ICRIE 2023 4th INTERNATIONAL CONFERENCE ON RECENT INNOVATIONS IN ENGINEERING

*Proceedings of the 4th International Conference on Recent Innovation in Engineering*
*ICRIE 2023, University of Duhok, College of Engineering, 13th – 14th September 2023*
*(Special issue for Passer journal of basic and applied sciences)*
*Paper No. 17*

# Optimizing Time Consumption for Smartphone-based Distributed Parallel Processing System

**Nashma Taha Muhammed[1], Zryan Najat Rashid[2], Subhi R. M. Zeebaree[3 *], 0000-0002-3895-2619 , JIHAN Abdulazeez Ahmed Rasool[4], Rizgar R. Zebari[5] and Mohammed A. M.Sadeeq[6]**

[1]IT Dept., Sulaimani Polytechnic UniversitySulaimani, Iraq
[2]Computer Network Dept., Sulaimani Polytechnic University, Sulaimani, Iraq
[3]Energy Eng. Dept., Technical College of Engineering, Duhok Polytechnic University, Iraq
[4]Computer Science Dept., College of Science, University of Duhok, Duhok, Iraq
[5]Computer Science Dept., College of Science, Nawroz University, Duhok, Iraq
[6]ITM Dept., Technical College of Administration, Duhok Polytechnic University, Duhok, Iraq
* Energy Eng. Dept., Technical College of Engineering, Duhok Polytechnic University, Iraq. Tel: +9647504332160
nashma.taha.m@spu.edu.iq; zryan.rashid@spu.edu.iq; subhi.rafeeq@dpu.edu.krd; drjihanrasool@uod.ac; rizgar.khuder@nawroz.edu.krd; mohammed.abdulrazaq@dpu.edu.krd

## ABSTRACT

This paper is focused on designing an efficient approach to retrieve information that is processed by mobile devices. So, this system is designed to combine various techniques that can be beneficial for using distributed parallel processing techniques with distributed cloud computing that is used for processing a heavy load as an investigation for the proposed task that came from the cloud node. The proposed system allows users to perform and generate heavy loads that can be distributed among multiple devices using parallel processing in which smartphones are used instead of computer devices as servers that give important enhancement concerning other works. A significant improvement will be provided to the proposed system's performance. Hence, it will influence a significant reduction of the (Waiting, CPU, User, and Maximum Execution) timings and minimize memory usage via using distributed parallel processing techniques for heavy loads. As a sample of important outcomes, comparing matrix-order=1024x1024 shows a decrease from (622,333 ms) using one server to (48,631 ms) using 16 servers, when a decreased ratio is equal to (92.18%).

**KEYWORDS:** Parallel Processing, Cloud Computing, Distributed Parallel Processing, Web server.

## 1 INTRODUCTION

The processing of large amounts of data is made possible by a rapidly developing technology called cloud computing, which includes scheduling mechanisms as a crucial component [1]. Cloud computing is a new IT technology that combines numerous ideas and sub-technologies like virtualization, processing power, storing, being able to share, distribution networks, and connection to create modern technology with significant growth in the IT sector [2]. With cloud computing, several tasks may run concurrently in the background [3].

Clouds are a sizable group of easily accessible and usable virtualized technologies [4]. A cloud offers a user-friendly environment and a variety of services, such as platform as a service (PaaS), software as a service (SaaS), and infrastructure as a service (IaaS), which are all examples of cloud computing [5]. Users must meet the requirements for accessing the required data from the cloud environment when they wish to access any data or files from the cloud server. Four primary categories of cloud deployment models exist

(private cloud, public cloud, community cloud, and hybrid cloud) [6]. Cloud computing is currently widely regarded as one of the most advanced computing concepts in the information technology environment. This was due to advancements in current computing paradigms such as grid computing, parallel computing, and distributed computing, as well as other computing paradigms [7]. A distributed system is a computer system in which several computers collaborate through a network to deliver services for a shared aim [8]. The purpose of a distributed system is to make distributed tasks and resources appear to consumers as if they were part of a single system [9]. Load balancing is being used to manage several services in cloud computing and is accomplished in a server cluster. One of the major challenges in cloud computing is load balancing. An effective load-balancing method should decrease response times, boost throughput, and maximize resource use [10]. The technique of equally distributing the workload between processors and devices is known as load balancing. Various load-balancing strategies are being used to transfer or shift workload across nodes to enhance system performance [11].

Shukur H et al. 2020 [12], several clustering parallel computing for distributed system tasks. The research focuses on previous distributed system studies to increase performance. Offering many alternatives with pros and cons helped me concentrate. These solutions face challenges from enhancing system performance to reacting rapidly to prevent system overhead. This paper used deep feature research and comparison of SYNC and ASYNC Modes for distributed system concurrent computation classifications. Gopala M, et al. 2022 [13], presented the enormous difficulties that cloud computing service providers face, especially in load balancing and other security-related issues. Because cloud computing is so difficult, academics study virtual machine security, fail tolerance, virtual machine migration, QoS satisfaction, and other related topics. For novice researchers in cloud computing load balancing, this research may be hampered. Chandrakala, B. [4] offered an overview of cloud computing technologies, including current and future trends, services, security concerns and obstacles, and cloud computing attacks. This paper discusses cloud computing applications. After overcoming data security risks, this technology may alter a workplace. They explored cloud computing's biggest data security issues, how to solve them, and their pros and cons.

## 2 BACKGROUND THEORY

The uniform appearance will assist the reader in reading the paper of the proceedings. It is, therefore, suggested to authors to use the example of this file to construct their papers. This particular example uses an American letter format with 25 mm margins left, right, top and bottom.

All text paragraphs should be single-spaced, with the first line intended by 10 mm. Double spacing should only be used before and after headings and subheadings, as shown in this example. The position and style of headings and subheadings should follow this example. No spaces should be placed between paragraphs.

### 2.1 Smartphone Computing

Smartphone usage has changed through the years as computational and storage capacities have improved. Smartphones can now analyze huge amounts of data, allowing them to be used for personal or business purposes [14]. Mobile computing has quickly established itself as a widespread technology that is replacing traditional computer paradigms by enabling users to make use of portable, context-aware, and interactive computing [15]. Mobile Cloud Computing (MCC) is the combination of cloud computing with mobile computing [16].

### 2.2 Mobile Cloud Computing

MCC (Mobile Cloud Computing) is a new development in computing that allows cloud users to access a variety of services remotely [17]. Mobile Cloud Computing (MCC) is a distributed computing technology that improves on the resource limitations posed by the Mobile Device (MD). This technique

takes advantage of the strong computing capabilities available through Cloud Computing Infrastructure (CCI). Mobile device capabilities have been enhanced because of the advent of MCC [18], [19]. Data security concerns extend across the MCC architecture's several models, such as the User Layer (UL), Communication Layer (CL), and Cloud Service Provider Layer (CSPL) [20]. Figure 1 depicts the MCC architecture.

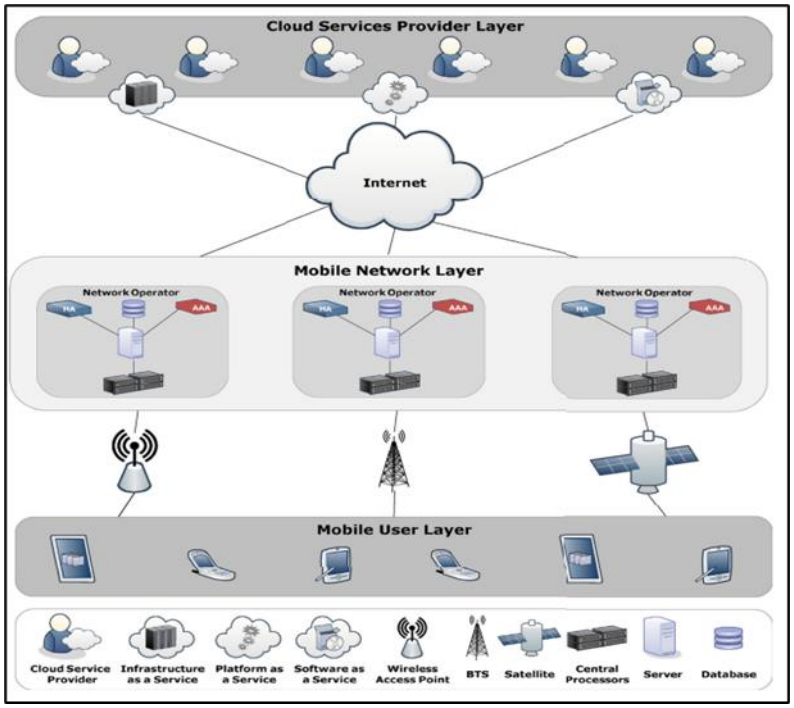


**Figure 1:** Architecture of mobile cloud computing [21]

- Mobile User Layer: This layer is made up of various mobile cloud service consumers who utilize their mobile devices to access cloud services (e.g., smartphones and tablets) [21].
- Mobile network: Mobile devices and network providers make up a mobile network. They construct and manage the communication between the mobile device's functional interface and the network operator [22].
- Internet service: The internet service acts as a link between the mobile network as well as the cloud [22].
- Cloud Services Provider Layer: This layer is made up of several cloud computing service providers who offer a variety of cloud computing services, such as IaaS, PaaS, and SaaS [21].

### 2.3 Load Balancing

There is no doubt that load balancing is one of the most important issues in cloud computing (CC) [23]. The load balancing idea allows data centres to avoid over or underloading in a virtual machine, which is a problem in the cloud computing area [24]. Load balancing optimizes virtual machines in Cloud Computing. Workload balancing improves resource allocation and user satisfaction. Cloud load balancing reduces data transmission and reception delays and prevents overloaded nodes from affecting cloud data centre QoS [25]. There are two types of load-balancing methods which are (static cloud and dynamic) cloud load-balancing:

- Static-Load Balancing: Node state is ignored. All nodes and properties are known beforehand. Past data powers the algorithm. It's easy to construct since it doesn't need system status information. [26].

- Dynamic-Load Balancing Techniques: This method uses a system state. Dynamic node states drive the algorithm. Status Table stores cloud node states. Dynamic algorithms balance load but are hard to build. [26].

## 3 THE PROPOSED SYSTEM

In the early days, there were many discussions about the design and implementation of this system for processing a heavy load that clients determine. Parallel processing is one of the techniques that are being used to process heavy loads; it speeds up computations by dividing them into smaller jobs that multiple processors can handle. One of the crucial reasons for constructing or developing the proposed system is because, naturally, most mobile phones are not used continuously by their owners and go to waiting situations; the proposed system takes this advantage to be used as a server. Another objective of the proposed system is to implement load balancing among servers that participate in the system for processing heavy loads. Load balancing is a technique for spreading traffic among different servers. The above goals contribute to the system being one of the important system designs that were not available before.

### 3.1 Requirements of OTCSPDPPS

T Some requirements are needed to develop and implement the proposed system, including software and hardware requirements.

- **Software Requirements:** Some software tools are required for preparing the proposed system to prepare the proposed system, such as the client side uses ASP.net for programming, which Visual Studio has used. Server-side combing Java and Dart programming together to create mobile applications using Android Studio. Webserver (Cloud-side) Software requirement: Created APIs to communicate with clients and servers. Using SQL Server to create a database for storing data.
- **Hardware Requirement**: The implementation of hardware requirements is divided into three parts: server-side, client-side, and cloud-side.

### 3.2 The Architecture of the Proposed System

This section is about showing the design stages of developing the system. Each main part is divided into further subparts, each of which will be discussed in its section within this chapter to declare its purpose in the proposed system. The IP of the system used for the proposed system is (94.176.232.18:83). There are three main parts of the general structure for the proposed system. Client-Side: After opening the main page, the client must decide the dimension of the matrices and then enter several dimensions to create two equal matrices. After that, send a notification to servers to resolve the task. The system can accept multiple servers. Cloud-Side: All data generated by the proposed system are contained in a database of the management system. Figure 2 shows the architecture of the proposed system. The client enters data to be solved, which is then stored in the cloud's database. After that, the cloud side distributes the jobs among the registered servers. Server-Side: Servers must register themselves automatically when they are connected for the first time to a system without the system performing any steps. In the next step, the server processes the amount of data it receives from the cloud. After each server has processed its specified amount of data, the results will be sent back to the cloud side. So, the cloud side returns the result for the client side, which includes time information and matrices results. Also, sending some critical information such as CPU information, and device information.

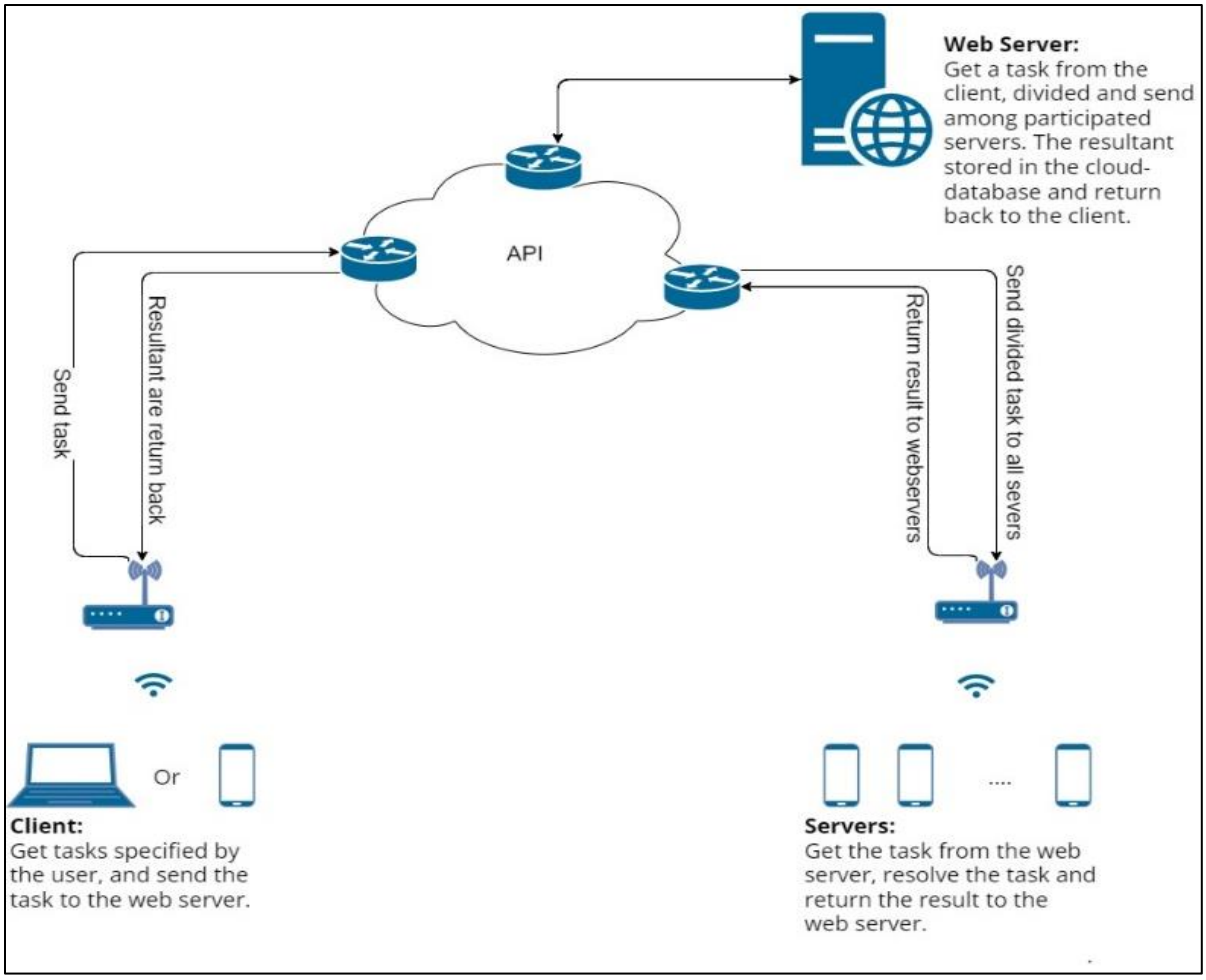


**Figure 2:** Architecture of the proposed system.

### 3.3 System Scenarios

The proposed system OTCSPDPPS has five main scenarios depending on the number of servers that registered themselves in the system to process matrix algebra with different dimensions decided by the client. For all scenarios, two equal dimensions of matrices will be sent to the web server (cloud-side) and, in turn, distributed among the registered servers.

- **Scenario-1: One-Server Multi Dimension Matrix (OSMDM):** In this scenario, the proposed system used just one server to process the task that the client sent it. The client sends tasks with different dimensions of the matrix, ranging from 2x2 to 4096x4096. Depending on this scenario, the effect of parallel processing may not appear because all data will be processed by just one server. Figure 3 shows its mechanism. Firstly, the client will be initialized by the client-user; after that, one server will be selected from the ready servers on the server side. The client will select all the specified tasks (Number of Tasks=12=T) for this scenario with Matrix order of (2x2, 4x4, 8x8, 16x16, 32x32, 64x64, 128x128, 256x256, 512x512, 1024x1024, 2048x2048, and 4096x4096). The client will select the first task and call task number 1, represented in the flowchart by the variable (i). The web server receives the selected task from the client, stores it in its database, and sends it to the selected server. If the server completes the task processing, all the results will be returned and stored in the cloud-side database. Following that, all data will be sent and displayed on the client side. Then, a new task will be selected by the client and sent to the cloud side to be processed

in the same way as Task-1, and so on. Anytime the client-user wants to terminate the task processing, he can do it manually.

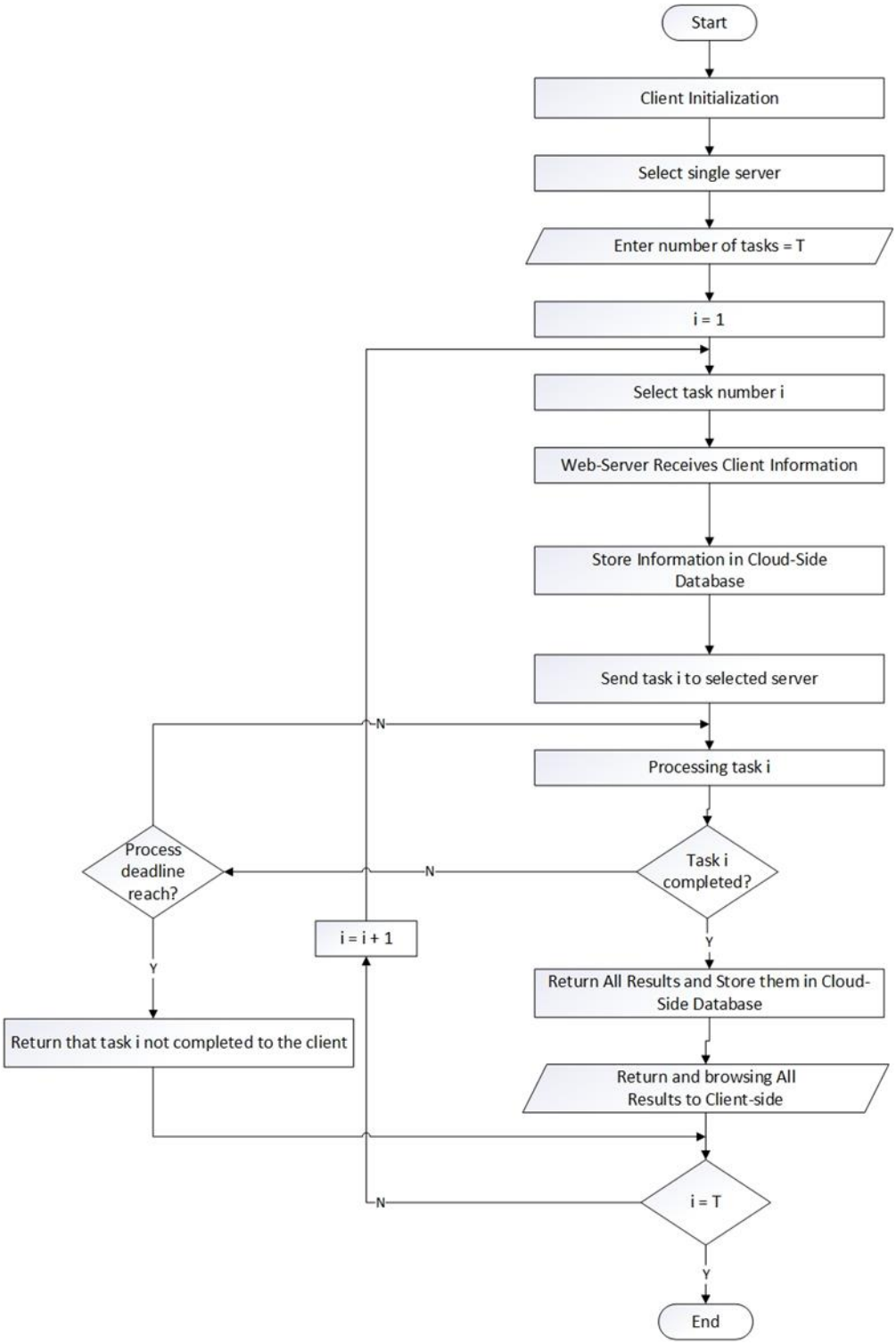


**Figure 3:** Flowchart of the mechanism of scenario-1 (OSMDM).

- **Scenario-2: Two-Servers Multi Dimension Matrix (TSMDM):** In this scenario, the proposed system is organized to implement parallel processing as well as load balancing by using two servers (S1 and S2) to process the specified tasks (Number of Tasks=12=T). Similar to scenario 1, the system can solve the tasks when matrix dimensions vary from 2x2 to 4096x4096. The previous steps are just like scenario 1. But, when the web server receives the selected task from the client and stores it in its database, it will divide the task between the selected servers and then send the task to them.
- **Scenario-3: Four-Servers Multi Dimension Matrix (FSMDM):** In this scenario, four servers (S1, to S4) are used to process the task. The system can solve the tasks when matrices' dimensions vary from 4x4 to 8192x8192. Note that the matrix order 2x2 cannot be divided among 4 servers, so the tasks for this scenario will start from 4x4 matrix order, the specified tasks (Number of Tasks=12=T).
- **Scenario-4: Eight-Server Multi Dimension Matrix (ESMDM):** In this scenario, these tasks are processed by eight servers (S1, to S8). The system can solve the tasks when matrices' dimensions vary from 8x8 to 8192x8192. Note that the matrix orders (2x2 and 4x4) cannot be divided among 8 servers, so the tasks for this scenario will start from 8x8 matrix order, the specified tasks (Number of Tasks=11=T).
- **Scenario-5: Sixteen-Servers Multi Dimension Matrix (SSMDM):** In this scenario, these tasks are processed by sixteen servers (S1, to S16).

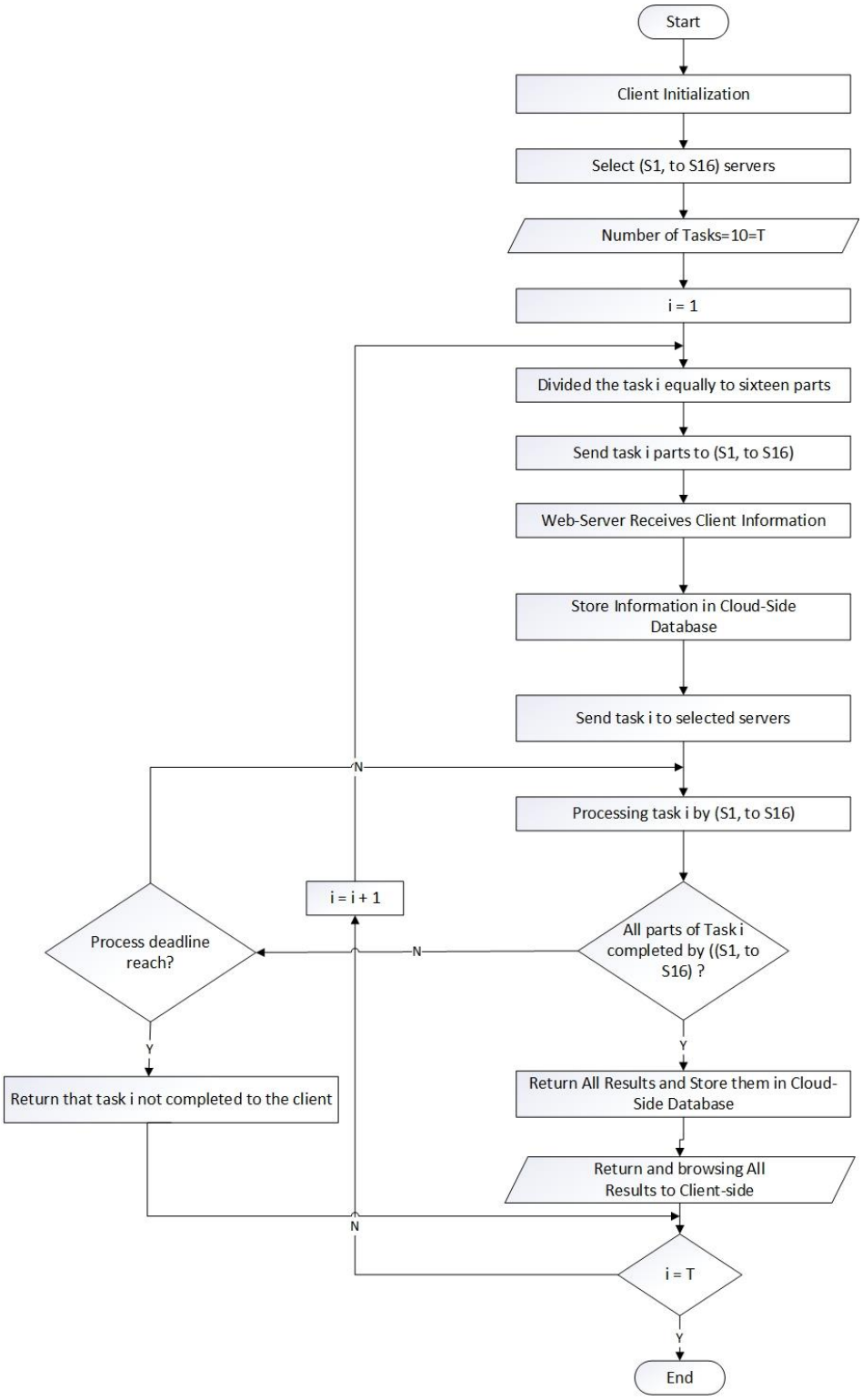


**Figure 4:** Mechanism of scenario-5 (SSMDM).

In this scenario, there is a big decrease in time. All timing results are decreased because of using sixteen servers. So, parallel processing has appeared how affect the processing task. The system can solve

the tasks when matrix dimensions vary from 16x16 to 8192x8192. Note that the matrix orders (2x2, 4x4, and 8x8) cannot be divided among 16 servers, so the tasks for this scenario will start from the 16x16 matrix order, the specified tasks (Number of Tasks=10=T). At the same time, Figure (3.17) shows its mechanism. The same rest steps of scenario 2 will be repeated but for scenario 5 structure.

As part of this study, we compare the first and last scenarios. Figures 3 and 4 describe each scenario. The plots in two related charts are shown in Figures (5, 6 and 7, 8).

## 4 IMPLEMENTATION RESULTS AND DISCUSSION

**Implementation Results of Scenario-1: One-Server Multi Dimension Matrix (OSMDM):** This scenario is implemented depending on one client and one server to process the task. After the implementation of this scenario, the results of all the matrix orders were obtained by the server without any problems. However, the system faced a problem with returning the results to the cloud side, which did not exceed the matrix order (1024x1024). This limitation occurred due to the limitation of sending the resultant matrix as one part. Hence, as a solution to this problem, the resultant matrix is divided into sub-matrices and returned to the cloud side, and in turn, returned to the cline side. The resultant matrix of order (2048x2048) is divided into 4 submatrices and of order (4096x4096) into 8 sub-matrices. The reconstruction of these sub-matrices has been done on the client side according to the sequence order of them sent by the Web server.

**Implementation Results of Scenario-5: Sixteen-Server Multi Dimension Matrix (SSMDM):** This scenario is implemented depending on one client and sixteen servers to process the task. In this scenario, the first three matrix orders (2x2, 4x4, and 8x8) cannot be implemented because the matrix orders are less than the number of participated servers. The problem of incapability of returning the results of the matrix order when it is greater than (4096x4096), to the Cloud-side is appeared and solved as illustrated in scenario 1. There is a big decrease in the obtained timing results, which are decreased because of using sixteen servers. Hence, the effect of parallel processing via sixteen servers is clearer than that of previous scenarios.

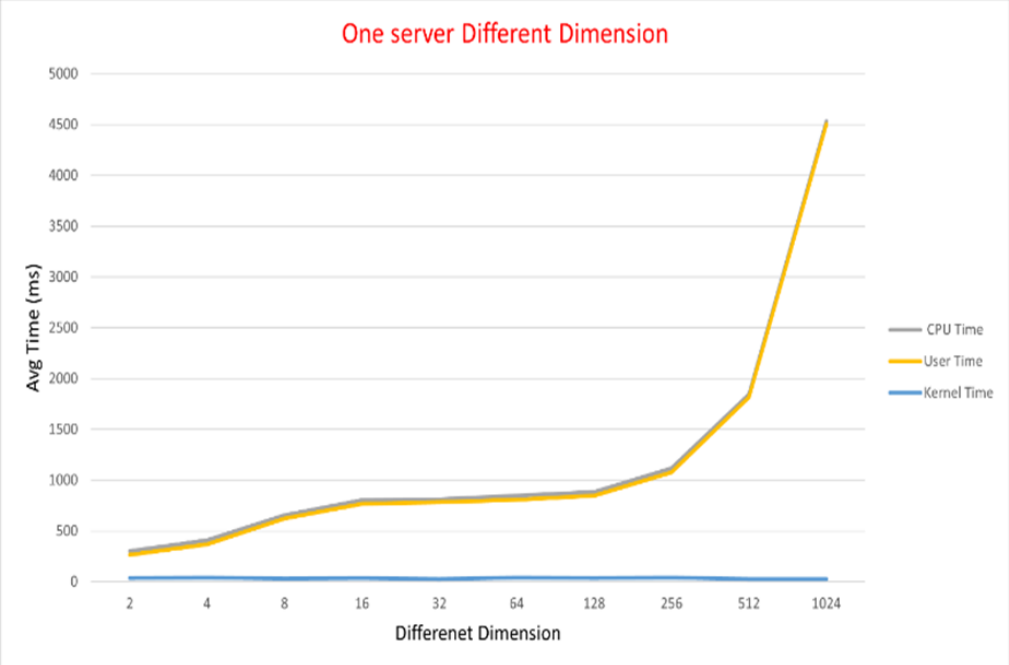


**Figure 5:** Average (CPU, User, and Kernel) timing values with different matrix orders for Scenario-1 (OSMDM).

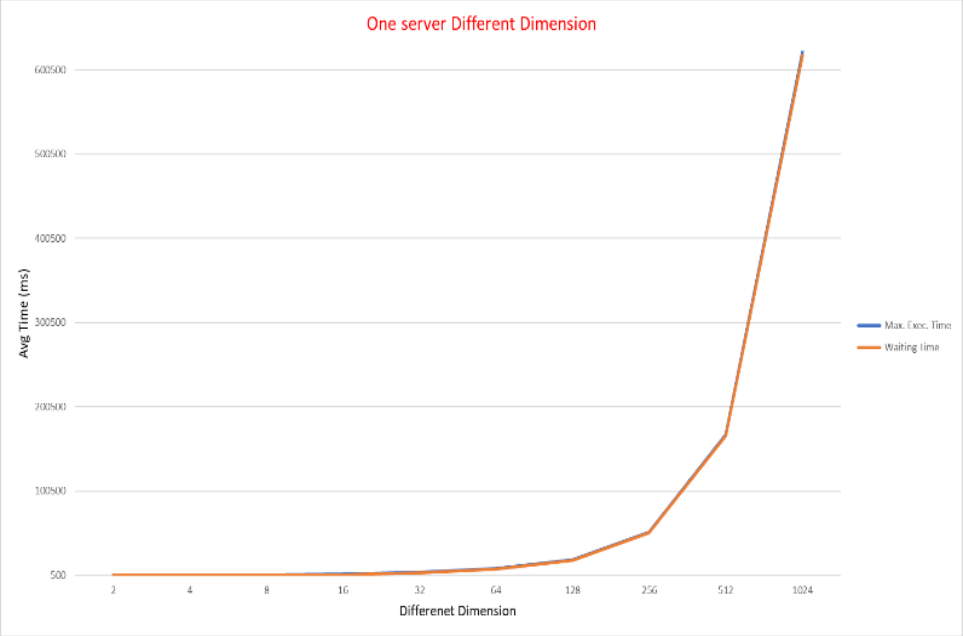


**Figure 6:** Average (Max. Exec. and Waiting) timing values with different matrix orders for Scenario-1 (OSMDM).

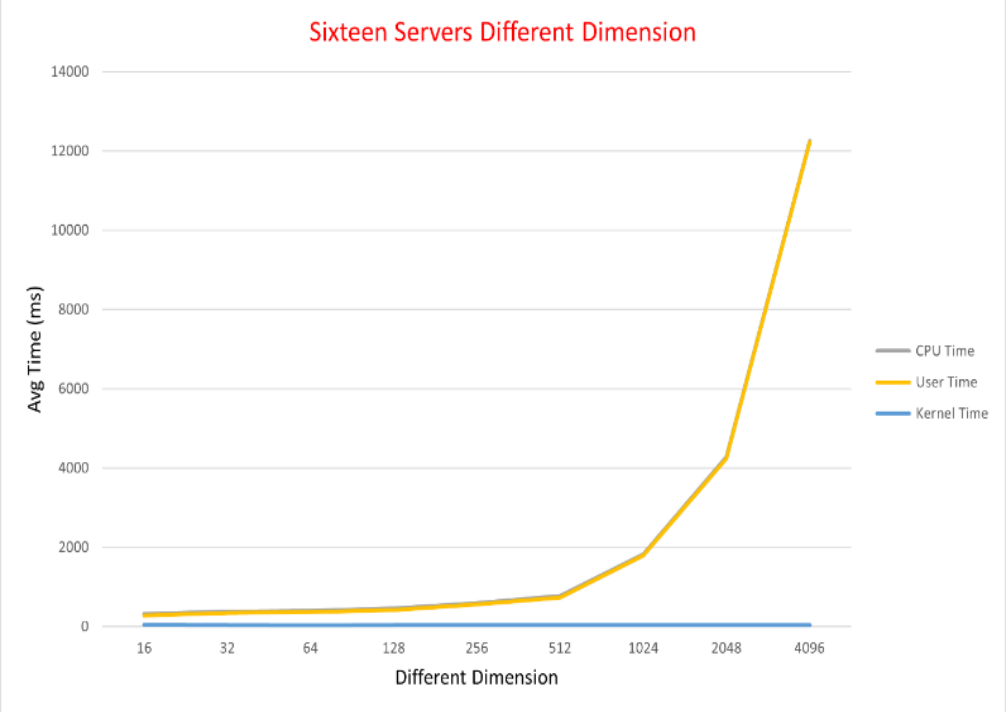


**Figure 7:** Average (CPU, User, and Kernel) timing values with different matrix orders for Scenario-5 (SSMDM).

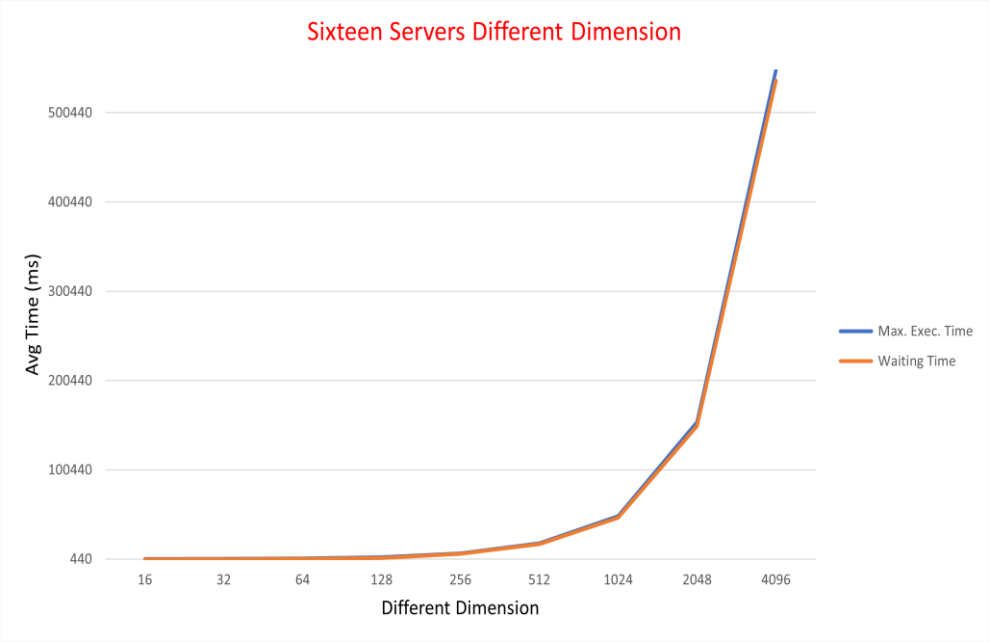


**Figure 8:** Average (Max. Exec. and Waiting) timing values with different matrix orders for Scenario-5 (SSMDM).

There has been an increase in the amount of data and the time required to process it. While these data are widely available, only a few users and systems can utilize them efficiently. Therefore, developing strategies to improve parallel processing methods using cloud computing is crucial. The requirement for efficient processing with heavy loads while handling difficult tasks weighing the advantages of using distributed parallel processing systems versus cloud computing. The suggested system's key features include employing cloud computing and parallel processing to develop a system with single clients and several servers (mobile devices). So, this paper compares the first and the last scenarios. The first scenario consumes more time because it uses a single server for processing all the tasks. In this scenario, the effectiveness of parallel processing does not appear in high demand while it needs more time. Looking at the results of this scenario, when the matrix-order=2x2, the Max. Exec. The time needed for task processing is (590 ms). When the matrix-order=1024x1024, it needs more execution time, which is (622,333 ms). These results show that the effect of parallel processing did not appear in this scenario while it needs more time for task processing. Also, the second scenario is the best one that depends on the proposed system, which obtains the least time-consuming processing task. Looking at the results of this scenario, when the matrix-order=16x16, the Max. Exec. Time is (**446** ms), while it was (671 ms) for scenario 4. But, when the matrix-order=1024x1024, it needs more execution time which is (**48,631** ms), while it was (103,286 ms) for scenario 4. Also, when the matrix-order=2048x2048, it needs more execution time, which is (**154,114** ms), while it was (316,083 ms) for scenario 4.

There is a big decrease in the time for matrix orders (1024x1024 and 2048x2048). Hence, it seems that the processing speed increased more than 2 times, which is an excellent indicator of good results and an efficient parallel processing system. All the time results are decreased because of the use of 16 servers. So, parallel processing has appeared how affect the processing task. In the end, it can be concluded that all the technology around the world is focused on reducing processing time and getting results as fast as possible.

## 5 CONCLUSION

The principal contribution of this paper is the full design of the proposed system, which integrates numerous technologies, including distributed parallel processing and cloud computing, into a single design. Adding to obtaining real results, using a single client of any desired type (computer, mobile phone, or tablet). An unrestricted number of servers (smartphones) will take part in processing massive amounts of data via distributed parallel processing. The main aim of the paper is to design and implement a proposed Optimizing Time consumption for Smartphone-based Distributed Parallel Processing Systems. Overall, this paper highlights the fact that users can solve complex tasks interactively with minimal processing time by using the proposed system. From the obtained results, it can be concluded that the processing time decreases with the increasing number of servers when treated with a heavy load. Hence, the effects of parallel processing are noticed well. As a sample of comparison, the processing time for matrix-order=1024x1024 is decreased from (622,333 ms) using one server to (48,631 ms) using 16 servers, with a decreased ratio of (92.18%). As well as the processing time is increasing with the increasing number of servers when treated with a light load. As a sample of comparison, the processing time for matrix-order=2x2 is increased from (590 ms) using one server to (738 ms) using 2 servers. The increasing ratio of the processing time is (25.08%). Hence, it is not recommended to use parallel processing with a light load.

## 6 ACKNOWLEDGEMENTS

The authors would like to express full thanks to the Computer Network Department at Sulaimani Polytechnic University, for preparing the specific laboratories to perform the real implementation of this research.

## REFERENCES

1. Z. Rashid, S. Zebari, K. Sharif, and K. Jacksi, “Distributed Cloud Computing and Distributed Parallel Computing: A Review,” International Conference on Advanced Science and Engineering (ICOASE), Kurdistan Region, Iraq, 2018.
2. A. M. Fatemi Faraz, “Cloud Computing Challenges and Opportunities: A Survey,” 1st International Conference on Telematics and Future Generation Networks (TAFGEN), 2015.
3. G. Saravanan, S. Neelakandan, P. Ezhumalai, and S. Maurya, “Improved wild horse optimization with levy flight algorithm for effective task scheduling in cloud computing,” Journal of Cloud Computing, vol. 12, no. 1, Dec. 2023, doi: 10.1186/s13677-023-00401-1.
4. B. Chandrakala, “A Review on Chronicle of Cloud Computing Security and Storage Environment Models,” 2023.
5. D. Rani and R. K. Ranjan, “A Comparative Study of SaaS, PaaS and IaaS in Cloud Computing,” International Journal of Advanced Research in Computer Science and Software Engineering, vol. 4, no. 6, pp. 458–461, 2014.
6. S. Namasudra, “Data Access Control in the Cloud Computing Environment for Bioinformatics,” International Journal of Applied Research in Bioinformatics, vol. 11, no. 1, pp. 40–50, Jan. 2021, doi: 10.4018/ijarb.2021010105.
7. T. Alam, “IAIC Transactions on Sustainable Digital Innovation (ITSDI) Cloud Computing and its role in the Information Technology,” Transactions on Sustainable Digital Innovation (ITSDI), vol. 1, no. 2, pp. 108–115, 2020, [Online]. Available: https://pandawan.aptisi.or.id/index.php/att/article/view/59%0Ahttps://doi.org/10.34306/itsdi.v1i2.103
8. Z. A. S. A. et al Najat Z, “Design and Analysis of Proposed Remote Controlling Distributed Parallel Computing System Over the Cloud,” International Conference on Advanced Science and Engineering (ICOASE), 2019.
9. S. Akter, Md. A. Hossain, and Md. M. Rahman Redoy Akanda, “A Noble Security Analysis of Various Distributed Systems,” International Journal of Engineering, Science and Information Technology, vol. 1, no. 2, pp. 62–71, 2021, doi: 10.52088/ijesty.v1i2.101.
10. H. Alazzam, W. Mardini, A. Alsmady, and A. Enizat, “Load balancing in cloud computing using water flow-like algorithm,” in PervasiveHealth: Pervasive Computing Technologies for Healthcare, ICST, Dec. 2019. doi: 10.1145/3368691.3368720.
11. V. Mathur, “A Comparative Study of Load Balancing Techniques in Distributed Systems,” 2017. [Online]. Available: www.ijiset.com