A Novel Algorithm for Optimizing Selection of Cloud Instance Types in Multi-cloud Environment

Wenqiang Liu
School of Computer Science and Technology
Donghua University
Shanghai, China

Pengwei Wang*
School of Computer Science and Technology
Donghua University
Shanghai, China

Ying Meng
School of Computer Science and Technology
Donghua University
Shanghai, China

Guobing Zou
School of Computer Engineering and Science
Shanghai University
Shanghai, China

Zhaohui Zhang
School of Computer Science and Technology
Donghua University
Shanghai, China

***Abstract*—With the development of cloud computing, the cloud market is becoming more and more complicated. There are many cloud providers and different cloud instance types, which brings users some confusion when they select cloud instance types. In order to solve the cloud instance type selection problem in multi-cloud environment, a Cloud Instance Type Selection Algorithm based on Genetic Algorithm (CITSA-GA) is proposed. CITSA-GA mainly includes two-dimensional encoding with the constraint between adjacent genes, selection operation adopting the elite retention strategy and the roulette strategy, crossover operation using the first fit strategy, and mutation operation with mutation bounds. We perform some experiments to prove the effectiveness of the proposed CITSA-GA.**

***Index Terms*—Cloud Computing, Instance Type Selection, Multi-cloud, Multi-objective Optimization, Modified Genetic Algorithm.**

I. Introduction

Cloud computing refers to both the applications delivered as services over the Internet and the hardware and software in the data centers that provide those services [1]. Using cloud computing can save the cost of purchasing and maintaining hardware resources, and because there are many cloud support technologies, it has high reliability and security, so cloud computing is very popular with individuals and business. Infrastructure as a Service (IaaS) is a service type provided by cloud computing, which provides different cloud instance types, and users can choose them according to their actual demands, which is the research object of this paper.

With the development of cloud computing, the cloud market is becoming more and more complex. There are many cloud service providers, and each cloud service provider has many cloud data centers, which belong to different regions. At the same time, there are several cloud instance types in each cloud data center. Like Amazon Elastic Compute Cloud (EC2) [2], which is a representative of IaaS, there are 18 cloud regions in June 2018. In ap-northeast-2 region, the number of on-demand instance types whose operation systems are Linux/UNIX is 54, and these instance types are grouped into 5 instance families based on their capabilities, meanwhile in each family there are many different instance types and each one has different price and configuration, which is shown in TABLE I. Since the complexity of cloud market, there are some confusion when users select cloud instance types [3]. Consequently, the cloud instance type selection problem in multi-cloud environment has important practical significance and needs to be solved.

In this background, this paper studies the cloud instance type selection problem in multi-cloud environment. Firstly, the problem definition is given, which is defined as a multi-objective optimization problem. Then, we propose a Cloud Instance Type Selection Algorithm based on Genetic Algorithm (CITSA-GA) to solve this problem. Finally, some experiments are conducted to verify the efficiency and effectiveness of CITSA-GA.

The remainder of this paper is organized as follows. In section 2, the related work about the cloud instance type selection problem is discussed. Section 3 describes the mathematical description of cloud instance type selection problem. Then, CITSA-GA is proposed in section 4. After that, some experiments are performed in section 5. Finally, section 6 concludes this paper and presents future directions.

II. Related Work

In this section, some related work about cloud instance type selection problem and the application of meta-heuristic algorithms in resource optimization scheduling in cloud computing is introduced.

Cloud instance type selection problem aims to find the best solution set of instances (different types and numbers) to meet the demand of users. Tordsson et al. [4] define this problem as

TABLE I
The Statistics Result of Cloud Instance Types
(Provider: Amazon; Category: On-demand Instance; Region: ap-northeast-2; OS: Linux/UNIX; Time: Jun 2018)

Instance Family	Type Number	Example	ECU	Price($)
General Purpose	19	m5.large	10	0.118
		m4.large	6.5	0.123
Compute Optimized	11	c5.large	8	0.096
		c4.large	8	0.114
GPU Instances	6	p2.xlarge	12	1.465
		p3.2xlarge	23.5	4.981
Memory Optimized	8	x1.16xlarge	174.5	9.671
		r4.large	7	0.16
Storage Optimized	10	i3.large	7	0.183
		d2.xlarge	14	0.844

* Pengwei Wang is the corresponding author. (wangpengwei@dhu.edu.cn)

978-1-7281-2583-1/19/$31.00 ©2019 IEEE
DOI 10.1109/ICPADS47876.2019.00033

a 0-1 integer programming problem with multiple constraints and one objective, which is maximizing total computing capacity, and use CPLEX solver to solve this problem. After that, Li et al. [5] consider the cloud brokering mechanism of virtual machine placement in dynamic scenarios.

In our previous work [6], we propose a cloud instance type selection framework and define a cloud instance type selection problem with cost and instance types proportional constraints, and in this work, the objective is maximizing the total computing capacity. In order to solve this problem, a dynamic programming approach is used by mapping the problem to the unbound knapsack problem. However, this paper does not consider the number constraint of cloud instances, as a result, this method tends to choose those instance types that have a high performance-price ratio. Generally, the performance of these instance types is not high, so the number of them in the selection scheme if often very large. For example, in a cluster system, the performance of the whole system is not merely the accumulation of individual performance. If there are too many individuals, communication delay will be high, so the performance of the whole system may be affected. Consequently, it is not enough to just consider the total performance, and the number of cloud instances should also be considered. Regarding the cloud instance type selection problem in multi-cloud environment, the communication delay is a very important factor, which is not considered in the current work, and we will take it into account in our algorithm.

In our previous work [7], a two-stage Cloud Instance Type Selection Model (CITSM) is proposed to help users select the cloud instance types. The first stage is a Complete Pareto Set Generation Algorithm (CPSGA) which can generate a complete Pareto set of selection schemes. Then, the Optimal cloud instance type selection Scheme Screening Algorithm (OSSA) is used to select one scheme from the complete Pareto set. CITSM is very efficient and effective. However, it does not aim to solve the selection problem in multi-cloud environment.

In the field of resource optimization scheduling in cloud computing, meta-heuristic algorithms are the usually used algorithm. Virtual machine placement problem aims to find the best physical machine to host the virtual machines [8]. In order to optimize energy consumption in virtual machine placement problem, the authors in [9] propose an improved particle swarm optimization algorithm. Gao et al. [10] propose a multi-objective ant colony optimization algorithm to solve it. Although there are many studies on the field of resource optimization scheduling in cloud computing, there are few studies to solve the cloud instance type selection problem by using meta-heuristic algorithm.

Based on the problems in above related work, we re-analyze the cloud instance type selection problem to establish a multi-objective mathematical problem, and propose CITSA-GA to solve this problem, which mainly includes two-dimensional encoding with the constraint between adjacent genes, selection operation adopting the elite retention strategy and the roulette strategy, crossover operation using the first fit strategy, and mutation operation with mutation bounds. The experiments prove that the effect of CITSA-GA is better than Genetic Algorithm (GA) and Particle Swarm Optimization Algorithm (PSO).

III. Problem Definition

The objective of cloud instance type selection problem in multi-cloud environment is to maximize the total computing capacity (TC), minimize the total price (TP) and minimize the total communication delay (TD) of the selection scheme. Simultaneously, some constraints should be satisfied. Thus, the cloud instance type selection problem is defined as follows:

$$\begin{cases} maximize & TC \\ minimize & TP \\ minimize & TD \end{cases} \tag{1}$$

Subject to:

$$Pa_i \leq maxPa \ (1 \leq i \leq n_C, maxPa \times n_V \geq 1) \tag{2}$$

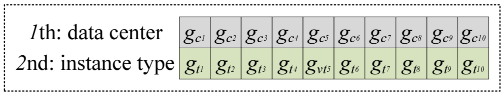
where n_V, n_C are used to represent the number of cloud instances that user required and cloud data center, respectively. Pa_i indicates the ratio of allocating cloud instances to the ith cloud, and $maxPa$ is the maximum allocation ratio, which can reduce the impact of the data center crash on the user.

IV. Proposed Method

In order to solve the cloud instance type selection problem in multi-cloud environment, the Genetic Algorithm is adopted, and we improve it to get CITSA-GA, mainly including two-dimensional encoding with the constraint between adjacent genes, selection operation adopting the elite retention strategy and the roulette strategy, crossover operation using the first fit strategy, and mutation operation with mutation bounds.

A. Genetic Representation

A two-dimensional encoding method is proposed, which is encoded according to the cloud data center and cloud instance type. At the same time, for each gene, we take the form of integer encoding.



*1*th: data center	g_{c1}	g_{c2}	g_{c3}	g_{c4}	g_{c5}	g_{c6}	g_{c7}	g_{c8}	g_{c9}	g_{c10}
*2*nd: instance type	g_{t1}	g_{t2}	g_{t3}	g_{t4}	g_{vt5}	g_{t6}	g_{t7}	g_{t8}	g_{t9}	g_{t10}

Fig. 1. An example of Two-Dimensional Encoding:
$n_C = 5$, $n_T = 6$, $n_V = 10$

For example, there are 5 cloud data centers ($n_C = 5$), each with 6 cloud instance types ($n_T = 6$), and users need 10 cloud instances ($n_V = 10$). In the case of two-dimensional encoding, the encoding result is shown in Fig. 1. For each gene at the dimension of cloud data center, there are 5 possible values, and for instance type, there are 6 possible values.

In the first dimension, the number of selected cloud data center follows a non-decreasing arrangement. While the first dimension is the same, the number of the selected cloud instance type is also non-decreasing. Based on this constraint, the search space can be reduced, because the range of each gene becomes smaller.

B. Fitness Function

We use the ideal point method to deal with the objective function. As a result, the optimization goal of the cloud instance type selection problem can be transformed to minimize the fitness function.

$$f = \sqrt{(TC - \hat{TC})^2 + (TP - \hat{TP})^2 + (TD - \hat{TD})^2} \quad (3)$$

where $\hat{TC}$, $\hat{TP}$ and $\hat{TD}$ denote the ideal value about TC, TP and TD, respectively.

C. Genetic Operation

In CITSA-GA, the genetic operations are modified according to the cloud instance type selection problem in multi-cloud environment.

For the selection operation, a roulette strategy is adopted. At the same time, in order to improve the convergence speed of the proposed algorithm, we also introduce the elite retention strategy in the selection operation.

In crossover operation, there is a crossover probability P_c, which determines whether each chromosome do crossover operation. When performing the crossover operation, the original GA is not suitable, since the new chromosome obtained by directly interchanging the genes at the corresponding positions between two chromosomes often cannot satisfy the constraint between adjacent genes. In order to solve this problem, the first fit strategy is taken.

Regarding the mutation operation, for each gene, it will change with the mutation probability P_m. When performing the mutation operation, the constraint between adjacent genes should be considered. This constraint is equivalent to defining the upper and lower bounds of the genetic mutation, and then the mutation operation is completed by random method within this range.

V. Experiments and Discussions

In this section, experimental setting is introduced and some experiments are conducted to verify the effectiveness of CITSA-GA.

A. Experimental Setting

Experiments are performed on a GNU Linux Operating System with Intel(R) Core(TM)i5-7500 at 3.40 GHz and 16GB of RAM. Moreover, we use Python3.6 to implement CITSA-GA.

The data used in this paper is the on-demand instance type data with Linux/UNIX operation system of Amazon EC2 [11]. The data centers are us-west-1, eu-central-1, ap-northeast-1, ap-south-1 and sa-east-1. The instance types are c5.large, c5.xlarge, c5.2xlarge, c5.4xlarge, c5.9xlarge and c5.18xlarge.

We use three algorithms as compared methods to verify the effectiveness of CITSA-GA. The first one is Traversal algorithm, because it can get the optimal result. At the same time, GA and PSO are the usually used algorithm in the field of resource optimization scheduling in cloud computing, like workflow scheduling [12], task scheduling [13], and so on. Thus, we also use GA and PSO as our compared algorithms, and the setting of parameters is: $max_{Pa} = 50\%, n_V \in [1, 20]$.

TABLE II
Frequency of the Optimal Solution.

Algorithm	**CITSA-GA**	**GA**	**PSO**	**Traversal**
Frequency	82.5%	32%	35.5%	100%

B. Experimental Results and Discussions

In order to verify the effectiveness of CITSA-GA, we separately calculate the highest f value when the algorithm reaches convergence, the lowest f value and the frequency at which the algorithm can converge to its optimal value (the lowest f value). For ease of display, we normalize the f value and map the result to 0-1.

Fig. 2 is the comparison result of the highest f value under different n_V. It can be seen from the figure that when the value of n_V is small ($n_V < 3$), all algorithms can reach the minimum f value, but when the value of n_V becomes larger, some algorithms are unstable, especially PSO.

Fig. 3 is the lowest f value comparison of various methods. We can know that, in most cases, these methods can achieve the best results, but from overall, PSO still has problems in the aspect of finding the best solution. CITSA-GA is the best method in the comparison of lowest f value.

Finally, the frequency when each algorithm achieves its optimal solution (lowest f value) is counted, which is shown in Table II. Since Traversal algorithm traverses the whole search space, it can find the optimal solution, but when the n_V is large, the huge search space makes Traversal algorithm has no practical value. Our proposed algorithm has a very good convergence frequency to the optimal solution, meanwhile, it

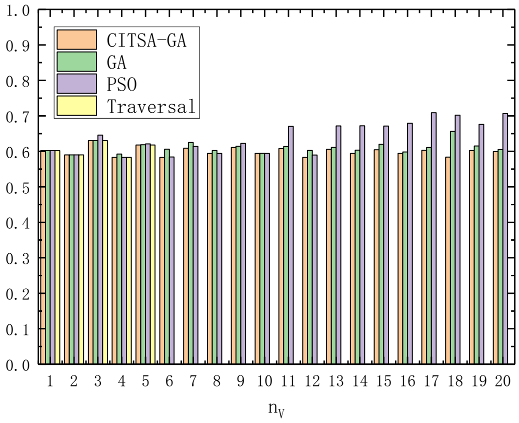


Fig. 2. The highest f value comparison of various methods under different cloud instance numbers that user required.

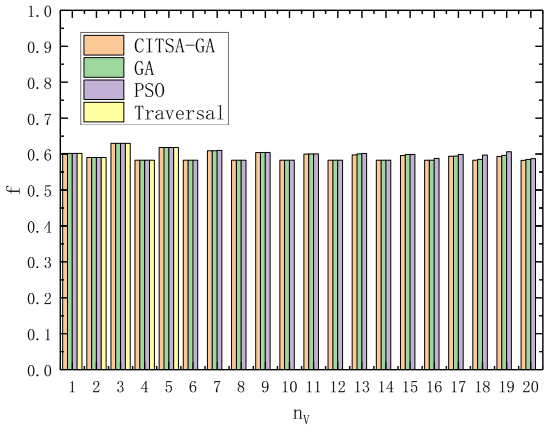


Fig. 3. The lowest f value comparison of various methods under different cloud instance numbers that user required.

has the lowest f value, so CITSA-GA can solve the problem of cloud instance type selection very well.

VI. Conclusions

In this paper, the problem of cloud instance type selection in multi-cloud environment is studied, and CITSA-GA algorithm is proposed based on GA to solve this problem. CISTA-GA mainly includes two-dimensional encoding with the constraint between adjacent genes, selection operation adopting the elite retention strategy and the roulette strategy, crossover operation using the first fit strategy, and mutation operation with mutation bounds. Compared with Traversal Algorithm, GA and PSO, the proposed CITSA-GA can achieve the lowest f value. The future work includes two directions: one is studying the cloud instance types selection problem under different pricing modes combined with the price prediction [14], and the other is to consider the cloud data storage [15] while selecting cloud instance types.

Acknowledgment

This work was partially supported by the National Natural Science Foundation of China (NSFC) under Grant 61602109, Natural Science Foundation of Shanghai under Grant 19ZR1401900, Fundamental Research Funds for the Central Universities, and DHU Distinguished Young Professor Program under Grant LZB2019003.

References

[1] M. Armbrust, A. Fox, R. Griffith, A. Joseph, R. Katz, A. Konwinski, G. Lee, D. Patterson, A. Rabkin, I. Stoicam, and M. Zaharia. ”A view of cloud computing”. *Communications of the ACM*, 2010, 53(4):50-58.

[2] *Amazon EC2*. Accessed on: Jun. 2018, [Online] Available: https://aws.amazon.com/ec2.

[3] J. Mei, K. Li, Z. Tong, Q. Li, and K. Li. ”Profit maximization for cloud brokers in cloud computing”. *IEEE Transactions on Parallel and Distributed Systems*, 2018, 30(1):190-203.

[4] J. Tordsson, R.S. Montero, R. Moreno-Vozmediano, and I.M. Llorente. ”Cloud brokering mechanisms for optimized placement of virtual machines across multiple providers”. *Future Generation Computer Systems*, 2012, 28(2):358-367.

[5] W. Li, J. Tordsson, and E. Elmroth. ”Modeling for dynamic cloud scheduling via migration of virtual machines”. In *IEEE Third International Conference on Cloud Computing Technology and Science*, 2011, 163-171. IEEE, Athens, Greece.

[6] P. Wang, W, Zhou, and Z, Zhang. ”A dynamic programming-based approach for cloud instance types selection and optimization”. *International Journal of Information Technology and Management*, 2019. (to be published)

[7] W. Liu, P. Wang, Y. Meng, Q. Zhao, C. Zhao and Z. Zhang, ”A Novel Model for Optimizing Selection of Cloud Instance Types”. *IEEE Access*, 2019, 7:120508-120521.

[8] M. Masdari, S.S. Nabavi, and V. Ahmadi. ”An overview of virtual machine placement schemes in cloud computing”. *Journal of Network and Computer Applications*, 2016, 66:106-127.

[9] S. Wang, A. Zhou, C.H. Hsu, X. Xiao, and F. Yang. ”Provision of data-intensive services through energy-and qos-aware virtual machine placement in national cloud data centers”: *IEEE Transactions on Emerging Topics in Computing*, 2016, 4(2):290-300.

[10] Y. Gao. H. Guan, Z. Qi, Y. Hou, and L. Liu. ”A multi-objective ant colony system algorithm for virtual machine placement in cloud computing”. *Journal of Computer and System Sciences*, 2013, 79(8):1230-1242.

[11] *Amazon EC2 On-Demand Instances*. Accessed on: Jun. 2018, [Online] Available: https://aws.amazon.com/ec2/pricing/on-demand.

[12] P.T. Thant, C. Powell, M. Schlueter, and M. Munetomo. ”A level-wise load balanced scientific workflow execution optimization using NSGA-II”. In *17th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing*, 2017, 882-889. IEEE, Madrid, Spain.

[13] A.S. Sofia, and P. GaneshKumar. ”Multi-objective task scheduling to minimize energy consumption and makespan of cloud computing using NSGA-II”. *Journal of Network and Systems Management*, 2018, 26(2):463-485.

[14] W. Liu, P. Wang, Y. Meng, C. Zhao, and Z. Zhang. ”Amazon EC2 Spot Instance Price Prediction using kNN Regression”. In 2018 Asia-Pacific Services Computing Conference, 2018. IEEE, Zhuhai, China.

[15] P. Wang, C. Zhao, and Z. Zhang. ”An ant colony algorithm-based approach for cost-effective data hosting with high availability in multi-cloud environments”. *15th IEEE International Conference on Networking, Sensing and Control*, 2018, 1-6. IEEE, Zhuhai, China.