Journal of Engineering and Applied Sciences 14 (2): 562-566, 2019
ISSN: 1816-949X

Using Simulation to Improve Max-Min Algorithm for Minimizing the Workflow Execution Cost in the Cloud

Ali S.A. Al-Haboobi, Mohammed R.A.M. Hammoodi and Ahmed Raheem Kadhim
Department of Computer Science, Faculty of Computer Science and Mathematics,
University of Kufa, Kufa, Iraq

Abstract: Cloud computing has demonstrated to be a modern model that offers IT assets depend on the pay-per-use basis as a service over the internet. Scientific workflow applications are benefiting from running on the cloud computing services. However, the challenge is to optimize workflow scheduling algorithms are still required an additional work. This study proposes max-min+ which is an extension for the max-min algorithm that is able to minimize the total execution time and cost of workflow execution. We tested the proposed algorithm via WorkflowSim with using 5 realistic scientific workflow applications. The results of max-min+ outperforms the original max-min algorithm by minimizing the overall workflow execution time.

Key words: Cloud computing, workflow, scheduling, execution time, cost, applications

INTRODUCTION

Recently, cloud computing (Buyya *et al*., 2009) has fast becoming a key instrument in the IT services. It offers IT assets for commercial and scientific customers in an operative way. It plays an important part in handling of scientific workflow applications using distributed systems (Deelman *et al*., 2006). Cloud computing is able to provide resources to companies for computational and storage purposes and therefore, their cost can significantly be decreased. For example, CyberShake and Montage are scientific workflow applications that need many resources for large data processing that is available in cloud computing (Bharathi *et al*., 2008).

Task scheduling is an important part in cloud computing that can achieve a high performance computing. As a result, design scheduling algorithms are required for achieving perfect mapping tasks to virtual machines. Several scheduling algorithms have been proposed such as max-min, HEFT and min-min.

The contribution of this work is to propose max-min+, an extension to the original max-min algorithm. It can decrease the total task execution time (makespan) when a workflow is running in the cloud.

Literature review: Recently, the scheduling of workflow applications have shown an increasing interest by researchers. Many researchers have designed algorithms for minimizing the total execution time and execution cost of workflows.

The RASA algorithm has proposed that integrated the min-min with max-min algorithms by Parsa and Maleki (2009). It combined them to gain their benefits and remove their drawbacks. For instance, if min-min schedules a task to a virtual machine depend on its work, then max-min schedules the following task to a virtual machine depend on its work. The same procedure is repeated until all submitted tasks are executed.

An improved max-min algorithm has presented by Elzeki *et al*. (2012). Each task is being scheduled on a Virtual Machine (VM) based on calculating the expected completion time of the submitted tasks on each VM. Next, the task with the maximum completion time is allocated to the slowest VM which has the overall minimum completion time. It achieved better results when the submitted tasks have comparatively different completion time and execution time.

Topcuoglu *et al*. (2002) has presented a heuristic algorithm called Heterogeneous-Earliest-Finish-Time (HEFT). The algorithm calculates the average of computational cost of each task and the communication cost between the virtual machines of two tasks. Then, the rank function is used to calculate the total of average computational cost and communication cost. Next, the tasks are sorted in non-ascending order by the rank function. The task with the highest upward rank value sets the highest priority and selects by the algorithm and maps to the resource which minimizes the earliest finish time. The Period Ant Colony Optimization (PACO)

Corresponding Author: Ali S.A. Al-Haboobi, Department of Computer Science, Faculty of Computer Science and Mathematics, University of Kufa, Kufa, Iraq

algorithm has presented based on the ACO algorithm by Sun *et al.* (2013). The results of PACO algorithm have outperformed the min-min algorithm.

The Hyper-Heuristic Scheduling (HHSA) algorithm has presented by Tsai *et al.* (2014). It is considered a solution for cloud computing and it can significantly reduce the makespan when compared with a scheduling algorithm such as min-min.

An enhanced MMST algorithm has presented depend on max-min by Ming and Li (2012) the reason that max-min cannot achieve better result. Therefore, MMST increases utilizing the resource of tasks as well as reducing their waiting time.

MATERIALS AND METHODS

Scientific workflow representation: Scientific workflow involves a collection of tasks which follow a specific order in processing. A workflow is frequently modelled as Directed Acyclic Graph (DAG), G = (T, E) where T is a collection of tasks = $\{T_1, T_2, ..., T_n\}$ and E is a collection of edges where each edge connects each two tasks and denotes their data dependency (Topcuoglu *et al.*, 2002). Each task in workflow can perform in case all its parents have completely processed.

Background: Generally, many scheduling algorithms have been presented for minimizing the makespan of workflow that is running in cloud. While the problem of scheduling tasks is identified to be NP-hard (Ullman, 1975) which means no algorithms can produce the optimal solution in polynomial time. In this study one of heuristic algorithm that can run in polynomial time and generate optimal results.

Max-min algorithm: Max-min is a heuristic algorithm that used to allocate tasks to the virtual machines depend on the minimum expected completion time. It is working in two stages where in the first stage the tasks are listed in order from the largest to smallest. While the virtual machines are arranged in order from the fastest to the slowest. In the second stage, it maps the largest task with the fastest virtual machine to obtain the minimum expected completion time. Then, it allocates the second largest task with the second fastest virtual machine and this process is repeated until all submitted tasks in tasks are executed (Dong and Akl, 2006).

Min-min algorithm: The min-min is also a heuristic algorithm that is able to minimize the makespan. It assigns the smallest task from all tasks to the fastest virtual machine from all virtual machines. Next, it maps the second smallest task from the remaining tasks to the second fastest virtual machine from the remaining virtual machines. It repeats the same approach until all submitted tasks are executed.

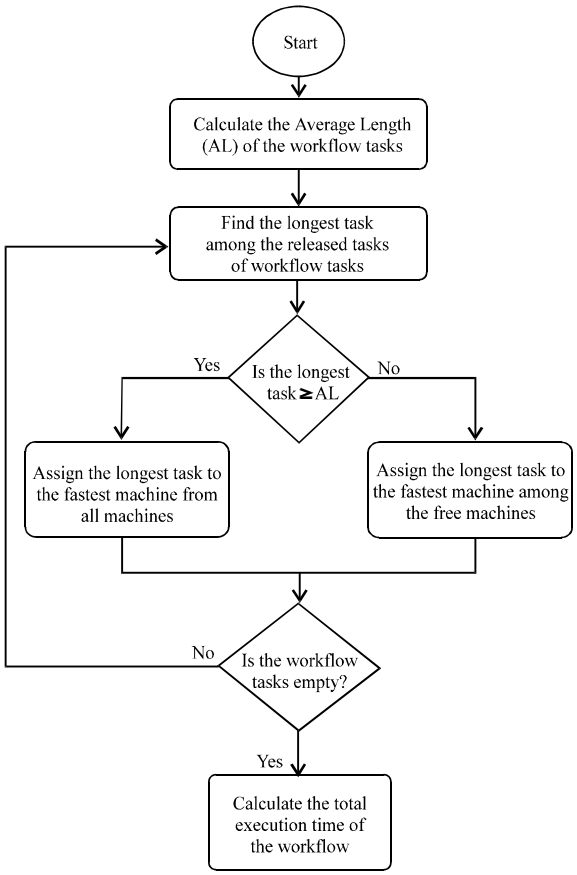

Fig. 1: Flowchart of the max-min+algorithm

Improving max-min: Max-min sets the highest priority to the largest tasks instead of the smallest tasks. Max-min assigns the task with the largest size to a machine with the fastest speed in order to obtain the minimum completion time. Then, the same process repeats until all submitted tasks of the workflow are executed. In case max-min is used in scheduling of a workflow, there are possibilities of scheduling a smaller task to a faster virtual machine as well as a larger task to a slower virtual machine. As a result, the overall computational cost will increase and consequently max-min needs improving for tackling these issues (Fig. 1).

To avoid these problems, the max-min+ algorithm has been presented depend on the original max-min algorithm. It can achieve better result than max-min by reducing the computational cost of workflow. Firstly, it computes the

average computational cost of the workflow tasks. Secondly, it selects the longest task from the released tasks of workflow. Thirdly, if the longest task is greater than or equal to the average of workflow tasks, it is scheduled to the fastest machine from the free machines. Otherwise, it is scheduled to the slowest machines from the free machines. The proposed of max-min+ algorithm is presented as a flowchart for workflow scheduling as shown in Fig. 1.

RESULTS AND DISCUSSION

Experimental setup: To assess the performance of the max-min+algorithm, the result of the proposed algorithm is used to compare with the result of max-min. The WorkflowSim 1.1.0 toolkit was used to simulate the proposed algorithm (Chen and Deelman, 2012). WorkflowSim is an extension of the CloudSim framework that is an open source (Calheiros *et al*., 2011). It can support algorithms in dynamic scheduling.

The experiment was performed on one data center and seven virtual machines. Each virtual machine has one CPU with 512 MB of RAM and bandwidth of 1000 Mbps. The speed of virtual machines was (5000, 500, 3000, 3000, 2000, 1000, and 500) MIPS, respectively. Five realistic scientific workflows have been used by the experiment as shown in Fig. 3. The scientific workflows involve: Montage, CyberShake, LIGO which are using in (Astronomy, Earthquake Science, Gravitational Physics), respectively while SIPHT and Epigenomics are using in biology (Bharathi *et al*. 2008). The following formula is used to compute the Average Length (AL) of the workflow tasks:

$$AL = \sum_{j=1}^{m} \frac{(T_j)}{m}$$

Where:

T_j = Symbolizes the task execution time of the jth task

m = The total number of tasks in workflow

Table 1 shows the results of max-min+ and max-min using WorkflowSim. The results of the graphical analysis used the bar chart with 14 tests as shown in Fig. 3-6. Figure 1-6 showed that the makespan of max-min+ is less than the makespan of max-min for all datasets and all workflows. In some datasets, the results of max-min+ has significantly reduced the makespan for Montage 25,

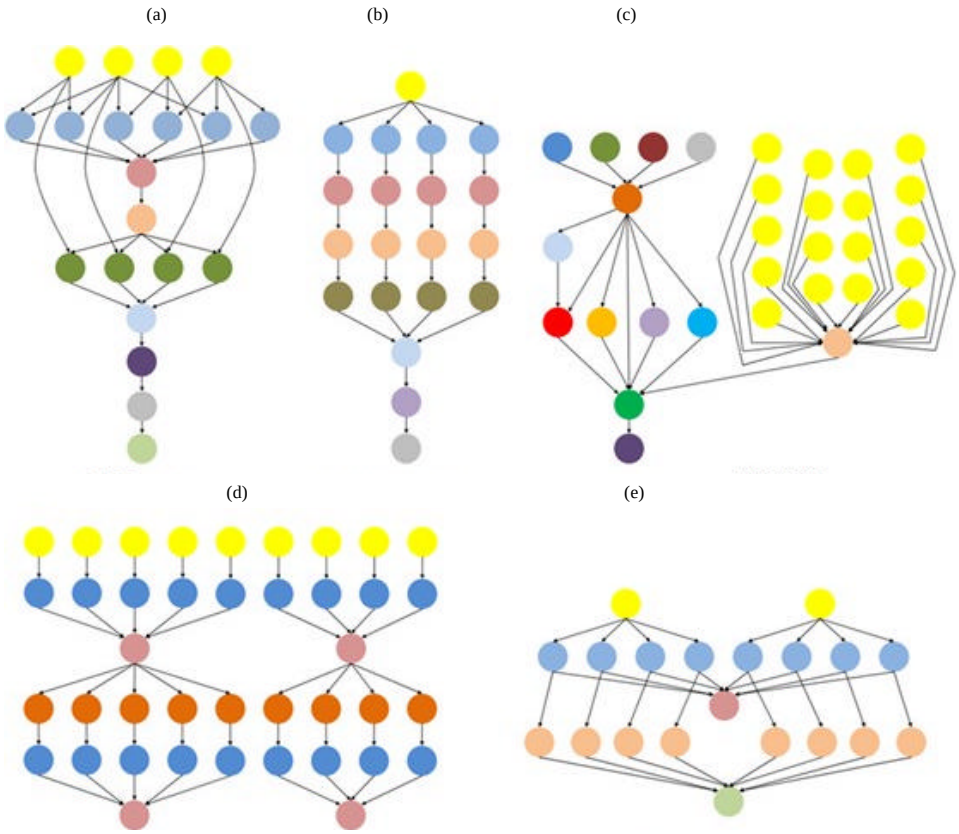

Fig. 2: The 5 realistic scientific workflows with their structure (Bharathi *et al*., 2008); a) Montage; b) Epigenomics; c) SIPHT; d) LIGO and e) CyberShake

Table 1: The comparison of total execution time of max-min+ and max-min algorithms in seconds using different numbers of tasks of five scientific workflows

Workflow	Max-min+	Max-min
Montage_25	23.48	40.56
Montage_50	41.56	73.08
Montage_100	97.01	98.17
Epigenomics_24	1905.22	2868.43
Epigenomics_46	3417.50	8578.78
Epigenomics_100	33833.60	39915.20
Sipht_100	1815.04	3211.54
Sipht_1000	12109.69	12410.58
LIGO_30	699.93	2073.35
LIGO_50	1016.86	1932.65
LIGO_1000	15554.41	15683.73
CyberShake_30	85.20	127.62
CyberShake_50	114.18	124.55
CyberShake_100	285.65	297.65

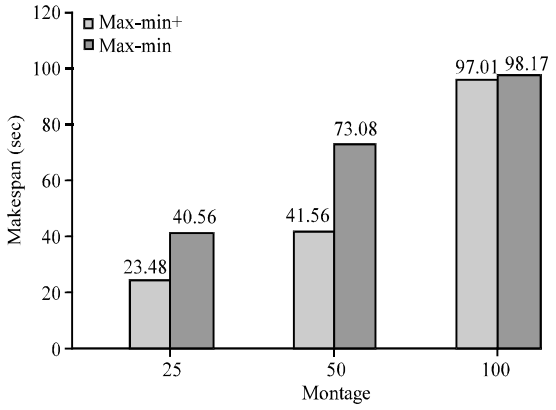

Fig. 3: The Montage workflows with the compared makespan of max-min+ and max-min

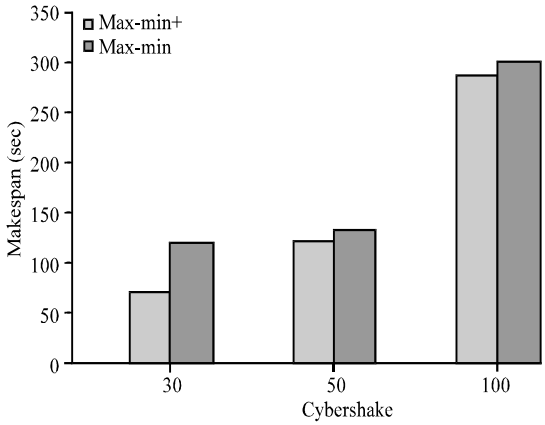

Fig. 4: The CyberShake workflows with the compared makespan of max-min+ and max-min

Montage 50, Epigenomics 24, Epigenomics 46, Sipht 100, LIGO 30, LIGO 50 and CyberShake 30 by approximately (42, 43, 33, 60, 43, 66, 47 and 33%), respectively.

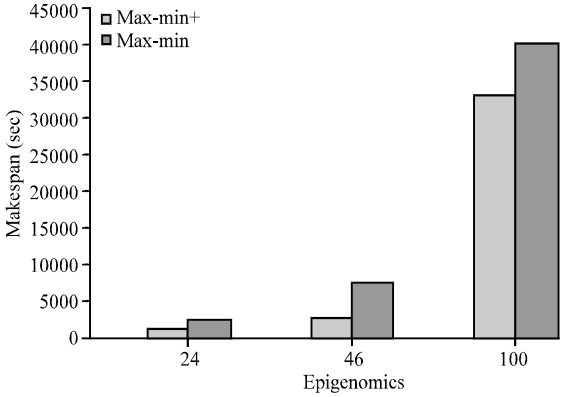

Fig. 5: The Epigenomics workflows with the compared makespan of max-min+ and max-min

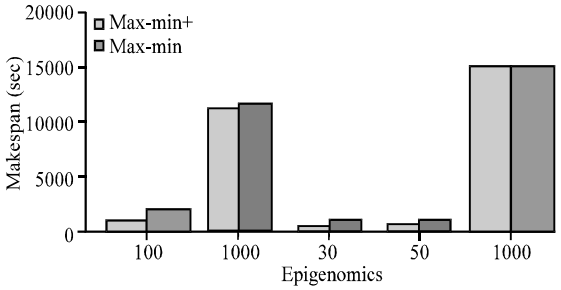

Fig. 6: The Sipht and LIGO workflows with the compared makespan of max-min+ and max-min

CONCLUSION

In the max-min algorithm, when a smaller task is allocated to a faster machine as well as a larger task is allocated to a slower machine, the overall makespan of workflow execution will increase. Consequently, we proposed an extension of the max-min algorithm based on the original max-min algorithm which is useful for small scale distributed environment. We evaluated it using WorkflowSim and the results show that max-min+ outperformed the original max-min algorithm in all cases. The research is dedicated on minimizing the overall total execution time of the workflow. Additionally, we plan to add more QoS parameters such as budget and deadline that user is able to deliver constraints to enhance the performance of a workflow executing in cloud.

REFERENCES

Bharathi, S., A. Chervenak, E. Deelman, G. Mehta and M.H. Su *et al.*, 2008. Characterization of scientific workflows. Proceedings of the 3rd Workshop on Workflows in Support of Large-Scale Science, November 17, 2008, IEEE, Austin, Texas, USA., ISBN:978-1-4244-2827-4, pp: 1-10.

Buyya, R., C.S. Yeo, S. Venugopal, J. Broberg and I. Brandic, 2009. Cloud computing and emerging IT platforms: Vision, hype and reality for delivering computing as the 5th utility. Future Gener. Comput. Syst., 25: 599-616.

Calheiros, R.N., R. Ranjan, A. Beloglazov, C.A.F. de Rose and R. Buyya, 2011. CloudSim: A toolkit for modeling and simulation of cloud computing environments and evaluation of resource provisioning algorithms. Software: Pract. Experience, 41: 23-50.

Chen, W. and E. Deelman, 2012. Workflowsim: A toolkit for simulating scientific workflows in distributed environments. Proceedings of the 2012 IEEE 8th International Conference on E-Science (E-Science'12), October 8-12, 2012, IEEE, Chicago, Illinois, ISBN:978-1-4673-4467-8, pp: 1-8.

Deelman, E., D. Gannon, M. Shields and I. Taylor, 2006. Workflows and e-Science: An overview of workflow system features and capabilities. Future Gener. Comput. Syst., 25: 528-540.

Dong, F. and S.G. Akl, 2006. Scheduling algorithms for grid computing: state of the art and open problems. Technical Report No. 2006-504, School of Computing, Queen, s University, Kingston. http://www.dcs.warwick.ac.ukhttp://cs.felk.cvut.cz/~xobitko/ga/about.html.

Elzeki, O.M., M.Z. Reshad and M.A. Elsoud, 2012. Improved max-min algorithm in cloud computing. Int. J. Comput. Applic., 50: 22-27.

Ming, G. and H. Li, 2012. An Improved Algorithm based on max-min for Cloud Task Scheduling. In: Recent Advances in Computer Science and Information Engineering, Qian, Z., L. Cao, W. Su, T. Wang, H. Yang (Eds.). Springer, Berlin, Heidelberg, ISBN:978-3-642-25788-9, pp: 217-223.

Parsa, S. and R.E. Maleki, 2009. RASA: A new task scheduling algorithm in grid environment. World Appl. Sci. J., 7: 152-160.

Sun, W., N. Zhang, H. Wang, W. Yin and T. Qiu, 2013. PACO: A period ACO based scheduling algorithm in cloud computing. Proceedings of the 2013 International Conference on Cloud Computing and Big Data, December 16-19, 2013, IEEE, Fuzhou, China, ISBN:978-1-4799-2829-3, pp: 482-486.

Topcuoglu, H., S. Hariri and M.Y. Wu, 2002. Performance-effective and low-complexity task scheduling for heterogeneous computing. IEEE Trans. Parallel Distrib. Syst., 13: 260-274.

Tsai, C.W., W.C. Huang, M.H. Chiang, M.C. Chiang and C.S. Yang, 2014. A hyper-heuristic scheduling algorithm for cloud. Cloud Comput. IEEE. Trans., 2: 236-250.

Ullman, J.D., 1975. NP-complete scheduling problems. J. Comput. Syst. Sci., 10: 384-393.