Hindawi
Discrete Dynamics in Nature and Society
Volume 2020, Article ID 8105145, 13 pages
https://doi.org/10.1155/2020/8105145

Research Article

Fuzzy Theory-Based Data Placement for Scientific Workflows in Hybrid Cloud Environments

Zheyi Chen,[1] Xu Zhao,[2] and Bing Lin[3]

[1]*College of Engineering Mathematics and Physical Sciences, University of Exeter, Exeter EX4 4QF, UK*
[2]*Fujian Provincial Key Laboratory of Network Computing and Intelligent Information Processing, Fuzhou 350118, China*
[3]*College of Physics and Energy, Fujian Normal University, Fuzhou 350117, China*

Correspondence should be addressed to Bing Lin; wheellx@163.com

Received 11 June 2020; Accepted 28 July 2020; Published 17 August 2020

Guest Editor: Chi-Hua Chen

In hybrid cloud environments, reasonable data placement strategies are critical to the efficient execution of scientific workflows. Due to various loads, bandwidth fluctuations, and network congestions between different data centers as well as the dynamics of hybrid cloud environments, the data transmission time is uncertain. Thus, it poses huge challenges to the efficient data placement for scientific workflows. However, most of the traditional solutions for data placement focus on deterministic cloud environments, which lead to the excessive data transmission time of scientific workflows. To address this problem, we propose an adaptive discrete particle swarm optimization algorithm based on the fuzzy theory and genetic algorithm operators (DPSO-FGA) to minimize the fuzzy data transmission time of scientific workflows. The DPSO-FGA can rationally place the scientific workflow data while meeting the requirements of data privacy and the capacity limitations of data centers. Simulation results show that the DPSO-FGA can effectively reduce the fuzzy data transmission time of scientific workflows in hybrid cloud environments.

1. Introduction

With the widespread applications of Big Data technologies, the amount of data generated by modern network environments is greatly increasing. Therefore, traditional distributed computing modes such as grid computing may not meet the requirements of massive data processing. In recent years, cloud computing has emerged as a research hotspot [1–5], where hybrid cloud environments show the advantages of high sharing, high availability, and customization. Specifically, the hybrid cloud environments are composed of the data centers distributed in different geographical locations, including multiple private and public data centers [6]. On the one hand, the public cloud is good at providing high reliability and large capacity with the resource-sharing feature. On the other hand, the private cloud is adept at offering high flexibility and security, which guarantees data privacy during the work process.

Due to the complexity of the work process and increasing data volumes, scientific research studies with strict work steps cannot be managed manually. To address this problem, the workflow technology was proposed [7], where scientific workflows [8] can be used to manage, monitor, and execute these scientific processes. However, the amount of data involved in scientific workflows is commonly huge, which may need to be stored in the data centers in different geographical locations and transferred across data centers during the operation of scientific workflows. Therefore, it has become a research hotspot to effectively execute the data placement for scientific workflows in hybrid cloud environments under limited bandwidth conditions with the goal of reducing data transmission time [9–12].

Some researchers have contributed to addressing the problem of data placement for scientific workflows. Yuan et al. [13] proposed a k-means clustering algorithm-based data placement method, which utilized the dependency

among data and considered the load balancing in data centers. Reddy et al. [14] designed an entropy-based data placement strategy for enhancing the map-reduce performance in Hadoop clusters, where the k-means clustering algorithm was used to group different datasets. However, this method was not suitable for hybrid cloud environments with different capacities of data centers. Li et al. [15] designed a data placement solution for hybrid data centers, which can reduce the data transmission time. Also for hybrid data centers, the data placement strategy proposed in [16] reduced the data transmission volume and the data transmission times across data centers. However, they did not consider some essential factors in data placement, such as differences in data centers (e.g., capacities and bandwidth) and bandwidth fluctuations. Zheng et al. [17] and Cui et al. [18] developed a data placement scheme based on the GA, which may easily fall into the local optimal solution during operation. As for optimization objectives in data placement, Liu et al. [19] set the transmission times of crossing data centers as an objective, Deng et al. [20] and Zhao et al. [21] targeted the data transmission volume, and Chen et al. [22] aimed for reducing the transmission costs. However, these methods did not involve the network bandwidth and its fluctuations, and thus, it is hard for them to map the data transmission time from their models to real-world network environments [23].

Moreover, most of traditional data placement strategies are based on deterministic environments. However, uncertainty is an essential feature of network environments, which may have a significant impact on data transmission [24]. Due to various loads between data centers, bandwidth fluctuations, network congestion, and other hardware characteristics, the data transmission time may be changeable even if the same data are transmitted between fixed data centers. Therefore, the uncertainty should be considered when building the data placement model for scientific workflows. In response to the uncertainty, the fuzzy theory has emerged as an effective tool [25]. Sun et al. [26] and Lei [27] fuzzified the processing time, completion time, and deadline, and then, the job scheduling method was studied under specific constraints. Based on the analytic hierarchy process (AHP) model, a data placement strategy was proposed in [28] to select the most suitable storage sites, which applied the fuzzy comprehensive evaluation to candidate data centers for different users. However, they did not involve the data placement problem of scientific workflows, and their fuzzy object and optimization goal was not the data transmission time.

To address the above problems, we proposed an effective data placement strategy for scientific workflows in hybrid cloud environments. The main contributions of this paper are summarized as follows:

(i) We define and model the data placement problem for scientific workflows in hybrid cloud environments. Specifically, we fuzzify the data transmission time into triangular fuzzy numbers and regard it as the optimization objective of the proposed model.

(ii) Based on the problem definitions and modeling, the DPSO-FGA is proposed as the second contribution for reducing the fuzzy data transmission time while considering the uncertainty of data transmission time, the different numbers and capacities of private data centers, and network bandwidth limitations, which can well adapt to real-world network environments.

(iii) We validate the effectiveness of the proposed DPSO-FGA method by using various scientific workflows in hybrid cloud environments, which can outperform the classic CFRA and CFGA methods in terms of fuzzy data transmission time.

The rest of this paper is organized as follows. Section 2 defines the data placement problem for scientific workflows in hybrid cloud environments. In Section 3, the proposed DPSO-FGA is discussed in detail. Section 4 shows the performance evaluation of the proposed method with simulation experiments. Finally, we conclude this paper and look for future work in Section 5.

2. Problem Definitions and Modeling

2.1. Problem Definitions

Definition 1. Hybrid cloud environment

A hybrid cloud environment consists of public and private data centers, where each private data center has a certain capacity, while each public data center has no capacity limitation. Thus, a hybrid cloud environment is defined as

$$\begin{cases} DC = \{DC_{\text{pub}}, DC_{\text{pri}}\}, \\ DC_{\text{pub}} = \{dc_1, dc_2, \ldots, dc_n\}, \\ DC_{\text{pri}} = \{dc_{n+1}, dc_{n+2}, \ldots, dc_{n+m}\}, \\ dc_i = \{V_i, \Delta_i\}, \end{cases} \tag{1}$$

where DC_{pub} is the set of public data centers, DC_{pri} is the set of private data centers, dc_i represents the i-th data center, and V_i indicates the maximum capacity of a data center. Specifically, the capacity of a public data center is unlimited, while a private data center may reserve some storage space with an upper limit V_i, and $\Delta_i \in \{0, 1\}$ represents the attribute of dc_i. If $dc_i \in DC_{\text{pub}}$, then $\Delta_i = 0$, and dc_i can be used to store public data. If $dc_i \in DC_{\text{pri}}$, then $\Delta_i = 1$, and dc_i can be used to store both public and private data. For any two data centers dc_i and dc_j, b_{ij} represents the network bandwidth between them, which is assumed to be known and fluctuate within a certain range.

Definition 2. Scientific workflow

The scientific workflow is a data-intensive application consisting of tasks and datasets, where a task may be related to multiple datasets and a dataset may also be related to multiple tasks. There is a data dependency relationship between the tasks, where the output datasets of a task may be the input datasets of other tasks. Meanwhile, there is also a

sequential relationship between the tasks, where a task may only be executed after all its predecessor tasks have been executed. After all the tasks are completed, the scientific workflow ends. In particular, the task without a predecessor task is the beginning task and the task without a successor task is the ending task. Moreover, datasets can be divided into initial and generated datasets, where the original input datasets of a scientific workflow are the initial datasets and the datasets generated during the running process are the generated datasets. Also, datasets can be divided into private and public datasets, where private datasets can only be stored in private data centers and the tasks using them as the input datasets must also be scheduled to the same data centers. By contrast, public datasets have no restriction on storage locations. Therefore, a scientific workflow is defined as a directed acyclic graph (DAG), denoted by G as

$$\begin{cases} G = \langle T, E, DS \rangle, \\ T = \{t_1, t_2, \ldots, t_c\}, \\ E = \{e_{12}, e_{13}, \ldots, e_{ij}\}, \\ DS = \{ds_1, ds_2, \ldots, ds_l\}, \\ t_i = \{I_i, O_i, DC(t_i)\}, \\ ds_i = \langle v_i, gt_i, lc_i \rangle, \end{cases} \tag{2}$$

where T is the set of tasks in G, E is the set of data dependencies between different tasks in G, and DS is the set of datasets in G. Specifically, t_c represents the c-th task and e_{ij} indicates the data dependency between tasks t_i and t_j, where $e_{ij} = 1$ indicates that t_i is the direct predecessor task of t_j. Moreover, ds_l is the l-th dataset, I_i is the input dataset of t_i, O_i is the output dataset of t_i, and $DC(t_i)$ is the data center for executing t_i. Furthermore, v_i is the size of the dataset ds_i, gt_i is the task number of generating ds_i, in which gt_i of the initial dataset is 0, and lc_i is the serial number of the data center storing ds_i.

It should be noted that the settings of privacy datasets in scientific workflows need to satisfy three logical rules. Specifically, for the task t_i in the hybrid cloud environment DC, when the set of input or output datasets of t_i (denoted by {Ii, Oi}) contains the privacy dataset ds_i, the following holds:

(i) Rule 1. $lc_i \equiv A$.

(ii) Rule 2. $DC(t_i) \equiv A$.

(iii) Rule 3. For each privacy dataset $ds_j \in \{I_i, O_i\}$, $lc_j \equiv DC(t_j) \equiv A$.

According to Definition 2, private datasets can only be stored in private data centers, while their storage locations cannot be changed. As shown in Rule 1, private datasets cannot be transmitted across data centers, and thus, the data center for executing a task using the dataset as an input or output must be fixed. As known from Rule 2, locations of private datasets for fixed tasks must be consistent with execution locations of the tasks, and thus, the privacy datasets cannot be stored in other data centers. Otherwise, the tasks cannot be executed.

Definition 3. Fuzzy data transmission time

When optimizing uncertainty problems, there are commonly three types of theories, including the probability theory, gray theory, and fuzzy theory. Specifically, the probability theory can be applied in sampling problems with massive samples, the gray theory is suitable for the problems with fewer samples, and the fuzzy theory can be used to solve the problems with unclear extensions of concepts [29]. As for addressing the data placement problem of scientific workflows, the fuzzy theory can be regarded as an effective tool because this problem has no clear boundary or the limitation involves uncertainty.

In the past research, the data transmission time was usually defined as the ratio of the dataset size to the bandwidth between data centers, without considering the other essential factors such as bandwidth fluctuations. However, the data transmission time is uncertain in real-world network environments. In response to this uncertainty, by utilizing the fuzzy theory, triangular fuzzy numbers are introduced to represent the data transmission time. For each independent data transmission process, the mapping $\langle dc_i, ds_k, dc_j \rangle$ indicates that the dataset ds_k is transmitted from the data center dc_i to dc_j. Therefore, the fuzzy data transmission time is defined as

$$\tilde{T}_{\text{transfer}}(dc_i, ds_k, dc_j) = (a^1_{ikj}, a^2_{ikj}, a^3_{ikj}), \tag{3}$$

where a^1_{ikj} and a^3_{ikj} are the lower and upper bound elements of the triangular fuzzy number, respectively. When $a^1_{ikj} = a^2_{ikj} = a^3_{ikj}$, the triangular fuzzy number degenerates into a real number. Moreover, the membership function indicates the degree which the element x belongs to the fuzzy interval. When $x = a^2_{ikj}$, the element x completely belongs to the interval. The membership function of the triangular fuzzy number is defined as

$$\mu = \begin{cases} \dfrac{x - a^1_{ikj}}{a^2_{ikj} - a^1_{ikj}}, & a^1_{ikj} \le x \le a^2_{ikj}, \\ \dfrac{x - a^3_{ikj}}{a^2_{ikj} - a^3_{ikj}}, & a^2_{ikj} \le x \le a^3_{ikj}, \\ 0, & \text{else.} \end{cases} \tag{4}$$

Definition 4. Calculation of fuzzy number

(1) The model involves addition and comparison operations between fuzzy numbers. For the triangular fuzzy numbers $\tilde{s} = (s_1, s_2, s_3)$ and $\tilde{t} = (t_1, t_2, t_3)$, the above operations are defined as follows:

(i) Addition operation (calculating the fuzzy data transmission time):

$$\tilde{s} + \tilde{t} = \tilde{t} + \tilde{s} = (s_1 + t_1, s_2 + t_2, s_3 + t_3). \tag{5}$$

(ii) Comparison operation (comparing the fuzzy completion time and choosing suitable values).

For $\widetilde{s} = (s_1, s_2, s_3)$, three comparison values are defined as

$$\begin{aligned} c_1(\widetilde{s}) &= \frac{(s_1 + 2s_2 + s_3)}{4}, \\ c_2(\widetilde{s}) &= s_2, \\ c_3(\widetilde{s}) &= s_3 - s_1. \end{aligned} \tag{6}$$

According to the literature [30], if $c_1(\widetilde{s}) > c_1(\widetilde{t})$, $\widetilde{s} > \widetilde{t}$. If $c_2(\widetilde{s}) > c_2(\widetilde{t})$, $\widetilde{s} > \widetilde{t}$; if $c_3(\widetilde{s}) > c_3(\widetilde{t})$, $\widetilde{s} > \widetilde{t}$; otherwise, $\widetilde{s} = \widetilde{t}$.

(2) The model involves the addition, subtraction, multiplication, division, fuzzification, and defuzzification operations between fuzzy and real numbers. For a triangular fuzzy number $\widetilde{s} = (s_1, s_2, s_3)$ and a real number t, the above operations are defined as follows:

(i) Addition and subtraction operations:

$$\begin{aligned} \widetilde{s} + t &= t + \widetilde{s} = (s_1 + t, s_2 + t, s_3 + t), \\ \widetilde{s} - t &= -(t - \widetilde{s}) = (s_1 - t, s_2 - t, s_3 - t). \end{aligned} \tag{7}$$

(ii) Multiplication and division operations:

$$\begin{aligned} \widetilde{s} \cdot t &= t \cdot \widetilde{s} = (s_1 t, s_2 t, s_3 t), \\ \frac{\widetilde{s}}{t} &= \left(\frac{s_1}{t}, \frac{s_2}{t}, \frac{s_3}{t}\right), \quad \text{where } t \neq 0. \end{aligned} \tag{8}$$

(iii) Fuzzification and defuzzification operations.

On the one hand, the fuzzification operation, according to the literature [26], is defined as

$$\begin{cases} s_1 \in [\delta_1 s, s], \\ s_2 = s, \\ s_3 \in [s, \delta_2 s], \end{cases} \tag{9}$$

where $\delta_1 < 1, \delta_2 > 1$.

On the other hand, the defuzzification operation is commonly used to quantitatively compare fuzzy numbers and analyze results. Li [31] defined the mean and standard deviation of fuzzy numbers under uniform distribution and proportional distribution, where the proportional distribution is suitable for the uncertainty problem of data transmission time. For the triangular fuzzy number $\widetilde{S}_{\text{total}}$, the mean and standard deviation are defined as

$$\begin{cases} \widetilde{S}_\mu = \dfrac{\int x\widetilde{S}^2_{\text{total}}(x)\mathrm{d}x}{\int \widetilde{S}^2_{\text{total}}(x)\mathrm{d}x} = \dfrac{s_1 + 2s_2 + s_3}{4}, \\ \widetilde{S}_\sigma = \left[\dfrac{\int x\widetilde{S}^2_{\text{total}}(x)\mathrm{d}x}{\int \widetilde{S}^2_{\text{total}}(x)\mathrm{d}x} - S^2_\mu\right]^{1/2} = \left[\dfrac{2(s_1 - s_2)^2 + (s_1 - s_3)^2 + 2(s_2 - s_3)^2}{80}\right]^{1/2}, \\ \min \widetilde{S}_{\text{total}} = (s_1, s_2, s_3) \overset{\text{translate}}{\Rightarrow} \min \widetilde{S}_\mu + \partial \widetilde{S}_\sigma, \partial \geq 0, \end{cases} \tag{10}$$

where $\widetilde{S}_\mu$ is the mean of the triangular fuzzy number $\widetilde{S} = (s_1, s_2, s_3)$, which reflects the most likely value of the fuzzy number under probability measurement. $\widetilde{S}_\sigma$ indicates the standard deviation of $\widetilde{S} = (s_1, s_2, s_3)$, which reflects the uncertainty degree of the fuzzy number. ∂ represents the weight of $\widetilde{S}_\sigma$.

Definition 5. Data placement strategy

The purpose of effective data placement is to reduce the data transmission time while meeting the order of task execution, the proportion of dataset privacy, and the capacity constraints of data centers. Only when the datasets required by a task are transmitted to the same data center, the task can be executed. Moreover, the time of scheduling a task to a data center is much shorter than the data transmission time [32], and thus, the model may need to focus on the data placement strategy. Before executing a task, the data center with the least fuzzy data transmission time will be chosen to schedule the task. Therefore, the data placement strategy is defined as

$$\begin{cases} S = (DS, DC, M, \widetilde{T}_{\text{total}}), \\ M = \underset{i=1,2,\ldots,|DC|}{\cup} \{dc_i, ds_k, dc_j\}, \\ \widetilde{T}_{\text{transfer}}(dc_i, ds_k, dc_j) = \left(a^1_{ikj}, a^2_{ikj}, a^3_{ikj}\right), \\ \widetilde{T}_{\text{total}} = \sum_{i=1}^{DC} \sum_{j \neq i}^{DC} \sum_{k=1}^{DS} \widetilde{T}_{\text{transfer}}(dc_i, ds_k, dc_j) \cdot e_{ijk}, \end{cases} \tag{11}$$

where M represents the mapping between the dataset DS and the set of data centers DC. $\{dc_i, ds_k, dc_j\}$ indicates that the dataset ds_k is transmitted from the data center dc_i to dc_j.

$\widetilde{T}_{\text{transfer}}(dc_i, ds_{\underline{k}}, dc_j)$ is the fuzzy data transmission time of $\{dc_i, ds_k, dc_j\}$. $\widetilde{T}_{\text{total}}$ is the total fuzzy data transmission time during the operation of scientific workflows, where $e_{ijk} = \{0,1\}$ indicates whether there is $\{dc_i, ds_k, dc_j\}$ during this time ($e_{ijk} = 1$ for yes and $e_{ijk} = 0$ for no).

2.2. Modeling. According to the above definitions, the data placement problem for scientific workflows is modeled based on the fuzzy theory with the objective of minimizing the fuzzy data transmission time while considering the capacity constraints of data centers. The problem model is defined as

$$\begin{cases} \min \widetilde{T}_{\text{total}} \\ \text{s.t. } \forall i, \sum_{j=1}^{DS} v_j \cdot u_{ij} \le V, \end{cases} \tag{12}$$

where $u_{ij} = \{0,1\}$ indicates whether the dataset ds_j is stored in the data center dc_i ($u_{ij} = 1$ for yes and $u_{ij} = 0$ for no).

3. Effective Data Placement for Scientific Workflows Based on DPSO-FGA

In light of the advantages of the particle swarm optimization (PSO), genetic algorithm (GA), and fuzzy theory, we propose an adaptive discrete particle swarm optimization algorithm based on the fuzzy theory and genetic algorithm operators (DPSO-FGA) to implement the effective data placement for scientific workflows, with the goal of minimizing the fuzzy data transmission time.

3.1. PSO Algorithm. The PSO algorithm was derived from the literature [33], which was inspired by the regularity in the cluster activities of flying birds. Based on the information exchanges between individuals, the movement of the entire population will gradually become orderly, and eventually, the optimal solution can be obtained, where the solution of the optimization problem is called "particle". When running the PSO algorithm, a fixed-size particle swarm is randomly initialized, where each particle constantly keeps iterating and updating through tracking the optimal solution found by itself and the population. The update of particles contains two aspects as follows.

(i) Speed update:

$$\begin{aligned} V_i(t+1) = wV_i(t) + c_1 r_1 [p_i(t) - X_i(t)] \\ + c_2 r_2 [g(t) - X_i(t)]. \end{aligned} \tag{13}$$

(ii) Location update:

$$X_i(t+1) = X_i(t) + V_i(t+1), \tag{14}$$

where the detailed definitions of the symbols can be found in [29].

3.2. Fitness Function. A fitness function needs to be defined for particles to track the optimal solution during the update process. As the optimization goal, the fuzzy data transmission time is used to define the fitness function as

$$F(S) = \widetilde{T}_{\text{total}}(X_i), \tag{15}$$

where $F(S)$ represents the fitness function of data placement strategy S and $\widetilde{T}_{\text{total}}(X_i)$ indicates the fuzzy data transmission time of the particle X_i.

If the total size of datasets placed in a data center does not exceed its maximum capacity, the particle can be a feasible solution. Otherwise, it is infeasible. When making the selection between feasible and infeasible solutions, the feasible one will be directly selected. When making the selection between feasible solutions, the particle with the smaller fitness function will be selected. When making the selection between infeasible solutions, the particle with the smaller fitness function will be also selected because it is more likely to become a feasible solution in subsequent operations.

3.3. Particle Encoding. The particle encoding needs to meet three principles, including completeness, nonredundancy, and soundness [34]. Specifically, the n-dimensional particles are discretely encoded [35], where n represents the number of datasets involved in the scientific workflow. Therefore, the structure of the particle i at the t-th iteration is defined as

$$X_i(t) = \left[x_i^1(t), x_i^2(t), \ldots, x_i^n(t)\right], \tag{16}$$

where $x_i^k(t)$ represents the placement location of the k-th dataset at the t-th iteration.

The following is an example of the particle encoding, where the particle number is 3, the current iteration number is 10, the number of datasets is 10, and the number of data centers is 4. Moreover, the datasets with underlines indicate that they are privacy datasets, where the data centers used for storing them cannot be changed during the subsequent update process:

$$X_3(10) = [1, 2, \underline{4}, 3, 2, \underline{1}, \underline{3}, 4, 2, 1]. \tag{17}$$

3.4. Particle Update. The traditional update of the PSO algorithm is shown in equations (13) and (14)fd16, which reveal some weaknesses in practical operation, such as low search ability, small solution space, and the premature convergence to the local optimum. To address these problems, crossover and mutation operations in the GA are introduced into the update. Since there are certain proportions of privacy datasets, the data centers used for storing them cannot be changed during the update process.

For the inertial part of the traditional update, the mutation operation is defined as

$$A_i(t+1) = \omega \oplus M_u(X_i(t)) = \begin{cases} M_u(X_i(t)), & r_0 < \omega, \\ X_i(t), & \text{else}, \end{cases} \tag{18}$$

where $r_0 \in (0,1)$ represents a random factor and the mutation operation $M_u()$ is to randomly change a quantile in the encoded particle within a range of values. It should be noted that the quantiles of privacy datasets cannot be mutated. Moreover, an infeasible particle should select a quantile, which makes the particle infeasible, to be mutated. Thus, the quantile to be selected should be the location of an overloaded data center. For instance, the following mutation operation is taken based on equation (17) as

$$\begin{aligned} &X_3(10) = [1, 2, \underline{4}, 3, 2, \underline{1}, \underline{3}, 4, 2, 1], \\ &\Downarrow \text{Mutation} \\ &A_3(11) = [1, 3, \underline{4}, 3, 2, \underline{1}, \underline{3}, 4, 2, 1], \end{aligned} \tag{19}$$

where the 2nd quantile is selected by the mutation operation and the corresponding data center number changes from 2 to 3.

For the individual and population cognitions in the traditional update, the cross operation is introduced as

$$\begin{aligned} B_i(t+1) &= c_1 \oplus C_p\left(A_i(t+1), p_i(t)\right) = \begin{cases} C_p\left(A_i(t+1), p_i(t)\right), & r_1 < c_1, \\ A_i(t+1), & \text{else}, \end{cases} \\ C_i(t+1) &= c_2 \oplus C_g\left(B_i(t+1), g(t)\right) = \begin{cases} C_g\left(B_i(t+1), g(t)\right), & r_2 < c_2, \\ B_i(t+1), & \text{else}, \end{cases} \end{aligned} \tag{20}$$

where $r_1, r_2 \in (0,1)$ represent random factors; $C_p(A_i(t+1), p_i(t))$ and $C_g(B_i(t+1), g(t))$, are to randomly select two quantiles of the encoded particles $A_i(t+1)$ and $B_i(t+1)$ and cross with the values at the same positions of $p_i(t)$ and $g(t)$. It should be noted that the storage locations of privacy datasets cannot be changed when crossing. For instance, the following crossover operation is taken based on (17) fd19 as

$$\begin{aligned} \frac{p_3(10)}{g(10)} = &\quad [1, 2, \underline{4}, 2, 4, \underline{1}, \underline{3}, 4, 2, 1], \\ &\downarrow\downarrow \text{Crossover} \\ \frac{A_3(11)}{B_3(11)} = &\quad [1, 2, \underline{4}, 2, 4, \underline{1}, \underline{3}, 4, 2, 1], \\ &\downarrow\downarrow \text{Turn Into} \\ \frac{B_3(11)}{C_3(11)} &\quad [1, 2, \underline{4}, 2, 4, \underline{1}, \underline{3}, 4, 2, 1], \end{aligned} \tag{21}$$

where the crossover operations happens at the 4th and 5th quintiles and the serial numbers of the data centers on the quantiles change from 2 to 4.

In summary, the process of the particle update is defined as

$$X_i(t+1) = c_2 \oplus C_g\Big(c_1 \oplus C_p\left(\omega \oplus M_u\left(X_i(t)\right), p_i(t)\right), g(t)\Big). \tag{22}$$

3.5. Mapping from Particles to Data Placement Results. Algorithm 1 shows the mapping from encoded particles to data placement results, where G represents the scientific workflow, DC indicates the hybrid cloud environment, X denotes the encoded particles, and S expresses the data placement strategy.

The execution steps of Algorithm 1 are listed as follows:

Step 1 (line 1). Initialize the current storage capacity of data centers $dc_{cur(i)}$ to 0 and the fuzzy data transmission time $\widetilde{T}_{\text{total}}$ to (0, 0, 0).

Step 2 (lines 2~7). Place each initial dataset into data centers according to their numbers and update $dc_{cur(i)}$. If $dc_{cur(i)}$ exceeds the maximum capacity of data centers, the solution corresponding to the particle is infeasible and the current operation will be stopped and returned.

Step 3 (lines 8~15). During the task traversal, the data center dc_j with the smallest fuzzy data transmission time will always be selected to place the task t_j. If the solution corresponding to the particle is infeasible (i.e., the sum of $dc_{cur(j)}$, $sum(I_j)$, and $sum(O_j)$ exceeds the maximum capacity of data centers), the current operation will be stopped and returned. Otherwise, the output dataset O_j of the task t_j will be placed into a corresponding data center and the storage capacity of data centers will be updated.

Step 4 (lines 16~20). Traverse all tasks, calculate the fuzzy data transmission time $\widetilde{T}_j$ for each dataset that needs to be transmitted across data centers, and sum them to obtain the total fuzzy data transmission time $\widetilde{T}_{\text{total}}$.

Step 5 (line 21). Output $\widetilde{T}_{\text{total}}$, X, and the corresponding data placement strategy.

3.6. Model Parameters. The inertial weight w in (13) has a direct influence on the convergence of the PSO algorithm [36], which can affect the search speed of particles in solution space. Thus, we propose a new method to define the weight w as follows, which can adaptively adjust the value of w based on the pros and cons of the corresponding solution of the current particle (i.e., the degree of difference between the current and historically optimal particles):

Procedure dataPlacement(G, DC, X)
Input: G, DC, X
Output: $S = (DS, DC, Map, \widetilde{T}_{\text{total}})$
1: **Initialization:** set the current storage capacity of data centers $dc_{cur(i)}$ to 0 and the fuzzy data transmission time $\widetilde{T}_{\text{total}}$ to (0, 0, 0)
2: **for each** ds_i **of** DS_{ini}//*Determine whether the particle would cause the data center overloaded*
3: $dc_{cur(X[i])} += v_i$//*Place the dataset ds_i in the data center $dc_{X[i]}$*
4: **if** $dc_{cur(X[i])} > V_{X[i]}$
5: return ***this particle is infeasible***
6: **end if**
7: **end for**
8: **for** $j = 1$ **to** $|T|$//*Determine whether the data center is overloaded during task execution*
9: Place the task t_j in the data center dc_j with the least fuzzy data transmission time
10: **if** $dc_{cur(j)} + sum(I_j) + sum(O_j) > v_j$
11: **return *this particle is infeasible***
12: **end if**
13: Place the output dataset O_j of t_j into the corresponding data center
14: Update the storage capacity of the data center
15: **end for**
16: **for** $j = 1$ **to** $|T|$//*Calculate the fuzzy transmission time of the corresponding data layout*
17: Find the data centers ds_i in the placement of task t_j's input dataset I_j
18: Calculate the fuzzy data transmission time $\widetilde{T}_j$ generated by the input dataset I_j to ds_j
19: $\widetilde{T}_{\text{total}} += \widetilde{T}_j$
20: **end for**
21: Output $\widetilde{T}_{\text{total}}$, $X$and the corresponding data placement strategy
End procedure

Algorithm 1: Mapping from encoded particles to data placement results.

$$w = w_{\max} - (w_{\max} - w_{\min}) \cdot \exp\left\{\frac{d(X_i(t), g(t))/|DS|}{(d(X_i(t), g(t))/|DS|) - 1.01}\right\}, \tag{23}$$

where $d(X_i(t), g(t))$ represents the degree of difference between the current particle $X_i(t)$ and the historically optimal particle $g(t)$ of the current population (i.e., the number of different values on the same quantiles). In the early stage of training, $d(X_i(t), g(t))$ is usually large with the large value of w. Therefore, it is necessary to expand the search range of particles in solution space, in order to find the optimal solution and avoid prematurely falling into local optimum. In the later stage of training, $d(X_i(t), g(t))$ becomes small with the small value of w. Thus, it is better to narrow the search range of particles and accelerate the search speed of particles for the optimal solution.

Moreover, the individual and population cognition factors (i.e., c_1 and c_2) are defined by using the gradient descent method [37].

4. Performance Evaluation

4.1. Parameter and Environment Settings. The scientific workflow model comes from five different scientific fields [38], including CyberShake, Epigenomics, Inspiral, Montage, and Sipht. Each scientific field has a scientific workflow with a different number of tasks, and each scientific workflow has a unique task structure, number of datasets, and computational requirements [39]. Specifically, a medium-sized (about 50 tasks) workflow in each field is selected for experiments, and the parameter and environment settings are shown in Table 1.

Moreover, some extrasettings are shown as follows:

(i) Maximum capacity: the datum capacity is set to $\sum_{i=1}^{|DS|} v_i / (|DC| - 1)$, where the maximum capacity of the three private data centers is set to 2.6 times the datum capacity.

(ii) Bandwidth (M/s) between data centers: the bandwidth between dc_1 and $\{dc_2, dc_3, dc_4\}$ is set to {10, 20, 30}, the bandwidth between dc_2 and $\{dc_3, dc_4\}$ is set to {150, 150}, and the bandwidth between dc_3 and dc_4 is set to 100.

(iii) Proportion of privacy datasets: due to the difference of datasets among various workflows, the proportions of privacy datasets of in the scientific workflows, including CyberShake, Epigenomics, Inspiral, Montage, and Sipht, are set to {0.25, 0.2, 0.2, 0.2, 0.02}.

(iv) Fuzzy parameter: based on the fuzzy theory, the data transmission time T is fuzzified into the triangular fuzzy number $\widetilde{T}$, where the fuzzy parameters are set to $\sigma_1 = 0.85, \sigma_2 = 1.2$.

4.2. Comparison Algorithms. The proposed DPSO-FGA is compared with the constraint fuzzy randomized algorithm (CFRA) and constraint fuzzy greedy algorithm (CFGA), which can improve the performance of the randomized algorithm (RA) and greedy algorithm (GA) in data placement. The CFRA and CFGA rely on the fuzzy theory while considering some essential conditions, including the application scenarios of scientific workflows, privacy settings, and capacity constraints. The conditions refer to meet the maximum capacity

Table 1: Environment and parameter settings.

Parameter	Value
Population size	100
Maximum number of iterations	1000
$w_{\max}$, $w_{\min}$	0.9, 0.4
$c_1^{\text{start}}, c_1^{\text{end}}$	0.9, 0.2
$c_2^{\text{start}}, c_2^{\text{end}}$	0.4, 0.9
Hybrid cloud environment	$DC = \{DC_{\text{pub}}, DC_{\text{pri}} \mid DC_{\text{pub}} = \{dc_1\}, DC_{\text{pri}} = \{dc_2, dc_3, dc_4\}\}$
Software environment	Windows 10 64-bit, Python 3.7
Hardware environment	Intel® Core™ i7-6700HQ CPU @2.60GHZ, RAM 8.00 GB

requirements of data centers and the proportion of private datasets during the data placement process.

(i) The steps of CFRA

Step 1. Set privacy datasets and the maximum capacity of data centers, initialize parameters, and keep the same values as the corresponding parameters in the DPSO-FGA.

Step 2. Generate a random population that meets the conditions according to the discrete encoding method in the DPSO-FGA. The population contains a certain number of individuals, and each individual represents a candidate solution for data placement.

Step 3. Define the fitness function as the fuzzy data transmission time of the corresponding solution of the individual encoding.

Step 4. Calculate the fitness value of each individual and compare it with the current best individual of the population. If the current individual is the better one, update the best individual of the population by using the current one.

Step 5. End the traversal and output the best individual with its fitness value.

(ii) The steps of CFGA

Step 1. Set privacy datasets and the maximum capacity of data centers, initialize parameters, and keep the same values as the corresponding parameters in the DPSO-FGA.

Step 2. Design a data placement strategy. According to the task execution sequence of the scientific workflow, traverse the datasets that have not been deployed for tasks. If the current task has been placed, the dataset will also be placed to the same data center by using the GA. If the current task has not been placed but there is already a placed dataset, the dataset will be placed to the same data center as the placed dataset by using the GA. If the current task has not been placed and there is no already placed dataset, the dataset will be placed into the data center with the smallest fuzzy data transmission time by using the GA.

Step 3. Calculate the fuzzy data transmission time of the current data placement strategy and output the strategy.

4.3. Experimental Results and Analysis. To avoid the randomness of results, 10 independent experiments are carried out on five scientific workflows under different environment settings. Table 2 records the average fuzzy data transmission time of different algorithms under various scientific workflows.

In the subsequent experimental results, the fuzzy data transmission time is defuzzified, in order to make the comparison between the algorithms more intuitive, where ∂ is set to 1.

Figure 1 shows the defuzzification results for the fuzzy data transmission time of different algorithms under various scientific workflows, where the names of the scientific workflows are indicated by their first letter.

From the perspective of algorithms, the DPSO-FGA outperforms the CFRA and CFGA. This is because that the CFGA may easily fall into the local optimum by using the GA during execution, and thus, it ignores the global performance. Moreover, the overall performance of the CFRA is better than CFGA since the search space of the CFRA is larger than the CFGA and will not fall into the local optimum, and thus, the CFRA can obtain a good solution when the algorithm runs for a long time. However, the CFRA does not consider the fitness of the current particle when a solution is generated, and thus, the performance of the CFRA is worse than the DPSO-FGA. From the perspective of workflows, the data transmission time of the same algorithm in various scientific workflows is significantly different. Although all these scientific workflows contain about 50 tasks, the number of datasets used varies greatly. For example, the CyberShake uses datasets only about 70 times, while the Sipht uses datasets up to 4000 times, which results in the different data transmission time between them.

As the number of private data centers in a hybrid cloud environment sometimes changes, the performance of DPSO-FGA needs to be evaluated with different numbers of private data centers. Thus, we change the number of private data centers without modifying other default settings. Specifically, these three algorithms are tested when the number of private data centers is set to {3, 5, 6, 8, 10}, where the bandwidth between newly added private data centers and public data centers is set to 20 M/s and the bandwidth between other private data centers is set to 120 M/s. The experimental results are shown in Figure 2.

From the perspective of algorithms, the performance of the DPSO-FGA outperforms the CFRA and CFGA, and the reasons have been analyzed in Figure 1. From the perspective of private data centers, as the number of private data centers

Table 2: Average fuzzy data transmission time of different algorithms under various scientific workflows.

Workflow	Algorithm	Average fuzzy data transmission time (s)
CyberShake	DPSO-FGA	(375683.37, 415977.97, 448166.12)
	CFRA	(613685.49, 658202.55, 728115.64)
	CFGA	(760212.74, 819721.35, 890377.31)
Epigenomics	DPSO-FGA	(297570.10, 300909.35, 326971.76)
	CFRA	(561274.26, 606573.37, 654489.23)
	CFGA	(1192066.03, 1242136.17, 1290256.56)
Inspiral	DPSO-FGA	(2356942.36, 2521230.04, 2734877.94)
	CFRA	(5859201.90, 6335216.52, 7023988.75)
	CFGA	(8025939.27, 8547697.91, 9737463.34)
Montage	DPSO-FGA	(794955.76, 840543.65, 863448.55)
	CFRA	(1242644.26, 1331192.71, 1445410.25)
	CFGA	(1920877.78, 2058800.54, 2311869.58)
Sipht	DPSO-FGA	(10733197.95, 12051002.69, 1289802.65)
	CFRA	(11543263.40, 12494008.57, 13725762.53)
	CFGA	(13105310.17, 14687168.03, 1591557.53)

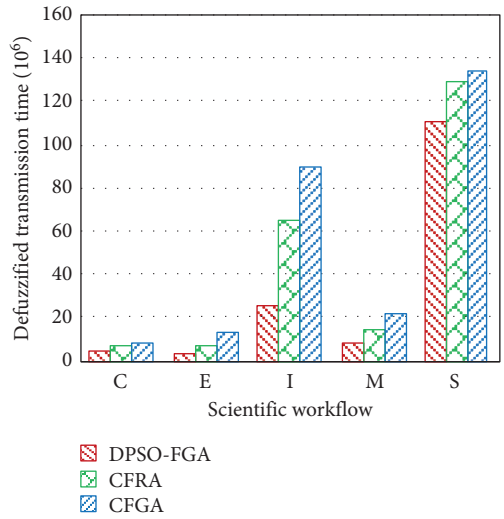

Figure 1: Average fuzzy data transmission time of different algorithms under various scientific workflows.

increases, the data transmission time of these three algorithms also inclines. This is because, with the increasing number of private data centers, the privacy datasets, which are randomly set according to the privacy proportion, are dispersed and fixed in more private data centers. Therefore, the fixed tasks, which require these private datasets, need to be executed in more scattered locations, and it will lead to increasing data transmission time.

Since the maximum capacity of private data centers is regarded as a constraint, the sensitivity of the DPSO-FGA to this constraint needs to be evaluated. Specifically, the CyberShake is selected as the scientific workflow for experiments, the multiple of datum capacity is set to {2, 2.6, 3, 5, 8}, and the rest of the settings remain default. The experimental results are shown in Figure 3.

When the maximum capacity of private data centers increases and the bandwidth between data centers remains the same, each data center is able to store more datasets and the datasets required for executing tasks become more concentrated. Therefore, the data transmission time of the DPSO-FGA is reduced. Specifically, the fastest decline in data transmission time happens when the maximum capacity is 2~3 times than that of the datum capacity, and the slowest decline happens when the maximum capacity is 5~8 times than that of the datum capacity. This is because when the maximum capacity of data centers is small, the available space is small and the placement locations of datasets are restricted. Thus, the maximum capacity has a significant impact on data transmission time. When the maximum capacity of data centers becomes larger, each data center can store more datasets, and it will be easy to meet the operational requirements of scientific workflows. Therefore, the maximum capacity has little effect on the data transmission time.

Finally, the performance of the DPSO-FGA is evaluated under different bandwidths between data centers. Specifically, the CyberShake is selected as the scientific workflow for experiments, the multiple of the bandwidth between data centers relative to the default one is set to {0.5, 0.8, 1.5, 3, 5}, and the rest of the settings remains default. The experimental results are shown in Figure 4. The data transmission time

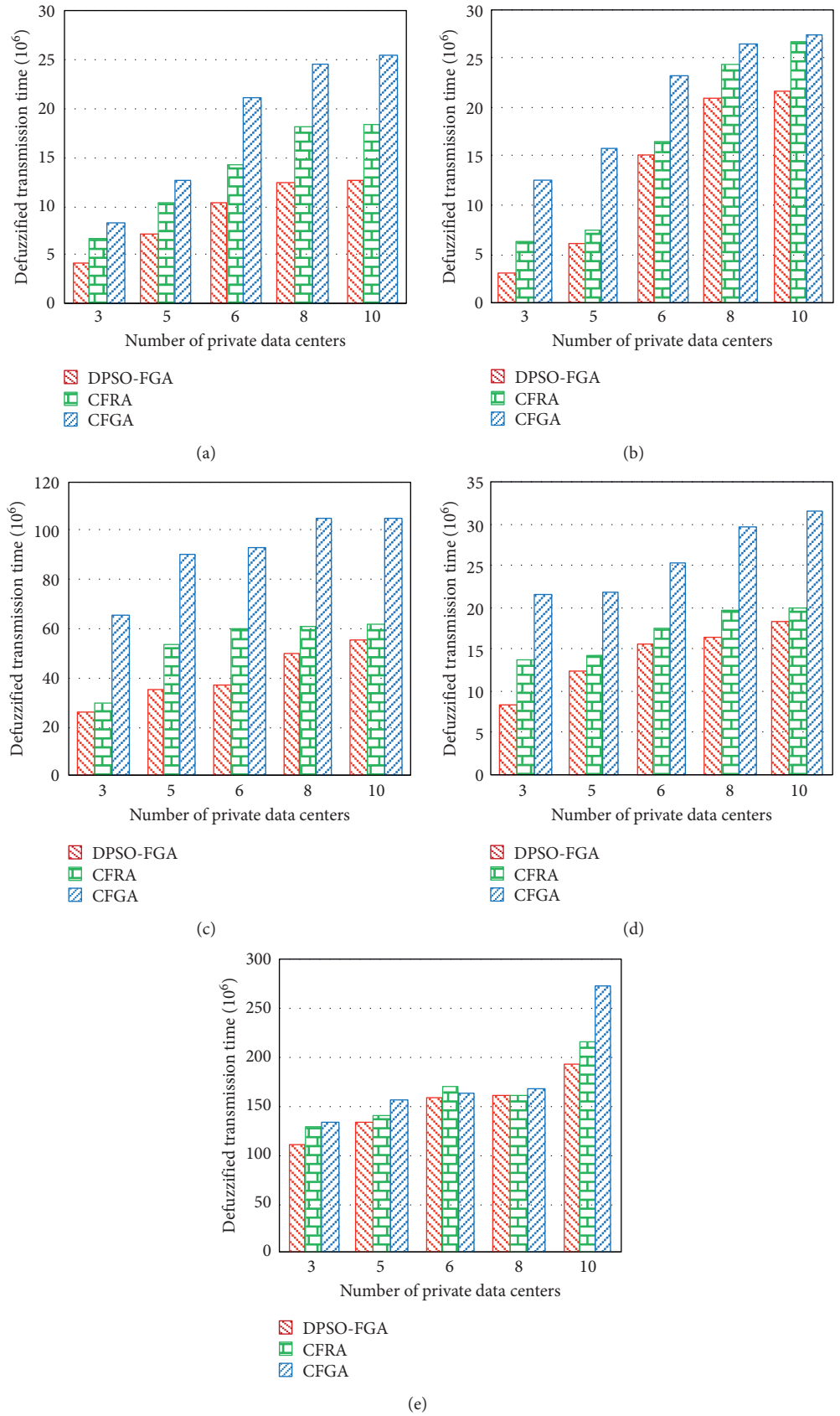

Figure 2: Average data transmission time of different algorithms with various numbers of private data centers: (a) Cybershake; (b) Epigenomics; (c) Inspiral; (d) Montage; (e) Sipht.

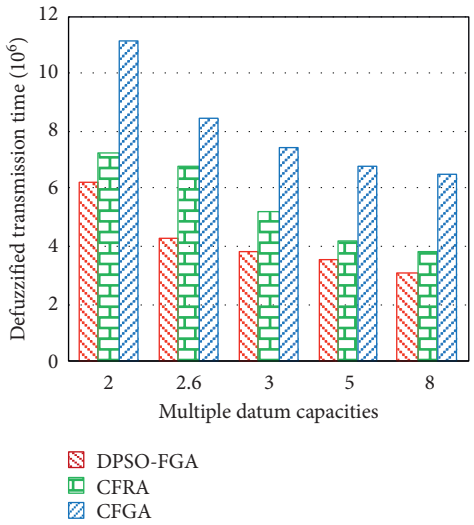

Figure 3: Average fuzzy data transmission time of different algorithms with various capacities of private data centers.

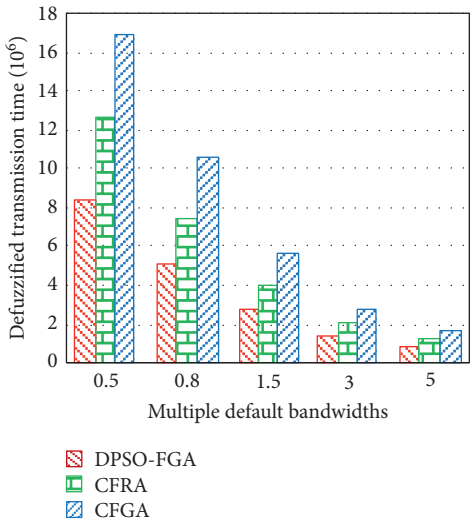

Figure 4: Average fuzzy data fuzzy transmission time of different algorithms with various bandwidths between data centers.

decreases greatly as the bandwidth increases, which indicates that the bandwidth changes between data centers will not significantly affect the data placement strategy.

5. Conclusions

In this paper, we propose a DPSO-FGA-based data placement method for scientific workflows in hybrid cloud environments. Based on the fuzzy theory, the DPSO-FGA fuzzifies the data transmission time for adapting to real-world network environments while considering the characteristics of hybrid cloud environments, bandwidth fluctuations, capacity limitations of private data centers, and dependencies between different scientific workflow tasks. Simulation results demonstrate the effectiveness of the proposed DPSO-FGA method. In the future, we will study the impact of other essential factors on the proposed method, such as different proportions of private datasets in scientific workflows and capacities of various private data centers. Moreover, under the conditions that are not critical for data transmission time, such as business network environments, data transmission costs between different clouds should also be regarded as a prioritized optimization goal. Therefore, a comprehensive model for minimizing the fuzzy data transmission time and costs will be researched.

Data Availability

The data used to support the findings of this study are produced by a public workflow generator available at https://confluence.pegasus.isi.edu/display/pegasus/WorkflowGenerator.

Conflicts of Interest

The authors declare that there are no conflicts of interest regarding the publication of this paper.

Authors' Contributions

Zheyi Chen and Xu Zhao are equally contributed to this work. Zheyi Chen and Xu Zhao developed the model, carried out the parameter estimations, and planned as well as performed the experiments. Zheyi Chen wrote the main part of the manuscript, while Xu Zhao and Bing Lin provided the support for writing materials. Bing Lin also took part in the design and evaluation of the model. Zheyi Chen and Bing Lin reviewed the manuscript. All the authors read and approved the final manuscript.

Acknowledgments

This work was supported by the National Key R&D Program of China (Grant no. 2018YFB1004800), Natural Science Foundation of China (Grant nos. 61672159, 41801324, and 61972165), Natural Science Foundation of Fujian Province (Grant nos. 2019J01286, 2019J01244, and 2018J01619), Young and Middle-Aged Teacher Education Foundation of Fujian Province (Grant no. JT180098), Open Foundation of Engineering Research Center of Big Data Application in Private Health Medicine, Fujian Province University (Grant no. KF2020001), Talent Program of Fujian Province for Distinguished Young Scholars in Higher Education, and China Scholarship Council (no. 201706210072). The authors sincerely thank Dr. Jia Hu and Dr. Geyong Min for providing useful advice that greatly improved this paper.

References

[1] B. Varghese and R. Buyya, "Next generation cloud computing: new trends and research directions," *Future Generation Computer Systems*, vol. 79, pp. 849–861, 2018.

[2] Z. Chen, J. Hu, G. Min, A. Y. Zomaya, and T. El-Ghazawi, "Towards accurate prediction for high-dimensional and highly-variable cloud workloads with deep learning," *IEEE Transactions on Parallel and Distributed Systems*, vol. 31, no. 4, pp. 923–934, 2020.

[3] X. Chen, F. Zhu, Z. Chen et al., "Resource allocation for cloud-based software services using prediction-enabled feedback control with reinforcement learning," *IEEE Transactions on Cloud Computing (TCC)*, 2020.

[4] K. Peng, H. Huang, S. Wan et al., "End-edge-cloud collaborative computation offloading for multiple mobile users in heterogeneous edge-server environment," *Wireless Networks*, 2020.

[5] X. Chen, H. Wang, Y. Ma, X. Zheng, and L. Guo, "Self-adaptive resource allocation for cloud-based software services based on iterative QoS prediction model," *Future Generation Computer Systems*, vol. 105, pp. 287–296, 2020.

[6] D. S. Linthicum, "Emerging hybrid cloud patterns," *IEEE Cloud Computing*, vol. 3, no. 1, pp. 88–91, 2016.

[7] B. Zhang, L. Yu, Y. Feng, L. Liu, and S. Zhao, "Application of workflow technology for big data analysis service," *Applied Sciences*, vol. 8, no. 4, Article ID 591, 2018.

[8] M. Atkinson, S. Gesing, J. Montagnat, and I. Taylor, "Scientific workflows: past, present and future," *Future Generation Computer Systems*, vol. 75, pp. 216–227, 2017.

[9] X. Li, L. Zhang, Y. Wu, X. Liu, E. Zhu et al., "A novel workflow-level data placement strategy for data-sharing scientific cloud workflows," *IEEE Transactions on Services Computing*, vol. 12, no. 3, pp. 370–383, 2019.

[10] B. Lin, F. Zhu, J. Zhang, J. Chen, X. Chen et al., "A time-driven data placement strategy for a scientific workflow combining edge computing and cloud computing," *IEEE Transactions on Industrial Informatics*, vol. 15, no. 7, pp. 4254–4265, 2019.

[11] W. Guo, B. Lin, G. Chen, Y. Chen, and F. Liang, "Cost-driven scheduling for deadline-based workflow across multiple clouds," *IEEE Transactions on Network and Service Management*, vol. 15, no. 4, pp. 1571–1585, 2018.

[12] B. Lin, W. Guo, N. Xiong, G. Chen, A. V. Vasilakos et al., "A pretreatment workflow scheduling approach for big data applications in multicloud environments," *IEEE Transactions on Network and Service Management*, vol. 13, no. 3, pp. 581–594, 2016.

[13] D. Yuan, Y. Yang, X. Liu, and J. Chen, "A data placement strategy in scientific cloud workflows," *Future Generation Computer Systems*, vol. 26, no. 8, pp. 1200–1214, 2010.

[14] K. H. K. Reddy, V. Pandey, and D. S. Roy, "A novel entropy-based dynamic data placement strategy for data intensive applications in Hadoop clusters," *International Journal of Big Data Intelligence*, vol. 6, no. 1, pp. 20–37, 2019.

[15] X. Li, Y. Wu, X. Liu et al., "Datacenter-oriented data placement strategy of workflows in hybrid cloud," *Journal of Software*, vol. 27, no. 7, pp. 1861–1875, 2016, in Chinese.

[16] Z. Liu, T. Xiang, B. Lin et al., "A data placement strategy for scientific workflow in hybrid cloud," in *Proceedings of IEEE International Conference on Cloud Computing (CLOUD)*, pp. 556–563, San Francisco, CA, USA, July 2018.

[17] P. Zheng, L.-Z. Cui, H.-Y. Wang, and M. Xu, "A data placement strategy for data-intensive applications in cloud," *Chinese Journal of Computers*, vol. 33, no. 8, pp. 1472–1480, 2010, in Chinese.

[18] L. Cui, J. Zhang, L. Yue, Y. Shi, H. Li, and D. Yuan, "A genetic algorithm based data replica placement strategy for scientific applications in clouds," *IEEE Transactions on Services Computing*, vol. 11, no. 4, pp. 727–739, 2018.

[19] S.-W. Liu, L.-M. Kong, K.-J. Ren, J.-Q. Song, K.-F. Deng, and H.-Z. Leng, "A two-step data placement and task scheduling strategy for optimizing scientific workflow performance on cloud computing platform," *Chinese Journal of Computers*, vol. 34, no. 11, pp. 2121–2130, 2011, in Chinese.

[20] K. Deng, K. Ren, M. Zhu et al., "A data and task co-scheduling algorithm for scientific cloud workflows," *IEEE Transactions on Cloud Computing (TCC)*, Early Access, vol. 8, no. 2, pp. 349–362, 2020.

[21] Q. Zhao, C. Xiong, X. Zhao et al., "A data placement strategy for data-intensive scientific workflows in cloud," in *Proceedings of IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing (CCGRID)*, pp. 928–934, Shenzhen, China, May 2015.

[22] Z. Chen, J. Hu, G. Min, and X. Chen, "Effective data placement for scientific workflows in mobile edge computing using genetic particle swarm optimization," *Concurrency and Computation: Practice and Experience (CCPE)*, Article ID e5413, 2019.

[23] B. Lin, T. Xiang, G. Chen et al., "Time-driven data placement strategy for scientific workflows in hybrid cloud," *Computer Integrated Manufacturing Systems*, vol. 25, no. 4, pp. 909–919, 2019, in Chinese.

[24] A. Vafamehr, M. E. Khodayar, S. D. Manshadi, I. Ahmad, and J. Lin, "A framework for expansion planning of data centers in electricity and data networks under uncertainty," *IEEE Transactions on Smart Grid*, vol. 10, no. 1, pp. 305–316, Jan. 2019.

[25] Z. Kovacic and S. Bogdan, *Fuzzy Controller Design: Theory and Applications*, CRC Press, Boca Raton, FL, USA, 2018.

[26] L. Sun, L. Lin, M. Gen, and H. Li, "A hybrid cooperative coevolution algorithm for fuzzy flexible job shop scheduling," *IEEE Transactions on Fuzzy Systems*, vol. 27, no. 5, pp. 1008–1022, 2019.

[27] D. Lei, "Fuzzy job shop scheduling problem with availability constraints," *Computers & Industrial Engineering*, vol. 58, no. 4, pp. 610–617, 2010.

[28] X. Wu and H. Guan, "Data set replica placement strategy based on fuzzy evaluation in the cloud," *Journal of Intelligent & Fuzzy Systems*, vol. 31, no. 6, pp. 2859–2868, 2016.

[29] X. Zhou, G. Zhang, J. Sun, J. Zhou, T. Wei, and S. Hu, "Minimizing cost and makespan for workflow scheduling in cloud using fuzzy dominance sort based HEFT," *Future Generation Computer Systems*, vol. 93, pp. 278–289, 2019.

[30] M. Sakawa and R. Kubota, "Fuzzy programming for multi-objective job shop scheduling with fuzzy processing time and fuzzy due date through genetic algorithms," *European Journal of Operational Research*, vol. 120, no. 2, pp. 393–407, 2000.

[31] R. Li, *Theory and Application of Fuzzy Multi-Criteria Decision*, Science Press, Beijing, China, 2002, in Chinese.

[32] K. Deng, K. Ren, J. Song et al., "A clustering based coscheduling strategy for efficient scientific workflow execution in cloud computing," *Concurrency and Computation: Practice and Experience (CCPE)*, vol. 25, no. 18, pp. 2523–2539, 2014.

[33] J. Kennedy and R. Eberhart, "Particle swarm optimization," in *Proceedings of IEEE International Conference on Neural Networks (ICNN)*, pp. 1942–1948, Perth, Australia, November-December 2002.

[34] J. Su, W. Guo, C. Yu et al., "Fault-tolerance clustering algorithm with load-balance aware in wireless sensor network," *Chinese Journal of Computers*, vol. 37, no. 2, pp. 445–456, 2014, in Chinese.

[35] M. A. Rodriguez and R. Buyya, "Deadline based resource provisioningand scheduling algorithm for scientific workflows on clouds," *IEEE Transactions on Cloud Computing*, vol. 2, no. 2, pp. 222–235, 2014.

[36] M. Masdari, F. Salehi, M. Jalali, and M. Bidaki, "A survey of PSO-based scheduling algorithms in cloud computing,"

Journal of Network and Systems Management, vol. 25, no. 1, pp. 122–158, 2017.

[37] Y. Shi and R. Eberhart, "A modified particle swarm optimizer," in *Proceedings of IEEE International Conference on Evolutionary Computation (ICEC)*, pp. 69–73, Anchorage, AK, USA, May 1998.

[38] S. Bharathi, A. Chervenak, E. Deelman et al., "Characterization of scientific workflows," in *Proceedings of the IEEE Workshop on Workflows in Support of Large-Scale Science*, pp. 1–10, Austin, TX, USA, November 2008.

[39] "Workflow generator," 2014, https://confluence.pegasus.isi.edu/display/pegasus/WorkflowGenerator.