International Journal of Cloud Computing and Database Management

E-ISSN: 2707-5915
P-ISSN: 2707-5907
IJCCDM 2020; 1(2): 22-30
Received: 12-02-2020
Accepted: 16-03-2020

Maddela Kavya
Department of Computer Science, Sri Venkateswara University, Tirupati, Andhra Pradesh, India.

Corresponding Author:
Maddela Kavya
Department of Computer Science, Sri Venkateswara University, Tirupati, Andhra Pradesh, India.

A novel task scheduling algorithm with improved make span based on prediction of tasks computation time algorithm for cloud computing

Maddela Kavya

DOI: https://doi.org/10.33545/27075907.2020.v1.i2a.16

Abstract
In this paper new scheduling algorithm called Prediction of Tasks Computation Time algorithm (PTCT) to estimate minimum task execution time/Makespan time for cloud computing environment. Now-a-days all cloud service providers providing all resources to end users at very cheap rate and at the same time by designing scheduling algorithms cloud service providers are ensuring that all users can get response data in quick time. Various scheduling algorithms are implemented in cloud environment such MINMIN, MAXMIN, QOS GUIDE etc. MINMIN algorithm will schedule all task with less execution time first and then schedule remaining task. In simple terms MINMIN algorithm give priority to less execution time task. MAXMIN algorithm will schedule all task with more execution time and then schedule small execution time task. In simple terms MAXMIN give priority to high execution time first. Many more scheduling algorithms are there but above two algorithms are very much popular. This two algorithms will not look for resources which can take minimum execution time and propose PTCT algorithm will look for all resources/processors/machines and then form a matrix which contains estimated execution time for all jobs and then by applying PCA (Principal Component Analysis) algorithm it will predict or choose resource which took minimum execution time and then assign new task to that selected minimum execution time resource. Here resource could be computer or processor or Virtual Machine.
In propose PTCT algorithm we build an array with all task and processors as Directed Acyclic Graph (DAG) and then build a matrix with all processors and task. A matrix will contain estimated execution task time on each processor and all rows of a matrix will filled with all processor's execution time for all tasks. On generated matrix we will apply PCA algorithm to choose processor which take less execution time for selected task. This process continues till all task assigned to all processors. By applying PTCT algorithm we can further decrease computation and communication cost at cloud side. To implement this paper, we design 3 algorithms in the form of simulation and then compare execution/Make span time between them. In all 3 algorithms PTCT algorithms took less execution time for all tasks.

Keywords: Minimum Task Execution Time, Make span Time, Principal Component Analysis

Introduction
Cloud computing has grown to be a major technological enabler in companies and organizations [1, 2, 3]. It has been shown to increase reliability, deliver cost-cutting solutions, and provide 24/7/365 access to hard/soft resources from anywhere based on pay/use pricing policy [4, 5]. The cloud offers services in the structure of Software as a Service (SaaS), Infrastructure as a Service (IaaS), and Platform as a Service (PaaS) [3]. Task scheduling is a major challenge in widely distributed heterogeneous systems (e.g., cloud computing), which chooses the preeminent resources for a provided task. Also, in heterogeneous systems, task scheduling is more convoluted in comparison to homogeneous computing (HC) systems because of the various communication and execution rates amid various processors.
The main aim of cloud computing is to provide a highly efficient platform for appropriate exploitation of computational properties embedded in organizations, and to support the enterprise to capitalize on end-user demands [9]. However, the decentralized and heterogeneous nature of cloud networks makes them intricate to deal with. Last but not least, deciding on suitable assets for tasks has become an acute issue due to the swift rise of users and resources. For heterogeneous clustering systems, task scheduling is a computationally demanding problem, even under abridged conventions, as it is NP-hard [9, 12].

The overarching aim of this research is to improve the performance of task scheduling, while reducing computational costs. A key objective is to predict the ideal algorithm for incoming/available data as and when needed. In order to achieve this, we perform a systematic analysis of heuristic techniques for resource utilization by means of Principal Components Analysis (PCA) in the cloud environment. Moreover, we analyze the requirements and consequences of utilizing Quality of Service (QoS) with the proposed Prediction of Tasks Computation Time algorithm (PTCT).

Literature Survey

As talked about in Section 2, there exists an assortment of heuristic planning calculations, which can work in both group and online modes. A portion of these plans are fitting in heterogeneous booking situations, anyway they can't generally accomplish great make span, speedup, decreased expenses and expanded productivity [6, 7, 8, 1, 13]. Henceforth, QoS-based procedures are basic in getting the most extreme targets in order to hold QoS qualities for the two errands and assets.

Wang and Yu [29] propose an upgraded Min-Min calculation to think about the capability of undertaking planning for distributed computing. As recently demonstrated, the Min-Min calculation initially decides the undertakings with shorter execution times and afterward the assets which bring about the most limited occasions. This can prompt postpones while looking at the utilization of the calculation in the cloud condition. Zhang *et al.,* [30] propose QoS limitations in the cloud condition as a standard for planning an undertaking in the Min-Min calculation, named Mul-QoS-Min-Min. The proposed calculation discovers assets with comparable assignments to convey task planning, at that point demands clients to complete their needs. The reproduction results demonstrate that the exhibition of the Mul-QoS-Min-Min conspire is improved as far as execution times, when benchmarked against the conventional Min-Min calculation.

Both Mao *et al.,* [31] support the Max-Min calculation so as to balance out the heap for the cloud. The calculation moderates a table that holds insights concerning task position and assesses the constant outstanding burden for virtual machines (VMs) with the evaluated task execution times. The Max-Min calculation supports the use of assets and diminishes task planning reaction time by utilizing VMs rather than traditional resources.

Li *et al.,* [32] plans errands utilizing improved max-min task booking then biggest undertaking is excessively huge contrasted with different assignments in Meta-task for this situation generally speaking make span is expanded in light of the fact that too huge assignment is executed by slowest asset.

Henning *et al.,* [34] study task planning for the equal strategy challenge with a fixed number of processors and the best timetable for superior results. They demonstrate this can be accomplished by mapping assignments to machines as per priority limitations. In [35], the creators propose an undertaking planning component for distributing figuring processors to a purported "task diagram layouts". Since the creators don't consider the system association as a standard, this is regarded one of the restrictions of their investigation. To conquer this restriction, Sinnen and Sousa [36] use arrange dispute in their errand booking strategy, without considering the expense charged to clients for utilizing these assets. Two variables must be considered in the distributed computing condition, i.e., elite of information move and fulfillment of spending requirements. The creators in [37] and [38] acquaint a cost-effective calculation with select the most fitting framework in a cloud situation to actualize the work process dependent on utilizing the cutoff time and cost sparing requirements. Li and Su [39] show a planning calculation, which can be applied in enormous diagram preparing, where both expense and timetable length imperatives are thought of. Be that as it may, their plan doesn't consider bombed gadgets.

Issues of errand booking have been broadly considered in the writing. True to form, a huge number of approaches have been proposed because of its urgent consequences for execution [9, 15]. The heuristic calculation dependent on list booking techniques [9] is one of the traditional planning calculations for cloud conditions. This gives low time multifaceted nature, anyway the restrictions of insignificant all-inclusiveness and poor intermingling have. In [42], the creators study load adjusting in the cloud condition to dodge issues, which may happen because of increment in power utilization, hub disappointment, and machine disappointment. Be that as it may, the exploration managed a predetermined number of parameters, e.g., there is no investigation on the impacts of dynamic booking, increment in the quantity of errands and machines, too the development of clients. In [43] extra parameters are thought of Advancement of errand planning is tended to by presenting the iterative determination administrator. Be that as it may, this investigation ignores the issue of burden adjusting. Shimada *et al.,* [44] proposed a novel calculation, which can move the assignment with the shorter way while killing excess undertakings. Be that as it may, the issue of the expansion in the quantity of machines as the quantity of undertakings builds stays an open test. In [45] the creators propose a model to build the general framework use, in any case, load adjusting and other execution parameters should be additionally improved. Different works, for example [52 - 54] investigate the participation and coordinated effort among cloud servers utilizing multi-operator ways to deal with best relegate assets to approaching undertakings.

Proposed work

This section introduces the general framework of the proposed PTCT algorithm, including algorithmic details.

In heterogeneous computing, effective task scheduling is of the utmost importance to increase the advantages of accomplishing an application. Consequently, the task scheduling problem has been widely studied and many algorithms have been proposed including list scheduling, clustering, and task duplication scheduling based on Genetic Algorithm. In summary, list-scheduling algorithms are ideal in delivering low cost solutions, in comparison to other approaches. Clustering algorithms perform better in the case of homogeneous processors. Finally, task duplication scheduling algorithms are utilized for communication intensive programs. A point to note is that a review of the open literature on task scheduling revealed a number of enhancements for homogeneous processors [8, 10, 16-18], however there appears to be less progress in the case of heterogeneous processors [19–22]. This provides further motivation for the development of our proposed framework in the context of a heterogeneous environment

Table 1: Computation costs of tasks in Fig 1

Edge (E)	Node (V)
(1, 2)	7.48
(1, 3)	0.48
(2, 3)	6.75
(1, 4)	8.85
(1, 5)	2.52
(1, 6)	1.39
(2, 7)	8.71
(2, 8)	9.49
(4, 8)	6.75
(6, 8)	5.51
(4, 9)	8.71
(5, 9)	0.48
(3, 10)	9.493
(7, 10)	6.75
(8, 10)	8.27
(9, 10)	5.69
(10, 10)	4.58
(5, 11)	6.73
(8, 11)	1.12
(10,11)	3.77

Consider the following two attributes, Earliest Start Time (EST) and Earliest Finish Time (EFT), used to outline the objectives of the task scheduling issue. EST (VI, PJ) represents the EST for task VI on processor PJ, and similarly, EFT (VI, PJ) represents EFT for task VI on processor PJ. EST (VI) and EFT (VI) represent the values of these attributes over the set of processors, respectively. For any initial entry task, Ventry, EST (Ventry) = 0, the values of EST and EFT are calculated from the entry to the exit tasks, traveling the task graph from top to bottom. All immediate predecessor tasks of VI should be scheduled to allow the calculation of EST.

4. Results and discussions

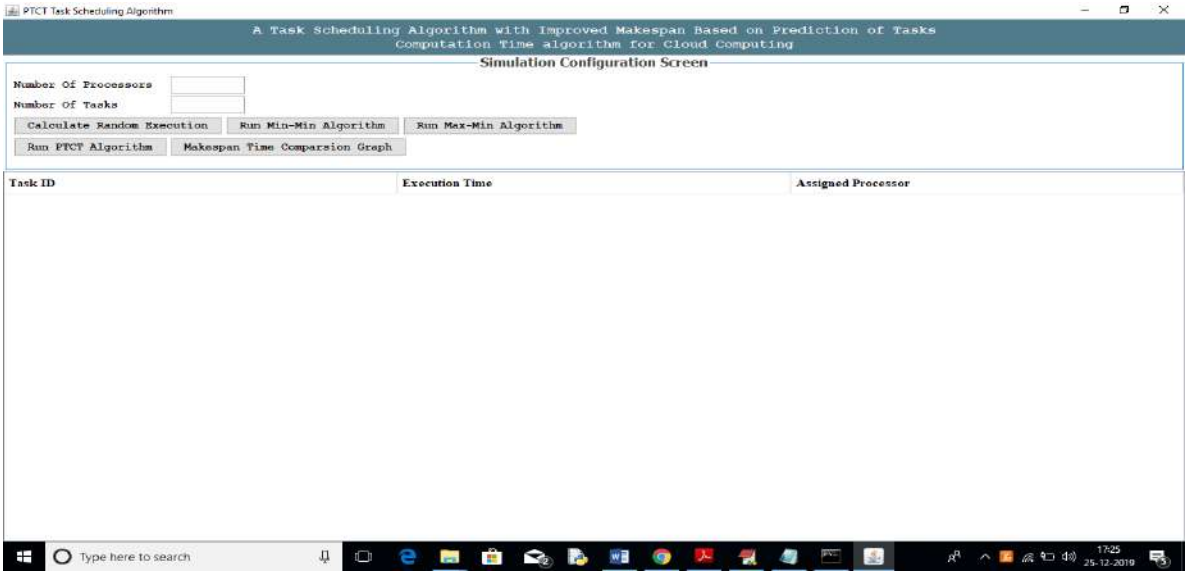


Fig 1: In above screen enter number of processors and number of tasks

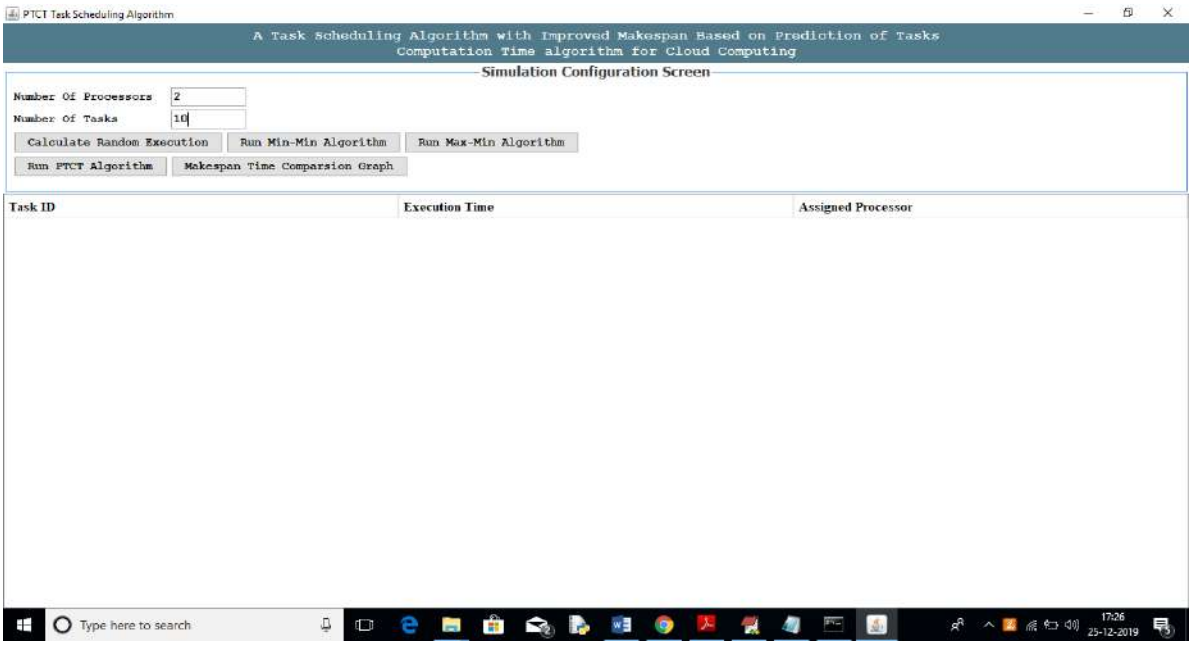


Fig 2: In above screen I entered number of processors as 2 and number of task as 10 which means all 10 tasks has to schedule and run in given 2 processors. Now click on 'Calculate Random Execution Time' button to assign some execution time to each task and based on this execution time algorithms will schedule tasks to processors See below screen

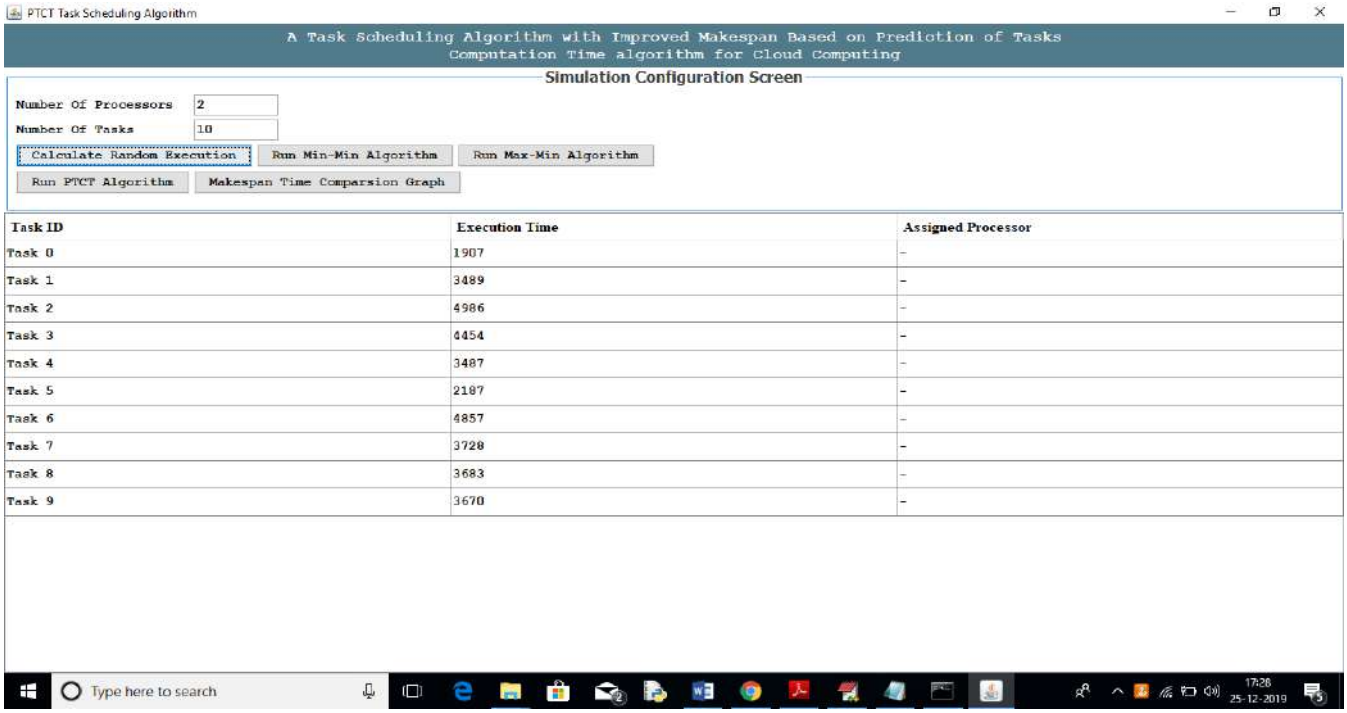
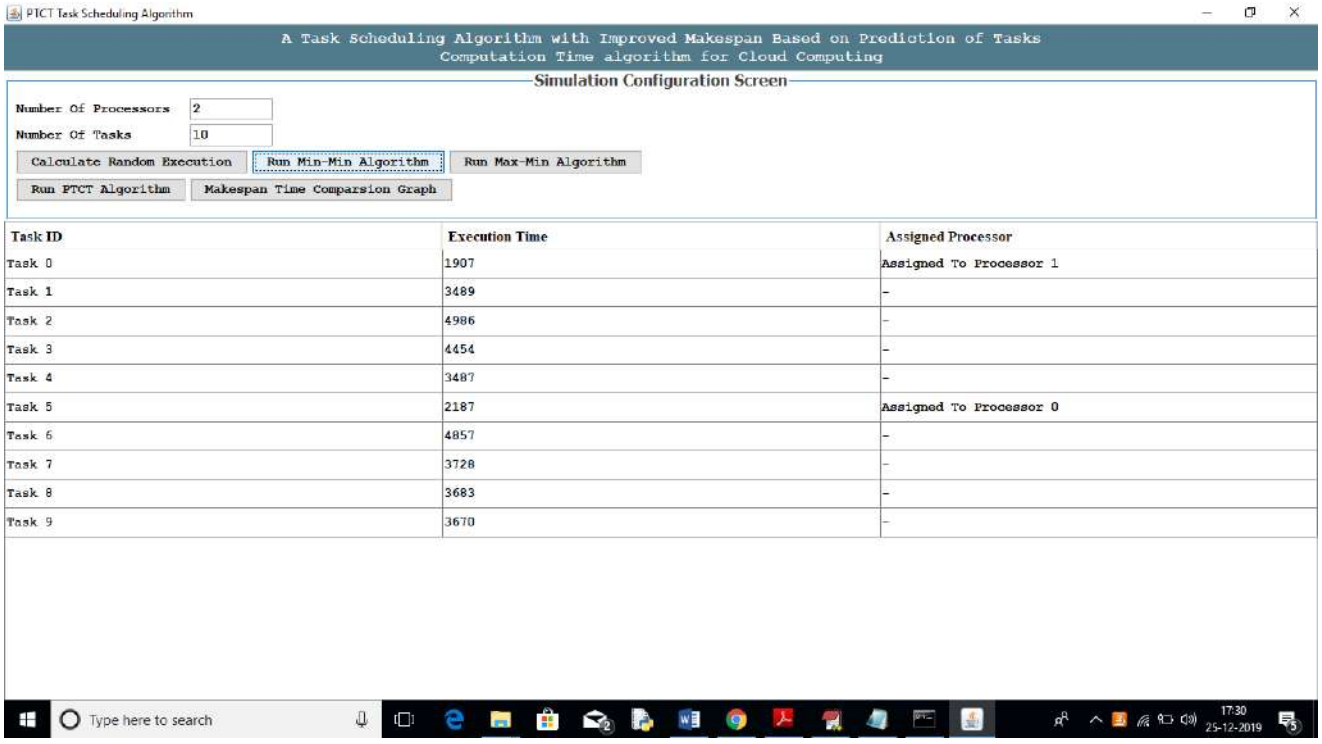


Fig 3: In above screen all 10 tasks got some random execution time and now click on 'Run Min-Min Algorithm' button to schedule this 10 tasks to 2 processors. We can see MINMIN will schedule less execution time task first, in third column empty value is there as processor not yet assign to task



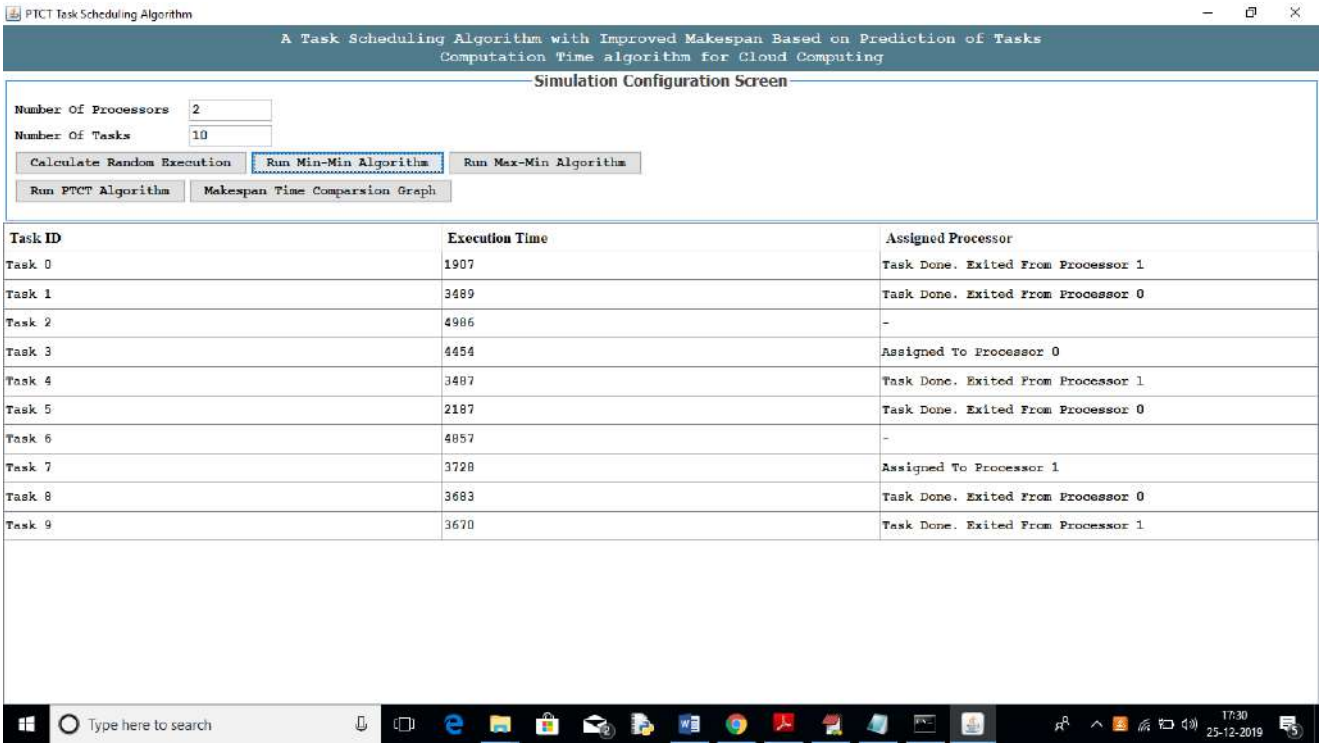


Fig 4: In above 2 screen we can see MINMIN scheduling task based on freeness of resource and execution time. In third column we can see task is assign to which processor and after task completion we will get message as task done on which processor. After all task execution we will get total execution time for all tasks. See below screen

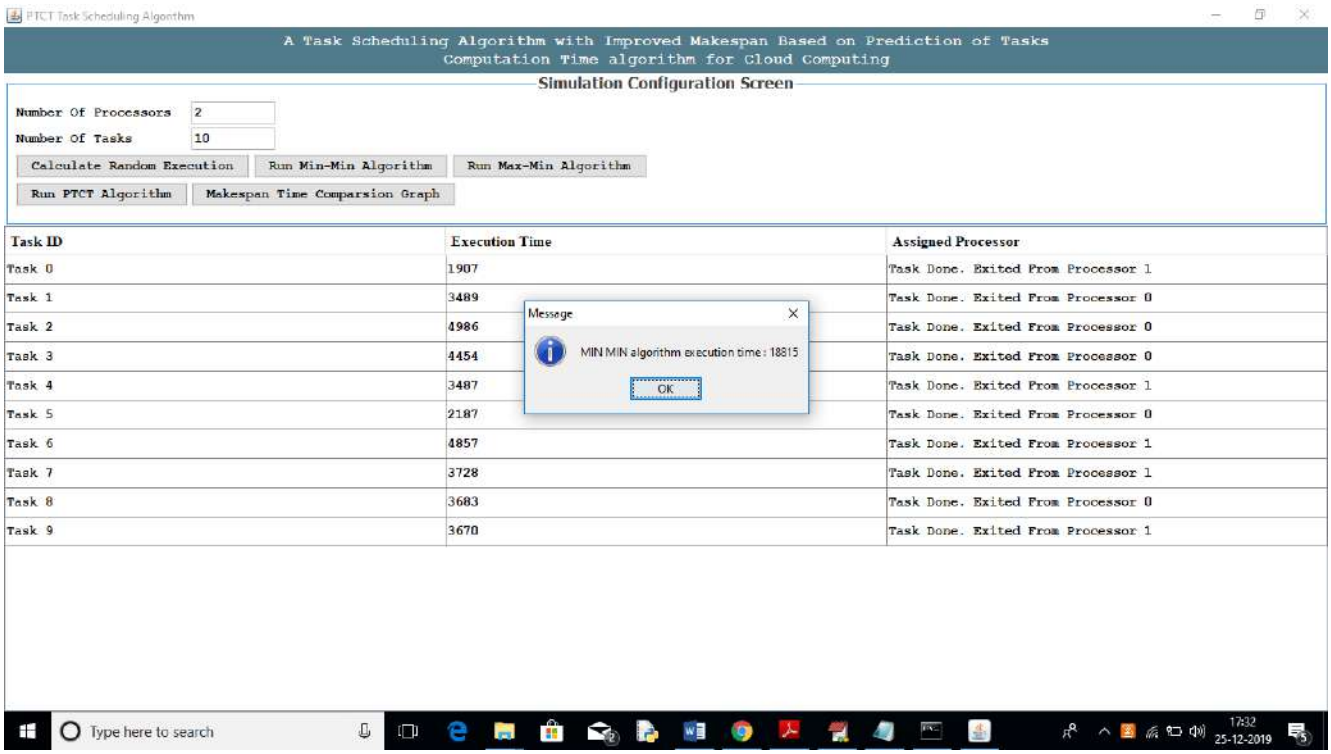


Fig 5: In above screen we can see MINMIN took 18815 MILLI seconds to complete all tasks. Similarly click on 'Run Max-Min Algorithm' button to schedule all tasks based on MAXMIN algorithm

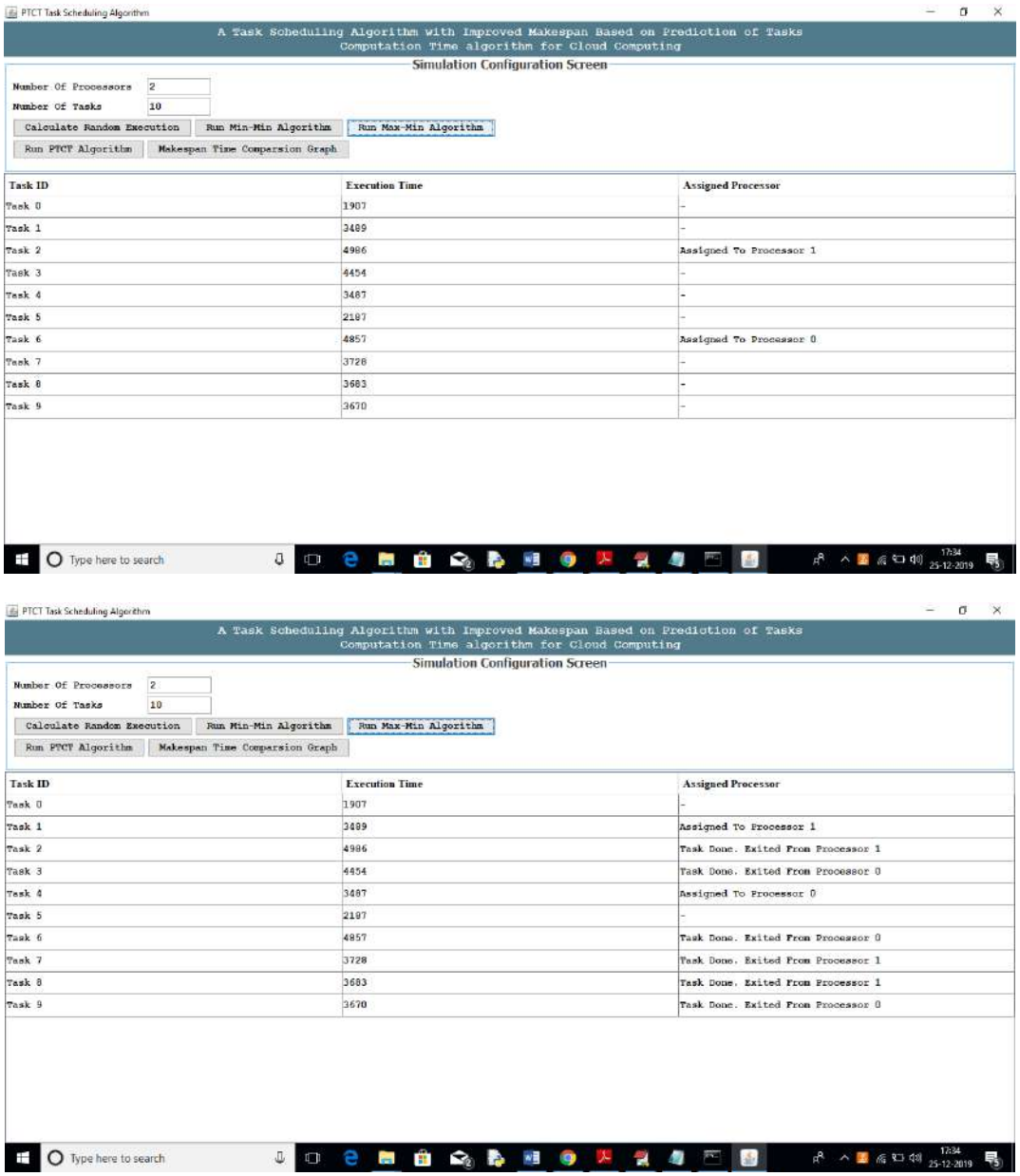


Fig 6: Above two screen showing scheduling process of MAXMIN algorithm and below is MAXMIN algorithm total execution time

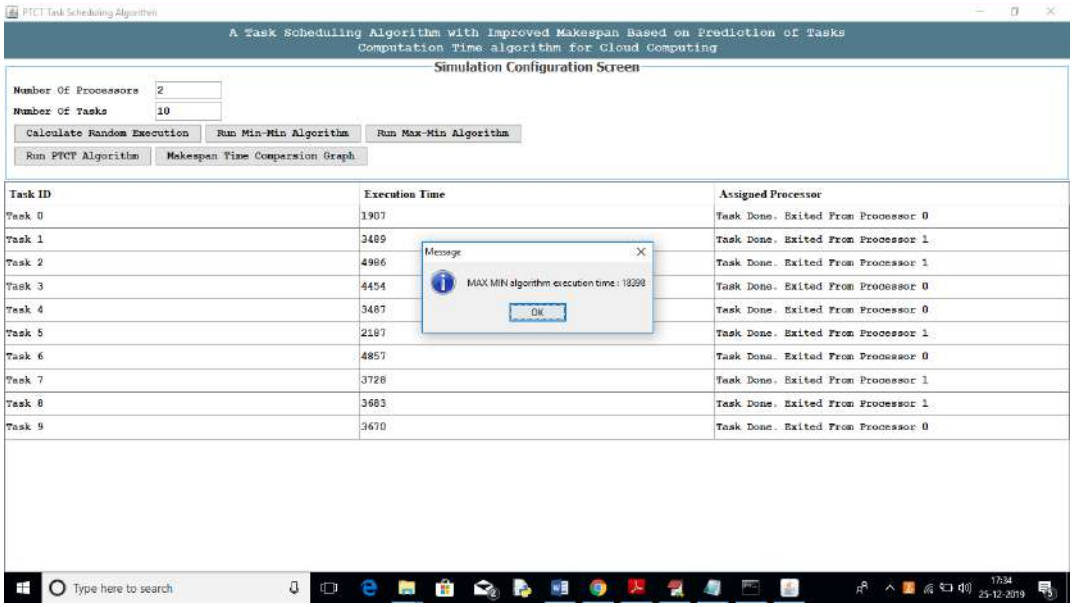
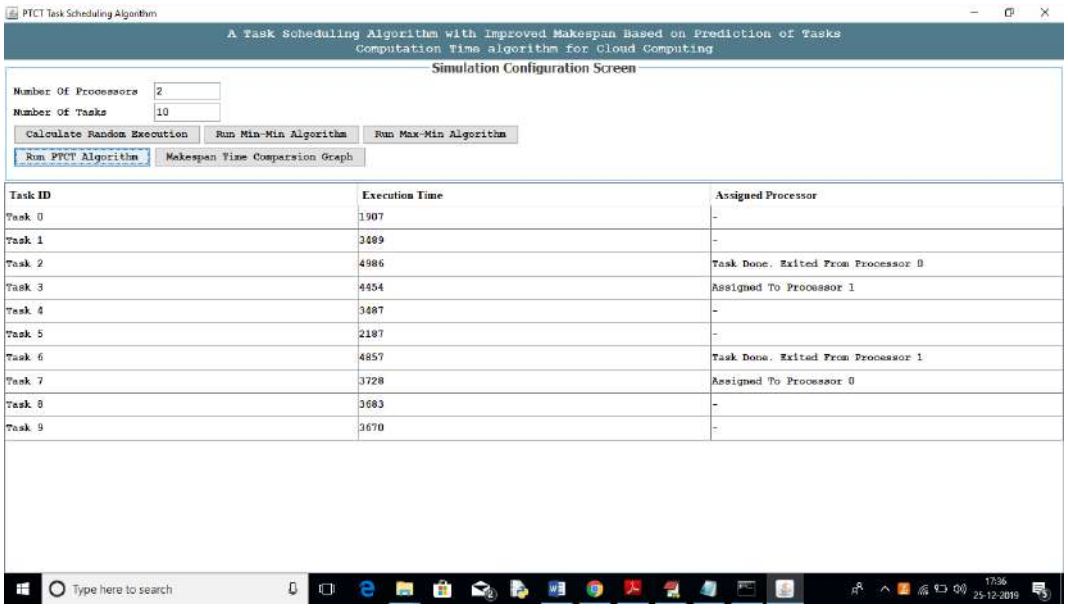


Fig 7: In above screen we can see MAXMIN took 18398 Milli Seconds to complete all tasks and we can say MAXMIN took less time compare to MINMIN. Similarly click on 'Run PTCT Algorithm' button to schedule tasks based on PTCT algorithm concept.



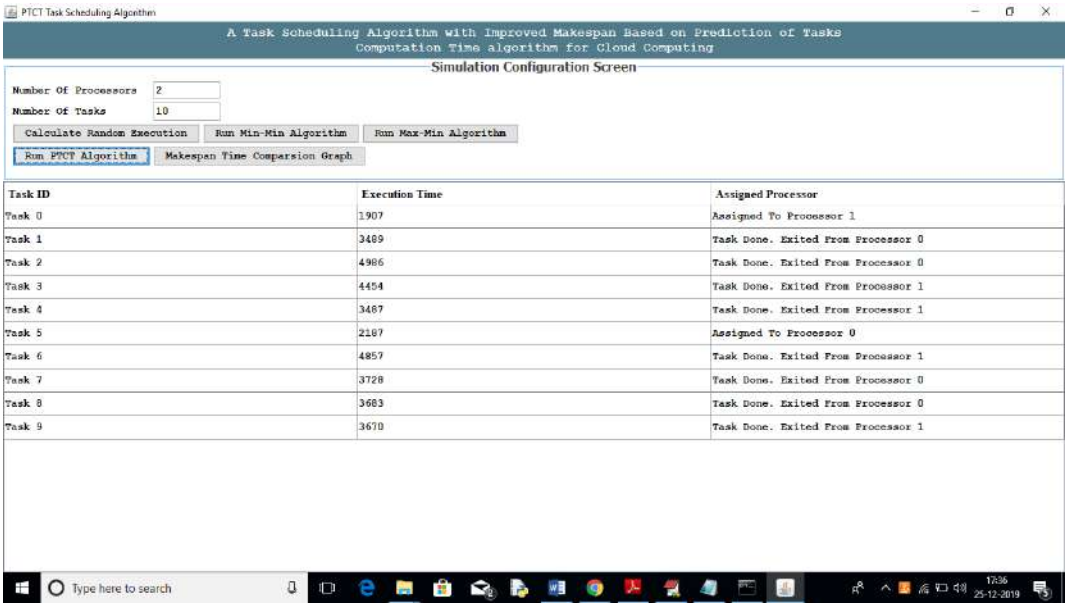


Fig 8: Above two screen showing PTCT scheduling output and below is total PTCT execution time

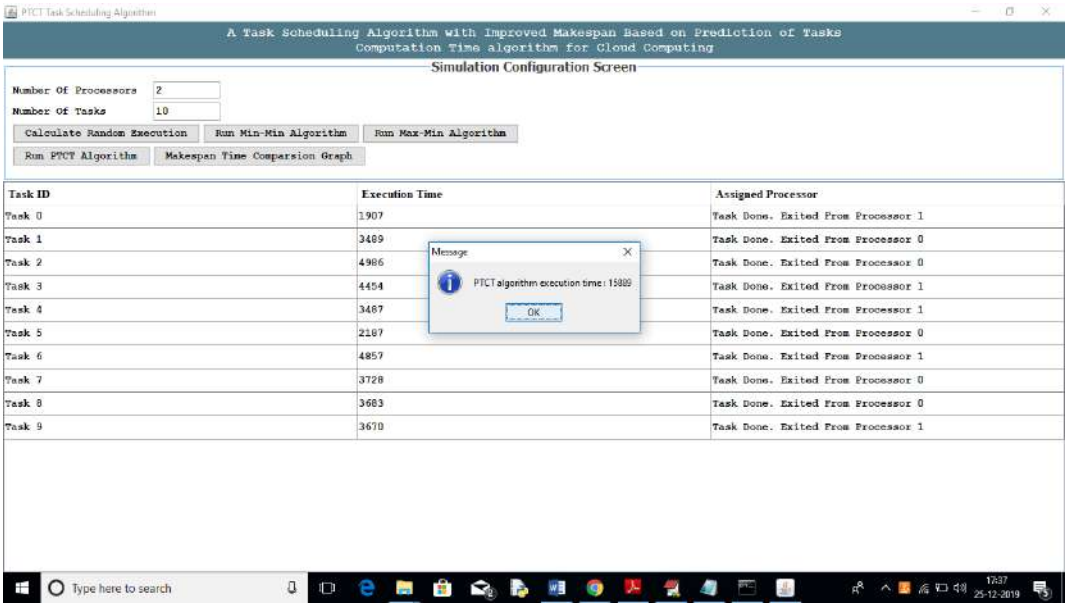


Fig 9: In above screen we can see PTCT took 15889 Milli Seconds to complete all tasks execution and is better than other 2 algorithms. Now click on 'Makespan Time Comparison Graph' button to see all algorithms execution time graph

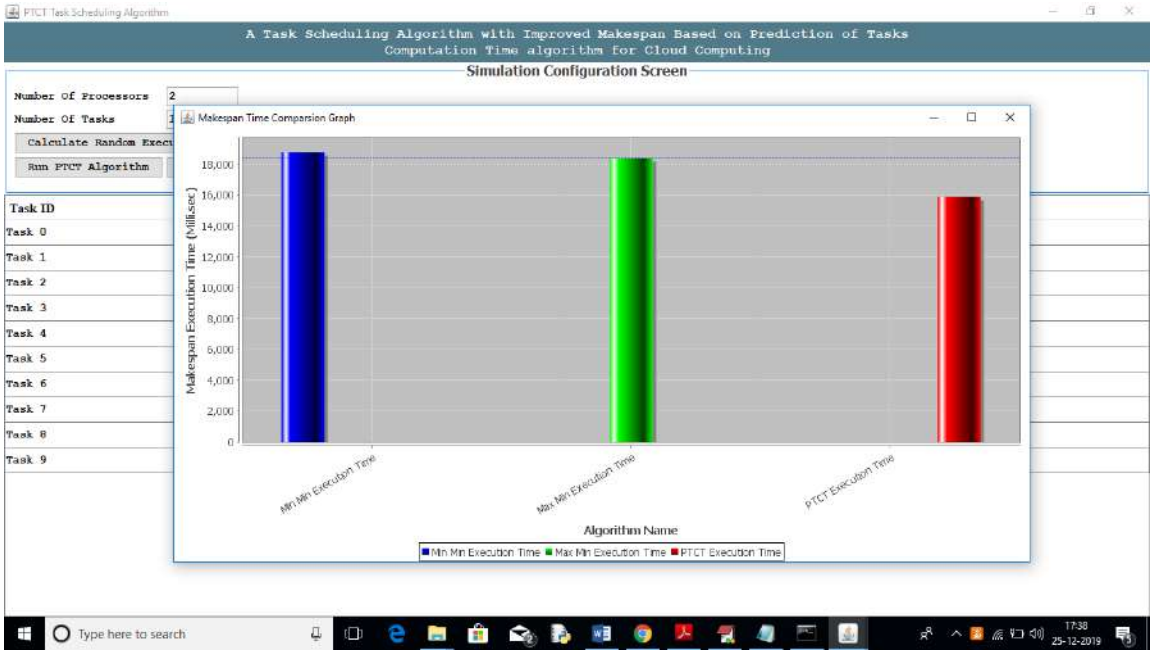


Fig 10: In above graph x-axis represents algorithm name and y-axis represents execution time in MILLI seconds, From above graph we can conclude PTCT propose algorithm better than other 2 algorithms, this code is dynamic so u can give any number of tasks and processors

Conclusion

In this proposed work, a novel algorithm, Prediction of Tasks Computation Time, was presented. This results in a performance improvement in cloud-based task scheduling by using Principal Component Analysis. This permits the reduction of the size of the Expected Time to Compute (ETC) matrix. The proposed algorithm was applied to simulated task graphs, and its performance was assessed in terms of speed-up, make span, schedule length ratio and efficiency. The simulation results showed improved performance, when benchmarked with four state-of-the-art scheduling algorithms, namely Min-Min, Max-Min, QoS-guided and MIM-MAM. In the cloud computing context, the simulation results indicated that the proposed PTCT can reduce the overall make span and task execution time. The simulation setup was based on static scheduling, where task arrival at the processors and speed are assumed to be known. Future work will consider dynamic scheduling for real-world application graphs and benchmarking in real-world problems. The focus will be on improving the total energy utilization and consumption of task scheduling using the PTCT algorithm and comparing the findings with relevant state-of-the-art algorithms for cloud energy consumption, such as Gree Di and Gree AODV [47 – 51].

References

1. Avetisyan, Arutyun I, *et al*. Open cirrus: A global cloud computing testbed" Computer. 2010; 43.4:35-43
2. Panda SK, Jana PK. “Efficient task scheduling algorithms for a heterogeneous multi-cloud environment”, J Supercomputer. 2015; 71(4):1505-1533.
3. Buyya, Rajkumar, *et al*. "Cloud computing and emerging IT platforms: Vision, hype, and reality for delivering computing as the 5th utility" Future Generation computer systems. 2009; 25.6:599-616.
4. Beaty Kirk A, Vijay K Naik, CS Perng. "Economics of cloud computing for enterprise IT", IBM Journal of research and development. 2011; 55.6:12-1.
5. Foster I, Zhao Y, Raicu I, Lu S. Cloud computing and grid computing 360-degree compared. Grid Computing Environments Workshop, IEEE, 2008, 1-10.
6. Masood Anum, *et al*. HETS: Heterogeneous Edge and Task Scheduling Algorithm for Heterogeneous Computing Systems. Proceeding of 2015 IEEE 17th International conference on high-performance computing and communications, 2015 IEEE 7th International Symposium.
7. Hoffmann R, Prell A, Rauber T. Dynamic task scheduling and load balancing on cell processors, in: 18th Euromicro International conference on parallel, distributed and network-based processing (PDP), 2010, 205-212.
8. Munir E, Ullah, Jianzhong Li, Shengfei Shi. QoS sufferage heuristic for independent task scheduling in grid, Information technology Journal. 2007; 6(8):1166-1170.
9. Buyya R, Yeo C, Venugopal S, Broberg J, Brandic I. Cloud computing and emerging IT platforms: vision, hype, and reality for delivering computing as the 5th utility, Future generation computer systems 2009; 25(6):599-616.
10. Bawa, Rajesh Kumar, Gaurav Sharma. Modified min-min heuristic for job scheduling based on QoS in Grid environment, Information management in the knowledge economy (IMKE), 2nd International Conference on, IEEE, 2013.
11. Napper J, Bientinesi P. Can cloud computing reach the top500? In proceedings of the combined workshops on unconventional high performance computing workshop plus Memory access workshop, Ischia, Italy, 2009, 17-20.
12. WANG En Dong, LI Xu. QoS-oriented monitoring model of cloud computing resources availability”, International conference on computational and information sciences, 2013.
13. Chiyu Zhang, Ran Huang, Jinhui Zhang. Distributed adaptive consensus tracking of unknown heterogeneous linear systems via output feedback, Proceedings of the

35th Chinese control conference. 2016; 27-29. Chengdu, China.
14. Beheshti Z, Shamsuddin SMH. A review of population-based meta-heuristic algorithms, Int J Adv Soft Comput Appl. 2013; 5(1):1-35.
15. Feng, Chen, Hong Xu, and Baochun Li. An alternating direction method approach to cloud traffic management, arXiv preprint arXiv. 1407, 8309, 2014.
16. Begum Suriya CSR. Prashanth. Stochastic based load balancing mechanism for non-iterative optimization of traffic in cloud. Wireless Conference on, IEEE, 2016.
17. Smirnov, Andrey V *et al*. Network traffic processing module for infrastructure attacks detection in cloud computing platforms, soft computing and measurements (SCM), XIX IEEE International Conference on, IEEE, 2016.
18. Kang Lu, Xing Ting. Application of adaptive load balancing algorithm based on minimum traffic in cloud computing architecture Logistics, informatics and service sciences (LISS), International Conference on, IEEE, 2015.
19. Rajendra Sahu, Anand K Chaturvedi, ABV-IIITM Gwalior, India, Many-Objective Comparison of Twelve Grid Scheduling Heuristics, International Journal of Computer Applications (0975–8887), Volume 13–No.6, January. 2011; 13(6).
20. Amudha (Assistant Professor) T, Dhivyaprabha (MPhil Research Scholar) TTT, QoS priority based scheduling algorithm and proposed framework for task scheduling in a grid environment, Department of computer application, school of computer science & Eng, Bharathiar University, Coimbatore – 46, IEEEs-International conference on recent trends in information technology, ICRTIT 2011, MIT, Anna University, Chennai. June, 2011, 3-5.
21. Patel, Gaurang, Rutvik Mehta, Upendra Bhoi. "Enhanced load balanced min-min algorithm for static Meta task scheduling in cloud computing", Procedia Computer Science. 2015; 57:545-553.