DAFFODIL INTERNATIONAL UNIVERSITY JOURNAL OF SCIENCE AND TECHNOLOGY, VOLUME 18, ISSUE 2, JULY 2023
ISSN 1818-5878 (Print) 2408-8498 (Online)

A MODIFIED THROTTLED LOAD BALANCING ALGORITHM TO ACCELERATE CLOUD SYSTEM BY REDUCING RESPONSE TIME

*Md. Toufecul Islam[1], Prof. Dr. Mohammod Abul Kashem[1], and Tanjina Jahan[1]
[1]Department of Computer Science and Engineering, Dhaka University of Engineering and Technology

Email: touficcse@gmail.com

Abstract: The load balancing becomes an important factor in maintaining system stability and performance. As an effect, a strategy for improving system performance by balancing workload across Virtual Machines (VMs) is required. To accomplish load balancing and Quality of Service, scheduling algorithms are utilized. The Modified Throttled Load Balancing Algorithm (MTLBA) reduces response time as well as manages and balances load amongst virtual machines. Here used the Cloud Analysts simulation toolkit to test the modified technique. The results of our MTLBA were likened to the existing Throttled Load Balancing Algorithm. The results demonstrated that the suggested MTLBA outperforms existing Throttled algorithms.

Keywords: *Processing Time, Cloud Computing, Response Time, Load Balancing.*

1. INTRODUCTION

As the need for accessing shared resources and services on the Internet increases fast, assuring the competency of time, service grade, and cost-savings have become the most critical objectives [1]. To address this need, the Cloud Computing paradigm was introduced in June 2007. Shortly afterward, with the help of major corporations such as Google, Microsoft, IBM and many more, Cloud Computing is encouraged to flourish. Cloud computing (Usually referred to as "the cloud") is the movement of constructing computer networks, other distributed computing infrastructure to pool together diverse computing resources as needed [2]. It paves the way for customers to provide services, release resources easily, avoid supplier contacts, and pay only for whatever they consume. (Pay-by-use) [3]. The load balancer, Quality of Service (QoS) and system models essentially represent the quality of service in the cloud, as stated by [4]. From this, it is clear that to guarantee a performance of high level in the cloud, balance of the load is an essential tool, as well as one of the opportunities for study into cloud technology [5]. In order to boost the efficiency of cloud-based features, the utilization of resources must address fundamental problems like allocation, response, connectivity, unused resource discovery, modeling resources, resource provisioning, and resource utilization planning [6]. Specifically, the method for using resources is based on the number of connections over time along with service processing time [7]. From there, we may examine ways to reduce processing time and improvise a methodology for load balancing request allocation. This is one of the studies aimed at enhancing cloud technology and making it more sophisticated and flawless. Cloud computing has passed through various stages of development since the 1980s, including layout and computing for utility reasons, Application vendors, and software as a service. The phrase "cloud computing" did not develop significantly until 2006. At this period, Amazon launched its “Elastic Compute Cloud”, which provides customers with the ability to rent computer and data processing capacity for the purpose of running their business applications. Additionally, Amazon commences the provision of commercial apps that could be accessed via web browsers. Shortly after a span of three years, Google App Engine was launched in 2009, marking a significant milestone in the development of cloud computing infrastructure. But in association with the ongoing progress, this also ensue certain issues with cloud computing, particularly the issue of excessive load of cloud [8]. Consequently, several algorithms have developed to address the issue of distributing the load on cloud [9]. In this research, there present the Modified Throttled Load Balancing Algorithm (MTLBA), which improves load balancing by reducing response and data center processing time.

2. RELATED WORK

D. A. Shafiq et al [10] has researched and assessed the techniques to resource management in cloud. Where the workload distributions amongst processors are the same but the processing times for distinct processes are different. Therefore, at any moment, certain nodes may be substantially busy while others are idle. Shubham Sidana et al [11] has proposed

Mid-Point Algorithm (MPA) for load balancing in a cloud-based environment. In this algorithm, load balancing is attempted by sorting the flow of Virtual Machines and amount of time of the cloud. The Virtual Machines and Cloudlet lists are transmitted to Broker for deployment. The broker receives the inventory of virtual machines and cloudlets after that. Broker manages resources using the mid-point algorithm. The technique rearranges the virtual machine list and the list of cloudlets until there is exactly one item on each list. This algorithm allocates resources to low-capacity devices in a manner that requires less processing work, and vice versa. This algorithm is limited in that there is no mechanism for active migration (directly transferring requests for virtual machines). The limitation of this method is that there is no direct way to transfer requests for virtual machines. M. van der Boor et al [12] present two modifications of regular Join-Idle-Queue (JIQ) method in which tokens are allocated inconsistently or infrequently traded from several dispatchers. For work distribution, the credentials of inactive servers are used by Join the Inactive Queue (JIQ) algorithms. In particular, JIQ strategies entail minimal transmission of information while achieving zero obstruction and delay time within the limit of numerous servers. Consequently, the author made use of product form structures and constraints on fluid to present that, even for arbitrarily low total loads, the JIQ technique unable to attain zero obstructing and delays, by utilizing product-form representations and fluid limits. Surprisingly, the least-busy dispatcher suspends servers that aren't actively processing token requests and slows down token delivery, thereby functioning as a bottleneck. JIQ's advancements have increased the number of large-scale systems. Kalka and Mohit [13] changed the inefficient HEFT technique for distributing work across CPU cores, thereby decreasing program loading time based on the processor's order and assigned tasks. Among HEFT CPOP and proposed algorithm, the proposed one reduces task duration and load balancing issues. Mehdi Sookhak et al [14] presented an efficient far data method to validate the authenticity of cloud-stored data. They utilized outsourced algebraic property data blocks to remotely verify file authenticity and minimize server and client computational expenses. They intended to implement a new data structure that partitions and subjugates the table for effective data updating that requires minimal computation and auditor of communications costs. Abdalkafor et al [15] researched and specified the parameters that would enhance the functioning of the cloud system. Hieu N. Le et al [16] described the throttled load balancing algorithm for cloud based system. Due to scanning of all VMs, there needs a huge time for resource allocation. K. Dubey et al [17] discussed about an efficient approach for Virtual Machine distribution in cloud computing. S. Devi et al [18] present the deployment of a cloud-based Internet of Things preservation strategy that makes advantage of the copious quantities of data created by IoT. N. Xuan Phi et al [19] proposed an efficient modified load balancer for cloud. But in that algorithm if no VM is available there, the whole process starts from the initial stage.

From the above background study, there addressed utilization of resources, balancing workload and response time are regarded as key challenges for cloud computing. Load balancing becomes a crucial aspect of stabilizing the cloud system. With these directions, to achieve minimum response time, modification of Throttled Load Balancing Algorithm will be modified.

3. THROTTLED LOAD BALANCING ALGORITHM (TLB)

According this method, the load balancer manages a database of virtual machine indices including their state of availability (Available or Busy). In the first step, the client/server communicates with the data center to request a certain VM. The data center will often request virtual machine distribution using a load balancer. The load balancer performs top-to-bottom searches of the index table until the first accessible virtual machine is discovered or the index table is completely searched. If the data center identifies a virtual machine, it will transmit the request to that machine. The data center also verifies the load balancing of new distributions and updates the index table appropriately. If the requested virtual machine (VM) cannot be found, the load balancer will send a "-1" response back to data center. Data center is liable for processing the center's request.

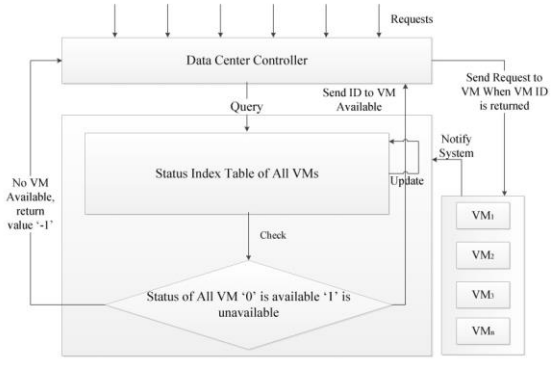

Fig. 1. Throttled Load Balancing Algorithm

DAFFODIL INTERNATIONAL UNIVERSITY JOURNAL OF SCIENCE AND TECHNOLOGY, VOLUME 18, ISSUE 2, JULY 2023
ISSN 1818-5878 (Print) 2408-8498 (Online)

4. MODIFIED THROTTLED LOAD BALANCING ALGORITHM (MTLBA)

Steps of MTLBA:

Step 1: By maintaining and updating two index tables and a waiting queue, the MTLBA Load Balancer is able to balance or disperse load.

• Available Index: The status of VMs in the available index table is '0'

• Unavailable Index: The status of VMs in the available index table is '1'

At the onset of this algorithm, all virtual machines are added to the "Available Index" table, while the "Unavailable Index" table remains devoid of entries.

Step 2: A fresh request has been received by the Data Centre Controller.

Step 3: Upon receipt of the request, the Data Center Controller initiates a query to the MTLBA Load Balancer to obtain subsequent allotments.

Step 4: MTLBA Load Balancer notices and transfers the ID of Virtual Machine from the "Available Index" table.

• The Data Center Controller transmits the request to the selected VM identified by its ID.

• The Data Center Controller reports the MTLBA Load Balancer for a new allotment.

• The MTLBA Load Balancer will revise this VM into the Unavailable Index and delay for the new request from Data Center Controller.

If the table "Available Index" is blank (all VMs is busy or unavailable):

• MTLBA Load Balancer will transfer the request to the waiting queue.

• Requests are organized by the Data Center Controller.

Step 5: After concluding the request and receiving the response from VM, the Data Center Controller will inform the MTLBA Load Balancer and then update the "Available Index" table.

Step 6: The Data Centre Controller will perform Step 3 repeatedly until the "Available Index" table is vacant if the number of requests is enormous.

In MTLBA if VMs are unavailable then after it become available again than the ID of VMs are transferred to the waiting queue. This increases the system's overall effectiveness.

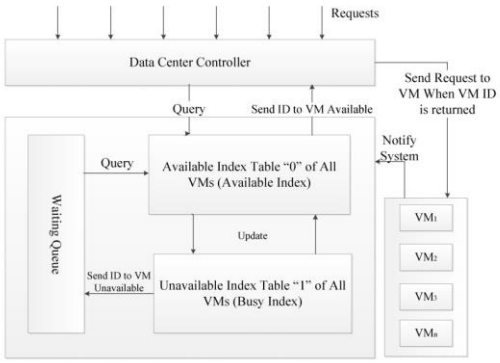

Fig. 2. Modified Throttled Load Balancing Algorithm (MTLBA)

5. THE EVALUATION AND SIMULATION

In this paper, the technique evaluates using the "Cloud Analyst Simulation Toolkit". We evaluate the characteristics such as total response time and data center processing time.

A. Cloud Analyst Simulation Toolkit

The installation phase of the "Cloud Analyst Simulation Toolkit":

The Internet Cloudlets that the users of UserBase create will be distributed globally. Some Internet attributes including Latency and Communication Duration to the Data Centre, are attached to Internet Cloudlets according to the cloud-based Application Service Broker's allocation rules.

Here, the Internet Cloudlet will be routed to or processed by a specific Virtual Machine (VM) according to the load balancing policies of the Data Centre Controller's VM Load Balancer. The CloudSim framework is used for all cloudlet work and results delivery [20].

Once the outcomes from Virtual Machine have been received, Data Center Controller relay them the UserBase through Internet, updating the transmission with the latest information on Service Latency based on Internet Properties.

Once UserBase receives the outcome or result, then it modifies the response time. The process continues till the simulation is complete and outcomes or results are provided.

B. Simulation Setting

There simulated six UserBases, each of which corresponds to a specific time zone, and the average user utilizes the app for approximately two hours. That each internet user makes a fresh request once in every five minutes:

TABLE I: UserBase Configuration Attributes.

User Base	Region	Request Per User Per Hour	Data Size Per Request (Bytes)	Peak Hour	Simultaneous Online Users During Peak Hours	Simultaneous Online Users During Off-Peak Hours
UB1	0	100	100	13:00-15:00	400,000	40000
UB2	1	100	100	15:00-17:00	100,000	10000
UB3	2	100	100	20:00-22:00	300,000	30000
UB4	3	100	100	01:00-03:00	150,000	15000
UB5	4	100	100	21:00-23:00	50,000	5000
UB6	5	100	100	09:00-11:00	80,000	8000

Here parameters have the following meanings:

Peak Hour: The time when users have the most access is referred to as peak hour.

Simultaneous Online Users during Peak Hours: At peak hours, the number of users who are able to access the system.

Simultaneous Online Users during Off-Peak Hours: At off-peak hours, users who utilize the system. The above mentioned values are extracted from the simulation class's and virtual machine's primary configuration table. (Fig. 3).

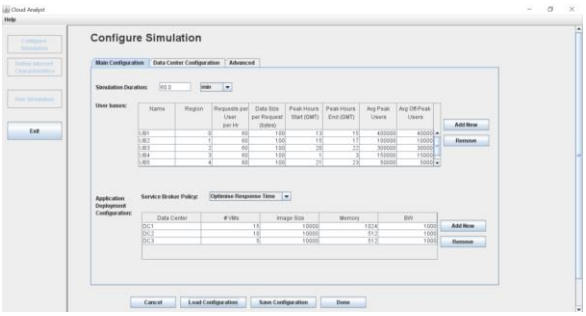

Fig. 3. User and Virtual Machines Parameters

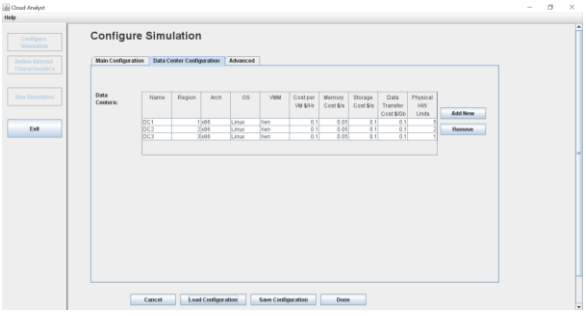

Fig. 4. Parameters for a data center's setup

Here, run the simulation for three times, each time using a different set of policies. Particularly-

1st Time: Comparison simulation result between existing Throttled Algorithm and MTLBA using 3 data center with 15 virtual machines.
2nd Time: Comparison simulation result between existing Throttled Algorithm and MTLBA using 5 data center with 25 virtual machines.

6. OUTCOME OF SIMULATIONS AND ANALYSES

A. Scenario 1: Simulate with 3 data center with 15 virtual machines (VMs).

In Throttled algorithm, there identification of virtual machines in the index table using the perception technique from commencing to conclusion of the table will result in the current status of requests being queued when there are numerous virtual machines (VM) in the system. Utilizing two index tables (Available and Unavailable) and one waiting queue with the MTLBA, the method only require disseminating requests to Virtual Machines in Available index table without searching for them. Beside that if there no virtual machine available than the ID of VMs are transferred to waiting queue. This enhances the overall efficiency of the system.

Simulation Result of Throttled Algorithm (3 Data Center with 15 Virtual Machines)			
	Avg (ms)	**Min (ms)**	**Max (ms)**
Overall response time:	157.75	36.79	529.01
Data Center processing time:	11.63	0.09	42.23
Simulation Result of MTLBA (3 Data Center with 15 Virtual Machines)			
	Avg (ms)	**Min (ms)**	**Max (ms)**
Overall response time:	152.22	36.79	528.98
Data Center processing time:	6.02	0.09	36.76

Fig. 5. Simulation Result of Throttled VS MTLBA (3 Data Center with 15 Virtual Machines)

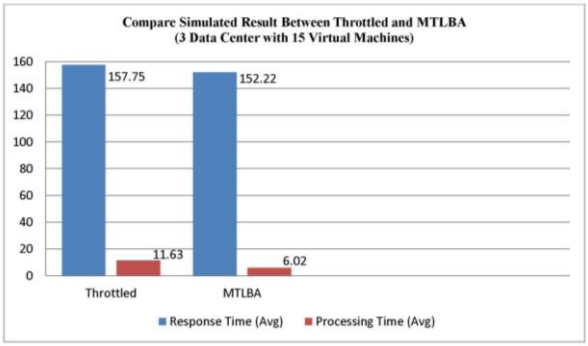

Fig. 6. Compare Between the Simulation Result of Throttled VS MTLBA (3 Data Centers with 15 VMs)

DAFFODIL INTERNATIONAL UNIVERSITY JOURNAL OF SCIENCE AND TECHNOLOGY, VOLUME 18, ISSUE 2, JULY 2023
ISSN 1818-5878 (Print) 2408-8498 (Online)

B. Scenario 2: Simulate with 5 data center with 50 virtual machines (VMs).

Simulation Result of Throttled Algorithm (5 Data Center with 50 Virtual Machines)			
	Avg (ms)	**Min (ms)**	**Max (ms)**
Overall response time:	121.27	36.59	271.23
Data Center processing time:	13.83	0.09	42.42

Simulation Result of MTLBA (5 Data Center with 50 Virtual Machines)			
	Avg (ms)	**Min (ms)**	**Max (ms)**
Overall response time:	114.58	36.71	297.13
Data Center processing time:	7.07	0.09	36.89

Fig. 7. Simulation Result of Throttled VS MTLBA (5 Data Center with 50 Virtual Machines)

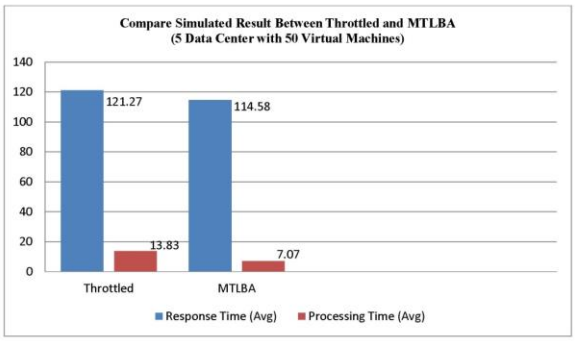

Fig. 8. Compare Between the Simulation result (5 Data Centers with 50 VMs)

In Figure 8, we can see that, our MTLBA algorithm performs much better than the Throttled algorithm. From simulations of experimental results can easily see that, With the MTLBA the response time and data center processing time are faster than Throttled algorithm. This indicates that the MTLBA algorithm is superior to the existing Throttled algorithm in terms of load balancing.

7. CONCLUSION

The following objectives have been attained by this article's analysis and establish a more effective algorithm (MTLBA) based on existing Throttled Algorithm to enhance load balancing over earlier methods. This study emphasis on widely used load balancing techniques in today's cloud environment. The MTLBA algorithm's findings have achieved these objectives including reducing the volume of requests waiting in delivery queues and speeding up hubs cloud processing and response times in comparison to existing Throttled. This also implies that MTLBA algorithm outperforms existing Throttled in terms of cloud computing performance. MTLBA has shown effectiveness as the number of VMs grows, decreasing the cloud data centers' response and processing times. Future research will focus on ways to enhance the algorithm's performance.

References

[1] S. Afzal and G. Kavitha, “Load balancing in cloud computing – A hierarchical taxonomical classification,” in Journal of Cloud Computing, vol. 8, no. 1, Dec. 2019

[2] M. Vanitha and P. Marikkannu, “Effective resource utilization in cloud environment through a dynamic well-organized load balancing algorithm for virtual machines,” in Computers & Electrical Engineering, vol. 57, pp. 199–208, Jan. 2017

[3] P. Kumar and R. Kumar, “Issues and Challenges of Load Balancing Techniques in Cloud Computing,” in ACM Computing Surveys, vol. 51, no. 6, pp. 1–35, Feb. 2019

[4] S. K. Mishra, M. A. Khan, B. Sahoo, D. Puthal, M. S. Obaidat and K. Hsiao, "Time efficient dynamic threshold-based load balancing technique for Cloud Computing," *2017 International Conference on Computer, Information and Telecommunication Systems (CITS)*, Dalian, China, 2017, pp. 161-165,

[5] J. Junaidi, P. Wibowo, D. Yuniasri, P. Damayanti, A. M. Shiddiqi, and B. A. Pratomo, “APPLIED MACHINE LEARNING IN LOAD BALANCING,” in JUTI: Jurnal Ilmiah Teknologi Informasi, vol. 18, no. 2, p. 76, Jul. 2020

[6] F. Tang, L. T. Yang, C. Tang, J. Li, and M. Guo, “A Dynamical and Load-Balanced Flow Scheduling Approach for Big Data Centers in Clouds,” in IEEE Transactions on Cloud Computing, vol. 6, no. 4, pp. 915–928, Oct. 2018

[7] M. Effatparvar and M. S. Garshasbi, “A Genetic Algorithm for Static Load Balancing in Parallel Heterogeneous Systems,” in Procedia - Social and Behavioral Sciences, vol. 129, pp. 358–364, May 2014

[8] S. H. H. Madni, M. S. A. Latiff, Y. Coulibaly, and S. M. Abdulhamid, “Recent advancements in resource allocation techniques for cloud computing environment: a systematic review,” in Cluster Computing, vol. 20, no. 3, pp. 2489–2533, Dec. 2016

[9] K. Dasgupta, B. Mandal, P. Dutta, J. K. Mandal, and S. Dam, “A Genetic Algorithm (GA) based Load Balancing Strategy for Cloud Computing,” in Procedia Technology, vol. 10, pp. 340–347, 2013

[10] D.A. Shafiq, N. Z. Jhanjhi, A. Abdullah, and M. A. Alzain, “A Load Balancing Algorithm for the Data Centres to Optimize Cloud Computing Applications,” in IEEE Access, vol. 9, pp. 41731–41744, 2021

[11] S. Sidana, N. Tiwari, A. Gupta, and I. S. Kushwaha, “NBST algorithm: A load balancing algorithm in cloud computing,” on 2016 International Conference on Computing, Communication and Automation (ICCCA), Apr. 2016

[12] M. van der Boor, S. Borst, and J. van Leeuwaarden, "Load balancing in large-scale systems with multiple dispatchers," in IEEE Xplore, May 01, 2017. https://ieeexplore.ieee.org/document/8057012

[13] K. Dubey, M. Kumar, and S. C. Sharma, "Modified HEFT Algorithm for Task Scheduling in Cloud Environment," in Procedia Computer Science, vol. 125, pp. 725–732, 2018

[14] M. Sookhak, A. Gani, M. K. Khan, and R. Buyya, "WITHDRAWN: Dynamic remote data auditing for securing big data storage in cloud computing," in Information Sciences, vol. 380, pp. 101–116, Feb. 2017

[15] Subhi Abdalkafor, A. Abdalqahar Jihad, and E. Tariq Allawi, "A cloud computing scheduling and its evolutionary approaches," in Indonesian Journal of Electrical Engineering and Computer Science, vol. 21, no. 1, p. 489, Jan. 2021

[16] H. N. Le and H. C. Tran, "ITA: The Improved Throttled Algorithm of Load Balancing on Cloud Computing," in International journal of Computer Networks & Communications, vol. 14, no. 1, pp. 25–39, Jan. 2022

[17] K. Dubey and S. C. Sharma, "An extended intelligent water drop approach for efficient VM allocation in secure cloud computing framework," in Journal of King Saud University - Computer and Information Sciences, Nov. 2020

[18] S. Devi, D. Sumathi, V. Venkataraman, Chunduru Anilkumar, Kirankumar Kataraki, and S. Balakrishnan, "CLOUD load balancing for storing the internet of things using deep load balancer with enhanced security," in Measurement: Sensors, vol. 28, pp. 100818, Jun. 2023

[19] N. Xuan Phi, C. T. Tin, L. N. Ky Thu, and T. C. Hung, "Proposed Load Balancing Algorithm to Reduce Response Time and Processing Time on Cloud Computing," in International journal of Computer Networks & Communications, vol. 10, no. 3, pp. 87–98, May 2018

[20] Wickremasinghe, R. N. Calheiros, and R. Buyya, "CloudAnalyst: A CloudSim-Based Visual Modeller for Analysing Cloud Computing Environments and Applications," 2010 24th IEEE International Conference on Advanced Information Networking and Applications, 2010

AUTHORS PROFILE

Md. Toufecul Islam is pursuing his M.Sc degree in Computer Science and Engineering (CSE) at Dhaka University of Engineering and Technology (DUET).

Prof. Dr. Mohammod Abul Kashem received his B.Sc. and M.Sc.Engg. degrees from State University "Lvivska Polytechnica," Ukraine, in 1996 and 1997, respectively. In 2001 he earned Ph.D. in Control Systems and Processes from National University "Lviv Polytechnic" Ukraine. Subsequently, Dr. Kashem completed his Post Doctorate fellowship at University Lumiera Lyon2, France. (Erasmus Mundas Scholarship, European Commission) in 2016 and he was appointed as a professor at the CSE Department of Dhaka University of Engineering and Technology (DUET) in the year of 2013.

Tanjina Jahan is pursuing her M.Sc degree in Computer Science and Engineering (CSE) at Dhaka University of Engineering and Technology (DUET).