PDM: Privacy-Aware Deployment of Machine-Learning Applications for Industrial Cyber-Physical Cloud Systems

Xiaolong Xu, Ruichao Mo, Xiaochun Yin, Mohanmmad R. Khosravi, Fahimeh Aghaei, Victor Chang, Guangshun Li*

Abstract—The cyber-physical cloud systems (CPCSs) release powerful capability in provisioning the complicated industrial services. Due to the advances of machine learning in attack detection, a wide range of machine-learning applications are involved in industrial CPCSs. However, how to ensure the implementation efficiency of these applications, and meanwhile avoid the privacy disclosure of the datasets due to data acquisition by different operators, remain challenging for the design of the CPCSs. To fill this gap, a privacy-aware deployment method, named PDM, is devised for hosting the machine-learning applications in the industrial CPCSs. In PDM, the machine-learning applications are partitioned as multiple computing tasks with certain execution order, like workflows. Specifically, the deployment problem is formulated as a multi-objective problem for improving the implementation performance and resource utility. Then the most balanced and optimal strategy is selected by leveraging an improved differential evolution technique. Finally, through comprehensive experiments and comparison analysis, PDM is fully evaluated.

Index Terms—CPCSs; Machine learning; Privacy-aware deployment; NSDE

1 Introduction

Cyber physical cloud systems (CPCSs) take full advantage of the strucutre optimazation of the cloud infrastructure to extend the traditional cyber-physical systems (CPSs). The new framework greatly improves the interaction among network physical devices and contributes to performing large-scale data storage and analysis. By means of the advantages of easy scalability and strong capability of CPCSs, service providers manage the industrial applications in the virtual manners and thus provide highly reliable services for a large number of users [1] [2]. Currently, the industrial CPCSs have received extensive attention from various organizations in multiple fields, including smart grid, etc.[3]. With the explosive growth in service scale and type from end-users, the industrial CPCSs are becoming increasingly complicated, intelligent, and autonomous in realizing interactions between the physical devices and the heterogeneous networks. Furthermore, the interactions suffers from the potential network attacks as a result of multi-user acquisition and frequent network communications [4].

Fortunately, machine learning (ML) is an efficient technology to detect the network attacks and develop corresponding defense schemes in advance, so as to improve the security of industrial CPCSs [5]. In particular, the industrial CPCSs deploy machine-learning-based appliations to implement the accurate attack detection and ensure their own security and reliability [6]. Technically the machine-learning based security assessment learns the current state information and the past of the industrial CPCSs. To improve the implementation efficiency, the ML-based applications are partitioned as multiple subtasks to realize parallel and distributed processing with the support of the workflow. In such workflows, each subtask is responsible for every aspect of machine learning, including problem formulation, model construction, and model verification. Besides, each subtask often requires massive amounts of data from the CPCSs or the intermediate data for execution. Machine learning and workflow complement each other to further improve the service performance and security of the industrial CPCSs [7].

- *Xiaolong Xu is with the School of Computer and Software, Nanjing University of Information Science and Technology, Nanjing 210044, China, is with the Facility Horticulture Laboratory of Universities in Shandong, WeiFang University of Science & Technology, ShouGuang 262700, China, is with Jiangsu Collaborative Innovation Center of Atmospheric Environment and Equipment Technology (CICAEET), Nanjing University of Information Science and Technology, Nanjing 210044, China, is with the State Key Laboratory for Novel Software Technology, Nanjing University, Nanjing 210023, China.*
  Email: xixu@ieee.org;
- *Ruichao Mo is with the School of Computer and Software, Nanjing University of Information Science and Technology, Nanjing 210044, China.*
  Email: ruichaomo@gmail.com
- *Xiaochun Yin is with the Facility Horticulture Laboratory of Universities in Shandong, WeiFang University of Science & Technology, ShouGuang 262700, China.*
  Email: xiaochunyin@wfust.edu.cn
- *Mohammad R. Khosravi is with the Department of Computer Engineering, Persian Gulf University, Bushehr 75169, Iran.*
  E-mail: m.khosravi@mehr.pgu.ac.ir
- *Fahimeh Aghaei is with the Electrical and Electronics Engineering Department, Ozyegin University, Orman Sk.34794 Istanbul, Turkey.*
  E-mail: fahimeh.aghaei@ozu.edu.tr
- *Victor Chang is with the School of Computing, Engineering and Digital Technologies, Teesside University, TS1 3BX Middlesbrough, U.K*
  E-mail: V.Chang@tees.ac.uk
- *Guangshun Li is with the Qufu Normal University, Qufu 273165, China.*
  E-mail: Guangshunli@qfnu.edu.cn (Corresponding author)

Manuscript received ; revised

Generally, the implementation efficiency is bound up with the locations of the tasks in the machine-learning workflows (MLWs) and the datasets, thus the internal relationship between the data and the workflow tasks needs to be investigated. Therefore, it is critical to distribute massive data to the storage nodes reasonably in response to data acquisition for MLWs that are distributed in the cloud [8]. In the industrial CPCSs, as there are multiple concurrent machine learning applications to be deployed, it is difficult to guarantee the execution efficiency of all the workflows. The reason is that the generated intermediate data from some computing tasks in the machine-learning workflows are also the source data for the other computing tasks which also affect the execution efficiency of the workflows. Therefore, it is crucial to properly schedule the workflows and data for the CPCSs to the cloud, thereby improving the implementation efficiency of the workflow applications [9].

Meanwhile, in the industrial CPCSs, massive physical devices are equipped to collect data from the end-users in real-time to complete the required services[10]. Accordingly, a large amount of historical data from the end-users will be transmitted in the cloud to assess the security of the systems and achieve the further analysis of the data. But the privacy disclosure problem still exists, since the multiple data operators will access the same data storage nodes in the cloud[11][12]. Generally, when the MLWs used to ensure system security are executed, these private dat[**?**]a will be leveraged by different operations to complete the security assessment of the industrial CPCSs, resulting in the disclosure of the private data.

Furthermore, the energy consumption of the cloud is also increasing with the rising data processing requirements of the industrial CPCSs. In particular, the transmission and processing of massive data consume a lot of energy, when MLWs are executed. Nowadays, the optimization of power consumption of the cloud data centers promotes the healthy and sustainable development of industrial CPCSs and has become the primary part in green computing[13]. On the other hand, to ensure the quality of the service, the dependability and stability of the cloud have also received increasing attention[14]. The load balance of the physical nodes enhances the reliability of the cloud data center, which is an effective measure to ensure service performance, and reduces the possibility of a single node being overloaded or even crashing [15]. Overall, the allocation of computational and storage resources dominates the performance of the machine-learning applications which is determined by the deployment strategies of the workflows.

Based on the above analysis, it is significantly essential to ensure the implementation efficiency of the machine-learning applications, and meanwhile eliminate the potential risk of privacy disclosure of the datasets since data acquisition by different operators, remains challenging for the effective operations of the industrial CPCSs. To satisfy these requirements in the industrial CPCSs, a privacy-aware deployment method, named PDM, is devised for hosting the machine-learning applications and jointly optimizing the data acquisition time, power efficiency and resource utility. Specifically, the key contributions are four folds.

- The machine-learning applications in the industrial CPCSs are partitioned as multiple subtasks by workflow technology.
- The non-dominated sorting differential evolution (NSDE) technique is fully investigated to obtain the deployment strategies for the machine-learning applications.
- The most balanced and optimal deployment strategy is selected through the utility value evaluation by utilizing the simple additive weighting (SAW) and multiple criteria decision making (MCDM) techniques.
- Extensive experiments and comparison analysis are conducted to demonstrate the performance of PDM.

The rest of this paper is organized as follows. Section 2 introduces the related work. Section 3 designs a CPCSs service framework with fat-tree. Section 4 elaborates the system model and formalize the goal fitness function. Section 5 presents the design of PDM. Section 6 examines the performance of PDM experimentally. Section 7 draws the conclusion and introduces the future work.

2 Related Work

Driven by the rapid advancement of cloud computing, it is becoming more and more common to send data generated by CPCSs during the operation process to the cloud data platform for processing, which improves the operating performance of CPCSs. Besides, optimizing application tasks and data deployment strategy achieves contribute to reducing data acquisition time, thereby further improving the operating performance of CPCSs. Meanwhile, data privacy is one of the most concerned problems in CPCSs, which brings a lot of attention from academia and industry.

In recent years, there are a series of research works to study the task scheduling problem in CPCSs, an energy-aware task scheduling algorithm based on the greedy algorithm is proposed for the heterogeneous cloud in [16]. To realize the implementation of scientific workflows, a novel energy-efficient resource allocation scheme was proposed in response to the expanding cloud [17]. Besides, to ensure the stability and security of many applications and data in the cloud, it is also wise to maintain the load balance of nodes. Liu et al. [18] proposed a duplicate placement approach, aiming to optimize data transmission efficiency, enhance the parallel placement performance and improve the load balance. Zhao et al. [19] comprehensively investigated the data acquisition time and the load balance, and a data placement method.

On the other hand, data privacy of the machine learning application in the cloud has attracted massive attention and plenty of privacy-preservation methods are presented to address the security problems. In [20], the k-nearest neighbor algorithm and local-sensitive hashing were utilized to keep the private information of the images storing in the data center and guarantee the traceability of images. Besides, a protocol based on a watermark was proposed to avoid unauthorized copying or modification. In [21], Li et al. implemented fully homomorphic encryption with different keys to conduct data encryption to guarantee the security of training data. Then, the cost of deep learning in cloud computing was analyzed to ensure the economy and security of the proposed method. In [22], a privacy-preservation communication scheme was presented to ensure data safety when the vehicles transmit data to the cloud. Mollah et al. [23] introduced multiple security problems in the mobile cloud computing , especially of data security and location privacy. In [24], an intrusion detection method with machine learning was proposed which utilized physical knowledge and advanced machine learning technology to extract significant features from a large number of noisy physical measurements, thus ensuring the efficiency of attack detection. In [25], a new adaptive management architecture was designed to realize safety and security in CPCs. In particular, an adaptive controller was developed to ensure that the closed-loop dynamic system has consistent limit boundedness.

To the best of our knowledge, there are very few studies to ensure the implementation efficiency of the machine-learning applications, and prevent the privacy disclosure of the datasets due to data acquisition by different operators. To address these challenges in the industrial CPCSs, a privacy-aware deployment method, named PDM, is proposed in this paper for executing the machine-learning applications and jointly optimizing the data acquisition time, power efficiency and resource utilization.

3 Fat-Tree Supported Infrastructure Construction for CPCSs

The Fat-Tree Supported Infrastructure Construction for CPCSs is provided in Appendix A.

4 Multi-Objective Problem Definition

The multi-objective problem is defined through the analysis of the resource utility of the CPCSs, including power consumption and load balance, and the implementation performance of the applications which mainly refers to the data acquisition time.

4.1 Service Framework for CPCSs

Fig.1 shows a service framework for industrial CPCSs. The system first collects data from physical devices, such as vehicles, sensors, traffic lights, industrial equipment. Then, the data is sent to the cloud platform via bidirectional or monodirectional and used for storage, processing, and analysis. By deploying sub-tasks of the MLKs on different computing nodes in the cloud to ensure the efficiency of the MLKs and data privacy, strong data support is established to realize applications such as intelligent transportation systems, smart homes and intelligent manufacturing for the CPCSs.

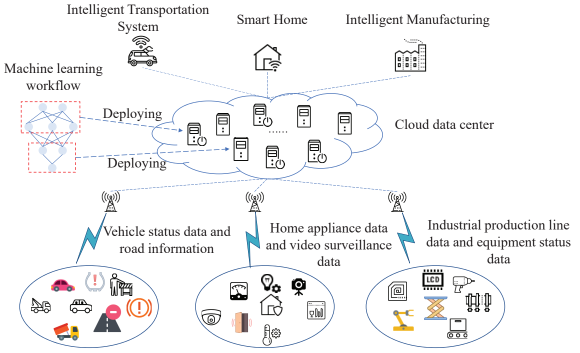

Intelligent Transportation System
Smart Home
Intelligent Manufacturing
Machine learning workflow
Deploying
Deploying
Cloud data center
Vehicle status data and road information
Home appliance data and video surveillance data
Industrial production line data and equipment status data

Fig. 1: Service framework for CPCSs.

4.2 Machine-Learning Workflow Model

Suppose that a machine-learning workflow includes WN tasks and DN workflow original source data. The task set is represented by $TS = \{t_0,\ ...,\ t_{wn},\ ...,\ t_{WN-1}\}$. Besides, the workflow original source data collection is represented by $OD = \{od_0,\ ...,\ od_{dn},\ ...,\ od_{DN-1}\}$. After task execution, an intermediate data set $MD = \{md_0,\ ...,\ md_{wn},\ ...,\ md_{WN-1}\}$ is generated, and the intermediate data md_{wn} of each task t_{wn} is the original source data for its successor tasks. In addition, the historical data sets $HD = \{hd_0,\ ...,\ hd_p,\ ...,\ hd_{P-1}\}$ that have been placed in the cloud should be considered for data acquisition. Therefore, the relations among all tasks and their respective data sources are specified as $\beta = \{\beta_0,\ ...,\ \beta_{wn},\ ...,\ \beta_{WN-1}\}$, where $\beta_{wn} = \{od_1,\ ...,\ od_i,\ md_1,\ ...,\ md_j,\ hd_1,\ ...,\ hd_k\}$. However, some privacy-conflict datasets cannot be placed on the same node to ensure the data security. So these privacy-conflict datasets can be defined as a set $PD = \{\theta_0,\ ...,\ \theta_s,\ ...,\ \theta_S\}$, where $\theta_s = \{(d_i,\ d_j)|d_i,\ d_j \in \{OD, MD\}\}$ implies that d_i and d_j cannot be placed on the same storage node.

4.3 Data Acquisition Model

In the cloud data center of CPCSs, assume that t_{wn} and the data d required by t_{wn} are placed in v_i and v_j, respectively. Then the relationship between v_i and v_j is determined by $\rho_{i,j}$.

- If v_i and v_j are placed in the same node, $\rho_{i,j}$ is set to 0.
- If v_i and v_j are connected to the uniform switch, $\rho_{i,j}$ is set to 1.
- If v_i and v_j are located in the same pod, but they are not connected to the uniform switch, $\rho_{i,j}$ is set to 2.
- If v_i and v_j are not located in the uniform pod, $\rho_{i,j}$ is set to 3.

Suppose that the node scale in the cloud is L, then the relationship between any nodes is represented as a 2-dimensional tensor $\rho_{i,j}$ $(i, j \in \{1, 2, 3..., L\})$, and $\rho_{i,j} = \{0, 1, 2, 3\}$.

With the node distribution value $\rho_{i,j}$ for each pair of nodes, acquisition time T_{ac} of t_{wn} to acquire $d(d \in \beta_{wn})$is accordingly expressed as

$$\mathrm{T}_{ac} = \begin{cases} \rho_{i,j}, \rho_{i,j} = 0 \\ (\rho_{i,j} + 1) \cdot d/B_{he}, \rho_{i,j} = 1 \\ \rho_{i,j} \cdot (d/B_{he} + d/B_{ea}), \rho_{i,j} = 2 \\ (\rho_{i,j} - 1) \cdot (d/B_{he} + d/B_{ea} + d/B_{ac}), \rho_{i,j} = 3 \end{cases} \tag{1}$$

where B_{he}, B_{ea} and B_{ac} are the transmission bandwidth between node and edge switch, the transmission bandwidth across edge and aggregation layers and the transmission bandwidth across the aggregation and core layers.

The total acquisition time T_{wn} of t_{wn} consists of the acquisition time for acquiring its necessary data β_{wn} and the intermediate result md_{wn}. Thus, T_{wn} of task t_{wn} is expressed by

$$T_{wn} = \sum_{d \in \beta_{wn} \cup md_{wn}} T_{ac}. \tag{2}$$

Let WN be the amount of subtasks in a machine-learning workflow. Then, the acquisition time T_{total} of all sub-tasks in this workflow is represented as

$$T_{total} = \sum_{m=0}^{WN-1} T_{wn}. \tag{3}$$

Since subtasks of the cloud environment obtain the data set they request in a parallel and distributed manner, the average data collection time is evaluated as the main evaluation criterion in this paper. Ultimately, the average data acquisition time T_{avg} is expressed as

$$T_{avg} = T_{total}/WN. \tag{4}$$

4.4 Transmission Power Consumption Model

The consumed power for data extraction and acquisition is mainly generated by the operation of switches at different layers.

Suppose the transmission rate of each switch is trs, and the transmission power is tps. Thus, the transmission time t_{switch} for the task of data size d transmitting across switches and nodes is expressed as

$$t_{switch} = d/trs. \tag{5}$$

Suppose that the number of switches that v_j accesses to the data set on v_i is $NS_{i,j}$. Then the consumed power of all the switches caused by accessing the datasets is expressed as

$$E_{switch} = NS_{i,j} \cdot t_{switch} \cdot tps. \tag{6}$$

The power consumption E_{wn} generated by t_{wn} for obtaining all the requested data is deduced by

$$E_{wn} = \sum_{d \in \beta_{wn} \cup md_{wn}} E_{switch}. \tag{7}$$

Ultimately, the total transmission power consumption E for all the tasks is expressed as

$$E = \sum_{wn=0}^{WN-1} E_{wn}. \tag{8}$$

4.5 Load Balance Model

Suppose that the virtual machine required by t_{wn} and d_{dn} are vm_{wn} and vm_{dn}, respectively. The capacity of v_i and v_j are C_i and C_j. The number of the computing nodes is CS. And correspondingly the number of the storage nodes is SS. Assume that the computing node that executes t_{wn} and the storage node for storing d_n are v_i and v_j, respectively. Thus, $\delta^{cal}_{wn,i} = 1, \delta^{store}_{dn,j} = 1$. Otherwise, $\delta^{cal}_{wn,i} = 0, \delta^{store}_{dn,j} = 0$. Furthermore, the utilization Z^{cal}_i of v_i is expressed by

$$Z_i^c = \sum_{wn=0}^{WN-1} \delta^c_{wn,i} \cdot vm_{wn}/C_i. \tag{9}$$

Besides, the utilization Z^{store}_j of v_j is expressed by

$$Z_j^s = \sum_{dn=0}^{DN-1} \delta^s_{dn,j} \cdot vm_{dn}/C_j. \tag{10}$$

Moreover, $\overline{Z^c}$ is utilized to represent the mean resource usage of the computing nodes in CPCSs which is expressed by

$$\overline{Z^c} = \sum_{i=0}^{CS-1} Z_i^c/CS. \tag{11}$$

Similarly, $\overline{Z^s}$ is lerveraged to represent the mean resource usage of the storage nodes in CPCSs which is expressed by

$$\overline{Z^s} = \sum_{j=0}^{SS-1} Z_j^s/SS. \tag{12}$$

$\widetilde{Z^c}$ and $\widetilde{Z^s}$ are utilized to represent the average utilization difference values for computing nodes and storage nodes which are calculated by

$$\widetilde{Z^c} = \frac{1}{CS} \cdot \sum_{i=0}^{CS-1} (Z_i^c - \overline{Z^c})^2, \tag{13}$$

and

$$\widetilde{Z^s} = \frac{1}{SS} \cdot \sum_{j=0}^{SS-1} (Z_i^s - \overline{Z^s})^2. \tag{14}$$

Finally, the variance $\widetilde{Z}$ of the mean usage of computing nodes and storage nodes is measured by

$$\widetilde{Z} = \frac{1}{2} \cdot (\widetilde{Z^c} + \widetilde{Z^s}). \tag{15}$$

4.6 Objective Functions and Constraint

While solving the privacy-aware deployment and privacy-aware data processing problems of machine learning workflow, this paper aims to minimize the average data acquisition time, power consumption and load balancing of cloud nodes. The optimization objectives is expressed as

$$Min(T_{avg}, \widetilde{Z}, E). \tag{16}$$

Furthermore, the privacy-conflict of different data sets is considered as the optimization goal to avoid privacy leakage problems caused by operating on different data sets. Generally, let $\mu_k = \{d_0,\ d_1,\ d_2,\ ...,\ d_a,\ d_b,\ ...,\ d_{K-1}\}$ be the locations that the all datasets placed on the storage node, where K is the total number of datasets. Then, the constraint relationship between the privacy-conflict datasets is described as a constraint, which is defined by

$$\{d_a,\ d_b\} \notin PD \mid \forall \mu_k \in \mu,\ \forall a, b = 1, 2, ..., K-1|. \tag{17}$$

5 Method Design

5.1 Encoding

The deployment strategies for the MLWs are encoded as the chromesomes at the endcoding phase. Generally, the deployment strategies are represented as $X = \{X^T, X^{MD}, X^{OD}\}$, where $X^T = \{x_0^T, x_1^T, ..., x_m^T, ..., x_{WN-1}^T\}$ represents the deployment strategies with WN tasks. Besides, the deployment strategies of WN creating intermediate data and N workflow original source data are represented by $X^{MD} = \{x_0^{MD}, x_1^{MD}, ..., x_{wn}^{MD}, ..., x_{WN-1}^{MD}\}$ and $X^{OD} = \{x_0^{OD}, x_1^{OD}, ..., x_{dn}^{OD}, ..., x_{DN-1}^{OD}\}$, respectively. The deployment strategy for these data cannot be the same node that stores the historical data in the cloud.

5.2 Objective Functions

To solve the multi-objective optimization problem, it is necessary to jointly optimize the average data implementation efficiency, the total consumed power, and the load balance of the node to achieve the balance of the three objective functions.

Average Data Implementation Efficiency: For the task in TS, Eq.(1) is leveraged to calculate the time for storing the intermediate result, and Eq.(4) is utilized to calculate the implementation efficiency of the required data and the average data implementation efficiency.

Total Consumed Power: For the task in TS, the transmission consumed power of the switch is calculated by Eq.(6) and Eq.(7). Then the consumed power of the switch to perform all tasks is calculated by Eq.(8).

Load Balance: After calculating the load of computing nodes and storage nodes in the cloud, the different values for all these two kinds of nodes are determined. Then, Eq.(15) is utilized to calculate the average load balance variance of each node.

5.3 Privacy-aware Deployment Stategies Acquired by NSDE

5.3.1 *Initialization*

As a genetic algorithm, NSDE generates an initial population before the process of evolution. The parent population is expressed as $X = \{X_0, ..., X_i, ..., X_{CU-1}\}$, where X_i is the i-th chromosome. Suppose that the optimization problem has WD tasks, WD intermediate data, and DN workflow original source data, then X_i is expressed as $\{X_i^T, X_i^{MD}, X_i^{OD}\}$. X_i^T represents the strategies for placing WN tasks, X_i^{MD} represents the strategies for WN data of the intermediate result and X_i^{OD} represents the strategies for placing N original source data of the workflow.

5.3.2 *Mutation*

The operation of mutation is to combine the differential genes of three chromosomes to produce a mutant chromosome. Generally, in Eq.(18), X_a, X_b and X_c are selected as the parent chromsome randomly from X. Then, the genotypes on these three chromosomes are combined to construct a new chromosome with a new genotype. The mutation factor C with values between 0 and 1 are randomly set to increase genotype diversity. Therefore the new chromosome H_i obtained by mutation is expressed by

$$H_i = X_a + C \cdot (X_b - X_c). \tag{18}$$

A mutation population $H = \{H_0, H_1, ..., H_i, ..., H_{CU-1}\}$ is obtained, and the size of the mutation population is still CU.

5.3.3 *Crossover*

To further increase the genetic diversity of the population, the crossover operation is performed on mutation chromosome set H and parent population set X to produce a crossover population. Generally, in Eq.(19), the crossover factor CF with values between 0 and 1 is randomly generated at first. Then, a random number is set as a flag to select the crossover gene $R_{i,j}$. If the random number is greater than CF, the genotype it will select from $X_{i,j}$. Otherwise, the genotype will select from $H_{i,j}$. Therefore, the crossover gene $R_{i,j}$ acquisition is expressed by

$$R_{i,j} = \begin{cases} H_{i,j}, \mathrm{j} = \mathrm{rand}(0, 2\mathit{WN} + \mathit{DN} - 1) || \mathrm{rand}(0,1) \leq \mathit{CF}, \\ X_{i,j}, \mathrm{rand}(0,1) > \mathrm{CF}. \end{cases} \tag{19}$$

Therefore, the crossover population $R = \{R_0, R_1, ..., R_{CU-1}\}$ is acquired, and the population size of R is CU.

5.3.4 *Selection*

In the stage of selection, the chromosome with good genotypes in the crossover population and the parent population need to be selected as a chromosome in the next generation population. Therefore, X and R are combined into $Y = \{Y_0, Y_1, ..., Y_i, ..., Y_{2CU-1}\}$, and the population size of Y is $2CU$. The crowded distance calculation and fast non-dominated sorting operations are performed on Y. When performing fast non-dominant sorting, Y is divided into multiple dominant layers $L_i (i = 0, 1, 2, \dots)$. Since the gene of chromosomes in L_i is better than L_{i+1}, so that all chromosomes in L_{i+1} are completely dominated by all chromosomes in L_i. The chromosomes in L_i are better than L_{i+1} as excellent chromosomes are retained in the offspring. Besides, in the same dominant layer L_i, each chromosome performs crowding distance calculation to preferentially retain chromosomes with better crowding distance to the offspring. Then, the chromosomes in the better dominant layer and those in the same dominant layer with better crowding distance will preferentially retain the offspring X until the size of X is CU. Furthermore, the genetic operations are conducted by NSDE on X until the solution in the population begins to converge or the maximum number of iterations is reached, thereby multiple non-dominated strategies for the MLWs deployment are acquired.

5.4 Optimal privacy-aware deployment strategy

When NSDE terminates, multiple strategies are acquired for the MLWs deployment. The optimal one needs to be selected as the optimal privacy-aware deployment strategy. Therefore, SAW and MCDM are used to compute out the utility values of X_i, thereby realizing the standardization of multiple indicators.

Since each solution has three objective function values, three corresponding weights w_1, w_2 and w_3 are set according to SAW for the three targets, and the sum of tree weights is always equal to 1. The higher the importance of the fitness, the greater the relevant weight is.

Suppose that the average data implementation efficiency, the total consumed power, and the load balance of X_i are expressed by TC^i, EC^i, and UC^i, respectively. Besides, TC^{min}, TC^{max}, EC^{min}, EC^{max}, UC^{min} and UC^{max} represent the minimum and maximum value of the average data implementation efficiency, the total consumed power, and the load balance of all individuals, respectively. Thus, the utility value v_i of X_i is measured by

$$v_i = w_1 \cdot \frac{TC^{\max} - TC^i}{TC^{\max} - TC^{\min}} + w_2 \cdot \frac{EC^{\max} - EC^i}{EC^{\max} - EC^{\min}} + w_3 \cdot \frac{UC^{\max} - UC^i}{UC^{\max} - UC^{\min}}, \tag{20}$$

where $w_1 + w_2 + w_3 = 1$.

Then, the deployment strategy with the maximum utility value is obtained as the optimal deployment strategy for the MLWs.

5.5 Method Overview

As shown in Fig.2, PDM is mainly divided into two stages. In the first stage, the initial population X, crossover population R and mutation population H are obtained. Specifically, the initial population is first generated randomly, and the number of individuals in the initial population is N. If the individuals in the population are not stratified, crossover, mutation and fast non-dominated sort are executed in the initial population. Furthermore, according to (18) and Eq.(19), the crossover population H and the mutant population R are obtained. Finally, the initial population is mixed with the population that has completed genetic manipulation to form Y. In the second stage, the individuals in the current population are evaluated to determine whether the solutions are stratified and dominated by each other. Finally, the optimal solution is obtained through Eq.(20).

6 Experimental Evaluation

6.1 Experimental Setting

In this experiment, parameters B_{he} ,B_{ea}, B_{ac}, r and p of the cloud infrastructure constructed by fat-tree are set as 200kb/s, 300kb/s, 400kb/s, 300kb/s and 5W, respectively. Besides, to demonstrate the effectiveness of PDM, two algorithms are selected for comparative analysis, as follows.

- State-aware placement (SP): The deployment strategy of the subtasks and datasets of the MLWs makes the nodes in the cloud computing center in the state of load balance.
- Time-aware placement (TP) [17]: The main idea of TP is that the MLWs takes the shortest time to retrieve data when it is deployed for the subtasks and data of the MLWs.

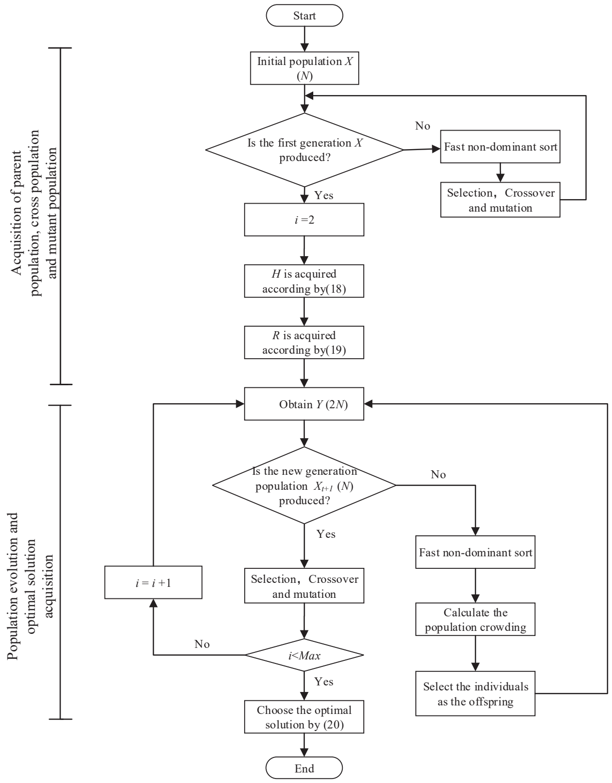

Start
Initial population X (N)
Acquisition of parent population, cross population and mutant population
Is the first generation X produced?
No
Fast non-dominant sort
Selection，Crossover and mutation
Yes
i =2
H is acquired according by(18)
R is acquired according by(19)
Obtain Y (2N)
Population evolution and optimal solution acquisition
Is the new generation population X_{i+1} (N) produced?
No
Yes
Fast non-dominant sort
i = i +1
Selection，Crossover and mutation
Calculate the population crowding
No
i<Max
Yes
Select the individuals as the offspring
Choose the optimal solution by (20)
End

Fig. 2: The flow chart of PDM.

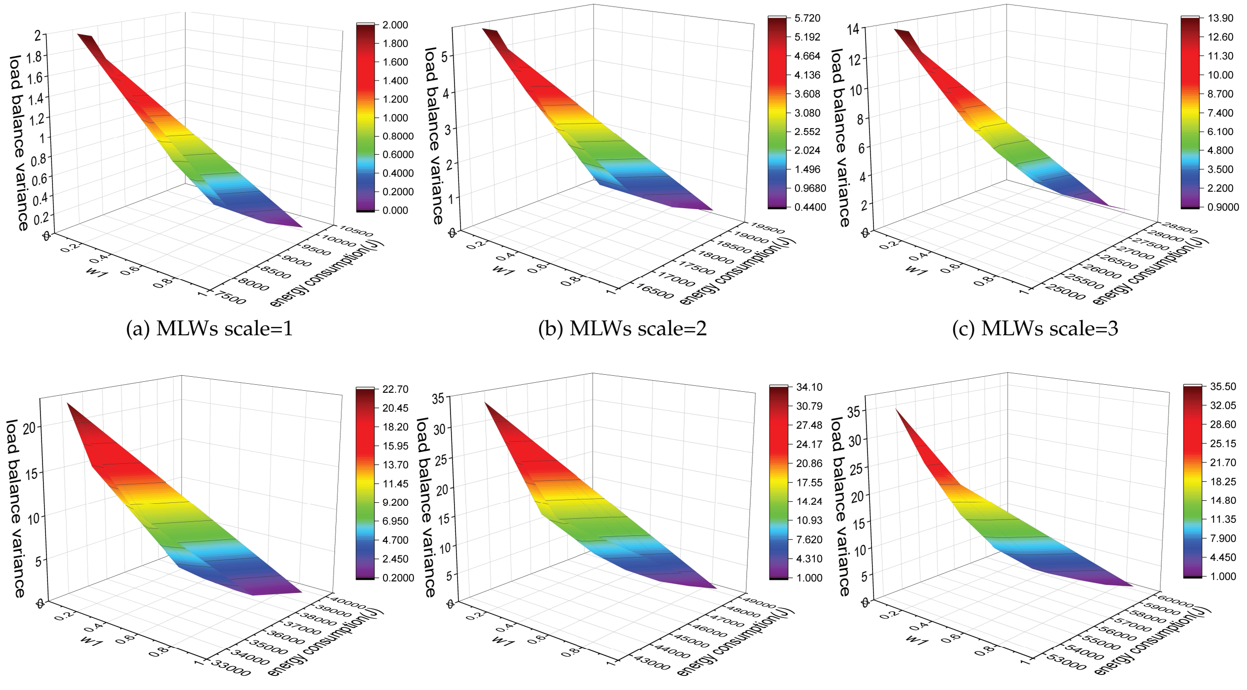
6.2 Impact on the Utility Weights

In the experiment, the changes of utility value caused by the different weights are observed by adjusting the corresponding proportion of three objective function values in Eq.(20). Specifically, when some weight value is changed, the other two weights are set to the same value and remain the same all the time. In addition, the value of w_1 is adjusted to show the impact of w_1 on load balance and energy consumption under different scales of the MLWs. As shown in Fig. 3, with the increase of w_1, the optimization performance of PDM in load balance variance and energy consumption is gradually getting worse. As assumed in Section 5.4, w_1 represents the weight of the average data realization efficiency. Therefore, as the value of w_1 continues to increase, the average data realization efficiency accounts for an increasing proportion in the optimization process, which leads to the poor performance of PDM in terms of load balance variance and energy consumption. Meanwhile, as shown in Fig.3, when the value of w_1 is the same, as the scale of MLWs continues to increase, the value of load balance variance and energy consumption continues to increase.

6.3 Performance Evaluation

6.3.1 *Implementation efficiency of workflow original source data*

As each workflow has one or more original source data for implementation, the implementation efficiency should be evaluated. Fig.4 shows the concrete results for SP, TP and PDM with different scales of workflows. From Fig. 4 we can find that TP is more efficient than SP and PDM



(a) MLWs scale=1

(b) MLWs scale=2

(c) MLWs scale=3

(d) MLWs scale=4

(e) MLWs scale=5

(f) MLWs scale=6

Fig. 3: Results of the weight w_1 on the load balance and the implementation efficiency metrics under different MLW scales.

achieves the best time efficiency among these three methods. We can deduce that some datasets which serve for multiple workflows, choose the proper locations to minimize the overall data implementation efficiency.

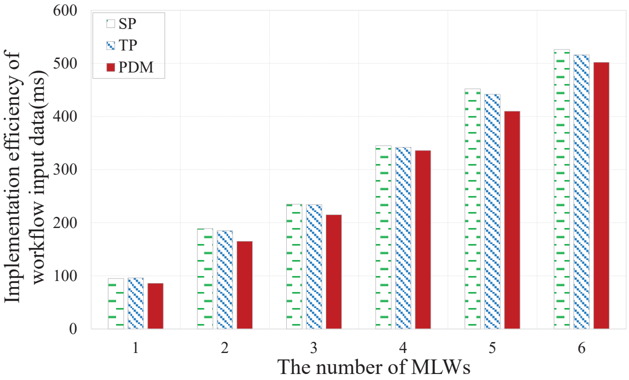


Fig. 4: Results of the implementation efficiency for obtaining the workflow original source data by using SP, TP, and PDM.

6.3.2 Deployment time of generated data

As each workflow could generate multiple datasets, the deployment time for these generated datasets should be evaluated. Fig. 5 shows a comparative analysis of the deployment time of generated data by using SP, TP, and PDM with different scales of workflows. As shown in Fig. 5, because TP and SP do not consider the impact of the deployment time of the data set during the execution process, the deployment time of the data set generated during the execution of TP and SP is relatively close. However, since PDM fully considers the time of data set deployment during the execution process, the time efficiency of PDM is better than that of TP and SP. Besides, as the number of workflows increases, the time performance of PDM deployment workflows to generate data sets is still better. Therefore, PDM further effectively shortens the execution time of the workflow by selecting an appropriate deployment strategy for the data set generated by the workflow.

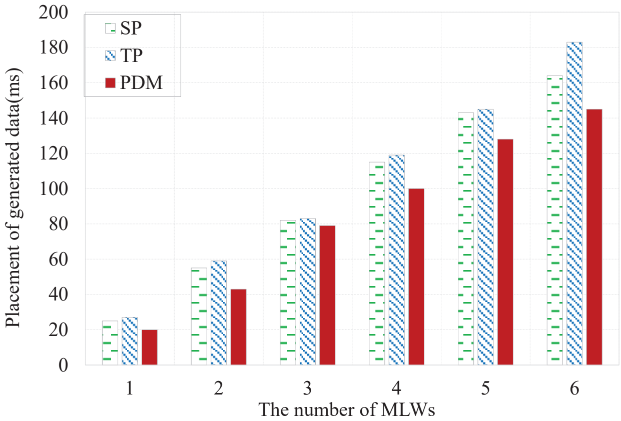


Fig. 5: Results of the deployment time of generated data by using SP, TP, and PDM.

6.3.3 Implementation efficiency of generated and history data

Likewise, the implementation efficiency for these generated datasets and history datasets should be evaluated. Fig. 6 shows a comparative analysis of the implementation efficiency of history data by SP, TP, and PDM with different scales of workflows. In Fig. 6, we can find that TP is more efficient than SP since TP represents the acquisition time aware deployment of the workflows and the datasets, but PDM achieves the best time efficiency. It is deduced that PDM chooses the proper deployment policy for generated datasets and history datasets of the MLWs to minimize the overall data acquisition and deployment time. Furthermore, Fig. 7 shows the comparative analysis of the total time for data acquisition and deployment by SP, TP, and PDM with different scales of workflows, respectively, and the experimental results further prove the reliability of PDM.

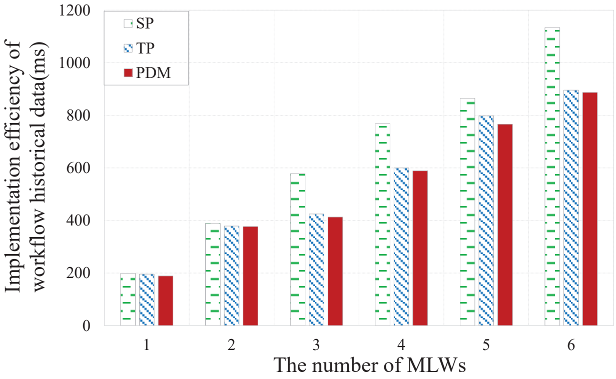


Fig. 6: Results of the implementation efficiency of history data by using SP, TP, and PDM.

6.3.4 *Load balance variance for privacy-aware deployment*

The load balance is an important metric to evaluate the data deployment method. The smaller the load balance variance of the node in the cloud indicates the better the load performance of the cloud. Fig. 8 shows a comparative analysis of the load balance variance for deployment by SP, TP, and PDM. It is intuitive from Fig. 8 that PDM has the lowest load balance variance and SP has the highest load balance variance.

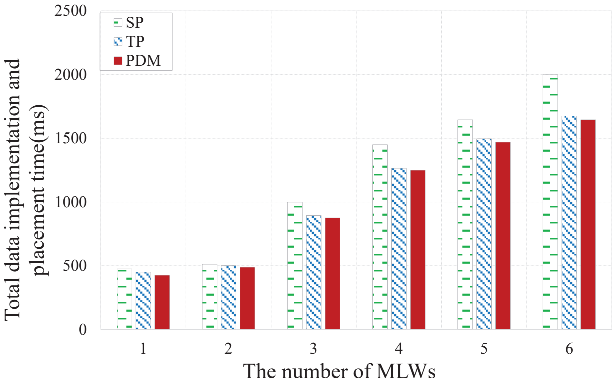


Fig. 7: Results of the total time of data acquisition and deployment by using SP, TP, and PDM.

6.3.5 *Consumed power for privacy-aware deployment*

The consumed power is another important metric to evaluate the efficiency of methods, which includes the consumed power of nodes and switches. Fig. 9 represents the comparison of the consumed power by SP, TP, and PDM. In Fig. 9, when the workflow scale is small, the performance of SP, TP, and PDM is not much different. However, as the scale of the workflow increases, the energy consumption gap between PDM and other methods is also increasing, indicating that when the scale of the workflow is large, PDM saves more energy.

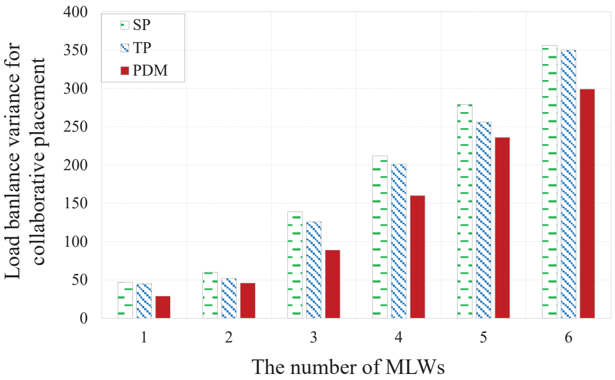


Fig. 8: Results of the load balance variance for privacy-aware deployment by using SP, TP, and PDM.

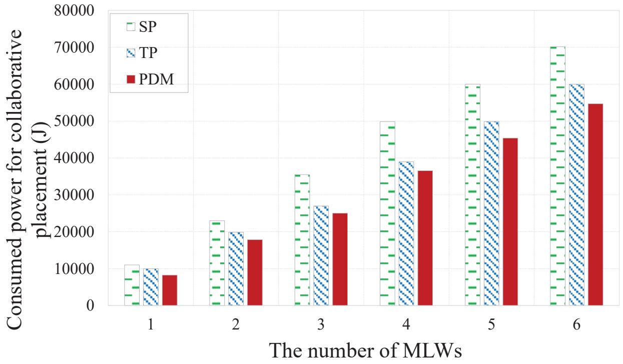


Fig. 9: Results of the consumed power for privacy-aware deployment by using SP, TP, and PDM.

7 Conclusion

A privacy-aware deployment method, named PDM, was proposed in this paper. Initially, the deployment of the MLWs and the data was modeled as a multi-objective optimization problem. Then, NSDE was leveraged to solve this problem. Furthermore, the SAW and MCDM are utilized to find the optimal deployment strategy. Ultimately, a large number of experiments were proceeded to verify the effectiveness of PDM. In future, we hope to apply the PDM to the real scenes and further optimize the performance of PDM.

8 Acknowledgment

This research is supported by the Financial and Science Technology Plan Project of Xinjiang Production and Construction Corps under grant no. 2020DB005. Besides, this work is also supported by the National Natural Science

Foundation of China under grant no. 61702277 and the Priority Academic Program Development of Jiangsu Higher Education Institutions (PAPD) fund.

Appendix

Fat-tree Support Infrstructure Construction for CPCSs

The fat-tree topology is one of a mature network topologies to construct cloud infrastructure. It consists of three layers: core layer, aggregation layer and edge layer[26]. The switches in the aggregation layer and the edge layer form multiple pods so that the switches and nodes can be effectively managed. Therefore, the fat-tree topology is utilized to erect the cloud infrastructure for CPCSs, and provide high-throughput transmission services and effective network communication. There are multiple parallel paths between two physical nodes in the fat-tree network topology, the cloud platform has good fault tolerance, which makes CPCSs provide reliable services.

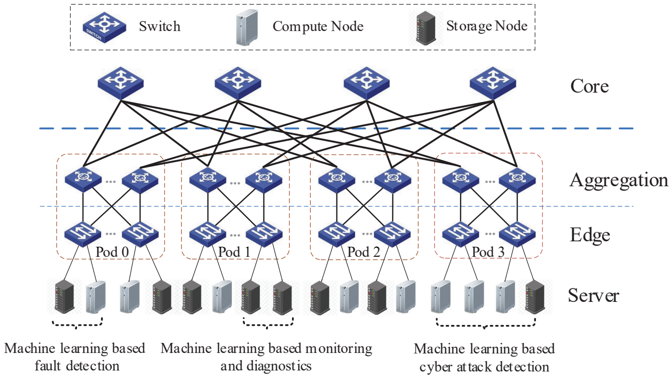

Switch
Compute Node
Storage Node
Core
Aggregation
Edge
Pod 0
Pod 1
Pod 2
Pod 3
Server
Machine learning based fault detection
Machine learning based monitoring and diagnostics
Machine learning based cyber attack detection

Fig. 10: A Fat-tree Support Infrstructure Construction for CPCSs.

Fig. 10 shows a fat-tree supported cloud infrastructure for CPCSs. In the cloud infrastructure, two aggregation switches and two edge switches build a pod, which each supports the implementation of applications such as machine learning based fault detection, monitoring and diagnostics, and cyber-attack detection for CPCSs. Assuming that there are S pods in the cloud, the number of nodes connected to each pod is $(S/2)^2$. Therefore, the number of edge switches and aggregation switches in each pod is $S/2$. Assuming that there are $(S/2)^2$core switches in the cloud, the number of ports on each switch in the network is S, and the $S^3/4$ nodes is able to connect.

References

[1] A. Humayed, J. Lin, F. Li, and B. Luo, "Cyber-physical systems security—a survey," *IEEE Internet of Things Journal*, vol. 4, no. 6, pp. 1802–1831, 2017.

[2] F. Farivar, M. S. Haghighi, A. Jolfaei, and M. Alazab, "Artificial intelligence for detection, estimation, and compensation of malicious attacks in nonlinear cyber-physical systems and industrial iot," *IEEE transactions on industrial informatics*, vol. 16, no. 4, pp. 2716–2725, 2019.

[3] M. Wolf and D. Serpanos, "Safety and security in cyber-physical systems and internet-of-things systems," *Proceedings of the IEEE*, vol. 106, no. 1, pp. 9–20, 2017.

[4] X. Wang, L. T. Yang, Y. Wang, L. Ren, and M. J. Deen, "Adtt: A highly-efficient distributed tensor-train decomposition method for iiot big data," *IEEE Transactions on Industrial Informatics*, 2020. doi: 10.1109/TII.2020.2967768

[5] Z. Zhang, X. Chen, J. Ma, and J. Shen, "Slds: secure and location-sensitive data sharing scheme for cloud-assisted cyber-physical systems," *Future Generation Computer Systems*, vol. 108, pp. 1338–1349, 2020.

[6] Y. Liu, X. Liu, A. Liu, N. N. Xiong, and F. Liu, "A trust computing-based security routing scheme for cyber physical systems," *ACM Transactions on Intelligent Systems and Technology (TIST)*, vol. 10, no. 6, pp. 1–27, 2019.

[7] J. Zhou, H.-N. Dai, and H. Wang, "Lightweight convolution neural networks for mobile edge computing in transportation cyber physical systems," *ACM Transactions on Intelligent Systems and Technology (TIST)*, vol. 10, no. 6, pp. 1–20, 2019.

[8] H. Kim and Y. Kim, "An adaptive data placement strategy in scientific workflows over cloud computing environments," in *NOMS 2018-2018 IEEE/IFIP Network Operations and Management Symposium*. IEEE, 2018, pp. 1–5.

[9] X. Ren, P. London, J. Ziani, and A. Wierman, "Joint data purchasing and data placement in a geo-distributed data market," in *ACM SIGMETRICS Performance Evaluation Review*, vol. 44, no. 1. ACM, 2016, pp. 383–384.

[10] Z. Shen, P. P. Lee, J. Shu, and W. Guo, "Encoding-aware data placement for efficient degraded reads in xor-coded storage systems," in *2016 IEEE 35th Symposium on Reliable Distributed Systems (SRDS)*. IEEE, 2016, pp. 239–248.

[11] J. Zhao, R. Mortier, J. Crowcroft, and L. Wang, "Privacy-preserving machine learning based data analytics on edge devices," in *Proceedings of the 2018 AAAI/ACM Conference on AI, Ethics, and Society*. ACM, 2018, pp. 341–346.

[12] X. Wang, L. T. Yang, L. Song, H. Wang, L. Ren, and J. Deen, "A tensor-based multi-attributes visual feature recognition method for industrial intelligence," *IEEE Transactions on Industrial Informatics*, 2020. doi: 10.1109/TII.2020.2999901

[13] R. Xiong, J. Luo, and F. Dong, "Optimizing data placement in heterogeneous hadoop clusters," *Cluster Computing*, vol. 18, no. 4, pp. 1465–1480, 2015.

[14] X. Xu, Q. Liu, Y. Luo, K. Peng, X. Zhang, S. Meng, and L. Qi, "A computation offloading method over big data for iot-enabled cloud-edge computing," *Future Generation Computer Systems*, 2018.

[15] M. Gaggero and L. Caviglione, "Model predictive control for energy-efficient, quality-aware, and secure virtual machine placement," *IEEE Transactions on Automation Science and Engineering*, no. 99, pp. 1–13, 2018.

[16] S. Zhang, B. Wang, B. Zhao, and T. Jing, "An energy-aware task scheduling algorithm for a heterogeneous data center," in *Trust, Security and Privacy in Computing and Communications*, 2013.

[17] X. Xu, W. Dou, X. Zhang, and J. Chen, "Enreal: An energy-aware resource allocation method for scientific workflow executions in cloud environment," *IEEE Transactions on Cloud Computing*, vol. 4, no. 2, pp. 166–179, 2016.

[18] L. Liu, J. Song, H. Wang, and P. Lv, "Brps: A big data placement strategy for data intensive applications," in *IEEE International Conference on Data Mining Workshops*, 2017.

[19] E. D. Zhao, Y. Q. Qi, X. X. Xiang, and C. Yi, "A data placement strategy based on genetic algorithm for scientific workflows," in *Eighth International Conference on Computational Intelligence & Security*, 2013.

[20] Z. Xia, X. Wang, L. Zhang, Z. Qin, X. Sun, and K. Ren, "A privacy-preserving and copy-deterrence content-based image retrieval scheme in cloud computing," *IEEE Transactions on Information Forensics and Security*, vol. 11, no. 11, pp. 2594–2608, 2016.

[21] P. Li, J. Li, Z. Huang, T. Li, C.-Z. Gao, S.-M. Yiu, and K. Chen, "Multi-key privacy-preserving deep learning in cloud computing," *Future Generation Computer Systems*, vol. 74, pp. 76–85, 2017.

[22] L. Zhang, X. Men, K.-K. R. Choo, Y. Zhang, and F. Dai, "Privacy-preserving cloud establishment and data dissemination scheme for vehicular cloud," *IEEE Transactions on Dependable and Secure Computing*, 2018.

[23] M. B. Mollah, M. A. K. Azad, and A. Vasilakos, "Security and privacy challenges in mobile cloud computing: Survey and way ahead," *Journal of Network and Computer Applications*, vol. 84, pp. 38–54, 2017.

[24] W. Yan, L. K. Mestha, and M. Abbaszadeh, "Attack detection for

securing cyber physical systems," *IEEE Internet of Things Journal*, vol. 6, no. 5, pp. 8471–8481, 2019.

[25] X. Jin, W. M. Haddad, and T. Yucelen, "An adaptive control architecture for mitigating sensor and actuator attacks in cyber-physical systems," *IEEE Transactions on Automatic Control*, vol. 62, no. 11, pp. 6058–6064, 2017.

[26] X. Xu, S. Fu, L. Qi, X. Zhang, Q. Liu, Q. He, and S. Li, "An iot-oriented data placement method with privacy preservation in cloud environment," *Journal of Network and Computer Applications*, vol. 124, pp. 148–157, 2018.