IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received February 13, 2020, accepted February 27, 2020, date of publication March 2, 2020, date of current version March 11, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.2977643

Sensing Cloud Computing in Internet of Things: A Novel Data Scheduling Optimization Algorithm

ZEYU SUN[1,2], ZHIGUO LV[1], HUIHUI WANG[3], (Senior Member, IEEE), ZHIXIAN LI[1], FUQIAN JIA[2], AND CHUNXIAO LAI[2]

[1]School of Computer Science and Information Engineering, Luoyang Institute of Science and Technology, Luoyang 471023, China
[2]School of Information Engineering, Henan Institute of Science and Technology, Xinxiang 453003, China
[3]Department of Engineering, Jacksonville University, Jacksonville, FL 32211, USA

Corresponding author: Zhiguo Lv (lzg96wl@163.com)

This work was supported in part by the National Natural Science Foundation of China under Grant U1604149, in part by the Henan Province Education Department Cultivation Young Key Teachers in University under Grant 2016GGJS-158, in part by the Luoyang Institute of Science and Technology High-Level Research Start Foundation under Grant 2017BZ07, in part by the Henan Province Education Department Natural Science Foundation under Grant 19A520006, in part by the Key Science and Technology Program of Henan Province under Grant 192102210249 and Grant 192102210116, and in part by the Scientific Research Fund of Hunan Provincial Education Department under Grant 19A275.

ABSTRACT In order to addressing the issues of data matching deviation and load imbalance during the data scheduling process of the Internet of Things, Sensing Cloud Computing in IoT: A Novel Data Scheduling Optimization (SCC-DSO) Algorithm is proposed in this paper. First, according to the processing capacity of the IoT working node, a data placement algorithm is designed to reasonably place the input data of the job. Second, the data scheduling queue is optimized based on the data block storage location information to reduce non-local execution of data scheduling. Furthermore, a data prefetching method is designed to pull the data required for non-local data scheduling in advance, and shorten the waiting time of the task for input data. Finally, simulation experiment evaluated by the task localization execution rate and response time is performed. The effectiveness and stability of the algorithm is verified compared with other algorithms.

INDEX TERMS Sensing-cloud computing, Internet of Things, data scheduling, optimization algorithm.

I. INTRODUCTION

The data-intensive computing model brings great challenges to traditional grid computing [1]–[4]. In the IoT computing environment, computing nodes and storage nodes are independent of each other and interconnected through a high-speed network [5]–[8]. This mode has many advantages, for example the storage and computing are independent to facilitate the upgrade of the two parts of the resource, and the storage system can be seamlessly connected to different computing systems. However, in the era of big data, facing the processing of massive data, the grid architecture with separate storage and computing may make network transmission a bottleneck for system performance [9]–[11]. In an IoT cluster, working nodes serve two roles: compute nodes and storage nodes. This architecture brings great flexibility to data parallel processing. Tasks submitted by users can be scheduled for execution at the node where the data is located, that is, local execution of the task, thereby avoiding the transmission of task input data across the network and improving System network resource utilization [12]– [15]. The Internet of Things, as a typical example of big data processing systems, is mainly used to perform data-intensive tasks. In the face of huge amounts of data, large-scale data transmission in the network will seriously degrade system performance. Therefore, data locality research on the IoT cluster system is of great significance for improving task execution efficiency and cluster resource utilization.

In order to improve the reliability of data, most IoT clusters adopt a redundant storage strategy. When data is stored in the system, multiple copies are created and stored on different working nodes. This strategy improves the disaster preparedness of the file system, and also balances the load of each working node in the cluster.

However, this data placement strategy easily leads to non-local execution of data scheduling [16]–[19]. In addition, the input data for data scheduling in the Internet of Things system usually has the same size as a Distributed File System

The associate editor coordinating the review of this manuscript and approving it for publication was Lu Liu.

(HDFS) data block. During the submission process of the job, the schedulers creates a Task queue, and establish task-to-data block mapping for each data schedule in the queue [20]–[24]. For a multi-copy storage strategy, the same data schedule may appear in multiple task queues. During the job executing process, when a working node has disengaged resources, it sends a task request to the scheduler. After the scheduler accepts the request, it searches for a local task for execution according to the task list of the working node. If the search fails, the scheduler looks for the unscheduled data schedule from the task linked list of other nodes followed the same rack first and then different racks, resulting in non-local execution of the data schedule.

II. RELATED WORK

Paper [25] aimed at solving the problem of how to optimize the allocation of system resources during the operation of the Internet of Things to reduce the operating cost of the job. Based on the load of the job input and the available system resources, a cost model for the job was established. Based on this model, system performance is optimized. In this model, the execution of data scheduling is divided into three stages: reading data, executing functions, and writing data. The three-phase task execution cost is described by a cost function. The cost of the final data scheduling is a linear combination of the above three cost functions, and the cost function of the data task is established in the same way. Based on the cost model, the author optimizes the allocation of resources under a certain budget.

Paper [26] established a prediction model of job execution time named Reservation FirstFit and Feedback Distribution (RF-FD) based on configuration parameters of the IoT cluster. The model includes three parts: job performance measurement, model establishment and prediction. First, the job execution time is counted under different system configuration parameters. Then, a multiple linear regression model is established between the configuration parameters and the job execution time. Finally, the execution time of the job is predicted based on the established performance model. This model only considers the maximum data scheduling concurrency and data task concurrency of each node.

A method is proposed in Paper [27] (architecture for data synchronization based on fog computing, RSYNC) which studies the effect of cluster configuration parameters on the performance of IoT operations, and builds a performance prediction model based on it. An automatic parameter tuning method is adopted in paper [28] for IoT cluster systems to adjust the configuration parameters from three layers: the application layer, the Internet of Things layer, and the resource layer according to the MAPE-K cycle. In this method, the author establishes a prediction model about networked job performance. This model only considers the impact of three factors, such as the number of job input files, the size of the input data, and the number of nodes, on job performance. It is pointed out in paper [29] that the optimization choices and forecasting the demand of resources of IoT cluster configuration brings difficulties to users. Then, a smart computing-based IoT performance forecasting model is proposed. During the operation of the job, different numbers of data tasks are set, and the node utilization of the job in each configuration is tested, and a polynomial regression prediction models is established based on this. The limitation of existing performance prediction methods is that most models only consider the impact from the cluster configuration parameters to job execution performance. However, the impact from the current resource utilization of the cluster to job performance is rarely considered. In addition, these existing IoT job performance prediction models are mostly oriented to homogeneous IoT clusters, and the prediction accuracy in heterogeneous environments is not high.

In addition, due to the numerous parameters of the Internet of Things, the time load required to select the configuration parameters suitable for the prediction model is heavy. The paper [30] analyzed the relationship between user service level agreements and service nodes. Constrained energy consumption model based on the reliability of the system is given. Under the premise of meeting this agreement, the load power of the service node is reduced and the operating cost is saved. Paper [31] adopted (Dynamic voltage and frequency scaling, DVFS) technology to dynamically adjust the power of the working nodes to reduce energy consumption. This method first establishes the system node usage rate model under load balancing conditions, and then uses feedback theory to control the data usage rate to reduce the energy loss of the system while ensuring the cluster node load balance. The paper [32] established a power distribution optimization model based on the queuing theory to address the issue of how to reasonably allocate the power of each node in the cluster on the premise of ensuring node performance. Experimental results show that different application scenarios have different power distribution strategies. Under the constraints of cost, the performance of the entire cluster may not be optimal when the working node is running at the maximum power consumption mode. The power distribution strategy is affected by factors such as specific application characteristics and working node processor performance characteristics.

In addition, many researchers have developed distributed systems based on energy awareness and virtualization technology to optimize energy consumption in data center cloud computing environments [33]– [36]. In order to improve the utilization of resources in the cluster system, part of the research work uses virtual machine scheduling and migration technology to aggregate virtual machines on the premise of not reducing user service quality, thereby saving system resources and improving system energy utilization. Whether it is a task scheduling strategy or a virtualization-based energy consumption optimization strategy, the key issue is to understand the characteristics of energy consumption when different loads are running in a cluster environment. To this end, the paper [37] established a linear regression model of system energy consumption based on data collected by hardware counters inside the system. The paper [38] established

a virtual machine energy consumption prediction model in order to optimize the energy utilization rate during virtual machine aggregation. The limitations of the existing methods are that the factors considered by most energy consumption prediction models are not comprehensive and the accuracy of the prediction is not high. In addition, the choice of factors affecting energy consumption in the model remains at the empirical level.

This paper focuses on the problem of local optimization of data during the execution of data scheduling in IoT clusters. This method considers three factors that affect the efficiency of localization of data scheduling: i) where data scheduling input data is placed, ii) data scheduling allocation, and iii) the dynamic characteristics of the system when the data scheduling is running. The method is mainly divided into three parts:

Firstly, the processing capacity of the working nodes is evaluated according to the performance prediction model proposed in section II, and the corresponding number of data blocks is allocated according to the size of the processing capacity;

Secondly, according to the metadata information of the data block stored by each working node, the mapping relationship between the data schedule and the data block is generated and the allocation queue of the data schedule is optimized.

Finally, during the data scheduling execution process, in order to avoid the non-local execution of data scheduling caused by the prediction error of the performance prediction model, a data pre-fetching algorithm is designed to pre-fetch the input data required for data scheduling to the working node where the data scheduling is located by using the characteristics of parallel execution of computing and network transmission to reduce the waiting time of tasks for input data.

III. DATA BLOCK PLACEMENT METHOD

A. RACK-AWARE BASED DATE PLACEMENT STRATEGIES

The file system in the IoT cluster uses a distributed storage method, where each file is cut into data blocks of equal size and stored on different working nodes [38]–[41]. In order to improve the reliability of the file system and the ability of concurrent access, each data block has multiple copies, and these copies are stored on multiple nodes to prevent data access failure caused by a single node failure. Different file system copy placement strategies are different. For example, HDFS file systems in IoT clusters use rack-aware data copy placement strategies [42], [43]. When the client uploads a file, the file system first places the data block on the node where the client is located, and then uses Pipline to store the second copy on other nodes in the same rack as the node where the client is located. Finally, the third copy is stored on those nodes with different racks compared with the node where the client is located.

Figure 1 illustrates the rack-aware data block placement process. The rack-aware data block placement method has the following advantages: first, the strategy of storing two copies

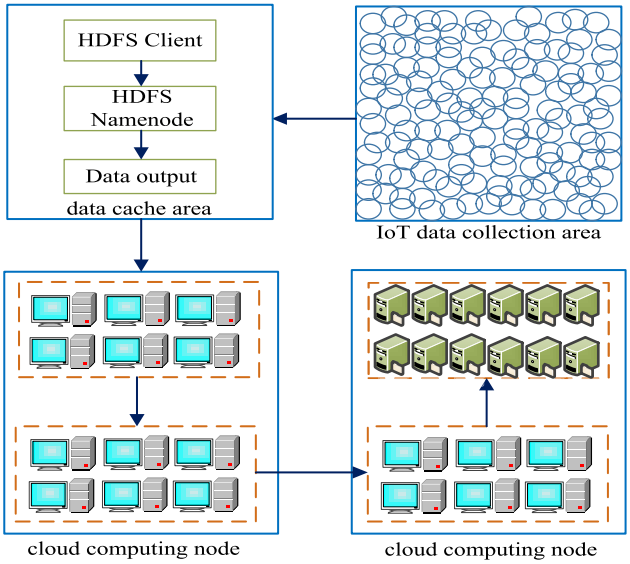


FIGURE 1. Data storage process in the perception cloud architecture.

in the same rack reduces read and write overhead between racks; second, storing copies in different racks avoids failure caused by rack failures. In addition, during the node selection process, the storage capacity of the working nodes is also considered, and the data blocks are stored as evenly as possible in each storage node to ensure the load balance of the system during the running process.

According to the above analysis, it can be seen that the rack-aware copy strategy is a data storage strategy for a homogeneous IoT cluster. It takes into account the storage capacity and stores the data evenly to ensure the load balance of the system [44], [45]. In heterogeneous IoT clusters, the processing capabilities between working nodes are different, and there may be huge differences in performance between different working nodes. If a storage strategy that only considers storage capacity is adopted, then the working nodes with weak processing capacity become the performance bottleneck of the operation. It causes non-local execution of data scheduling and imbalance in system load. As a result, the system performance is reduced.

B. HETEROGENEOUS SENSING DATA PLACEMENT ALGORITHM

In IoT clusters, the execution time of data scheduling of the same input data size on different nodes is different. The performance of the working nodes in processing IoT tasks is affected by several factors, such as the node data frequency, memory size, and disk read/write speeds [46], [47]. In addition, the execution time of the task is also related to the characteristics of the task itself, such as the size of the input data, whether it meets locality, and so on. When the data scheduling does not satisfy the data locality, the input data needs to be transmitted from other nodes in the cluster through the network. In this case, the execution time is also affected by

the network bandwidth [48]– [50]. The purpose of measuring the processing capacity of working nodes is to solve the placement scheme of data blocks. Therefore, the processing capacity of nodes discussed in this section mainly refers to the performance of nodes when data scheduling meets locality.

Definition 1: For an application of the Internet of Things system, if the input file size is *Input* (APP_i), the number of data blocks in the input file can be obtained by the following formula:

$$B = \frac{\text{Input}\ (APP_i)}{b} \tag{1}$$

where b is the storage space occupied by a single data block in the IoT file system.

Definition 2: For the node $Node_i$ in the IoT cluster, suppose that any data scheduling map_k in an application App_j is scheduled to be executed by the scheduler. The input data size of map_k is m and the data locality is met, then the calculation rate for data scheduling in the node $Node_i$ coresponding to the application App_j can be obtained by the following formula:

$$P\left(Node_i, App_j, map_k\right) = \frac{m_k}{t_k} \tag{2}$$

where t_k is the execution time of data schedules map_k on $Node_i$.

Definition 3: For the node $Node_i$ in the IoT cluster, if any data scheduling map_k in an application App_j is scheduled to be executed by the scheduler, the running time of the task on the node $Node_i$ can be obtained by the following formula:

$$\begin{aligned} &T\left(Node_i, App_j, map_k\right) \\ &= \sum_{s=1}^{S} \sigma_s K\left(v\left(map_k\right), v\left(map_s\right)\right) + a \end{aligned} \tag{3}$$

In formula (3), $v(map_k)$ is the feature vector of data scheduling, which includes the hardware characteristics and load characteristics of the cluster nodes; $K(v(map_k),v(map_s))$ is the Gaussian radial basis function; a is the deviation.

From the analysis in Section III, we know that the running time of the Internet of Things job is composed of two parts: node running time and data running time. When the job is in the node execution phase, data scheduling run on the working nodes in the IoT cluster in parallel method. The node running time depends on the completion time of the working node that performs the data scheduling with the longest time. Therefore, reducing the load of the working node can reduce overall running time of the network. In the ideal case, when the data schedule assigned by each working node satisfies data locality, and each working node can complete the assigned data schedule at the same time, the job can obtain the minimum completion time during the network operation phase. Suppose that the time taken by the application IoT application APP_i to run data scheduling map_k on node $Node_j$ is$t(Node_j,APP_i,\ map_k)$, then for an IoT cluster consisting of$n$ nodes, the optimal data block placement solution can be

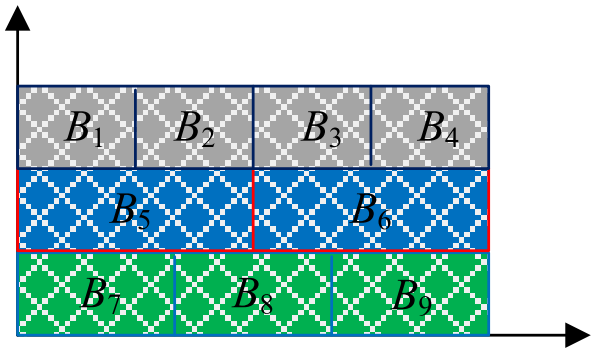


FIGURE 2. Optimal data block placement.

equivalent to solving the following optimization problems:

$$\begin{aligned} &min\left(max\left\{\sum_{k=1}^{f(j)} t\left(Node_j, App_i, map_k\right)\right\}\right) \\ &s.t.\ \ 1 \le j \le n; \quad \sum_{j=1}^{n} f\ (j) = B \end{aligned} \tag{4}$$

In formula (4), $f(j)$ represents the number of data blocks stored in the node $Node_j$; B represents the total number of input data blocks of the application App_i.

$$\begin{cases} T\left(Node_i, App_p, map_q\right) \times f\ (i) \\ = T\left(Node_j, App_p, map_q\right) \times f\ (j) \\ 1 \le i, j \le n\ i \ne j, \sum_{i=1}^{n} f\ (i) = B \end{cases} \tag{5}$$

The precondition of the formula (4) is that the degree of copy of the data block is 1. Assume that there are three working nodes in the cluster with different processing capabilities. Ideally, the data block placement and data scheduling execution time of each node corresponding to the optimal solution of the optimization problem represented by formula (4) is shown in Figure 2. From Figure 2 we can see that, under the condition of a single copy, the best data block strategy is to be able to satisfy each node to complete its assigned data scheduling at the same time. Based on the above analysis, the optimization problem shown in formula (4) can be transformed into the solution of the following problems:

C. DATA PLACEMENT ALGORITHM DATA SCHEDULING QUEUE OPTIMIZATION

In the perceptual cloud computing system, the input data of the job is stored on the working nodes in a multi-copy block storage method. During the job initialization phase, the scheduler builds a data scheduling queue that satisfies the locality of data according to the distribution of data blocks stored in the cluster [8], [51], [52]. The multi-copy storage mechanism causes the scheduling of the same input data in the data scheduling queues on different nodes. The main task of optimizing the data scheduling queue is to adjust the position of each data schedule in the queue based on the data block allocation result generated by the SCC-DSO algorithm,

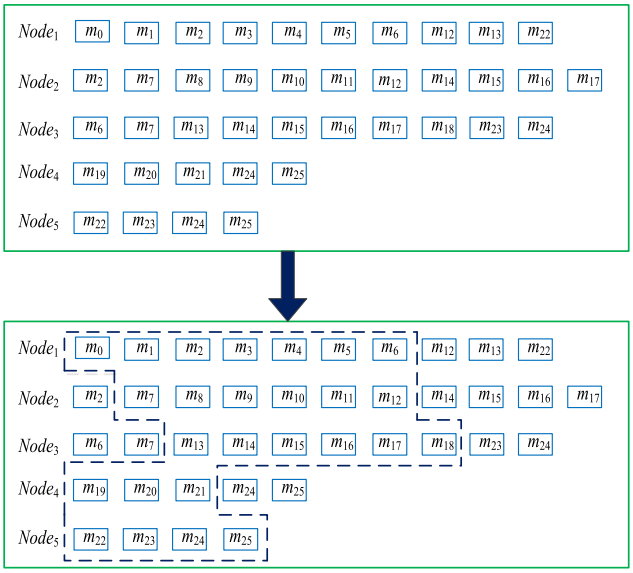


FIGURE 3. Data scheduling queue optimization.

and to generate an optimized queue for data scheduling, so that the assignment of tasks among the working nodes does not affect each other and improve data scheduled local execution efficiency. In order to describe the process of data scheduling queue pre-processing and pre-allocation, assume that the IoT cluster is composed of 5 nodes, and the input data of a job is composed of 26 data blocks. The degree of data block replication in the file system is 2, and the data block distribution obtained by the SCC-DSO algorithm is: $Node_1 = \{m_0, m_1, m_2, m_3, m_4, m_5\}$, $Node_2 = \{m_6, m_7, m_8, m_9, m_{10}, m_{11}\}$, $Node_3 = \{m_{12}, m_{13}, m_{14}, m_{15}, m_{16}, m_{17}\}$, $Node_4 = \{m_{18}, m_{19}, m_{20}, m_{21}\}$, $Node_5 = \{m_{22}, m_{23}, m_{24}, m_{25}\}$. It is described in Figure 3 that the queues before and after the initial data scheduling of each working node.

As shown in Figure 3, after the initial task queue is optimized, the task allocation of each node does not affect each other. Because the SCC-DSO algorithm allocates data blocks according to the processing capacity of the working nodes, once the working nodes have free computing resources, they send task requests to the scheduler, and the scheduler allocates data scheduling according to the optimized task queue, thereby improving the locality of the data scheduling.

In order to further optimize the scheduling queue, this paper introduces the ant colony algorithm of artificial intelligence. For the data scheduling model, we have introduced 4 degree functions, which are delay functions: Delay (e): $E \to R^+$; delay jitter function: Delay-jit (e): $E \to R^+$; cost function: cost (e): $E \to R^+$; and Loss-packet (e): $V \to R^+$. Loss or damage may occur during data scheduling. If packet loss or damage is high, the data will not be complete.

$$\text{delay}\left(P_T\left(s,u\right)\right) = \sum_{e \in P_t(s,u)} \text{delay}\left(e\right) + \sum_{n \in P_T(s,u)} \text{delay}\left(n\right) \tag{6}$$

$$\text{cost}\left(T\left(s,M\right)\right) = \sum_{e \in P_T(s,u)} \text{cost}\left(e\right) + \sum_{n \in P_T(s,u)} \text{cost}\left(n\right) \tag{7}$$

$$\text{bandwidth}\left(P_T\left(s,u\right)\right) = \min\left\{\text{bandwidth}\left(e\right), n \in P_T\left(s,u\right)\right\} \tag{8}$$

$$\text{delay-jit}\left(P_T\left(s,u\right)\right) = \sum_{e \in P_t(s,u)} \text{delay-jit}\left(e\right) + \sum_{n \in P_T(s,u)} \text{delay-jit}\left(n\right) \tag{9}$$

$$\text{loss-packet}\left(P_t\left(s,u\right)\right) = 1 - \prod_{n \in P_T(s,u)} \left(1 - \text{loss-packet}\left(n\right)\right) \tag{10}$$

D. DATA PREFETCHING

Local execution of data scheduling can avoid reading data remotely, shortening the task's waiting time, and improving job execution efficiency. Adjust the scheduling order of data scheduling according to the principle of maximizing locality to optimize the data scheduling queue. However, the dynamic changes in the work load of the working nodes may cause errors between the actual execution schedule and the predicted schedule of some data scheduling. As a result, local data scheduling in the task queue maintained by some nodes is completed earlier or delayed in the later stage of job execution. This causes tasks migrate from nodes with slow progress to nodes with fast progress, then increases the non-localized execution rate of data scheduling.

The purpose of the node task queue monitoring is to track the execution progress of each task queue, and select the source and destination nodes for the migration data scheduling during the data prefetch process. Node selection results are affected by several factors, such as the timing of node selection. In this section we describe the selection method of source node and destination node through formal methods. Assume that the number of working nodes in the IoT cluster is n, and the set of nodes is $Node = \{Node_1, Node_2, \ldots, Node_n\}$.

Definition 4: After the task queue of the working nodes is optimized, the $Node_i$'s optimization task set of the data scheduling queue can be expressed as:

$$QNode_i = \{map_{i1}, map_{i2}, \ldots, map_{is}\} \tag{11}$$

Definition 5: When data migration is implemented, the time for the system to select the source node and the destination node. Assuming that the total number of data scheduling for IoT jobs is m, the node selection time threshold λ can be expressed as:

$$\lambda = 1 - \theta \times \frac{n}{m} \quad 1 \le \theta < \left\lceil \frac{m}{n} \right\rceil \tag{12}$$

Here θ represents the experience value, which can be adjusted according to the type of IoT job. If the average execution time of data scheduling in the job is short, this value can be set to a larger value, and vice versa.

Definition 6: When the system selects the source node and the destination node according to the node selection time threshold, the estimated remaining time of the optimized data scheduling in the task queue.

Let the optimized task queue of the node $Node_i$'s be $QNode_i$. When selecting the source node and the destination node, the total number of data scheduling in $QNode_i$ is m, the progress of the ongoing data scheduling m_k is ρ, and the input data block of task m_k is B_k, the number of remaining unscheduled data scheduling is r, then the remaining completion time $R(QNode_i)$ of the queue $QNode_i$ is:

$$\left.\begin{aligned} R\left(QNode_i\right) &= T \times r + B_k \times (1-\rho) \times \overline{V_i} \\ \overline{V_i} &= \sum_{j=1}^{m-r-1} \frac{B_j}{t_j(m-r-1)} \end{aligned}\right\} \tag{13}$$

where T represents the predicted execution time of data scheduling; $\overline{V_i}$ represents the average execution rate of completed data scheduling in $QNode_i$.

After selecting the source and destination nodes for the task migration, it is needed to determine whether the task is suitable for migration. The completion time of the Map phase of an IoT job depends on the time taken by the node that completed the data scheduling with the most time. That is the optimization time of the task queue that has the longest remaining completion time in the task queue set. If part of the data in the queue migrates to other nodes (destination nodes) can reduce the execution time of the entire job Map phase. In the best case, the task queues on the source and destination nodes have the same remaining completion time after migration. However, in order to ensure the data locality of data scheduling, the nodes must ensure that these tasks meet the data locality before the nodes perform the migration tasks, that is, the input data required by the tasks has been successfully prefetched. Therefore, when performing migrating tasks, it is needed to consider the impact of the time taken by data prefetch on task migration. If the optimization task queue of the destination node is empty within the time of data prefetching, it may cause the destination node to steal the data scheduling of other nodes, which may cause non-local execution of the data scheduling.

Definition 7: The time required for data to travel from the remote node to the target node over the network. This value is an empirical value. Different types of IoT job correspond to different prefetch delays. In addition, the value reflects the relationship between the size of the data block and the network transmission rate. When the data block is large, the transmission takes up more network resources. Under this condition, the time should be set to a larger value. Otherwise, the time should be set to smaller value.

Let the destination node of the migration task be *TNode*, the optimized task queue of this node is *TQueue*; the source node of the migration task is *SNode*, and the optimized task queue of this node is *SQueue*. Let the selection time threshold of node isλ. At timeλ, the data being executed in the *TQueue* queue is *mt* and its execution progress is ρ_t; the data being executed in the *SQueue* queue is m_s, and its execution progress is ρ_s; the data prefetch delay is φ. According to formula (8), the remaining completion times of the optimized queues *SQueue* and *TQueue* can be obtained respectively, which are represented as $R(SQueue)$ and $R(TQueue)$. The migration task migration should also meet the following conditions:

$$\left.\begin{aligned} R(TQueue) &> \phi \\ R(SQueue) - T(SNode) &> \phi \end{aligned}\right\} \tag{14}$$

Here *T(SNode)* represents the predicted execution time of data scheduling on the source node. The first condition guarantees the locality of the data scheduling data of the destination node. The second condition avoids the waste of system resources caused by the source node and the destination node performing the same data scheduling at the same time.

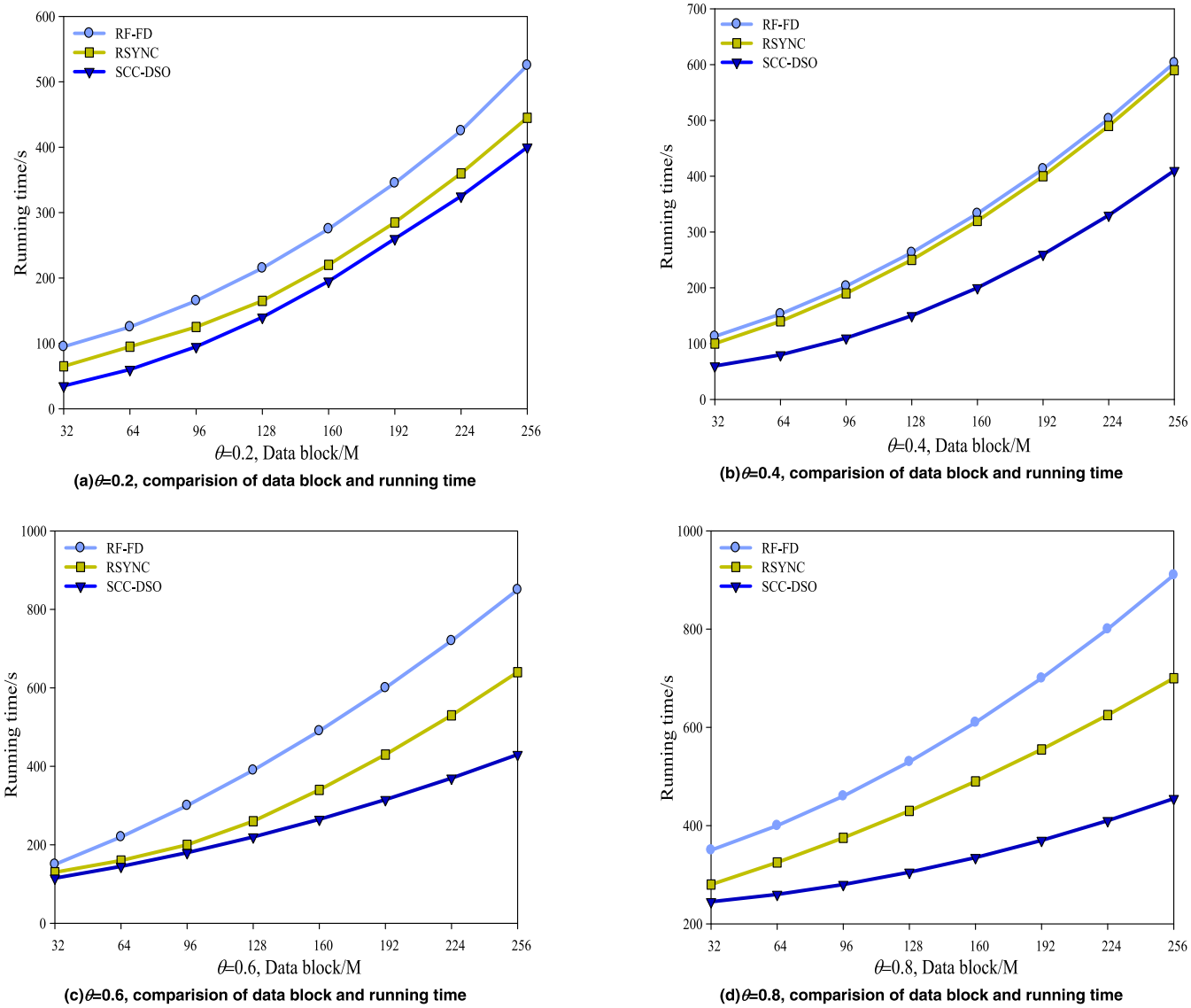
In the perceptual cloud computing system, assume the destination node of the task migration is *TNode*, the current number of network connections is T_t, the migration task is *mt*, and the internal data prefetch delay of the same node is ϕ_1, The data prefetch delay between two nodes located in the same rack is ϕ_2, the data prefetch delay between two nodes located in different racks is ϕ_3, and the set of nodes storing c copies of prefetch data block B is $CNode$=$\{CNode_1, CNode_2, \ldots, CNode_c\}$. Assume that a copy storage node of data block$B$ is $CNode_i$, and the current network connection number of this node is T_i. $CNode_i$ and *TNode* are located in different racks respectively. The prefetch load factor at node $CNode_i$can be expressed as:

$$\begin{aligned} &PLFactor\left(TNode, CNode_i\right) \\ &= \sqrt{(\phi_i - \phi_t)^2 + (T_i - T_t)^2} \end{aligned} \tag{15}$$

The Source Worker Node Choosing (SWNC) algorithm uses the prefetch load factor to select the source data node:

TABLE 1. The pseudocode of SCC-DSO algorithm.

```
Input: IoT load input file F and its size FSsize; the size of each data block
of the file system,represented as b; data scheduling prediction model
mode_l; the number of working nodes in IoT cluster, represented as n
Output: Prefetch data source node DSNode
1  Node = {Node_1, Node_2, ... , Node_n};
2  BlockAllocat = new HashMap{node.ID, list<block.ID}};
3  dulBlockAllocat = new HashMap{node.ID, list<block.ID}};
4  [BlockAllocat, dulBlockAllocat] = runHABA(F, Fsize, mode_l, n);
5  InitMapQueue ← initMap(Node);
6  MapPreassignQueue ← runMPA(InitMap.Queue, n, Block.Allocat);
7  [SNode, TNode, OpQueue] = runSTC(MapPreassign.Queue,Block
                                    Allocat, n);
8  needMigration = decideMigration(OpQueue);
9  for (i=0;i<=n;i++)
9    if (needMigration) then
10   isContain = containBlock(OpQueue, SNode, TNode);
11     if (! Is Contain) then
12       DSNode = SNodes[i]
13       preFetch(DSNode,TNode);
14       upDate(MapPreassignQueue);
15       go to line 7
18     end if
19   end if
20 end for
```



(a) θ=0.2, comparision of data block and running time

(b) θ=0.4, comparision of data block and running time

(c) θ=0.6, comparision of data block and running time

(d) θ=0.8, comparision of data block and running time

FIGURE 4. Comparison of execution time of different jobs under single copy conditions.

first, query the node location and network load information of the copy of the migration task input data block; then, calculate the prefetch load factor of each replica node of the prefetched data block according to formula (15); finally, select the appropriate prefetch source data node according to the prefetch load factor of each replica node.

E. SCC-DSO ALGORITHM DESCRIPTION

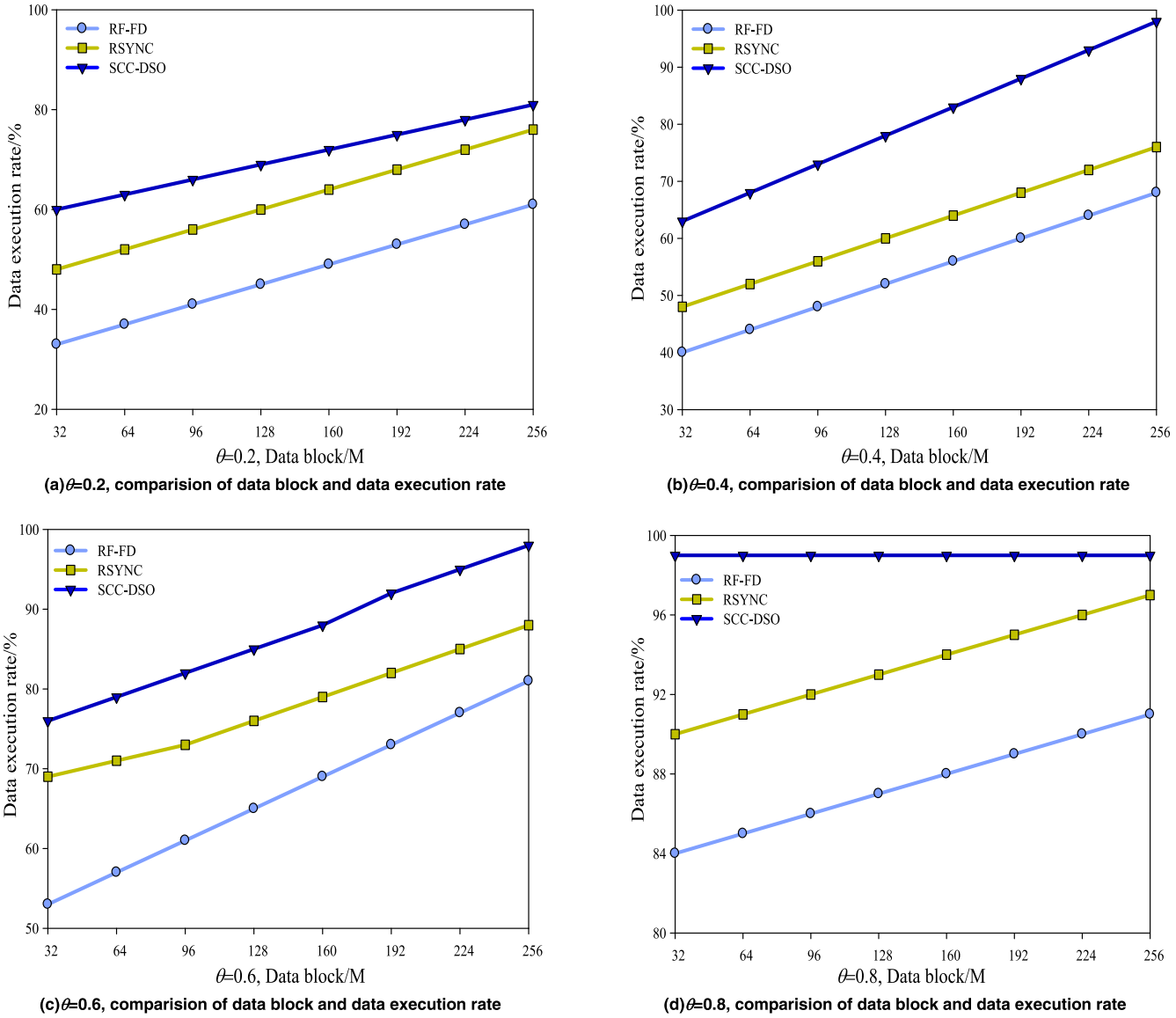
The implementation process based on the data localization optimization method is given below:

Step 1: Extract performance characteristics from the completed data scheduling log information of each node, construct a sample data set of the performance prediction model proposed in section II, and train the prediction model to obtain the values of various parameters in the model.

Step 2: Based on the processing capacity and performance prediction model of the working nodes in the IoT cluster, according to the principle of maximizing the local execution rate of data scheduling, the first copy placement strategy of the data block is solved by formula (5). Then, the remaining copies are stored according to the heterogeneous-aware data placement algorithm.

Step 3: According to the metadata information of the data block stored on each working node and the data block allocation result generated by the SCC-DSO algorithm, the SCC-DSO algorithm is executed to adjust the position of each data scheduling in the task queue of the node so that the assignment of tasks of each working node does not affect each other and improves the local execution rate of data scheduling.

Step 4: Monitor the task queue of the node, track the execution progress of the task queue of each node, and select the source node and destination node for the migration data scheduling during the data prefetch process. Then, based on the execution status of the task queue in the source node and the destination node, it is determined whether to perform



(a) θ=0.2, comparision of data block and data execution rate

(b) θ=0.4, comparision of data block and data execution rate

(c) θ=0.6, comparision of data block and data execution rate

(d) θ=0.8, comparision of data block and data execution rate

FIGURE 5. Scheduled execution rate of different job data under single copy conditions.

a task migration operation. If the judgment result is true, performes tep 5, otherwise finish the optimization process.

Step 5: Determine whether the migrated data schedule stores the input data of the data schedule at the destination node. If it contains, the source data node selection operation is not required, and go to step 4; otherwise, execute the SCC-DSO algorithm to select a suitable prefetch sourc data nodes.

Step 6: Prefetch the input data blocks of the migration task to the destination node through the network.

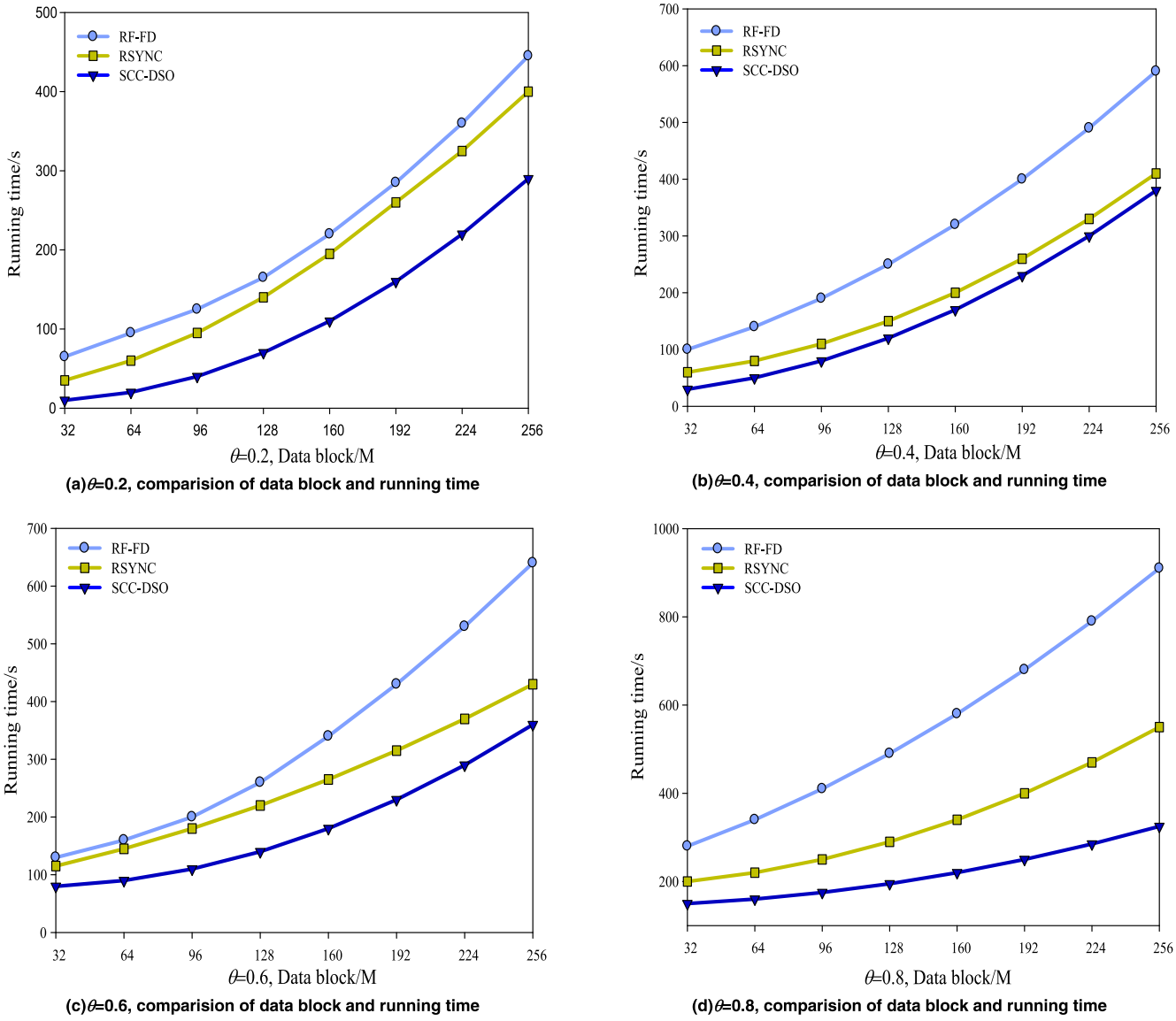
Step 7: Establish a mapping relationship between the migration task and the pre-fetched data block, and add the task to the data scheduling queue of the destination node.Then, go to step 4.

IV. EXPERIMENTAL RESULTS AND ANALYSIS

In this paper, multiple experiments are designed to verify the effectiveness and stability of the SCC-DSO algorithm. The copy degree of HDFS file system is an important factor affecting the performance of network operations. In addition, the network load in the cluster nodes will also affect the prefetch of data. For this reason, the paper first tests the method proposed in the paper compared with RF-FD [26] and RSYNC algorithm [27]. The results are compared and analyzed at single-copy scenarios with different data block sizes and network loads. Since RSYNC can only be used for single copy scenarios, the performance of the RF-FD method is tested for multi-copy scenarios. The results are compared and analyzed with different data block sizes and network loads, too.

A. PERFORMANCE OF SCC-DSO ALGORITHM UNDER SINGLE COPY CONDITION

This subsection tests the performance of the SCC-DSO method in a cheap and heavy network overhead environment



(a) θ=0.2, comparision of data block and running time

(b) θ=0.4, comparision of data block and running time

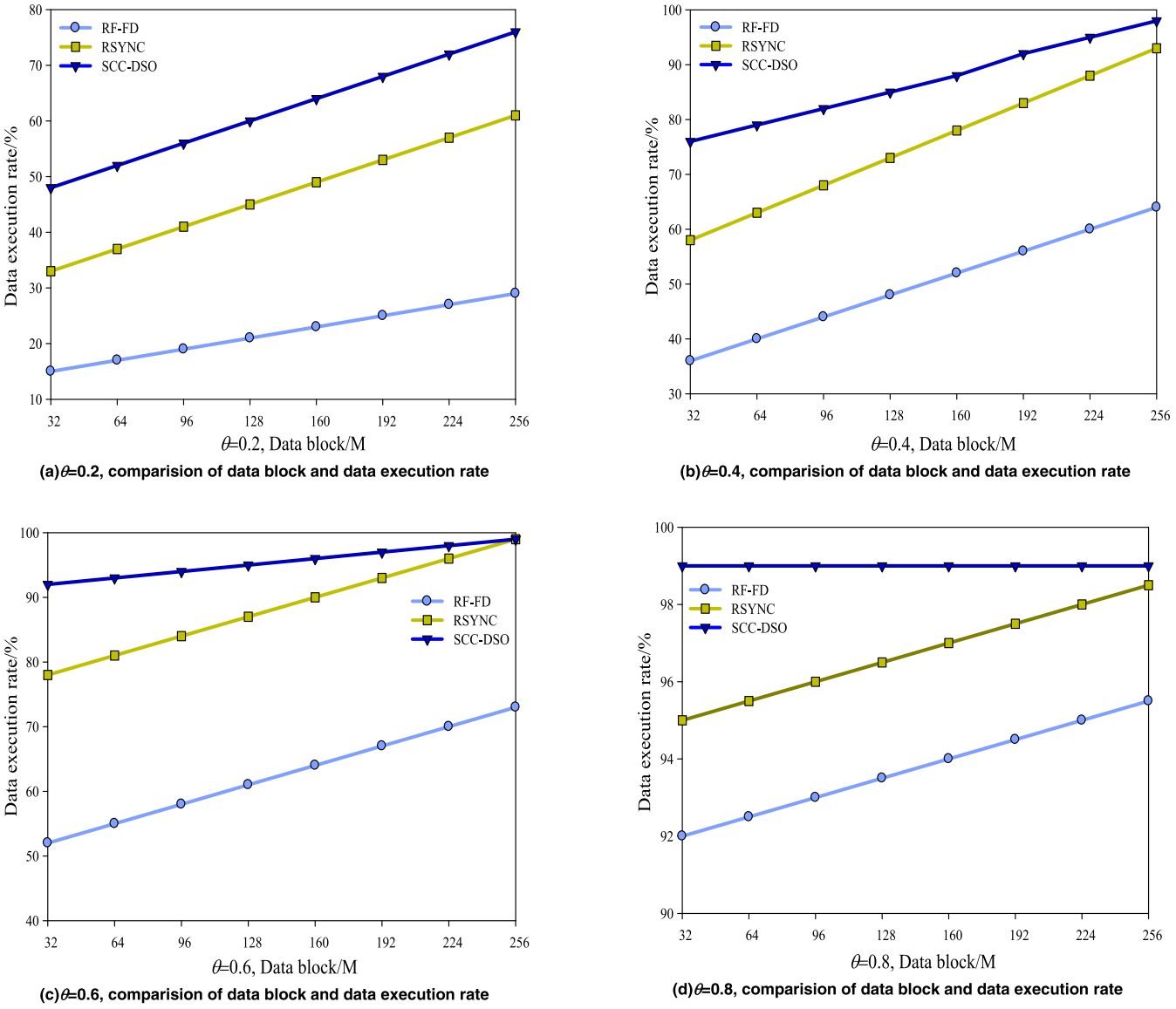
(c) θ=0.6, comparision of data block and running time

(d) θ=0.8, comparision of data block and running time

FIGURE 6. Comparision of execution time of jobs under multiple copies.

when the copy degree of HDFS file system is 1. Since the IoT cluster system is interconnected through the Gigabit network, the default network environment is regarded as a cheap network load environment. A cheap network load means that the nodes have a wider network bandwidth. Each case is tested for 50 times. The experimental results are average values which are calculated for 50 times.

Figure 4 shows the execution time of each test case in the network operation phase under low network conditions. It can be seen from Figure 4 that the SCC-DSO has better performance than the other two optimization methods. In the best case, the performance of the SCC-DSO method obtain average improvement of 13% compared with the RF-FD method and improvement of 7% compared with the RSYNC method. The localized execution rate of data scheduling during the execution of each test case is shown in Figure 5. It can be seen from Figure 5 that the SCC-DSO method has a higher localized execution rate than the other two methods. When the data block size is 64M, the localized execution rate of the SCC-DSO method can be as high as 91%, while the localized execution rates of the RF-FD and RSYNC methods are only 62.18% and 85%. The performance of SCC-DSO and the localization execution rate of data scheduling have been greatly improved compared to the RF-FD method. This is because the data blocks in the RF-FD method are randomly placed on the working node according to the utilization rate of the node's hard disk. The strategy does not take into account the differences in computing capabilities of the working nodes in the heterogeneous IoT, causing nodes with strong computing capabilities to complete local tasks first, and then "stealing" local data scheduling of working nodes with weaker computing capabilities, resulting in large amounts of data scheduling Non-local execution.



(a) θ=0.2, comparision of data block and data execution rate

(b) θ=0.4, comparision of data block and data execution rate

(c) θ=0.6, comparision of data block and data execution rate

(d) θ=0.8, comparision of data block and data execution rate

FIGURE 7. Scheduling execution rates of different job data under multiple copies.

The non-local execution of data scheduling results in cross-node transmission of data scheduling, thereby increasing the time of the job during the network operation phase. The performance of the SCC-DSO method and the localization execution rate of data scheduling of the SCC-DSO method have been improved to a certain extent compared with RSYNC method. This is because that the SCC-DSO method selects the input data placement node based on the ratio of the computing power of each working node and store appropriate number of data blocks in equal proportion. The nodes with strong computing power store a larger number of data to schedule input data blocks, reducing the number of data block migrations. However, the task execution time and the input data size are not a simple linear relationship. It also related to the load characteristics and the performance characteristics of the working nodes. Therefore, the RSYNC method still causes some working nodes to perform non-local data scheduling. The SCC-DSO method proposed in this paper uses the performance prediction model to evaluate the processing capacity of the working node, which is more accurate than RSYNC's prediction. In addition, the SCC-DSO method can dynamically monitor the execution progress of the local data scheduling list of each working node, and pre-fetch input data to the node where the non-local data scheduling is located. Thus, the SCC-DSO method can reduce the waiting overhead of non-local data scheduling for the transmission of input data across the node network.

B. PERFORMANCE OF SCC-DSO ALGORITHM UNDER MULTIPLE COPIES CONDITION

Figures 6 and 7 show the job execution time and localized execution rate with adopting various optimization methods

in a narrow network bandwidth environment. The simulation results show that the SCC-DSO algorithm proposed in this paper is superior in job execution time and localized execution rate of data scheduling. Under average conditions, compared with the RF-FD method, the SCC-DSO method improves the performance of the job during the execution phase by 19.8%. Compared with the RSYNC method, the performance of the execution phase of the job is improved by 7.6% on average. It can be seen from Figure 6, with the low network load, the data execution time of the three methods has been extended to a certain extent, and the localized execution rate of data scheduling has been improved to a certain extent. The main reason is that in a low network bandwidth environment, the network becomes a bottleneck of system performance. Due to non-local data scheduling, the time overhead of waiting for input data on nodes with powerful computing power increases. So its execution efficiency is reduced. In this case, for the RF-FD method, a working node with weak computing power needs to perform more local data scheduling, which increases the execution time of the job. As shown in Figure 7, for the SCC-DSO and RSYNC methods, the input data for data scheduling is stored based on the node's computing capacity. Therefore, compared to the wide network bandwidth situation, the narrow network bandwidth has weaker effect on the local network execution rate for its data scheduling. However, the performance of the SCC-DSO method is still better than RSYNC. The main reason is that the data prefetch mechanism is used in the SCC-DSO method, which reduces the waiting time of the working node for non-local data scheduling input data network transmission.

V. CONCLUSION

This paper focuses on the performance degradation of jobs in the IoT cluster during the execution stage. A perceptual cloud based algorithm for data scheduling optimization is proposed. This method places different numbers of data blocks at working nodes according to their processing capabilities. The dynamic migration of data scheduling is introduced to improve localized execution rate. Compared with other algorithms, the SCC-DSO algorithm can sense the heterogeneous processing capabilities of working nodes and allocate different sizes of data to working nodes according to their processing capabilities by means of performance prediction methods. The reliability of data stored by working nodes is also taken into account. By obtaining the initial data scheduling list information of each working node, a data scheduling queue optimization algorithm based on data block distribution is proposed. A data scheduling preallocation optimization queue is constructed to ensure that different working nodes perform local data scheduling in parallel without mutual interference. By monitoring the runtime information of the task queues of each working node, a data prefetch algorithm based on the minimum completion time of the task queue is proposed. The parallelism of computing and network transmission operations is used to reduce the waiting time of working nodes for non-local data scheduling input data. Experimental results show that the algorithm in this paper effectively reduces the execution time of non-local data scheduling. Compared with the RF-FD and RSYNC methods, the localized execution rate and job performance of the SCC-DSO algorithm have been significantly improved.

REFERENCES

[1] Z. Sun, G. Zhao, M. Li, and Z. Lv, "Job performance optimization method based on data balance in the wireless sensor networks," *Int. J. Online Eng.*, vol. 13, no. 12, pp. 4–17, Dec. 2017.

[2] T. Wang, H. Luo, X. Zheng, and M. Xie, "Crowdsourcing mechanism for trust evaluation in CPCS based on intelligent mobile edge computing," *ACM Trans. Intell. Syst. Technol.*, vol. 10, no. 6, pp. 1–19, Oct. 2019.

[3] Z. Sun, Y. Zhang, Y. Nie, W. Wei, J. Lloret, and H. Song, "CASMOC: A novel complex alliance strategy with multi-objective optimization of coverage in wireless sensor networks," *Wireless Netw.*, vol. 23, no. 4, pp. 1201–1222, 2017.

[4] J. Zhu, Y. Song, D. Jiang, and H. Song, "A new deep-Q-learning-based transmission scheduling mechanism for the cognitive Internet of Things," *IEEE Internet Things J.*, vol. 5, no. 4, pp. 2375–2385, Aug. 2018.

[5] T. Wang, Y. Mei, W. Jia, X. Zheng, G. Wang, and M. Xie, "Edge-based differential privacy computing for sensor-cloud systems," *J. Parallel Distrib. Comput.*, vol. 136, pp. 75–85, Feb. 2020.

[6] J. Lu, Y. Xin, Z. Zhang, X. Liu, and K. Li, "Game-theoretic design of optimal two-sided rating protocols for service exchange dilemma in crowdsourcing," *IEEE Trans. Inf. Forensics Secur.*, vol. 13, no. 11, pp. 2801–2815, Nov. 2018.

[7] H. Song and M. Brandt-Pearce, "A 2-D discrete-time model of physical impairments in wavelength-division multiplexing systems," *J. Lightw. Technol.*, vol. 30, no. 5, pp. 713–726, Mar. 1, 2012.

[8] W. Tian, K. Haoxiong, Z. Xi, W. Kun, S. A. Kumar, and L. Anfeng, "Big data cleaning based on mobile edge computing in industrial sensor-cloud," *IEEE Trans. Ind. Informat.*, vol. 16, no. 2, pp. 1321–1329, Feb. 2020.

[9] W. Tian, L. Hao, W. Jia, A. Liu, and M. Xie, "MTES: An intelligent trust evaluation scheme in sensor-cloud enabled industrial Internet of Things," *IEEE Trans. Ind. Informat.*, vol. 16, no. 3, pp. 2054–2062, Mar. 2020.

[10] S. Zeyu, W. Weiguo, W. Huanzhao, C. Heng, and X. Xiaofei, "A novel coverage algorithm based on event-probability-driven mechanism in wireless sensor network," *EURASIP J. Wireless Commun. Netw.*, vol. 2014, pp. 1–17, Dec. 2014.

[11] T. Liu, B. Wu, H. Wu, and J. Peng, "Low-cost collaborative mobile charging for large-scale wireless sensor networks," *IEEE Trans. Mobile Comput.*, vol. 16, no. 8, pp. 2213–2227, Aug. 2017.

[12] X. Liu, "Node deployment based on extra path creation for wireless sensor networks on mountain roads," *IEEE Commun. Lett.*, vol. 21, no. 11, pp. 2376–2379, Nov. 2017.

[13] T. Wang, D. Zhao, S. Cai, W. Jia, and A. Liu, "Bidirectional prediction based underwater data collection protocol for end-edge-cloud orchestrated system," *IEEE Trans. Ind. Informat.*, to be published, doi: 10.1109/TII.2019.2940745.

[14] X. Liu, "A novel transmission range adjustment strategy for energy hole avoiding in wireless sensor networks," *J. Netw. Comput. Appl.*, vol. 67, pp. 43–52, May 2016.

[15] Z. Sun, L. Wei, C. Xu, T. Wang, Y. Nie, X. Xing, and J. Lu, "An energy-efficient cross-layer-sensing clustering method based on intelligent fog computing in WSNs," *IEEE Access*, vol. 7, pp. 144165–144177, 2019.

[16] Y. Nie, H. Wang, Y. Qin, and Z. Sun, "Distributed and morphological operation-based data collection algorithm," *Int. J. Distrib. Sensor Netw.*, vol. 13, no. 7, pp. 1–16, Jul. 2017.

[17] Z. Sun, G. Zhao, and X. Pan, "PM-LPDR: A prediction model for lost packets based on data reconstruction on lossy links in sensor networks," *Int. J. Comput. Sci. Eng.*, vol. 19, no. 2, pp. 177–188, 2019.

[18] S. Bitam, S. Zeadally, and A. Mellouk, "Fog computing job scheduling optimization based on bees swarm," *Enterprise Inf. Syst.*, vol. 12, no. 4, pp. 373–397, Apr. 2017.

[19] X. Liu, "Routing protocols based on ant colony optimization in wireless sensor networks: A survey," *IEEE Access*, vol. 5, pp. 26303–26317, 2017.

[20] M. Mededjel, G. Belalem, and A. Neki, "Towards a traceability system based on cloud and fog computing," *Multiagent Grid Syst.*, vol. 13, no. 1, pp. 47–68, Apr. 2017.

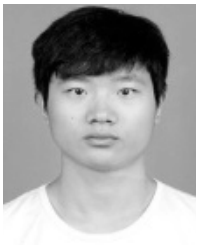
[21] Z. Sun, X. Xing, B. Song, Y. Nie, and H. Shao, "Mobile intelligent computing in Internet of Things: An optimized data gathering method based on compressive sensing," *IEEE Access*, vol. 7, pp. 66110–66122, 2019.
[22] X. Liu and P. Zhang, "Data drainage: A novel load balancing strategy for wireless sensor networks," *IEEE Commun. Lett.*, vol. 22, no. 1, pp. 125–128, Jan. 2018.
[23] Z. Sun, L. Li, X. Xing, Z. Lv, and N. N. Xiong, "A novel nodes deployment assignment scheme with data association attributed in wireless sensor networks," *J. Internet Technol.*, vol. 20, no. 2, pp. 509–520, 2019.
[24] T. Wang, P. Wang, S. Cai, Y. Ma, A. Liu, and M. Xie, "A unified trustworthy environment establishment based on edge computing in industrial IoT," *IEEE Trans Ind. Informat.*, to be published, doi: 10.1109/TII.2019.2955152.
[25] Q. Li, W. Wu, Z. Sun, L. Wang, J. Huang, and X. Zhou, "A novel hierarchal scheduling strategy for rendering system," in *Proc. Int. Conf. Identificat., Inf., Knowl. Internet Things (IIKI)*, Beijing, China, Oct. 2015, pp. 206–209.
[26] T. Wang, J. Zhou, A. Liu, M. Z. A. Bhuiyan, G. Wang, and W. Jia, "Fog-based computing and storage offloading for data synchronization in IoT," *IEEE Internet Things J.*, vol. 6, no. 3, pp. 4272–4282, Jun. 2019.
[27] J. Tang, A. Liu, J. Zhang, N. Xiong, Z. Zeng, and T. Wang, "A trust-based secure routing scheme using the traceback approach for energy-harvesting wireless sensor networks," *Sensors*, vol. 18, no. 3, p. 751, Mar. 2018.
[28] T. Wang, L. Qiu, A. K. Sangaiah, G. Xu, and A. Liu, "Energy-efficient and trustworthy data collection protocol based on mobile fog computing in Internet of Things," *IEEE Trans Ind. Informat.*, vol. 16, no. 5, pp. 3531–3539, May 2020, doi: 10.1109/TII.2019.2920277.
[29] Z. Sun, X. Xing, B. Yan, and Z. Lv, "CMTN-SP: A novel coverage-control algorithm for moving-target nodes based on sensing probability model in sensor networks," *Sensors*, vol. 19, no. 2, p. 257, Jan. 2019.
[30] H. Song and M. Brandt-Pearce, "Range of influence and impact of physical impairments in long-haul DWDM systems," *J. Lightw. Technol.*, vol. 31, no. 6, pp. 846–854, Mar. 15, 2013.
[31] Z. Sun, G. Zhao, and X. Xing, "ENCP: A new energy-efficient nonlinear coverage control protocol in mobile sensor networks," *EURASIP J. Wireless Commun. Netw.*, vol. 2018, no. 1, pp. 1–15, Jan. 2018.
[32] X. Liu, "Survivability-aware connectivity restoration for partitioned wireless sensor networks," *IEEE Commun. Lett.*, vol. 21, no. 11, pp. 2444–2447, Nov. 2017.
[33] Z. Sun, X. Xing, T. Wang, Z. Lv, and B. Yan, "An optimized clustering communication protocol based on intelligent computing in information-centric Internet of Things," *IEEE Access*, vol. 7, pp. 28238–28249, 2019.
[34] H. Tao, M. Z. A. Bhuiyan, M. A. Rahman, T. Wang, J. Wu, S. Q. Salih, Y. Li, and T. Hayajneh, "TrustData: Trustworthy and secured data collection for event detection in industrial cyber-physical system," *IEEE Trans Ind. Informat.*, vol. 16, no. 5, pp. 3311–3321, May 2020, doi: 10.1109/TII.2019.2950192.
[35] Z. Sun, Y. Zhang, X. Xing, H. Song, H. Wang, and Y. Cao, "EBKCCA: A novel energy balanced k-coverage control algorithm based on probability model in wireless sensor networks," *KSII Trans. Internet Inf. Syst.*, vol. 10, no. 8, pp. 3621–3640, 2016, doi: 10.3837/tiis.2016.08.011.
[36] Y. Wu, H. Huang, N. Wu, Y. Wang, M. Z. Alam Bhuiyan, and T. Wang, "An incentive-based protection and recovery strategy for secure big data in social networks," *Inf. Sci.*, vol. 508, pp. 79–91, Jan. 2020.
[37] Z. Sun, Z. Lv, Y. Hou, C. Xu, and B. Yan, "MR-DFM: A multi-path routing algorithm based on data fusion mechanism in sensor networks," *Comput. Sci. Inf. Syst.*, vol. 16, no. 3, pp. 867–890, 2019.
[38] T. Wang, L. Qiu, A. K. Sangaiah, A. Liu, M. Z. A. Bhuiyan, and Y. Ma, "Edge computing based trustworthy data collection model in the Internet of Things," *IEEE Internet Things J.*, to be published, doi: 10.1109/JIOT.2020.2966870.
[39] Sun, Li, Wei, Li, Min, and Zhao, "Intelligent sensor-cloud in fog computer: A novel hierarchical data job scheduling strategy," *Sensors*, vol. 19, no. 23, p. 5083, Nov. 2019.
[40] H. Teng, Y. Liu, A. Liu, N. N. Xiong, Z. Cai, T. Wang, and X. Liu, "A novel code data dissemination scheme for Internet of Things through mobile vehicle of smart cities," *Future Gener. Comput. Syst.*, vol. 94, pp. 351–367, May 2019.
[41] L. Yunchang, L. Chunlin, L. Youlong, S. Yanling, and Z. Jing, "Scheduling multimedia services in cloud computing envieronment," *Enterprise Inf. Syst.*, vol. 12, no. 2, pp. 218–235, 2018.
[42] J. Yue, M. Xiao, and Z. Pang, "Distributed fog computing based on batched sparse codes for industrial control," *IEEE Trans Ind. Informat.*, vol. 14, no. 10, pp. 4683–4691, Oct. 2018.
[43] Y. Zhang, "Classified scheduling algorithm of big data under cloud computing," *Int. J. Comput. Appl.*, vol. 41, no. 4, pp. 262–267, 2019.
[44] W. Xiuran and W. Feng, "Massive data balance scheduling in cloud computing environment," *Int. J. Mechatronics Appl. Mech.*, vol. 2019, no. 5, pp. 100–105, 2019.
[45] Y. Zhang, W. Huang, T. Zhang, and T. Zhang, "A novel topology optimization theory and parallel data analysis model based resource scheduling algorithm for cloud computing," *Recent Adv. Elect. Electron. Eng.*, vol. 11, no. 4, pp. 449–456, Nov. 2018.
[46] R. Xie and X. Jia, "Data transfer scheduling for maximizing throughput of big-data computing in cloud systems," *IEEE Trans. Cloud Comput.*, vol. 6, no. 1, pp. 87–98, Jan. 2018.
[47] G. Yang, T. Liang, X. He, and N. Xiong, "Global and local reliability-based routing protocol for wireless sensor networks," *IEEE Internet Things J.*, vol. 6, no. 2, pp. 3620–3632, Apr. 2019.
[48] L. A. Tawalbeh, W. Bakheder, and H. Song, "A mobile cloud computing model using the cloudlet scheme for big data applications," in *Proc. IEEE 1st Int. Conf. Connected Health, Appl., Syst. Eng. Technol. (CHASE)*, Wsahington, DC, USA, Jun. 2016, pp. 73–77.
[49] W. Tian, B. M. Z. Alam, W. Guojun, Q. Lianyong, W. Jie, and H. Thaier, "Preserving balance between privacy and data integrity in edge-assisted Internet of Things," *IEEE Internet Things J.*, to be published, doi: 10.1109/JIOT.2019.2951687.
[50] T. Qiu, K. Zheng, H. Song, M. Han, and B. Kantarci, "A local-optimization emergency scheduling scheme with self-recovery for a smart grid," *IEEE Trans Ind. Informat.*, vol. 13, no. 6, pp. 3195–3205, Dec. 2017.
[51] W. Li, H. Song, and F. Zeng, "Policy-based secure and trustworthy sensing for Internet of Things in smart cities," *IEEE Internet Things J.*, vol. 5, no. 2, pp. 716–723, Apr. 2018.
[52] Z. Sun, R. Tao, L. Li, and X. Xing, "A new energy-efficient multi-target coverage control protocol using event-driven-mechanism in wireless sensor networks," *Int. J. Online Eng. (iJOE)*, vol. 13, no. 2, pp. 53–67, Feb. 2017.

ZEYU SUN received the B.S. degree in computer science and technology from the Henan University of Science and Technology, in 2003, the M.Sc. degree from Lanzhou University, in 2010, and the Ph.D. degree from Xi'an Jiaotong University, in 2017. He is currently an Assistant Professor with the School of Computer and Information Engineering, Luoyang Institute of Science and Technology, Luoyang, Henan, China. He is also the External Master Tutor with the School of Information Engineering, Henan Institute of Science and Technology, Xinxiang, Henan. His research interests include wireless sensor networks, mobile computing, and the Internet of Things. He is a member of the China Computer Federation.

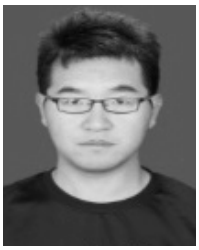
ZHIGUO LV received the B.S. degree in applied electronic technology from Henan Normal University, in 2000, the M.S. degree in communication and information system from the Guilin University of Electronic Technology, in 2008, and the Ph.D. degree from Xidian University, in 2019. He is currently a Lecturer with the School of Computer and Information Engineering, Luoyang Institute of Science and Technology. His research interests include compressive sensing, wireless sensor networks, and MIMO systems.

HUIHUI WANG (Senior Member, IEEE) received the Ph.D. degree in electrical engineering from The University of Virginia, Charlottesville, VA, USA, in August 2013. In 2011, she was an Engineering Intern with Qualcomm Inc. In August 2013, she joined the Department of Engineering, Jacksonville University, Jacksonville, FL, USA, where she is currently an Associate Professor and the Founding Chair of the Department of Engineering. She is the author of more than 50 articles and holds one U.S. patent. Her research interests include cyber-physical systems, the Internet of Things, healthcare and medical engineering based on smart materials, robotics, haptics based on smart materials/structures, ionic polymer metallic composites, and micro-electromechanical systems.

ZHIXIAN LI received the M.Sc. degree from the Department of Information and Control Engineering, Xi'an University of Architecture and Technology, in 2005. He is currently an Associate Professor with the School of Computer and Information Engineering, Luoyang Institute of Science and Technology, Luoyang, Henan, China. His research interests include wireless sensor networks, virtual reality, and the Internet of Things.



FUQIAN JIA received the B.S. degree in Internet of Things project from the Henan Institute of Science and Technology, in 2018, where he is currently pursuing the M.Sc. degree. His research interests include wireless sensor networks, artificial intelligence, and intelligent control.



CHUNXIAO LAI received the B.S. degree in computer science and technology from Anyang Normal University, in 2013. He is currently pursuing the M.Sc. degree with the Henan Institute of Science and Technology. His research interests include wireless sensor networks and information technology.