A Multi-stage Dynamic Game-Theoretic Approach for Multi-Workflow Scheduling on Heterogeneous Virtual Machines from Multiple Infrastructure-as-a-Service Clouds

Yuandou Wang, Jiajia Jiang, Yunni Xia(✉), Quanwang Wu(✉), Xin Luo(✉), and Qingsheng Zhu

College of Computer Science, Chongqing University, Chongqing 400030, China
xiayunni@hotmail.com, wqw@cqu.edu.cn, luoxin21@gmail.com

Abstract. Distributed computing systems such as clouds continue to evolve to support various types of scientific applications, especially scientific workflows, with dependable, consistent, pervasive, and inexpensive access to geographically-distributed computational capabilities. Scheduling multiple workflows on distributed computing systems like Infrastructure-as-a-Service (IaaS) clouds is well recognized as a fundamental NP-complete problem that is critical to meeting various types of Quality-of-Service (QoS) requirements. In this paper, we propose a multi-objective optimization workflow scheduling approach based on dynamic game-theoretic model aiming at reducing workflow make-spans, reducing total cost, and maximizing system fairness in terms of workload distribution among heterogeneous cloud virtual machines (VMs). We conduct extensive case studies as well based on various well-known scientific workflow templates and real-world third-party commercial IaaS clouds. Experimental results clearly suggest that our proposed approach outperform traditional ones by achieving lower workflow make-spans, lower cost, and better system fairness.

Keywords: Dynamic game · Workflow scheduling
Multi-clouds system · Multi-objective optimization · QoS

This work is supported in part by the International Joint Project funded jointly by the Royal Society of the UK and the National Natural Science Foundation of China under grant 61611130209, National Science Foundations of China under grants Nos. 61472051/61702060, the Science Foundation of Chongqing under No. cstc2017jcyjA1276, China Postdoctoral Science Foundation No. 2015M570770, Chongqing Postdoctoral Science special Foundation No. Xm2015078, and Universities Sci-tech Achievements Transformation Project of Chongqing No. KJZH17104, Chongqing grand R&D projects Nos. cstc2017zdcy-zdyf0120 and cstc2017rgzn-zdyf0118.

J. E. Ferreira et al. (Eds.): SCC 2018, LNCS 10969, pp. 137–152, 2018.
https://doi.org/10.1007/978-3-319-94376-3_9

1 Introduction

Recently, various scientific fields employ workflows to analyze large amounts of data and to perform complex simulations and experiments efficiently. A process in such scientific applications can be modeled as a workflow by dividing it into smaller and simpler tasks. These tasks can then be distributed to multiple computing resources [1]. They usually present graphical interfaces to combine different technologies along with efficient methods for using them, and thus increase the efficiency of scientists. They are usually represented as directed graphs with their nodes representing discrete computational components and the edges representing connections along which data and results can communicate among components. They have different types and usually their execution needs computing platforms with different QoS requirements, e.g. most completion time, load balancing, economics.

Recently, cloud computing is recognized as a promising solution and paradigm for providing a flexible, on-demand computing infrastructure over the Internet for large-scale scientific-workflow-based applications. The services that can be provided from the cloud include Software-as-a-Service (SaaS), Platform-as-a-Service (PaaS), and Infrastructure-as-a-Service (IaaS) [2]. SaaS clouds offer web applications/software over the Internet, running on cloud infrastructure. PaaS and SaaS clouds are thus less suitable for scientific workflows than IaaS ones because they mainly offer an environment to design, develop and test web based applications. Instead, IaaS clouds offer an easily accessible, flexible, and scalable infrastructure suitable for the deployment of large-scale scientific workflows based on on-demand and pay-per-use patterns [3].

One of the most challenging NP-complete problems that researchers try to address is how to schedule large-scale scientific applications to distributed and heterogeneous computational nodes, e.g., IaaS clouds, such that quantitative objective functions such as process make-span are optimized, and certain execution constraints such as communication cost and storage requirements are considered and fulfilled. From the end-users perspective, a low make-span is always preferred, whereas from the systems perspective system-level efficiency and fairness are often considered as a good motivation such that the scientific applications and tasks are supposed to be fairly distributed among computational resources in order to avoid hot spots and performance bottle-necks. However, a careful investigation into related work shows that only a few schemes are able to deal with both perspectives, such as optimizing user objectives (e.g., make-span) while fulfilling other constraints, and providing a good fair workload distribution among physical computational resources of clouds.

The primary aim of the paper is therefore to propose a multi-objective scheduling method to address the real-time workflow scheduling problem on multiple IaaS cloud. Specifically, we consider a multi-objective optimization workflow scheduling approach based on dynamic game-theoretic model. It aims at reducing workflow make-spans, reducing cost, and maximizing system fairness in terms of workload distribution among heterogeneous VMs. We conduct extensive case studies as well based on various well-known scientific workflow templates and

Table 1. Notations and description summary

Notations	Description
N	The total number of workflows
n_i	The total number of tasks in workflow
K	The total number of tasks
M	The total number of cloud service providers (CSPs)
m_p	The total number of virtual machines in CSP p
T_{ij}	The j^{th} task of the workflow i
T^l_{option}	The set of the optional task (s) at stage l
VM_{pk}	The k^{th} virtual machine of CSP p
VM	The set of virtual machines ($VM = \{VM_{11}, VM_{12}, \ldots, VM_{pk}\}$)
VM^l_{idle}	The set of the idle virtual machines at stage l
C_{ijpk}	The completion time of T_{ij} on VM_{pk}
t_{ijpks}	The setup time of T_{ij} executed at VM_{pk}
t_{ijpk}	The cutting time of T_{ij} executed at VM_{pk}
ET_{ijpk}	The execution time of T_{ij} executed at VM_{pk}
x_{ijpk}	A Boolean variable indicating whether VM_{pk} is selected for T_{ij}
u_{pk}	The unit-price-per-time of VM_{pk}
w^l_i	The weight matrix of i^{th} player at l^{th} stage
S^l	The strategies set of players in the l^{th} stage of the game tree
S^l_i	The strategy set of i^{th} player in the l^{th} stage of the game tree
$u^l_i(S^l_i)$	The utility function of i^{th} player at l^{th} stage
L	The maximum stage number of a game tree
a^0	The actions combination at l^{th} stage
a^l	The actions combination at l^{th} stage
h^0	The origin history information of game tree ($h^0 = \emptyset$)
h^l	The l^{th} stage history information set of game ($h^l = (a^0, a^1, \ldots, a^{l-1})$)
H^l	The total history information set in l stages ($H^l = \{h^l\}$)
$A_i(H^l)$	The optional action set of i^{th} player in l stages of the game tree

heterogeneous VMs created on real-world third-party commercial IaaS clouds, i.e., Amazon, Tencent, and Ali clouds. Experimental results clearly suggest that our proposed approach outperforms traditional ones by achieving lower workflow make-spans, lower cost, and better system fairness. Table 1 summarized the notations and description.

The paper is structured as follows. In Sect. 2, we review related work. In Sect. 3, we present the formulation for the heterogeneous-VM-based multi-workflow scheduling problem. In Sect. 4, we present the real-time multi-objective scheduling algorithm based on the dynamic game-theoretic model. In Sect. 5, we conduct extensive case studies to validate our proposed approach. This paper concludes in Sect. 6 with a summery.

2 Related Work

Along with rapidly growing data and computational requirements of large-scale workflow applications, scheduling multiple workflows in distributed systems has become a important and challenging research topic. In this section, we briefly cover a part of the important or relevant related work.

2.1 Multi-objective Workflow Scheduling

The optimization model for workflow (single or multiple) scheduling aim at finding tradeoffs among multiple quantitative objectives, e.g., make-span, cost, reliability, energy consumption, security, or load balancing. Extensive efforts, e.g., [4–11] are paid in this direction. Durillo *et al.* [12] proposed tradeoff solutions generated using a multi-objective-heterogeneous-earliest-finish-time (MOHET) algorithm for multi-objective workflow scheduling problem. Yassa *et al.* [13] proposed an approach based on dynamic voltage and frequency scaling (DVFS) technique for multi-objective workflow scheduling in clouds to minimize energy consumption, and introduced a hybrid particle swarm optimization (PSO) algorithm to optimize the scheduling performance.

Many multi-objective evolutionary algorithms have been extended to deal with the multi-objective problems. Khajemohammadi *et al.* [14] proposed a genetic fast workflow scheduling over grid infrastructures. Zhu *et al.* [15] proposed an evolutionary multi-objective optimization (EMO)-based workflow scheduling algorithm with novel schemes for problem-specific encoding and population initialization, fitness evaluation and genetic operators. Chen *et al.* [16] proposed an ant colony optimization (ACO) algorithm to schedule large-scale workflows with make-span and cost. Padmaveni *et al.* [17] introduced a hybrid algorithm called particle swarm memetic (PSM) algorithm make-span and deadline as the optimization objectives.

Recently, the Pareto-optimal methods are frequently employed. It aims at pursuing a set of compromise solutions that represent good approximations to the Pareto-optimal fronts (PFs). For instance, Zheng *et al.* [18] proposed a Pareto-based fruit fly optimization algorithm (PFOA) to solve the task scheduling and resource allocating (TSRA) problem in cloud computing environment. Hou *et al.* [19] studied the Pareto optimization to schedule crude oil operations in a refinery via genetic algorithm. Ebadifard *et al.* [20] introduced a recent heuristic algorithm called black-hole-optimization (BHO) framework for workflow scheduling based on Pareto optimizer algorithm. It allows users to select the best from the proper solution set of candidate scheduling plans.

2.2 Game-Theoretic-Based Scheduling

Game theory models and methodologies are widely applied to the multi-constraint process scheduling on cloud social, economic and resource scheduling problems. Fard *et al.* [21] suggested a novel pricing model and truthful scheduling mechanism to find the best resource using the game-theoretic concepts. Duan *et al.* [22] modeled workflow scheduling the problem as a sequential cooperative game and proposed a communication and storage-aware multi-objective algorithm with network bandwidth and storage requirements as the constraints. Sujana *et al.* [23] applied the game multi objective algorithm for minimizing the execution time and cost of single workflow applications.

3 Model and Formulation

In this section, we first present the problem description and formulation of multi-objective workflow scheduling over heterogeneous cloud VMs. Then, we propose a finite multi-stage game model, i.e., Fig. 1, to reconcile multiple objectives and introduce a dynamic game-theoretic-based algorithm to reduce make-span, optimize system fairness and reduce the total cost.

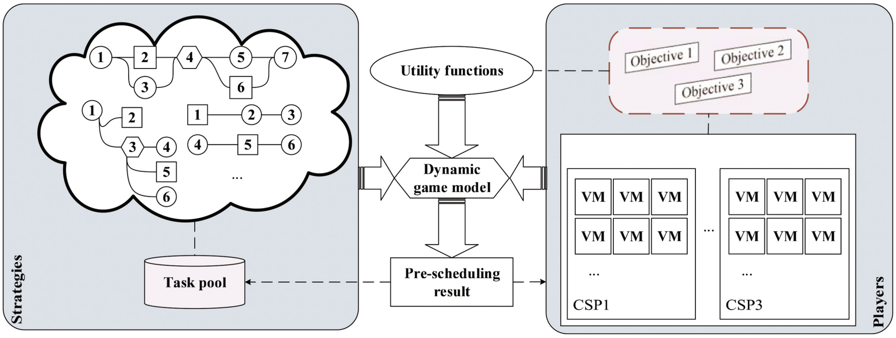


Fig. 1. The abstract model for multi-workflow scheduling problem

3.1 Problem Formulation

In this study, we consider that scientific computational processes can be described by multiple workflows which are supposed to be scheduled into heterogeneous VMs created over multiple IaaS CSPs. Each workflow can be represented by a directed acyclic graph (DAG), $W = (V, E)$, where V is a set of n tasks, i.e., $\{t_1, t_2, \ldots, t_n\}$. E is a set of precedence dependencies. Each task t_i represents an individual application with a certain task execution time v_i on a VM. A precedence dependency $e_{ij} = (t_i, t_j)$ indicates that t_j starts only after the data from t_i are received. The source and destination of a dependency e_{ij} are called the parent and the child task, respectively. Each workflow has an input and output task, which are added to the beginning and the end, respectively. When multiple workflows are ready for execution, we first partition their tasks into multiple phases based on their hops from the input task as shown in Fig. 2. After the partition, tasks are scheduled to VMs according to our proposed method and tasks at earlier phases are scheduled earlier than those at later ones. Three quantitative objectives are considered: make-span, fairness, and total cost. Note that reducing make-span, i.e., the time required to execute all workflows, usually contradicts with cost reduction and thus we consider game-theoretic approaches to reconcile such conflicting optimization aims. The fairness maximization objective aims at achieving fair distribution of workloads among all VMs and avoiding hot-spots and performance bottle-necks.

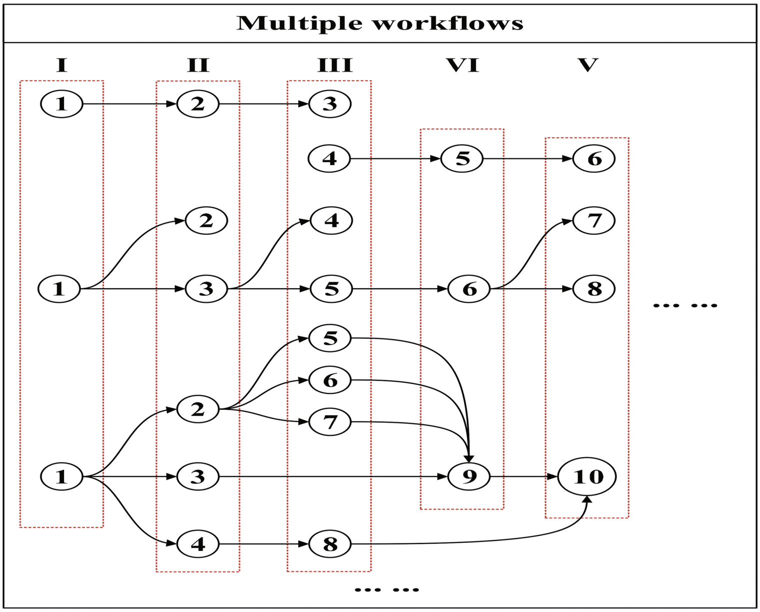


Fig. 2. The partition procedure of multiple workflows

The following hypotheses are stipulated to facilitate the development of the game-theoretic-based method: (1) VMs are created on multiple CSPs. (2) Each task can be executed by only one VM. (3) The task execution duration is the interval between the task setup time and the task cutting time. (4) The dynamic game is finite because the number of workflows and tasks are finite. The game is thus able to end within finitely many moves and every player has finitely available choices at every moment.

Based on the above hypotheses, we can formulate the problem into a multi-stage dynamic game-theoretic model:

$$Min\ f_1 = make - span = Max\ C_{ijpk} \tag{1}$$

$$Max\ f_2 = fairness\ index = \frac{(\sum_{i=1}^{K} ET_{ijpk})^2}{K \cdot \sum_{i=1}^{K} {ET_{ijpk}}^2} \tag{2}$$

$$Min\ f_3 = cost = \sum_{p=1}^{m} \sum_{k=1}^{m_p} \sum_{i=1}^{n} \sum_{j=1}^{n_i} (t_{ijpk} - t_{ijpks}) \cdot u_{pk} \cdot x_{ijpk} \tag{3}$$

subject to:

$$i \in [1, n], j \in [1, n_i], p \in [1, m], k \in [1, m_p] \tag{4}$$

$$C_{ijpk} \leq C_{i,j+1,p,k} - t_{i,j+1,p,k} - t_{i,j+1,p,k,s}, C_{ijpk} \geq 0 \tag{5}$$

$$\sum_{k \in VM(T_{ij})} x_{ijpk} = 1 \tag{6}$$

$$VM(T_{ij}) \subset VM \tag{7}$$

3.2 The Proposed Dynamic Game Model

In this paper, the multi-stage dynamic game theory is applied to deal with the conflicts and competition among multiple optimization objectives for the multi-workflow scheduling problem. The optimization objectives can be seen as players in the multi-stage dynamic game model, and the players are usually assumed to be fully rational. The game equilibrium solutions can be obtained as the optimal results. It is assumed that players take actions sequentially and the choice of the former player has an effect on the selection of the latter because the latter can observe the action of the former. The condition upon which the later makes a choice is denoted as h^l. The utility functions of the first/second/third player correspond to make-span ($u_1 = f_1$), the utility function of the second player is the second objective function which corresponds to, fairness ($u_2 = f_2$), and the total cost ($u_3 = f_3$), respectively. Consequently, the multi-stage dynamic game formulation for the problem can be described as follows:

$$G(h^l) = \{p_1, p_2, p_3; \{S_1^l\}_{l=0}^L, \{S_2^l\}_{l=0}^L, \{S_3^l\}_{l=0}^L; u_1, u_2, u_3\} \tag{8}$$

Let $H^l = \{h^l\}$ be the history set of all possible l stage. The pure strategies for player i are defined as a contingency for every possible history h^l. Formally, the l^{th} stage history information of the game is denoted as $h^l = (a^0, a^1, \ldots, a^{l-1})$. The mapping $\varphi\ _i$: $s_i \rightarrow \{S_i^l\}_{l=0}^L$ indicates the pure strategy for player i, which is a collection of mappings from all possible histories into available actions.

S_i^l is a mapping $\varphi\ _i$: $H^l \rightarrow A_i(H^l)$, i.e., for all h^l, S_i^l meets $S_i^l(h^l) \in A_i(h^l)$. The mapping $\varphi\ _i$: $f_i \rightarrow u_i$ indicates the utility functions of the game. At each stage l, the player i calculates its pure Nash equilibrium solution based on the history information in the last stage, i.e. h^{l-1}.

4 The Algorithm to Obtain Approximate Equilibrium

According to earlier discussions, each sequential game is represented by a game tree with game length of $L+1$, shown as Fig. 3. To determine the optimal behaviors of players, we employ the sub-game perfectness in the finite multi-stage game with perfect information. A multi-stage dynamic game with perfect information may have multiple Nash equilibriums some of which are with non-credible threats or promises. The sub-game perfect Nash equilibrium (SPNE) is those able to pass credibility tests. The SPNE solution can be found through a standard procedure [24] by the backward induction method. However, the standard procedure requires a traverse through the game tree and unfortunately such tree for the multi-VM multi-workflow problem is extremely large. We therefore consider approximate equilibrium solutions with reduced complexity. The approximate equilibriums can be defined as follows:

$$\sum_{l=0}^{L} u^l(S_1^l, S_2^l, S_3^l, h^{l-1}) \geq \sum_{l=0}^{L} u^l(S_1^*, S_2^*, S_3^*, h^{l-1}) \tag{9}$$

The approximate equilibrium $S^* = (S_1^*, S_2^*, S_3^*)$ is a set of strategies based on the game in Eqs. (8)–(9), where S^* is combination of the pure strategies Nash equilibriums at L stages. The decision strategies space S equals variables space X.

We introduce a multi-stage dynamic game-theoretic (MDGT) algorithm, **Algorithm 1**, to obtain the approximate equilibrium solutions. In this algorithm, during each stage l of the implementation of the workflow planning, a dynamic-game theory-based real-time scheduling method is triggered so that the tasks can be assigned to the most suitable VMs based on the real-time cloud environment. The aim of the scheduling layer is to map optional tasks to the most appropriate VMs. The algorithm repeatedly handles each stage until all tasks are scheduled and the major steps within each stage are as follows:

Step 1: create a real-time scheduling task pool with multi-phase tasks from multiple workflows to put T_{option}^l into it. T_{option}^l is supposed to meet topological dependence of its corresponding workflow, i.e., a task is executed only after all its preceding ones are executed.

Step 2: assign VM_{idle}^l to three objectives in turns. Each virtual machine of VM_{idle}^l which is allocated to f_i could choose the corresponding task from the real-time scheduling task pool. The mapping of tasks to idle virtual machines is called the strategies of the players.

Step 3: calculate the utility functions based on Eqs. (1)–(3) using the pure strategy Nash equilibrium based on historical information h^{l-1}. Each T^l_{option} will best match with VM^l_{idle} at stage l.

Step 4: construct a finite dynamic game model and obtain the equilibrium solutions.

Algorithm 1. The MDGT scheduling algorithm

Input: *task_pool* : a priority queue of workflows; T^l_{option} ; VMs ; w^l_i

Output: *opt_stra* : the real-time scheduling strategies

```
$opt\_stra = \emptyset$
for $l$ =1 to $L$ do
    if $task\_pool \neq \emptyset$ is true then
        dequeue from $task\_pool$ as $T^l_{option}$
        calculate $u^l_1(S^l_1)$ by using Eq.(1) and $w^l_1$;
        save the startegy branch of payoff to opt_stra;
    else
        break
    end if
    if $task\_pool \neq \emptyset$ is true then
        dequeue from $task\_pool$ as $T^l_{option}$
        calculate $u^l_2(S^l_2)$ by using Eq.(2) and $w^l_2$;
        save the startegy branch of payoff to opt_stra;
    else
        break
    end if
    if $task\_pool \neq \emptyset$ is true then
        dequeue from $task\_pool$ as $T^l_{option}$
        calculate $u^l_3(S^l_3)$ by using Eq.(3) and $w^l_3$ ;
        save the startegy branch of payoff to opt_stra;
    else
        break
    end if
end for
return opt_stra
```

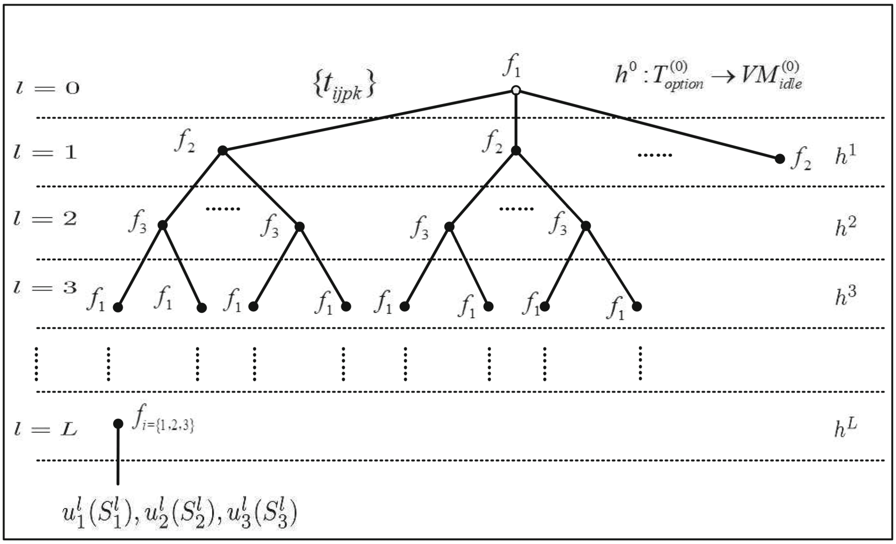


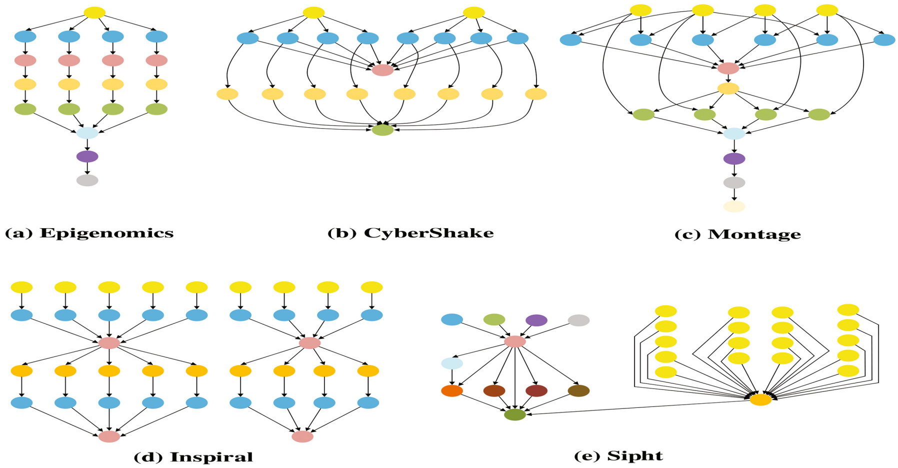
Fig. 3. The game tree template of multiple players

Table 2. The unit price of heterogeneous VMs from three IaaS providers

CSP	vType	vCPU	Memory	Unit price/hour
Amazon EC2	t2.nano	1	0.5	\$0.0058/h
	t2.small	1	2	\$0.023/h
	t2.medium	2	4	\$0.0464/h
	t2.large	2	8	\$0.0928/h
Tecent Cloud	s2.standard	1	1	\$0.0185/h
	s2.standard	1	2	\$0.0246/h
	s2.standard	2	4	\$0.0493/h
	s2.standard	2	8	\$0.0739/h
Ali Cloud	ecs.c5.large	2	4	\$0.1053/h
	ecs.g5.large	2	4	\$0.1244/h
	ecs.sn1.medium	2	8	\$0.1373/h
	ecs.c5.xlarge	2	8	\$0.2502/h

5 Case Study

In this section, we conduct extensive case study based on 5 well-known scientific workflow templates as shown in Fig. 4 and real-world third-party commercial clouds, i.e., Amazon EC2, Tencent, and Ali Clouds. Every task in all workflows



(a) Epigenomics

(b) CyberShake

(c) Montage

(d) Inspiral

(e) Sipht

Fig. 4. The case templates of five workflows

implements a GaussCLegendre calculation procedure with 8M of digits through executing the Super-Pi program on VMs. We create heterogeneous VMs on these clouds and expose them to the scheduling algorithms.

Table 2 shows the price-per-unit-time of such VMs with different resource configurations. A resulting scheduling scheme generated by our proposed method is shown in Fig. 5.

We compare our proposed method with a traditional non-game-theoretic algorithm proposed in [25]. Note that: (1) we notice several other game-theoretic scheduling algorithms, e.g., [22,23,26], but find out that they are intended for different problems and based on different architectural configurations and resource constraints. We are therefore unable to compare them with our proposed method, (2) other non-game-theoretic methods can be found in, e.g., [5,27]. However, our tests show that their performance is actually very close to that of the baseline one, (3) we are pretty aware of the fact that meta-heuristic algorithms, e.g., PSO and GA-based ones, could well be promising options. However, we do not implement them and compare them with our proposed method because we consider scientific applications to be time-critical and meta-heuristic algorithms are with high time complexity.

Tables 3 and 4 present the comparisons of make-span and cost, respectively. As the total number of tasks from five workflows increases, the number of game stage increases. And our proposed MDGT method performs better than the baseline method.

In Fig. 6, we show the comparisons of fairness indexes with different IaaS cloud service providers. The curves represents that our method outperforms the baseline method. Similarly, the results in Fig. 7 show that the MDGT method performs better than baseline method on the average fairness.

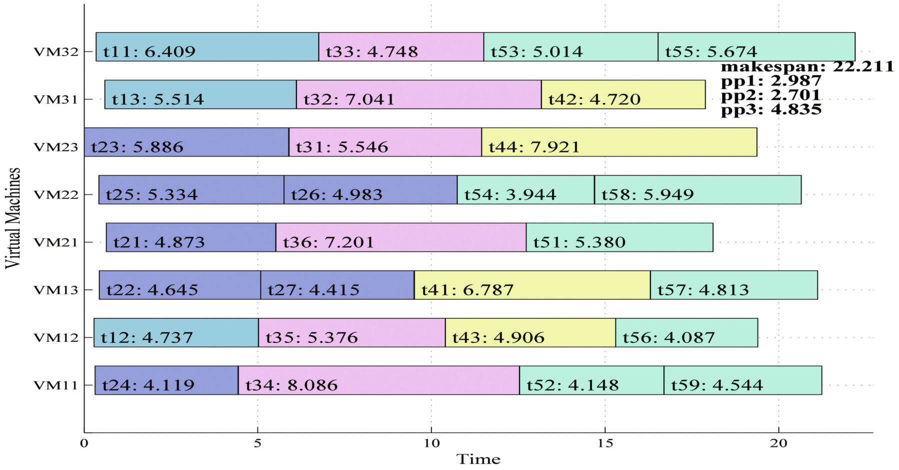


Fig. 5. An example of multi-objectives scheduling scheme for five workflows by using our proposed method

Table 3. The comparisons of make-span

No. of tasks	Game stage	Make-span	
		MDGT method	Baseline method
29	4	22.607	25.296
79	10	57.686	61.343
99	12	70.313	75.586
129	16	92.018	98.199
149	19	106.066	112.267
179	22	127.505	134.302
199	25	140.638	148.571
229	29	163.029	170.737
249	31	175.052	185.57
279	35	197.558	207.182
299	37	210.133	222.017
329	41	231.856	243.573
349	44	245.639	258.34
379	47	266.626	280.722
399	50	280.888	293.773

Table 4. The comparisons of the total cost

No. of tasks	Game stage	Cost	
		MDGT method	Baseline method
29	4	1.309	1.388
79	10	3.534	3.679
99	12	4.418	4.643
129	16	5.723	6.06
149	19	6.647	6.989
179	22	7.974	8.375
199	25	8.843	9.314
229	29	10.151	10.729
249	31	11.061	11.658
279	35	12.419	13.075
299	37	13.247	13.98
329	41	14.616	15.358
349	44	15.489	16.296
379	47	16.796	17.699
399	50	17.78	18.644

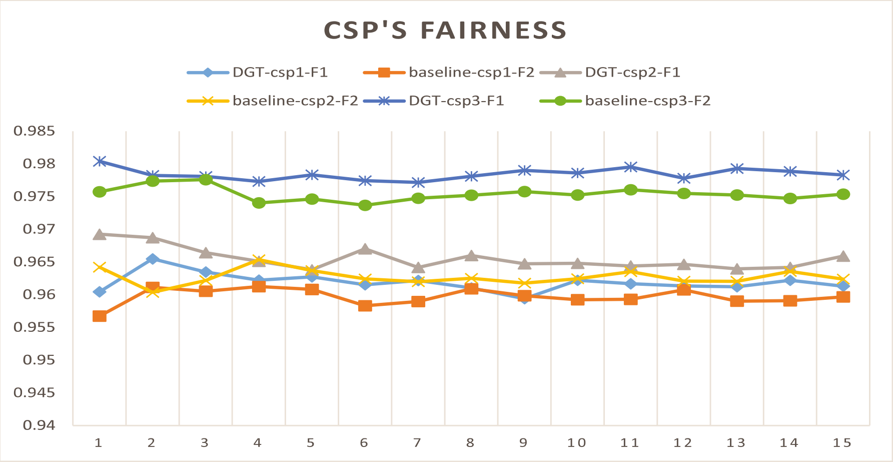


Fig. 6. The comparisons of different CSPs fairness

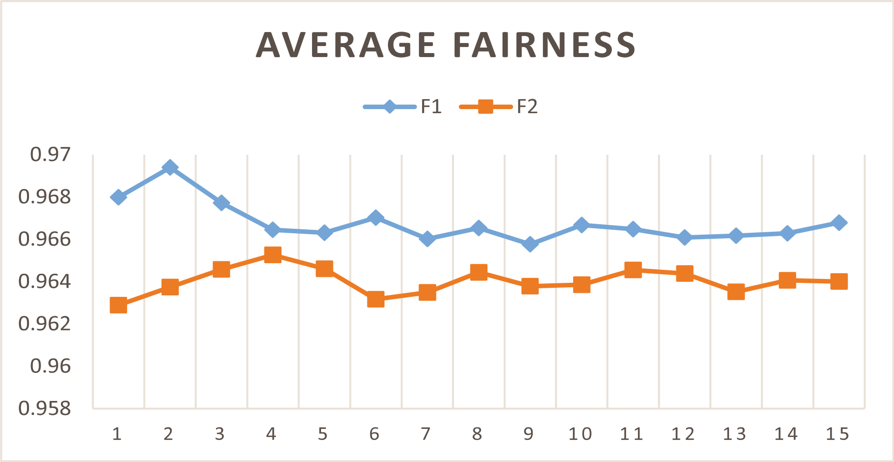


Fig. 7. The comparisons of average fairness

6 Conclusion

In this paper, we studied multi-objective multi-workflow scheduling problem over heterogeneous VMs created on multi-Clouds platforms and introduce a multi-stage dynamic game-theoretic (MDGT) scheduling approach. The proposed method is featured by approximation algorithm for identifying equilibrium solutions aiming at optimizing both workflow make-span, system fairness and the total cost. In addition, we conduct extensive experiments based on various well-kwon scientific workflow templates and real-world third-party commercial IaaS clouds. Experimental results demonstrate that our approach outperforms traditional baseline ones.

References

1. Rodriguez, M.A., Buyya, R.: A taxonomy and survey on scheduling algorithms for scientific workflows in IaaS cloud computing environments. Concurr. Comput. Pract. Exp. **29**(8), e4041 (2017)
2. Ye, X., Liang, J., Liu, S., Li, J.: A survey on scheduling workflows in cloud environment. In: Proceedings of the 2015 International Conference on Network and Information Systems for Computers. ICNISC 2015, pp. 344–348 (2015)
3. Buyya, R.: Market-oriented cloud computing: vision, hype, and reality of delivering computing as the 5th utility. In: 2009 9th IEEE/ACM International Symposium on Cluster Computing and the Grid. CCGRID 2009, vol. 25, no. 6, p. 1 (2009)
4. Chirkin, A.M., Belloum, A.S.Z., Kovalchuk, S.V., Makkes, M.X.: Execution time estimation for workflow scheduling. In: 2014 9th Workshop on Workflows in Support of Large-Scale Science, pp. 1–10 (2014)
5. Shi, L., Zhang, Z., Robertazzi, T.: Energy-aware scheduling of embarrassingly parallel jobs and resource allocation in cloud. IEEE Trans. Parallel Distrib. Syst. **28**(6), 1607–1620 (2017)

6. Chirkin, A.M., et al.: Execution time estimation for workflow scheduling. Futur. Gener. Comput. Syst. Int. J. eScience **75**, 376–387 (2017)
7. Wu, Q., Ishikawa, F., Zhu, Q., Xia, Y., Wen, J.: Deadline-constrained cost optimization approaches for workflow scheduling in clouds. IEEE Trans. Parallel Distrib. Syst. **28**(12), 3401–3412 (2017)
8. Yao, G., Ding, Y., Hao, K.: Using imbalance characteristic for fault-tolerant workflow scheduling in cloud systems. IEEE Trans. Parallel Distrib. Syst. **28**(12), 3671–3683 (2017)
9. Chen, H., Zhu, X., Qiu, D., Liu, L., Du, Z.: Scheduling for workflows with security-sensitive intermediate data by selective tasks duplication in clouds. IEEE Trans. Parallel Distrib. Syst. **28**(9), 2674–2688 (2017)
10. Liu, J., Pacitti, E., Valduriez, P., de Oliveira, D., Mattoso, M.: Multi-objective scheduling of scientific workflows in multisite clouds. Futur. Gener. Comput. Syst. **63**, 76–95 (2016)
11. Shukla, S.: An evolutionary study of multi-objective workflow scheduling in cloud computing. Int. J. Comput. Appl. **133**(14), 14–18 (2016)
12. Durillo, J.J., Nae, V., Prodan, R.: Multi-objective workflow scheduling: an analysis of the energy efficiency and makespan tradeoff. In: Proceedings - 13th IEEE/ACM International Symposium on Cluster, Cloud, and Grid Computing. CCGrid 2013, pp. 203–210 (2013)
13. Yassa, S., Chelouah, R., Kadima, H., Granado, B.: Multi-objective approach for energy-aware workflow scheduling in cloud computing environments. Sci. World J. **2013**, 13 (2013)
14. Khajemohammadi, H., Fanian, A., Gulliver, T.A.: Fast workflow scheduling for grid computing based on a multi-objective Genetic Algorithm. In: Proceedings of the IEEE Pacific RIM Conference on Communications, Computers, and Signal Processing, pp. 96–101 (2013)
15. Zhu, Z., Zhang, G., Li, M., Liu, X.: Evolutionary multi-objective workflow scheduling in cloud. IEEE Trans. Parallel Distrib. Syst. **27**(5), 1344–1357 (2016)
16. Chen, W.-N., Zhang, J.: An ant colony optimization approach to a grid workflow scheduling problem with various QoS requirements. IEEE Trans. Syst. Man, Cybern. Part C: Applications Rev. **39**(1), 29–43 (2009)
17. Padmaveni, K., Aravindhar, D.J.: Hybrid memetic and particle swarm optimization for multi objective scientific workflows in cloud. In: 2016 IEEE International Conference on Cloud Computing in Emerging Markets (CCEM), pp. 66–72 (2016)
18. Zheng, X., Wang, L.: A Pareto based fruit fly optimization algorithm for task scheduling and resource allocation in cloud computing environment. In: 2016 IEEE Congress on Evolutionary Computation (CEC), pp. 3393–3400 (2016)
19. Hou, Y., Wu, N., Zhou, M., Li, Z.: Pareto-optimization for scheduling of crude oil operations in refinery via genetic algorithm. IEEE Trans. Syst. Man, Cybern. Syst. **47**(3), 517–530 (2017)
20. Ebadifard, F., Babamir, S.M.: Optimizing multi objective based workflow scheduling in cloud computing using black hole algorithm. In: 2017 3rd International Conference on Web Research. ICWR 2017, pp. 102–108, April 2017
21. Fard, H.M., Prodan, R., Fahringer, T.: A truthful dynamic workflow scheduling mechanism for commercial multicloud environments. IEEE Trans. Parallel Distrib. Syst. **24**(6), 1203–1212 (2013)
22. Duan, R., Prodan, R., Li, X.: Multi-objective game theoretic schedulingof bag-of-tasks workflows on hybrid clouds. IEEE Trans. Cloud Comput. **2**(1), 29–42 (2014)

23. Sujana, J.A.J., Revathi, T., Karthiga, G., Raj, R.V.: Game multi objective scheduling algorithm for scientific workflows in cloud computing. In: IEEE International Conference on Circuit, Power and Computing Technologies. ICCPCT 2015, pp. 1–6 (2015)
24. Pettit, P., Sugden, R.: The backward induction paradox. J. Philos. **86**(4), 169–182 (1989)
25. Topcuoglu, H., Hariri, S., Wu, M.: Performance-effective and low-complexity task scheduling for heterogeneous computing. Parallel Distrib. Syst. **13**(3), 260–274 (2002)
26. Zhang, L., Zhou, J.: Task scheduling and resource allocation algorithm in cloud computing system based on non-cooperative game. In: 2017 IEEE 2nd International Conference on Cloud Computing and Big Data Analysis (ICCCBDA), pp. 254–259 (2017)
27. Balouek-Thomert, D., Bhattacharya, A.K., Caron, E., Gadireddy, K., Lefevre, L.: Parallel differential evolution approach for cloud workflow placements under simultaneous optimization of multiple objectives. In: 2016 IEEE Congress on Evolutionary Computation (CEC), pp. 822–829 (2016)