A Multi-Objective Cloud Workflow Scheduling Optimization Based on Evolutionary Multi-objective Algorithm with Decomposition

XueLi Yao (yao@huel.edu.cn)
Henan University of Economics and Law https://orcid.org/0000-0003-3461-4388

Research Article

Keywords: multi-objective optimization, cloud workflow scheduling, evolutionary algorithm, decomposition method, local search

Posted Date: June 28th, 2021

DOI: https://doi.org/10.21203/rs.3.rs-604125/v1

A Multi-objective Cloud Workflow Scheduling Optimization Based on Evolutionary Multi-objective Algorithm with decomposition

*Xue Li-Yao

School of Information, Henan University of Economics and Law, Zheng Zhou, 450000, China, yao@huel.edu.cn

Abstract: In the cloud computing environment, cost-effective workflow task scheduling is the key problem that cloud computing service providers need to solve. However, previous scheduling methods only consider one-sided demands, such as minimizing running time or running cost. In this paper, the cloud workflow scheduling model including two minimizing time and execution cost are established, and then the MOEA/D algorithm based on weight vector adjustment and local search is proposed, and the algorithm is applied in the model solving process. Firstly, the weight vector adjustment method is employed to obtain more evenly distributed solutions; and in order to obtain more evenly distributed solutions and hope to speed up the convergence speed of the solution process, this paper adds local search operators into the solution process of evolutionary algorithm, and proposes MOEA/D algorithm based on local search and weight vector adjustment as an improved multi-objective optimization algorithm to solve the cloud workflow scheduling model based on time and execution cost, it can be turned out that MOEA/D algorithm based on local search and weight vector adjustment can obtain more evenly distributed solutions than MOEA/D algorithm and NSGA-II algorithm on the basis of faster convergence speed, which provides decision support for cloud workflow scheduling decision-makers.

Keywords: multi-objective optimization, cloud workflow scheduling, evolutionary algorithm, decomposition method, local search

1. Introduction

In the fields of astronomy, geography, bioinformatics and physics, workflow is often used to model and execute large-scale complex problems [1]-[2]. Among them, workflow in the field of scientific research can support large-scale and complex scientific processes, such as simulating experiments, proving scientific falsehood and visualizing scientific data[3]-[5]. To be sure, workflow provides an effective way to process and extract information from the growing mass of data. And, because workflow is often composed of many tasks with complex control and data dependence, the execution of workflow tasks usually involves invoking many different distributed computing services for intensive data analysis and collaborative knowledge discovery. Workflow can make full use of the characteristics of cloud computing environment, such as resource flexibility, scalability, and payment on demand, which enables users to deploy large-scale and complex applications at a very low cost, and dynamically adjust the resource configuration at different stages of the application. Therefore, workflow tasks are very suitable for running in the cloud computing environment.

Cloud workflow scheduling algorithm is a key factor that enables workflow to effectively utilize the characteristics of cloud computing environment. The scheduling algorithm is responsible for effectively scheduling workflow tasks in a group of computing resources while maintaining task data dependence. As a typical NP hard problem [6], it is unable to find the optimal solution of cloud workflow scheduling problem in polynomial time. Traditional scheduling algorithms, such as first come first service, priority ranking and so on, can only get one solution, and users can not make appropriate decisions according to their own preferences. In addition, many traditional scheduling algorithms seldom consider the multi-objective nature of workflow scheduling in cloud computing environment, such as users want to spend the minimum cost while minimizing the completion time.

Multi objective optimization technology makes it possible for users to make preference decisions

from multiple optimization solutions. For multi-objective optimization of cloud workflow scheduling, there is no one decision that can optimize all the objectives at the same time, but can obtain a set of compromise Pareto decisions with multiple conflicting objectives[7]. It is difficult to obtain the real Pareto decision set of cloud workflow scheduling, and it is often unnecessary to obtain the real Pareto decision set. Generally, a group of Pareto asymptotically optimal decisions with uniform distribution in the target space are obtained where evolutionary algorithm can effectively solve complex problems by learning from the evolution operations of natural organisms such as heredity and mutation [8]-[9]. Based on the above analysis, the cloud workflow scheduling method based on evolutionary multi-objective optimization has very important value and significance. This paper establishes a multi-objective optimization model of cloud workflow based on time as well as execution cost, then an improved MOEA/D algorithm based on weight vector adjustment and local search is proposed, Compared with MOEA/D algorithm and NSGA-II algorithm, the algorithm proposed in this paper can obtain a group of Pareto optimal solutions with uniform distribution on the basis of faster convergence speed, which can provide decision support for cloud workflow scheduling problem.

The specific chapters of this paper are arranged as follows: chapter 2 mainly introduces the literature review of cloud workflow scheduling problem and multi-objective algorithm; in chapter 3, we analyze the goal conflict of cloud workflow scheduling and establish the workflow scheduling model in cloud environment; the MOEA/D algorithm based on local search and weight vector adjustment is designed and applied to the cloud workflow scheduling model based on completion time and execution cost in chapter 4.And the fifth chapter summarizes and prospects followed.

2. Literature Review

2.1 Cloud workflow scheduling

The research content of cloud workflow scheduling is to map the task set of workflow and the computing resource set of running task one by one. Once the task is successfully executed, the results will be returned to users through the Internet. According to the number of cloud workflow scheduling optimization objectives, cloud workflow scheduling can be divided into single objective optimization cloud workflow scheduling and multi-objective optimization cloud workflow scheduling.

Among them, the single objective optimization of cloud workflow scheduling research mainly aims at one of the cost, time and other indicators. Wu et al. [10] mainly adopted a task level scheduling method based on market model to minimize the overall operation cost on the premise of meeting the constraints of cloud workflow task service quality. Liu et al. [11] used the co evolutionary genetic algorithm to study the case of reducing the running cost and running time under the condition of meeting the user deadline. Tirapat et al. [12] used hybrid genetic algorithm and particle swarm optimization algorithm to minimize the total cost, including execution cost and data transmission cost, under the limitation of task completion time. Mosleh et al. [13] calculated the completion time of data access by considering the network service time and the arrival rate of network input and output requests under the deadline limit, and then analyzed and allocated the cost of data path according to the task priority, so as to save the execution cost and data transmission cost. Sossa et al. [14] used deadline constrained meta heuristics to schedule scientific workflow applications on infrastructure service level cloud. Calheiros et al. [15] adopted an optimal scheduling method in the hybrid cloud environment to optimize the execution cost under the condition of meeting the execution time constraint. Feller et al. [16] modeled the load migration problem as a multidimensional bin packing model, and used the method based on ant colony algorithm to solve the model. Lin et al. [17] focused on the optimization of execution time, which reduced the execution time by allocating resources flexibly.

Single objective optimization of cloud workflow scheduling has been unable to meet the growing needs of users, such as minimizing the running time and the execution cost. In this trend,

multi-objective optimization of workflow scheduling in cloud computing environment has attracted more and more attention of researchers. Barrionuevo et al. [18] proposed a heuristic list scheduling algorithm based on Pareto domination, which optimizes the task completion time and the user cost of task execution at the same time, and provides a set of optional optimal scheduling schemes for users. Zhu et al. [19] used evolutionary multi-objective optimization algorithm to solve the cloud workflow scheduling problem of optimizing task completion time and task execution cost at the infrastructure as a service level, and proposed a new scheme of coding method, population initialization, fitness evaluation and genetic operator operation for the problem. Wang et al. [20] proposed an optimal scheduling algorithm which optimizes the task completion time and reliability of cloud workflow application at the same time. Fard et al. [21] proposed an effective multi-objective workflow scheduling algorithm in heterogeneous systems which considers the indicators including task completion time, task execution user cost, reliability and energy consumption, and gives an effective scheduling scheme on the basis of meeting user related constraints. Wu et al. [22] considered the time limit and budget constraints, optimized the energy consumption and reliability simultaneously, and used the general list scheduling algorithm and tuning mechanism to solve the multi constraint multi-objective optimization cloud workflow scheduling problem. Padmaveni et al. [23] used memetic algorithm to optimize task completion time and task execution cost simultaneously. Compared with genetic algorithm, the workflow scheduling scheme solved by this algorithm has better scheduling scheme. Pandey et al. [24] used particle swarm optimization algorithm to optimize the execution cost and data transmission cost. Saurabh et al. [25] used a multi-objective optimization method based on cat swarm to schedule workflow tasks in cloud computing environment. The objective of optimization is to minimize the cost of task execution, task running time and CPU idle time. Yang et al. [26] proposed a novel intermediate data storage strategy to reduce the execution of scientific workflow and the cost of data transmission. This strategy can automatically and dynamically select appropriate intermediate data sets to store or delete in the cloud environment. Duan et al. [27] proposed a communication and storage aware method to optimize both task execution time and execution cost under bandwidth and storage constraints.

2.2 Multi-objective optimization algorithm

Based on the characteristics of population evolution in natural evolution, evolutionary algorithm (EA) can realize the diversity and globality of search in the process of evolution. It is not limited by the shape and continuity of search space, and has good robustness and versatility. It is widely used to solve complex NP hard problems. Moreover, EA can get a set of optimal solutions in one run, Users can make decisions according to their own preferences. In the research of multi-objective algorithm, vector evaluation genetic algorithm (VEGA) proposed by Schaffer [28]; Goldberg proposed to select the non-dominated solution set by Pareto dominance relation and obtain the diversity of non-dominated solution set by niche technology [29]. Fonseca and Fleming proposed the first MOEA [30] based on Pareto dominance relation, namely multi-objective genetic algorithm (MOGA). Among them, the genetic algorithm [31] (NSGA) proposed by Srinivas and DEB in 1994 and the niche non dominated genetic algorithm (NPGA) [32] proposed by horn in the same year are the most representative. After that, Zitzler and Thiele proposed the intensive Pareto evolutionary algorithm (SPEA) [33], and elite individuals began to become the focus of evolutionary multi-objective optimization algorithm design. For example, in 2002, Zitzler proposed the second generation of intensive Pareto evolutionary algorithm [34] (SPEA2), and in 1999, Knowles et.al proposed the Pareto archive evolutionary strategy [35] (PAEs). In 2002, Deb et al. [36] proposed the second generation non dominated solution sorting algorithm (NSGA-II). In SPEA algorithm, individual fitness is determined by Pareto strength value, which improves the diversity of population by clustering; SPEA2 adopts the enhanced fitness allocation and clustering method; PAEs uses grid based methods for individual selection and diversity preservation;

NSGA-II makes individual selection based on fast non dominated solution ranking and crowding distance, which reduces the computational complexity and effectively improves the convergence of the solution.

In addition, some new concepts such as hybrid method, coevolution, parallel method and quantum evolution are proposed one after another. Some traditional evolutionary algorithms such as particle swarm optimization, ant colony algorithm and artificial immune algorithm are also introduced into the algorithm design of multi-objective optimization; Individual selection mechanism is no longer limited to the concept of Pareto domination. For example, Zitzler proposed a multi-objective optimization algorithm IBEA [37] based on evaluation index in 2004. IBEA uses binary performance index such as HV as the standard of individual evaluation; In 2007, Zhang et al. [38] proposed an evolutionary multi-objective optimization algorithm based on decomposition (MOEA/D), which decomposes the multi-objective optimization problem into a set of scalar subproblems by decomposition method, and then uses evolutionary algorithm to solve the subproblem [39] - [42].

3. Cloud workflow scheduling problem model

Usually, the workflow task of complex problem is modeled as a directed acyclic graph (DAG), in which each vertex represents the computing task of workflow, and each edge represents the data and execution dependency of workflow task. Given a workflow with n tasks, m virtual machine provided by a cloud service provider, and each virtual machine has k common types, then there are $m^n \cdot k^m$ scheduling schemes, and the scheduling algorithm should not only meet the task constraints of the workflow, but also consider the user's quality of service requirements, which are often conflicting, such as completion time and cost, this brings great challenges to cloud workflow scheduling, and multi-objective optimization of cloud workflow scheduling is the focus of this paper.

3.1 Task model

The task model of cloud workflow in design phase can be modeled as directed acyclic graph （DAG） $G_{cs} = (V_{cs}, E_{cs})$， the node set of DAG $V_{cs} = \{s_1, s_2, ..., s_n\}$ represents the n task sequences submitted by the user to the workflow management system，the start task is indicated by s_1， the end task is indicated by s_n.The edge set E_{cs} in DAG represents the data transmission dependency between n workflow tasks，for the data transmission dependency $e_{i,j}$ between task s_i and task s_j , there are $e_{i,j} \in E_{cs}$， and the weight on the side represents the size of the data transmission.

In order to describe the task model of the cloud workflow in more detail, here is a four-task workflow as an example, as shown in Fig.1. S_1 sends data of $W_{1,2}, W_{1,3}, W_{1,4}$ to the subsequent tasks S_2, S_3, S_4， S_2, S_3, S_4 start execution after obtaining the data. After the execution is completed, the data with the data volume of $W_{2,5}, W_{3,5}, W_{4,5}$ are sent to task S_5, and it can only be executed when S_5 has all the required data.

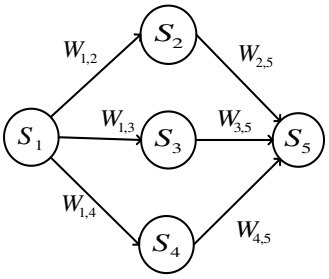


Fig. 1 Cloud workflow task model illustration

3.2 Resource model

In order to describe the resource model in the form of a directed acyclic graph more specifically, a virtual machine set containing four virtual machines is used as an example for illustration. In Fig. 2, virtual machines V_1, V_2, V_3, V_4 are linked to each other through the network，the data on the connection side in Fig. 2 is the communication bandwidth and minimum link delay of the network link. For example, the network communication bandwidth from V_1 to V_2 is $B_{1,2}$ and the minimum link delay is $D_{1,2}$, and the network communication bandwidth from V_2 to V_1 is $B_{2,1}$ and the minimum link delay is $D_{2,1}$, similar to other virtual machines.

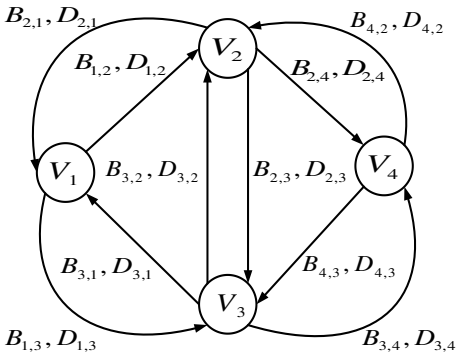


Fig. 2 Illustration of the resource model of the cloud computing environment

3.3 Scheduling model

According to the above cloud workflow task model and the resource model of the cloud computing environment, the scheduling of cloud workflow tasks can be abstracted as a mapping between two directed acyclic graphs，that is, from the DAG collection of workflow tasks to the virtual machine resource DAG. Specifically, the mapping from DAG's cloud workflow task model $G_{cs} = (V_{cs}, E_{cs})$ to DAG's cloud computing resource model $G_{pc} = (V_{pc}, E_{pc})$， a reasonable scheduling plan will always make each task in the cloud workflow correspond to a suitable virtual machine, and the total number of tasks in the model adopted in this article is not greater than the total number of virtual machines, and it is assumed that each workflow task can only be allocated at the same time to a virtual machine resource, the parameters of the above graph mapping model are defined as follows:

Task mapping： $(x_{s_i,v_l}) | V_{cs} | \times | V_{pc} |, \forall s_i \in V_{cs}, \forall v_l \in V_{pc}$ 。If the cloud workflow task s_i executed on the virtual machine v_l, there is $x_{s_i,v_l} = 1$, otherwise, $x_{s_i,v_l} = 0$.

Data transmission：$(y_{e_{i,j},l_{l,o}})|E_{cs}|\times|E_{pc}|,\forall e_{i,j}\in E_{cs},\forall l_{l,o}\in E_{pc}$，if the data transmission edge $e_{i,j}$ between tasks is mapped to the virtual machine link $l_{l,o}$ for transmission, then $y_{e_{i,j},l_{l,o}}=1$, otherwise, $y_{e_{i,j},l_{l,o}}=0$.

3.4 Scheduling goals

As we all know,the service-based cloud computing must be based on improving the user experience, and the ET is the most influential target. And , the commercial cloud computing technology must consider the user's economic cost, the execution cost are the main components of user's economic cost, and they are also the goals that users care about. To sum up, ET and EC are selected as scheduling goals respectively, and establishes the cloud workflow scheduling model based on multi-objective optimization of formula (1).

$$\begin{cases}\min(ET)=\min(\sum\limits_{t_i\in CP}\sum\limits_{j=1}^{m}(x_{i,j}\cdot(c(t_i)/p(y_i)))+\sum\limits_{w_{i,j}\in CP}\sum\limits_{k=1}^{m}\sum\limits_{l=1}^{m}(x_{i,k}\cdot x_{j,l})\cdot(w_{i,j}/b_{k,l}+d_{k,l}))\\ \min(EC)=\min(\sum\limits_{i=1}^{m}(\lceil rt(v_i)\rceil\cdot vcharge(y_i)))\end{cases}\quad(4\text{-}4)$$

4. The improved MOEA/D algorithm

MOEA/D algorithm has been widely used in the field of multi-objective optimization, which uses decomposition strategy to transform the solution of multi-objective optimization problem into the solution of multiple single objective problems. At present, the commonly used methods are weighted sum method, Chebyshev decomposition method and boundary crossing method.

The Chebyshev decomposition method is used to describe the flow of the MOEA/D algorithm. Let N be the population size and m be the number of optimization objectives. MOEA/D decomposes a multi-objective optimization problem into For N scalar quantum problems, MOEA/D optimizes these N sub-problems at the same time in an evolution. In MOEA/D, the neighbors $\{\lambda^1,\lambda^2,...,\lambda^N\}$ of the weight vector λ^i are taken as the nearest weight vectors, and the neighbors of the corresponding sub-problem i are the sub-problems with the weight vector λ^i. In each generation, the information to be stored in MOEA/D using Chebyshev decomposition method is:

（1）N points that make up a group：$x^1,x^2,...,x^N\in\Omega$，where x^i is the current optimal solution of subproblem i.

（2）$FV^1,FV^2,...,FV^N$， and $FV^i=F(x^i),i=1,2,...,N$.

（3）$z=(z_1,z_2,...,z_m)^T$， where z_i is the optimal value of the objective function f_i found so far.

（4）The external population (EP) is used to store the non-dominated solutions found in the search process.

In this paper, the weight vector adjustment strategy is introduced to improve the solution distribution of MOEA/D under the framework of MOEA/D, and then the local search is added to improve the efficiency of MOEA/D.

4.1 Adjustment of weight vector

In order to obtain the Pareto optimal solution uniformly distributed on the front of PF, a very direct method is to delete the solution in the crowded region and add the solution to the sparse region. However, for the problem of PF discontinuity, it is a challenge to identify the discontinuous region. For the corresponding subproblem of the discontinuous region, it will waste computing resources, because

there is no Pareto optimal solution in the discontinuous region. This paper uses an external elite population to guide the change of subproblems, that is, to add subproblems to truly sparse regions instead of discontinuous regions, and to delete subproblems to dense regions. When an elite individual is located in a sparse region, it will be introduced into the evolutionary population, and a new weight vector will be generated and added to the corresponding subproblem. This strategy of elite population is helpful to add subproblems to the really sparse region, because when the evolutionary population evolves to a certain extent, the number of subproblems increases.Besides,this paper adopts the strategy based on weight vector adaptive adjustment to obtain the uniformly distributed Pareto optimization solution. The adjustment strategy can remove the redundant solutions of subproblems, which can help improve the computational efficiency of the algorithm.

In order to evaluate the sparse and crowded areas in the population, this paper adopts a crowded evaluation method using k nearest neighbors proposed by Deb et al. The calculation formula is as follows formula (20. Here， $L_2^{NN_i^j}$ is the Euclidean distance from the j-th solution to the i-th solution closest to it.

$$V^j = \prod_{i=1}^{k} L_2^{NN_i^j} \tag{2}$$

Based on the crowding evaluation method of k nearest neighbors, the following formula can be used to evaluate the sparseness of individuals in the population. The calculation formula is as follows:

$$SL(ind^j, pop) = \prod_{i=1}^{m} L_2^{NN_i^j} \tag{3}$$

The specific adjustment algorithms are as follows:

Algorithm 4.1 Delete the sub-problem of the crowded area

Input： $evol\_pop$ ： Current evolutionary population；

nus ： Maximum number of sub-questions to be deleted;

Output： $evol\_pop'$ ： The evolution group after deleting the crowded area sub-problem.

Step 1 Update the outer population of the current evolutionary population EP：

For each individual in the evolutionary population, if $g^{te}(x^i \mid \lambda^j, z) < g^{te}(x^j \mid \lambda^j, z)$, $x^i, x^j \in evol\_pop$， $i, j = 1, 2, ... \mid evol\_pop \mid$,let $x^j = x^i, FV^j = FV^i$， FV^j、FV^i indicate the objective function of x^j and x^i.

Step 2 Calculate the sparseness of each individual in the evolutionary population according to formula (4-3).

Step 3 Delete the congested sub-question.

Step 4 Termination condition:

If the number of deleted sub-questions does not reach the required number, delete the individual with the smallest sparse degree and then go to Step 2，Otherwise, output the remaining population as the evolutionary population after deleting the crowding sub-problem $evol\_pop'$.

Algorithm 4.2 Add new sub-problems to sparse areas

Input： $evol\_pop'$ ： Population after deleting sub-problems； z ： Reference point；

EP ： External population； nus ： The maximum number of sub-questions

Output：$evol\_pop^{''}$：Adjusted population

Step 1 Remove individuals dominated by $evol\_pop^{'}$ in the external population EP

Step 2 Calculate the sparseness of individuals in the external population EP in $evol\_pop^{'}$

Step 3 Add a sub-problem to the sparse area.

Step 4 Termination condition:

If the number of inserted sub-problems reaches nus, output the current population as $evol\_pop^{''}$, otherwise go to Step 2.

4.2 Local search

In order to obtain better convergence results, this paper introduces a local search method in the framework of the MOEA/D algorithm, and uses a three-point quadratic interpolation approximation method as the local search method. This method is simple to calculate and suitable for local search. Operator, the local search algorithm is as follows:

Algorithm 4.3 Local search

Input：x：Sub-problem to perform partial search

Step 1：Select neighbourhood individuals

Choose the optimal two individual decision variables $x^{i_b}=(x_1^{i_b},x_2^{i_b},...,x_n^{i_b})^T$ and $x^{i_c}=(x_1^{i_c},x_2^{i_c},...,x_n^{i_c})^T$ in the neighborhood of the sub-problem and the sub-problem x itself, where the decision variables x are denoted as $x^{i_a}=(x_1^{i_a},x_2^{i_a},...,x_n^{i_a})^T$, where, n represents the number of decision variables.

Step 2 Get a new sub-question:

Approximate the decision variables of the new sub-problem according to the following formula $x^{i'}=(x_1^{i'},x_2^{i'},...,x_n^{i'})^T$：

If $(x_k^{i_a}-x_k^{i_c})g_{i_b}^{te}+(x_k^{i_c}-x_k^{i_b})g_{i_a}^{te}+(x_k^{i_b}-x_k^{i_a})g_{i_c}^{te}>\varepsilon$， then：

$$x_k^{i'}=\frac{1}{2}\cdot\frac{\left[(x_k^{i_a})^2-(x_k^{i_c})^2\right]g_{i_b}^{te}+\left[(x_k^{i_c})^2-(x_k^{i_b})^2\right]g_{i_a}^{te}+\left[(x_k^{i_b})^2-(x_k^{i_a})^2\right]g_{i_c}^{te}}{(x_k^{i_a}-x_k^{i_c})g_{i_b}^{te}+(x_k^{i_c}-x_k^{i_b})g_{i_a}^{te}+(x_k^{i_b}-x_k^{i_a})g_{i_c}^{te}}$$

Here，$k=1,2,...,n$ is the decision variable，$\varepsilon=10^{-7}$。

3 Update：

If $g^{te}(x_k^{i'}\mid\lambda^i,z)\le g^{te}(x_k^{i}\mid\lambda^i,z)$，then $x^i=x^{i'}$。

4.3 MOEA/D algorithm based on local search and weight vector adjustment

Based on the above-mentioned weight vector adjustment strategy and local search method, this paper proposes an improved MOEA/D algorithm, denoted as the LS-MOEA/D algorithm. The algorithm solution process is as follows:

Algorithm 4.4：LS-MOEA/D

Input: Stop criterion: Maximum evolutionary algebra G_{max}；N：Number of sub-problems；$\lambda^1,\lambda^2,...,\lambda^N$：Uniformly distributed weight vectors；T：The number of neighbors of the sub-problem;

$evolrate$：Timing of weight vector adjustment;

wag：The frequency of weight vector adjustment.

Output：$PS:\{x^1,x^2,...,x^N\}$和$PF:\{F(x^1),F(x^2),..,F(x^N)\}$.

Step 1 initialization

Step 1.1：Set $EP=\varnothing$，$gen=0$.

Step 1.2：Calculate the Euclidean distance of any two vectors, and select the nearest weight vector as its neighbor for each weight vector. Assume $B(i)=\{i_1,i_2,...,i_T\},i=1,2,...,N$,, $\lambda^{i_1},\lambda^{i_2},...,\lambda^{i_T}$ Is the nearest T weight vectors of λ^i.

Step 1.3：Initial population $x^1,x^2,...,x^N$，set $FV^i=F(x^i),i=1,2,...,N$。

Step 1.4：Initialize the reference point according to the question $z=(z_1,z_2,...,z_m)^T$.

Step 2 Update：

For $i=1,2,...,N$ do：

Step 2.1：Randomly select two neighbors x^k and x^l from $B(i)$ and use the evolution operator to generate a new solution y.

Step 2.2：For the newly generated y, the problem-related improvement strategy is used to generate y'.

Step 2.3：If $z_j<f_j(y')$，then $z_j=f_j(y'),j=1,2,...,m$.

Step 2.4：If $g^{te}(y'\mid\lambda^i,z)\le g^{te}(x^j\mid\lambda^i,z),j\in B(i)$，then $x^j=y',FV^j=F(y')$。

Step 2.5：Remove all solutions in EP that are dominated by $F(y')$; if $F(y')$ is not dominated by any solution in EP, move $F(y')$ into EP.

Step 2.6 Local search: Algorithm 4.3 is used for local search.

Step 3 Weight vector adjustment:

If $gen\ge evolrate*G_{max}$ and $gen\ mod\ wag=0$，adjust the weight vector:

Step 3.1 For the newly generated population, the non-dominated solution sorting method based on proximity distance is used to update the EP.

Step 3.2 Algorithm 4.1 is used to delete the sub-problems of crowded areas.

Step 3.3 Algorithm 4.2 is used to add sub-problems to sparse regions.

Step 3.4 Find the nearest T weight vectors for λ^i to construct a new neighbor list $B(i)$, where $i=1,2,...,N$，otherwise go to **Step 4**.

Step 4 Stop judgment：

If $gen\ge G_{max}$，stop and output $PS:\{x^1,x^2,...,x^N\}$ and $PF:\{F(x^1),F(x^2),..,F(x^N)\}$, otherwise $gen=gen+1$ and go to Step2.

5. Experimental results and analysis

5.1 Scheduling method

The solution process of the improved cloud workflow scheduling method based on local search and weight vector adjustment is described as follows:

1. Initialization, which mainly includes the following steps:

a. Read parameter configuration data, including the maximum number of virtual machines, delay between links, virtual machine performance, etc., and establish a simulated cloud computing resource model.

b. Read the workflow topology data and establish the DAG task model of the cloud workflow.

c. Read task mapping constraints. This article adopts a full mapping structure, that is, any virtual machine can be selected for each workflow task.

2. Determine optimization goals. This article optimizes two sets of goals (time and execution cost and time and transmission cost).

3. Encoding and decoding:

a. coding: The mapping selection of n workflow tasks to m virtual machines (each virtual machine has k types) can be abstracted as a one-dimensional array of n+m integers, and the first n integers in the array represent the choice of virtual machines , The following m integers represent the choice of the type of virtual machine. For the convenience of evolutionary algorithm calculations, these n+m integers are mapped to (0,1). The mapping formula is:

$$x_{i,j} = \begin{cases} x_{i,j} / m, \text{ if } 0 \le \text{j<n} \\ x_{i,j} / k, \text{ if n} \le \text{j<n+m} \end{cases}, 0 \le i < n \qquad (3)$$

Among them, represents the selection of virtual machine or virtual machine type for the first workflow task. Here， $x_{i,j}$ represents the selection of virtual machine or virtual machine type for the i-th workflow task.

b. Decoding: Map back to the problem space through the inverse mapping formula, the inverse mapping formula is:

$$x_{i,j} = \begin{cases} \lfloor x_{i,j} * m \rfloor, \text{ if } 0 \le \text{j<n} \\ \lfloor x_{i,j} * k \rfloor, \text{ if n} \le \text{j<n+m} \end{cases}, 0 \le i < n \qquad (4)$$

4. Initialize the population, randomly initialize n real numbers between (0,1), determine the mapping relationship after decoding as the mapping from the initial workflow to the task, and calculate the objective function value corresponding to the initial population individual.

5. To solve the population evolution, the evolution process is the same as that in Algorithm 4.4.

6. At the end of the algorithm, a set of decisions obtained in the last generation and their objective function values are output for users to choose.

5.2 Experimental evaluation index of multi-objective

The results obtained by the multi-objective evolutionary algorithm are multiple asymptotically optimal solutions which are close to the optimal solution. For these asymptotically optimal solutions, the first concern is the convergence degree to the real PF, and the second concern is the properties of the solution distribution (uniform distribution and broad). In this paper, we use the HV to measure the volume of a hypercube surrounded by an approximation solution set P and a reference point R^P to achieve these two goals.

Let R^P be a reference point, $P = \{p^1, p^2, ..., p^N\}$ indicates a set of non-dominated solutions, and when it satisfies $\forall 1 \le i \le N, p_i \text{ p } R^p$, p_i is an optimal solution with uniform distribution on PF

$$HV(P, R^p) = volume\left(\bigcup_{i=1}^{N} HyperRectangle\left(p^i, R^p \right) \right) \qquad (5)$$

Here $HyperRectangle(p_i, R^p) = \{v \mid p_i \le v_i \le R^p, i = 1,...,m\}$ indicates the volume of hypercube formed by the i th non dominant solution of reference points R^p . For the two target cases shown in Fig. 3, the index consists of the area of multiple rectangles in the gray part of the figure.

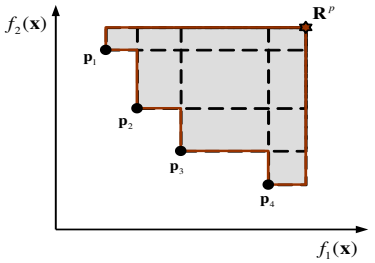


Fig. 3 HV diagram

For a reference point R^p , the larger the area of multiple rectangles, that is, the larger the value of super volume (HV), which means that the closer the solution set of the algorithm is to the lower left corner of the coordinate, the better the convergence and distribution effect of the algorithm (for the minimization problem).

5.3 Parameter setting

The improved MOEA/D algorithm proposed in this paper (LS-MOEA/D, MOEA/D algorithm based on local search and weight vector adjustment), NSGA-II and MOEA/D algorithm are compiled and implemented on vs2015, and the statistics of experimental results are realized by MATLAB. For four types and three scales, a total of 12 workflows, the three algorithms (LS-MOEA/D, MOEA /D and NSGA-II) adopt the same evolutionary operator (SBX for crossover operator and PM for polynomial mutation operator), and the individual adopts real number coding from 0 to 1. The specific parameter settings are shown in the table below (n in the table represents the number of decision variables).

Tab. 1 SBX and PM parameter setting

Parameter	LS-MOEA/D	MOEA/D	NSGA-II
Crossover Probability	1	1	1
Crossover distribution index	20	20	20
Mutation Probability	1/n	1/n	1/n
Mutation distribution index	20	20	20

LS-MOEA/D, MOEA/D and NSGA-II all run 30 times independently for 12 workflow scheduling problems, the population size of all problems is set to 100, the stop criterion of all algorithms is set to the maximum number of function evaluations is 300000, for LS-MOEA/D, the number of subproblems to be adjusted (deleted or added) is set to 5, and the time of weight vector adjustment is evol_ Rate = 0.5, adjust the frequency wag=10, and set the number of neighbor vectors for MOEA/D and LS-MOEA/D to 10.

5.4 Experimental results

For the results of 30 independent runs of the three algorithms on 12 workflows, the statistical results of the last generation PF are given in table 4.2, including the average value and variance of HV index, and the average value of HV index of the three algorithms is sorted. The larger the HV index is, the higher the ranking is. In order to test the superiority of LS-MOEA/D compared with MOEA/D and NSGA-II, and 30 times of HV index data for t-test, the significance level is 0.05. If LS-MOEA/D method is better than the comparison of MOEA/D or NSGA-II at the significance level of 0.05, it is indicated by "+" sign, If LS-MOEA/D method is significantly worse than the comparison MOEA / D or NSGA-II at a level of 0.05, it is indicated by '-' sign. If both of them are not significant, it is indicated by '=' sign. From the final statistics of HV, LS-MOEA/D algorithm proposed in this paper is superior to

NSGA-II and MOEA / D in terms of time and execution cost, Not only the average performance of HV index is the best, but also the variance of HV index is small, which shows that the algorithm proposed in this paper is relatively stable.

For the scheduling of 12 kinds of actual workflows based on time and execution cost, the average HV trend chart of 30 independent runs is shown in Fig. 4. From the following HV trend, it can be observed that the proposed method LS-MOEA/D has faster convergence for most of the 12 kinds of actual workflows, and this trend becomes more obvious with the increase of the problem scale, It shows that the local search operator used in LS-MOEA/D can accelerate the convergence of the cloud workflow scheduling problem based on time and execution cost. Moreover, for the 0.5 times of the maximum function evaluation times in HV trend, LS-MOEA/D algorithm begins to use the weight vector adjustment method to adjust the population distribution, From the trend of HV, we can see that the HV of the method proposed in this paper starts to improve further. Combined with the POF of the last generation population in Fig. 5, we can see that compared with the other two methods, the method proposed in this paper has better distribution for most of the 12 actual workflows, This also shows that the weight vector adjustment method has a good adjustment effect on the distribution of cloud workflow scheduling problem solution. To sum up, the LS-MOEA/D proposed in this paper can quickly and effectively get a group of evenly distributed decisions when solving the cloud workflow scheduling problem based on time and execution cost. For cloud workflow scheduling with more tasks, the proposed algorithm can get a scheduling scheme with less time and less execution cost in the same time, It can provide good decision support for cloud workflow scheduling based on time and execution cost.

Tab.2 Statistical data of HV index for 30 runs of ET-EC as scheduling target

Workflow	LS-MOEA/D	MOEA/D	NSGA-II
CyberShake-30	1083.4285	1083.1689	**1083.4631**
	(0.0335)	(0.0435) +	**(0.0196) -**
CyberShake-50	2271.8222	2267.2925	2271.4922
	(2.1862)	(4.2267) +	(2.4055) =
CyberShake-100	**4474.8942**	4407.0431	4427.9576
	(31.0694)	(47.0246) +	(43.2708) +
Epigenomics-24	**1044326.3730**	1044220.7865	1044214.5424
	(24.2207)	(78.1525) +	(34.7772) +
Epigenomics-46	**2767550.0085**	2763513.8491	2763996.6391
	(17.7896)	(28.4686) +	(18.3378) +
Epigenomics-100	**113946497.5414**	113351862.0430	113036079.2420
	(118335.0683)	(166740.2771) +	(419149.9091) +
Inspiral-30	63155.6168	63151.7381	63152.9056
	(14.0289)	(20.7049) =	(4.5199) =
Inspiral-50	**124702.3928**	123331.0256	123418.9720
	(145.6975)	(1134.8579) +	(1048.3007) +
Inspiral-100	**156625.3244**	150861.9754	154093.5633
	(1457.4480)	(1570.4151) +	(1549.5953) +
Montage-25	107.4386	107.6992	107.9583
	(0.8166)	(0.2048) =	(0.1399) =
Montage-50	**144.6738**	143.3617	144.0308
	(0.0544)	(0.0622) +	(0.0487) +
Montage-100	**382.1310**	369.5895	380.7976
	(2.0350)	(2.4675) +	(2.0838) +

Rank	1.25	2.75	2.0

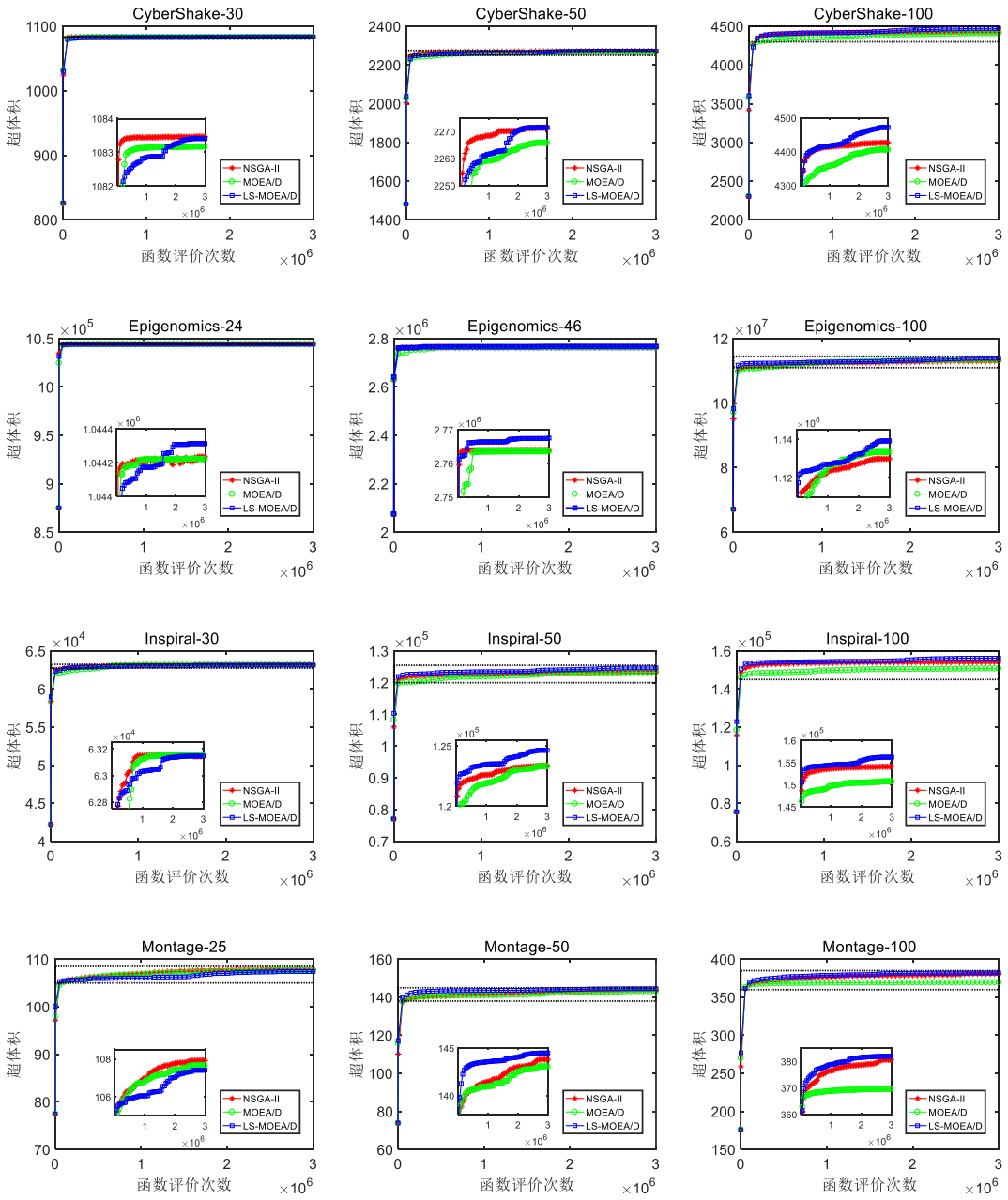
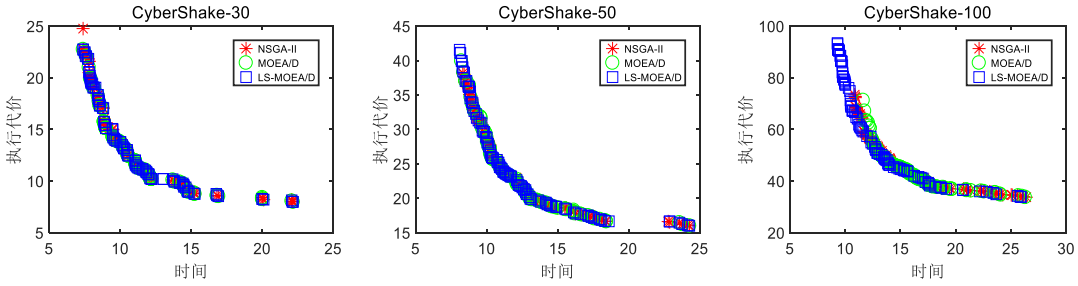


Fig.4 The trend of Average HV



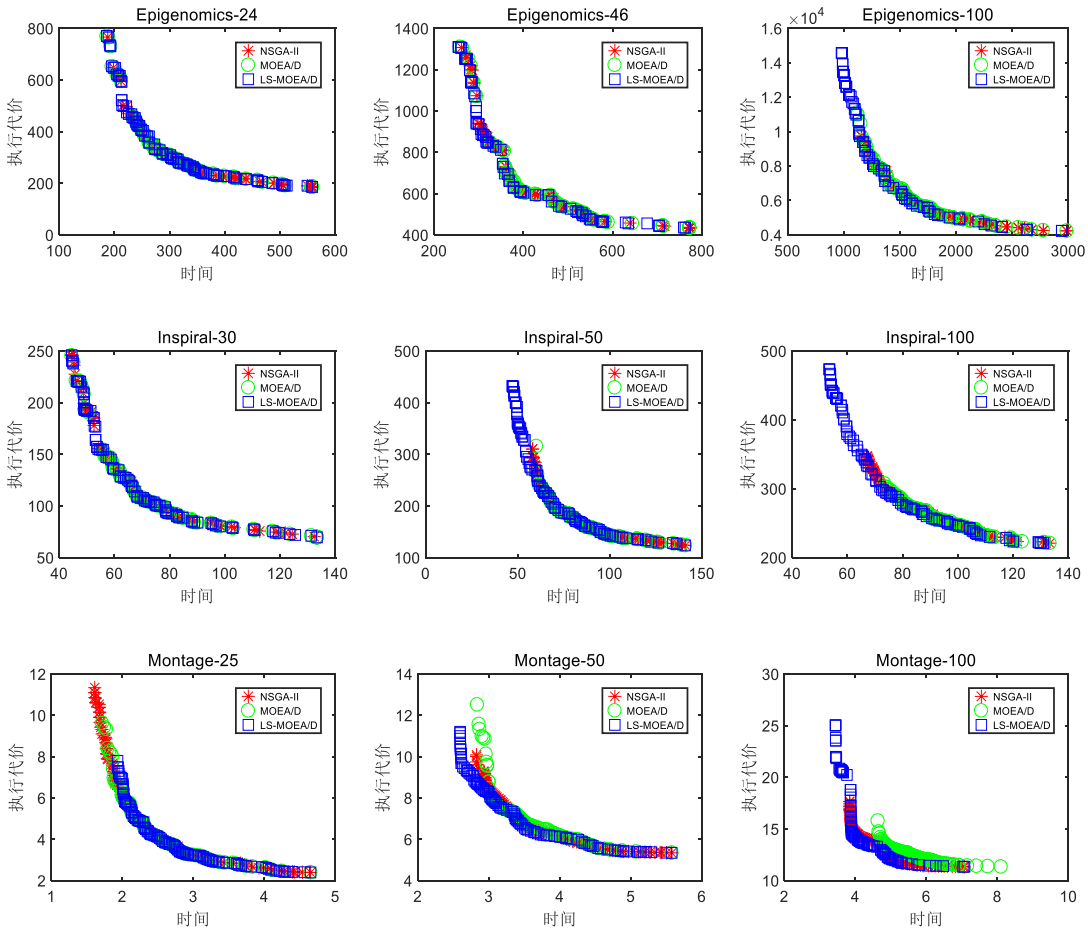


Fig.5 PF results of last generation population with ET and EC as scheduling objectives

6. Conclusion and future work

This paper establishes a cloud workflow scheduling model based on completion time and execution cost, then proposes a MOEA/D algorithm based on local search and weight vector adjustment, whose results show that in solving the cloud workflow scheduling model based on completion time and execution cost. Firstly, by deeply analyzing the cloud workflow scheduling process and its characteristics, the cloud workflow scheduling model based on completion time and execution cost is established. And followed by this, MOEA/D algorithm based on local search and weight vector adjustment is proposed and applied to cloud workflow scheduling problem. The experimental results show that the proposed algorithm has better effect than MOEA/D algorithm and NSGA-II algorithm for most actual workflow scheduling schemes. At the same time, a group of uniformly distributed Pareto dominant solutions are obtained, which can effectively provide decision support for cloud workflow scheduling problem.

In the future research of workflow scheduling problem, we can add the scheduling scheme of traditional heuristic method to the initialization population on the basis of this research. Meanwhile, considering that this paper adopts the random crossover and mutation evolutionary operator, we can use the heuristic crossover and mutation operator to accelerate the convergence of the algorithm in the future.

Ethics Approval and Consent to participate

Not applicable

Consent for publication

Not applicable

Availability of data and material

The labeled dataset used to support the findings of this study are available from the corresponding author upon request.

Compering interests

The authors declare that they have no competing interests.

Funding

Key Scientific Research Project Plan of Henan Provincial Department of Education (20A120012).

Authors' contributions

Xue Li-Yao as the primary contributor, completed the analysis, experiments and paper writing.

Acknowledgements

Thank you Yang Tao for his contributions to the layout and language of the article.

References

[1] Mell P, Grance T. The NIST definition of cloud computing[J]. 2011.

[2] Cloud computing: Principles and paradigms[M]. John Wiley & Sons, 2010.

[3] Bharathi S, Chervenak A, Deelman E, et al. Characterization of scientific workflows[C]//Workflows in Support of Large-Scale Science, 2008. WORKS 2008. Third Workshop on. IEEE, 2008: 1-10.

[4] Yildiz U, Guabtni A, Ngu A H H. Business versus scientific workflows: A comparative study[C]//Services-I, 2009 World Conference on. IEEE, 2009: 340-343.

[5] Barker A, Van Hemert J. Scientific workflow: a survey and research directions[C]//International Conference on Parallel Processing and Applied Mathematics. Springer Berlin Heidelberg, 2007: 746-753.

[6] Ullman J D. NP-complete scheduling problems[J]. Journal of Computer and System sciences, 1975, 10(3): 384-393.

[7] Wu F, Wu Q, Tan Y. Workflow scheduling in cloud: a survey[J]. The Journal of Supercomputing, 2015, 71(9): 3373-3418.

[8] Deb K, Sindhya K, Hakanen J. Multi-objective optimization[M]//Decision Sciences: Theory and Practice. CRC Press, 2016: 145-184.

[9] Fogel D B. The Advantages of Evolutionary Computation[C]//BCEC. 1997: 1-11.

[10] Wu Z, Liu X, Ni Z, et al. A market-oriented hierarchical scheduling strategy in cloud workflow systems[J]. The Journal of Supercomputing, 2013: 1-38.

[11] Liu L, Zhang M, Buyya R, et al. Deadline‐constrained coevolutionary genetic algorithm for scientific workflow scheduling in cloud computing[J]. Concurrency and Computation: Practice and Experience, 2017, 29(5).

[12] Tirapat T, Udomkasemsub O, Li X, et al. Cost optimization for scientific workflow execution on cloud computing[C]//Parallel and Distributed Systems (ICPADS), 2013 International Conference on. IEEE, 2013: 663-668.
[13] Mosleh M A S, Radhamani G, Hazber M A G, et al. Adaptive Cost-Based Task Scheduling in Cloud Environment[J]. Scientific Programming, 2016, 2016.
[14] Rodriguez M A, Buyya R. Deadline based resource provisioningand scheduling algorithm for scientific workflows on clouds[J]. IEEE Transactions on Cloud Computing, 2014, 2(2): 222-235.
[15] Calheiros R N, Buyya R. Meeting deadlines of scientific workflows in public clouds with tasks replication[J]. IEEE Transactions on Parallel and Distributed Systems, 2014, 25(7): 1787-1796.
[16] Feller E, Rilling L, Morin C. Energy-aware ant colony based workload placement in clouds[C]//Proceedings of the 2011 IEEE/ACM 12th International Conference on Grid Computing. IEEE Computer Society, 2011: 26-33.
[17] Lin C, Lu S. Scheduling scientific workflows elastically for cloud computing[C]//Cloud Computing (CLOUD), 2011 IEEE International Conference on. IEEE, 2011: 746-747.
[18] Durillo J J, Fard H M, Prodan R. Moheft: A multi-objective list-based method for workflow scheduling[C]//Cloud Computing Technology and Science (CloudCom), 2012 IEEE 4th International Conference on. IEEE, 2012: 185-192.
[19] Zhu Z, Zhang G, Li M, et al. Evolutionary multi-objective workflow scheduling in cloud[J]. IEEE Transactions on Parallel and Distributed Systems, 2016, 27(5): 1344-1357.
[20] Wang X, Yeo C S, Buyya R, et al. Optimizing the makespan and reliability for workflow applications with reputation and a look-ahead genetic algorithm[J]. Future Generation Computer Systems, 2011, 27(8): 1124-1134.
[21] Fard H M, Prodan R, Barrionuevo J J D, et al. A multi-objective approach for workflow scheduling in heterogeneous environments[C]//Proceedings of the 2012 12th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing (ccgrid 2012). IEEE Computer Society, 2012: 300-309.
[22] Wu F, Wu Q, Tan Y, et al. Unified Multi-constraint and Multi-objective Workflow Scheduling for Cloud System[C]//International Conference on Algorithms and Architectures for Parallel Processing. Springer International Publishing, 2015: 635-650.
[23] Padmaveni K, Aravindhar J. Memetic Algorithm for Multi-objective Workflow Scheduling In Cloud[J]. Indian Journal of Science and Technology, 2016, 9(45).
[24] Pandey S, Wu L, Guru S M, et al. A particle swarm optimization-based heuristic for scheduling workflow applications in cloud computing environments[C]//Advanced information networking and applications (AINA), 2010 24th IEEE international conference on. IEEE, 2010: 400-407.
[25] Bilgaiyan S, Sagnika S, Das M. A multi-objective cat swarm optimization algorithm for workflow scheduling in cloud computing environment[M]//Intelligent Computing, Communication and Devices. Springer India, 2015: 73-84.
[26] Yuan D, Yang Y, Liu X, et al. A cost-effective strategy for intermediate data storage in scientific cloud workflow systems[C]//Parallel & Distributed Processing (IPDPS), 2010 IEEE International Symposium on. IEEE, 2010: 1-12.
[27] Duan R, Prodan R, Li X. Multi-objective game theoretic schedulingof bag-of-tasks workflows on hybrid clouds[J]. IEEE transactions on cloud computing, 2014, 2(1): 29-42.
[28] Schaffer J D. Multiple objective optimization with vector evaluated genetic algorithms[C]//Proceedings of the 1st international Conference on Genetic Algorithms. L. Erlbaum Associates Inc., 1985: 93-100.
[29] Google Apps[Online]. Available: https://gsuite.google.com/ (10 February 2006).

[30] Fonseca C M, Fleming P J. Genetic Algorithms for Multiobjective Optimization: Formulation Discussion and Generalization[C]//Icga. 1993, 93(July): 416-423.
[31] Srinivas N, Deb K. Muiltiobjective optimization using nondominated sorting in genetic algorithms[J]. Evolutionary computation, 1994, 2(3): 221-248.
[32] Erickson M, Mayer A, Horn J. Multi-objective optimal design of groundwater remediation systems: application of the niched Pareto genetic algorithm (NPGA)[J]. Advances in Water Resources, 2002, 25(1): 51-65.
[33] Zitzler E, Thiele L. Multiobjective evolutionary algorithms: a comparative case study and the strength Pareto approach[J]. IEEE transactions on Evolutionary Computation, 1999, 3(4): 257-271.
[34] Zitzler E, Künzli S. Indicator-based selection in multiobjective search[C]//International Conference on Parallel Problem Solving from Nature. Springer Berlin Heidelberg, 2004: 832-842.
[35] Knowles J D, Corne D W. Approximating the nondominated front using the Pareto archived evolution strategy[J]. Evolutionary computation, 2000, 8(2): 149-172.
[36] Deb K, Pratap A, Agarwal S, et al. A fast and elitist multiobjective genetic algorithm: NSGA-II[J]. IEEE transactions on evolutionary computation, 2002, 6(2): 182-197.
[37] Ziztler E, Laumanns M, Thiele L. SPEA2: Improving the strength Pareto evolutionary algorithm for multiobjective optimization[J]. Evolutionary Methods for Design, Optimization, and Control, 2002: 95-100.
[38] Zhang Q, Li H. MOEA/D: A multiobjective evolutionary algorithm based on decomposition[J]. IEEE Transactions on evolutionary computation, 2007, 11(6): 712-731.
[39] Dai C, Lei X. An improvement decomposition-based multi-objective evolutionary algorithm with uniform design[J]. Knowledge-Based Systems, 2017, 125: 108-115.
[40] Wang R, Zhang Q, Zhang T. Decomposition-Based Algorithms Using Pareto Adaptive Scalarizing Methods[J]. IEEE Transactions on Evolutionary Computation, 2016, 20(6): 821-837.
[41] Wu M, Li K, Kwong S, et al. Adaptive Two-Level Matching-Based Selection for Decomposition Multi-Objective Optimization[J]. IEEE Transactions on Evolutionary Computation, 2017.
[42] Cai X, Li Y, Fan Z, et al. An external archive guided multiobjective evolutionary algorithm based on decomposition for combinatorial optimization[J]. IEEE Transactions on Evolutionary Computation, 2015, 19(4): 508-523.