Retraction Notice

The Editor-in-Chief and the publisher have retracted this article, which was submitted as part of a guest-edited special section. An investigation uncovered evidence of systematic manipulation of the publication process, including compromised peer review. The Editor and publisher no longer have confidence in the results and conclusions of the article.

HL and BX either did not respond directly or could not be reached.

Image segmentation and visualization allocation method of engineering training resources in biology training

Huixing Li[a],* and Bin Xu[b]
[a]Nanyang Institute of Technology, School of Biological and Chemical Engineering, Nanyang, China
[b]Nanyang Institute of Technology, Zhang Zhongjing School of Chinese Medicine, Nanyang, China

Abstract. Existing processes for visual allocation of engineering training resources utilizing computers are inefficient and extend the time needed to complete learning tasks. A new process visual allocation method in engineering training resources in a biology teaching process is proposed. The biological teaching engineering training resource perception module is set up to send the resource data to the controller through the module. By quantifying the computing power and resource consumption of each node, the nodes that currently are bottlenecks as well as nodes with potentially more computing power yet highly likely to stand idle (high-potential idle nodes) are found. Then, the optimal data allocation proportion is found through the idea of greedy search, and the amount of data actually processed by parallel instances on each node is adjusted to match its computing power. According to the dynamic partition strategy of engineering training resources, a process visual allocation model is established by using ant colony scheduling algorithm. Taking the task volume of 500 MB as an example, in the homogeneous cluster environment, the completion time of this method is 89.7 s, which is 28.6 and 30.9 s shorter than the methods based on cloud computing and mobile edge computing; in heterogeneous cluster environment, the completion time of this method is 99.4 s, which is 41.4 and 43.1 s shorter than the methods based on cloud computing and mobile edge computing. Therefore, this method reduces the task completion time and improves the efficiency of task scheduling. © *2022 SPIE and IS&T* [DOI: 10.1117/1.JEI.31.6.061814]

Keywords: biology teaching process; engineering; training resources; process type; visualization; distribution.

Paper 210779SS received Nov. 2, 2021; accepted for publication Mar. 8, 2022; published online Jun. 23, 2022; retracted Jun. 28, 2023.

1 Introduction

The 21st century is an era of rapid development of science and technology. The development of science and technology has led to rapid changes in modern social life. With the increasingly fierce international competition, the requirements for all citizens' scientific literacy have been greatly improved. Many countries list science courses as the core courses. In a science curriculum, biology content accounts for a large proportion and is an organic part of scientific literacy. China's manufacturing industry has been developing into a "2025 oriented" manufacturing industry, and the "internet power" and "internet plus" strategies are in full implementation in the service of a national strategy toward achieving key technological innovation, enhancing core competitiveness, and supporting economic transformation and upgrading. There is an urgent need for a large number of high-quality individuals trained in bioengineering who have an innovative spirit and ability. The reality we face is that China is the country with the largest scale of engineering education in the world, but the quality of engineering talent is not satisfactory. The new generation of information technology industry is in urgent need of talent, yet there is a large gap in new industries and new science and technology talents.[1] Engineering practice is the cornerstone of bioengineering education. The purpose of engineering education

*Address all correspondence to Huixing Li, nithuixingli@outlook.com is not limited to knowledge, but also practical applications. Biological practice teaching can raise the learning of theoretical knowledge to the cultivation of innovative practice ability. The cultivation of engineering talent needs to be implemented into the engineering practice teaching system.[2]

The construction of scientific and reasonable engineering practice teaching system is directly related to the shaping of engineering students' practical ability and innovative thinking. To improve the quality of biology teaching, it is necessary to redesign a set of scientific and systematic engineering teaching system practices to meet the requirements of the new economy, make full use of engineering training resources, cultivate new engineering talent, and then promote the construction of new engineering.[3] However, in the process of biology teaching, the readability of engineering training resources is low and problems such as scheduling and resource allocation occur frequently.

The discrete distribution of resources also makes resources more difficult to manage, which is prone to leaving training resources idle and wasting them. These objective problems seriously limit the further improvement and development of bioengineering teaching. Therefore, this paper proposes a flow-based visual allocation method of engineering training resources in the biology teaching process, which enhances the relevance and readability of training resource information, so as to provide a guarantee of effective information for the biology teaching process.

2 Flow Visualization Allocation Method of Engineering Training Resources in Biology Teaching Process

2.1 *Set Up the Perception Module of Biological Teaching Engineering Training Resources*

The training resource awareness module is extended on the basis of the switch side code to retain the safe channel for establishing the connection between the switch and the controller. The network resource awareness module sends the data to the area controller through the intelligent identification network data forwarding module.[4,5] Before realizing specific functions, we must first complete initialization. Initialization is the basis of the whole program. Perfect initialization program design plays an important role in the realization of subsequent functions. Program initialization mainly completes the functions of global variable definition, function name registration, error handling function registration, component resource initialization, and so on. The main functions of program initialization are shown in Fig. 1.

Module registration registers the process module of the resource awareness mechanism with the component. After the module registration is completed, when the process of the resource awareness mechanism exits unexpectedly, the system can capture that process and give an error prompt through the registered name.[6] Error handling function registration uses the signal mechanism of Linux to capture some abnormal signals of the current process and call the corresponding error handling function for exception handling. This design uses the callback function processing method to process the captured signal and call the fault specified in signal. The handler function processes signals. The main function of component resource initialization is to complete the registration of the component resource information, including the following two parts: (1) call: outer info init() to complete the initial acquisition of CPU, memory and disk information, and

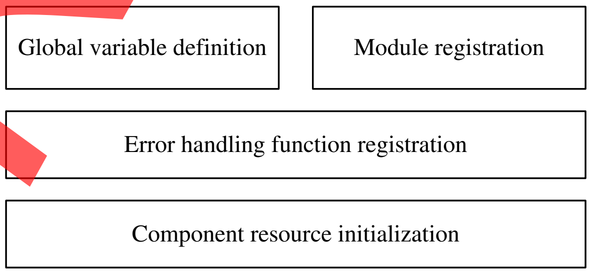
Global variable definition
Module registration
Error handling function registration
Component resource initialization

Fig. 1 Initialization task.

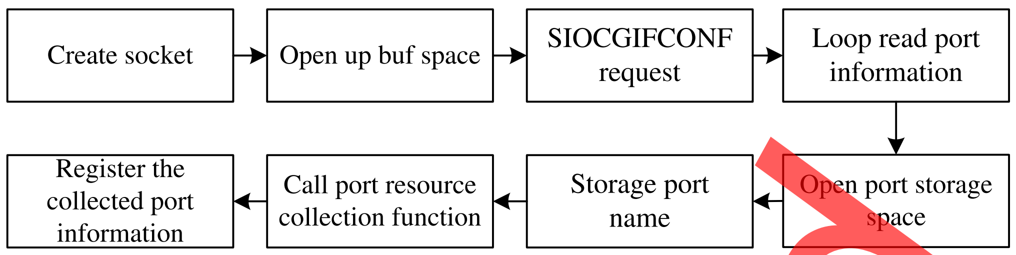
Create socket
Open up buf space
SIOCGIFCONF request
Loop read port information
Open port storage space
Storage port name
Call port resource collection function
Register the collected port information

Fig. 2 Port initialization process.

register with the regional controller, router_info_init () obtains resources by calling the resource collection module and (2) call port init() to initialize the port information. The process of port information initialization is as follows: first, create a socket and open up enough memory space to store data. Second, SIOCGIFCONF is used to obtain the list of all interfaces and store it in the struct port structure variable. For each port obtained, the port count is increased by 1. Finally, according to the interface list, call the resource collection module to obtain the port IP, MAC, and bandwidth information and complete the registration. The process of port initialization is shown in Fig. 2.

In the engineering training resource perception module, the program design mainly collects component capability information and component status information. To dynamically match user requirements, dynamically perceive and select the best service resource entity, in addition to the function of collecting component resources and topology resources, the network system also needs to have a well-designed update mechanism to feed back the changes of network components and network topology to the controller in time.[7,8] The area controller can send query or configuration messages to the lower layer components as required. The query messages sent by the controller include service resource query and network resource query. The network resource perception module completes the response to the network resource query message. The implementation method of the controller query method is to design the main calling function to receive the query variables in the network resource perception module, analyze the variables within the function, and return the component resource information according to the requirements of the controller.[9] For each type of information, the corresponding macro is defined in the program. The query function successively uses different macro values to sum with the type variable. If the result is true, the specific query function is called for resource collection. The specific contents of macro variables are the same as those in Table 1.

As shown in Table 1, the area controller does not divide the capability and status information of each type of resources in detail when querying. When the lower component receives the query message, it returns the capability and status information of each type of resources at the same time.

2.2 *Identify High Potential Nodes and Overloaded Nodes*

Based on the perception of biological teaching engineering training resources, the intent is to find all the nodes involved in the teaching task and collect the information of real-time resource

Table 1 Macro variables and their meanings.

Name	Meaning	Name	Meaning
CPU_INFO	CPU information query judgment macro	DISK_INFO	Disk information query judgment macro
BAND_INFO	Bandwidth information query judgment macro	PORT_INFO	Port information query judgment macro
MEM_INFO	Memory information query judgment macro	TOPO_INFO	Neighbor information query judgment macro

consumption. For each node, we need to score its resource consumption. According to the scores of each node, we identify the overloaded nodes that may compete for resources and the idle nodes with sufficient computing resources.[10] Then, we take the sum of the calculated resource consumption of all nodes as a unified benchmark to view the proportion of each node. After measuring the CPU utilization, memory occupancy, and IO usage of each node, we divide it by the sum of the corresponding resource consumption of all nodes to obtain the proportion of a computing resource consumption of this node in the computing resource consumption of all nodes.[11] The specific scoring strategy is as follows: the relation of CPU consumption to other variables is shown by Eq. (1)

$$\alpha = \frac{c_i}{\sum_{i=1}^{m} c_i}. \tag{1}$$

In Eq. (1), α represents the CPU utilization ratio of the node i to the CPU utilization ratio of all nodes; c_i indicates the CPU utilization of the node i; and m indicates the number of all nodes. Refer to Eq. (2) for the calculation method of CPU utilization, which is stated as

$$c_i = 100 - \frac{100(t_1 - t_2)}{T_1 - T_2}. \tag{2}$$

In Eq. (2), t_1 and t_2 represent the CPU idle time of two time nodes, and T_1 and T_2 represent the total CPU time of the node at two times, respectively. Similarly, the expressions of memory consumption and disk Input/Output (IO) usage can be obtained. Then, we integrate the three pieces of information about the resources to express the resource consumption of the whole node. The specific relation is shown by Eq. (3)

$$W = \vartheta_1 \alpha + \vartheta_2 \beta + \vartheta_3 \gamma. \tag{3}$$

In Eq. (3), W represents the resource consumption score of the node; β represents the memory utilization ratio of the node in the memory utilization ratio of all nodes; γ represents the proportion of node IO utilization in all node IO utilization; and ϑ_1, ϑ_2, and ϑ_3 represent the scale coefficients of different computing resources. The coefficient represents the proportion coefficient of different computing resources. For a biology teaching task, there may be a different emphasis on resource consumption. For example, computing intensive task applications mainly consider the CPU in computing resources, so it is necessary to appropriately increase the proportion of ϑ_1 in the coefficient. For data intensive application tasks, CPU calculation is not very important. Instead, high-throughput data involves the high utilization rate of disk IO, and a large amount of data needs to be saved in memory for calculation. Therefore, it is necessary to focus on improving the proportion of ϑ_2 and ϑ_3 in the coefficients. This design can effectively target different computing types of teaching tasks, and developers can independently set different computing resource weights to meet the needs. After quantifying the resource consumption of all participating work nodes according to the equation, it is necessary to effectively identify overloaded nodes and high-potential idle nodes according to the finally calculated score. If the score of a machine node is too high compared with other machine nodes, it indicates that the resource consumption of each node in the cluster is uneven. The node with too high a score is likely to become an overloaded node and compete for resources, which is likely to mean that the data processing capacity or computing capacity of the node is saturated.[12] If a machine node has a low score compared with other nodes, it means that its current resource consumption is less. The node can load and bear more data computation, making it a high-potential idle node. Therefore, we only need to obtain the highest and lowest scores of all nodes, and then check whether the gap between the maximum score and the minimum score in the node exceeds a certain threshold through Eq. (4), which is given by

$$\chi = \frac{W_{\max}}{W_{\min}}. \tag{4}$$

In Eq. (4), χ represents the score gap, and $W_{\max}$ and $W_{\min}$ represent the maximum and minimum scores, respectively. If the gap exceeds the preset threshold, it means that the cluster is likely to have the problem of computing resource tilt, and the data partition strategy needs to be adjusted to alleviate the current situation of computing resource tilt.[13] At the same time, the node at the highest point may also become an overloaded node that needs its resource bottleneck to be solved urgently. Other nodes can share the amount of calculation data of these overloaded nodes more or less equally.

2.3 *Design Adaptive Dynamic Partition Strategy*

If the proportion of data allocated by the node is too high, the amount of data processed by parallel instances on the node will also increase, which will lead to higher resource utilization of the node, and more likely to compete for resources and throughput saturation. However, if the proportion of data allocated by nodes is too low, on the one hand, the data pressure will be handed over to other nodes, aggravating the load of other nodes.[14] On the other hand, the amount of data processed by the parallel instance on the node will be too small, resulting in a sharp reduction in the parallelism of the overall task, making the parallel effect worse.[15] Therefore, it is necessary to reasonably use the data partition strategy to allocate the amount of data of each node. The data partition strategy method is shown in Fig. 3.

As shown in Fig. 3, the partition policy consists of three components. The data partition component is mainly responsible for maintaining, saving, and adjusting the data allocation proportion of downstream parallel instances. It is the most critical part in the adaptive dynamic partition strategy. The component contains a dynamic greedy algorithm to adjust the data distribution ratio in real time according to the scores of the nodes where all downstream parallel instances are located. The real-time monitoring component is mainly responsible for real-time monitoring of the throughput changes of downstream parallel instances after collecting dynamic data partition strategy, and reporting to the feedback component. The feedback component is mainly responsible for evaluating the effectiveness of the dynamic data partition strategy adjustment. We judge whether to permanently apply the data allocation proportion adjustment according to the throughput change of downstream parallel instances transmitted by the real-time monitoring component.[16] Because nodes and parallel instances have a one-to-many relationship, we can obtain the information of all parallel instances downstream of the data partition and their nodes. The scores of these participating nodes are included in the scores of all nodes, so we can directly obtain the scores of these participating nodes. Since all parallel instances in the same node use all computing resources on the node, these parallel instances share all resources of the node in theory. As tasks are deployed on nodes, they are treated uniformly by nodes. Therefore, it can be seen that the data allocation proportion of these parallel instances on the same node should be the same. At the same time, we summarize and sum the data allocation proportion of these parallel instances, that is, the data allocation proportion of the node. First, we compare the scores of downstream nodes to obtain the node with the highest score. This node is likely to be an overloaded node. It is necessary to reduce the data allocation proportion of parallel instances deployed on it in the data partition. Use Eq. (5) to determine the current adjustment step, which is given as

$$h = \begin{cases} H/a, a \le 5 \\ H/5, a > 5 \end{cases}. \tag{5}$$

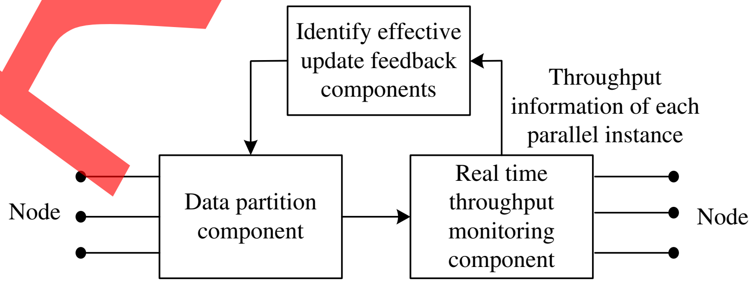
Identify effective update feedback components
Throughput information of each parallel instance
Node
Data partition component
Real time throughput monitoring component
Node

Fig. 3 Dynamic data partition policy component diagram.

In Eq. (5), h represents the adjustment step; H represents the total step size; and a indicates the number of times the data partition policy is continuously tuned. After more than five optimizations, the step size has reached the minimum, the convergence speed decreases and the stability increases. The variable step length can not only ensure the convergence to a smaller value, but also improve the efficiency. Next, reduction of the data allocation proportion of suspected overloaded nodes is needed according to the current step size, as well as updating the allocation proportion of other nodes to keep the sum of the total proportion unchanged. The equation for synchronously updating the data allocation proportion is given by

$$\begin{cases} W' = \sum_{i=1}^{m} W \\ \varphi = \varphi_0 + h\eta W_{m-1}/W' \end{cases}. \tag{6}$$

In Eq. (6), W' represents the sum of all scores excluding suspected overloaded nodes; φ_0 and φ represent the data distribution ratio before and after the update, respectively; η represents the proportion of suspected overloaded nodes; and W_{m-1} indicates the score of node $m-1$ arranged from low to high. After adjusting the data allocation ratio of all downstream participating nodes, including overloaded nodes, it is necessary to return the new data allocation ratio to the parallel instances deployed on these nodes, because the final data will be really partitioned on these parallel instances.[17] Since there may be multiple parallel instances in the same node and these parallel tasks use the computing resources of the node evenly, these parallel instances share the data distribution proportion equally.

2.4 *Visual Resource Flow Allocation Model Based on Ant Colony Algorithm*

According to the dynamic partition strategy of engineering training resources, a process visual allocation model is established by using ant colony scheduling algorithm. Ant colony scheduling algorithm is an optimization algorithm for the big data visualization platform designed in this paper, so the following settings are made before algorithm modeling: the whole cluster is built on cheap machines, the hardware configuration of each node is not exactly the same, and the performance is different. The resources owned by each node are known, and the execution time of the task can be estimated by the node resources and task size. Each task is relatively independent. After the task is submitted, the execution process cannot be interrupted until the end of task execution. When the cluster executes multiple tasks, the resources required by the tasks are different, and the resource load is also different. The goal of ant colony scheduling algorithm is to find the optimal solution of training resource allocation and scheduling, and the index to determine whether the optimal solution is the overall task completion time.[18] The overall task completion time should be the latest time for all nodes to complete the task, and the shortest time is the required optimal solution. In the process of ant colony search, pheromone concentration affects the transfer probability, so the initial pheromone concentration also affects the convergence speed of the whole algorithm.[19] The higher the pheromone concentration, the greater the probability that the ant colony will assign tasks to the node, so the stronger the node performance, the higher the pheromone concentration on the node should be. The performance of the node is described by quads (CPU, bandwidth, memory, disk speed), so the initial pheromone concentration is the average of various performance parameters. When the expectation is higher, the probability of task allocation to the node is greater, so the node takes more time to execute the task. The expected function is constructed with the equation

$$\lambda_i(\tau) = \begin{cases} \tau_0/\mu, \tau > 0 \\ 1, \tau = 0 \end{cases}. \tag{7}$$

In Eq. (7), $\lambda(\tau)$ represents the expected function; τ represents time; τ_0 represents the average running time of the current optimal solution obtained in the last round of search on each node; and μ indicates the time when the task has been completed. Ant colony scheduling algorithm accelerates the convergence speed of ant colony by secreting pheromone on nodes. Therefore, when a task is assigned, the pheromone concentration needs to be updated. The updated equation of pheromone is given by

$$\kappa(\tau+1) = (1-\theta)\kappa(\tau) + \Delta\kappa(\tau). \tag{8}$$

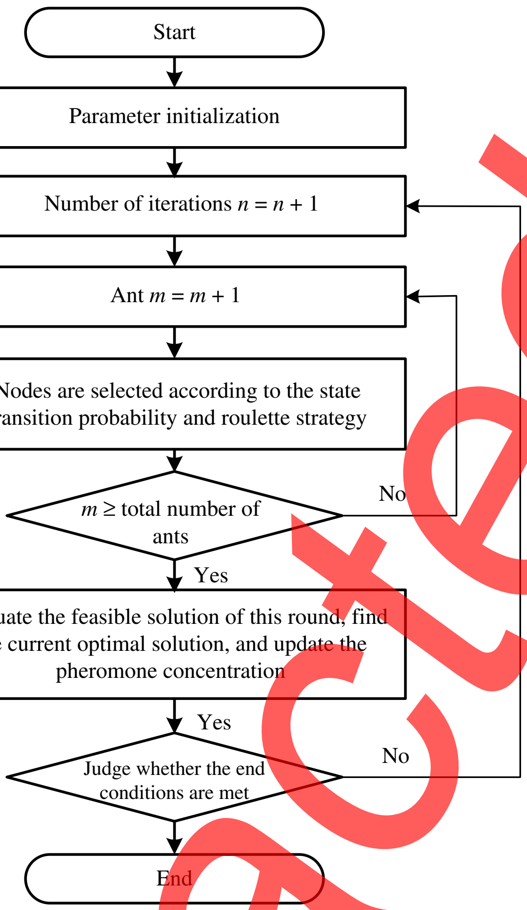
Start
Parameter initialization
Number of iterations $n = n + 1$
Ant $m = m + 1$
Nodes are selected according to the state transition probability and roulette strategy
$m \geq$ total number of ants
No
Yes
Evaluate the feasible solution of this round, find the current optimal solution, and update the pheromone concentration
Yes
Judge whether the end conditions are met
No
End

Fig. 4 Allocation process of training resources.

In Eq. (8), $\kappa(\tau)$ and $\kappa(\tau + 1)$ represent the initial and increased values of pheromone concentration, respectively; $\Delta\kappa(\tau)$ is the pheromone concentration secreted by ants on nodes; and θ represents volatilization factor. To prevent the pheromone concentration from increasing and drowning the influence of the heuristic function, the pheromone concentration will be reduced due to volatilization before updating the pheromone concentration every time.[20] The greater the volatilization factor, the lower the concentration of residual pheromone. The ant colony will allocate tasks according to the state transition probability.[21] However, if the task is only assigned to the node with the greatest probability each time, the whole algorithm may converge to the local optimal solution.[22] Therefore, a strategy should be set to ensure that each node may be assigned to a task, and the probability of being assigned to a task is related. The fitness of each node is calculated by roulette strategy, and the ratio of the probability of each node being selected to the total fitness is obtained. A random number between [0, 1] is generated by the program. When it falls into a certain interval, it selects to submit the task to the node corresponding to the interval. The task can be submitted to any node with probability through roulette strategy. The allocation process of training resources is shown in Fig. 4.

The basic ant colony algorithm is re-modeled in the cluster environment to optimize the training resource allocation process. So far, the design of process visual allocation method of engineering training resources in biology teaching process has been completed.

3 Experiment

3.1 *Experimental Preparation*

This paper presents a flow-based visual allocation method of engineering training resources in biology teaching process. The method is tested below. Multiple distributed architectures are used

Table 2 Node environment configuration.

Environment configuration	Node 1	Node 2	Node 3	Node 4	Node 5
Java	jdkl. 8	jdkl. 8	jdkl. 8	jdkl. 8	jdkl. 8
Remote	ssh	ssh	ssh	ssh	ssh
HDFS	NN, DN	NN, DN	DN	DN	DN
YARN	RM, NM	RM, NM	NM	NM	NM
Hbase	Master	Master	Region	Region	Region

to build the test environment, and there are five nodes in the cluster. The operating system of each node is CentOS. Different environments need to be configured according to the different roles of each node. The specific node environment configuration is shown in Table 2.

The test cluster is composed of five nodes. During task execution, the stored content needs to be read between nodes, and passwords are usually required for access between nodes, which will lead to a large number of manual operations. Configuring ssh through public and private key authentication avoids the password problem. In addition to the big data node cluster, there is also a web server in the test environment. The server works in the data service layer of the visualization platform system architecture, undertakes the front-end request and data response, and provides services through the HTTP interface. To ensure the rationality of the test results, the data set needs to simulate the randomness of the actual biology teaching scene. The content of the test data set comes from the engineering training resources of biology teaching in a school. The word text is continuously randomly selected through the program and stored in a specified text. The size of each text is about 10 MB, so that the frequency of each word in the test data set is different.

3.2 *Experimental Result*

To prove the superiority of this method, it is compared with the resource allocation methods based on cloud computing and mobile edge computing. It is done mainly through the job completion time index to test the application effect of different allocation methods. The experiment is tested in homogeneous cluster and heterogeneous cluster environments, respectively.[23,24] Homogeneous cluster means that all nodes in the cluster have the same performance, including CPU, memory, disk, bandwidth, etc. To ensure the consistency of the cluster, five completely consistent nodes are cloned through the virtual machine and configured as a cluster in the isomorphic test environment. Use the generated test data set to test multiple groups of data, and the test results are shown in Table 3. Heterogeneous cluster refers to the inconsistent performance of each node in the cluster, which is also the most common cluster construction mode in actual production. To ensure the heterogeneity of the cluster, five nodes with different performance are set up through the virtual machine to configure the cluster in the heterogeneous test environment.

Table 3 Comparison of allocation methods under homogeneous clusters.

Task volume (MB)	Completion time of this method (s)	Method completion time based on cloud computing (s)	Method completion time based on moving edge calculation (s)
100	7.2	15.7	16.9
200	11.3	21.6	22.7
300	25.6	47.8	48.5
400	40.5	72.5	72.4
500	89.7	118.3	120.6

Table 4 Comparison of allocation methods under heterogeneous clusters.

Task volume (MB)	Completion time of this method (s)	Method completion time based on cloud computing (s)	Method completion time based on moving edge calculation (s)
100	9.2	18.2	19.3
200	20.4	26.1	28.4
300	41.8	63.6	64.1
400	58.6	90.7	92.6
500	99.4	140.8	142.5

The data set is consistent with that in homogeneous clusters, and the test results are shown in Table 4.

According to the results in Tables 3 and 4, the completion time of the three allocation methods increases with the gradual increase of the amount of tasks. And the completion time in heterogeneous clusters is greater than that of homogeneous clusters. Whether in homogeneous or heterogeneous clusters, the effect of the allocation method proposed in this paper is better than the resource allocation methods based on cloud computing and mobile edge computing. Taking the task volume of 500 MB as an example, in the homogeneous cluster environment, the completion time of this method is 89.7 s, which is 28.6 and 30.9 s shorter than the methods based on cloud computing and mobile edge computing. In heterogeneous cluster environment, the completion time of this method is 99.4 s, which is 41.4 and 43.1 s shorter than the methods based on cloud computing and mobile edge computing. Therefore, this method significantly improves the efficiency of task scheduling, shortens the computing time of information processing, and optimizes the performance of flow-based visual allocation of training resources.

4 Conclusion

This paper studies the flow visualization allocation method of engineering training resources in biology teaching process, mainly through the design of a good data partition strategy to alleviate the inclination of computing resources in the cluster, so that the scheduling task can run efficiently in the distributed stream processing environment. Through experiments, it is verified that the design method has less average completion time and improves the efficiency of resource allocation. However, in the research process of this paper, some problems are still found. This paper marks the node with the highest score as a suspected overloaded node, and then reduces the data distribution proportion of this node. However, in a few cases, nodes with low scores will overload nodes instead. The root cause is that the current scoring method is relatively simple. Therefore, in the next step, we can design more elaborate methods and related equations to quantify the resource consumption of each node in combination with other factors reflecting the node overload or resource competition. The adaptive dynamic data allocation proportion adjustment method proposed in this paper is based on the greedy idea. The reduction and increase of data allocation proportion are not fine enough. Although it is currently overloaded, the throughput decreases due to excessive proportion allocation after adjustment, resulting in the feedback node abandoning this adjustment. Therefore, in future, we can design a data allocation strategy with more detailed allocation proportion and fewer adjustment steps. The artificial intelligence and machine learning strategy can be used in future research work.

Acknowledgments

We acknowledge the support by the New Engineering Research Project of Nanyang Institute of Technology in 2020 under Grant No. NIT2020XGKJY-03.

References

1. X. L. Wang et al., “The network teaching platform construction of medical microbiology,” *Appl. Mech. Mater.* **343**, 141–144 (2013).
2. H. Wang et al., “Construction of practical education system for innovative applied talents cultivation under the industry-education integration,” in *Proc. 5th Int. Conf. Front. Educ. Technol.*, pp. 68–72 (2019).
3. Y. Feng et al., “Construction of five-level practical teaching system for bioengineering under emerging engineering education background,” *Sheng Wu Gong Cheng Xue Bao* **36**(5), 1012–1016 (2020).
4. D. Kesner and D. Ventura, “A resource aware semantics for a focused intuitionistic calculus,” *Math. Struct. Comput. Sci.* **29**(1), 93–126 (2019).
5. M. Shanmugam et al., “A wearable embedded device for chronic low back patients to track lumbar spine position,” *Biomed. Res.* **1**, S118–S123 (2017).
6. H. Raei et al., “SeCARA: a security and cost-aware resource allocation method for mobile cloudlet systems,” *Ad Hoc Networks* **86**, 103–118 (2019).
7. S. Shiqing et al., “A resource-aware service function chain coordinated composition and mapping algorithm under 5G network,” *J. Xi’an Jiaotong Univ.* **54**(8), 153–162 (2020).
8. S. Maheswaran et al., “Identification of artificially ripened fruits using smart phones,” in *Int. Conf. Intell. Comput. and Control*, pp. 178–183 (2017).
9. P. Vamvakas et al., “Risk-aware resource management in public safety networks,” *Sensors* **19**(18), 3853 (2019).
10. R. Mo et al., “Multi-objective cross-layer resource scheduling for internet of things in edge-cloud computing,” in *IEEE 13th Int. Conf. Cloud Comput.*, IEEE, pp. 345–352 (2020).
11. S. A. AlQahtani, “An efficient resource allocation to improve QoS of 5G slicing networks using general processor sharing-based scheduling algorithm,” *Int. J. Commun. Syst.* **33**(4), e4250 (2020).
12. D. Bankov et al., “Resource allocation for machine-type communication of energy-harvesting devices in Wi-Fi HaLow networks,” *Sensors* **20**(9), 2449 (2020).
13. M. Y. Abdelsadek et al., “Cross-layer resource allocation for critical MTC coexistent with human-type communications in LTE: a two-sided matching approach,” *IET Commun.* **14**(18), 3223–3230 (2020).
14. V. Chudasama and M. Bhavsar, “A dynamic prediction for elastic resource allocation in hybrid cloud environment,” *Scalable Comput. Pract. Exp.* **21**(4), 661–672 (2020).
15. O. Beaude et al., “A privacy-preserving method to optimize distributed resource allocation,” *SIAM J. Optim.* **30**(3), 2303–2336 (2020).
16. S. Ghosh and D. De, “Weighted majority cooperative game based dynamic small cell clustering and resource allocation for 5G green mobile network,” *Wireless Pers. Commun.* **111**(3), 1391–1411 (2020).
17. A. A. Khan et al., “Optimizing downlink resource allocation in multiuser mimo networks via fractional programming and the Hungarian algorithm,” *IEEE Trans. Wireless Commun.* **19**(8), 5162–5175 (2020).
18. A. I. Saleh et al., “Ant colony prediction by using sectorized diurnal mobility model for handover management in PCS networks,” *Wireless Networks* **25**(2), 765–775 (2019).
19. K. Majbouri Yazdi et al., “Prediction optimization of diffusion paths in social networks using integration of ant colony and densest subgraph algorithms,” *J. High Speed Networks* **26**(2), 141–153 (2020).
20. R. Ramachandranpillai and M. Arock, “Spiking neural P ant optimisation: a novel approach for ant colony optimisation,” *Electron. Lett.* **56**(24), 1320–1322 (2020).
21. D. P. Mahato et al., “On scheduling transaction in grid computing using cuckoo search-ant colony optimization considering load,” *Cluster Comput.* **23**(2), 1483–1504 (2020).
22. J. Wang et al., “Literature-based learning and experimental design model in molecular biology teaching for medical students at Tongji University,” *Biochem. Mol. Biol. Educ.* **49**(2), 189–197 (2021).

23. R. Zhang and A. C. S. Chung, “MedQ: lossless ultra-low-bit neural network quantization for medical image segmentation,” *Med. Image Anal.* **73**, 102200 (2021).
24. N. S. Punn and S. Agarwal, “Modality specific U-net variants for biomedical image segmentation: a survey,” arXiv:2107.04537 (2021).

Biographies of the authors are not available.