Designing an Efficient Fog Computing System with Heterogeneous Nodes

Abhay Manavadariya
DA-IICT
Gandhinagar, Gujarat
202311032@daiict.ac.in

Prof. Tapaskumar Maiti
DA-IICT
Gandhinagar, Gujarat
tapas_kumar@daiict.ac.in

Prof. Rahul Mishra
DA-IICT
Gandhinagar, Gujarat
rahul_mishra@iitp.ac.in

***Abstract*—We know the increasing usage of cloud computing, but there are still issues that are not solved because of the base problems with cloud computing, which include irregular latency, poor mobility support, and a lack of location awareness. Fog computing can address those problems by providing flexible resources and services to end users at the edge of the network. Fog Computing is a geographically distributed computing architecture that decentralizes the storage, processing, and analysis of data away from centralized cloud data centers (Servers), distributing these functions closer to the edge devices. This can significantly reduce latency, enhance data management efficiency, and improve response times in network applications. In this, we are trying to reduce the time delay and improve the efficiency of the user's request and response. We create 4 architectures to check time delay and efficiency. The architectures are: 1. single client-server architectures; 2. single client-fog-server architectures; 3. single client-multiple fog-server architectures; and 4. multiple client-multiple fog-server architectures. And compare the time delay and efficiency among them.**

I. INTRODUCTION

Today, the Internet of Things (IoT) and other smart devices, like wearable smart devices, smart cities, and vehicles, are widely available and the future of the internet. According to estimates made by IDC (International Data Corporation), "The IoT will continue to rapidly expand the traditional IT industry" in 2015, a 14% increase over 2014 [3]. But there are limited computations and storage on smart devices. To overcome this limitation,Cloud computing was introduced, which can provide flexible resources to applications on those devices. To increase the application of the Internet of Things (IoT), we use the power of cloud computing technology, but there are still unsolved problems like low latency, geo-distribution, location awareness, and time efficiency. To solve this problem, fog computing technology was introduced.

Fog computing is a distributed computing technology that relocates data storage and processing closer to the network edge. It's like a substation over the network, which is the middle layer between the cloud and edge devices. Fog can solve the problems of low latency, geo-distribution, location awareness, and time efficiency by being closer to the network edge devices. The architecture of fog computing is shown in Fig. 1.

In the client-fog architecture, fog computing can enable computing directly at the edge of the network, which is the future of the Internet. One example is a commercial edge router that improves processor speed, the number of cores, and built-in network storage. In the Fog computing structure, the facilitator that provides resources to the edge devices is called the fog node. Fog nodes are intermediary layers between the local processing unit and the cloud.

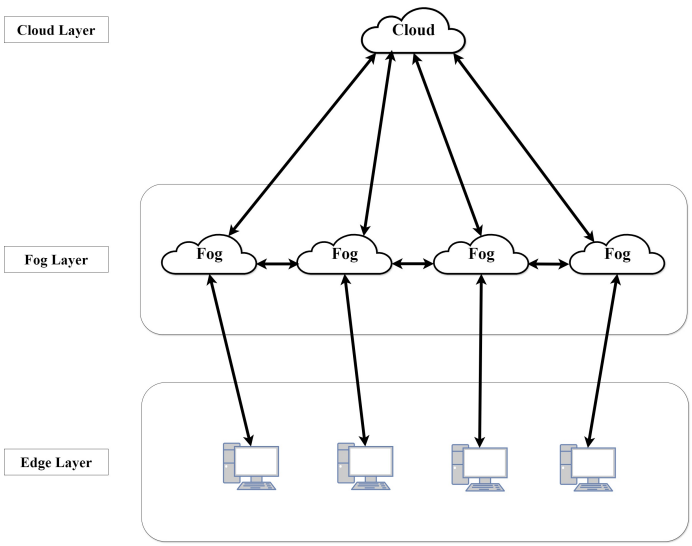

Fig. 1. Client-Fog Architecture

Fog computing is an extension of cloud computing. It directly computes the request and response nearest to the edge device and reduces the latency. As fog works near edge devices, it improves security and scalability.

II. RELATED WORK

This section summarizes the existing work on task offloading in Fog computing, which considers heterogeneous resources.

Most of the research papers work on homogeneous resources [1] [2]. Clients can communicate with fog nodes, but they cannot communicate with one another. We worked on heterogeneous resources, in which the fog nodes can also

communicate with each other.

In heterogeneous architecture, we work on two types of approaches: single devices and multiple devices. In a single heterogeneous device architecture, let us consider a computer device. We divided the CPU into a number of cores. Suppose the device has a 12-core CPU, so divide the CPU into 12 parts, and those parts work as fog nodes. Out of the multiple cores, one core acts as a server, and others act as fog nodes. When the client sends multiple tasks to the fog node, all the fog nodes distribute the tasks among themselves, which will reduce the time consumption compared to when the client sends the task to a single fog or server. So, as we increase the number of cores, it will reduce the time.

In multiple heterogeneous device architectures, one device acts as the server, some act as a fog node, and some act as a client. Clients are communicated to the fog node, and fog nodes are communicated to each other and the server. Multiple clients will send tasks concurrently, and clients will be distributed among fog nodes, which have enough processing power. The task distributions are decided based on the specification of the fog node (the device's configuration).

Finally, client will assign a fog node based on parameters such as distance between client and the fog system, configuration of the fog system that will be enough to process the task of particular client, etc.

III. ARCHITECTURE

A. Client-Server Architecture

Fig. 2. client-Server Architecture

In a client-server architecture, requests are sent to servers by clients. These requests contain information about the desired action or resource. Upon receiving a request, the server processes it and sends back a response to the client, confirming the action or providing the requested information. An essential aspect of managing a client-server architecture is performance monitoring and optimization. Measuring the execution time of requests is crucial for evaluating the efficiency and responsiveness of the system.

B. Single client-Single Fog Architecture

As you see in Fig. 3, Single client-Single fog architecture, a fog node serves as an intermediary between a single client and a server. By adding a layer between the client and server, this architecture increases the flexibility and capabilities of the system. The parts and their functions are broken down as follows:

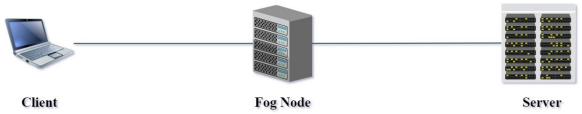

Fig. 3. Single client-Single fog Architecture

Client: The client, which can be a computer, smartphone, or Internet of Things device, initiates communication by sending requests to the fog node. This could include requests for real-time data analysis, resource allocation, or service provisioning.

Fog Node:- The fog node acts as an intermediate computing entity positioned between the client and the server. It serves several purposes, including data preprocessing, caching, and offloading computational tasks from the server. Fog nodes are typically located closer to the client compared to centralized cloud servers, enabling faster response times and reducing network latency. They are particularly useful in edge computing scenarios where real-time processing and low-latency communication are critical.

Server: The server remains the central hub responsible for fulfilling client requests and providing core services or resources. However, with the introduction of fog computing, the server's workload may be alleviated as certain tasks are offloaded to the fog node. This redistribution of tasks ensures that the server can focus on high-level computations, database management, or complex processing tasks, while routine or latency-sensitive operations are handled closer to the client.

C. Single client-Multiple Fog Architecture

In the single client-multiple fog architecture, several fog nodes arranged in a deliberate manner throughout the network facilitate communication between a single client and a server. These fog nodes collaborate to establish a network for distributed computation and data processing, not only by facilitating communication between the client and server but also by interacting with one another. Each fog node works with other fog nodes to maximise resource utilisation and improve system efficiency. These duties may include content delivery, analytics, or data preprocessing. Through the use of cooperation and inter-node communication, the fog nodes as a group lower latency, ease network congestion, and enhance overall system responsiveness.

D. Multiple client-Multiple Fog Architecture

In the multiple client-multiple fog architecture, multiple clients communicate with a server through a network of fog

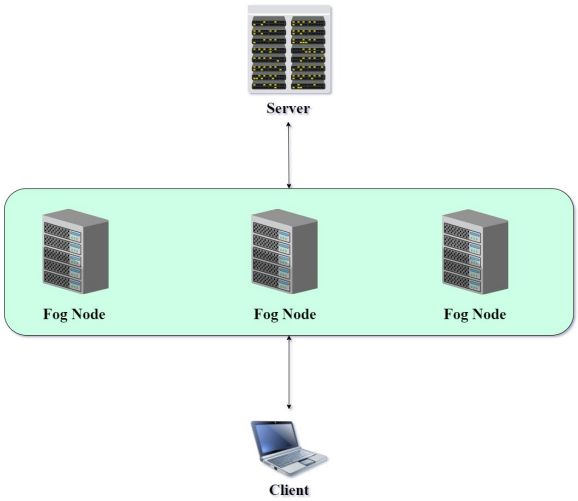

Fig. 4. Single client-Multiple Fog Architecture

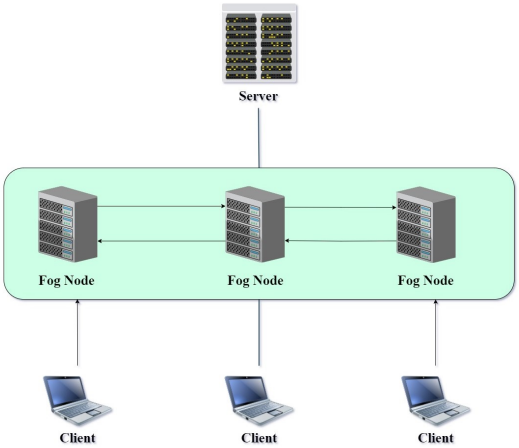

Fig. 5. Multiple client-Multiple Fog Architecture

nodes strategically distributed across the network.These fog nodes help process information closer to where the devices are. By collaborating with each other, fog nodes optimize resource utilization and enhance system performance, reducing latency and improving responsiveness for multiple clients simultaneously. This way of doing things is flexible, strong, and efficient, especially when many devices need to talk to the main server at the same time.

IV. RESULTS

As shown in Fig. 6, as a result of our implementation, we chose three algorithms: first come first served (FCFS), least connection load balancing, and round robin. Out of these three algorithms, round-robin gives a more efficient result. So we chose the round-robin algorithm.

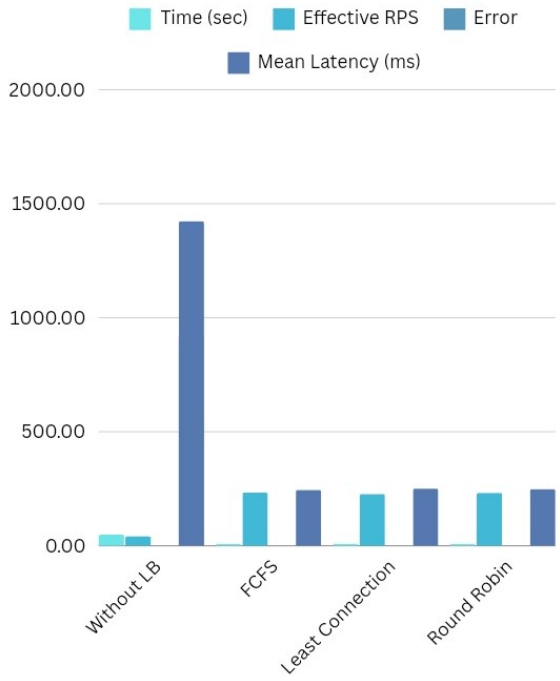

Fig. 6. Result of Algorithm

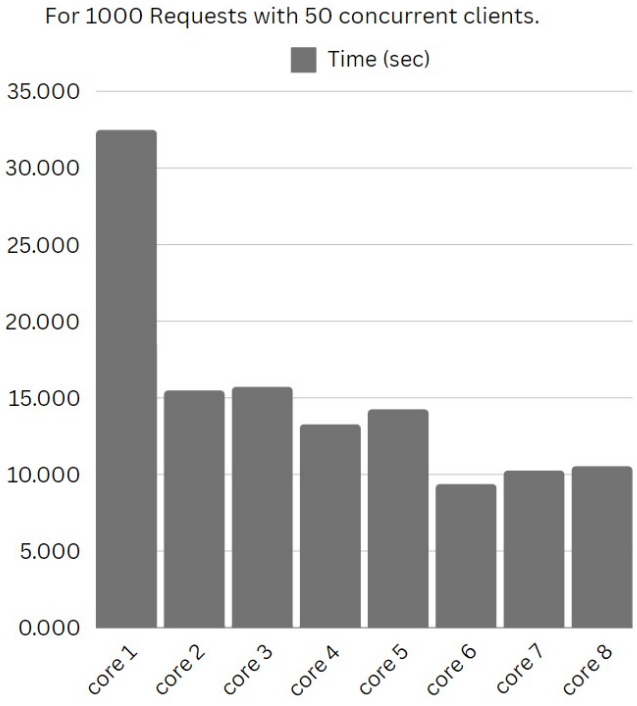

Fig. 7. Result according to the number of Fog node

In Fig. 7, results according to the number of fog nodes, we fix the number of request 1000 and concurrency 5 and plot the relation of the number core (fog node) and time in seconds. The process takes 32 seconds for 1000 requests on a single core. We can compare this with other processes. As we increase the number of cores, the time will decrease, but the time also depends on the number of concurrent clients. As we use 6 cores, it takes less than 10 seconds, while as we increase the number of cores to seven, the time will increase.

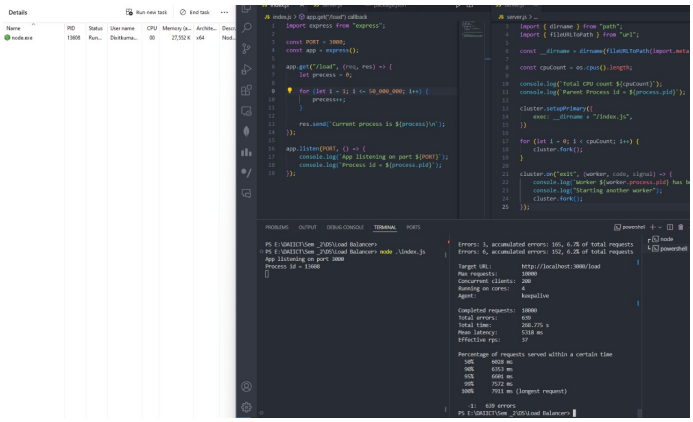

Fig. 8. Output 1

As shown in Fig. 8, Output 1, the result of the simple single-client-server architecture is that the client sends 10,000 requests with 200 concurrent requests on four cores. From that, 339 errors were found and total time to complete all 10,000 request was 268.77 seconds. 50

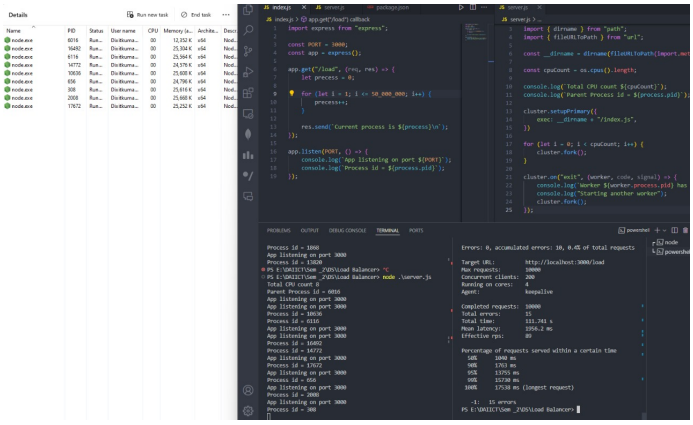

Fig. 9. Output 2

As shown in Fig. 9, Output 2, this figure shows the result of client-fog architecture with 8. As we can see,there are 8 child processes created with 8 different process IDs. The client sends 10,000 requests with 200 consents to the fog node, that is evenly distributed and computes the request. As a result, the time taken to complete 50% of the task in 1040 ms and 100% of the of the task in 111.741 seconds is comparatively shorter for client-server, less time consumed, and more effective as the total error is 15.

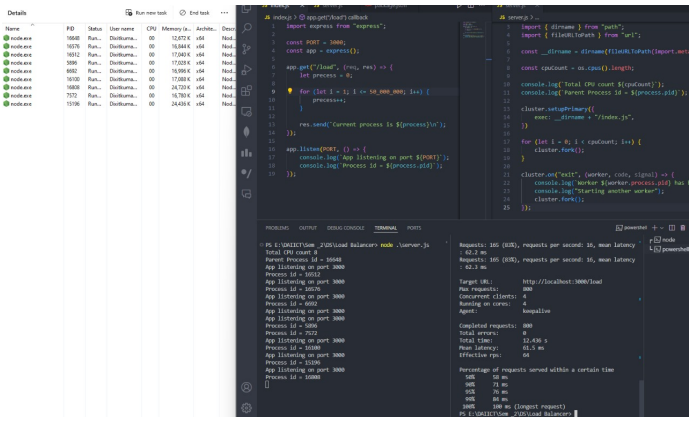

Fig. 10. Output 3

As shown in Fig. 10, Output 3, the result of the client-fog architecture with 8 cores. As we can see,there are 8 child processes created with 8 different process IDs. The client sends an 800 request with 4 concurrences to the fog node, which is evenly distributed and computes the request As a result, the time it takes to complete 50% of tasks is 58 ms, and 100% of tasks are completed in 12.436 seconds. As shown in Fig. 11, Output 3, the result of the simple

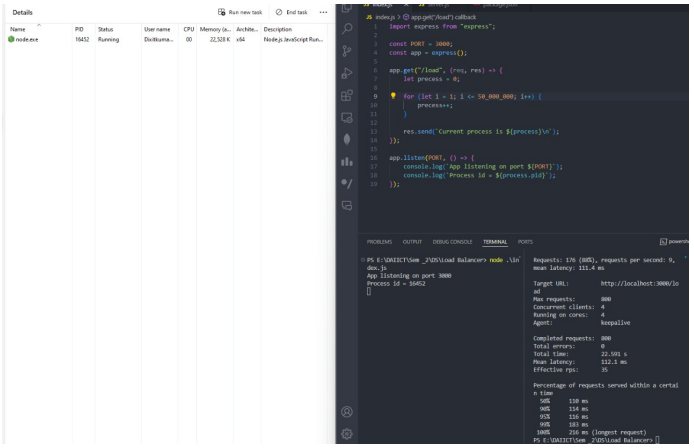

Fig. 11. Output 4

single-client-server architecture with 800 requests and 4 concurrences. The time taken to complete 50% task in 110ms and 100% task completed in 22.591 s. As we can compare the same requests and concurrences on single client-server and client-fog architectures, the client-server took 22.591 seconds while the client-fog took 12.436 seconds.

V. FUTURE WORK

In our future work, we will focus on expanding the system by increasing the number of fog nodes. This expansion aims

to enhance system scalability and performance to meet the growing demands of edge computing applications.

One key aspect of our future work involves the development of dynamic task offloading algorithms that take into account the heterogeneous nature of fog nodes. Specifically, we will design algorithms that optimize task allocation based on system configurations, such as variations in core count and processing power among fog nodes.

Furthermore, we plan to implement system configuration-aware task allocation strategies to ensure efficient resource utilization and workload distribution across fog nodes. These strategies will be essential for maximizing system performance and minimizing latency in real-time edge computing environments.

Additionally, we will explore various optimization techniques, including load balancing algorithms, to further improve system efficiency and ensure equitable resource utilization across fog nodes. Scalability and reliability will remain key considerations as we address challenges associated with system growth and strive to maintain high availability and fault tolerance.

Overall, our future work aims to advance the capabilities of fog computing systems, enabling them to efficiently handle diverse workloads in dynamic edge computing environments.

VI. Conclusion

In conclusion, our fog computing project has made significant strides in leveraging edge resources to enhance the performance and scalability of distributed computing systems. Through the deployment of heterogeneous fog nodes and the implementation of dynamic task offloading mechanisms, we have demonstrated the potential of fog computing to effectively handle diverse workloads in edge environments.

By increasing the number of fog nodes and optimizing task allocation based on system configurations, such as core count and processing power, we have laid the foundation for a more efficient and responsive fog computing system. Our performance evaluations have provided valuable insights into the benefits of our enhancements, showcasing improved system throughput and reduced latency.

References

[1] R. Mishra, H. P. Gupta, P. Kumari, D. Y. Suh and M. J. Piran, "A Task Offloading and Reallocation Scheme for Passenger Assistance Using Fog Computing," in IEEE Transactions on Network and Service Management, vol. 19, no. 3, pp. 3032-3047, Sept. 2022, doi: 10.1109/TNSM.2022.3172602. keywords: Task analysis;Edge computing;Delays;Cloud computing;Buildings;Transport ation;Network topology;Fog computing;passenger mobility;task reallocation;transportation,

[2] R. Mishra et al., "A Game Theory-based Transportation System using Fog Computing for Passenger Assistance," 2021 IEEE 22nd International Symposium on a World of Wireless, Mobile and Multimedia Networks (WoWMoM), Pisa, Italy, 2021, pp. 1-10, doi: 10.1109/WoWMoM51794.2021.00013. keywords: Performance evaluation;Wireless communication;Prototypes;Games;Quality of service;Nash equilibrium;Time factors;Fog computing;offloading;passenger assistance;transportation system,

[3] Gil Press. IDC: Top 10 technology predictions for 2015. http://goo.gl/zFujnE, 2014.