EAI Endorsed Transactions
on Scalable Information Systems
Research Article EAI.EU

Research on the Deployment Strategy of Big Data Visualization Platform by the Internet of Things Technology

Guangtao Zhang*

Information Engineering College, Yangzhou Polytechnic College, Yangzhou 225000, Jiangsu, China.

Abstract

In order to improve the performance of the big data visualization platform and improve the task scheduling capability of the platform, a big data visualization platform is constructed based on Field Programmable Gate Array (FPGA) chip application power equipment. This study proposes to combine genetic algorithm and ant colony scheduling (ACOS) algorithm to design a big data visualization platform deployment strategy based on improved ACOS algorithm. Firstly, big data technology is analyzed. Then, the basic theory of ant colony algorithm is studied. According to the basic theory of ACOS and genetic algorithm, an improved ACOS algorithm model is constructed. The improved ACOS algorithm scheduler is compared with the other three schedulers. Under the same environment, the completion time of scheduling the same job and different task amounts is analyzed. The Central Processing Unit (CPU) utilization is analyzed when different schedulers completely different workloads. The results show that the constructed big data visualization platform based on the improved ACOS algorithm model has higher task scheduling efficiency than other schedulers, and can greatly shorten the data processing time. The experimental results show that under the homogeneous cluster, the completion time of the improved ACOS algorithm generally lags the capacity scheduler and the fair scheduler. Under the heterogeneous cluster, the improved ACOS algorithm scheduler can reasonably allocate tasks to nodes with different performances, reducing the task completion time. When the number of completed tasks increases from 50 to 200, the time increases by 45s, and the completion time is smaller than other schedulers. The CPU utilization of different task volumes is the highest, and the utilization rate increases from 81% to 95%. The improved ACOS algorithm scheduler has the shortest data processing time and the highest efficiency. This work provides a certain reference value for optimizing the deployment strategy of the big data visualization platform and improving the performance of the platform.

Keywords: CPU, Field Programmable Gate Array, genetic algorithm, IOT, ant colony scheduling, big data.

Received on 16 Ferbuary 2023, accepted on 10 April 2023, published on 05 May 2023

doi: 10.4108/eetsis.v10i3.3051

*Corresponding author. Email: guangtaozhang204016@gmail.com

1. Introduction

In recent years, the Internet of Things (IoT), as a new type of information technology application means, can manage the use of objects through various sensing devices, identification, and information technology. Therefore, it has been widely used in many fields [1]. In the process of building a big data visualization platform, the application of IoT technology can realize the connection between data and the Internet according to the agreed protocol and can exchange and communicate information at any time to realize intelligent identification, positioning, tracking, and monitoring of data. This also provides certain technical support for improving the level of big data management [2].

The purpose of data visualization is to allow users to receive information clearly and effectively. Compared with complex and boring text numbers, humans better perceive size, color, and shape. At present, there are three commonly

used implementation methods for data visualization. The first is a chart function module provided by a tabular data visualization manufacturer. It forms a report on the data by combining it with the database Structured Query Language (SQL) in an Excel-like manner. A second implementation method is a visualization of the software based on graphics, which can display data in the form of graphics or charts through user-defined designs. The third implementation is web front-end visualization [3].

Davis et al. (2020) proposed a method for building a logistics big data visualization platform based on blockchain technology [4]. Cocoros et al. (2021) designed and implemented a web-side data visualization platform based on the Browser/Server (B/S) architecture mode for the problem of urban geological subsidence in geographic information [5]. Ming et al. (2018) designed a set of easy-to-integrate data visualization platform frameworks based on the mainstream Java EE platform in the industry. They implemented the main functional modules of the platform [6]. Chen et al. (2022) combined the actual drive test data of the Long-Term Evolution (LTE) network with the big data analysis method to study the LTE network overlapping coverage and weak coverage optimization methods based on big data [7]. Harb et al. (2020) designed a big data news visualization platform based on Hadoop high-availability cluster, built a Hadoop high-availability cluster, and deployed related services [8]. At present, although there are many kinds of big data platforms, there is almost no research on big data visualization platforms that collect, process, and store data in Field Programmable Gate Array (FPGA) chips. Atat et al. (2018) introduced the Cyber-Physical Systems (CPS) taxonomy by providing a broad overview of data collection, storage, access, processing, and analysis. Compared with other surveys, this is the first panoramic survey of CP's big data. The goal is to provide a panoramic summary of different Cyber-Physical (CP) aspects [9]. FPGA chips are not simply limited to research and design chips, and products in many fields can be optimized with the help of specific chip models [10].

Therefore, it is of great significance to build a dedicated big data visualization platform and need to improve the task scheduling capability of the platform. In order to design a big data collection and analysis platform based on FPGA chip application circuit equipment. A large data visualisation platform based on Field Programmable Gate Array (FPGA) chip application power equipment is created in order to enhance the performance of the platform and its job scheduling capabilities. In order to build a large data visualisation platform deployment strategy based on an enhanced ACOS algorithm, this research suggests combining genetic algorithm with ant colony scheduling algorithm.

Contribution of the work

- Proposes an ant colony scheduling (ACOS) algorithm-based deployment method for a huge data visualisation platform. The performance of the big data visualization platform based on the improved ACOS algorithm model constructed by this study is tested and compared.
- The novel part of the algorithm is the enhancement of the conventional ACOS algorithm and the incorporation of the genetic algorithm, which compensates for the early pheromone deficiency in the ant colony algorithm and enhances the performance of the whole algorithm.
- The purpose of this research is to provide some references for large data visualisation platform deployment strategies, scheduling optimization for computer tasks, and performance enhancement.
- The results demonstrate that the developed big data visualisation platform, based on the enhanced ACOS algorithm model, has greater job scheduling efficiency than existing schedulers, and can significantly reduce the data processing time.

The paper structure follows: Section 2 defines the existing works related to the proposed method. Section 3 shows the proposed method in detail explanation. Section 4 shows the results of the proposed method. Finally, the conclusion is defined in Section 5.

2. Related Works

Huge numbers of wireless sensor networks have emerged to monitor a wide variety of infrastructure in fields as diverse as healthcare, energy, transportation, smart cities, building automation, agriculture, and industry, all of which contribute to the continuous streams of data being produced by the rapidly expanding Internet of Things (IoT). Within IoT processes, Big Data technologies play a crucial role as visual analytics tools, delivering useful knowledge in real-time to support crucial decision making. This study offers a thorough analysis of the current state of visualization approaches, software, and methodologies for the Internet of Things. By analyzing the visual analytics pipeline, Protopsaltis (2020) places data visualization within the visual analytics process. The Authors examine the several kinds of charts that can be used for data visualization and provide guidelines for when to use each one, depending on the specifics of the scenario at hand [11]. Some of the most promising visualization technologies are explored in greater depth. Since Big Data methods aren't uniform across IoT applications, they're looking into specific domains' unique visualization challenges. There is also a discussion of visualization techniques for spotting anomalies. Finally, they discuss the most pressing issues associated with IoT visualizations.

The advent of the IoT has ushered in a new era, one that sees people abandoning their more anachronistic practices in favour of a more technologically advanced way of life. Internet of Things-enabled changes include "smart" versions of cities, residences, transportation systems, utilities, and entire businesses. Extensive studies and analyses have been conducted to improve technology using IoT. However, many problems and obstacles remain that must be fixed before IoT can reach its full potential. IoT's many uses, challenges, enabling technologies, social and environmental implications, etc., all need to be taken into account while tackling these problems. This review article by Kumar S. et

al., (2019) aims to do just that, providing a thorough debate from both a technological and sociological point of view. Some of the difficulties and core issues surrounding the Internet of Things are explored in this article, along with its architecture and key application areas [12]. The article also highlights the existing literature and shows how they have contributed to various parts of the Internet of Things. In addition, we have talked about how large data and its analysis are crucial to the Internet of Things. Readers and researchers will gain a better grasp of the Internet of Things and its practical implications after reading this article.

Smart buildings that can take use of the Internet of Things are gaining popularity. In spite of this, it can be difficult to store and analyze a vast quantity of high-speed real-time data from smart buildings. In order to overcome this obstacle, several cutting-edge Big Data management tools and sophisticated analytical methods have been developed in recent years. To close the knowledge gap in Big Data Analytics, a comprehensive IoT Big Data Analytics (IBDA) framework is required. In this work, we offer an IBDA framework for collecting and processing data in real time from Internet of Things (IoT) sensors located within a smart building. Python and the Cloudera Big Data platform were used to create the first iteration of the IBDA framework. Using a scenario in which real-time data from a smart building is analyzed to automatically manage factors like oxygen concentration, lighting, and the presence of smoke or toxic chemicals, we show how this framework can be put to use. Based on these first findings, it appears that the suggested framework is beneficial for IoT-enabled Big Data Analytics in the context of smart buildings. The complicated combination of Big Data Analytics and IoT shown in this study by M R Bashir and A Q Gill (2016) is a significant step toward solving the enormous volume and velocity difficulty of real-time data in the smart building area [13]. The utilization of this framework in different fields will allow for its continued assessment and development.

Massive amounts of information are being produced by the IoT. Programming and statistical methods are needed to analyze and manage such large amounts of data. Using this humongous trove of data, Big Data technology drives innovation in the form of fresh services, software, and groundwork for future study and development. K Shivanjali (2019) delves at the four Vs of Big Data and how they pertain to Internet of Things (IoT)-driven technologies [14]. This study also discusses how big data is intertwined with IoT technologies and the significance of pre-processing, meta-data, data storage formats, and data management. Since the advent of the Internet of Things, connectivity has become ubiquitous. IoT businesses are undergoing radical change as a result of the introduction of cutting-edge technologies like Cloud Computing and Edge Computing. In order to obtain low-latency and higher-efficiency solutions, this article examines in which layers of the IoT reference model edge computing operates. In addition to discussing cloud computing, its architecture, gathering data, and cleaning it up, this study examines the IoT reference model layers related to cloud computing. In addition, this paper covers a wide range of cloud-based IoT platforms, including Amazon Web Services (AWS), Google Cloud IoT, Microsoft Azure, and Cisco IoT Cloud. The authors analyzed the value of visualizing Big Data and provided insights on a variety of visualization methods. Finally, this study discusses a wide range of important obstacles presented by Big Data in the Internet of Things, including security concerns and suggestions for further studies.

The Internet has facilitated the rapid development of technology and communication, which has led to a growth in the number of machines and sensor-based devices that are interconnected. The Internet of Things (IoT) is a notion that arose from the idea of linking machines and other objects together online (Internet of Things). The Internet of Things enables the interconnection of previously unrelated objects, including smartwatches, automobiles, home appliances like washing machines, doors, door locks, lighting, etc. Big data is generated every day by the thousands by these sensor devices. Analysis of this information can help us address a variety of pressing issues. In her 2020 article, Preeti Gulia covers various Big data tools and strategies that can be applied to Internet of Things (IoT) architectures [15]. It also demonstrated how Big Data may be applied to the intelligent analysis of data sets from the Internet of Things. There is a thorough breakdown of the various Big-data Analytics systems, and guidance is provided on how to choose which is ideal for Internet of Things (IoT) information.

As the concept of the Internet of Things (IoT) continues to gain traction in the IT sector, a plethora of wireless devices are being created for the purpose of monitoring a wide range of infrastructure in order to continuously optimize data in a number of sectors, including the healthcare industry, the logistics sector, the energy sector, the agricultural intelligence sector, the building automation sector, and the data-generating sectors. Using big data approaches, IoT operations may be visualized in real time to generate actionable insights that can inform strategic decision-making. In this study, we examine in depth how large data visualization helps IoT methods, tools, and programs. The research article written by Sudhir Allam (2017) examines visual analytics by situating data visualization within the visual analysis stage. It provides a look at the many data visualization tools out there and an analysis of guidelines for using each one, taking into account the unique requirements of each application. This study will investigate visualization difficulties and how big data affects the Internet of Things (IoT), despite the fact that big data technologies are compartmentalized for each IoT domain [16]. Literature reviews are needed to set the framework of the study and ensure a thorough investigation of the issue. This study does not present any conclusions, but it does summarize current methods for visualizing massive data in the Internet of Things and their applications to deep learning. In addition, we'll discuss how big data plays a part in IoT visualization. The paper focuses on demonstrating the fundamental ideas of Big Data visualization in a time-based context.

In the year 2020 and beyond, the rate at which data is being generated and sent via the Internet is expected to expand exponentially, as will the number of devices and equipment that are connected to the Internet for the same purposes. Combining the relatively recent developments of

big data and the Internet of Things could result in a technological sea change. In most cases, only about an eighth of an iceberg's total mass is exposed above water. Nearly eight percent of it stretches out into the water, beyond our line of sight. It's the same with data: we collect it in vast quantities across all fields, but only a fraction of it is ever put to good use. We now have massive amounts of hidden information. Therefore, it is important to analyze this data and integrate big data with the Internet of Things. This paper by Mrs. Poonam (2021) defines key concepts related to Big Data and the Internet of Things, explores the role that Big Data plays in the IoT, and looks at several data visualization techniques for making sense of all that information [17].

The number of devices connected to the Internet of Things (IoT) keeps growing. IoT big data, collected by Yuya Sasaki (2022), consists of a wide variety of IoT data from Internet-connected devices such cars, smart appliances, smartphones, and environmental sensors. In order to optimize urban planning, address air pollution issues, and enhance corporate decisions, among other possible use cases, effective analytic tools are required to process the massive amounts of data produced by the Internet of Things (IoT). In this comprehensive overview, the author examines existing tools that can effectively evaluate data from the Internet of Things. Currently available technologies can be split into two distinct types: batch and stream processing. As such, they investigate batch processing systems for spatial, temporal, and trajectory data, specifically those based on Hadoop and Spark. They also examine stream processing architectures that are aware of the fog and the periphery [18]. While several current systems are capable of doing analyses on specific datasets and activities, none of them are equipped to deal with the whole gamut of features that make up IoT big data. The author discusses the future of IoT big data analysis systems and presents some outstanding questions in the field. The results of this poll will be used to inform the design of future Internet of Things (IoT) big data analytical systems and improve our understanding of existing ones.

3. Materials and Methods

3.1 Big data technology analysis

Hadoop is a distributed software that processes massive amounts of data. It can be built in an inexpensive cluster environment. Hadoop Distributed File System (HDFS), MapReduce (MR), and Yet Another Resource Negotiator (Yarn) are the three core components of Hadoop [19,20].

HDFS is designed based on the Google File System (GFS), a low-cost distributed file system. Distributed file system refers to planning multiple computers under the same local area network as a cluster. Any computer in the cluster can access the resources in the cluster [21]. The architecture of HDFS is shown in Figure 1.

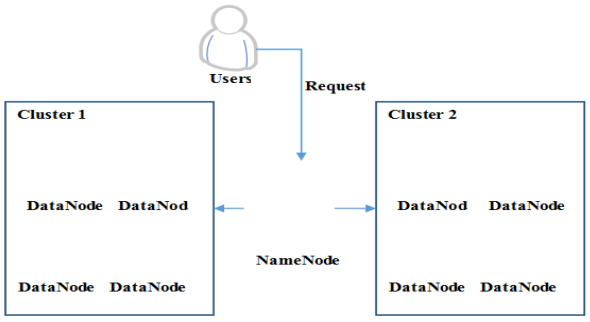

Figure 1. Structure of HDFS

In Figure 1, HDFS is a master-slave architecture. A computer in the mainframe is deployed as the Name Node, responsible for external communication and cluster node management. The rest of the computers are deployed as Data Nodes, responsible for maintaining their own data content. The system, based on the Transmission Control Protocol (TCP) protocol, communicates with the Master through a long connection and regularly sends a data packet to notify the Master of its own existence [22].

MapReduce (MR) is a computing model for big data. It divides complex problems into several small tasks through the algorithm of divide and conquers and iterates the two processes of the map and reduce to obtain the original solution [23]. The operation process of MR is shown in Figure 2:

In Figure 2, an application submits a job to the Job Tracker. Additionally, the Job Tracker must start the task, which consumes many resources. There is only one Job Tracker in the entire cluster, and it is easy to have a single point of failure when the resource consumption is so severe [24]. Hadoop2.x version designed a new resource scheduler, Yarn, also known as MRv2 [25].

The core idea of Yarn is to divide the Job Tracker function of MRv1 into two components to solve the problem of excessive burden [26]. Yarn is a master-slave architecture. The basic framework of Yarn is shown in Figure 3.

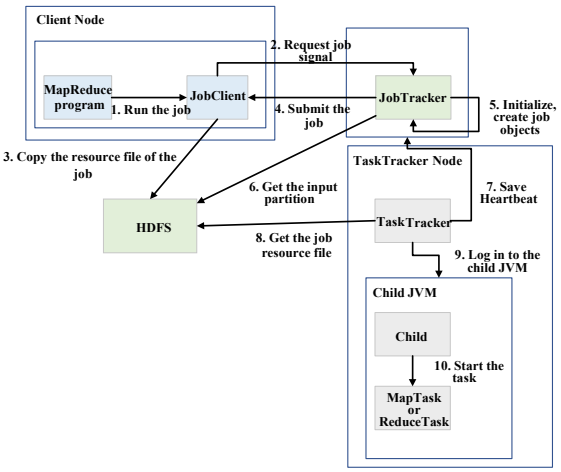

Figure 2. MapReduce startup process

In Figure 3, Yarn comprises four components: Resource Manager, Node Manager, Application Master, and Container. The Resource Manager replaces the task scheduling function of the Job Tracker, like the Name Node in HDFS. It is a master node responsible for external communication and node resource management. Node Manager is responsible for resource management of a single node, like Data Node in HDFS. Application Master assumes the function of Task Tracker to manage running application instances [27]. Each task can only use all the resources of the corresponding Container at most.

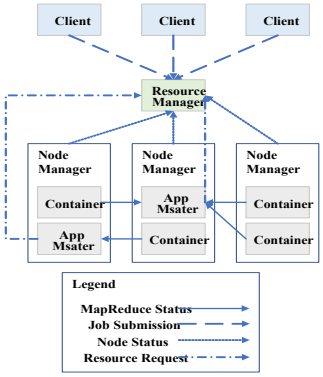

Figure 3. The basic framework of Yarn

The Hadoop platform is widely used in big data, but its core module MR still has major defects. MR does not support in-memory computing. All data is stored on a disk, which consumes much time during the calculation process, resulting in computational delays. This computational model is only suitable for batch processing data, and it is more difficult to deal with some real-time problems [28]. The Spark computing framework is developed to solve these problems. Spark's in-memory computing features make Spark quickly become the best way to meet big data business. There are four main components of Spark running in a cluster: resource manager, task driver, worker node, and execution driver. Its programming model is shown in Figure 4.

In Figure 4, when a Spark application is submitted to the cluster, the Spark Context is created and connected to the Cluster Manager to apply for the Executor. After the Executor receives the task, it starts a thread from the thread pool to execute the task and returns the result after the calculation is completed [29].

3.2 Data acquisition scheme of application circuit based on an FPGA chip

The design method of FPGA includes hardware and software design. Hardware includes FPGA chip circuits, memory, input and output interface circuits, and other devices. Software is the corresponding program. The application circuit data acquisition scheme based on an FPGA chip treats FPGA as a data node. The network communication sends the data to the big data visualization platform to realize the data collection function of the platform [30]. The specific process is shown in Figure 5.

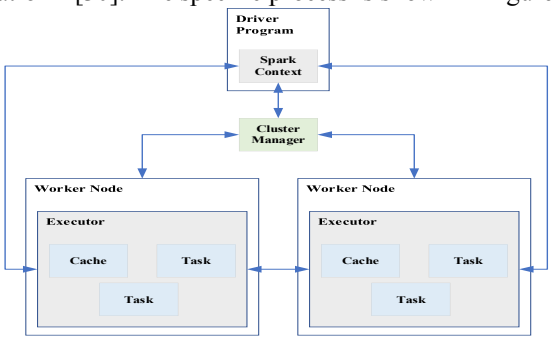

Figure 4. Spark's computing model

In Figure 5, the FPGA remote data acquisition module is designed using wireless node data. Data collection uses Microcontroller Unit (MCU) as the main control. The power-on and power-off detection of the FPGA and the Done program download signal detection record the FPGA startup time, shutdown time, and the number of program programming. This data is stored in Flash and Real-Time Communication (RTC) and is sent locally to the cloud via the Serial Peripheral Interface (SPI) protocol using WIFI capabilities.

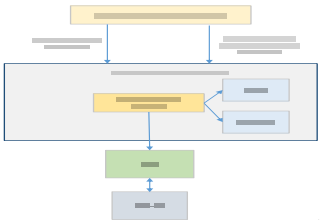

Figure 5. Data acquisition scheme based on FPGA chip

3.3. The basic theory of ant colony algorithm

Ant colony algorithm is obtained by simulating the activities of ants foraging. Ants leave pheromones as marks as they crawl. The higher the concentration of pheromone, the more ants are attracted. This phenomenon is called the positive feedback phenomenon. In the ant colony algorithm, the value of the objective function is regarded as food, and the adaptive storage corresponding to the function is the path that leaves the pheromone. In constructing the objective function, the solution components corresponding to the assumptions that meet the definition are continuously added to the partial solution to obtain the complete solution [31].

Ant colony algorithm mainly includes two processes: constructing a path and updating pheromone. When building a path, a random scaling rule is used as a probability to choose the next path. The random proportion rule is shown in Eq. (1):

$$q_{ij}{}^{h}(t) = \frac{\omega_{ij}{}^{\alpha}(t)\eta_{ij}{}^{\beta}}{\sum_{j \in M_i^h} \omega_{ij}(t)^{\alpha}\eta_{ij}{}^{\beta}} \quad (1)$$

i and j points represent the start and end points of the path; η_{ij} represents the derivative of the distance between the start point and the endpoint of the path; $\omega_{ij}(t)$ represents the concentration of pheromone; M_i^h represents the set of path nodes. α and β are weighted values, representing importance [32].

The ways of updating pheromone include pheromone volatilization and pheromone enhancement. When the endpoint is reached, the pheromone concentration of each path is updated. Firstly, at the end of each round, all the pheromones along the path evaporate. Then, the pheromone is released according to the path length constructed by each ant's current round [33], as shown in Eqs. (2) and (3):

$$\omega_{ij}(t) = (1-\lambda)\omega_{ij} + \sum_{h=1}^{m} \Delta\omega_{ij}^{h} \tag{2}$$

$$\omega_{ij} = \begin{cases} F/l_h & If\ the\ hth\ ant\ goes\ through\ the\ path\ ij \\ 0 & otherwise \end{cases} \tag{3}$$

m represents the number of ants; λ represents the volatilization rate of pheromone; $\Delta\omega_{ij}^{h}$ represents the pheromone left by the h-th ant on the path ij; l_h represents the sum of the lengths of the path that the h-th ant passes through [34]. Ant colony algorithm has strong adaptability and robustness. It can share its own information, automatically adjust the search direction, and adjust the random proportion rules according to different problems. By modifying the model of the ant colony algorithm, the Hadoop task scheduling problem, which is a combinatorial optimization problem, is solved, which can not only reduce the difficulty of solving but also shorten the solving time [35].

3.4 Establishment of an improved ACOS algorithm model

The ACOS algorithm is modified from the ant colony algorithm, and some assumptions are made before the algorithm modeling. 1. The entire cluster is built on cheap machines. The hardware configuration of each node is not the same, and the performance varies. 2. The resources owned by each node are known, and the task's execution time can be estimated by the node resources and the task size. 3. Each task is relatively independent. After the task is submitted, the execution process cannot be interrupted until the end of the task execution. 4. When the cluster executes multiple tasks, the resource load generated by the different resources required by the tasks is also different [36,37].

The basic parameters of the ACOS are defined as follows:

Definition 1: task set T, $T = \{T_1, T_2, \cdots, T_n\}$, T represents the task set of resources to be allocated in the current cluster; $T_1, T_2, \cdots, T_n$ represent n independent tasks.

Definition 2: cluster N, $N = \{N_1, N_2, \cdots, N_m\}$, N represents all nodes in the current cluster; $N_1, N_2, \cdots, N_m$ represents m node resources in the cluster.

Definition 3: node resource N_j, each node in the cluster is a Personal Computer (PC); N_j uses a quadruple to represent the performance of a single node, $N_j = \{N_j^c, N_j^n, N_j^m, N_j^s\}$. N_j^c represents the CPU (central processing unit) performance of node j; N_j^n represents the bandwidth of node j; N_j^m represents the memory size of node j; N_j^s represents the disk speed of node j.

Definition 4: the time matrix RT is shown in Eq. (4):

$$RT = \begin{bmatrix} RT_{11} & RT_{12} & \cdots & RT_{1m} \\ RT_{21} & RT_{22} & \cdots & RT_{2m} \\ \vdots & \vdots & \ddots & \vdots \\ RT_{n1} & RT_{n2} & \cdots & RT_{nm} \end{bmatrix} \tag{4}$$

In Eq. (4), RT_{nm} represents the pre-completion time of task n on node m; when task n is not allocated to node m, RT_{nm} is 0.

Definition 5: the time for node j to complete all tasks is Time $_j$, as shown in Eq. (5):

$$\text{Time}_j = \sum_{i=1}^{n} RT_{ij} \tag{5}$$

The definitions of the terms related to the ACOS are described by defining the basic parameters.

The goal of the ACOS is to find the optimal solution for Hadoop task scheduling and the indicator to determine whether the optimal solution is the overall task completion time. The overall task completion time should be the latest time for all nodes to complete the task $\max\{\text{Time}_1, \text{Time}_2 \cdots, \text{Time}_m\}$, and $\min\{\max\{\text{Time}_1, \text{Time}_2 \cdots, \text{Time}_m\}\}$ is the optimal solution required.

The pheromone concentration affects the transition probability during the search process of the ant colony. The greater the pheromone concentration, the greater the probability that the ant colony will assign the task to the node [38]. The performance of the node is the quadruplet N_j. The structure of the initial pheromone concentration is shown in Eq. (6):

$$\begin{cases} \tau_j(0) = a\frac{N_j^c}{N^c} + b\frac{N_j^s}{N^s} + c\frac{N_j^m}{N^m} + d\frac{N_j^n}{N^n} \\ N^x = \frac{\sum_{j=1}^{m} N_j^x}{m}, (x = c, s, m, n) \end{cases} \tag{6}$$

In Eq. (6), N^c, N^s, N^m and N^n represent the average node value of each performance parameter; a, b, c, and d represent the weight factor of each item, and their sum is 1.

The heuristic function $\eta_j(t)$ represents the expectation that task i is assigned to node j at any time t. When the expectation is higher, the probability that task i is assigned to node j should be greater. The more time a node spends performing tasks, the less expected that the node should be assigned tasks. The constructed expectation function is shown in Eq. (7)

$$\begin{cases} \eta_j(t) = \begin{cases} \frac{\text{Time}_{cur\_opt}}{\text{Time}_j}, t > 0 \\ 1, t = 0 \end{cases} \\ \text{Time}_c = \frac{\sum_{i=1}^{n}\sum_{j=1}^{m} RT_{ij}}{m} \end{cases} \tag{7}$$

In Eq. (7), Time $_j$ is the time when node j has completed the task at time t. When Time $_j$ is larger, $\eta_j(t)$ is smaller. The expectation is set to be a constant one at the initial time. Time_c is the average running time of the current optimal

solution obtained from the previous round of search on each node [39].

At any time t, the probability of ant k in the ant colony assigning task i to node j is shown in Eq. (8):

$$p_j^k(t) = \begin{cases} \frac{[\tau_j(t)]^{\alpha} \cdot [\eta_j(t)]^{\beta}}{\sum_{sc \text{ allowed}_k} [\tau_s(t)]^{\alpha} \cdot [\eta_s(t)]^{\beta}}, j \in \text{allowed}_k \\ 0, \text{Other} \end{cases} \quad (8)$$

In Eq. (8), allowed_k represents the node to which ant k can assign tasks; $\tau_j(t)$ represents the pheromone concentration of node j at time t; α and β are weighted values representing the degree of importance [40].

The ACOS accelerates the convergence speed of the ant colony by secreting pheromones on the nodes. Therefore, after a task i is assigned to node j. The pheromone concentration on node j needs to be updated. The update of the constructed pheromone is shown in Eq. (9):

$$\tau_j(t+1) = (1-\rho)\tau_j(t) + \Delta\tau_j(t) \quad (9)$$

ρ is the volatile factor. In order to prevent the increase of the pheromone concentration and mask the influence of the heuristic function, the pheromone concentration is reduced by volatilization before each update of the pheromone concentration. The larger the volatile factor, the lower the residual pheromone concentration [41]. $\Delta\tau_j(t)$ represents the concentration of pheromone secreted by ants on node j, as shown in Eq. (10):

$$\Delta\tau_j(t) = \frac{Q}{\text{Time}_j} \quad (10)$$

In Eq. (10), Time_j represents the completion time of the assigned task on node j during the i-th search. The node pheromone concentration increases more with shorter task running time, and Q is constant. Nodes that are not assigned tasks $\Delta\tau_j(t) = 0$.

When the ant colony assigns tasks, they will be assigned according to the state transition probability $p_j^k(t)$. However, if the task is only assigned to the node with the largest $p_j^k(t)$ each time, the entire algorithm may converge to a locally optimal solution. Therefore, a policy is set to ensure that every node is likely assigned a task. The probability of being assigned a task is related to the size of $p_j^k(t)$. The roulette strategy is a simple algorithm strategy. The specific implementation process is to calculate the $p_j^k(t)$ of each node and record it as the fitness f_j. The selected probability of each node is shown in Eq. (11):

$$P_j = \frac{f_j}{\sum_{i=1}^{m} f_i} \quad (11)$$

In Eq. (11), $\sum_{j=1}^{m} P_j = 1$. According to the fitness of each node, P_j is calculated and mapped to the roulette interval size, and the interval size is proportional to P_j. The program generates a random number between [0, 1], and when the random number falls into a certain interval. It is selected to submit a task to the node corresponding to the interval. The roulette strategy can realize that the task can be submitted to any node with probability. This probability is proportional to $p_j^k(t)$[42].

A genetic algorithm is an adaptive global optimization search algorithm. Its algorithm idea learns the genetic theory of biological inheritance and evolutionary phenomena in genetics. It can actively obtain the relevant content of the solution space in the process of algorithm implementation to realize its self-adaptive solution process. A genetic algorithm has the characteristics of a wide search range, strong randomness, and strong adaptability. The integration of genetic algorithm and ACOS algorithm can form an improved ACOS algorithm [43,44]. The specific process is shown in Figure 6.

The specific steps include:

Step 1 sets the initial value parameters and the population size of the genetic algorithm. When $N = 100$, the crossover probability is 0.6, the mutation probability is 0.1, and the population iteration number for initializing the genetic algorithm is set to $NC = 0$.

Step 2 codes the genetic algorithm and calculates the current fitness value.

Step 3 selects the best individual to enter the next generation.

Step 4: the probability of selection, crossover, and mutation can realize genetic operation.

Step 5 calculates the fitness value of individuals in the new population.

Step 6 adds 1 to the evolutionary algebra, that is, $NC = NC + 1$.

Step 7 determines whether the end condition of the genetic algorithm is satisfied and goes to the next step.

Step 8 selects the individual with the highest fitness as the optimal solution and uses the optimal solution to initialize the ant colony pheromone value.

Step 9: the selection probability p_j^k calculates the probability that the ant selects the server node and determines whether all nodes are selected. If so, go to the next step.

Step 10 performs a pheromone update for each resource according to the global pheromone update algorithm.

Step 11 determines whether the termination condition is reached.

This experiment mainly uses the job completion time index to test the effect of the improved ACOS algorithm on the optimization of the visualization platform. A Hadoop cluster is installed in a Linux system to implement the algorithm. The Java language develops the code logic. The improved ACOS, first-in-first-out scheduler (FIFOS), capacity scheduler, and fair scheduler complete the same job content in the same environment time. The test program selects WordCount. The program will specify all words in the document for word frequency statistics and sort according to the statistical results. The delimiter between words is specified as a single space. In order to ensure the reasonableness of the test results, the data set needs to simulate the randomness of actual business scenarios. Part of the tested data set comes from the "Collection of Names of Chinese Institutions of Higher Education" downloaded from the government website, and part comes from the electrical and electronic laboratories of different universities. The total data set is randomly selected by the program, and the word text is stored in a specified text. The size of each text is about 100MB, and the implementation of each word frequency in the test data set is different. The WordCount program is used to count the word frequency of the text.

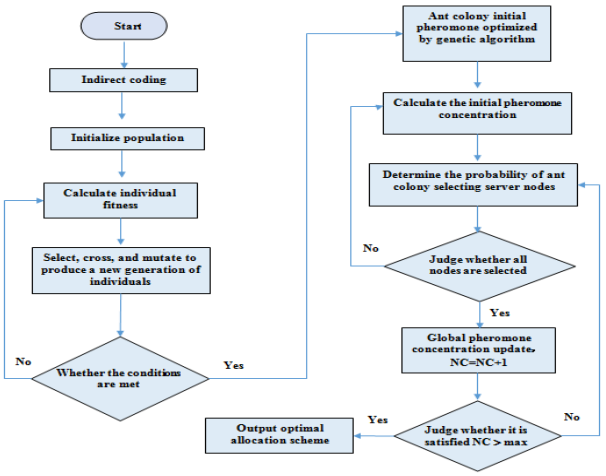

Figure 6. Flow of the improved ACOS algorithm

4 Results and Discussion

4.1 Test analysis of schedulers under homogeneous cluster

A homogeneous cluster means that all nodes in the cluster have the same performance, including CPU, memory, disk, and bandwidth. The virtual machine cloned five identical nodes and configured the cluster in the homogeneity test environment to ensure the consistency of the cluster. The generated test data set is used to test on multiple data sets. The test results are shown in Figure 7:

In Figure 7, the four scheduling methods change the completion time when the task volume gradually increases. Among them, the FIFO has the worst effect because the scheduler only maintains one queue internally. Only after the task at the top of the queue has allocated resources will the next task resource allocation be made. Capacity Scheduler has multiple queues, and Fair Scheduler maintains multiple resource pools so multiple tasks can share cluster resources, which works best in a homogeneous environment. ACOS consumes a part of the time because it needs to find the optimal solution through an iterative algorithm first. In a homogeneous cluster, there is no obvious difference in the computing power of nodes, resulting in an insignificant algorithm effect, and the overall completion time lags Capacity and Fair Scheduler.

Figure 7. Comparison of scheduling algorithms under homogeneous clusters

4.2 Test and Analysis of Schedulers in Heterogeneous Clusters

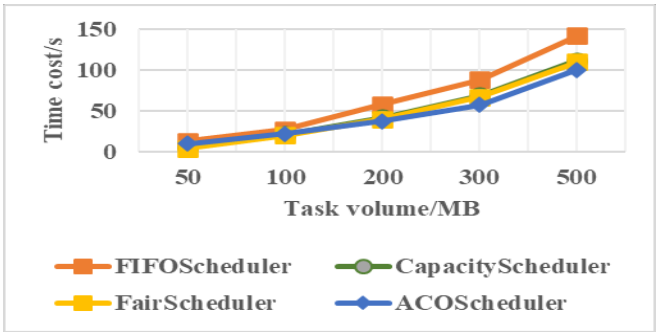

Figure 8. Comparison of scheduling algorithms under heterogeneous clusters

A heterogeneous set refers to the inconsistent performance of each node in the cluster and is also the most common cluster construction mode in actual production. In order to ensure the heterogeneity of the cluster, five nodes with different performances are set up through virtual machines and configured as a cluster in a heterogeneous test environment. The dataset is consistent with a homogeneous cluster. The test results are shown in Figure 8:

In Figure 8, the four scheduling methods change the completion time when the task volume gradually increases. Among them, the FIFO scheduler is limited by a simple model, and the effect is relatively poor in homogeneous or heterogeneous clusters. The result shows that FIFO Scheduler is not suitable for task scheduling in cluster mode. Different from homogeneous clusters, ACO Scheduler needs algorithm iteration when the amount of tasks is small, which makes the completion time longer. As the amount of tasks increases, the proportion of task processing time begins to be higher than the algorithm iteration time. The ACO Scheduler is starting to show its dominance. Tasks are reasonably allocated to nodes with different performances, reducing task completion time. When the task volume is 500MB, the task completion time of ACO Scheduler is reduced by 9.8% compared with Fair Scheduler. The result is 11.4% lower

than Capacity Scheduler, which optimizes the performance of the entire platform.

4.3 Analysis of time to complete different number of tasks

The four schedulers test the completion time of scheduling different numbers of tasks in the same environment. In the experiment, there are 50 server nodes and 50-200 tasks. The specific results are shown in Figure 9.

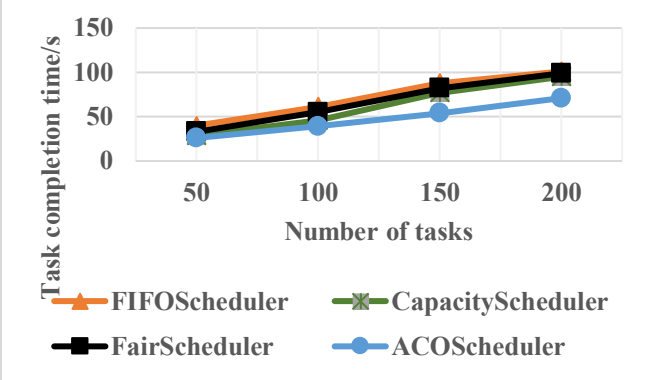

Figure 9. Time comparison of different schedulers to complete different numbers of tasks

In Figure 9, the completion time of the four schedulers increases with the number of tasks. Under the same number of tasks, the completion time of different schedulers shows a certain change law. The ACO Scheduler has the shortest completion time, followed by the Capacity Scheduler, and the FIFO Scheduler with the longest completion time. When the number of tasks is 50, the completion time of ACO Scheduler is 26s; when the number of tasks is 200, the completion time is 71s, and the time difference is 45s. The time difference of the other three schedulers is all greater than the 60s. The improved ACOS algorithm has the shortest data processing time and the highest efficiency.

4.4 Analysis of CPU utilization of different task volumes

This part aims to investigate the CPU utilization of the four schedulers testing the scheduling of different amounts of tasks in the same environment. The task volume is 50-500MB. The specific results are shown in Figure 10.

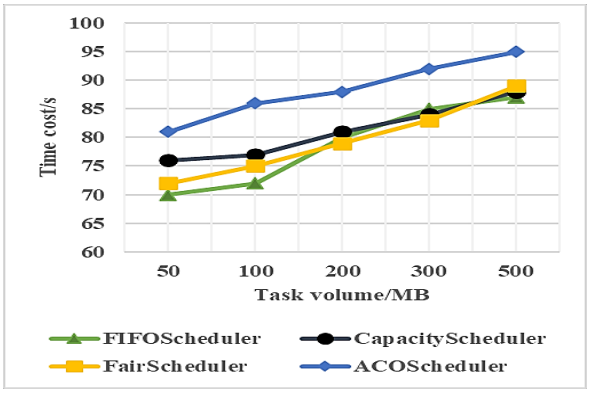

Figure 10. Comparison of CPU utilization of tasks completed by different schedulers

In Figure 10, as the number of tasks increases, the CPU utilization of the ACO Scheduler is also increasing, and the utilization rate increases from 81% to 95%, which is higher than other schedulers. The improved ACOS algorithm utilizes the advantages of the genetic algorithm to find server nodes with sufficient resources faster in the early stage, reducing the idle time of the CPU and improving CPU utilization. This algorithm maintains good effectiveness in high-performance computing task scheduling.

Experiments show that the improved ACOS algorithm improves the efficiency of task scheduling under heterogeneous clusters, shortens the computing time of data processing, and optimizes the performance of the FPGA big data visualization platform. Additionally, the utilization of the CPU is improved. The larger the data, the more obvious the algorithm effect is.

5. Conclusion

In order to design a big data collection and analysis platform for FPGA chip-based application circuit devices, this study designs a big data visualization platform deployment strategy based on the ACOS algorithm. Ant colony algorithm is used as the basis. The fusion genetic algorithm is used to construct the model based on the improved ACOS algorithm. Finally, the completion time of the ACO scheduler, FIFO scheduler, capacity scheduler, and fair scheduler for scheduling the same job content in the same environment and the completion time of the different number of tasks are compared. The CPU's use for various job volumes is compared and examined. The outcomes demonstrate that the fair scheduler and capacity scheduler generally outperform the modified ACOS algorithm scheduler in terms of completion time for homogeneous clusters. The upgraded ACOS algorithm scheduler must execute algorithm iteration when there are few tasks, which lengthens completion time based on the heterogeneous cluster. As the number of activities rises, the time required to accomplish each work decreases when compared to alternative schedulers, and the benefits become increasingly clear. The duration increases by 45s as the number of completed tasks rises from 50 to 200, and the completion time is quicker than that of other schedulers. The maximum

CPU use, which rises from 81% to 95%, is the one that can finish various tasks. The scheduler for the upgraded ACOS algorithm processes data the fastest and most effectively. The developed big data visualisation platform based on the upgraded ACOS algorithm model has superior job scheduling efficiency when compared to existing schedulers, which can significantly reduce the amount of time needed to process data and increase CPU usage. Due to the energy constraints, the suggested fusion algorithm can only transform the great genetic algorithm result into the starting pheromone of the ACOS algorithm. The genetic algorithm's original population will eventually incorporate the ACOS algorithm for advancement. To improve the performance of the algorithm, the iteration time will be further optimised.

Data Availability: The data used to support the findings of this study are available from the corresponding author upon request.

Conflicts of Interest: The authors declare that they have no competing interests.

References

[1] Arif C, Setiawan BI, Saptomo SK, Taufik M, Wiranto, Mizoguchi M. Developing it infrastructure of evaporative irrigation by adopting iot technology. IOP Conference Series: Earth and Environmental Science. 2021, 622(1).

[2] Xu Y, Zhang Z, Liu M. Design of cancer classification and visualization platform based on Internet big data. Journal of Physics: Conference Series.2020,1650(3).

[3] Song Z, Yang Y, Guo H. Analysis of data crawling and visualization methods for recruitment industry information. Journal of Physics: Conference Series. 2021, 1971(1).

[4] Davis R, Vochozka M, Vrbka J, Octav N. Industrial Artificial Intelligence, Smart Connected Sensors, and Big Data-driven Decision-Making Processes in Internet of Things-based Real-Time Production Logistics. Economics, Management, and Financial Markets. 2020, 15(3): 9-16.

[5] Cocoros NM, Kirby C, Zambarano B, Ochoa A, Eberhardt K, Rocchio Sb C, Ursprung WS, Nielsen VM, Durham NN, Menchaca JT, Josephson M, Erani D, Hafer E, Weiss M, Herrick B, Callahan M, Isaac T, Klompas M. RiskScape: A Data Visualization and Aggregation Platform for Public Health Surveillance Using Routine Electronic Health Record Data. Am J Public Health. 2021,111(2):269-276.

[6] Ming C, Bo L, Zuo H. College Entrance Examination Voluntary Filing System Based on Big Data. International Journal of Advanced Research in Big Data Management System. 2018, 2(21): 23-36.

[7] Chen J, Tian J, Jiang S, Zhou Y, Li H, Xu J. The Allocation of Base Stations with Region Clustering and Single-Objective Nonlinear Optimization. Mathematics. 2022, 10(13): 2257.

[8] Harb H, Mroue H, Mansour A, Nasser A, Cruz Em. A hadoop-based platform for patient classification and disease diagnosis in healthcare applications. Sensors. 2020, 20(7):1931.

[9] Atat R, Liu L, Wu J, Li G, Ye C, Yang Y. Big data meet cyber-physical systems: A panoramic survey. IEEE Access. 2018, 6: 73603-73636.

[10] Zhao Y, Ye P, Yang K, Meng J, Lei M. A field programmable gate array based synchronization mechanism of analog and digital local oscillators in bandwidth-interleaved data acquisition systems. Review of Scientific Instruments. 2021, 92(3).

[11] Protopsaltis A, Sarigiannidis P, Margounakis D, Lytos A.. Data Visualization in Internet of Things: Tools, Methodologies, and Challenges. ARES '20: Proceedings of the 15th International Conference on Availability, Reliability and Security, 2020, 1-11.

[12] Kumar S, Tiwari P, Zymbler M. Internet of Things is a revolutionary approach for future technology enhancement: a review. J Big Data. 2019, 6(111).

[13] Bashir MR, Gill AQ. Towards an IoT Big Data Analytics Framework: Smart Buildings Systems. 2016 IEEE 18th International Conference on High Performance Computing and Communications; IEEE 14th International Conference on Smart City; IEEE 2nd International Conference on Data Science and Systems (HPCC/SmartCity/DSS), Sydney, NSW, Australia, 2016, pp. 1325-1332.

[14] Khare S, Totaro M. Big Data in IoT. 2019 10th International Conference on Computing, Communication and Networking Technologies (ICCCNT), Kanpur, India, 2019, pp. 1-7.

[15] Preeti G, Ayushi C. Big data analytics for IOT. International Journal of Advanced Research in Engineering and Technology (IJARET). 2020, 11(6): pp. 593-603.

[16] Sudhir A. Exploratory study for big data visualization in the Internet of things. International Journal of Creative Research Thoughts (IJCRT). 2017,5(3):805-809.

[17] Mrs Poonam and Mrs Aditi Mittal. Eminent Data Visualization Tools for Integration of Big Data with IoT. International Journal of Advanced Research in Science, Communication and Technology (IJARSCT). 2021, 5(1).

[18] Yuya S. A Survey on IoT Big Data Analytic Systems: Current and Future. IEEE Internet of Things Journal. 2022, 9(2).

[19] Pandey V, Saini P. Constraint programming versus heuristic approach to MapReduce scheduling problem in Hadoop YARN for energy minimization. J Supercomput . 2021,77: 6788-6816.

[20] Bawankule KL, Dewang RK, Singh A K. Performance Analysis of Hadoop YARN Job Schedulers in a Multi-Tenant Environment on HiBench Benchmark Suite. International Journal of Distributed Systems and Technologies (IJDST). 2021,12( )3: 64-82.

[21] Saraswat H, Sharma N. Enhancing the Traditional File System to HDFS: A Big Data Solution. International Journal of Computer Applications. 2017, 167(9):975-8887.

[22] Ergüzen, ünver M. Developing a File System Structure to Solve Healthy Big Data Storage and Archiving Problems Using a Distributed File System. Applied Sciences.2018, 8(6):913.

[23] Pachghare A, Jadhav A, Panigrahi S, Smitha D. Implementation of MapReduce Using Pig for Election Analysis. International Conference on Innovative Computing and Communications. 2019, 56: 231.

[24] Kim YP, Hong CH, Yoo C. Performance impact of JobTracker failure in Hadoop. International journal of communication systems. 2015, 28(7):1265-1281.

[25] Subbulakshmi T, Manjaly JS. TaskTracker Aware Scheduler with Resource Availability Control for Hadoop MapReduce. International Journal of Advanced Intelligence Paradigms. 2018,1(1):1.

[26] Huang W, Meng L, Zhang D, Zhang W. In-Memory Parallel Processing of Massive Remotely Sensed Data Using an Apache Spark on Hadoop YARN Model. IEEE Journal of Selected Topics in Applied Earth Observations & Remote Sensing. 2017, 10(1): 3-19.

[27] Jarrah M, Al-Quraan M, Jararweh Y, Al-Ayyoub M. MedGraph: a graph-based representation and computation to handle large sets of images. Multimedia Tools & Applications. 2017, 76(2):2769-2785.
[28] Chen J, Li K, Zhuo T, Bilal K, Yu S, Weng C, Li K. A Parallel Random Forest Algorithm for Big Data in a Spark Cloud Computing Environment. IEEE Transactions on Parallel & Distributed Systems. 2017, 28(4): 919-933.
[29] Guo Y, Zhang Z, Jiang J, Wu W, Zhange C, Cui B, Li J. Model averaging in distributed machine learning: a case study with Apache Spark. The VLDB Journal. 2021, 30(4):693-712.
[30] Fu SY. Design of high speed data acquisition system for linear array CCD based on FPGA. Procedia Computer Science. 2020, 166: 414-418, 2020.
[31] Mendrofa H, Muis A. Serial Manipulator Control Optimization Using Ant Colony Algorithm. Journal of Physics: Conference Series. 2021, 1993(1).
[32] Wan L, Du C. An approach to evaluation of environmental benefits for ecological mining areas based on ant Colony algorithm. Earth Science Informatics. 2021,14(2):797-808.
[33] Yi G, He Y, Gao L, He W. Propagation Path Optimization of Product Attribute Design Changes Based on Petri Net fusion Ant Colony Algorithm. Expert Systems with Applications. 2021, 173.
[34] Shi, Zhang Y. A Novel Algorithm to Optimize the Energy Consumption Using IoT and Based on Ant Colony Algorithm. Energies. 2021, 14(6):1-17.
[35] Wu F. Contactless Distribution Path Optimization Based on Improved Ant Colony Algorithm. Mathematical Problems in Engineering. 2021,7: 1-11.
[36] Lv G, Chen S. Routing optimizationin wireless sensor network based on improved ant colony algorithm. International Core Journal of Engineering. 2020, 6(2):1-11.
[37] Ghosh M, Dey N, Mitra D, Chakrabarthi A. A novel quantum algorithm for ant colony optimization. IET Quantum Communication. 2021, 3(1):13-29.
[38] Sekiner SU, Shumye A, Geer S. Minimizing Solid Waste Collection Routes Using Ant Colony Algorithm: A Case Study in Gaziantep District. Journal of Transportation and Logistics. 2021, 6(1): 29-47.
[39] Euchi J, Sadok A. Optimising the travel of home health carers using a hybrid ant colony algorithm. Transport. 2021, 3:1-22.
[40] Chen Y, Zhou X. Path Planning of Robot Based on Improved Ant Colony Algorithm in Computer Technology. Journal of Physics: Conference Series. 2021, 1744(4).
[41] Zhou Y, Fu X. Research on the combination of improved Sobel operator and ant colony algorithm for defect detection. MATEC Web of Conferences. 2021, 336(11).
[42] Wang Y, Yang R R, Xu YX, Li X, Shi JL. Research on Multi-Agent Task Optimization and Scheduling Based on Improved Ant Colony Algorithm. IOP Conference Series: Materials Science and Engineering. 2021, 1043(3).
[43] Luan J, Yao Z, Zhao F, Song X. A novel method to solve supplier selection problem: Hybrid algorithm of genetic algorithm and ant colony optimization. Mathematics and Computers in Simulation.2019,156:294-309.
[44] Cai L, Qi Y, Wei W, Wu J, Li J. mrMoulder: a recommendation-based adaptive parameter tuning approach for big data processing platform. Future Generation Computer Systems. 2019, 93: 570-582, 2019.