mathematics

MDPI

Article

PQ-Mist: Priority Queueing-Assisted Mist–Cloud–Fog System for Geospatial Web Services

Sunil K. Panigrahi [1], Veena Goswami [2], Hemant K. Apat [2], Ganga B. Mund [1], Himansu Das [1,*] and Rabindra K. Barik [2,*]

[1] School of Computer Engineering, Kalinga Institute of Industrial Technology, Bhubaneswar 751024, India; ctcsunil@gmail.com (S.K.P.); mund@kiit.ac.in (G.B.M.)
[2] School of Computer Applications, Kalinga Institute of Industrial Technology, Bhubaneswar 751024, India; veena@kiit.ac.in (V.G.); hemant.fca@kiit.ac.in (H.K.A.)
* Correspondence: das.himansu2007@gmail.com or himanshufcs@kiit.ac.in (H.D.); rabindra.mnnit@gmail.com or rabindrafca@kiit.ac.in (R.K.B.)

Abstract: The IoT and cloud environment renders enormous quantities of geospatial information. Fog and mist computing is the scaling technology that handles geospatial data and sends it to the cloud storage system through fog/mist nodes. Installing a mist–cloud–fog system reduces latency and throughput. This mist–cloud–fog system has processed different types of geospatial web services, i.e., web coverage service (WCS), web processing services (WPS), web feature services (WFS), and web map services (WMS). There is an urgent requirement to increase the number of computer devices tailored to deliver high-priority jobs for processing these geospatial web services. This paper proposes a priority-queueing assisted mist–cloud–fog system for efficient resource allocation for high- and low-priority tasks. In this study, WFS is treated as high-priority service, whereas WMS is treated as low-priority service. This system dynamically allocates mist nodes and is determined by the load on the system. In addition to that, the assignment of tasks is determined by priority. Not only does this classify high-priority tasks and low-priority tasks, which helps reduce the amount of delay experienced by high-priority jobs, but it also dynamically allocates mist devices within the network depending on the computation load, which helps reduce the amount of power that is consumed by the network. The findings indicate that the proposed system can achieve a significantly lower delay for higher-priority jobs for more significant rates of task arrival when compared with other related schemes. In addition to this, it offers a technique that is both mathematical and analytical for investigating and assessing the performance of the proposed system. The QoS requirements for each device demand are factored into calculating the number of mist nodes deployed to satisfy those requirements.

Keywords: edge computing; cloud computing; geospatial data; fog computing; mist computing; priority queue; geospatial web services; WMS; WFS

MSC: 60K25; 60K30; 68M20

check for updates

Citation: Panigrahi, S.K.; Goswami, V.; Apat, H.K.; Mund, G.B.; Das, H.; Barik, R.K. *PQ-Mist*: Priority Queueing-Assisted Mist–Cloud–Fog System for Geospatial Web Services. *Mathematics* **2023**, *11*, 3562. https://doi.org/10.3390/math11163562

Academic Editors: Steve Drekic and Vassilis C. Gerogiannis

Received: 30 June 2023
Revised: 17 July 2023
Accepted: 16 August 2023
Published: 17 August 2023

Copyright: © 2023 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (https://creativecommons.org/licenses/by/4.0/).

1. Introduction

The demand for cloud services has significantly expanded in recent years as more people have access to the technology required to run cloud computing. These days, off-device computation and storage are accomplished through the usage of cloud services [1,2]. The internet of things (IoT) and cloud environments generate an enormous number of geographical data. The harnessing technology that analyzes geospatial data and provides it to the cloud storage system via fog/mist nodes is referred to as fog and mist computing [3–5].

According to the report that was presented by [6,7], the size of the geospatial analytics market is projected to increase from USD 74.78 billion in 2023 to USD 148.91 billion at a

CAGR of 14.77% during the forecast period of 2023–2028. So, these statements motivate us to deliver geospatial services quickly and effectively.

As a result, cloud computing, fog computing, and mist computing are currently the most common forms of computing platforms. These forms of computing make virtualized and scalable resources available through web services. Consequently, both the difficulty of deploying and maintaining web applications and the quality of the environment in which computing takes place are improved [8,9]. Cloud applications and the service providers that support them are gaining popularity as a result of the unique features that they possess. These properties include ease of maintenance, resilience, and sustainability, all of which make it possible to schedule resources and maintain performance control. To improve service delivery for various geospatial data, cloud, fog, mist, and edge computing systems, they have been adopted by many authors [10,11]. However, adding an extra layer to the conventional mist computing system is inefficient for many geospatial IoT devices. We need to evaluate each layer experimentally and analytically to make it cost-efficient and improve system performance. Geospatial IoT devices still communicate directly with cloud services, allowing for more complex computations to be performed. Cloud computing has enabled robust geospatial computing systems to share geospatial data among various parties. This cloud architecture allows many users to access geospatial data through geospatial web services (GWS) [12,13].

In a mist–cloud–fog system, particularly for GWS, the computation jobs need to be correctly divided between the mist and fog nodes. This ensures that the mist, the fog devices, and the cloud can coordinate effectively and adequately. As a result, we use queuing theory for the performance study to examine this kind of resource allocation approach [10,11,14].

To enable online geospatial services, the open geospatial consortium (OGC) has recently published a set of specifications that are either an adaptation or an extension of the usual online service standards. Standardizing service interfaces and data models is possible with several well-known products such as WMS (web map service), WFS (web feature service), WCS (web coverage service), and CSW (catalogue service for the web). In the meantime, a web processing service (WPS) interface can be utilized to gain access to any environmental model or geospatial algorithm that is classified as a geoprocessing service [15–17].

The queuing model has seen extensive use for this kind of study since it may shed light on various QoS parameters, including the response time for the system, CPU utilization, mean throughput, and many more. Mist computing is used to provide improved QoS for high-priority tasks. However, due to the limited computing available on most devices, finishing some allotted work within a delay threshold is possible. This is one of the drawbacks of using mist computing. In [18], the authors suggested that the limited processing capabilities of mist devices place a ceiling on the number of task requests. As a result, we utilize a priority queueing strategy to maximize the high-priority task request rather than consider another low-priority task.

Thus, this study presents a priority-queuing assisted mist–cloud–fog system for geospatial applications to allocate resources to efficiently provide high- and low-priority tasks. WFS is given a high-priority status in this study, whereas WMS is given a low-priority status. This system determines how to dynamically allocate mist nodes in a way that is dependent on the load being placed on the system. The load defines how the system should allocate mist nodes.

1.1. Contributions

The present research paper is structured with the following contributions:

- It presents a description of geospatial computing paradigms, geospatial web services, and different performance evaluations strategies, with varieties of queueing approaches associated with edge, mist, fog, and cloud computing perspectives.

- It introduces the priority queueing-assisted mist–cloud–fog system for geospatal web services.
- It provides the analytical queueing approach along with performance analysis for the proposed system.
- It also carries the performance measurement and experimental results of the proposed system with the variability of arithmetic outcomes in graphs.

1.2. Organizations

The rest of the paper is organized as follows. Section 2 explains the related work. It details the geospatial computing paradigm, the geospatial web services, and the performance evaluations of various models used in different application domains. Section 3 presents a detailed description of the proposed priority queueing analytical approach for the mist–cloud–fog system for geospatial web services. Section 4 presents the experimental results and performance evaluation of the proposed model and the proposed model's performance measurement and experimental results, which are based on the variability of arithmetic outcomes in graphs. Section 5 draws the concluding remarks of the present research paper.

2. Related Work

2.1. Geospatial Computing Paradigms

2.1.1. Geospatial Edge Computing

The term "edge computing" refers to an advanced technology that transfers a module, data, or service from one internet hub to the subsequent hub. A customer who is either physically present or easily duped is close. The perimeter of this computing system is where the data are generated and handled. Through the utilization of this computing technology, edge devices are given the ability to interact with cloud platforms [16,19]. Before cloud platforms can be upgraded to leverage cloud services, the model must alter to sort information by velocity, volume, and variety. Edge computing equalizes data providers and consumers. Near-edge cloud computing jobs are carried out. This computing stores, distributes, caches, processes, and delivers data to clients. Because so many jobs are running on the edge computing network, the edge hubs need to be constructed to meet the requirements for data reliability, privacy, and data security [20]. In this edge computing design, the processing assets should be located close to the information sources. These advantages of cutting-edge standards outweigh those of the cloud framework. For example, mobile phones symbolize the transition between the cloud and the human body, whereas smart houses symbolize the transition between the cloud and the domestic sphere. The cloud-to-mobile edges comprise cloudlets and tiny data centers [1].

2.1.2. Geospatial Mist Computing

Mist computing offloads some computation to the cloud data center's network's edge, actuator devices, and sensors. Mist computing in embedded nodes' microcontrollers computed the network's edge [11,21,22]. Mist computing minimises latency and boosts autonomy. Cloud, fog, and mist computing are complementary because the fog layer's gateway can run computationally complex application tasks, while edge devices can run less intensive ones [23].

The user can access cloud data centre data. Mist computing provides varied services across computing nodes. Cisco invented fog and mist computing, which expands client–server architecture like edge. Geosptial mist computing has four layers: cloud, fog, mist, and edge [10,18].

2.1.3. Geospatial Fog Computing

Cisco invented fog computing in early 2012. This computing paradigm gives untrained users data center resources. It does not use cloud data centers for computing. Cloud servers make computation and data storage for convenient for customers, reducing latencies

relative to transmission overheads. They provide a user interface similar to smart devices. Local processing offers data compression, faster throughput, and decreased latency. Smart cities, residences, and healthcare use fog computing [24–26].

Fog computing uses fog devices. Raspberry Pi and Intel Edison fog devices are cloud-to-user gateways. Geographic big data analysis and distribution require scalable and efficient geospatial fog computing systems. Fog computing minimizes latency and increases throughput for ignorant clients. Fog architecture stores geographical data near local devices instead of a cloud infrastructure data server [2,27,28].

A fog computing system processes customer requests and returns responses. Cloud computing supplies storage and analysis. All resource utilization components, including fog servers, respond to unequal demands. Inefficient resource management reduces QoS and increases energy usage [29–32]. Smart cities use fog computing to manage urban data. Fog computing can promote smart cities, urban business, industry, tourism, and transit management [30,33,34].

2.1.4. Geospatial Cloud Computing

The method of cloud computing deals with an enormous number of large data by dividing up the available computer resources among multiple locations in the cloud. The paradigm of cloud computing allows for the pooling of resources and the provision of services on demand. You are able to do data analysis and visualization with the help of this computing method [17,23,35].

A multi-tenant design is supported by geospatial cloud computing systems, and a single instance can serve several customers for processing, storage, and data transfer. Putting in place enhancements and additional software benefits the user. In cloud GIS architecture, geospatial web services are the essential component of the core functional feature. The discovery of app data and features is performed by a number of geospatial cloud computing solutions using geospatial web services. Because of this, they are utilized in the SOA infrastructure operations of enterprise organizations [35–37].

There are three client tiers available in a geospatial cloud computing system: thin, thick, and mobile. Mobile clients use mobile devices. Thin clients are those that function on web browsers, whereas thick clients are those that function on desktop or standalone systems. In order to connect to cloud servers, thick clients require an additional module or piece of software. On the application layer, servers are responsible for running geospatial web services. This facilitates communication between the many service providers and the end users. Within the application-tier, there is a separate dedicated server for every one of the application services (WPS, WCS, WMS, and WFS, respectively).

The Table 1 highlights the various aspects of geospatial computing paradigms by addressing the cloud, fog, mist, and edge computing paradigms for geospatial applications.

Table 1. Features of geospatial computing paradigms through cloud, fog, mist, and edge computing.

Features	Cloud	Fog	Mist	Edge
Mobility management	No	Yes	Yes	Yes
Computing resources	Yes	Yes	Yes	Yes
Virtualization mechanism	Yes	Yes	Yes	No
Scalability support	Yes	Yes	Yes	Yes
IoT uses	Yes	Yes	Yes	Yes
Large-scale storage	Yes	No	No	No
Real time applications	No	Yes	Yes	Yes
Inter-operability support	No	Yes	Yes	Yes
High energy consumption	Yes	No	No	No
Low latency	No	Yes	Yes	Yes
Location awareness	No	Yes	Yes	Yes
Standardized	Yes	Yes	No	No
Geographically distributed	No	Yes	Yes	Yes
Large-scale processing power	Yes	No	No	No

The geospatial mist–cloud–fog model is presented in Figure 1 with the integration of the geospatial edge, mist, fog, and cloud computing system.

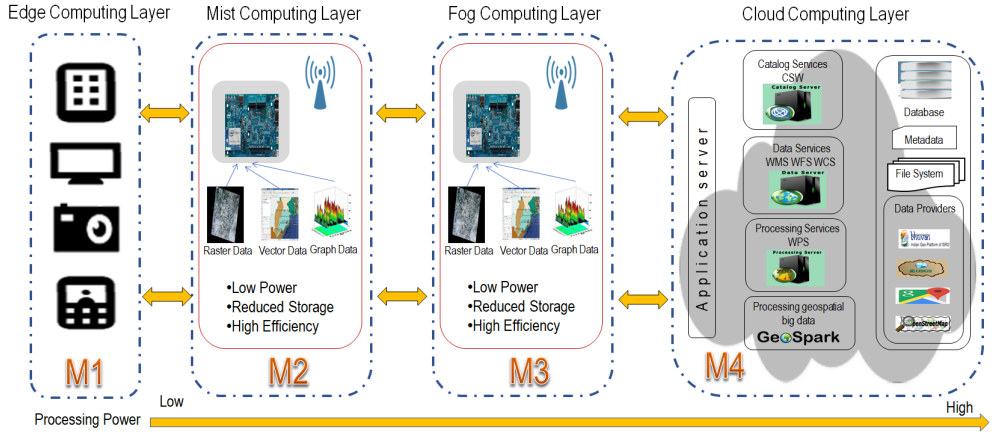


Figure 1. General mist–cloud–fog model for geospatial web services and geospatial data processing.

2.2. Geospatial Web Services

The development of a wide variety of web-based models by scientists is facilitated by geospatial web services (GWS). The development of technology based on cloud computing has opened the door to environmental modeling that is both quicker and more effective. There are public cloud, private cloud, and hybrid cloud products [2,36,38]. GWS caters to the requirements of environmental scientists developing and distributing their models in several ways to meet their needs. The phrase geoprocessing service describes any function or model for processing geospatial and associated data, whereas the term geospatial data service refers to geospatial services for collecting geospatial data. Both geospatial data and geoprocessing services can be derived from GWS in their own right [23,35,39].

Users can access, edit, and utilize hosted geospatial feature datasets through WFS. Distributed tools are used in WMS to produce and host both static and dynamic maps. Access to coverage data in practical formats for client-side rendering, as input into scientific models, and for usage by other clients is made available by a WCS. Users can use web processing services to run GIS calculations on geospatial data. WPS has standardized geospatial statistics methods and standardizes inputs and outputs for geospatial data within the geospatial cloud platform [40,41].

GWS alleviates the burden of tasks by utilizing the combined capacity of distributed services throughout the network. It is accomplished by using massive volumes of geographical data and functions flexibly [24,38].

Compared to the conventional method, in which each activity is carried out on an individual computer, this method facilitates greater remote participation, promotes collaboration, and enhances the repeatability of research.

Many web-based geospatial applications, also known as spatial data infrastructures (SDIs), have been built to utilize geospatial data, geoprocessing services, or both. When many GWSs are available online, researchers integrate various services to fulfill the requirements of more complicated applications. As a result, geospatial analysis and the deployment of GWS are both commonly carried out on the cloud (for example, on Amazon Web Services (AWS), Microsoft Azure, and Google Cloud) [2,41,42].

Specific cloud infrastructures and web services standards are typically created by integrating cloud, fog, and mist computing with diverse geographic applications.

2.3. Performance Evaluations Strategies

Many research works have been carried out where priority and non-priority queueing analytical methodologies have been employed to conduct performance assessments on computing-based systems based on edge, cloud, fog, and mist systems. This is to achieve

the objective of conducting performance assessments. Table 2 compares different queuing mathematical and analytical approaches used by other researchers in the context of the various application services. It can be observed that most of the research works preferred different strategies to the norm.

Table 2. Review of various queuing approaches used in edge, cloud, fog, and mist systems.

Various Queuing Approach							
Year	**Author**	**Reference**	**Edge**	**Mist**	**Fog**	**Cloud**	**Approach**
2011	Khazaei et al.	[43]	✗	✗	✗	✓	M/M/1
2011	Khazaei et al.	[43]	✗	✗	✗	✓	M/G/s
2012	Ellens et al.	[44]	✗	✗	✗	✓	M/M/c/N
2012	Do et al.	[45]	✗	✗	✗	✓	M/M/m/m
2013	Salah	[46]	✗	✗	✗	✓	M/M/1
2013	Pal and Hui	[47]	✗	✗	✗	✓	M/M/1
2014	Mohanty et al.	[48]	✓	✗	✗	✓	M/M/1
2014	Chiang et al.	[49]	✗	✗	✗	✓	M/M/c/N
2015	Evangelin and Vidhya	[50]	✗	✗	✗	✓	M/M/1
2015	Cheng et al.	[51]	✗	✗	✗	✓	M/M/1
2015	Bai et al.	[52]	✗	✗	✗	✓	M/M/c
2015	Kirsal et al.	[53]	✗	✗	✗	✓	M/M/c
2015	Guo et al.	[54]	✗	✗	✗	✓	M/M/1
2016	Akbari et al.	[55]	✗	✗	✗	✓	M/M/1
2017	Chang et al.	[56]	✗	✗	✗	✓	M/M/1
2017	Liu et al.	[57]	✗	✗	✓	✓	M/M/1
2017	El Kafhali and Salah	[30]	✗	✗	✓	✓	M/M/c
2017	Safvati and Sharzehei	[58]	✗	✗	✓	✓	M/M/1
2018	Tadakamalla et al.	[59]	✗	✗	✓	✗	M/M/1
2018	Sthapit et al.	[60]	✗	✗	✓	✗	M/M/c
2018	Chunxia and Shunfu	[61]	✗	✗	✓	✗	M/M/1
2018	Sophin et al.	[62]	✗	✗	✓	✗	M/M/c
2018	Vasconcelos	[63]	✗	✗	✓	✓	M/M/1
2019	Barik et al.	[2]	✗	✓	✗	✗	M/M/c
2019	Jafarnejad et al.	[64]	✓	✗	✓	✓	M/M/1
2019	Barik et al.	[23]	✗	✓	✓	✓	M/M/c
2019	Li et al.	[65]	✗	✓	✓	✗	M/M/1
2020	Kumar and Raja	[66]	✗	✗	✓	✓	M/M/1
2020	Xu et al.	[67]	✓	✗	✓	✓	M/M/1
2020	Patra et al.	[68]	✗	✓	✗	✗	M/M/1
2020	Bouanaka et al.	[21]	✗	✓	✓	✗	M/M/1
2021	Sedaghat et al.	[69]	✗	✗	✓	✓	M/M/1
2021	Sufyan and Banerjee	[70]	✗	✗	✓	✗	M/M/1
2021	Tadakamalla and Menasce	[71]	✗	✗	✓	✗	M/M/1
2021	Feitosa et al.	[72]	✗	✗	✓	✗	M/M/1
2021	Panigrahi et al.	[73]	✗	✓	✓	✗	M/M/1
2021	Behera et al.	[74]	✗	✗	✓	✗	M/M/c/N
2021	Hmissi and Ouni	[11]	✗	✓	✓	✗	M/M/1
2021	Dutta et al.	[22]	✗	✗	✓	✓	M/M/1
2021	Shahid et al.	[18]	✗	✓	✓	✗	M/M/1
2022	Mas et al.	[75]	✗	✗	✓	✓	M/M/1
2022	Rodrigues et al.	[76]	✗	✗	✓	✓	M/M/c/K
2022	Hamdi et al.	[77]	✗	✗	✓	✓	M/M/1
2022	Nikoui et al.	[5]	✗	✗	✓	✓	G/G/1
2022	Golkaret al.	[9]	✗	✗	✓	✓	Multi Queue Priority
2022	Maiti et al.	[76]	✗	✗	✓	✓	M/M/c
2023	Arefian et al.	[14]	✗	✓	✗	✓	M/M/1
2023	Hazra et al.	[78]	✗	✓	✗	✓	M/M/k
2023	Goswami et al.	[3]	✗	✓	✗	✓	M/M/c
2023	Yazdani et al.	[79]	✗	✓	✗	✓	M/M/1
2023	Saif et al.	[80]	✗	✓	✗	✓	M/M/1 and M/M/c
2023	Saif et al.	[81]	✗	✓	✗	✓	M/M/1 and M/M/c
2023	Mallick et al.	[13]	✗	✓	✗	✓	M/M/c

Many different networking fragments are available in an edge, mist, fog, and cloud computing platform, and each networking device operates according to the concept of "first come, first serve". The queueing model has become an important consideration in the system model to evaluate the effectiveness of cloud computing.

Muniir et al. [82] also demonstrated and detailed an integrated fog-assisted cloud architecture for IoT applications that improve performance, latency, scalability, and localized accuracy. To explore and analyze the performance of geospatial fog computing systems within the healthcare business, Barik et al. [2] developed a queuing mathematical and analytical technique. Barik et al. [23] developed a mathematical and analytical approach to queuing in order to explore and examine the performance of geospatial mist computing systems in the education and tourism industries.

Mukherjee et al. [4] took into account a high-priority queue and a low-priority queue in each fog node. These queues are filled with tasks that have directly arrived from the end-users and have been offloaded from the fog nodes. Each task's delay deadline determines which queue it is placed in. In addition, the Lyapunov drift algorithm was utilized for queue scheduling when the tasks in these two queues had stringent latency requirements.

Adhikari et al. [83] devised a plan for prioritizing the assignment of work by dividing it into three distinct groups according to the lengths of their respective due dates. In addition, they also established a rule-based task scheduling technique to discover an ideal sequence for the tasks and reduce the time spent waiting in the queue.

Bhushan and Ma [8] presented an analytical queuing model to implement priority-based job scheduling within a fog-cloud architecture. The model categorized the jobs into two groups to facilitate the implementation of the priority-based service offering. Class 1 refers to computing jobs with a higher priority and more sensitivity to delays. In contrast, Class 2 relates to computing tasks with lower priority and less sensitivity to delays.

He et al. [19] considered a scenario of cloud-assisted multi-access edge computing involving multiple mobile devices. It considered the mobile devices in question to be operating under an $M/G/1$ non-preemptive priority queueing model, with each edge server operating under an $M/G/m$ non-preemptive priority queueing model, and the cloud data center operating under an $M/G/\infty$ queueing model.

3. Proposed Model

This section presents the analytical queuing model for the mist–cloud–fog system by describing its four-tier network topology, as represented in Figure 2. The first tier is the bottom layer, the edge layer. It includes all IoT devices responsible for sensing a wide range of events and relaying the raw detected data to the upper layer immediately above them. It is predicated on the assumption that the total number of IoT devices remains constant and equal to X end customers. The access point connects IoT devices to the mist nodes, connecting them in wireless or cable connections. The access points receive that inbound traffic from end clients. These IoT device messages are gathered at the access points (positioned near the IoT devices) and then forwarded to the mist nodes for further processing. The mist computing layer is the next layer near the client layer. The mist computing layer comprises mist nodes that are clever enough to process, compute, and temporarily store the information that has been received and transmit any leftover requests or burdens to the fog tier for additional processing or storage. Each of these mist nodes connects to the fog gateway. They are in charge of transmitting data to and from the cloud via the fog gateway. The cloud layer is the uppermost tier. This layer comprises a large data center where virtual machines can process and store massive data.

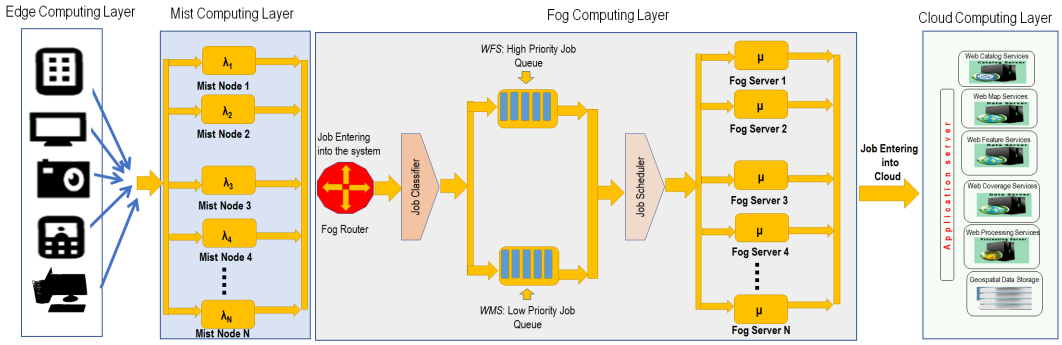


Figure 2. Proposed preemptive-resume priority queueing analytical approach.

Figure 3 describes the overall sequence diagram of the proposed architecture.

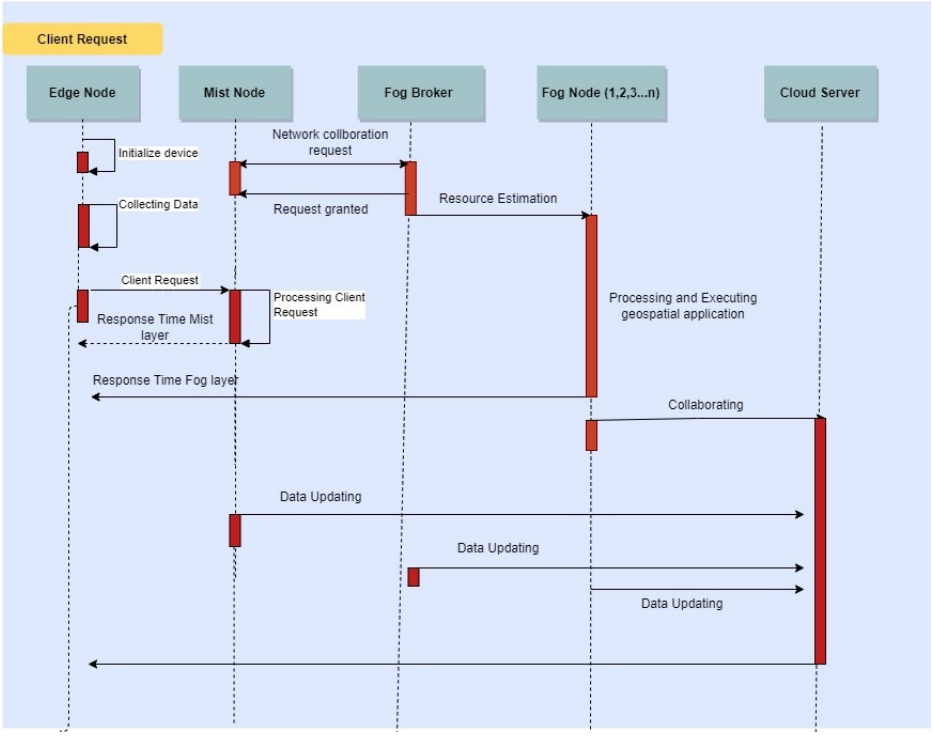


Figure 3. Sequence diagram of the proposed mist–cloud–fog system with preemptive-resume priority queueing approach.

Let us assume that tasks arrive as a Poisson process at a single exponential processor and that each job is assigned to one of the two priority classes upon arrival in the system. It is customary to number priority tasks so that small numbers correspond to higher priorities. Assume that the (Poisson) arrivals of the first or higher priority task have a mean arrival rate λ_h and those of the second or lower priority task have a mean arrival rate λ_ℓ. The total arrival rate is $\lambda = \lambda_h + \lambda_\ell$.

Queue disciplines that prioritize specific tasks are frequent in service systems. Priority can be based on elements such as the classification of tasks and the type of service. With the advent of cloud computing, a broad range of priority tasks were put in to improve system measures. Analyzing more variants entails much more complex underlying processes.

Here, we discuss the priority model of two-type as part of the $M/M//1$ set-up. To start, when considering the priority queues in the fog system, the following components need special attention:

- There is more than one class of tasks on the basis of their demands or significance to the system.

- The tasks of one class are more important than the other. When there are more than two classes, it is possible to organize them into a hierarchy of service priorities.
- The priority that agrees with a class of tasks may or may not be preemptive. If one task is prioritized in relation to another, the priority task will prevent the non-priority task from obtaining service.
- When service preemption is permitted, it can resume the service to the preempted task after the priority tasks are processed, from when the service was preempted or initiated from the start. They are disciplines of preventive recovery and preventive repetition, respectively.

Consider the preemptive-resume priority class for the $M/M/1$ queue. Tasks of type 1 are a higher priority for the service than tasks of type 2. By preemptive resume, we mean that a Class 1 task will be served immediately upon arrival if there are not already Class 1 tasks in the system. As a result, a Class 1 task may preempt a Class 2 task already on the service system. If a class 2 task is preemptive, it goes to the "top of the line" for Class 2 jobs, and when processed, the service is restarted, not repeated. Let the arrival and departure of the task take place according to Poisson and exponential distribution, respectively. The arrival and processing rates of the tasks of the two types are as follows: Type 1—arrival rate λ_h, processing rate μ_h; Type 2—arrival rate λ_ℓ, service rate μ_ℓ. Since the processing time of the task is exponentially distributed, the memory-less property of the processing-time distribution makes it easier to simplify the preemptive-resume analysis. Figure 4 depicts the flow chart of the proposed preemptive-resume priority queueing model for WFS and WMS.

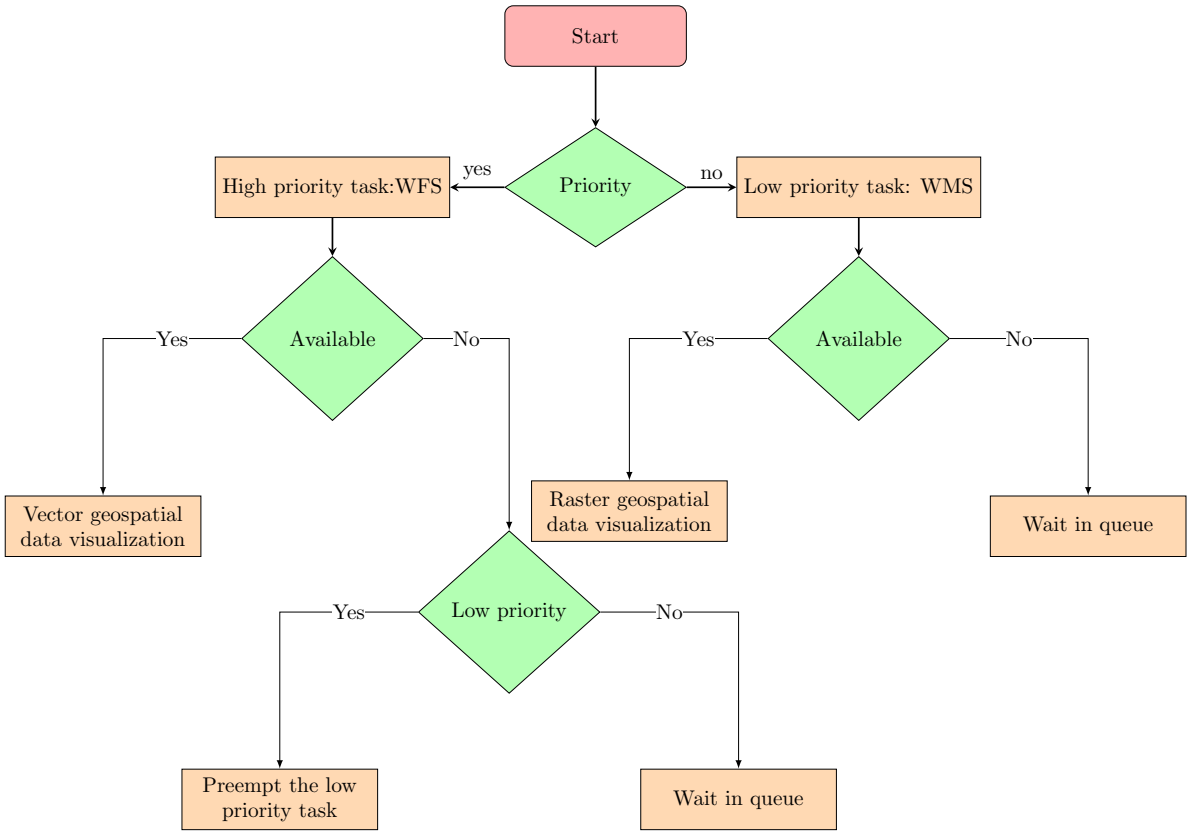


Figure 4. Flow chart of proposed preemptive-resume priority queueing model for WFS and WMS.

In the case of a preemptive priority scheme, system tasks are ranked in order of priority. The moment the high-priority task arrives, a low-priority task in the process is turned out from service immediately. The disrupted task is permitted back into service once the

system has no higher-priority task. As we assume a preemptive resume policy, when the service resumes, it proceeds from where it was disrupted.

Let us assume $\pi(m,n)$ is the steady state of two types of tasks, where the number of high-priority and low-priority tasks are m and n, respectively. The common notations and their representations used across the paper are given in Table 3. We have the following equations in steady-state by applying flow out = flow in:

$$(\lambda_h + \lambda_\ell)\pi(0,0) = \mu_h\pi(1,0) + \mu_\ell\pi(0,1), \tag{1}$$

$$(\lambda_h + \lambda_\ell + \mu_h)\pi(m,0) = \lambda_h\pi(m-1,0) + \mu_h\pi(m+1,0),\ m \geq 1, \tag{2}$$

$$(\lambda_h + \lambda_\ell + \mu_\ell)\pi(0,n) = \lambda_\ell\pi(0,n-1) + \mu_h\pi(1,n) + \mu_\ell\pi(0,n+1),\ n \geq 1, \tag{3}$$

$$\begin{aligned}(\lambda_h + \lambda_\ell + \mu_h)\pi(m,n) &= \lambda_h\pi(m-1,n) + \lambda_\ell\pi(m,n-1) \\ &\quad + \mu_h\pi(m+1,n),\ m,n \geq 1.\end{aligned} \tag{4}$$

Under equilibrium conditions ($\lambda_h < \mu_h$), the probability distribution for the number of tasks of type 1 in the system is

$$\pi(1,n) = \rho_h^n(1-\rho_h),\ \ n \geq 1$$

where $\rho_h = \lambda_h/\mu_h$. For the class 1 tasks, the class 2 tasks do not exist. Thus, we have

$$E(L_h) = \frac{\rho_h}{1-\rho_h}. \tag{5}$$

As the processing times of all tasks are exponentially distributed with the same mean, the complete tasks in the system do not depend on their processing. This number is, therefore, the same as in the system in which all tasks are completed in order of arrival. Hence,

$$E(L_\ell) = \frac{\rho_\ell}{1-\rho_h-\rho_\ell}\left[1 + \frac{\mu_\ell\rho_h}{\mu_h(1-\rho_h)}\right] \tag{6}$$

where $\rho_\ell = \lambda_\ell/\mu_\ell$.

$$E(L_h) + E(L_\ell) = \frac{\rho_h}{1-\rho_h} + \frac{\rho_\ell}{1-\rho_h-\rho_\ell}\left[1 + \frac{\mu_\ell\rho_h}{\mu_h(1-\rho_h)}\right]. \tag{7}$$

The mean number of low-priority tasks in the mist–fog system is

$$\sum_{n=0}^{\infty} n\ \pi(m,n) = \frac{\rho_\ell}{1-\rho_h-\rho_\ell}\left[1 + \frac{\mu_\ell\rho_h}{\mu_h(1-\rho_h)}\right].$$

The mean sojourn time of high-priority tasks in the mist–fog structure is

$$W_h = \frac{1}{\mu_h - \lambda_h} \tag{8}$$

and for low-priority tasks by

$$W_\ell = \frac{1}{\mu_\ell(1-\rho_h-\rho_\ell)}\left[1 + \frac{\mu_\ell\rho_h}{\mu_h(1-\rho_h)}\right]. \tag{9}$$

The average sojourn time in the queue of high-priority task class is given by

$$W_{q,h} = \frac{\rho_h}{\mu_h(1-\rho_h)} \tag{10}$$

The average sojourn time in the queue of low-priority task class is

$$W_{q,\ell} = \frac{1}{\mu_\ell(1-\rho_h-\rho_\ell)}\left[\rho_h + \rho_\ell + \frac{\mu_\ell \rho_h}{\mu_h(1-\rho_h)}\right]. \tag{11}$$

In steady-state, let $E(L_j)$ be the mean number of type-j jobs in the system. The $E(L_k)$ of the kth task class is generalized (Jaiswal 1968 [84]) and is given by

$$E(L_k) = \frac{\rho_k}{1-\sum_{j=1}^{k-1}\rho_j} + \frac{\lambda_k \sum_{j=1}^{k}(\lambda_j/\mu_j^2)}{(1-\sum_{j=1}^{k-1}\rho_j)(1-\sum_{j=1}^{k-2}\rho_j)}. \tag{12}$$

Table 3. Notations used.

Notation	Representation
λ_h	High-priority task arrival rate
λ_ℓ	Low-priority task arrival rate
μ_h	Service rate of high-priority task
μ_ℓ	Service rate of low-priority task
λ	Total task arrival rate
μ	Service rate of total task
ρ	System utilization factor
$E(L_h)$	Average number of high-priority tasks in the system
$E(L_\ell)$	Average number of low-priority tasks in the system
W_h	Average sojourn time of high-priority tasks in the system
W_ℓ	Average sojourn time of low-priority tasks in the system
$W_{q,h}$	Average sojourn time in the queue of high-priority task class
$W_{q,\ell}$	Average sojourn time in the queue of low-priority task class
C_h	Cost of having a task of high-priority class
C_ℓ	Cost of having a task of low-priority class
$\Delta(F(\lambda))$	Expected total cost

Optimal Cost for Task of Priorities

Let us assume that the priorities are pre-assigned. To compare several potential priority tasks, we require the associated cost factors. The optimum allocation of tasks is that for which the total cost is a minimum. Consider that C_h is the cost of having a task of high-priority class, and C_ℓ is the cost of having a task of low-priority class. Here, $C_h, C_\ell \geq O$. Especially, if $C_h = C_\ell$ is equal, then we are looking for a priority allocation that minimizes the expected number of tasks in all classes. In the model discussed here, there are two classes of tasks, specified by arrival and processing rates (λ_i, μ_i) for $i = h, \ell$, and the priority allocation is high for $i = h$ and low for $i = \ell$. Then the expected total cost is

$$F(\lambda_\ell, \mu_\ell, C_\ell; \lambda_h, \mu_h, C_h) = C_\ell E(L_\ell) + C_h E(L_h), \tag{13}$$

Using (5) and (6) in (13), we find the cost function to be

$$\Delta(F(\lambda_\ell)) = F(\lambda_\ell, \mu_\ell, C_\ell; \lambda_h, \mu_h, C_h) = \frac{C_\ell \rho_\ell}{1-\rho_h-\rho_\ell}\left[1 + \frac{\mu_\ell \rho_h}{\mu_h(1-\rho_h)}\right] + \frac{C_h \rho_h}{1-\rho_h}.$$

We alter this priority allocation to low for $i = h$ and high for $i = \ell$ and study the impact of this variation on the cost function (13). Therefore,

$$\Delta(F(\lambda_h)) = F(\lambda_h, \mu_h, C_h; \lambda_\ell, \mu_\ell, C_\ell) = \frac{C_h \rho_h}{1-\rho_\ell-\rho_h}\left[1 + \frac{\mu_h \rho_\ell}{\mu_\ell(1-\rho_\ell)}\right] + \frac{C_\ell \rho_\ell}{1-\rho_\ell}.$$

After simplification, we have

$$\Delta(F(\lambda_\ell)) < \Delta(F(\lambda_h)),$$

this implies that the class h tasks should be allocated higher priority when

$$C_h\,\mu_h \;>\; C_\ell\,\mu_\ell.$$

The algorithm for finding optimal cost for task of priorities is described in Algorithm 1.

Algorithm 1 Algorithm for finding optimal cost for task of priorities

Input: $\lambda_h, \lambda_\ell, \mu_h, \mu_\ell, C_h, C_\ell$.
Output: $\Delta F(\lambda_h), \Delta F(\lambda_\ell)$,

Initialize:
$\rho_h = \frac{\lambda_h}{\mu_h} < 1, \rho_\ell = \frac{\lambda_\ell}{\mu_\ell} < 1.$
$C_h \leftarrow$ Cost of having a task of high-priority class.
$C_\ell \leftarrow$ Cost of having a task of low-priority class.
Compute:
$E(L_h) = \frac{\rho_h}{1-\rho_h}$;
$E(L_\ell) = \frac{\rho_\ell}{1-\rho_h-\rho_\ell}\left[1 + \frac{\mu_\ell \rho_h}{\mu_h(1-\rho_h)}\right]$;
$Z = \Delta(F(\lambda_i)) = C_h\,E(L_h) + C_\ell\,E(L_\ell)$;
Compute:
$\Delta(F(\lambda_\ell)) = \frac{C_\ell \rho_\ell}{1-\rho_h-\rho_\ell}\left[1 + \frac{\mu_\ell \rho_h}{\mu_h(1-\rho_h)}\right] + \frac{C_h \rho_h}{1-\rho_h}$;
$\Delta(F(\lambda_h)) = \frac{C_h \rho_h}{1-\rho_\ell-\rho_h}\left[1 + \frac{\mu_h \rho_\ell}{\mu_\ell(1-\rho_\ell)}\right] + \frac{C_\ell \rho_\ell}{1-\rho_\ell}$;
return Z
exit

4. Numerical Results

To illustrate the analytical results presented herein, some numerical results are illustrated in tables and figures. The calculations were made with double accuracy and performed in a 64-bit windows ten professional OS possessing Intel® Core i5 6200U processor @2:30 GHz and 8 GB DDR3 RAM manufacturer Dell utilizing MAPLE 22 software. We reported the numeric results to only the nearest four digits, but the results were very accurate.

Figure 5 depicts the impact of ρ on the average number of tasks in the system for two priority classes. We observe that in the mist–fog system, the average number of tasks rises with the increase in ρ, and even more so in priority class-2, which is the low-priority class. When there are two priority classes, as far as the higher-priority task is referred, the system performs just like a regular M/M/1 system. Figure 6 describes the impact of ρ on the mean number of tasks in the system for five priority classes. One can see that with an increase in ρ, the mean number of tasks in the system increases. The average number of tasks in the system is less in the case of priority class 1 than in other priority classes. In this case, the system outperforms when the value of ρ is lower.

Figure 7 plots the dependence of ρ on the average waiting time in the queue (W_q). Observe that the system W_q increases with an increase in ρ. For higher values of ρ, the variation in mean waiting time in the queue for both classes increases. The mean waiting time in the queue is less for the higher-priority task than for the low-priority task. However, the impact of the low-priority task is twofold: the queue time and the processing time. Figure 8 illustrates the influence of the mean waiting time in the queue (W_q) on the ρ. With an increase in ρ, we note an increasing trend in all priority classes. For priority class 1, W_q is almost static when ρ is more significant than 0.5. We note that the performance of various classes impacts the system's performance. For instance, the processing time and wait time are more in class 5 than in class 4. Thus, a hierarchy of priorities would arise if there were more classes of tasks.

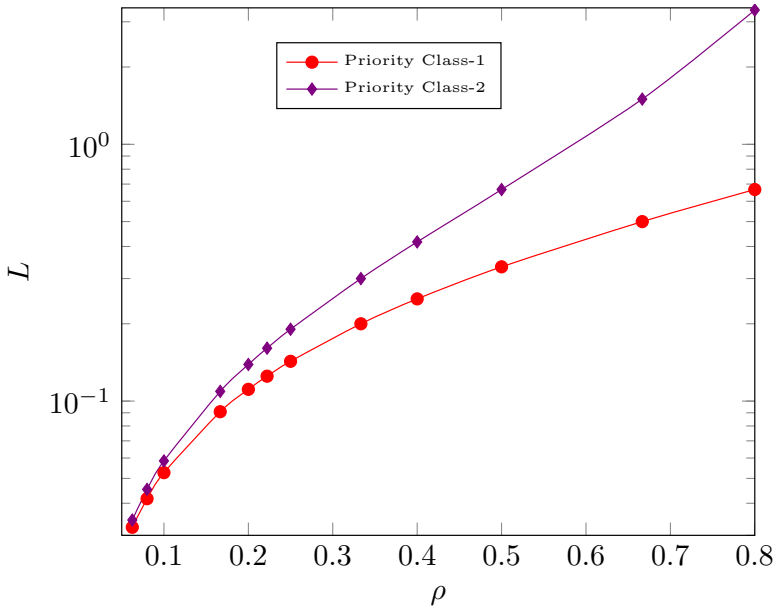


Figure 5. Impact of ρ on L with two priority classes.

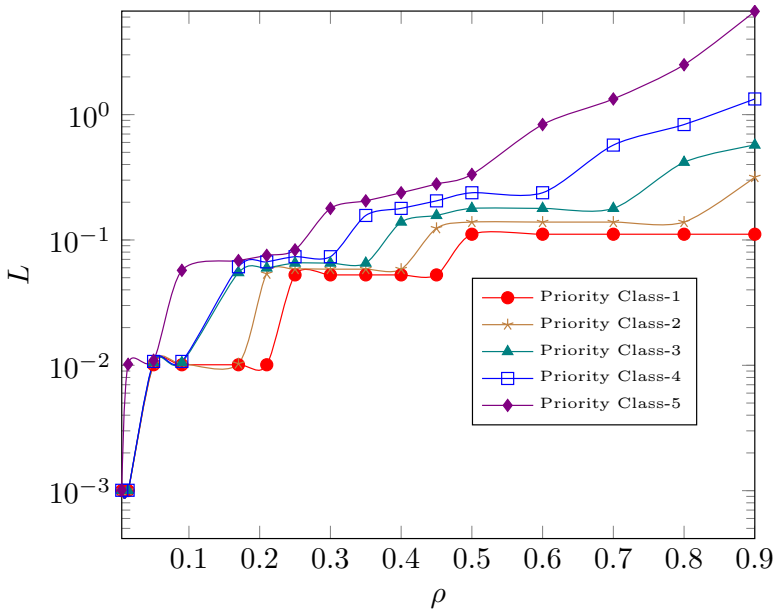


Figure 6. Impact of ρ on L with five priority classes.

Figures 9 and 10 show the impact of λ on server utilization % and the average waiting time in the mist–fog system (W), respectively. From Figure 9, we observe an increasing trend, with an increase in λ. But, with the rise of μ, we see a decreasing trend. Also, the server utilization % increases with the increase in arrival rate with a fixed service rate. Thus, we may carefully assume the arrival and service rate to ensure the balance of the server utilization of the system. From Figure 10, we note an increasing trend, with an increase in λ. For the small value of μ, the average waiting time rises monotonically. Moreover, with a fixed service rate, the average wait time increases with the increase in the arrival rate. To reduce the average wait time of the system, we can meticulously put in place the service and the arrival rate to achieve it.

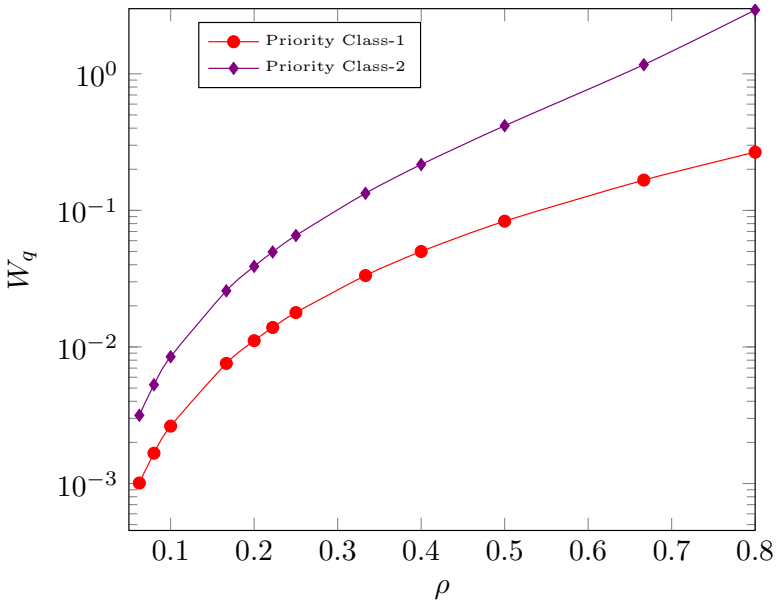


Figure 7. Impact of ρ on W_q with two priority classes.

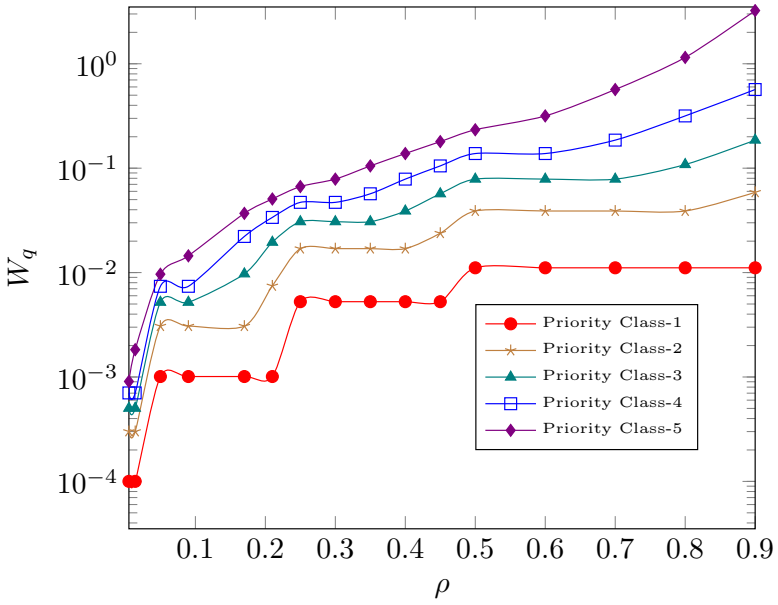


Figure 8. Impact of ρ on W_q with five priority classes.

Tables 4 and 5 present performance measures of the task allocation system of two priority class when $\mu_h = \mu_\ell$ and $\mu_h \neq \mu_\ell$, respectively. In Table 4, we vary λ_1 and assume other parameters as $\lambda_\ell = 5$ and $\mu_h = \mu_\ell = 10.9091$. Note that with the gain of λ_h, the performance indices increases. Comparing priority type 1 and type 2, observe that the mean number of tasks in the queue (system) and the average waiting time in the queue (system) is less in the case of priority class 1. In Table 6, we vary λ_h and assume other parameters as $\lambda_\ell = 5$, $\mu_h = 10.9091$, and $\mu_\ell = 10.9091$. We also compared the system when there was no priority task. The relevant results are presented in the second column of the tables as the overall result.

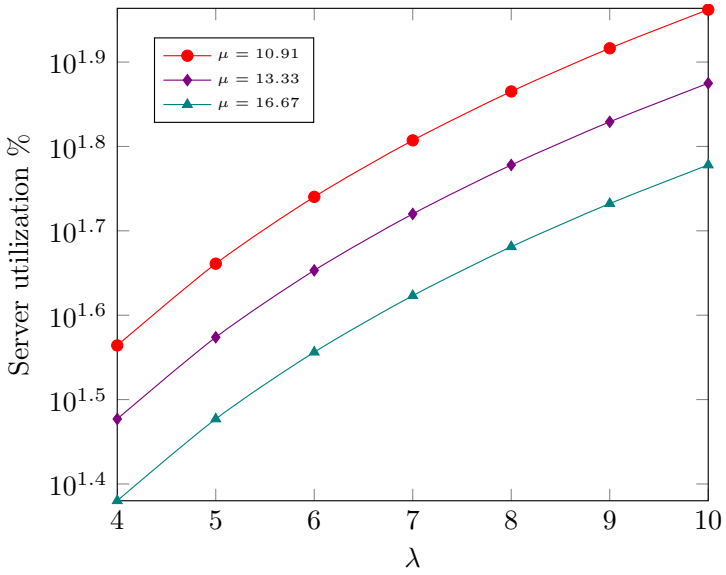


Figure 9. Effect of λ on percentage of server utilization.

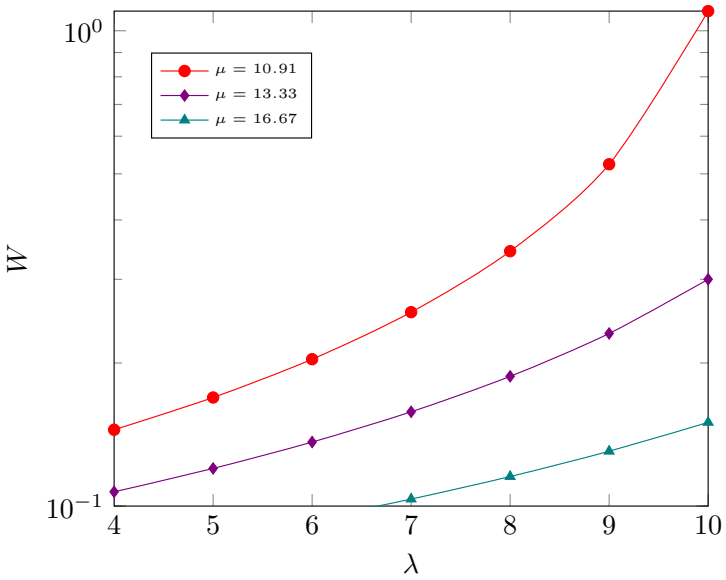


Figure 10. Effect of λ on W.

Table 4. Performance measures of two priority classes.

	$\lambda_\ell = 5, \mu_h = \mu_\ell = 10.9091$					
	$\lambda_h = 1$			$\lambda_h = 2$		
	Overall	**Type-1**	**Type-2**	**Overall**	**Type-1**	**Type-2**
L_s	**1.222222**	0.100917	1.121305	1.790698	0.22449	1.566208
L_q	**0.672222**	0.009251	0.662971	1.149031	0.041156	1.107875
W_s	**0.203704**	0.100917	0.224261	0.255814	0.112245	0.313242
W_q	**0.112037**	0.009251	0.132594	0.164147	0.020578	0.221575
	$\lambda_h = 3$			$\lambda_h = 4$		
	Overall	Type-1	Type-2	Overall	Type-1	Type-2
L_s	**2.75**	0.37931	2.37069	4.714286	0.578947	4.135338
L_q	**2.016667**	0.10431	1.912356	3.889286	0.212281	3.677005
W_s	**0.34375**	0.126437	0.474138	0.52381	0.144737	0.827068
W_q	**0.252083**	0.03477	0.382471	0.432143	0.05307	0.735401

Table 5. Performance measures of two priority classes when $\mu_h \neq \mu_\ell$.

	$\lambda_\ell = 5, \mu_h = 10.9091, \mu_\ell = 12$					
	$\lambda_h = 1$			$\lambda_h = 2$		
	Overall	**Type-1**	**Type-2**	**Overall**	**Type-1**	**Type-2**
L_s	**0.165636**	0.100917	0.064719	0.305812	0.224489	0.081322
L_q	**0.023970**	0.009251	0.014719	0.072478	0.041156	0.031322
W_s	**0.103523**	0.100917	0.107865	0.117619	0.112245	0.135537
W_q	**0.014981**	0.009251	0.024532	0.027876	0.020578	0.052203
	$\lambda_h = 3$			$\lambda_h = 4$		
	Overall	Type-1	Type-2	Overall	Type-1	Type-2
L_s	**0.484291**	0.37931	0.104981	0.719247	0.578947	0.140301
L_q	**0.159291**	0.10431	0.054981	0.302581	0.21228	0.090301
W_s	**0.134525**	0.126436	0.174968	0.156358	0.144737	0.233834
W_q	**0.044247**	0.03477	0.091635	0.065778	0.05307	0.150501

Table 6. Performance measures of five priority classes.

	$\lambda_1 = 1, \lambda_2 = \lambda_3 = \lambda_4 = \lambda_5 = 2, \lambda = 5, \mu = 10, \rho = 0.9$					
	Overall	**Type-1**	**Type-2**	**Type-3**	**Type-4**	**Type-5**
L_s	**9**	0.111111	0.31746	0.571429	1.333333	6.666667
L_q	**8.1**	0.011111	0.11746	0.371429	1.133333	6.466667
W_s	**1**	0.111111	0.15873	0.285714	0.666667	3.333333
W_q	**0.9**	0.011111	0.05873	0.185714	0.566667	3.233333
	$\lambda_1 = 0.5, \lambda_2 = \lambda_3 = \lambda_4 = \lambda_5 = 1, \lambda = 5, \mu = 10, \rho = 0.45$					
	Overall	Type-1	Type-2	Type-3	Type-4	Type-5
L_s	**0.818182**	0.052632	0.123839	0.156863	0.205128	0.27972
L_q	**0.368182**	0.002632	0.023839	0.056863	0.105128	0.17972
W_s	**0.181818**	0.105263	0.123839	0.156863	0.205128	0.27972
W_q	**0.081818**	0.005263	0.023839	0.056863	0.105128	0.17972

Figures 11 and 12 demonstrate the impact of processing rate μ_h on total cost of low- and high-priority tasks when $\lambda_h = \lambda_\ell$ and $\lambda_h \neq \lambda_\ell$, respectively. One may note that the total cost decreases as μ_h increases in both cases. When $\lambda_h = \lambda_\ell$, the total cost of the low-priority task decreases more rapidly for higher values of μ_h in comparison to the high-priority task. But the result is just reversed in the case of $\lambda_h \neq \lambda_\ell$.

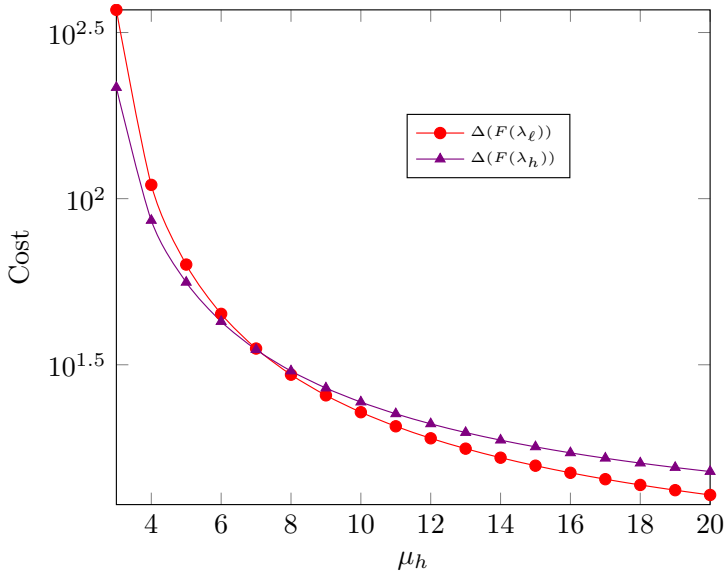


Figure 11. Cost function when $\lambda_h = \lambda_\ell$.

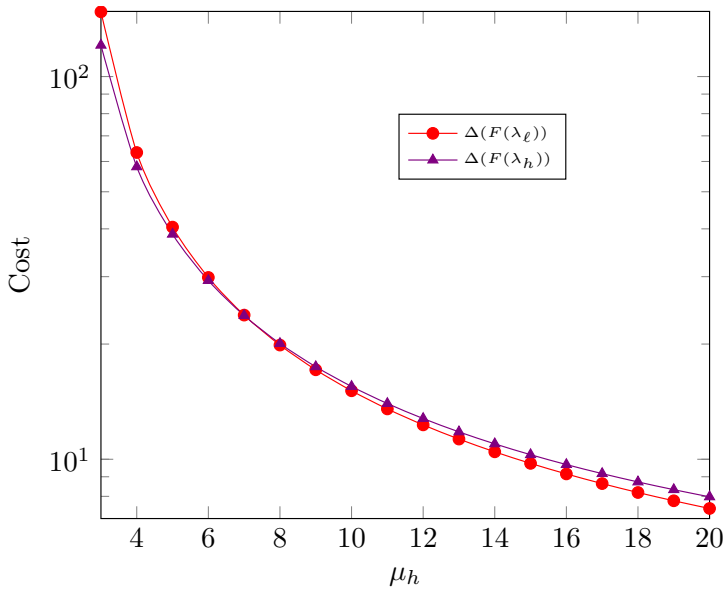


Figure 12. Cost function when $\lambda_h \neq \lambda_\ell$.

5. Concluding Remarks

In this article, the concepts of cloud computing, fog computing, and mist computing for geospatial web services, in particular WMS and WFS, are analyzed and explored. This paper proposed the preemptive-resume priority queueing strategy for the mist–cloud–fog system and associated components for improved data processing and analysis in geospatial web applications. Because of the mist and fog nodes, the number of geospatial data that need to be stored as well as processed is cut down, which results in transmission that is both efficient and has a lower latency and throughput. Additionally, a priority-based queuing strategy was presented in order to limit the dynamism of the suggested model and conduct analysis on it. Using the proper diagrams, the performance analysis, performance assessment, and performance measurement of the provided framework, in addition to the experimental results, have been discussed.

The proposed model is going to be put to the test in future application-oriented case studies, which will will include a wide range of parameters. The model that was suggested incorporates, among other things, specific data regarding the utilization of the CPU, the response time, the loss rate, the throughput, and the average number of jobs that were requested.

Author Contributions: Conceptualization, S.K.P., R.K.B. and V.G.; writing—review and editing, V.G., H.K.A. and R.K.B. ; Methodology, H.D. and G.B.M.; Software, S.K.P. and V.G.; Validation, G.B.M. and R.K.B.; Formal analysis, V.G. and G.B.M.; Resources, V.G. and H.K.A.; Visualization, S.K.P. and H.D. All authors have read and agreed to the published version of the manuscript

Funding: This research received no external funding.

Data Availability Statement: Not applicable.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Armstrong, M.P. High performance computing for geospatial applications: A retrospective view. In *High Performance Computing for Geospatial Applications*; Springer: Berlin/Heidelberg, Germany , 2020; pp. 9–25
2. Barik, R.K.; Dubey, H.; Mankodiya, K.; Sasane, S.A.; Misra, C. Geofog4health: A fog-based sdi framework for geospatial health big data analysis. *J. Ambient. Intell. Humaniz. Comput.* **2019**, *10*, 551–567. [CrossRef]
3. Goswami, V.; Sharma, B.; Patra, S.S.; Chowdhury, S.; Barik, R.K.; Dhaou, I.B. Iot-fog computing sustainable system for smart cities: A queueing-based approach. In Proceedings of the 2023 1st International Conference on Advanced Innovations in Smart Cities (ICAISC), Jeddah, Saudi Arabia, 23–25 January 2023; pp. 1–6.

4. Mukherjee, M.; Guo, M.; Lloret, J.; Iqbal, R.; Zhang, Q. Deadline-aware fair scheduling for offloaded tasks in fog computing with inter-fog dependency. *IEEE Commun. Lett.* **2019**, *24*, 307–311. [CrossRef]
5. Nikoui, T.S.; Rahmani, A.M.; Balador, A.; Javadi, H.H.S. Analytical model for task offloading in a fog computing system with batch-size-dependent service. *Comput. Commun.* **2022**, *190*, 201–215. [CrossRef]
6. Geobuiz 23: Global Geospatial Industry Market Size, Forecast, and Growth Trends Report. Available online: https://geospatialworld.net/consulting/reports/geobuiz/2023/index.html (accessed on 17 March 2023).
7. Geospatial Analytics Market Size & Share Analysis—Growth Trends & Forecasts (2023–2028). Available online: https://www.mordorintelligence.com/industry-reports/geospatial-analytics-market (accessed on 17 March 2023).
8. Bhushan, S.; Mat, M. Priority-queue based dynamic scaling for efficient resource allocation in fog computing. In Proceedings of the 2021 IEEE International Conference on Service Operations and Logistics, and Informatics (SOLI), Delhi, India, 2–4 December 2021; pp. 1–6.
9. Golkar, A.; Malekhosseini, R.; RahimiZadeh, K.; Yazdani, A.; Beheshti, A. A priority queue-based telemonitoring system for automatic diagnosis of heart diseases in integrated fog computing environments. *Health Inform. J.* **2022**, *28*, 14604582221137453. [CrossRef] [PubMed]
10. Barik, R.K.; Dubey, A.C.; Tripathi, A.; Pratik, T.; Sasane, S.; Lenka, R.K.; Dubey, H.; Mankodiya, K.; Kumar, V. Mist data: Leveraging mist computing for secure and scalable architecture for smart and connected health. *Procedia Comput. Sci.* **2018**, *125*, 647–653. [CrossRef]
11. Hmissi, F.; Ouni, S. An mqtt brokers distribution based on mist computing for real-time iot communications. *Res. Sq. preprint.* **2021**. [CrossRef]
12. Maiti, P.; Sahoo, B.; Turuk, A.K.; Kumar, A.; Choi, B.J. Internet of things applications placement to minimize latency in multi-tier fog computing framework. *ICT Express* **2022**, *8*, 166–173. [CrossRef]
13. Mallick, S.R.; Lenka, R.K.; Goswami, V.; Sharma, S.; Dalai, A.K.; Das, H.; Barik, R.K. Bcgeo: Blockchain-assisted geospatial web service for smart healthcare system. *IEEE Access* **2023**, *11*, 58610–58623. [CrossRef]
14. Arefian, Z.; Khayyambashi, M.R.; Movahhedinia, N. Delay reduction in mtc using sdn based offloading in fog computing. *PLoS ONE* **2023**, *18*, e0286483. [CrossRef]
15. Cai, P.; Jiang, Q. Gis spatial information sharing of smart city based on cloud computing. *Clust. Comput.* **2019**, *22*, 14435–14443. [CrossRef]
16. Das, J.; Ghosh, S.K.; Buyya, R. Geospatial edge-fog computing: A systematic review, taxonomy, and future directions. In *Mobile Edge Computing*; Springer: Berlin/Heidelberg, Germany, 2021; pp. 47–69.
17. Fareed, N.; Rehman, K. Integration of remote sensing and gis to extract plantation rows from a drone-based image point cloud digital surface model. *ISPRS Int. J. -Geo-Inf.* **2020**, *9*, 151. [CrossRef]
18. Shahid, H.; Shah, M.A.; Almogren, A.; Khattak, H.A.; Din, I.U.; Kumar, N.; Maple, C. Machine learning-based mist computing enabled internet of battlefield things. *ACM Trans. Internet Technol. (TOIT)* **2021**, *21*, 1–26. [CrossRef]
19. He, Z.; Xu, Y.; Liu, D.; Zhou, W.; Li, K. Energy-efficient computation offloading strategy with task priority in cloud assisted multi-access edge computing. *Future Gener. Comput. Syst.* **2023**, *148*, 298–313. [CrossRef]
20. Chavhan, S.; Gupta, D.; Gochhayat, S.P.; Khanna, A.; Shankar, K.; Rodrigues, J.J. Edge computing ai-iot integrated energy-efficient intelligent transportation system for smart cities. *ACM Trans. Internet Technol.* **2022**, *22*, 1–18. [CrossRef]
21. Bouanaka, C.; Laouir, A.E.; Medkour, R. Iedss: Efficient scheduling of emergency department resources based on fog computing. In Proceedings of the 2020 IEEE/ACS 17th International Conference on Computer Systems and Applications (AICCSA), Antalya, Turkey, 2–5 November 2020; pp. 1–6.
22. Dutta, A.; Misra, C.; Barik, R.K.; Mishra, S. Enhancing mist assisted cloud computing toward secure and scalable architecture for smart healthcare. In *Advances in Communication and Computational Technology*; Springer: Berlin/Heidelberg, Germany, 2021; pp. 1515–1526.
23. Barik, R.K.; Misra, C.; Lenka, R.K.; Dubey, H.; Mankodiya, K. Hybrid mist-cloud systems for large scale geospatial big data analytics and processing: Opportunities and challenges. *Arab. J. Geosci.* **2019**, *12*, 32. [CrossRef]
24. Das, J.; Mukherjee, A.; Ghosh, S.K.; Buyya, R. Spatio-fog: A green and timeliness-oriented fog computing model for geospatial query resolution. *Simul. Model. Pract. Theory* **2020**, *100*, 102043. [CrossRef]
25. Etemadi, M.; Ghobaei-Arani, M.; Shahidinejad, A. Resource provisioning for iot services in the fog computing environment: An autonomic approach. *Comput. Commun.* **2020**, *161*, 109–131. [CrossRef]
26. Silva, F.A.; Fé, I.; Gonçalves, G. Stochastic models for performance and cost analysis of a hybrid cloud and fog architecture. *J. Supercomput.* **2021**, *77*, 1537–1561. [CrossRef]
27. Sharma, S.; Saini, H. A novel four-tier architecture for delay aware scheduling and load balancing in fog environment. *Sustain. Comput. Inform. Syst.* **2019**, *24*, 100355. [CrossRef]
28. Wang, T.; Liang, Y.; Jia, W.; Arif, M.; Liu, A.; Xie, M. Coupling resource management based on fog computing in smart city systems. *J. Netw. Comput. Appl.* **2019**, *135*, 11–19. [CrossRef]
29. Alli, A.A.; Alam, M.M. Secoff-fciot: Machine learning based secure offloading in fog-cloud of things for smart city applications. *Internet Things* **2019**, *7*, 100070. [CrossRef]

30. El Kafhali, S.; Salah, K. Efficient and dynamic scaling of fog nodes for iot devices. *J. Supercomput.* **2017**, *73*, 5261–5284. [CrossRef]
31. El Kafhali, S.; Salah, K. Modeling and analysis of performance and energy consumption in cloud data centers. *Arab. J. Sci. Eng.* **2018**, *43*, 7789–7802. [CrossRef]
32. Zhang, C. Design and application of fog computing and internet of things service platform for smart city. *Future Gener. Comput. Syst.* **2020**, *112*, 630–640. [CrossRef]
33. Ghobaei-Arani, M.; Souri, A.; Rahmanian, A.A. Resource management approaches in fog computing: A comprehensive review. *J. Grid Comput.* **2019**, *18* , 1–42. [CrossRef]
34. Yousefpour, A.; Fung, C.; Nguyen, T.; Kadiyala, K.; Jalali, F.; Niakanlahiji, A.; Kong, J.; Jue, J.P. All one needs to know about fog computing and related edge computing paradigms: A complete survey. *J. Syst. Archit.* **2019**, *98*, 289–330. [CrossRef]
35. Evangelidis, K.; Ntouros, K.; Makridis, S.; Papatheodorou, C. Geospatial services in the cloud. *Comput. Geosci.* **2014**, *63*, 116–122. [CrossRef]
36. Barik, R.K. Cloudganga: Cloud computing based sdi model for ganga river basin management in india. In *Geospatial Intelligence: Concepts, Methodologies, Tools, and Applications*; IGI Global: Hershey, PA, USA, 2019; pp. 278–297.
37. Wieclaw, L.; Pasichnyk, V.; Kunanets, N.; Duda, O.; Matsiuk, O.; Falat, P. Cloud computing technologies in "smart city" projects. In Proceedings of the 2017 9th IEEE International Conference on Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications (IDAACS), Bucharest, Romania, 21–23 September 2017; Volume 1, pp. 339–342.
38. Liang, J.; Jin, F.; Zhang, X.; Wu, H. Ws4gee: Enhancing geospatial web services and geoprocessing workflows by integrating the google earth engine. *Environ. Model. Softw.* **2023**, *161*, 105636. [CrossRef]
39. AL Kharouf, R.A.; Alzoubaidi, A.R.; Jweihan, M. An integrated architectural framework for geoprocessing in cloud environment. *Spat. Inf. Res.* **2017**, *25*, 89–97. [CrossRef]
40. Barik, R.K.; Lenka, R.; Sahoo, S.; Das, B.; Pattnaik, J. Development of educational geospatial database for cloud sdi using open source gis. In *Progress in Advanced Computing and Intelligent Engineering*; Springer: Berlin/Heidelberg, Germany, 2018; pp. 685–695.
41. Goldberg, D.; Olivares, M.; Li, Z.; Klein, A.G. Maps & gis data libraries in the era of big data and cloud computing. *J. Map Geogr. Libr.* **2014**, *10*, 100–122.
42. Zhang, J.; Xu, L.; Zhang, Y.; Liu, G.; Zhao, L.; Wang, Y. An on-demand scalable model for geographic information system (gis) data processing in ancloud gis. *ISPRS Int. J. -Geo-Inf.* **2019**, *8*, 392. [CrossRef]
43. Khazaei, H.; Misic, J.; Misic, V.B. Performance analysis of cloud computing centers using M/G/m/m+ r queuing systems. *IEEE Trans. Parallel Distrib. Syst.* **2011**, *23*, 936–943. [CrossRef]
44. Ellens, W.; Akkerboom, J.; Litjens, R.; Van Den Berg, H. Performance of cloud computing centers with multiple priority classes. In Proceedings of the 2012 IEEE Fifth International Conference on Cloud Computing, Honolulu, HI, USA, 24–29 June 2012; pp. 245–252.
45. Do, C.T.; Tran, N.H.; VanNguyen, M.; Hong, C.S.; Lee, S. Social optimization strategy in unobserved queueing systems in cognitive radio networks. *IEEE Commun. Lett.* **2012**, *16*, 1944–1947. [CrossRef]
46. Salah, K. A queueing model to achieve proper elasticity for cloud cluster jobs. In Proceedings of the 2013 IEEE Sixth International Conference on Cloud Computing, Santa Clara, CA, USA, 28 June–3 July 2013; pp.755–761.
47. Pal, R.; Hui, P. Economic models for cloud service markets: Pricing and capacity planning. *Theor. Comput. Sci.* **2013**, *496*, 113–124. [CrossRef]
48. Mohanty, S.; Pattnaik, P.K.; Mund, G.B. A comparative approach to reduce the waiting time using queuing theory in cloud computing environment. *Int. J. Inf. Comput. Technol.* **2014**, *4*, 469–474.
49. Chiang, Y.J.; Ouyang, Y.C.; Hsu, C.H. Performance and cost-effectiveness analyses for cloud services based on rejected and impatient users. *IEEE Trans. Serv. Comput.* **2014**, *9*, 446–455. [CrossRef]
50. Evangelin, K.R.; Vidhya, V. Performance measures of queuing models using cloud computing. *Asian J. Eng. Appl. Technol.* **2015**, *4*, 8–11. [CrossRef]
51. Cheng, C.; Li, J.; Wang, Y. An energy-saving task scheduling strategy based on vacation queuing theory in cloud computing. *Tsinghua Sci. Technol.* **2015**, *20*, 28–39. [CrossRef]
52. Bai, W.H.; Xi, J.Q.; Zhu, J.X.; Huang, S.W. Performance analysis of heterogeneous data centers in cloud computing using a complex queuing model. *Math. Probl. Eng.* **2015**, *2015*, 980945 . [CrossRef]
53. Kirsal, Y.; Ever, Y.K.; Mostarda, L.; Gemikonakli, O. Analytical modelling and performability analysis for cloud computing using queuing system. In Proceedings of the 2015 IEEE/ACM 8th International Conference on Utility and Cloud Computing (UCC), Limassol, Cyprus, 7–10 December 2015; pp. 643–647.
54. Guo, L.; Yan, T.; Zhao, S.; Jiang, C. Dynamic performance optimization for cloud computing using M/M/m queueing system. *J. Appl. Math.* **2014**, *2014*, 756592. [CrossRef]
55. Akbari, E.; Cung, F.; Patel, H.; Razaque, A.; Dalal, H.N. Incorporation of weighted linear prediction technique and M/M/1 queuing theory for improving energy efficiency of cloud computing datacenters. In Proceedings of the 2016 IEEE Long Island Systems, Applications and Technology Conference (LISAT), Farmingdale, NY, USA, 29 April 2016; pp. 1–5.

56. Chang, Z.; Zhou, Z.; Ristaniemi, T.; Niu, Z. Energy efficient optimization for computation offloading in fog computing system. In Proceedings of the GLOBECOM 2017–2017 IEEE Global Communications Conference, Singapore, 4–8 December 2017; pp. 1–6.
57. Liu, L.; Chang, Z.; Guo, X.; Mao, S.; Ristaniemi, T. Multiobjective optimization for computation offloading in fog computing. *IEEE Internet Things J.* **2017**, *5*, 283–294. [CrossRef]
58. Safvati, M.; Sharzehei, M. Analytical review on queuing theory in clouds environments. In Proceedings of the Third National Conference on New Approaches in Computer and Electrical Engineering Young Researchers and Elite Club , 2017. Available online: https://www.researchgate.net/publication/316438195_Analytical_Review_on_Queuing_Theory_in_Clouds_Enviroments (accessed on 17 March 2023).
59. Tadakamalla, U.; Menascé, D. Fogqn: An analytic model for fog/cloud computing. In Proceedings of the 2018 IEEE/ACM International Conference on Utility and Cloud Computing Companion (UCC Companion), Zurich, Switzerland, 17–20 December 2018; pp. 307–313
60. Sthapit, S.; Thompson, J.; Robertson, N.M.; Hopgood, J.R. Computational load balancing on the edge in absence of cloud and fog. *IEEE Trans. Mob. Comput.* **2018**, *18*, 1499–1512. [CrossRef]
61. Chunxia, Y.; Shunfu, J. An energy-saving strategy based on multi-server vacation queuing theory in cloud data center. *J. Supercomput.* **2018**, *74*, 6766–6784. [CrossRef]
62. Sopin, E.S.; Daraseliya, A.V.; Correia, L.M. Performance analysis of the offloading scheme in a fog computing system. In Proceedings of the 2018 10th International Congress on Ultra Modern Telecommunications and Control Systems and Workshops (ICUMT), Moscow, Russia, 5–9 November 2018; pp. 1–5.
63. Vasconcelos, D.R.D. Smart shadow-predictive computing resources allocation for smart devices in the mist computing environment. Ph.D. Dissertation, Universidade Federal Do Ceará, Fortaleza, Brazil, 2018.
64. Jafarnejad Ghomi, E.; Rahmani, A.M.; Qader, N.N. Applying queue theory for modeling of cloud computing: A systematic review. *Concurr. Comput. Pract. Exp.* **2019**, *31*, e5186. [CrossRef]
65. Li, G.; Yan, J.; Chen, L.; Wu, J.; Lin, Q.; Zhang, Y. Energy consumption optimization with a delay threshold in cloud-fog cooperation computing. *IEEE access* **2019**, *7*, 159688–159697. [CrossRef]
66. Kumar, M.S.; Raja, M.I. A queuing theory model for e-health cloud applications. *Int. J. Internet Technol. Secur. Trans.* **2020**, *10*, 585–600. [CrossRef]
67. Xu, R.; Wu, J.; Cheng, Y.; Liu, Z.; Lin, Y.; Xie, Y. Dynamic security exchange scheduling model for business workflow based on queuing theory in cloud computing. *Secur. Commun. Netw.* **2020**, *2020*, 8886640. [CrossRef]
68. Patra, S.; Amodi, S.A.; Goswami, V.; Barik, R. Profit maximization strategy with spot allocation quality guaranteed service in cloud environment. In Proceedings of the 2020 International Conference on Computer Science, Engineering and Applications (ICCSEA), Gunupur, India, 13–14 March 2020; pp. 1–6.
69. Sedaghat, S.; Jahangir, A.H. Rt-telsurg: Real time telesurgery using sdn, fog, and cloud as infrastructures. *IEEE Access* **2021**, *9*, 52238–52251. [CrossRef]
70. Sufyan, F.; Banerjee, A. Computation offloading for smart devices in fog-cloud queuing system. *IETE J. Res.* **2021**, *69*, 1509–1521. [CrossRef]
71. Tadakamalla, U.; Menasce, D.A. Autonomic resource management for fog computing. *IEEE Trans. Cloud Comput.* **2021**, *11*, 2334–2350. [CrossRef]
72. Feitosa, L.; Santos, L.; Gonçalves, G.; Nguyen, T.A.; Lee, J.W.; Silva, F.A. Internet of robotic things: A comparison of message routing strategies for cloud-fog computing layers using m/m/c/k queuing networks. In Proceedings of the 2021 IEEE International Conference on Systems, Man, and Cybernetics (SMC), Melbourne, Australia, 17–20 October 2021; pp. 2049–2054.
73. Panigrahi, S.K.; Barik, R.K.; Behera, S.; Barik, L.; Patra, S.S. Performability analysis of foggis model for geospatial web services. In Proceedings of the 2021 11th International Conference on Cloud Computing, Data Science & Engineering (Confluence), Noida, India, 28–29 January 2021; pp. 239–243.
74. Behera, S.; Al Amodi, S.; Patra, S.S.; Lenka, R.K.; Goje, N.S.; Barik, R.K. Profit maximization scheme in iot assisted mist computing healthcare environment using M/G/c/N queueing model. In Proceedings of the 2021 IEEE International Conference on Electronics, Computing and Communication Technologies (CONECCT), Bangalore, India, 9–11 July 2021; pp. 1–6.
75. Mas, L.; Vilaplana, J.; Mateo, J.; Solsona, F. A queuing theory model for fog computing. *J. Supercomput.* **2022**, *78*, 11138–11155. [CrossRef]
76. Rodrigues, L.; Rodrigues, J.J.; Serra, A.D.B.; Silva, F.A. A queueing-based model performance evaluation for internet of people supported by fog computing. *Future Internet* **2022**, *14*, 23. [CrossRef]
77. Hamdi, A.M.A.; Hussain, F.K.; Hussain, O.K. Task offloading in vehicular fog computing: State-of-the-art and open issues. *Future Gener. Comput. Syst.* **2022**, *133*, 201–212. [CrossRef]
78. Hazra, A.; Rana, P.; Adhikari, M.; Amgoth, T. Fog computing for next-generation internet of things: Fundamental, state-of-the-art and research challenges. *Comput. Sci. Rev.* **2023**, *48*, 100549. [CrossRef]
79. Yazdani, A.; Dashti, S.F.; Safdari, Y. A fog-assisted information model based on priority queue and clinical decision support systems. *Health Inform. J.* **2023**, *29*, 14604582231152792. [CrossRef] [PubMed]
80. Saif, F.A.; Latip, R.; Hanapi, Z.M.; Alrshah, M.A.; Shafinah, K. Workload allocation towards energy consumption-delay trade-off in cloud-fog computing using multi-objective npso algorithm. *IEEE Access* **2023**, *11*, 45393–45404. [CrossRef]

81. Saif, F.A.; Latip, R.; Hanapi, Z.M.; Shafinah, K. Multi-objective grey wolf optimizer algorithm for task scheduling in cloud-fog computing. *IEEE Access* **2023**, *11*, 20635–20646. [CrossRef]
82. Munir, A.; Kansakar, P.; Khan, S. Ifciot: Integrated fog cloud iot architectural paradigm for future iots. *arXiv* **2017**, arXiv:1701.08474.
83. Adhikari, M.; Mukherjee, M.; Srirama, S.N. Dpto: A deadline and priority-aware task offloading in fog computing framework leveraging multilevel feedback queueing. *IEEE Internet Things J.* **2019**, *7*, 5773–5782. [CrossRef]
84. Jaiswal, N.K. *Priority Queues*; Academic Press: New York, NY, USA, 1968; Volume 50.

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.