Liu Xiaotian
Master of Engineering; Lecturer in School of Mechanical and Electric Engineering, *orcid.org/0000-0003-2408-3114*
School of Mechanical and Electric Engineering, Yancheng Polytechnic College, Jiangsu Yancheng 224005, P.R. China
Gu Daming
Master of Engineering; Associate professor in School of Mechanical and Electric Engineering, *orcid.org/0000-0002-5158-4128*
School of Mechanical and Electric Engineering, Yancheng Polytechnic College, Jiangsu Yancheng 224005, P.R. China

CLOUD COMPUTING WORKFLOW SCHEDULING WITH MAXIMUM REDUCTION OF EFFECTIVE RESOURCES

Abstract. *In order to solve the problem that resource utilization efficiency of large-scale scientific workflow is low in cloud computing environment, this paper proposes an algorithm of maximum effective resource reduction (MERR). The algorithm is mainly implemented in three steps. First, identify the delay limitation, finding the balance between the reduction of effective resource use and the increase of time. Second, task merge and merge the tasks with low resource utilization in original workflow scheduling. The third is resource consolidation, a best fitting method is adopted to combine resources that are not fully used, so as to improve the efficiency of resource utilization. Using CyberShake, Epigenomics, LIGO and Montage four kinds of scientific workflow to carry out simulation experiments. The results show that MERR has reduced using resources by 54%, and the average time increase is less than 10%, which are better than scheduling algorithm based on the critical path.*

Keywords: Workflow; Scheduling; Cloud computing; Resource utilization; Resource allocation

Problem statement

With the increase of resource capacity, the popularization of large-scale multi-core system cloud computing and the development of virtual technology, a large amount of resource information has been widely applied to improve large applications (such as enterprise data center and cloud computing system, etc.) of the performance. Due to the good scalability of workflow [1], it has certain advantages for the application of a large number of resources, which has attracted great attention. However, most of the workflows exchange for application performance improvement with lower resource utilization [2].

General resource scheduling algorithms focus on improving the utilization performance of limited resources [3], while the emergence of multi-core processors and cloud computing has aroused people's attention to resource utilization [4]. Existing most resource scheduling algorithms may be adapted to handle large amounts of resources by limiting resource Numbers on schedule, but only for part or specific problems [5].

Mao et al. [6] illustrated that dynamic resource allocation in the public cloud is generally one of two choices. However, resource utilization from different layers of workflows and inhomogeneous widths is low, this kind of situation for a period of time (usually 1 hour) still exist. Lee et al. [7] proved that the resource usage of workflow scheduling can be reduced by merging idle tasks. Among many scheduling algorithms, the scheduling algorithm based on key path [8-10] is quite popular, and they are also commonly used to minimize time.

Xie et al. [8] proposed a Dynamic Critical path Scheduling (DCS) algorithm based on process sets, which transformed multiple tree structures into a virtual processing tree.

Then, a Critical Path & Task Duplicating (CPTD) algorithm based on multi-core processor is proposed [9], tasks are assigned to multiple parallel computer cores and optimized to reduce total completion time.

Lee et al. [10] proposed a Critical Path First (CPF) algorithm to obtain the Critical Path by processing the tree. However, its resource utilization is relatively low, and for distributed resources, the core of a scheduler is a scheduling algorithm.

Considering the efficiency of resource utilization in running large-scale scientific workflow, in this paper, a MERR (Maximum Effective Resource Reduction) algorithm is proposed to reduce the increasing time of resource consumption. This algorithm is a post-optimization algorithm. Compared with the input scheduling resources, this algorithm regards the tasks in the existing workflow scheduling and combined scheduling algorithms as small resources, and seeks the minimum time increase to reduce the resource consumption. The main work summarized as follows:

1) MERR extends many workflow scheduling algorithms by allowing time to increase or delay to maximize resource reduction;

2) The main innovation of MERR is that it can find a balance between increasing time and reducing resource consumption, which can improve resource utilization, optimize resource allocation and reduce energy consumption.

Basic material

1. The problems existing in optimize workflow scheduling algorithm

This section mainly describes the optimization of workflow scheduling algorithms for workflow and system models.

1.1. Scientific workflow

The scientific workflow consists of the priority constraint task data set represented by the directed acyclic graph(DAG)[11], G=(V,E) contains the task set V, $V=\{v_0,v_1,...,v_n\}$,and the edge set E, The edge connecting two tasks indicates that they have a precedence constraint or data dependency, as shown in Fig. 1.

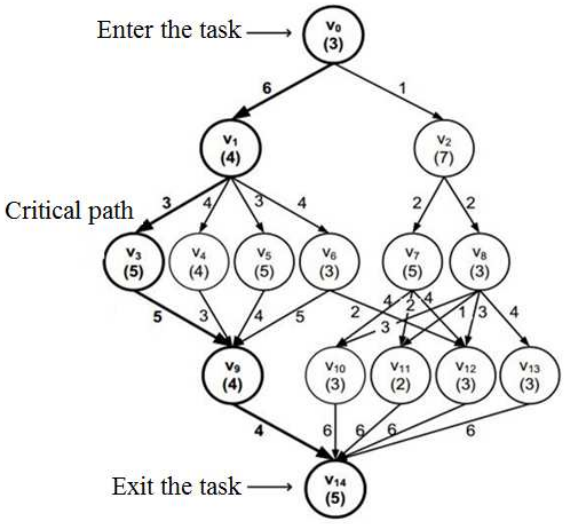


Figure 1 – Simple workflow diagram

In most cases, a task we can see as a ready to run tasks (or as preparation task). According to the last task (parent task) of the task v_i, it is determined that it is ready to complete the communication in the shortest time. In general, the earliest start time and completion time of task v_i are defined as follows:

$$EST(v_i)=\begin{Bmatrix} 0, & if\ \ v_i=v_{entry} \\ \max_{v_p\in P_i\{EST(v_p)+w_p+c_{p,i}\}} & otherwise \end{Bmatrix} \quad (1)$$

$$EFT(v_i)=EST(v_i)+w_i. \quad (2)$$

In the formula, v_{entry} is the root task without any parent task, P_i is the parent task set of v_i, w_i and w_p respectively represent the computational cost (i.e. execution time) of v_i and v_p, and $c_{p,i}$ is the communication cost from v_p to v_i.

The latest start time and completion time of v_i are defined as follows:

$$LST(v_i)=LFT(v_i)-w_i \quad (3)$$

$$LFT(v_i)=\begin{Bmatrix} EFT(v_i) & if\ v_i=v_{exit} \\ \min_{v_c\in C_i}\{LST(v_c)-c_{i,c}\} & otherwise \end{Bmatrix} \quad (4)$$

In the formula, C_i is the subtask of v_i, and v_{exit} represents the quit task.

The actual starting time and completion time of task v_i are respectively expressed as $AST(v_i)$ and $AFT(v_i)$, and if the actual completion time of other tasks scheduled for the same resource is less than $EST(v_i)$, then $AST(v_i)$ and $AFT(v_i)$ from the earliest start time and completion time are obtained. $ALST(v_i)$ and $ALFT(v_i)$ are defined in the same way,the LFT (or ALFT) of v_i is usually expressed as the last cut-off time of the task. Therefore, the delay beyond this cut-off increases the total time. Maximum delay is defined as the difference between LFT and EFT(or ALFT and AFT); The delay value is the synchronization requirement for subtask v_i.

1.2. The target system

The target system of this paper consists of the same computing resource R, $R=\{R_0,R_1,...,R_m\}$, resources may be physical computing nodes or a virtual machine, each resource in R consists of a set of p processing elements or (virtual) processor cores, that is $R_i=\{r_{i,0},r_{i,1},...,r_{i,p}\}$. This paper assumes that the cost of communication between single resource tasks can be ignored,based on the nuclear number of communications costs the cost of calculating power and costs, assuming the resources are the same.

1.3. Formulaic problem

It is assumed that workflow scheduling S^0 is the output of scheduling algorithm for given scientific workflow G and resource set R. This scheduling algorithm can be described as Gantt chart. Output scheduling is an execution task plan in G, a subset resource R0 from R, and contains three tuple sets. Every three tuple collection consists of task v_i, resource $r_{j,k}$ and $AST(v_i)$. The scheduling task of resource $r_{j,k}$ is expressed as $V_{j,k}$. The total resource time for resource $r_{j,k}$ is defined as the task execution time in $V_{j,k}$, and it is expressed by $RT(r_{j,k})$.

In order to find a combined scheduling S^* and an original output scheduling S^0, and at the same time minimize the increase of time, the optimization problem of workflow scheduling needs to be solved. However, when time delays are considered, the optimization of workflow scheduling becomes more complex. In particular, in view of the task completion time delay (usually caused by merging) could spread to more recursive task, not only including the previous task, but also contains the task scheduling tasks after the merger. In particular, in view of the task completion time delay (usually caused by merging) could spread to more recursive task, not only including the previous task, but also contains the task scheduling tasks after the merger.

The optimal goal of workflow scheduling is to find a balance between time delays and reduced resource usage.

For this reason, the Effective Reduction of Effective resources can be used in this paper to determine the actual Reduction in the use of time resources and the Increase in Makespan (MI). Effective Reduction (ER) of synthetic scheduling is defined as follows:

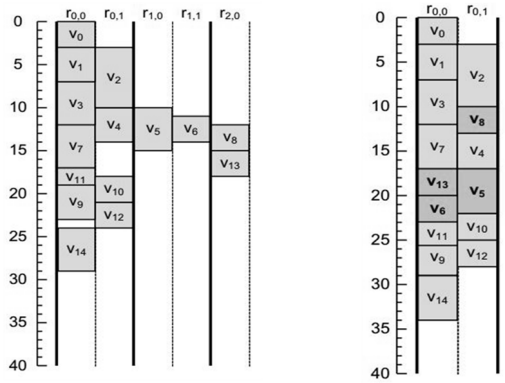
$$ER = \frac{\left(\left|R^0\right| - \left|R^*\right|\right)}{\left|R^0\right|} - \frac{\left(ms^* - ms^0\right)}{ms^0} \qquad (5)$$

In the formula, $\left|R^*\right|$ is the number of resources used for synthetic scheduling, ms^0 is the input scheduling time, and ms^* is the time of synthetic scheduling.

2. The algorithm of maximum effective resource reduction (MERR)

MERR is an optimization technology of workflow scheduling algorithm, and its optimal time increase and resource decrease can be balanced. MERR consists of the following stages or sub-algorithms: (1) delay constraint recognition; (2) task consolidation; (3) resource consolidation.

In essence, MERR combines two methods to find the optimal workflow scheduling algorithm: (1) fill the idle time according to the data correlation between tasks; (2) extrusion task. In general, consider merging only a small number of resource tasks. If the translation improves the validity of the resource, the task moves to another resource, allowing the delay time to adjust the degree of consolidation by modifying. In particular, MERR integrates within the original completion time by pushing down one or more tasks within the delay limit, as shown in figure Fig. 2 (b).



a *b*

Figure 2 – Optimize workflow scheduling:
a – use CPF to generate the original schedule;
b – use MERR to optimize the scheduling

2.1. Delay limitation identification

For a given input schedule, MERR merges the schedule and allows a specific time increment. Therefore, identifying the minimum increase is the key to minimizing the use of resources. The solution to this problem is the delay - limiting recognition algorithm, as shown in algorithm 1. The basic principle of this algorithm is to combine the total use time of small resources.

Algorithm 1. Delay limitation identification

1 $R' \leftarrow$ Resources stored according to ascending RT
2 According to RT group R'
3 srcnt 0
4 $ER^{\max} \leftarrow 0$
5 for $R'_i \in \left(R' - R'_{|R'|}\right)$ do
6 srcnt $\leftarrow$ srcnt+$\left|R'_i\right|$
7 $ms' \leftarrow ms^0 + RT'_i$
8 trcnt $\leftarrow \left|R^0\right|$ -srcnt
9 if srcnt > trcnt then
10 $srcnt' \leftarrow \left(\left\lceil \frac{srcnt}{trcnt} \right\rceil - 1\right) \square trcnt$
11 while $srcnt' > 0$ do
12 in R', $ms' \leftarrow ms^0 + srcnt'\ th\ \ RT$
13 $srcnt' \leftarrow srcnt' - trcnt$
14 $MI \leftarrow \frac{\left(ms' - ms^0\right)}{ms^0}$
15 $RUR \leftarrow srcnt / \left|R^0\right|$
16 if $ER > ER^{\max}$ then
17 $ER^{\max} \leftarrow ER$
18 $d^{\lim it} \leftarrow MI$

The resource stores (R0) used by the input scheduling are stored in ascending order and grouped by total resource usage time (Resource Time, RT) [12].

To identify the maximum effective reduction, the outer loop (line 5) iterates through each stored resource group. For specific resources (line 6 R_i'), if the resource has been merged, the total use time of the resource for the task is a better indicator of the maximum increase time. Since this article deals with resource groups, it is possible to effectively reduce the amount of resource usage that has been merged in any target resource, as well as the previous resource group (line 6). The number of resources is used to consolidate tasks that deal with other resources, so the reduction in resource usage (Reduction in Resource Usage, RUR) is determined through *srcnt* (line 15).

In Fig. 2(a), from R_0^0 to R_2^0, the RT of the three resources is 45, 8 and 6, respectively. Calculate RT for a particular resource by summing the execution time of all tasks, For example, the RT value of R_0^0 in figure 2(a) represents the total execution time of 11 tasks through two cores, namely $v_0, v_1, ..., v_{14}$ in $r_{0,0}$ and $v_2, v_4, ..., v_{12}$ in $r_{0,1}$. RT is used because in the worst case scenario, all these 11 tasks are combined into one core. As shown in Fig. 2, the resource order of RT is R_2^0, R_1^0 and R_0^0; the actual order after the sorting is R_0' and R_1' followed by, except for R_2' (that is, line 5, $R' - R'_{|R'|}$). In the first iteration, *srcnt*, *trcnt*, RT_i' and ms' were 1, 2, 6 and 35, respectively. Then, MI, RUR and ER were 0.21(21%), 0.33(33%) and 0.12(12%), respectively. If *srcnt* is greater than *trcnt* (line 9), the RT determination time of the recursive resource can be increased. In Fig. 2(a), in such calculations, the original resource is the second and the first. Then, $ms' = 29 + 6 + 8$, *srcnt* = 2, MI, RUR and ER are 0.48, 0.67 and 0.19 respectively. Therefore, the scheduling process is shown in Fig. 2(b). The ER value is 0.19 as a delay constraint because the optimal scheduling time in Fig. 2(b) increases. At the end of each iteration, the maximum effective resource ($ER^{\max}$) is compared with the current ER to identify the final $ER^{\max}$, which is the time delay limit ($d^{\max}$) for the task merge algorithm/phase.

2.2. Merge task

In fact, the integration of resource subset is the integration of tasks [13], as shown in algorithm 2. The increase in the completion time is obtained by integrating the time delay limit and the remaining resources, as shown in line 1 of algorithm 2, Input scheduling (S^0) use of resources (R^0) first reverse order allocation, because for most resources, given scheduling resource usage rate decreased gradually. Then consider Each task ($v_{i,j}^*$) of merging resources (r_i^*) and the task of other resources. The resource of the merge task ($v_{i,j}^*$) causes the minimum time to increase, and insert/merge $v_{i,j}^*$ to resources (line 9). Where, the FindMinMISlot function is used to determine how much time has increased.

Algorithm 2. Merge task

1 $R^* \leftarrow R^0$ Reverse ordering
2 $S^* \leftarrow S^0$
3 for $r_i^* \in R^*$ do
4 $R' \leftarrow R^* - r_i^*$
5 for $v_{i,j}^* \in V_i^*$ do
6 for $r_k' \in R'$ do
7 $AST\left(v_{i,j}^*, r_k'\right) \leftarrow FindMinMISlot\left(v_{i,j}^*, r'\right)$
8 if $AST\left(v_{i,j}^*, r_k'\right) \neq \infty$ then
9 insert $v_{i,j}^*$ into $AST\left(v_{i,j}^*, r_k'\right)$
10 update the schedule $\left(S^*\right)$
11 $d^{\lim it} \leftarrow d^{\lim it} - mi^{\min}$
12 break
13 if $r_i^* == \varnothing$ then
14 $R^* \leftarrow R^* - r_i^*$

After merging specific tasks, update scheduling and time delay limits (lines 10 and 11). The delay of the merge task is greater than any subsequent task or the LST of subsequent tasks (delay propagation), update scheduling (line 10) can handle a large number of tasks. More than LST delay will lead to the increase of time. In particular, task scheduling data is updated by tasks that are merged, so that the recognition process occurs in subsequent tasks cycle of the merged task. Once all tasks in a particular resource have been merged and the resource already does not exist after the consolidated task (algorithm 2, line 13), the task is removed.

2.3. Resource merger

Since resources can be regarded as service nodes or resource / virtual instance [14] of cloud computing terms, there are one or more processing units. In this sense, some of the resources used for input scheduling may not be fully used, and one or more cores may not be allocated to any task. Algorithm 3 combines these resources in an optimal fit method to deal with this problem. Merging resources helps further improve resource efficiency because the process reduces the amount of resources used for final output scheduling. Sort the resources in ascending order according to the number of cores used (algorithm 3, line 1),in order to improve the efficiency of the algorithm, this paper considers the task of changing resources and reduces the number of used cores, thereby further using resources in large amounts (algorithm 3, line 3). Then consider merging a small set of used resources (R'') and a large set of used resources (R').

Once a small number of used resources (r'') cannot be merged, the resource merge process is interrupted (algorithm 3, line 10), because there are no other small number of used resources in R'' and these resources have the same or more cores.

Algorithm 3. Merge resources

1 $R' \leftarrow$ Sort $R*$ resources in ascending order according to the # kernel used

2 $R' \leftarrow R'$ – Fully used resources

3 $R'' \leftarrow R'$ – Have # resources to use the core> $\frac{total\#cores}{2}$

4 for $r'' \in R''$ do

5 $R' \leftarrow R' - r_i''$

6 for $r_j' \in R'$ (from the latest res) do

7 if unused nucleus of r_j'# >=unused nucleus of r'' # then

8 merge r'' to r_j'

9 break

10 if no merge r'' then

11 break

3. Performance Evaluation and Analysis

3.1. Experiment introduction

In this article, we evaluate four different scheduling algorithms, which are the dynamic critical path dispatching algorithm DCS [8], key path and task replication scheduling algorithm for multi-core processors [9], the critical path priority algorithm CPF [10] and the MERR algorithm in this article, the compared algorithms are all based on critical path scheduling algorithms and are currently the most popular method for solving the minimization time. In order to run an infinite number of resources, this paper modifies the comparison algorithm slightly. The experimental platform used in the Intel Core Duo i3 processor, @ 2.29GHz frequency, 4.0GB RAM desktop, using matlab2011b for experimental simulation, the workflow data used for the experiment is downloaded from the following website: https://confluence.pegasus.isi.edu/display/pegasus/WorkflowGenerator.

This article uses four different resource sizes (Nuclear number: 1, 2, 4 and 8), However, this paper only shows the experimental results of resource size 8, because the experimental results of different resource sizes are similar. The workflow applications are CyberShake, Epigenomics, LIGO and Montage, as shown in Fig. 3, Table 1 gives specific information, the inter-node/resource bandwidth is set to 1Gbps, and each workflow consists of 20 variations of different characteristics, Epigenomics workflow traces include an additional 120 random jobs.

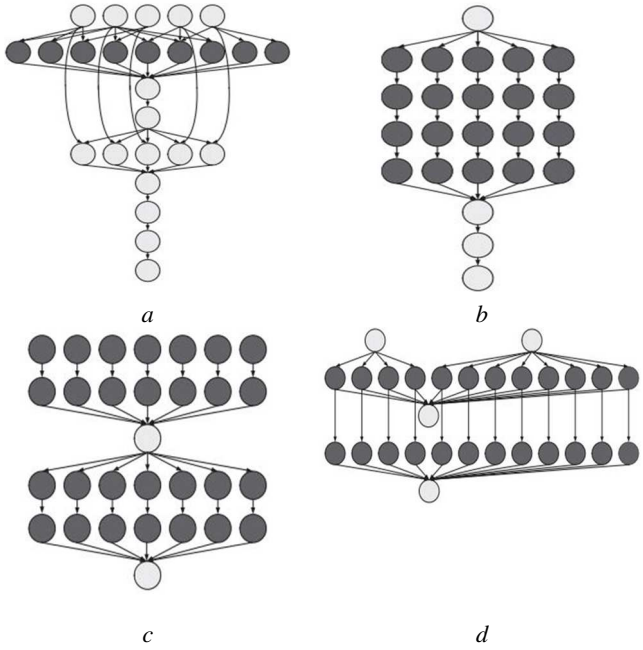


Figure 3 – Scientific workflow:
a – Montag; b – Epigenomics; c – LIGO; d – CyberShake

Table 1 – *Introduction to scientific workflow*

Appli-cation	# The number of jobs in Workflow	# The number of tasks in a job(Workflow size)
Cyber-Shake	220	50&[100,1000], interval 100
Epige-nomics	440	50, [100,1000], interval 100 and{2000,3000,4000,5000,6000}
LIGO	220	50&[100,1000], interval 100
Montage	220	50&[100,1000], interval 100

3.2. Performance analysis

In this paper, experimental results are evaluated according to the increase in completion time (MI), the reduction in resource utilization (RUR), the time delay limit ($d^{\lim it}$) and the final effective reduction amount (ER), as shown in table 2. The description of Table 2 is as follows: For each application, the two numbers in the first line represent the time of the original schedule (ms^0) and the time after the merge (ms^*) respectively, and the second line represent the increase ratio of each application time (that's the value of MI in parentheses), the third line is the number of resources ($\left|R^0\right|$)used by the original schedule and the number of resources ($\left|R^*\right|$) used by the merge schedule, the fourth line is the ratio of the reduced use of resources (RUR in parentheses), and the fifth line represents delay limit ($d^{\lim it}$).

As shown in table 2, the MI average degree of the proposed MERR is 10.5%, and 54% of RUR is obtained, resulting in about 43.7% of ER. As you can see the

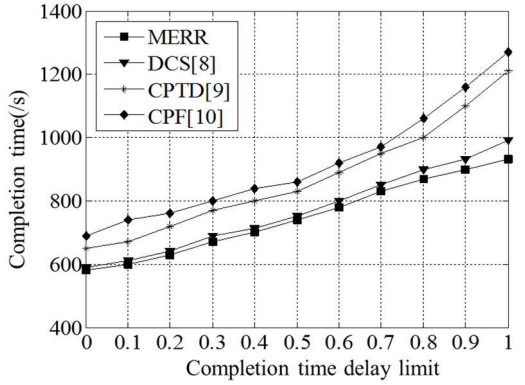
MERR significantly improved the resource utilization of Montage, this is mainly because the workflow structure, specific layer or two layers can run many tasks in parallel, and thus lead to excessive use of resources. ER values are obtained based on their original output scheduling, and the quality of these scheduling algorithms describes the differences between different algorithms.

To further illustrate its effectiveness, this paper further demonstrates by manually setting up experimental data sets with different delay time limits. The result is shown in Fig. 4~ Fig. 7.

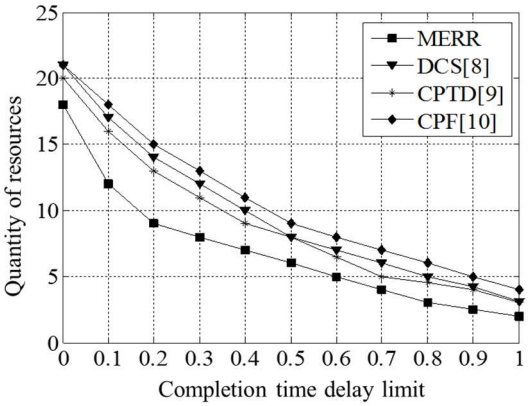
Table 2 – *Comparison of the average performance results of each algorithm*

Application	CPTD[9]	CPF[10]	DCS[8]	MERR
CyberShake	567,683	566,681	567,682	684,806
	(21%)	(23%)	(20%)	(15%)
	21.0,9.5	20.9,9.7	20.8,9.5	18.7,6.5
	(55%)	(54%)	(54%)	(65%)
	24%	24%	24%	23%
Epigenomics	20878	20876	20878	23634
	27108	23577	23083	29703
	(30%)	(13%)	(11%)	(6%)
	40.5,18.7	31.2,24.4	30.6,24.9	40.5,17.0
	(55%)	(22%)	(18%)	(59%)
	39%	18%	14%	36%
LIGO	1398,1398	1398,1401	1398,1398	1420,1420
	(0%)	(0%)	(0%)	(0%)
	16.2,14.2	15.4,13.4	15.6,14.2	16.2,14.0
	(12%)	(13%)	(9%)	(17%)
	2%	1%	0%	0%
Montage	212,241	212,242	212,241	231,255
	(14%)	(17%)	(14%)	(7%)
	42.0,11.3	42.0,11.2	42.0,11.3	41.3,10.9
	(73%)	(73%)	(73%)	(75%)
	18%	18%	18%	17%

For the CyberShake scientific workflow, as shown in Figure 4, by manually setting the delay limit, the MERR completion time is slightly lower than CPTD [9], CPF [10], and DCS[8] as the delay increases, at 600 – 1000 s, and have a low usage of resources. This is because MERR is a post-processing technique and is independent of the scheduling algorithm. CPTD transforms the task graph into the corresponding product processing tree without considering the time consumption of finding the key path in the processing tree, the other two methods are similar.



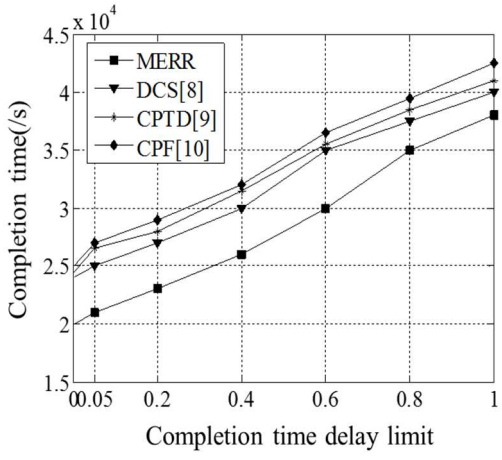
a



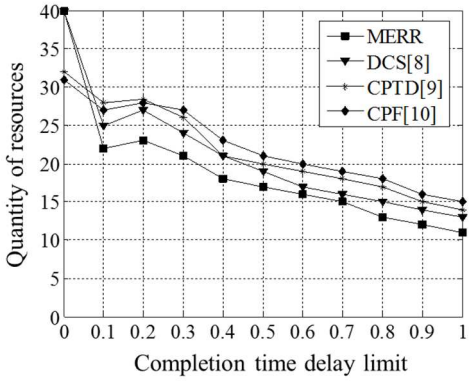
b

Figure 4 – Performance comparison of CyberShake with manually set delay limit: a – Increased time; b – Reduce the amount of resources used

For the Epigenomics scientific workflow, according to the time increase comparison, as shown in Fig.5, the proposed MERR resource usage is significantly higher than the other three algorithms. In addition, the completion time was increased by at least 1000 seconds. In general, poor scheduling quality makes resources ineffective, resulting in many time gaps. These four methods assign sub-deadlines to other tasks based on the critical path task deadline. However, the resource utilization rate of CPTD, CPF and DCS is low, especially for distributed resources.



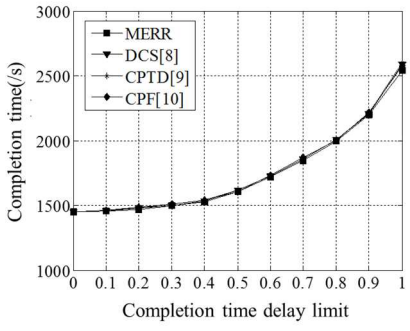
a



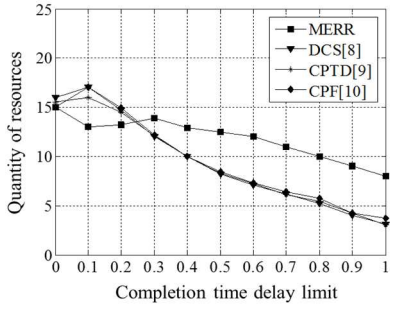
b

Figure 5 – Performance comparison of Epigenomics with manually set delay limit: a – Increased time; b – Reduce the amount of resources used

For the LIGO scientific workflow, it can be seen from Fig. 6 that the completion time curves of the CPTD, CPF, DCS and the proposed MERR are basically coincident, which is determined by the inherent nature of the LIGO workflow, that is, for the LIGO workflow, the scheduling method It does not seem to be a decisive factor. For the Montage scientific workflow, as shown in Fig. 7, the performance improvement is even more pronounced. Although the maximum ER value is obtained according to the specific MI value, the ER value can be obtained without MI value (see Fig. 4~ Fig. 7) .That is, when MI is not required, the average RUR obtained by MERR is 12%. It can be seen from FIG. 4 to FIG. 7 that MI is generally less than RUR, that is, the slope of RUR, especially with small delay limit, is significantly greater than the slope of MI. This is because higher ER values are obtained through MERR.

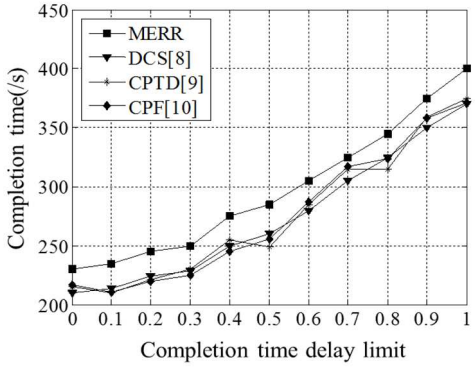


a

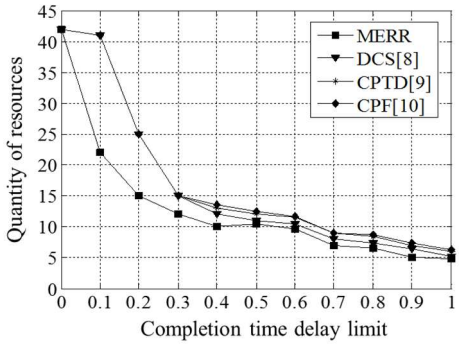


b

Figure 6 – Performance comparison of LIGO with manually set delay limit: a – Increased time; b – Reduce the amount of resources used



a



b

Figure 7 – Performance comparison of Montage with manually set delay limit: a – Increased time; b – Reduce the amount of resources used

3.3. Load problem

In addition to MERR performance, this article also measured its load. On Intel core dual-core i3 processor, @2.29GHz main frequency, 4.0GB RAM desktop, the actual running time of MERR is almost in milliseconds (6 to 58ms), ignoring the workflow type. In fact, the load depends on the degree of the merger. This load level demonstrates MERR's ability to improve resource utilization, and the execution time of scientific workflow is often many hours [15].

Conclusions

This paper proposes a workflow scheduling optimization algorithm, which is the algorithm of maximum effective resource reduction MERR. The proposed algorithm can be used for any existing workflow scheduling, because there is a post-processing technology in the algorithm. It consists of three main stages: first, find the balance between minimum time increase and maximum resource decrease, and improve resource utilization by combining tasks and resources. The validity of the proposed algorithm is demonstrated by the experimental results of four kinds of practical scientific workflow. By allowing smaller time to increase, you can effectively reduce resource usage and improve resource utilization.

Future research focuses on workflow improvement methods that support resource performance analysis and other applications.

References

1. Zhang F, Malluhi Q M, Elsayed T, et al. CloudFlow: A data-aware programming model for cloud workflow applications on modern HPC systems[J]. Future Generation Computer Systems, 2014, 51(4), 98-110.

2. Ganga K, Karthik S. A fault tolerent approach in scientific workflow systems based on cloud computing[C], 2013 International Conference on Pattern Recognition, Informatics and Mobile EngineeringSalem, India: IEEE Press, 2013, 387-390.

3. YUAN Zhengwu, LI Junqi. Cloud Resource Scheduling Based on Improved Particle Swarm Optimization Algorithm[J]. Computer Engineering and Design, 2016, 37(2), 401-404 (in Chinese).

4. HAN Jingjie. Research on Multi-core Task Scheduling Based on the Key Path of Integrated Scheduling[D]. Harbin: Harbin University of Science and Technology, 2014 (in Chinese).

5. CHEN Donglin, YAO Mengdi, LV Qiuyun. Construction of Scheduling System for Cloud Federation Across Multi-data Center[J]. Computer Engineering and Design, 2015, 36(2), 546-550 (in Chinese).

6. Mao M, Humphrey M. Auto-scaling to minimize cost and meet application deadlines in cloud workflows[C]// Conference on High Performance Computing Networking, Storage and Analysis, SC 2011, Seattle, WA, USA, November, 2011. 2011, 1-12.

7. Y.C. Lee, A.Y. Zomaya, Stretch out and compact: workflow scheduling with resource abundance[C].// Proceedings of the International Symposium on Cluster Cloud and the Grid (CCGRID). Delft Netherlands: IEEE Press, , 2013, 219–226.

8. XIE Zhiqiang, YANG Jing, ZHOU Yong, et al. Dynamic Critical Paths Multi-product Manufacturing Scheduling Algorithm Based on Operation Set[J]. Chinese Journal of Computers, 2011, 34(2), 406-412 (in Chinese).

9. XIE Zhiqiang, HAN Yingjie, QI Yonghong, et al. A Scheduling Algorithm for Multi-core Based on Critical Path and Task Duplication[J]. Journal of National University of Defense Technology, 2014, 29(1), 172-177 (in Chinese).

10. Y.C. Lee, H. Han, A.Y. Zomaya, On Resource Efficiency of Workflow Schedules[C]//. Proceedings of the 14th International Conference on Computational Science (ICCS). Guimaraes, Portugal, IEEE Press,, 2014: 534–545.

11. LI Siyun. Matching and Searching of Workflow Model[D]. Shanghai, Shanghai Jiaotong University, 2015 (in Chinese).

12. Zhan Z H, Liu X F, Gong Y J, et al. Cloud Computing Resource Scheduling and a Survey of Its Evolutionary Approaches [J]. Acm Computing Surveys, 2015, 47(4), 1-33.

13. ZHANG Yuan, ZHANG Yunyong, FANG Bingyi. A Kind of OpenStack Auto-Deployment Architecture Based on Workflow[J]. Telecommunications Science, 2014, 30(11), 14-21 (in Chinese).

14. LI Ming, WU Yue, CHEN Jia. Cloud Resource Scheduling Based on Semantic Search Engine[J]. Application Research of Computers, 2015, 32(12), 3735-3737 (in Chinese).

15. Wang S, Du Z. Dispersion cloud resources scheduling based on mobile agent[J]. Journal of Computational Methods in Sciences & Engineering, 2016, 16(2), 303-316.

Стаття надійшла до редколегії 03.10.2018

Рецензент: д-р техн. наук, доц. О.С.Рижков, директор ТОВ "Українсько-китайський центр шовкового шляху", Миколаїв.

Лю Сяотіан
Магістр інженерії; викладач у Школі механіки та електротехніки, *orcid.org/0000-0003-2408-3114*
Школа механіки та електричної інженерії, Янченский політехнічний коледж, Цзянсу Яньчэн 224005, П.Р. Китай
Гу Дамін
Магістр інженерії; доцент кафедри механіки та електротехніки, *orcid.org/0000-0002-5158-4128*
Школа механіки та електричної інженерії, Янченский політехнічний коледж, Цзянсу Яньчэн 224005, П.Р. Китай

ХМАРНЕ ОБЧИСЛЕННЯ РОБОЧОГО ПРОЦЕСУ ПЛАНУВАННЯ З МАКСИМАЛЬНИМ ЗМЕНШЕННЯМ ЕФЕКТИВНИХ РЕСУРСІВ

Анотація. *З метою розв'язання проблеми, що ефективність використання ресурсів великомасштабного наукового робочого процесу є низькою в середовищі обчислень у хмарі, пропонується алгоритм максимально ефективного зниження ресурсів (MERR). Алгоритм в основному реалізується в три етапи. По-перше, визначення обмеження затримки, пошук балансу між зменшенням ефективного використання ресурсів та збільшенням часу. По-друге, злиття та об'єднання завдань з низьким використанням ресурсів в оригінальному плануванні робочого процесу. По-третє, консолідація ресурсів, найбільш відповідний спосіб для поєднання ресурсів, які не використовуються повністю, з метою підвищення ефективності використання ресурсів. Було використано CyberShake, Epigenomics, LIGO і Montage - чотири види наукового робочого процесу для проведення експериментів із моделювання. Результати показують, що MERR зменшила використання ресурсів на 54%, а середній витрати часу підвищуються менше ніж на 10%, що краще, ніж алгоритм планування, заснований на критичному шляху.*

Ключові слова: робочий процес; планування; хмарні обчислення; використання ресурсів; розподілення ресурсів

Link to publication

APA *Liu, Xiaotian & Gu, Damingю (2018). Cloud Computing Workflow Scheduling with Maximum Reduction of Effective Resources. Management of Development of Complex Systems, 36, 63 – 70.*

ДСТУ *Лю Сяотіан. Хмарне обчислення робочого процесу планування з максимальним зменшенням ефективних ресурсів [Текст] / Лю Сяотіан, Гу Дамін // Управління розвитком складних систем. – 2018. – № 36. – С. 63 – 70.*