completion time of the tasks on the node. Note that scheduler will only reevaluate tasks that have already been committed to the node, not the tasks allocated to the node. Based on the updated information, the new estimated earliest completion time $\tau_{fdi,j}$ of task n_i in virtual machine VM$_j$ is given by

$$\tau_{fdi,j} = \text{ERAT\_updated}_{i,j} + \frac{S_I}{\text{bw}} + \text{ETM}_{i,j}, \qquad (2)$$

where ERAT_updated$_{i,j}$ denotes the earliest resource available time after updating.

The difference $D_{\tau_{i,j}}$ between the earliest completion time of initial estimate and the earliest completion time after the update is given by

$$D_{\tau_{i,j}} = \tau_{fdi,j} - \tau_{di,j}. \qquad (3)$$

When $D_{\tau_{i,j}}$ meets the criterion $D_{\tau_{i,j}} > D_{\text{threshold}}$, where $D_{\text{threshold}}$ is a predefined threshold, the scheduler will redistribute the tasks to a virtual machine that has the smallest $\tau_{fdi,j}$ with a predefined probability α (α refers to the task migration probability). In this paper, we assume that the scheduler reevaluates the remaining static resources before the next task n_k is assigned to the same virtual machine. Based on the latest information, the scheduler also computes $\tau_{fdk,j}$ of the task n_k. Once task n_k satisfies $D_{\tau_{i,j}} > D_{\text{threshold}}$, it will also be redistributed.

5. Method of Online Task Scheduling Based on Virtual Machine and Task Migration

This section begins with an introduction to the migration mechanism. On this basis, we further apply the existing virtual machine migration algorithm based on threshold (VMBTA) to the multilayer cloud architecture system. Finally, we propose an online virtual machine and task migration algorithm (VTMA) for combat cloud systems to improve the existing algorithms.

5.1. Virtual Machine Migration Mechanism. There are still some shortcomings in the method mentioned in Section 4. For example, when the number of tasks in a virtual machine reaches critical condition, tasks are allocated and redistributed more frequently, resulting in bigger energy consumption and lower efficiency. In this case, a new method is needed to tackle these issues.

The core technology of cloud computing is the virtualization technology [25], and one of the main advantages of virtualization is online migration. Migrating virtual machines can reduce the workload of physical machines and effectively balance resource allocation among physical machines [26]. In this work, we use dynamic migration as the migration mode of a virtual machine aiming to develop an adaptive decision-making method for the migration.

5.2. Virtual Machine Migration Algorithm Based on Threshold (VMBTA). When the utilization of a virtual machine reaches a specific threshold or the virtual machine needs more resources to complete its tasks, the virtual machine must migrate to a new physical machine to continue with current or new tasks. VMBTA [17, 18] is one of the most popular virtual machine migration methods to achieve load balance, reduce system load [16], and improve efficiency of the system. This method generally considers three factors when designing a mechanism for virtual machine migration.

(1) Triggering timing of virtual machine migration, that is, when the load of physical machines exceeds the threshold, the virtual machines in physical machine will be triggered to migrate.

(2) Choosing the source virtual machine, that is, selecting a virtual machine that meets conditions of migration as the source virtual machine.

(3) Choosing the target physical machine, that is selecting a physical machine (other than the physical machine where the source virtual machine resides)

for the purpose of migrating the source virtual machine to that physical machine.

According to the traditional threshold-based virtual machine migration theory and the specific situation of using the task redistribution scheduling strategy, the VMBTA adopted to the cloud structure system is given. The following describes the variables involved in the VMBTA.

Let $H = \{H_1, H_2, \ldots, H_n\}$ denote the physical machine collection in combat cloud. The resources collection of a physical machine (H_i) is $R_i = \{R_i^1, R_i^2, \ldots, R_i^p\}$, where p represents the number of resource types in the physical machine and R_i^p represents the amount of a resource in physical machine H_i, such as the number of CPUs, size of the memory, network bandwidth, and storage capacity of a hard drive. The virtual machine collection in physical machine H_i is $\mathrm{VM}_i = \{\mathrm{VM}_{i1}, \mathrm{VM}_{i2}, \ldots, \mathrm{VM}_{in}\}$, and the virtual machine collection in combat cloud is $\mathrm{VM} = \{\mathrm{VM}_1, \mathrm{VM}_2, \ldots, \mathrm{VM}_n\}$.

Definition 1. Utilization of the p^{th} resource in physical machine H_i:

$$\mathrm{Ava}R_i^p = \frac{R_{i\mathrm{used}}^p}{R_{i\mathrm{total}}^p}, \tag{4}$$

where R_{iused}^p represents current usage amount of the p^{th} resource in physical machine H_i and R_{itotal}^p denotes the total amount of the p^{th} resource in physical machine H_i.

Definition 2. Comprehensive utilization of resources in physical machine H_i:

$$\mathrm{Ava}R_i = \frac{1}{n}\sum_{p=1}^{n} \mathrm{Ava}R_i^p, \tag{5}$$

where n is the total amount of resource types in physical machine H_i and $\mathrm{Ava}R_i^p$ is the utilization of the p^{th} resource in physical machine H_i.

Definition 3. The resource proportion of virtual machine V_{ij} in physical machine H_i:

$$\mathrm{Pro}_{ji} = \frac{1}{n}\sum_{p=1}^{n} \frac{Q_{ij}^p}{R_i^p}, \tag{6}$$

where n is the number of resource types in a virtual machine, Q_{ij}^p is the total amount of the p^{th} resource in virtual machine V_{ij}, and R_i^p represents the total amount of the p^{th} resource in physical machine H_i.

We define three basic factors of the VMBTA as follows:

(1) Triggering timing of virtual machine migration: the comprehensive resource utilization of the physical machines in UAV combat cloud is used as the evaluation criterion of the virtual machine migration threshold. We set the threshold to φ. When the comprehensive utilization of a physical machine H_i in combat cloud satisfies $\mathrm{AvaR}_i > \phi$, the VMBTA will be triggered.

(2) Choosing the source virtual machine: the algorithm selects a virtual machine that occupies the largest proportion of the total resources in the source physical machine as the source virtual machine. If the source virtual machine fails to be migrated, select the virtual machine that occupies the second largest proportion of total resources as the source virtual machine and so on.

(3) Choosing the target physical machine: the algorithm selects a physical machine with the lowest utilization rate in combat cloud as the target physical machine. If the source virtual machine fails to be migrated, select the physical machine with the second lowest utilization rate as the target physical machine and so on.

Figure 3 shows the flowchart of the VMBTA. Since this algorithm is not the research content of this paper, only the flowchart is presented here.

5.3. Online Virtual Machine and Task Migration Algorithm (VTMA).

Each new method could be a good solution to the problem when it was applied at the beginning. However, problems change constantly with the development of technology and the increasing demand and the changes in the application field. Therefore, methods would need to be improved in practice.

The VMBTA has several problems in practice. This algorithm only considers a single performance indicator the trigger criterion for virtual machine migration and does not consider the various types and characteristics of tasks in the system. Therefore, this section will first refine the type of tasks and categorize tasks into four categories according to the main needs of the actual battlefield environment:

(1) Data-intensive tasks: it refers to a task of providing information processing services, such as data preprocessing and information fusion for a large amount of raw data collected by sensors or data that needs to be classified and summarized from various combat nodes.

(2) Computation-intensive tasks: it refers to the task that requires the combat system to provide high-speed and efficient computing services and a large number of computing tools in the combat process such as target identification and fire control calculation.

(3) Communication-intensive tasks: it refers to the task of information transmission and interactions between combat units, which requires high communication bandwidth and time sensitivity.

(4) Load-balancing tasks: it refers to the task that has a balancing demand for computation, storage and communications, and other resources.

In general, the transmission and migration of data-intensive tasks should be minimized because of their own characteristics with large amount of data, a wide range of categories, and nonstandardization. Computation-intensive tasks should be allocated to a physical node that has strong

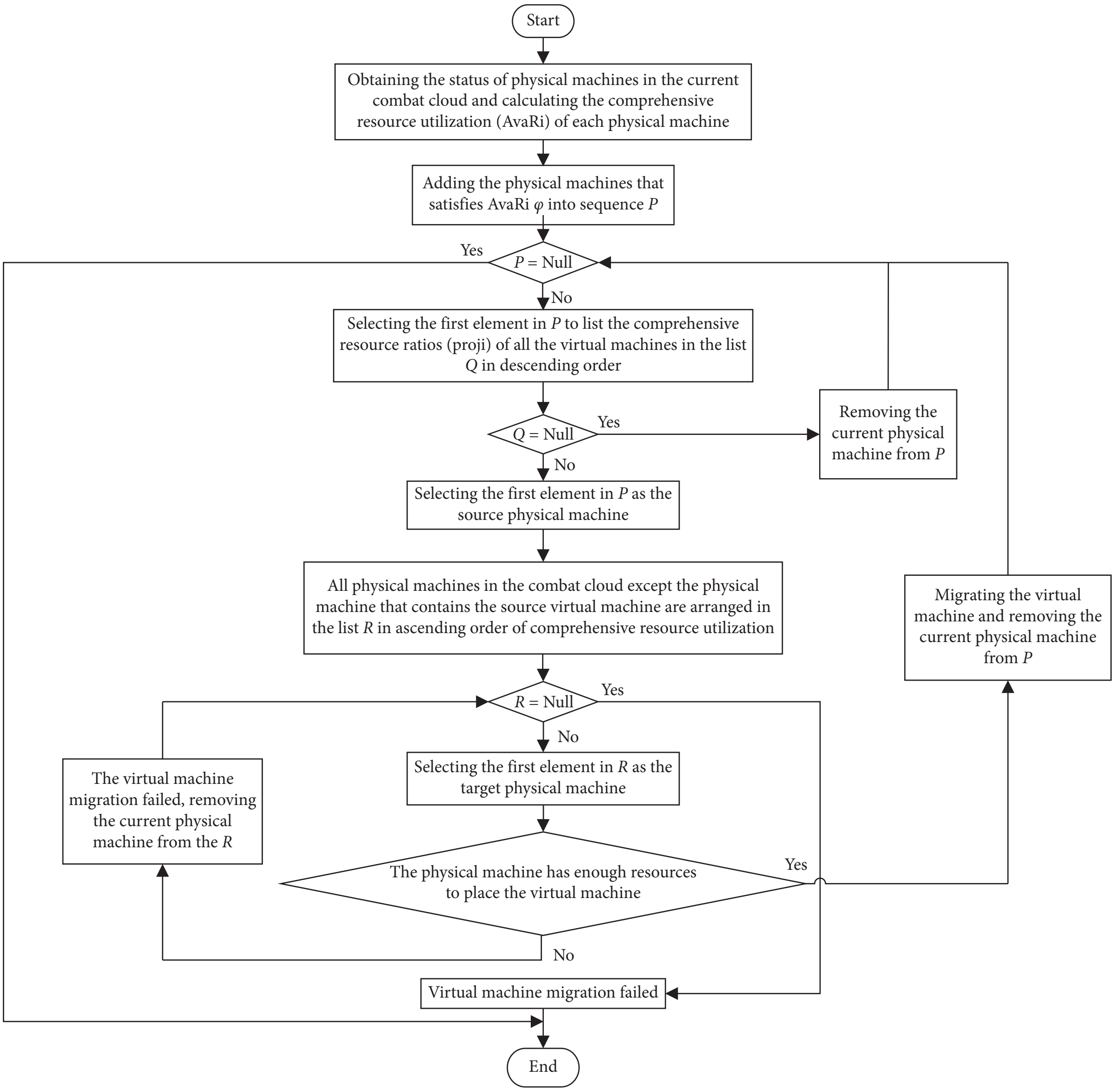

Start
Obtaining the status of physical machines in the current combat cloud and calculating the comprehensive resource utilization (AvaRi) of each physical machine
Adding the physical machines that satisfies AvaRi φ into sequence P
Yes
P = Null
No
Selecting the first element in P to list the comprehensive resource ratios (proji) of all the virtual machines in the list Q in descending order
Q = Null
Yes
Removing the current physical machine from P
No
Selecting the first element in P as the source physical machine
All physical machines in the combat cloud except the physical machine that contains the source virtual machine are arranged in the list R in ascending order of comprehensive resource utilization
Migrating the virtual machine and removing the current physical machine from P
R = Null
Yes
No
The virtual machine migration failed, removing the current physical machine from the R
Selecting the first element in R as the target physical machine
The physical machine has enough resources to place the virtual machine
Yes
No
Virtual machine migration failed
End

Figure 3: Flowchart of the VMBTA.

computing power for processing. However, the VMBTA cannot address these issues because it can cause unnecessary migration of virtual machines and increase the energy consumption of the cloud system. As such this section proposes the VTMA based on VMBTA.

The purpose of the VTMA is to reduce the overmigration of virtual machines and to make resource allocation meet resource demand and task characteristics more effectively. When most of the tasks running on the source virtual machine are data-intensive tasks, the workload of a virtual machine can be reduced by migrating non-data-intensive tasks to other virtual machines, thereby reducing system delay and energy consumption due to virtual machine migration. The related variables involved in the VTMA are described as follows: migration probability threshold of virtual machines is Φ, and the demand resource vector of task n_k is $N_k = \{N_k^1, N_k^2, \ldots, N_k^p\}$, where p is the number of categories of demand resource and $Q_{ij} = \{Q_{ij}^1, Q_{ij}^2, \ldots, Q_{ij}^p\}$ is a demand resource vector of the virtual machine VM_{ij}.

Definition 4. Migration probability of source virtual machine VM_{ij}.

$$\beta_{ij} = 1 - \frac{n_{\mathrm{data}}}{n_{\mathrm{total}}}, \tag{7}$$

where n_{data} is the number of data-intensive tasks in virtual machine VM_{ij} and n_{total} is number of entire tasks in virtual machine VM_{ij}.

Equation (7) indicates a negative correlation between the number of data-intensive tasks in the source virtual machine and the migration probability of the source virtual machine.

Definition 5. Degree of resource matching between the task n_k and the virtual machine VM_{ij}.

$$\sigma_{k,ij} = \sqrt{\frac{1}{n}\sum_{p=1}^{n}\left(\frac{N_k^p}{Q_{ij\mathrm{rest}}^p} - \frac{1}{n}\sum_{p=1}^{n}\frac{N_k^p}{Q_{ij\mathrm{rest}}^p}\right)^2}, \tag{8}$$

where n is the number of resource types in virtual machine VM_{ij}, $Q_{ij\mathrm{rest}}^p$ is the surplus of the p^{th} resource in virtual machine VM_{ij}, and N_k^p is the demand of the p^{th} resource in task n_k.

Definition 6. The comprehensive resource proportion of task n_k in virtual machine VM_{ij}.

$$\mathrm{Pro}_{k,ij} = \frac{1}{n}\sum_{p=1}^{n}\frac{N_k^p}{Q_{ij}^p}, \tag{9}$$

where n is the number of resource types in the virtual machine VM_{ij}, p is the number of resource types in the virtual machine, Q_{ij}^p is the total amount of the p^{th} resource in virtual machine V_{ij}, and N_k^p is the demand of the p^{th} resource in task n_k.

Algorithm 1 shows the algorithm of virtual machine and task migration.

The aim of this algorithm is to reduce the excessive migration of virtual machines and improve the efficiency of task processing. The input to the algorithm is a list of source virtual machines selected using the VMBTA and is stored in list P (lines 1–2). Firstly, the algorithm gets the first virtual machine in P and the tasks running on it. After that, the algorithm classifies these tasks with a classifier (such as neural networks) and marks the types of the tasks. Therefore, the migration probability of the source virtual machine can be calculated according equation (7) (Lines 3–6). If the migration probability is greater than the threshold Φ, the VMBTA for virtual machine migration will be used (Lines 7–9). Otherwise, the algorithm calculates the comprehensive resource proportion of non-data-intensive tasks in the virtual machine according to equation (9) and ranks the tasks in descending order to Q according to this proportion (Lines 10–12). While $Q \neq \varnothing$, on the entire combat cloud, except the virtual machine containing task n_k, the algorithm will select the virtual machine VM_{mn} that has the highest matching degree with task n_k as target virtual machine (Lines 13–15). While $R \neq \varnothing$, if the remaining resources of the VM_{mn} are not enough for task n_k, remove the VM_{mn} from R and select the new VM_{mn} from R for judgment (Lines 16–18). Otherwise, the migration of n_k is successful and the algorithm clears the R and Q to end its while loop (Lines 19–22). The while loop of R ends when migration is successful or the $R = \varnothing$ (Lines 16–24). The while loop of Q ends when migration is successful or the $Q = \varnothing$ (Lines 13–26). The algorithm ends when all the source virtual machines complete the virtual machine or task migration (Lines 3–29).

6. Simulation and Analysis

This section carries out two experiments to verify the optimization effect of online adaptive task redistribution mechanism and VTMA, respectively.

6.1. Experimental Design

6.1.1. Experimental Scenario. This section provides experimental simulations to demonstrate the adaptive process for task scheduling using the proposed algorithm in networked UAV combat cloud system. This process includes search, identification, analysis, fire attack, and final damage assessment of ground targets within a designated area. The combat cloud consists of 7 UAVs in total: 3 fire UAVs, 2 storage UAVs, and 2 communication UAVs.

6.1.2. Experimental Environment. The simulations were executed on Windows7 using CloudSim simulation package running on Java platform. The simulation environment is shown in Table 1.

6.1.3. Experimental Parameter Settings. The combat cloud consists of a variety of heterogeneous UAVs and the resources they carry on. Different UAVs have different capabilities for processing tasks, and we use three categories of physical machine parameters to characterize different types of UAVs. They are storage-type, compute-type, and communication-type physical machines, respectively. The range of the parameter values of the physical machines is shown in Table 2.

The system will generate different virtual machines according to the tasks. Table 3 shows the value range of the parameters of the virtual machines.

Table 4 shows the value range of the main parameters of the tasks in service request submitted by users.

6.2. Verification Experiment for Task Redistribution

6.2.1. Experiment Design. This experiment testified the impact of the adaptive task redistribution method on the assignment and processing efficiency of combat cloud when the networked UAV combat cloud received a large number of tasks over a period of time.

In the experiment, a data center of combat cloud that consists of 7 UAVs was generated. This center accepted 20 service requests (such as target searching, target recognizing, and situation assessing) randomly generated by the users, of which 4 service requests (such as threat assessment, fire control solution, and other service requests) were Type A, and the rest were Type B. Each service request contained 10–20 tasks. The service requests were submitted in two different ways in the experiment:

(1) Service requests are evenly distributed on the timeline in order to be submitted orderly so that each service request has sufficient time and resources to be processed, and tasks do not have to compete for resources with each other in most cases. We call this situation the loose service request mode.

(2) All service requests are submitted at very short intervals, which means that tasks will have to compete

```
Input: The VMs to Migration
Ensure: Reduce the excessive migration
Build a new list $P = \emptyset$
Put the VMs into the list $P$
While $P \neq \emptyset$ do
    Get the first VM ($\mathrm{VM}_{ij}$) in $P$ and get the tasks running on it
    Recognize the classification of each task
    Compute the $\beta_{ij}$
    If $\beta_{ij} > \Phi$then
        Run the VMBTA
        Remove the first VM ($\mathrm{VM}_{ij}$) from $P$
    Else do
        Build a new list $Q$
        Evaluate the non-data-intensive tasks' $\mathrm{Pro}_{k,ij}$ and list them with descending order $\longrightarrow Q$
        While $Q=\emptyset$
            Build a new list $R=\emptyset$
                Evaluate the first task's ($n_k$'s) $\sigma_{k,mn}$ in $Q$ with all other VMs and list them with descending order $\longrightarrow R$
            While $R\neq\emptyset$ do
                If the remaining resources of first VM ($\mathrm{VM}_{mn}$) in $R$ are not enough for $n_k$ then
                    Remove $\mathrm{VM}_{mn}$ from $R$
                Else do
                    Migrate $n_k$ to $\mathrm{VM}_{mn}$
                    Clear $R$
                    Clear $Q$
                End If
            End while
            Remove the first task from $Q$
        End While
        Remove the first VM ($\mathrm{VM}_{ij}$) from $P$
    End If
End whiles
```

Algorithm 1: VM-task migration algorithm (VTMA).

Table 1: Experimental environment for simulation.

Experimental environment	Parameter
Operating system	Windows 7 Ultimate
CPU	Intel (R) Core (TM) i3-2100 CPU @ 3.10 GHZ
RAM	2.00 GB
System type	32-bit operating system
Java version	1.8.0_45
Eclipse version	4.4.2
CloudSim version	3.0.2

Table 2: The value range of parameters of the physical machines.

Parameter	Min	Max
CPU cores	8	16
CPU clock speed (Mips)	5000	10000
RAM (MB)	$8 * 1024$	$16 * 1024$
Hard drive capacity (MB)	$400 * 1024$	$800 * 1024$
Network bandwidth (MB/s)	$8 * 1024$	$16 * 1024$

Table 3: The value range of parameters of the virtual machines.

Parameter	Min	Max
CPU cores	1	2
CPU clock speed (Mips)	5000	10000
RAM (MB)	512	1024
Hard drive capacity (MB)	$50 * 1024$	$100 * 1024$
Network bandwidth (MB/s)	1024	$2 * 1024$

for resources from each other and some tasks will have to wait for resources. We call this situation the compact service request mode.

In both the forms of submitting task, by changing the task redistribution probability α, the average completion time of service requests in combat cloud was used to evaluate

Table 4: The value range of the main parameters of the service requests.

Parameter	Min	Max
Number of tasks	10	20
UNITS	2000	10000
Demand for CPU cores	1	2

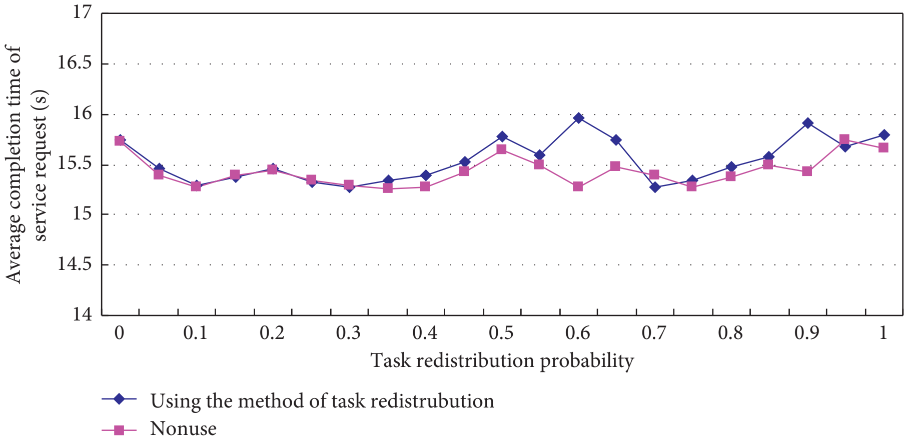


Figure 4: Average completion time of requests in loose service request mode.

the effect of the proposed online adaptive mechanism. The completion time is defined as the time taken from receiving the service request to successfully complete the service request. Due to the randomness of generating service requests, the experiments of both service request modes were conducted 100 times, and the average value was calculated as the test result.

6.2.2. Experimental Results and Analysis. Figure 4 shows the average completion time of requests in loose service request mode.

As shown in Figure 4, the online adaptive mechanism for task scheduling has no obvious effect on the processing time of services. In the loose service request mode, the phenomenon of resource contention is not obvious, and each task has enough resources to be processed in most cases. Therefore, in most cases, task reassignment was not required, which limited the effect of the adaptive mechanism for task redistribution.

Figure 5 shows the average completion time of requests in compact service request mode.

As shown in Figure 5, the online adaptive mechanism for task scheduling has a significant impact on the processing time of services. When the task reallocation probability α was in the range of (0, 0.75), the average completion time of service requests decreased notably with the increase of the task reallocation probability. When the task reallocation probability α was in the range of (0.75, 1), the average time to process services did not change significantly. In the compact service request mode, competing for resources is very frequent, and the actual completion time of tasks is usually later than the estimated completion time. For a service request consisting of multiple tasks, the delay of the task processing will be worsened. Therefore, when the task reallocation probability α was in the range of (0, 0.75), the average completion time of service requests decreased significantly as α increases. However, some tasks of Type A did not take up too many resources all the time. When the task reallocation probability α continued to increase, almost all tasks that had been robbed of resources were reallocated, which made the task allocation algorithm reused a lot and ultimately affected the efficiency of the system. As a result, in the experiments, when the task reallocation probability α was in the range of (0.75, 1), the average completion time of service requests changed little and even showed the slight increase.

6.3. Verification Experiment for Virtual Machine and Task Migration

6.3.1. Experiment Design. The experiment simulated the impact of using the VTMA on allocating tasks and processing efficiency of combat cloud. Before the experiment began, the VMBTA was used for simulation test several times. In the simulation test, the probability threshold φ of virtual machine migration in the VMBTA increased from 0% to 100% with a stepped increment of 10%. The test showed that the task processing efficiency was the highest in combat cloud when setting φ to 80%, so this experiment set φ to 80%.

In the experiment, the data center composed of seven physical machines was generated and accepted 20 service requests submitted randomly by the users, of which four service requests were Type A and the rest were Type B, and each service request contained 10–20 tasks. In the loose service request mode, the combat cloud has a low overall load and it is difficult to evaluate the performance of the

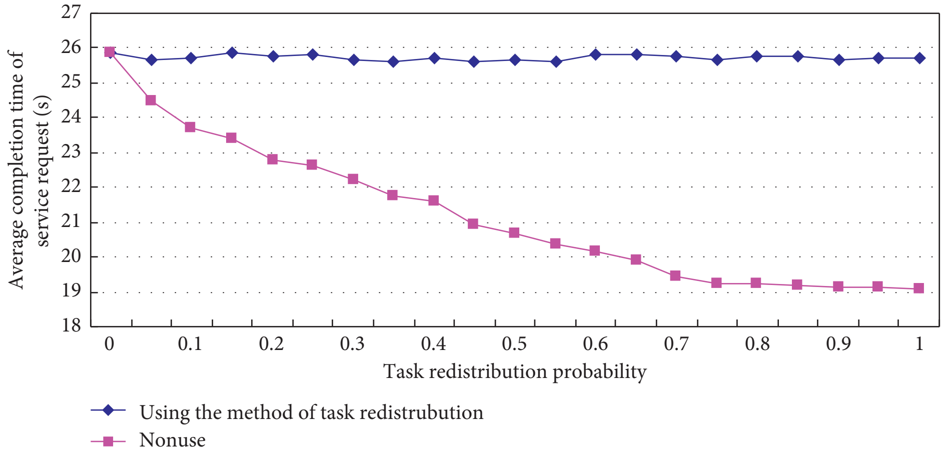


Figure 5: Average completion time of requests in compact service request mode.

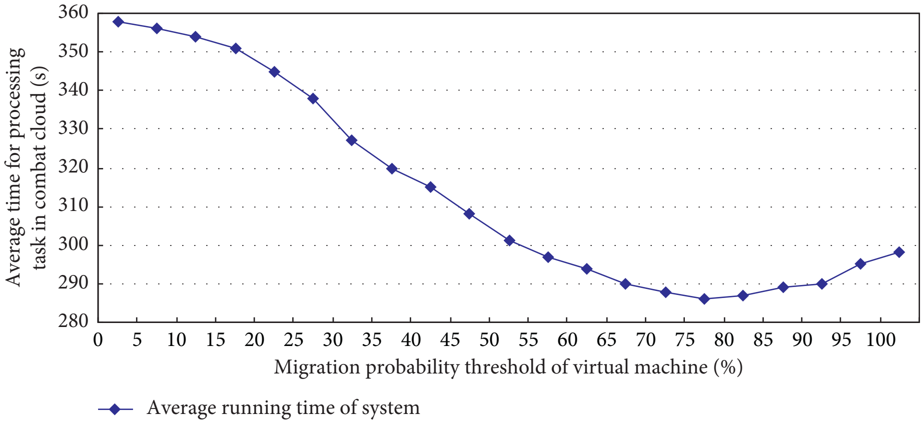


Figure 6: The time of processing tasks in network combat cloud of UAV.

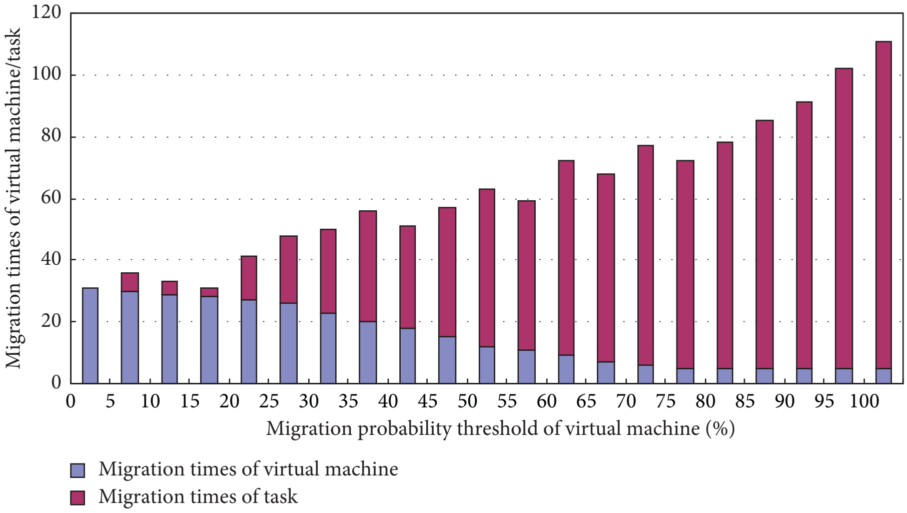


Figure 7: Migration condition of virtual machine and task with different migration thresholds.

VTMA. Therefore, this experiment used the compact service request mode to prove the effectiveness of the VTMA. The experiment evaluated the impact of using the VTMA on the task processing time and the migration of virtual machines and tasks by changing the migration probability threshold Φ of virtual machine in the VTMA. The experiments were simulated 100 times and took the average as the experiment result because of the randomness of generating service requests.

6.3.2. Experimental Results and Analysis. Figure 6 shows the time for processing tasks in UAV combat cloud when using the VTMA.

It is evident from Figure 6 that using the VTMA improved the efficiency of processing tasks in combat cloud when using the VMBTA for adaptive task scheduling, which reduced the time of processing tasks by up to 20.1%. When the migration probability was 75%, the system had the minimum task processing time. When the migration probability was 80% to 100%, the reason for the increase in processing time is the excessive number of task migrations.

Figure 7 shows the migration times of virtual machines and tasks with different thresholds of migration probability Φ when using the VTMA.

When the migration probability threshold Φ of a virtual machine was 0%, the triggering condition of the VTMA cannot be met, so the experiment process was equivalent to using the VMBTA only. As shown in Figure 7, when the migration probability was 75%, the VTMA reduced the number of the virtual machine migrations by 80%. When Φ was in (0%, 20%), only the number of task migrations decreased, so the decrease of processing time was not obvious. When the Φ was in (20%, 60%), the number of virtual machine migrations significantly reduced due to efficient task migration, which has led to a significant decrease of task processing time. When Φ was in (60%, 75%), the number of virtual machine migrations slowly decreased, resulting in a slow decline in task processing time in the combat cloud. When Φ was in (75%, 100%), task processing time has increased because the number of virtual machine migrations no longer decreases, and the number of task migrations significantly increases. It implies that this situation has led to unnecessary task migrations, leading to an increase in task completion time.

7. Conclusions

This paper first proposes a cloud combat system which provides a new idea for the future network-centric combat mode. Then, by analyzing the assignment process of a single static method, an adaptive task redistribution mechanism is established. This mechanism solves the problem that the system processing time exceeds the specified time and improves the efficiency of task assignment of cloud system. Finally, in order to further improve the efficiency of task processing in the cloud system, we have refined the different characteristics of the tasks and established a virtual machine and task migration mechanism. This mechanism determines whether a virtual machine/task is migrated or not based on the type and the number of tasks in the virtual machine, thus reducing excessive energy consumption caused by the migration of the virtual machine. The simulation results have shown that the migration mechanism increases the flexibility of configuration resources and therefore improves the efficiency of processing tasks in the networked UAV combat cloud system.

Data Availability

The data used to support the findings of this study are included within the article.

Conflicts of Interest

The authors declare that there are no conflicts of interest.

Acknowledgments

This research was partially funded by the Aeronautical Science Foundation of China under grant no. 2017ZC53021, the Seed Foundation of Innovation and Creation for Graduate Students in Northwestern Polytechnical University under grant no. ZZ2019159, and the Open Project Fund of CETC Key Laboratory of Data Link Technology under grant no. CLDL-20182101.

References

[1] F. Wanf, H. R. Zhou, and J. Hao, “Function architecture of UAV CGCS supporting network application,” *Electronics Optics & Control*, vol. 21, no. 2, pp. 50–54, 2014.

[2] S. K. Garg, S. Versteeg, and R. Buyya, “A framework for ranking of cloud computing services,” *Future Generation Computer Systems*, vol. 29, no. 4, pp. 1012–1023, 2013.

[3] Y. H. Ni, G. X. Ni, and X. W. Sun, “Research on the air formation collaborative warfare based on cloud computing,” *Electronic Measurement Technology*, vol. 40, no. 3, pp. 15–19, 2017.

[4] K. Zaerens, “Enabling the benefits of cloud computing in a military context,” in *Proceedings of the Services Computing Conference*, Jeju, South Korea, December 2011.

[5] M. V. Shenoy and K. R. Anupama, “Dtta—distributed, time-division multiple access based task allocation framework for swarm robots,” *Defence Science Journal*, vol. 67, no. 3, pp. 316–324, 2017.

[6] R. A. Raji and A. P. R. Kumar, “Advanced mission management system for unmanned aerial vehicles,” *Defence Science Journal*, vol. 64, no. 5, pp. 438–444, 2014.

[7] M. Cochez, J. Periauk, and V. Terziyan, “Evolutionary cloud for cooperative UAV coordination,” *Software and Computational Engineering*, vol. 52, pp. 16–23, 2014.

[8] D. Y. Deng, “Computer engineer: research of energy efficiency optimization-oriented dynamic migration technology for virtual machine in cloud computing,” 2016.

[9] A. Gao, Y. Hu, L. Li, and X. Li, “BP network control for resource allocation and QoS ensurance in UAV cloud,” *Journal of Sensors*, vol. 2018, no. 7, pp. 1–14, 2018.

[10] C. Luo, J. Nightingale, E. Asemota et al., “A UAV-cloud system for disaster sensing application,” in *Proceedings of the*

Vehicular Technology Conference, Boston, MA, USA, May 2015.

[11] Y. Zheng, "Research on the improvement of energy-saving scheduling algorithm for cloud computing data center," *Journal of Southwest University (Natural Science Edition)*, vol. 41, no. 12, pp. 135–142, 2019.

[12] D. Wu, "Cloud computing task scheduling policy based on improved particle swarm optimization," in *Proceedings of the 2018 International Conference on Virtual Reality and Intelligent Systems (ICVRIS)*, Changsha, China, May 2018.

[13] K. Liu, "Virtual machine migration strategy based on greedy algorithm for cloud data center," *Computer Applications*, vol. 39, no. 11, pp. 3333–3338, 2019.

[14] F. B. Li, H. Q. Yu, and G. S. Fan, "Virtual machine dynamic migration algorithm based on energy reduction," *Journal of East China University of Science and Technology (Natural Science Edition)*, vol. 43, no. 5, pp. 692–697, 2017.

[15] H. K. Chen, X. M. Zhu, G. P. Liu et al., "Uncertainty-aware online scheduling for real-time workflows in cloud service environment," *IEEE Transactions on Services Computing*, vol. 99, p. 1, 2018.

[16] R. Razali, R. Rahman, and M. Samad, "Virtual machine migration implementation in load balancing for cloud computing," *International Conference on Intelligent & Advanced Systems*, vol. 33, pp. 1–4, 2014.

[17] C. Zhao, L. S. Yan, and Y. H. Cui, "Dynamic adjusting threshold algorithm for virtual machine migration," *Journal of Computer Applications*, vol. 37, no. 9, pp. 2547–2550, 2017.

[18] C. Y. Yan, Z. H. Li, and X. R. Yu, "Virtual machine dynamic consolidation method based on adaptive overloaded threshold selection," *Journal of Computer Application*, vol. 36, no. 10, pp. 2698–2704, 2016.

[19] R. Taylor, "Pre-emptive scheduling of on-line real time services with task migration for cloud computing," in *Proceedings of the International Conference on Pattern Recognition*, Barcelona, Spain, February 2013.

[20] S. Selvarani and G. S. Sadhasivam, "Improved cost-based algorithm for task scheduling in cloud computing," *IEEE International Conference on Computational Intelligence & Computing Research*, vol. 28, pp. 1–5, 2011.

[21] K. Goztepe, "A decision framework for combat cloud computing strategy," *International Conference on Information Security & Cryptology*, vol. 39, pp. 261–273, 2013.

[22] Q. B. Nie, "Optimization management of task scheduling for cloud resource load balance," *Computer Engineering and Design*, vol. 38, no. 1, pp. 18–22, 2017.

[23] C. L. Wang, Y. G. Tao, and P. Yang, "Parallel task assignment optimization algorithm and parallel control for cloud control systems," *Acta Automatica Sinica*, vol. 43, no. 11, pp. 1973–1983, 2017.

[24] L. He, J. Rao, and F. Q. Zhao, "Task scheduling method based on energy optimization in cloud computing system," *Computer Engineering and Applications*, vol. 49, no. 20, pp. 19–22, 2013.

[25] P. D. Patel, M. Karamta, and M. D. Bhavsar, "Live virtual machine migration techniques in cloud computing: a survey," *International Journal of Computer Applications*, vol. 86, no. 16, pp. 18–21, 2014.

[26] X. Y. Wu, "Geology: research on virtual machine migration mechanism based on lord forecast in cloud computing," 2015.