A Dynamic Resource Allocation Algorithm in Cloud Computing Based on Workflow and Resource Clustering

Qinghong Shang

School of Physics and Electronic Engineering, Sichuan Normal University, China
shangqinghong@qq.com

Abstract

Since the complexity of large-scale and scientific computation, workflow has been used for task decomposition in cloud computing. A dynamic resource allocation algorithm based on workflow and resource clustering is proposed in this paper. The workflow is described by a directed acyclic graph, which represents the precedence relations and communication cost of subtasks. Fussy clustering algorithm is used to group nodes by the features, which includes the computing capability, transmission capability, storage capacity, cost and reliability. Subtasks are mapped to different resource in cluster by multi-objective optimization model. Resource reputation is added to feature vector by feedback mechanism for evaluating resource reliability. Simulation results show the algorithm reduces the completion time and cost. It improves the resource utilization and load balance.

Keywords: Cloud computing, Resource allocation, Workflow, Resource clustering

1 Introduction

With the development of artificial intelligence, big data and machine learning, cloud computing and its application are infiltrated into various fields [1-3]. The aim of cloud computing is to share heterogeneous and dynamic resource on remote computing nodes. Computing service and application can be used like electricity, water and gas for user in cloud computing. Cloud users gained data, software, hardware and bandwidth as they needed.

As the increasing of user requirements, it is an important issue to allocate the dynamic resource to the user efficiently. Since the complexity of large-scale and scientific computation, cloud workflow has been used for task decomposition [4]. Cloud workflow takes advantage of cloud computing and workflow mechanism. Most of existing workflow systems are static, while cloud workflow provides flexible configuration and virtualization definition. Scheduling workflow is crucial for reducing running time and cost in cloud computing.

In this paper we introduce an efficient methodology based on cloud workflow and resource clustering. The task is split into subtasks which compose workflow under precedence relations. The multi-objective optimization model is used to improve efficiency of workflow. The resource are clustered by computation capability and other multidimensional features which are described by feature vectors. The resource reliability is considered by the feedback of nodes' past behaviors. All resource features and mensuration are considered for resource selection. Simulation results show the algorithm reduces the completion time and cost. The main contribution of this paper is as bellows:

(1) Proposing a multi-objective optimization model to optimize the makespan and task cost.

(2) Using fuzzy clustering algorithm to divide resource into groups, which reduces the searching scope. Resource reputation is added to feature vector by feedback mechanism for evaluating resource reliability.

Section 2 introduces the related work. Section 3 shows the task scheduling model. Section 4 describes the resource clustering and selection method. Section 5 and 6 show the related performance results and conclusion.

2 Related Work

Resource allocation is an important component of cloud computing, which focuses on mapping the users' requests to system resource. Especially in large distributed system, the allocation and scheduling algorithm have significant influence on the system efficiency and load balance. However it is a kind of NP (Non-deterministic Polynomial) complete problem [5]. [6] proposed a greedy algorithm called min-min algorithm, which allocates task to the resource with the least running time. But it has a bad load balance since many tasks are sent to the better resource. Max-min algorithm is similar to min-min algorithm, which calculates the earliest completion time for each tasks on any available machine [7]. The large task with

*Corresponding Author: Qinghong Shang; E-mail: shangqinghong@qq.com
DOI: 10.3966/160792642021032202015

biggest earliest completion time is scheduled to the available machine. Long tasks have higher priority to be scheduled. HEFT (Heterogeneous Earliest Finish Time) focuses on minimizing earliest finish time of task with an insertion-based approach [8]. Sufferage algorithm computes the difference between minimum and second-smallest completion time of tasks, which called sufferage value [9]. The task is allocated to the resource with maximum sufferage. The above algorithms use execution time as a single objective optimization. The algorithms cannot satisfy the high QoS requirements for cloud users such as economic costs. [10] introduced a double objective and constrain method in IaaS clouds, which included budget and deadline constraints. But it didn't consider the data storage and transfer costs. [11] proposed a communication and storage-aware multi-objective algorithm that optimized the execution time and economic cost. In order to meet the high QoS requirements we discussed a dynamic scheduling algorithm with multi-objective optimization in this paper. Using fuzzy clustering to preprocess cloud resources reduces the searching time and cost. Also the resource reputation mensuration increases the system reliability and improves the load balance.

2.1 Workflow

The concept of workflow originated in the field of office automation. Task is decomposed into subtasks, which are monitored and executed by predefined rules. Workflow design includes topology, presentation, scheduling mechanism and decision model. Workflow is used to construct and manage grid applications in grid computing [12-13]. The combination of workflow and cloud computing has also become a research hotspot. [14] introduced a method to map the basic elements of a real workflow to entities by TOSCA for scientific workflows. It enabled workflow definitions that are portable across clouds, resulting in the greater reusability and reproducibility of workflows. [15] proposed a multi-model framework for workflow resource monitoring, prediction and adaptation in order to avoid resource shortage. [16] integrated a workflow engine called HyperFlow with a cloud platform PaaSage, so that workflow application can be deployed across multiple clouds easily.

2.2 Fussy Cluster Analysis

In the traditional clustering method, each object to be identified is strictly divided into a certain class. Fuzzy clustering is to divide data into various categories with a certain probability. Since the heterogeneity and uncertainty in cloud computing, it is difficult to describe and cluster resource exactly. There is a big gap between different resource in performance and type. We need to consider the similarities between different nodes to make them in the same cluster with higher similarity or correlation. [17] proposed a method to optimize grid resource allocation by fuzzy clustering with application preference. It is more focused on grid applications. [18] introduced an improved fuzzy clustering algorithm to group resource and analyzed the response time and average cost. It takes the assignment performance into account, but doesn't concern the relationship between tasks. [19] discussed a scalable machine learning library Fuzzy K-mean clustering which runs on Hadoop. In order to reduce resource searching time and improve load balance, resource with similar features and performance are grouped. It improved the mapping accuracy between user requests and resource providers.

2.3 Resource Reputation

Reliability problem is essential for cloud computing since resources are heterogeneous, virtualized, scalable and dynamic. Most researches use the ratio of successful tasks to total requests to measure the reliability. [20] proposed a method to forecast reliability by history data of task failure rate and resource utilization. [21] combined the resource management and reputation management to increase efficiency for collaborative cloud computing. It emphasize on evaluating node reputation with specific resource type. Our scheduling model is based on resource clustering. The reputation is considered as one of feature vector, so that the measurement of reputation is more specific. It is not an overall assessment for a node, but a reliability measurement for a particular resource type.

3 Task Scheduling Model

We proposed a multiple layer model for the resource allocation problem. As the Figure 1 shown, the lowest layer is the resource pool which provides computational and other service. Resources are heterogeneous and dynamic. They are clustered by fuzzy clustering model in layer 2 according to the similarity of resource. Multi-workflows are layered and split into subtasks. They are put into task queue and sorted by comprehensive score mentioned in section 3.2. The tasks are allocated to different clusters by user preference or system decision.

3.1 Workflow Model

Workflow is described by a directed acyclic graph defined in definition 1 and shown in Figure 2.

Definition 1: Workflow graph (WFG) is a directed acyclic graph, which defines the precedence relation of each subtasks. WFG = $\{T, E\}$ consists a set of tasks $T = \{t_1, t_2, \ldots, t_n\}$ and a set of edges $E = \{e_{ij}\}$. For each e_{ij} in E means the task i and j has precedence relation. The weight of edge is valued by the communication cost between two tasks.

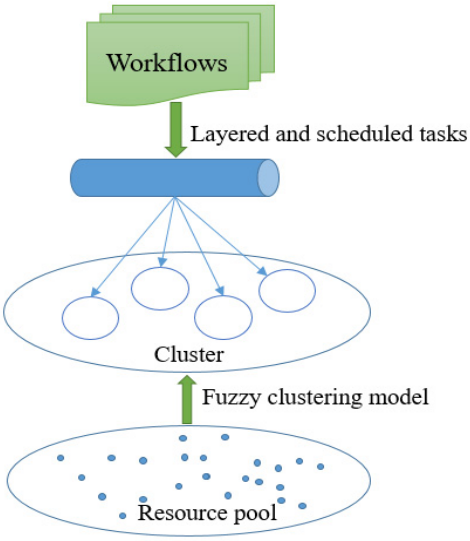


Figure 1. Workflows scheduling model

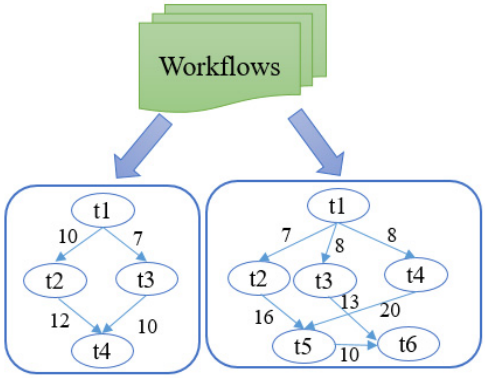


Figure 2. Workflow representation

Definition 2: Task t_i in a workflow is described as a set $t_i = \{TIMt_i, COSt_i, PREt_i, SUCt_i, PRIt_i, UPRt_i\}$. $TIMt_i$, and $COSt_i$ represent the average estimated execution time and communication cost. $PREt_i$ and $SUCt_i$ are the set of precursor and succeed tasks. $PRIt_i$ is the priority level of task t_i. User preferences are described by $UPRt_i$.

Definition 3: System resource is defined as a set $S = \{s_i | s_i = (COPs_i, COMs_i, Ms_i, Ts_i, COSs_i)\}$. $COPs_i$ and $COMs_i$ measure the computing and transmission capability of resource s_i. Ms_i represents the storage capacity and Ts_i describes the reputation of s_i. $COSs_i$ is the unit cost of resource s_i which used to compute the economic cost of each task.

The matrix X is used to map tasks to resource. X is defined as bellows:

$$X = \begin{pmatrix} X_{11} & \cdots & X_{1m} \\ \vdots & \ddots & \vdots \\ X_{n1} & \cdots & X_{nm} \end{pmatrix}$$

$\forall x_{ij} \in X$, if task t_i is allocated to resource s_j, $x_{ij} = 1$, otherwise $x_{ij} = 0$.

Let STT_{ij} equals the start time of task t_i on resource s_j. $PREt_i$ is the set of precursor tasks of t_i. $\forall t_k \in PREt_i$, $EDTt_k$ is the end time of precursor and $COTt_k$ is the communication time of precursor. ST_{ij} is the searching time for task t_i to resource s_j. RT_{ij} means the execution time of t_i on resource s_j. The end time of t_i EDT_{ij} can be calculated by STT_{ij} and RT_{ij}.

$$STT_{ij} = \max_{tk \in PREti} \{EDTt_k + COTt_k\} + ST_{ij} \quad (1)$$

$$EDT_{ij} = STT_{ij} + RT_{ij} \quad (2)$$

COS_T is the total cost of workflow T. EDT_T is the latest finish time of T.

$$COS_T = \sum_{i=1}^{n} \sum_{j=1}^{m} RT_{ij} COSs_j x_{ij} \quad (3)$$

$$EDT_T = \max_{ti \in T} \{EDT_{ij}\} \quad (4)$$

Parameter D is the longest completed time user can accept for a particular workflow, and C is maximum budget cost users are willing to pay for this workflow. The multi-objective optimization model is defined as bellows:

$$\left.\begin{array}{ll} Min & COS_T \\ Min & EDT_T \\ s.t. & COS_T \le C \\ & EDT_T \le D \end{array}\right\} \quad (5)$$

Formula 5 shows our objective is to minimize latest finish time and total task cost with constraint of parameter C and D. The scheduling results are dynamic based on the user preferences. We can optimize the finish time by increasing the weight of CPU performance. Also we can optimize the task cost by increasing the weight of cost parameter which are introduced in section 4.1. In addition to the workflow scheduling algorithm, resource searching time ST_{ij} is an important factor as formula 1, 2 and 4 shows. We use resource clustering and resource reputation mechanism to improve the resource searching and mapping problem which are introduced in section 4.

3.2 Task Scheduling

As above mentioned, we use $PRIt_i$ to measure the priority of task. According to the HEFT algorithm $PRIt_i$ is calculated by the execution time of t_i and communication cost between t_i and its successors. $TIMt_i$ represents the estimated execution time for task t_i. Here it can be treated as the average time of t_i running on different resource. In formula 6, $avg(f(e_{ij}))$ is the average communication cost between task t_i and its successors.

$$PRIt_i = TIMt_i + \max_{tj \in SUCti} \{avg(f(e_{ij})) + PRIt_j\} \quad (6)$$

The calculation of $PRIt_i$ is to traverse the workflow graph upward from the exit task. For the exit task

$SUCt_i = \varnothing$, according to the formula 6 $PRIt_i$ is as bellows:

$$PRIt_i = TIMt_i \tag{7}$$

The task scheduling algorithm is shown in algorithm 1. For each $t_i \in T$, if it is the exit subtask of a workflow, $PRIt_i$ is calculated by formula 7 according to the line 3 and 4 in algorithm 1. Otherwise $PRIt_i$ is calculated by formula 6 according to the line 5 and 6 in algorithm 1. Queue (s_i) is the task queue of resource s_i. Each task is put into different queue by $UPRt_i$ which is defined in definition 2. $UPRt_i$ is associate with resource type and is explained in section 4.

Algorithm 1. Task scheduling algorithm

Input: a set of tasks from workflow
Output: task scheduling sequence
Process:
1. $T = \{t_1, t_2, ..., t_n\}$
2. for each task $t_i \in T$
3. if $t_i ==$ exit
4. $PRIt_i = TIMt_i$
5. else
6. $PRIt_i = TIMt_i + \max_{tj \in SUCti} \{avg(f(e_{ij})) + PRIt_j\}$
7. mark and record the level of t_i
8. put t_i into Queue (s_k) according to $UPRt_i$
9. end for
10. schedule tasks in each Queue (s_k) by $PRIt_i$ and level

Competition exists between different workflows. If we only use the value of $PRIt_i$ to determine which task has higher priority, the rest of previously arrived task may not be scheduled for a long time. Our scheduling algorithm can also apply to multi-workflows. For each $t_i \in T$, the level of t_i is defined as the path length from beginning to t_i in workflow graph. When each t_i is put into Queue (s_k), its level l_i is recorded. When tasks from another workflow arrived, they are merged into the unfinished level of previous workflow. Figure 3 shows the directed acyclic graphs for three workflows.

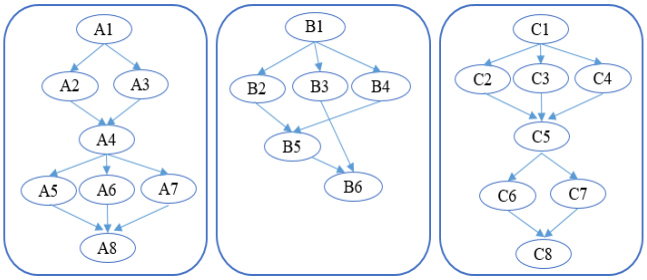
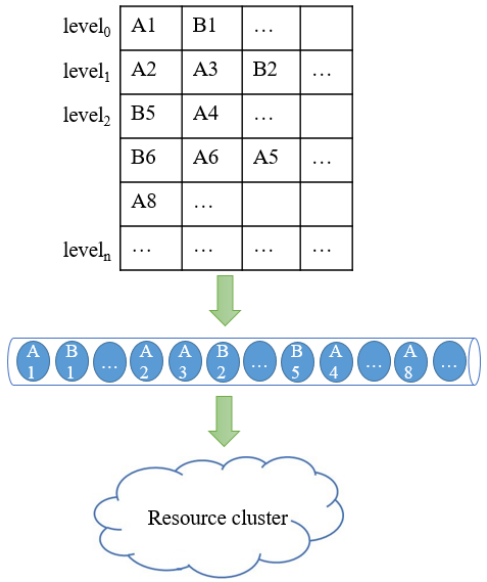
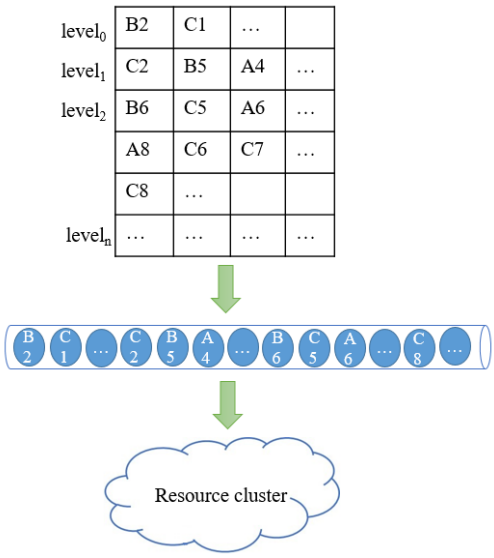


Figure 3. Directed acyclic graphs for multi-workflows

As Figure 4(a) shown, at the beginning tasks from workflow A and B are scheduled to a resource cluster. The order of these tasks are based on l_i and $PRIt_i$. Figure 4(b) shows when tasks from a new workflow C arrived, the original tasks A1, B1, A2 and A3 have already finished. C1 is put into the first unfinished level.



(a) two workflows



(b) three workflows

Figure 4. Multi-workflow scheduling model

4 Resource Clustering

4.1 Fuzzy Clustering Model

As mentioned above, tasks from multi-workflows are scheduled to a resource cluster by mapping $UPRt_i$ and resource features. The resource is defined as $S = \{s_i | s_i = (COPs_i, COMs_i, Ms_i, Ts_i, COSs_i)\}$. We measured the computing capability, transmission capability, memory, reputation and cost for each s_i. $\forall s_{im} \in S$, it is the m-th characteristic index for resource s_i.

Since the different dimensions of characteristic index, the first step of fuzzy clustering is data standardization gained by formula 8. $\overline{s_j}$ is the mean and STD_j is the standard deviation. s'_{ij} is the standardized value calculated by $\overline{s_j}$ and STD_j. s'_{jmin} is the minimum in $\{ s'_{1j}, s'_{2j}, \ldots, s'_{nj} \}$. s'_{jmax} is the maximum.

$$\left.\begin{aligned} \overline{s_j} &= (\sum_{i=1}^{n} s_{ij}) / n \\ STD_j &= \sqrt{\frac{1}{n}\sum_{i=1}^{n}(s_{ij} - \overline{s_j})} \\ s'_{ij} &= (s_{ij} - \overline{s_j}) / STD_j \\ s''_{ij} &= (s'_{ij} - s'_{jmin}) / (s'_{jmax} - s'_{jmin}) \end{aligned}\right\} \quad \textbf{(8)}$$

We use the method in formula 9 to establish the similar matrix P. $\forall p_{ij} \in P$, it represents the similarity between s_i and s_j. Then the transitive closure P^* is obtained by composition operation of similar coefficient matrix.

$$\left.\begin{aligned} p_{ij} &= \frac{1}{m}\sum_{k=1}^{m} \exp\left[-\frac{3}{4}\left(\frac{s''_{ik} - s''_{jk}}{STD''_k} \right)^2 \right] \\ STD''_k &= \sqrt{\frac{1}{n}\sum_{k=1}^{n}(s''_{ik} - \overline{s''_k})} \end{aligned}\right\} \quad \textbf{(9)}$$

Let $\lambda \in [0,1]$, the set S is clustered into different ways according to the value of λ. It is a threshold to control the similarity degree of clusters. We define $t(P)_\lambda = P^* - X_\lambda$. All elements in the matrix X_λ have the same value λ. Elements greater than 0 in $t(P)_\lambda$ are clustered into the same group.

We evaluate the resource by its comprehensive score. The score is obtained by formula 10. α_j is the weight of resource requirement. It can be adjusted according to the user preference $UPRt_i$.

$$score_i = \sum_{j=1}^{m} \alpha_j * s''_{ij} \quad \textbf{(10)}$$

The score of a cluster reflects the average performance. It can be calculated by the following formula 11. The cluster with higher $Cscore_k$ has priority to be scheduled. Also in a specified cluster, the resource is scheduled by $score_i$.

$$Cscore_k = \frac{1}{n_k} \sum_{s_i \in Cluster_k} \sum_{j=1}^{m} \alpha_j * s''_{ij} \quad \textbf{(11)}$$

4.2 Resource Allocation

Resource clustering reduces the blindness and randomness of resource selection. Tasks are allocated to appropriate cluster by user preference such as CPU-intensive, cost-intensive and transmission-intensive. According to formula 1, ST_{ij} is the searching time for task t_i to resource s_j. Clustering mechanism reduces the searching time and narrows the search scope. Tasks are allocated to a cluster according to $UPRt_i$ and $Cscore_k$.

In formula 12, $avg(RT_{ti})$ is the average runtime of task t_i on all resources. The parameter $ratio_{ti}$ is the ratio of average runtime of t_i to average runtime of all unstarted tasks. TR_{ti} is the remaining time for subsequent tasks calculated in formula 13. $EDTt_k$ is the end time of precursor and $COTt_k$ is the communication time of precursor. TW_{ti} is the maximum waiting time in formula 14. If resource s_j is busy and the waiting time exceeds TW_{ti}, the task t_i will be rescheduled. The resource allocation algorithm is shown in algorithm 2.

$$ratio_{ti} = avg(RT_{ti}) / \sum_{tj \in U} avg(RT_{tj}) \quad \textbf{(12)}$$

$$TR_{ti} = D - \max_{tk \in PREti} \{EDTt_k + COTt_k\} \quad \textbf{(13)}$$

$$TW_{ti} = TR_{ti} \cdot ratio_{ti} - RT_{ij} \quad \textbf{(14)}$$

Algorithm 2. Resource allocation algorithm

Input: a set of tasks from workflow

Output: resource allocation plan and scheduling queues

Scheduling process:

1. $T = \{t_1, t_2, \ldots, t_n\}$
2. for each task $t_i \in T$
3. put t_i into the queue of Cluster C_k according to $UPRt_i$ and $Cscore_k$
4. choose resource s_j with best $score_j$ in C_k
5. put t_i into the queue of s_j
6. calculate TW_{ti} according to formula 14
7. end for

Monitoring process:

1. while true
2. for each task t_i in the queue of s_j
3. if waiting time of $t_i > TW_{ti}$
4. notify and reschedule t_i
5. negative evaluation for s_j
6. end for
7. end while

The resources are clustered into different groups such as CPU-intensive, cost-intensive and transmission-intensive. In the line 3 of the scheduling process, the task t_i is allocated to different clusters according to its user preference. Users can choose time-first or cost-first. The $Cscore_k$ measures the comprehensive score of a cluster. The cluster with higher $Cscore_k$ has priority to be chosen. Meanwhile the $UPRt_i$ decides the priorities of tasks which are introduced in section 3.2. In the line 4 the task is sent to the resource with highest score which is calculated in formula 10. The task request is put into the queue of the selected resource and waits to be executed.

In the line 3 of the monitoring process, if the waiting time has already exceeded the maximum waiting time TW_{ti} calculated by the formula 14, it means the chosen resource is busy or out of order and cannot finish the task within the specified time. In the line 4 the task t_i is rescheduled to another resource by the scheduling process. In the line 5, it gives s_j a negative feedback if s_j cannot satisfy the requirement of a task. In the definition 3 Ts_i describes the reputation of s_i. The negative feedback reduces the value of Ts_j in definition 3 and leads to decreasing of its comprehensive score in formula 10. Contrarily, if a task runs successfully on resource s_j, it gives s_j a positive feedback. As formula 15 shown, let T_0 equals the basic score. For each task t_i allocated to resource s_j, it gives a feedback f_{ij}. Ts_j reflects the reputation of resource s_j. The Ts_j is dynamic, which improves the load balance of cluster.

$$Ts_j = T_0 + \sum f_{ij} \tag{15}$$

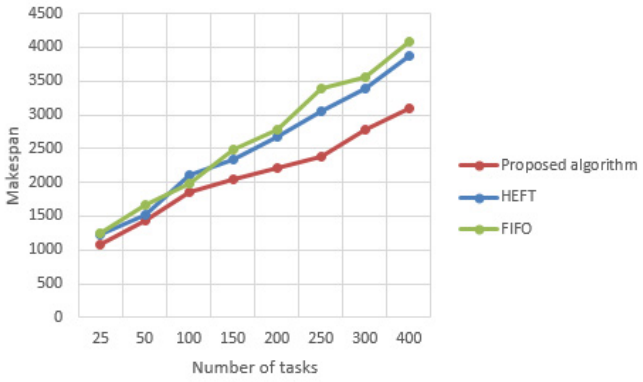
5 Experiment and Analysis

Our experiment and simulation are based on CloudSim [22], which is a toolkit for modeling and simulation of cloud computing environments and evaluation of resource provisioning algorithms. We simulate workflow scheduling based on resource cluster. Clustering mechanism reduces the searching time and narrows the search scope. It provides more precise mapping between tasks and virtual machines. Resources are clustered by features such as computing, transmission and storage capability. The parameters of virtual machines are defined in table 1.

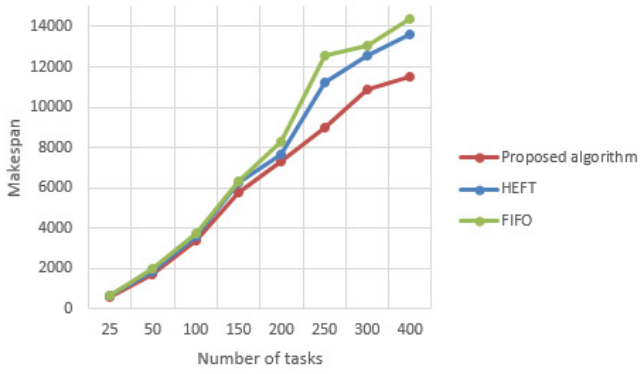
Table 1. Parameters of virtual machines

Parameters	Range
CPU number	1-4
CPU performance (MIPS)	250-1000
Storage capacity (GB)	512-4096
Bandwidth (Gb/S)	0.5-10
Cost ($/h)	1-4

The scientific workflows used in the experiments are Montage and LIGO [23-24]. Montage developed by NASA/IPAC is used to generate custom mosaics of the sky using input images. LIGO is a network of gravitational wave detectors. We set the task number n = {25, 50, 100, 150, 200, 250, 300, 400} and the resource number r = {20, 40, 60, 80, 100}. The parameters of resource are random in range of table 1. As mentioned above, fuzzy clustering algorithm reduces the searching time and narrows the searching scope. Also it improves the FIFO algorithm by multi-queues according to the resource clustering. In order to show the advantage of cluster in resource selection, we compare our algorithm with HEFT and FIFO algorithms in makespan and cost. As Figure 5 shown, the algorithm based on resource clustering we proposed has the minimal makespan. Especially when the number of task increased, the advantage is more obvious. Take Montage workflow as an example, when the task number equals 400 in Figure 5(a), the makespan of our algorithm is reduced by approximately 20% compared with HEFT algorithm and 24% compared with FIFO algorithm. As Figure 5(b) shown, the makespan of our algorithm is also better than the other two algorithms for LIGO workflow. It gains an improvement of 15% and 20% compared to HEFT and FIFO algorithms for the LIGO-400. The clustering method reduces search scope and time cost by mapping the user preference and resource features.



(a) Montage workflow

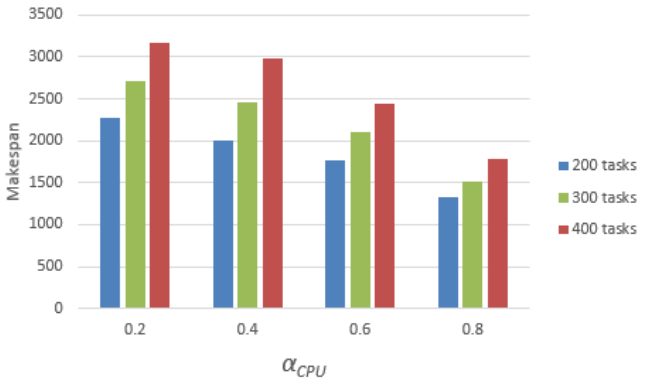


(b) LIGO workflow

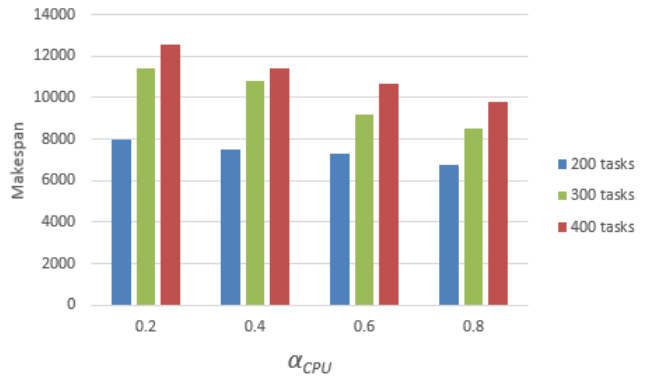
Figure 5. Makespan of three algorithms for Montage and LIGO

In formula 10, α_j is defined as the weight of resource requirement. It can be adjusted according to the user preference $UPRt_i$. We set the value of α_{CPU} from 0.2 to 0.8, which means tasks require massive calculation and CPU performance is the preferred choice. The task number is 200, 300 and 400. As Figure 6 shown, with the increasing of α_{CPU}, the makespan decreases for both Montage and LIGO. Take Montage-400 as an example, the makespan is reduced approximately 43% when the value of α_{CPU} is increased from 0.2 to 0.8. For LIGO-400 the makespan

is reduced 22% when the value of α_{CPU} is increased from 0.2 to 0.8.



(a)Montage workflow



(b) LIGO workflow

Figure 6. Makespan of different preference

Similarly, for cost-intensive task we can increase the value of α_{Cost} to decrease cost as Figure 7 shown. α_{Cost} affects the proportion of cost in measuring node comprehensive scores. Take Montage-400 as an example, the cost is reduced approximately 31% when the_value of α_{Cost} is increased from 0.2 to 0.8. As Figure 6 and Figure 7 shown, we can optimize the time and cost parameters in formula 5 by different user preferences.

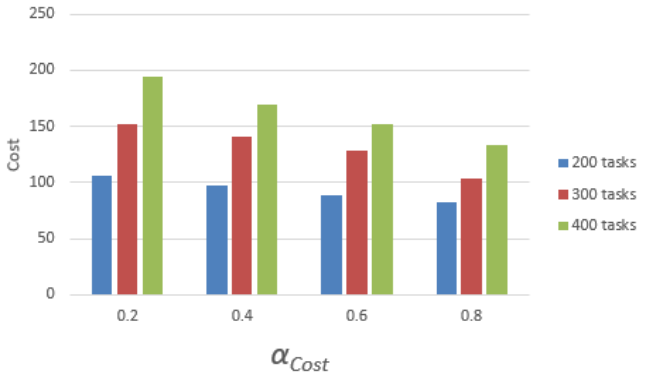


Figure 7. Cost of different preference

Speedup is the ratio of serial execution time to parallel execution time in formula 16. It is a measurement of algorithm performance. We compare the speedup of proposed algorithm with HEFT and FIFO algorithms for Montage workflow. Figure 8 shows our algorithm has better speedup. For example, when the task number equals 400, the average speedup of our algorithm is increased by approximately 11% compared with HEFT algorithm and 15% compared with FIFO algorithm.

$$Speedup = \frac{\min(\sum_{sj \in S} \sum_{ti \in T} RT_{ij})}{makespan} \quad \textbf{(16)}$$

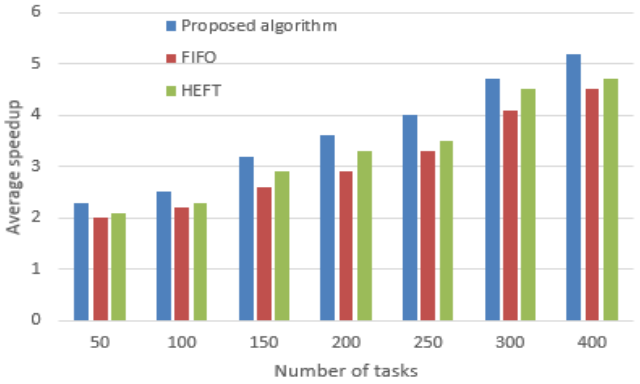


Figure 8. Average speedup of three algorithms

Figure 9 shows the resource utilization of proposed algorithm and the other two algorithms for the different scaled of LIGO workflow. As the number of tasks increasing, the resource utilization increases at the beginning. Then it decreases when task number exceeds 400, because the increasing of data and communication cost affects the resource utilization. In general, the average resource utilization of proposed algorithm is better than HEFT and FIFO algorithms. Take LIGO-300 as an example, it gains an improvement of 10% and 19% compared to the HEFT and FIFO algorithms.

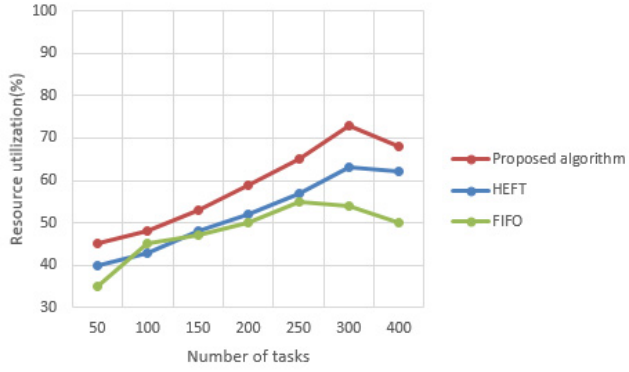


Figure 9. Resource utilization of three algorithms

6 Conclusion

In this paper we proposed a dynamic resource

allocation algorithm based on workflow and resource clustering. Since the heterogeneity and uncertainty in cloud computing, fussy clustering algorithm is used to describe the correlation and similarity between resource nodes. It reduces the blindness and randomness of resource selection. Subtasks are mapped to different resource in cluster by multi-objective optimization model. Tasks can get different scheduling results according to different preference factors. Simulation results show the proposed algorithm could decrease the makespan of tasks and increase resource utilization compared to HEFT and FIFO algorithms. Next, we will consider the problem of security scheduling by protecting users' privacy.

Acknowledgments

This work was supported by the Fund Project of Sichuan Provincial Department of Education (No. 15ZB0036).

References

[1] D. Lorencik, P. Sincak, Cloud robotics: Current trends and possible use as a service, *2013 IEEE 11th International Symposium on Applied Machine Intelligence and Informatics (SAMI)*, Herl'any, Slovakia, 2013, pp. 85-88.

[2] Y. Kim, E. Huh, Towards the Design of a System and a Workflow Model for Medical Big Data Processing in the Hybrid Cloud, *2017 IEEE 15th Intl Conf on Dependable, Autonomic and Secure Computing, 15th Intl Conf on Pervasive Intelligence and Computing, 3rd Intl Conf on Big Data Intelligence and Computing and Cyber Science and Technology Congress*, Orlando, FL, USA, 2017, pp. 1288-1291.

[3] T. N. B. Duong, N. Q. Sang, Distributed Machine Learning on IAAS Clouds, *2018 5th IEEE International Conference on Cloud Computing and Intelligence Systems (CCIS)*, Nanjing, China, 2018, pp. 58-62.

[4] N. Mohanapriya, G. Kousalya, Execution of workflow applications on cloud middleware, *2017 International Conference on Innovations in Information, Embedded and Communication Systems (ICIIECS)*, Coimbatore, India, 2017, pp. 1-6.

[5] M. Li, Y. Sun, H. Huang, J. Cui, A Flexible Resource Allocation Mechanism with Performance Guarantee in Cloud Computing, *2018 4th International Conference on Big Data Computing and Communications (BIGCOM)*, Chicago, IL, USA, 2018, pp. 181-188.

[6] M. Derakhshan, Z. Bateni, Optimization of tasks in cloud computing based on MAX-MIN, MIN-MIN and priority, *2018 4th International Conference on Web Research*, Tehran, Iran, 2018, pp. 45-50.

[7] S. Devipriya, C. Ramesh, Improved Max-min heuristic model for task scheduling in cloud, *2013 International Conference on Green Computing, Communication and Conservation of Energy*, Chennai, India, 2013, pp. 883-888.

[8] H. Topcuoglu, S. Hariri, M.-Y. Wu, Performance-effective and low-complexity task scheduling for heterogeneous computing, *IEEE Transactions on Parallel and Distributed Systems*, Vol. 13, No. 3, pp. 260-274, March, 2002.

[9] N. M. Reda, An Improved Sufferage Meta-Task Scheduling Algorithm in Grid Computing Systems, *International Journal of Advanced Research*, Vol. 3, No. 10, pp. 123-129, October, 2015.

[10] M. Malawski, G. Juve, E. Deelman, J. Nabrzyski, Cost- and deadline-constrained provisioning for scientific workflow ensembles in IaaS clouds, *Proceedings of the International Conference on High Performance Computing, Networking, Storage and Analysis*, Salt Lake City, UT, USA, 2012, pp. 1-11.

[11] R. Duan, R. Prodan, X. Li, Multi-Objective Game Theoretic Schedulingof Bag-of-Tasks Workflows on Hybrid Clouds, *IEEE Transactions on Cloud Computing*, Vol. 2, No. 1, pp. 29-42, January-March, 2014.

[12] S. Zhang, N. Gu, S. Li, Grid workflow based on dynamic modeling and scheduling, *International Conference on Information Technology: Coding and Computing*, Las Vegas, NV, USA, 2004, pp. 35-39.

[13] K. Kim, A Workflow Knowledge Grid/P2P Architecture for Massively Parallel and Very Large Scale Workflows, *2005 First International Conference on Semantics, Knowledge and Grid*, Beijing, China, 2005, pp. 7-7.

[14] R. Qasha, J. Cala, P. Watson, Towards Automated Workflow Deployment in the Cloud Using TOSCA, *2015 IEEE 8th International Conference on Cloud Computing*, New York, NY, USA, 2015, pp. 1037-1040.

[15] H. El-Kassabi, M. A. Serhani, R. Dssouli, N. Al-Qirim, I. Taleb, Cloud Workflow Resource Shortage Prediction and Fulfillment Using Multiple Adaptation Strategies, *2018 IEEE 11th International Conference on Cloud Computing*, San Francisco, CA, USA, 2018, pp. 974-977.

[16] M. Malawski, B. Balis, K. Figiela, M. Pawlik, M. Bubak, Support for Scientific Workflows in a Model-Based Cloud Platform, *2015 IEEE/ACM 8th International Conference on Utility and Cloud Computing*, Limassol, Cyprus, 2015, pp. 412-413.

[17] D. Sun, G. Chang, L. Jin, X. Wang, Optimizing grid resource allocation by combining fuzzy clustering with application preference, *Proceedings of the 2nd IEEE International Conference on Advanced Computer Control*, Shenyang, China, 2010, pp. 22-27.

[18] X. Wang, Y. Wang, Z. Hao, J. Du, The Research on Resource Scheduling Based on Fuzzy Clustering in Cloud Computing, *2015 8th International Conference on Intelligent Computation Technology and Automation*, Nanchang, China, 2015, pp. 1025-1028.

[19] D. Garg, K. Trivedi, Fuzzy K-mean clustering in MapReduce on cloud based hadoop, *2014 IEEE International Conference on Advanced Communications, Control and Computing Technologies*, Ramanathapuram, India, 2014, pp. 1607-1610.

[20] Y. Alsenani, G. V. Crosby, T. Velasco, A. Alahmadi, ReMot

Reputation and Resource-Based Model to Estimate the Reliability of the Host Machines in Volunteer Cloud Environment, *2018 IEEE 6th International Conference on Future Internet of Things and Cloud*, Barcelona, Spain, 2018, pp. 63-70.

[21] H. Shen, G. Liu, An Efficient and Trustworthy Resource Sharing Platform for Collaborative Cloud Computing, *IEEE Transactions on Parallel and Distributed Systems*, Vol. 25, No. 4, pp. 862-875, April, 2014.

[22] R. Buyya, R. Ranjan, R. N. Calheiros, Modeling and simulation of scalable Cloud computing environments and the CloudSim toolkit: Challenges and opportunities, *2009 International Conference on High Performance Computing & Simulation*, Leipzig, Germany, 2009, pp. 1-11.

[23] S. Bharathi, A. Chervenak, E. Deelman, G. Mehta, M. Su, K. Vahi, Characterization of scientific workflows, *2008 Third Workshop on Workflows in Support of Large-Scale Science*, Austin, TX, USA, 2008, pp. 1-10.

[24] G. Juve, A. Chervenak, E. Deelman, S. Bharathi, G. Mehta, K. Vahi, Characterizing and profiling scientific workflows, *Future Generation Computer Systems*, Vol. 29, No. 3, pp. 682-692, March, 2013.

Biography

Qinghong Shang received her Ph.D. degree in 2013 from the University of Electronic Science and Technology of China. She is currently a lecturer in the School of Physics and Electronic Engineering, Sichuan Normal University.