Cloud service workflow scheduling algorithm based on priority rules

Ying Zhao* and Bin Hu

Yunnan Electric Power Dispatching Control Center,
Kunming, Yunnan 650011, China
Email: yingzhao@mls.sinanet.com
Email: YunDWa@outlook.com
*Corresponding author

Zongjie Zhang and Rongkui Zhang

Yunnan Yundian Tongfang Technology Co., Ltd.,
Kunming, Yunnan 650017, China
Email: yixiao3@mls.sinanet.com
Email: gelangl@36haojie.com

Abstract: In order to solve the problems of high energy consumption and time cost of cloud service workflow task scheduling and poor resource utilisation of cloud platform, a cloud service workflow scheduling algorithm based on priority rules is proposed. Firstly, according to the characteristics of cloud service workflow architecture, the workflow instances to be processed are classified by MCUD algorithm. Secondly, according to the classification results, genetic algorithm is used to generate the scheduling strategy to meet the needs of users. Finally, priority rules, namely cloud task priority heuristic rules, are introduced to determine the workflow task order and update it to reduce the completion time and cost of workflow scheduling. The results show that this method can shorten the execution time, reduce the energy consumption cost, and improve the resource utilisation effect of cloud platform, which verifies the feasibility of this method and the effectiveness of the algorithm.

Keywords: MCUD algorithm; genetic algorithm; priority rules; workflow; objective function.

Reference to this paper should be made as follows: Zhao, Y., Hu, B., Zhang, Z. and Zhang, R. (2022) 'Cloud service workflow scheduling algorithm based on priority rules', *Int. J. Internet Manufacturing and Services*, Vol. 8, No. 3, pp.254–266.

Biographical notes: Ying Zhao received her Master's degree in Control Theory and Control Engineering from North China Electric Power University in 2009. She is currently a senior engineer in Power Dispatching Control Center of Yunnan Power Grid Corporation. Her research interests include power dispatching automation, etc.

Bin Hu received his Bachelor's degree in Power System and Automation from North China Electric Power University in 2005. Currently, he is a senior engineer in Power Dispatching Control Center of Yunnan Power Grid Corporation. His research interests include power dispatching automation, etc.

Zongjie Zhang received his Bachelor's degree in Information Management and Information System from Qujing Normal University in 2009. Currently, he is the technical manager of Tongfang Power Grid Operation Division of Yunnan Yundian. His main research direction is power grid scheduling informatisation.

Rongkui Zhang received his Bachelor's degree in Electronic Information Engineering from Yunnan Minzu University in 2011. Currently, he is a senior development engineer in Tongfang Power Grid Operation Division of Yunnan Yundian. His main research direction is power grid scheduling informatisation.

1 Introduction

In the context of the continuous development of cloud computing technology, with the help of virtualisation technology, different resources can be transmitted to the user end, providing users with services such as resource calculation and resource storage, allowing users to enjoy various application service types (Sun et al., 2020). A variety of cloud computing products have subsequently been produced. While providing services to users, their security and reliability will also be tested to a certain extent (Rezaeian et al., 2019). Due to the gradual complexity of various business processes in cloud services, various problems will be faced during the execution of business processes. Among them, the more common problems are mainly reflected in cost and time constraints (Nik et al., 2020). Although the existing methods have studied related issues, the research is still not enough to effectively solve the related issues. Therefore, how to effectively schedule the workflow in cloud services has become an urgent problem to be solved at present (Ijaz et al., 2021).

Zhang et al. (2020) proposed a workflow energy efficient scheduling algorithm based on cost constraints of heterogeneous cloud computing. Firstly, a task priority criterion was established and workflow scheduling was divided into three stages, namely task cost allocation, optimal execution task determination and task efficient scheduling. Then, constraint conditions are established to constrain the above three stages. The experimental results show that the algorithm can effectively reduce the energy consumed in the task execution, but due to the many steps to be processed, the execution time is long and the time cost is high. Bao et al. (2019) proposed a multi-objective cloud workflow scheduling algorithm based on grid distribution variance, particle encoding of the workflow, and discretisation; the Pareto optimal solution set method is used to map the workflow to In the grid coordinate system, the workflow in the coordinate system is optimised and adjusted; the difference particle self-learning strategy is used to finalise the multi-objective cloud workflow. The experimental results show that the algorithm can adapt to the general workflow scheduling requirements, but in multi-objective scheduling, it will consume more energy, and there is a problem of high energy consumption cost. Wang (2019) proposed a cloud workflow scheduling method based on an improved multi-objective evolutionary algorithm. First, a cloud workflow scheduling model was constructed, combined with a real number coding mechanism, to encode tasks in the model to avoid workflow crossover problem: Use the a multiobjective evolutionary algorithm based on decomposition (MOEA/D) algorithm to design a local search strategy, and perform scheduling processing for multi-objective workflows. The

experimental results show that the stability of this method is good, but the utilisation rate of cloud platform resources after scheduling is not high.

In order to solve the problems of high time cost, high energy consumption cost and low utilisation rate of cloud platform resources in traditional methods, this paper proposes a cloud service workflow scheduling algorithm based on priority rules. The following is the overall research framework of the method in this article.

First, establish a cloud service workflow system structure, use the minimum total cost under user designated total deadline (MCUD) algorithm to classify work tasks in the cloud service workflow system, resolve conflicts between similar work tasks, and reduce the time cost of workflow scheduling.

Secondly, based on the results of workflow instance classification, genetic algorithms are used to generate scheduling strategies that meet user needs, and cloud service workflow scheduling problems are solved through the steps of population initialisation, chromosome selection, crossover, and mutation.

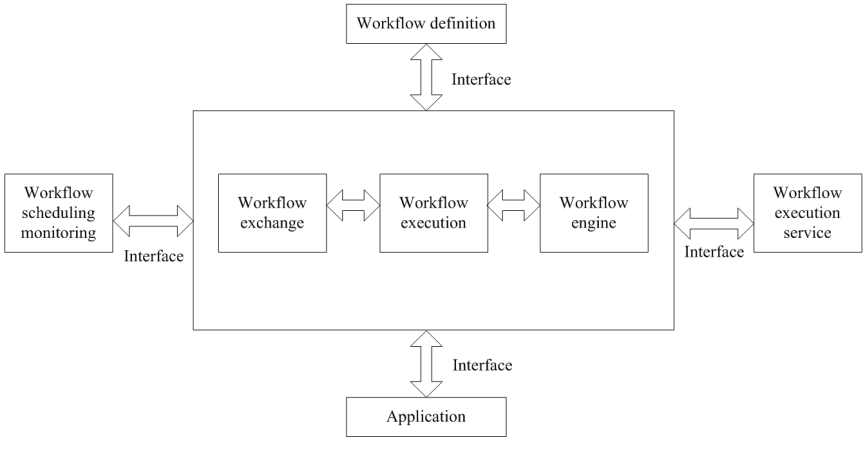
Finally, considering the time cost and execution cost of cloud service workflow scheduling, the objective function and constraint conditions are established: for the objective function, the cloud task priority heuristic rule is used to determine the task scheduling order, and the stage and priority are constructed through feasible solutions. In the rule improvement stage, cloud service workflow scheduling is realised.

2 Cloud service workflow preprocessing

2.1 Cloud service workflow architecture

Generally, the cloud service workflow system is composed of cloud services and workflow systems. Among them, the cloud service workflow can also be called PssA service, which is mainly used to execute the cloud service workflow, which can be regarded as a software application service. In the cloud service workflow system, users can also set up their own workflow through visual modelling tools (Karpagam et al., 2020). Starting from the interface and common components of cloud service workflow, this paper gives the architecture of cloud service workflow as shown in Figure 1.

Figure 1 Schematic diagram of the architecture of cloud service workflow



According to Figure 1, when a workflow is created, a corresponding workflow specification will follow. The specification specifically includes task information, task execution sequence, and constraints in task execution. Under the constraint conditions, a large number of workflows will be uniformly classified according to their types, and then they will be scheduled to form a workflow scheduling system that meets service requirements (Aziza and Krichen, 2020).

2.2 Workflow classification based on MCUD algorithm

In order to reduce the time cost of workflow scheduling, the MCUD algorithm is used to classify the work tasks in the cloud service workflow system (Chakravarthi and Shyamala, 2020). When the traditional method is used to classify work tasks, the competition between tasks is not considered, which is easy to cause resource conflicts. However, the MCUD algorithm can effectively resolve conflicts between similar work tasks through performance evaluation. Calculate the minimum processing time of workflow scheduling in task classification, effectively reducing time cost (Kintsakis et al., 2019). The specific steps of using MCUD algorithm to classify workflow are as follows:

In workflow scheduling, assuming that the minimum processing time of task T_i is T_i(min), the total processing time of tasks in the same type of workflow instance is T_{all}, and the total number of scheduled tasks is denoted by N, then the processing time of a certain task i for:

$$T_i = \frac{\mu(Y_t + \theta_i)}{T_{all}} \tag{1}$$

In the formula, Y_t represents the resources that can perform the task; θ_i represents the execution cost of the available resources; μ represents the longest execution time.

A certain type of workflow instance is arbitrarily divided into m values to form a set $I = \{i_1, i_2, \ldots, i_m\}$, and the task scheduling deadline is solved for the instances in the set. The specific formula is:

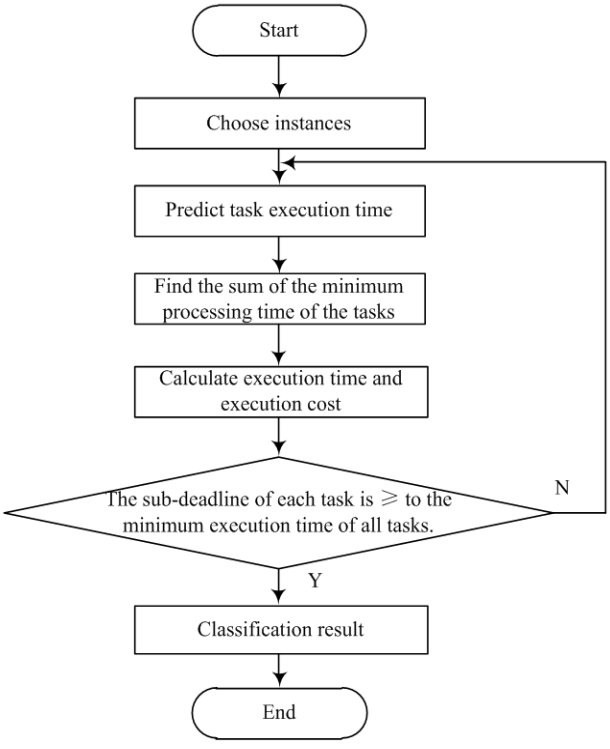
$$T_d^m = \Delta t + \frac{\Delta I \cdot \Delta t^2}{2} \tag{2}$$

In the formula, ΔI represents the synchronisation task; Δt^2 represents a random value. Divide them according to the task scheduling deadline values of different workflow instances to achieve workflow classification. Figure 2 shows the specific flow of workflow classification.

In workflow classification, it is necessary to ensure that the sub-deadline assigned to each task must be greater than or equal to its minimum execution time on all available service resources, and based on this standard, workflow classification is implemented.

Compared with traditional methods, the MCUD algorithm meets the deadline as a condition and can effectively resolve the competition between workflows. It can not only reduce the time for workflow scheduling, but also improve the utilisation of cloud service platform resources.

Figure 2 Workflow classification flowchart



3 Cloud service workflow scheduling algorithm based on priority rules

3.1 Solution to the cloud service workflow scheduling problem based on genetic algorithm

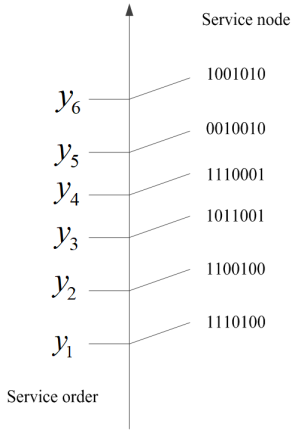
Based on the classification results of workflow instances, a genetic algorithm is used to generate scheduling policies that meet user needs, and to solve the cloud service workflow scheduling problem. Genetic algorithm is a kind of search algorithm, which is an imitation of the natural adaptation process. At present, with the continuous deepening of research, it has achieved more eye-catching application effects, and has gradually become one of the effective algorithms to solve the scheduling problem (Deng et al., 2019). Genetic algorithm has strong local random search ability. When the genetic algorithm has approached the neighbourhood of the optimal solution through the crossover operator, the local random search ability of mutation operator can accelerate the convergence to the optimal solution. Therefore, genetic algorithm is selected as the basis for solving the cloud service workflow scheduling problem.

Suppose that η_i is used to represent workflow variables, and A represents a set of cloud service nodes. The set A contains all workflow variables, and these variables are also oriented to all chromosomes. Sort the chromosomes, obtain the chromosome sequence number, and express the sequence as an individual (Peng et al., 2021). Set a fitness function to reflect the execution time cost of the workflow instance:

$$F(x) = \gamma |a_t - a_z| \cdot \overline{a} \tag{3}$$

In the formula, a_t represents the individual fitness; a_z represents the individual fitness differences; $\overline{a}$ represents the average fitness of the population; γ represents the fitness of the optimal individual. The larger the fitness value, the smaller the time cost of workflow scheduling, and vice versa, the larger the time cost (Alaei et al., 2020). Figure 3 shows an example of a chromosome.

Figure 3 Examples of chromosomes



The following describes the process of solving the cloud service workflow scheduling problem based on genetic algorithm in detail:

1 Population initialisation: first set the population size, then randomly generate a set of workflow instances within this range, and assign each random value to the corresponding service variable:

$$R = \sum_{x=1}^{M} \left(w_x + \mu_x \right)^2 \tag{4}$$

In the formula, w_x represents the service set of the workflow; μ_x represents the dependency between cloud services; M represents the population number; x represents the service variable.

2 Chromosome selection: form a new population by selecting appropriate operators, denoted by K, and its fitness function is $K(x)$, and then use random sampling to update the population to obtain the fitness function population:

$$K' = \sum_{i=1}^{N} \left(C_{i,t} \times C_{\max} \right) \tag{5}$$

In the formula, $C_{i,t}$ represents the ratio between individual fitness and average fitness; $C_{\max}$ represents the local maximum.

3 Chromosome crossover: In order to avoid falling into the local optimal problem in solving the cloud service workflow scheduling problem, a single point crossover operator is used to define a random function (Jiang et al., 2015):

$$D(t) = \prod_{i=1}^{n} \left(1 - \frac{T_a^i}{T_a^i - T_b^i}\right) \tag{6}$$

In the formula, T_a^i and T_b^i both represent the crossover probability. Then the workflow example after crossover is:

$$P_f = K_u \left(P_i + P_j\right) \tag{7}$$

In the formula, P_i and P_j both represent the global space and local space of the workflow search space; K_u represents the search space of the workflow.

4 Chromosome mutation: In order to avoid the problem of partial stagnation of the population, a mutation operator is used to replace the workflow example:

$$V_t = \sum_{t=1}^{N} p_t / P \tag{8}$$

In the formula, p_t represents the probability of mutation. Through the above steps, the cloud service workflow scheduling problem is solved, and the optimal solution of the problem is obtained.

3.2 Implementation of cloud service workflow scheduling based on priority rules

In order to realise the reasonable scheduling of cloud service workflow, based on the solution results of genetic algorithm, the multi-objective optimisation of workflow scheduling is solved. Considering the time cost and execution cost of cloud service workflow scheduling, taking the minimisation of execution cost and time cost as the objective function, in order to minimise the execution cost under time constraints, the objective function is established:

$$O_r = \min \sum_{i=1}^{T} f_{it} c_{it} \tag{9}$$

In the formula, f_{it} represents the time cost; c_{it} represents the execution cost of the scheduling task.

In multi-objective function processing, each objective is constrained by decision variables, and the processing of one objective requires the other objectives as the cost. Therefore, the corresponding constraints are:

$$\begin{cases} \sum_{i=1}^{T} f_{it} = 1 \\ \sum_{i=1}^{T} c_{it} = 1 \end{cases} \tag{10}$$

Aiming at the objective function established by formula (9), this paper uses the cloud task priority heuristic rule to determine the task scheduling sequence, which is specifically divided into a feasible solution construction phase and a priority rule improvement phase (Ranjan et al., 2020):

1 Feasible solution construction stage

The workflow is defined as $S(E, L)$, a cloud service instance is selected arbitrarily in the workflow according to the priority rules, and a solution of the workflow is obtained, denoted by s_r. Choose the cloud service level, select the appropriate service level according to the actual situation of the workflow, and then calculate the time cost of obtaining the workflow solution:

$$C(t) = \int_{-\infty}^{\infty} x(\theta) C(\theta) dt \tag{11}$$

In the formula, $x(\theta)$ represents the completion time of the calculation solution; $C(\theta)$ represents the cost of the calculation solution. If the two can meet the constraints of formula (10), it indicates that it is a feasible solution.

2 Priority rule improvement stage

Traditional methods usually use the optimal rules to schedule the workflow. Although this method can obtain the appropriate scheduling strategy, it usually consumes a lot of time and cost, and will affect the utilisation effect of cloud platform resources. Therefore, according to the feasible solution constructed in the above steps, priority rules are used to improve the workflow scheduling to reduce the scheduling cost.

Firstly, set the priority and update the workflow scheduling task. If some workflows exit the update due to the reduction of relaxation time in the update process, they do not need to be improved and can be eliminated directly, which is conducive to reducing the burden of the algorithm and reducing the scheduling cost. However, if more work flows out of the improvement step, it will affect the time coupling strength and increase the scheduling cost. Therefore, improve the priority rules to obtain the optimal rules of time coupling strength:

$$T\left(q^{-1}\right) = \left[u(k) + \xi(k) \times C(t)\right] \tag{12}$$

In the formula, $u(k)$ represents the time window of all workflow scheduling tasks; $\xi(k)$ represents the task with the highest priority.

Different feasible solutions are assigned corresponding priorities respectively, and then the highest priority workflow scheduling task selects the update service to form a new feasible solution, and then the priority update operation is repeated until the feasible solutions with the optimal time cost and cost is obtained.

To sum up, firstly, according to the features of the cloud service workflow architecture, the MCUD algorithm is used to classify the workflow instances to be processed to reduce the complexity of workflow task scheduling; secondly, according to the classification results, the genetic algorithm is used to generate schedules that meet the needs of users. The strategy is to minimise the execution cost of workflow task scheduling within a given processing time, and to minimise the execution time of task scheduling under the premise of limiting the execution cost; finally, a priority rule is introduced, that is, the cloud task priority order heuristic Rules determine the order of workflow tasks and update them to reduce workflow scheduling completion time and cost.

4 Experimental research

In order to verify the effectiveness and application value of the proposed cloud service workflow scheduling algorithm based on priority rules, experimental research is carried out. In order to reflect the effectiveness of the proposed method in the experiment, Zhang et al.'s (2020) method and Bao et al.'s (2019) method are compared with the proposed method as a comparison method, and the application effects of different methods are analysed.

4.1 Experimental scheme design

4.1.1 Experimental environment

This test was carried out in the CloudSim cloud computing environment, with a network bandwidth of 80 Mbps and a memory of 16GB. Since CloudSim usually only simulates a single workflow task, in order to adapt to the needs of multi-task scheduling, its base class is extended, and after the extension, it can simulate tasks with complex relationships.

4.1.2 Experimental parameter setting

In order to verify the effectiveness of the proposed method more comprehensively, in the simulation experiment, the scale of the workflow will be set, specifically divided into small-scale, medium-scale and large-scale, the corresponding parameters are N = 100, N = 500 and N = 1,000. In the experiment, the CloudSim platform provides a large number of workflow examples, a total of 150 examples, to support the orderly progress of the experiment.

4.1.3 Experimental indicators

1 Time cost: this indicator can reflect the scheduling efficiency of different scheduling methods. The lower the time cost, the higher the efficiency of workflow scheduling;

2 Energy consumption cost: this indicator refers to the memory and other costs consumed in workflow scheduling. The lower the energy consumption cost, the higher the application value;

3 Platform resource utilisation effect: it is mainly verified by cloud platform resource utilisation rate, memory utilisation rate and cloud platform resource reliability.

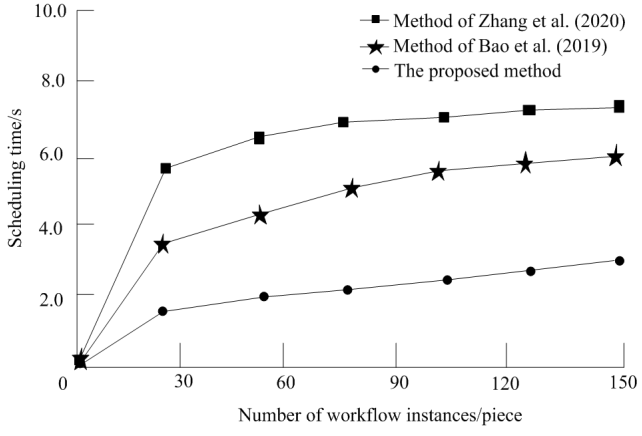
4 Comparison method: take the method of Zhang et al. (2020) and the method of Bao et al. (2019) as the comparison method to compare with the proposed method.

4.2 Analysis of experimental results

Experiment 1

Taking time cost as an experimental indicator, comparing the scheduling effect of the method of Zhang et al. (2020) and the method of Bao et al. (2019) with the proposed method, the results are shown in Figure 4.

Figure 4 Time cost comparison results



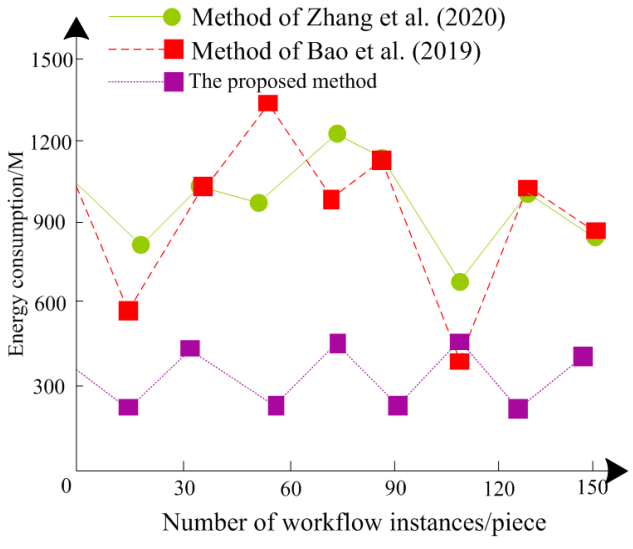
Analysing the experimental results in Figure 4, it can be seen that as the number of workflow instances increases, the scheduling time of different methods shows a gradual increases trend. The characteristics of the change trend are mainly manifested in the workflow scheduling time when the number of workflow instances is less than 25. The growth trend is obvious, and then the growth trend slows down over time. Through comparison, it can be seen that the longest scheduling time of the proposed method is only 2.8 s, while the highest scheduling time of Zhang et al.'s (2020) method and Bao et al.'s (2019) method reaches 7.5 s and 6.2 s. It can be seen that the proposed method The time cost of the method is lower, that is, the workflow scheduling efficiency is higher. This is because the proposed method uses the MCUD algorithm to classify the workflow instances to be processed, which reduces the complexity of workflow task scheduling and improves the workflow scheduling efficiency.

Experiment 2

Taking energy consumption cost as an experimental indicator, comparing the scheduling effect of the method of Zhang et al. (2020) and the method of Bao et al. (2019) with the proposed method, the results are shown in Figure 5.

Analysis of the experimental results in Figure 5 shows that the energy cost in workflow scheduling is not affected by the number of workflow instances, and the three methods all show irregular changes. Among them, the energy consumption cost of the proposed method is lower than that of the two traditional methods, and the energy consumption is always lower than 500 W. Although, Zhang et al.'s (2020) method has a lower energy consumption cost than the proposed method when the number of workflow instances is 110, on the whole, the energy cost of the proposed method is lower. It can be seen that the proposed method effectively reduces the energy consumption cost in workflow scheduling and has higher application value. This is because the proposed method uses a genetic algorithm to generate a scheduling strategy that meets user needs, that is, minimises the execution cost of workflow task scheduling within a given processing time, thereby reducing the energy cost of workflow scheduling.

Figure 5 Comparison results of energy consumption costs (see online version for colours)



Experiment 3

In order to further verify the effectiveness of the proposed method, the resource utilisation effect of the cloud platform is used as an experimental indicator to compare the scheduling effects of the method of Zhang et al. (2020) and the method of Bao et al. (2019) with the proposed method. The results are shown in Table 1.

Table 1 Comparison results of cloud platform resource utilisation effect

Method	*Cloud platform resource utilisation*	*Memory usage*	*Cloud platform resource reliability*
Zhang et al.'s (2020) method	Low	High	Low
Bao et al.'s (2019) method	Medium	High	High
The proposed method	High	Low	High

It can be seen from Table 1 that compared with the other two traditional methods, the proposed method has the characteristics of reliability, high resource utilisation of cloud platform and low memory utilisation. It can ensure the security performance of cloud service workflow scheduling, maximise resource utilisation and improve the application effect of adjustment algorithm. This is because the proposed method introduces a priority rule, that is, the cloud task priority order heuristic rule determines the workflow task order, which further improves the cloud service workflow scheduling effect.

5 Conclusions

In order to solve the problems of high time cost, high energy cost and low utilisation rate of cloud platform resources in traditional methods, this paper proposes a cloud service

workflow scheduling algorithm based on priority rules. The main innovations of this method are as follows:

1 The MCUD algorithm is used to classify the workflow, which effectively reduces the time cost of workflow scheduling.

2 The cloud service workflow scheduling problem is solved based on genetic algorithm to obtain the optimal solution, and priority rules are given to update the workflow scheduling task and priority at the same time, so as to obtain the optimal feasible solution of time cost, so as to further improve the adjustment effect.

3 The experimental results show that the longest workflow scheduling time of the proposed method is only 2.8 s, the energy consumption is always lower than 500 W, the energy consumption cost is much lower than the traditional method, and has the characteristics of reliability, high resource utilisation of cloud platform and low memory utilisation, which fully verifies the effectiveness and practicability of the method.

References

Alaei, M., Khorsand, R. and Ramezanpour, M. (2020) 'An adaptive fault detector strategy for scientific workflow scheduling based on improved differential evolution algorithm in cloud', *Applied Soft Computing*, Vol. 99, No. 2, p.106895.

Aziza, H. and Krichen, S. (2020) 'A hybrid genetic algorithm for scientific workflow scheduling in cloud environment', *Neural Computing and Applications*, Vol. 32, No. 12, pp.1–16.

Bao, X.A., Cao, Y.D., Zhang, N., Qian, J.Y. and Cao, J.W. (2019) 'Multi-objective cloud workflow scheduling algorithm based on grid variance', *Telecommunications Science*, Vol. 35, No. 2, pp.1–13.

Chakravarthi, K.K. and Shyamala, L. (2020) 'TOPSIS inspired budget and deadline aware multi-workflow scheduling for cloud computing', *Journal of Systems Architecture*, Vol. 114, No. 1, p.101916.

Deng, X., Jiang, P., Peng, X. and Mi, C. (2019) 'An intelligent outlier detection method with one class support tucker machine and genetic algorithm toward big sensor data in internet of things', *IEEE Transactions on Industrial Electronics*, Vol. 66, No. 6, pp.4672–4683.

Ijaz, S., Munir, E.U., Ahmad, S.G., Rafique, M.M. and Rana, O.F. (2021) 'Energy-makespan optimization of workflow scheduling in fog-cloud computing', *Computing*, Vol. 103, No. 9, pp.2033–2059.

Jiang, J.S., Yang, B., Miao, Z.M. and Zhu, B.S. (2015) 'A workflow task assignment method based on the properties of task and user', *Computer Simulation*, Vol. 32, No. 12, pp.222–225, p.230.

Karpagam, M., Geetha, K. and Rajan, C. (2020) 'A modified shuffled frog leaping algorithm for scientific workflow scheduling using clustering techniques', *Soft Computing*, Vol. 24, No. 1, pp.637–646.

Kintsakis, A.M., Psomopoulos, F.E. and Mitkas, P.A. (2019) 'Reinforcement learning based scheduling in a workflow management system', *Engineering Applications of Artificial Intelligence*, Vol. 81, No. 5, pp.94–106.

Nik, S.M., Naghibzadeh, M. and Sedaghat, Y. (2020) 'Cost-driven workflow scheduling on the cloud with deadline and reliability constraints', *Computing*, Vol. 102, No. 2, pp.477–500.

Peng, D., Tan, G., Fang, K., Chen, L. and Zhang, Y. (2021) 'Multiobjective optimization of an off-road vehicle suspension parameter through a genetic algorithm based on the particle swarm optimization', *Mathematical Problems in Engineering*, Vol. 2021, No. 9, pp.1–14.

Ranjan, R., Thakur, I.S., Aujla, G.S., Kumar, N. and Zomaya, A.Y. (2020) 'Energy-efficient workflow scheduling using container-based virtualization in software-defined data centers', *IEEE Transactions on Industrial Informatics*, Vol. 16, No. 12, pp.7646–7657.

Rezaeian, A., Naghibzadeh, M. and Epema, D. (2019) 'Fair multiple-workflow scheduling with different quality-of-service goals', *Journal of Supercomputing*, Vol. 75, No. 2, pp.746–769.

Sun, J., Yin, L., Zou, M., Zhang, Y. and Zhou, J. (2020) 'Makespan-minimization workflow scheduling for complex networks with social groups in edge computing', *Journal of Systems Architecture*, Vol. 108, No. 4, p.101799.

Wang. Y. (2019) 'Enhanced multi-objective evolutionary algorithm for workflow scheduling on the cloud platform', *Journal of Xidian University (Natural Science)*, Vol. 46, No. 1, pp.130–136.

Zhang, L.X., Zhou, L.Q., Wen, H., Xiao, M.S. and Deng, X.J. (2020) 'Energy efficient scheduling algorithm of workflows with cost constraint in heterogeneous cloud computing systems', *Computer Science*, Vol. 47, No. 8, pp.112–118.