Azerbaijan Journal of High Performance Computing, Vol 4, Issue 1, 2021, pp. 39-47
https://doi.org/10.32010/26166127.2021.4.1.39.47

Trust Base Job Scheduling in Cloud Computing

Farshad Rezaei, Shamsollah Ghanbari
Islamic Azad University, Ashtian Branch, Ashtian, Iran, arshad.rezaei23@gmail.com

*Correspondence: Farshad Rezaei, Islamic Azad University, Ashtian Branch, Ashtian, Iran, arshad.rezaei23@gmail.com

Abstract

Cloud computing is a new technology recently being developed seriously. Scheduling is an essential issue in the area of cloud computing. There is an extensive literature concerning scheduling in the area of distributed systems. Some of them are applicable for cloud computing. Traditional scheduling methods are unable to provide scheduling in cloud environments. According to a simple classification, scheduling algorithms in the cloud environment are divided into two main groups: batch mode and online heuristics scheduling. This paper focuses on the trust of cloud-based scheduling algorithms. According to the literature, the existing algorithm examinee latest algorithm is related to an algorithm trying to optimize scheduling using the Trust method. The existing algorithm has some drawbacks, including the additional overhead and inaccessibility to the past transaction data. This paper is an improvement of the trust-based algorithm to reduce the drawbacks of the existing algorithms. Experimental results indicate that the proposed method can execute better than the previous method. The efficiency of this method depends on the number of nods and tasks. The more trust in the number of nods and tasks, the more the performance improves when the time cost increases

Keyword: Cloud Computing, Task Scheduling, Trust Method, Distributed Systems, Heuristics Scheduling

1. Introduction

Cloud Computing is a trending technology that allows users to use computing resources remotely in a pay-per-use model. One of the main challenges in a cloud computing environment is task scheduling, in which tasks should be scheduled efficiently to minimize execution time and cost while maximizing resource utilization (Shukri, S. E., Al-Sayyed, R., Hudaib, A., & Mirjalili, S., 2021). Cloud environment enables the users to utilize many virtual resources for every requested task, making the manual and traditional scheduling techniques, not an efficient solution that introduces the need to have new efficient scheduling solutions (Arunarani, A. R., Manjula, D., & Sugumaran, V., 2019). Some researchers applied task scheduling to an environment similar to the cloud environment, such as Dai et al. (Dai, H., Zeng, X., Yu, Z., & Wang, T., 2019). However, over time, due to the limitation of resources and many requests and demands on qualitatively of different users, scheduling shared resources and allocating them is treated as a

key issue. Traditional scheduling algorithms are not efficient enough to respond to this growing requirement (Ghanbari, S., & Othman, M., 2012; Aghababaeipour, Ali, and Shamsollah Ghanbari, pp. 308-317. Springer, Cham, 2018; Ghanbari, Shamsollah. no. 1 (2019): 29-38.). That is why the requirement for scheduling algorithms tailored to the cloud network was strongly felt. So trust will be achieved when we get our expectations and received complete services (Zissis, D., & Lekkas, D., 2012; Khalifehlou, Z. A., & Gharehchopogh, F. S., 2012, May; Kim, W., 2009). Trust is communication between users and service providers, and during the implementation process, an important role plays in cooperation between them (Che, J., Duan, Y., Zhang, T., & Fan, J., 2011). In could computing, trust is highly regarded in the chosen algorithms (Mantri, A., Nandi, S., Kumar, G., & Kumar, S., 2011, July). According to the literature (Hoffman, D. L., & Peralta, M., 2007), 95% of users did not add their personal information on websites. So it seems trust will have a significant impact in terms of technology adoption (Friedman, B., Khan Jr, P. H., & Howe, D. C., 2000; McKnight, D. H., Choudhury, V., & Kacmar, C., 2002). To meet the challenges of large-scale applications, job scheduling algorithms are essential (Fox, A., Griffith, R., Joseph, A., et al., 2009). By mapping user-related jobs into the appropriate resources, the job scheduling mechanism can increase efficiency and reduce makespan (Özdamar, L., & Ulusoy, G., 1995). In this regard, various algorithms (Li, J., Qiu, M., Niu, J., Gao, W., Zong, Z., & Qin, X., 2010, August; Qi, P., & Li, L. S., 2012, August; Xu, B., Zhao, C., Hu, E., & Hu, B., 2011) have been proposed in this study; we have a plan to improve the model of the Cloud-DLS algorithm.

2. Related work

Some researchers applied task scheduling to environments similar to the cloud environment, such as in Dai et al. (2019). Another research by Lin et al. (2019) applied in the manufacturing sector to support production decisions made in smart factories. The main purpose in the schedule coming tasks with considering edge computing. A recent application of GA in task schedules was proposed by Jena and Mohanty (2018). Recent research for automating big data task scheduling in a cloud environment was proposed by Rjoub et al. (2019), the main purpose is to help the cloud users deal with their tasks with good performance. The latest related algorithm is called Cloud-DLS, which is based on Bayesian theory. It leads us to algorithm an efficient scheduling algorithm named Cloud-DLS (Wang, W., Zeng, G., Tang, D., & Yao, J., 2012) has been proposed. In general, a trust-based relationship is variable. One node bypassing the time whiles its successful communication with other nodes increases; trust other nodes to provide services that can be changed over time. There is a kind of trust between A and B called direct trust. It can be achieved through successful cooperation between them. Besides, there are recommended trust that if a node has no communication, the other nodes can receive it from other nodes. Based on its strategy, estimate it. Trust is not a simple issue but a process that can be calculated to a certain level. In this algorithm, there are two parameter names u and v: the number of successful and number of failure interactions. For computing u and v after n time sequences, the following formula is used:

$$V(n) = \sum_{i=1}^{N} v(i) n^{n-1} \quad (1)$$

$$U(n) = \sum_{i=1}^{N} u(i) n^{n-1} \quad (2)$$

To computing u(n) and v(n), general information and transaction processing that has been happened in the past exist and is stored in the data center. Then a decay factor allocates to them that can be a minute, a day, a month, a year, or any time interval. Complete information will be achieved to computing u and v; however, a node decided to use them or not. The problem with this method is that high memory usage is logged and cased. For computing them at its sequences:

$$V(i) = V(i-1)n + V(i)$$

$$U(i) = U(i-1)n + U(i)$$

To solve this problem, u and v can be computed in real

Time so there is no need to store the information. However, this method has its difficulty, including no access to the previous information.

3. Proposed method

Generally, the problem of calculating U(n) and V(n) is the high level of consumed memory, and the problem of calculating U(i) and V(i) methods is temporary decision-making. It is not possible to obtain a general attitude toward the system since the last time. The system has some disadvantages, including if one node requests service from another node and that node offers a negative response, repeatedly in the next communication, the request is resent regardless of this issue that such request was already sent without receiving any response from the node. In order to describe the implementation of the recommended method and by regarding the calculation of the formula mentioned in the article Base work, we have benefited from a dynamic buffer with n length, and the size of the buffer in different systems may be diverse. It may be changed with variety in the power of processing. Then the data related to transactions inside of buffer is saved and is applied in calculations related to trust at a brief time. To determine the size of this buffer, the benefit of the formula that depends on power and level of load may be tolerated by the processor at a specific time. The following equation can calculate the size of buffers:

$$Buffer = \left(\frac{CP}{CU}\right) * 100 \quad (3)$$

Where CU and CP are Power of processor and Processing

A load of processors, respectively. In order to calculate CU, we use the following equation: CU is equal to Frequency of processor * number of core processing so the size of the buffer can be calculated locally. Then, in order to save the required data queue is used. This queue can be either linear or loop. The linear queue due to potential problems that importance is the gradual movement of elements toward the end of the queue and lack of ability to use homes at the beginning of queue that is empty due to elimination shall not be applicable for this method. Thus, instead, the loop queues are used for saving data.

Table 1. Table for Definition of Parameters:

Remarks	Abbreviation
Processing load of processor	CU
The processing power of the CPU	CP
Recommended trust	(rt)
Direct trust	(dt)
Number of successful Interaction after nth sequence	V(n)
Number of failure Interaction after nth sequence	U(n)
Number of successful Interaction at the ith sequence	V(i)
Number of failure Interaction at the ith sequence	U(i)

The algorithm described above may be implemented in 2 phases which are described:

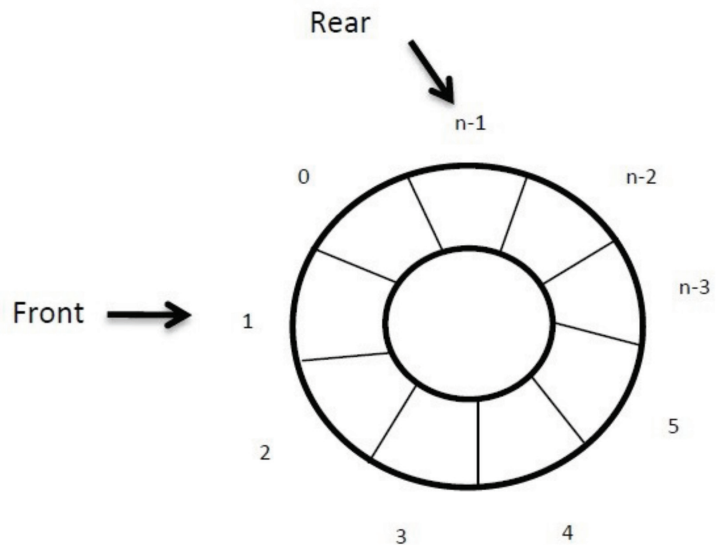

Fig. 1. Method of saving data in loop queue

• First Phase:

In the first phase shown in Fig 2, the full available information in the system, including time of transactions, information related to the success of failure, is discovered and recognized. Then if k equals 1, then the parameters u (i) and v (i) are calculated based on temporary information.

• Second Phase:

In the second phase shown in Fig. 3, upon studying the condition of k=1, the buffer size is calculated, and then data information is saved inside buffers than in the next stage, then the parameter v(n) and u(n)calculated.

4. Analysis of proposed methods

The following method has the following four advantages:

• The size of the buffer is determined based on the processing power of each system and is entirely local.

• If the number of transactions increased, the buffer size is also increased so more data could be stored.

• It is possible to obtain the valuable and applicable range of appropriate information with local properties of any system, and in case of requirement, we may refer to them.

• When data are saved inside a system with a specific amount and calculation ability on a regional basis, the overload will not be imposed into the system.

5. Analyzing trust level

Fig. 3 is related to implementation for calculating trust, and according to the descriptions offered in the previous session, it is beneficial from formula 3-1; meanwhile, level of direct trust is observed as (dt) equal 0.5, and indirect trust level is obtained by using the information of other modes during the implementation of the algorithm that is calculated by observing the level of λd 0.5 as it is revealed from the diagram described above when the level of is 0.5 it moves toward one by the lower slope.

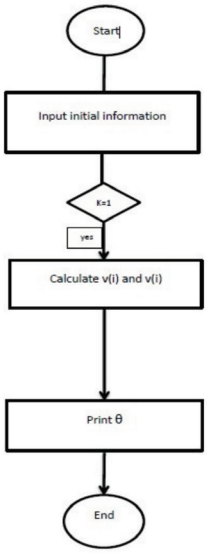

Fig. 2. Diagram of the first phase

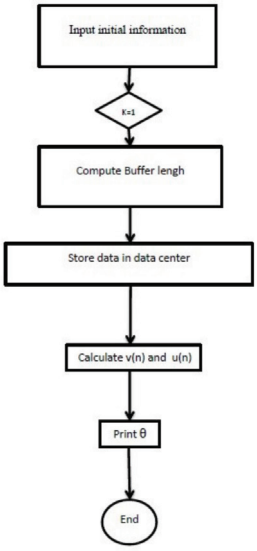

Fig. 3. Diagram of the second phase

6. Analyzing average consumed time based on number of jobs

As shown in Fig 4, when the number of jobs is increased, the average duration of performing calculations is also increased, but with this difference, the submitted method in this research shows higher time compared to the primary method. This condition is whereas the submitted method requires considering the data related to performed transactions in the determined range; thus, it is required for more time for processing and implementing jobs.

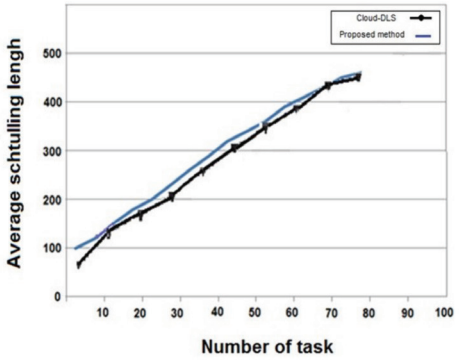

Fig. 4. Average scheduling length based on number of tasks

7. Analyzing level of success based on number of jobs

In Fig. 5, When the number of jobs is added or according to diagram 4-3, it is required for more time; but, on the other hand, we may obtain more successful transactions. This event is that whereas the submitted method on behalf of the related data benefits from previous transactions for making the decision; thus, it is required for more time for processing data; nevertheless, it may offer more acceptable efficiency for more successful transactions.

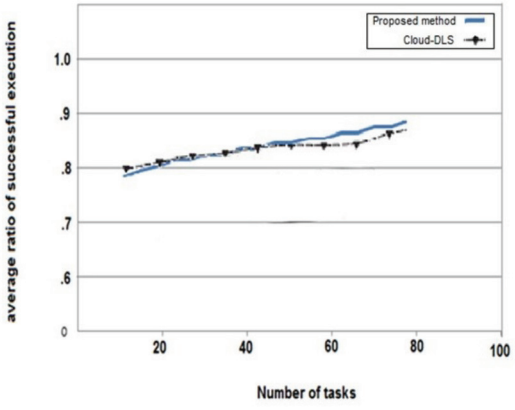

Fig. 5. The average ratio of successful execution based on the number of tasks

8. Analyzing level of success based on number of jobs

In Fig. 6, When the number of jobs is added or according to diagram 4-3, it is required for more time; but, on the other hand, we may obtain more successful transactions. This event is that whereas the submitted method on behalf of the related data benefits from previous transactions for making the decision; thus, it is required for more time for processing data; nevertheless, it may offer more acceptable efficiency for more successful transactions.

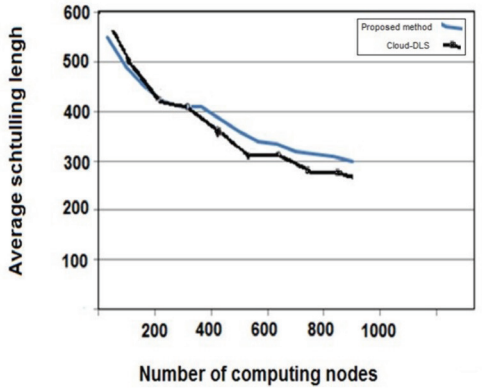

Fig. 6. Average scheduling length based on number of nodes

9. Analyzing average consumed time based on number of nodes

In Fig. 7, when the number of computing nodes is increased, the average consumed time for both methods is reduced, but the proposed method costs more time.

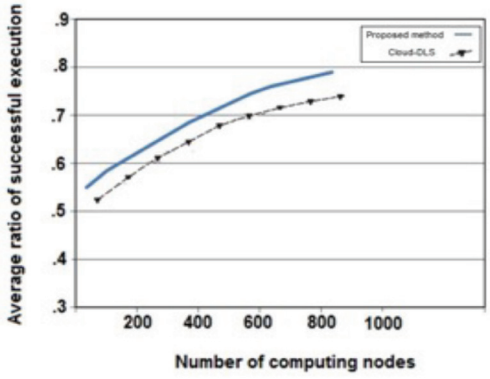

Fig. 7. Average consumed time for both methods is reduced

10. Conclusion

The recommended method has a more suitable scale and is better implemented in comparison to the previous method. Moreover, the efficiency of this method depends on the number of nodes and affairs, i.e., through the increasing number of nodes and

affairs, the performance is improved when time cost is increased. In order to calculate the level of trust in computer environments, it is applied from a kind of mechanism based on trust for reducing error for allocation of duties and guarantee for performing tasks in a safe environment. The main idea of this method was to solve the problems for the traditional formula of timetable, and the trustability was increased simultaneously.

References

S.E Shukri, R. AL-sayyed, A.Hudaib et al., Enhanced multi-verse optimizer for task scheduling in cloud computing environment. *Expert System With Applications (2020).*

Arunarani, A. R., Manjula, D., & Sugumaran, V. (2019). Task scheduling techniques in cloud computing: A literature survey. *Future Generation Computer Systems, 91,* 407-415.

Che, J., Duan, Y., Zhang, T., & Fan, J. (2011). Study on the security models and strategies of cloud computing. *Procedia Engineering, 23,* 586-593.

Dai, H., Zeng, X., Yu, Z., & Wang, T. (2019). A scheduling algorithm for autonomous driving tasks on mobile edge computing servers. *Journal of Systems Architecture, 94,* 14-23.

Dai, H., Zeng, X., Yu, Z., & Wang, T. (2019). A scheduling algorithm for autonomous driving tasks on mobile edge computing servers. *Journal of Systems Architecture, 94,* 14-23.

Fox, A., Griffith, R., Joseph, A., et al. (2009). Above the clouds: A berkeley view of cloud computing. *Dept. Electrical Eng. and Comput. Sciences, University of California, Berkeley, Rep. UCB/EECS, 28*(13), 2009.

Friedman, B., Khan Jr, P. H., & Howe, D. C. (2000). Trust online. *Communications of the ACM, 43*(12), 34-40.

Ghanbari, S., & Othman, M. (2012). A priority based job scheduling algorithm in cloud computing. *Procedia Engineering, 50*(0), 778-785.

Aghababaeipour, Ali, and Shamsollah Ghanbari. "A new adaptive energy-aware job scheduling in cloud computing." In *International Conference on Soft Computing and Data Mining, pp. 308-317.* Springer, Cham, 2018.

Ghanbari, Shamsollah. "Priority-aware Job Scheduling Algorithm in Cloud Computing: A Multi-criteria Approach." *Azerbaijan Journal of High Performance Computing 2, no. 1* (2019): 29-38.

Hoffman, D. L., & Peralta, M. (2007). Building con trust online. *Communication of the ACM, 1,* 80-85.

Jena, T., & Mohanty, J. R. (2018). GA-based customer-conscious resource allocation and task scheduling in multi-cloud computing. *Arabian Journal for Science and Engineering, 43*(8), 4115-4130.

Khalifehlou, Z. A., & Gharehchopogh, F. S. (2012, May). Security Directions in cloud Computing Environments. In *5th International Conference on Information Security and Cryptology (ISCTURKEY2012), Ankara, Turkey* (pp. 327-330).

Kim, W. (2009). Cloud computing: Today and tomorrow. *J. Object Technol., 8*(1), 65-72.

Li, J., Qiu, M., Niu, J., Gao, W., Zong, Z., & Qin, X. (2010, August). Feedback dynamic algorithms for preemptable job scheduling in cloud systems. In *2010 IEEE/WIC/ACM*

International Conference on Web Intelligence and Intelligent Agent Technology (Vol. 1, pp. 561-564). IEEE.

Lin, C. C., Deng, D. J., Chih, Y. L., & Chiu, H. T. (2019). Smart manufacturing scheduling with edge computing using multiclass deep Q network. *IEEE Transactions on Industrial Informatics, 15*(7), 4276-4284.

Mantri, A., Nandi, S., Kumar, G., & Kumar, S. (2011, July). High performance architecture and grid computing. In *International Conference, HPAGC.*

McKnight, D. H., Choudhury, V., & Kacmar, C. (2002). Developing and validating trust measures for e-commerce: An integrative typology. *Information systems research, 13*(3), 334-359.

Özdamar, L., & Ulusoy, G. (1995). A survey on the resource-constrained project scheduling problem. *IIE transactions, 27*(5), 574-586.

Qi, P., & Li, L. S. (2012, August). Job scheduling algorithm based on fuzzy quotient space theory in cloud environment. In *2012 IEEE International Conference on Granular Computing* (pp. 388-393). IEEE.

Rjoub, G., Bentahar, J., Wahab, O. A., & Bataineh, A. (2019, August). Deep smart scheduling: A deep learning approach for automated big data scheduling over the cloud. In *2019 7th International Conference on Future Internet of Things and Cloud (FiCloud)* (pp. 189-196). IEEE.

Shukri, S. E., Al-Sayyed, R., Hudaib, A., & Mirjalili, S. (2021). Enhanced multi-verse optimizer for task scheduling in cloud computing environments. *Expert Systems with Applications, 168,* 114230.

Wang, W., Zeng, G., Tang, D., & Yao, J. (2012). Cloud-DLS: Dynamic trusted scheduling for Cloud computing. *Expert Systems with Applications, 39*(3), 2321-2329.

Xu, B., Zhao, C., Hu, E., & Hu, B. (2011). Job scheduling algorithm based on Berger model in cloud environment. *Advances in Engineering Software, 42*(7), 419-425.

Zissis, D., & Lekkas, D. (2012). Addressing cloud computing security issues. *Future Generation computer systems, 28*(3), 583-592.

Submitted: 17.06.2020
Accepted: 14.05.2021