Simulation Modelling Practice and Theory 100 (2020) 102043

Contents lists available at ScienceDirect

Simulation Modelling Practice and Theory

journal homepage: www.elsevier.com/locate/simpat

Spatio-Fog: A green and timeliness-oriented fog computing model for geospatial query resolution

Jaydeep Das*,[a], Anwesha Mukherjee[b], Soumya K. Ghosh[c], Rajkumar Buyya[d]

[a] Advanced Technology Development Centre, Indian Institute of Technology Kharagpur, West Bengal 721302, India
[b] Department of Computer Science, Mahishadal Raj College, Garkamalpur, Mahishadal, Purba Medinipur, West Bengal 721628, India
[c] Department of Computer Science and Engineering, Indian Institute of Technology Kharagpur, West Bengal 721302, India
[d] Cloud Computing and Distributed Systems (CLOUDS) Laboratory, School of Computing and Information Systems, The University of Melbourne, VIC 3010, Australia

ARTICLE INFO

Keywords:
Geospatial query
Fog computing
Cloud computing
Delay-sensitive
Power-efficient

ABSTRACT

Geospatial data analysis is an emerging area of research today. Systems need to respond to user requests in a timely manner. In this paper we have proposed a fog computing framework namely Spatio-Fog, where the fog devices contain the geospatial data of their current region and process geospatial queries using resources in the proximity. The geospatial query resolution is performed by the fog device either itself or using cloud servers or fog device of other region depending on the geographical region related to the geospatial query. We have performed both empirical study and experimental analysis to demonstrate feasibility of our proposed approach. The empirical study illustrates that the proposed architecture Spatio-Fog reduces the power consumption and delay by approximately 43–47% and 47–83% respectively over the use of existing geospatial query resolution system. The experimental analysis demonstrates that the proposed framework reduces the power consumption and delay by 30–60% approximately than the existing geospatial query resolution system.

1. Introduction

Geospatial information storage, processing, and query resolution is a promising research area [1]. There are several industrial applications of geospatial data analysis such as mapping, telecom and network services, hot spot analysis, urban planning, transport services, environmental impact analysis, health and human services, disaster management, resource management, geology. Google Maps is a popular application of a web-based mapping solution on geospatial data, which is used for navigation services. Geospatial data analysis also helps in road traffic management, transport and urban planning. For risk assessment and disaster management geospatial data analysis also important. Usually, cloud servers are used to store and process the geospatial information [2]. However, as geospatial information is related to geographical regions, storing and processing large volume geospatial data inside the remote cloud servers can suffer from delay and energy consumption. Geospatial query processing involves geospatial data analysis along with different geospatial services, which have been discussed later in Section 3. While mobile devices request for any geospatial information, then to process the respective geospatial queries geospatial big data analysis is required. However, accessing geospatial data inside the remote cloud servers may degrade the quality of user experience by enhancing delay [3] and energy. Moreover,

* Corresponding author.
E-mail addresses: jaydeep@iitkgp.ac.in (J. Das), anweshamukherjee2011@gmail.com (A. Mukherjee), skg@cse.iitkgp.ac.in (S.K. Ghosh), rbuyya@unimelb.edu.au (R. Buyya).

https://doi.org/10.1016/j.simpat.2019.102043
Received 2 October 2019; Received in revised form 28 November 2019; Accepted 14 December 2019
Available online 16 December 2019
1569-190X/

storing and processing a massive amount of data on cloud data centres results in high energy consumption. The use of long distant cloud servers also compromise with Quality of Service (QoS) in terms of delay and energy, from the perspective of the client mobile devices [4,5]. Fog computing has been introduced to improve the QoS, where the intermediate devices between the end node and cloud servers, e.g., switch, router, perform data and computation offloading [6–8]. The intermediate devices which store and process data are called fog devices [9,10]. Use of fog devices in geospatial data storage and processing can reduce the delay and energy consumption over remote cloud servers.

Motivation and contribution:

Geospatial query processing at lower time and power consumption of user device is very much needed in various applications such as navigation, health and human services. For example, a user is travelling in a region for the first time and he/she is not feeling well, hence needs an information regarding the nearby hospital. In such scenarios, the user needs an immediate response. Access to remote cloud may cause delay. However, if the nearby fog device has that information, the user can get the response earlier.

The use of fog based framework in different applications such as health-care, home monitoring, has reduced the delay and energy consumption over the only cloud-based system. In these types of applications, the intermediate devices perform preliminary data processing. However, for large scale data storage and processing, cloud is still the only option. The major challenges of geospatial application are [1,11–14]:

- For geospatial data analysis, a large volume of multidimensional data has to be processed.
- For exploration, multiple approximate geospatial queries are involved in rapid succession.
- The approximate geospatial queries involve ranges specified along the spatio-temporal dimensions. The geospatial query evaluations require other concurrent rigorous and approximate geospatial queries.

The traditional fog computing based model unable to provide satisfactory QoS in case of geospatial query resolution. In [15,16], fog device is used for partial computation on geospatial data. However, the geospatial data storage and large scale processing are performed inside the cloud, and the load on cloud servers remains high, whereas resourceful fog devices remain underutilized. Hence, there should be a method for distributing the voluminous data among the fog devices and cloud, so that the resources of the fog nodes will be utilized as well as the load on the cloud will be reduced. Our objective is to introduce a new hierarchical fog based framework for the geospatial application, which will distribute the voluminous data among the fog devices and cloud servers. In such way, the load on cloud data centre will be reduced as well as the geospatial query processing time will be minimal. The contributions of this paper are:

- A fog computing based architecture is proposed where fog devices store the regional geospatial data based on the pre-calculated volume of data of the regions. The proposed system is referred as Spatio-Fog.
- The geospatial data processing and geospatial query resolution are performed inside the fog device. When a mobile user queries for geospatial information from his/her current location, the fog device resolves the geospatial query and sends the result to the requesting device without loading the cloud servers.
- The communication and computation costs in terms of delay and power consumption for the proposed architecture are measured in terms of empirical study and experimental analysis to show that the proposed framework is delay-sensitive and energy-efficient than the existing model for geospatial query resolution.

The rest of the article is organized as follows. Section 2 discusses on the related existing strategies. Section 3 discusses on the geospatial data analysis. Section 4 elaborates the proposed fog based architecture for geospatial query resolution. Section 5 presents the performance analysis. Section 6 concludes the paper with future research direction in the context of the Spatio-Fog architecture. The abbreviations of the terms and notations of the symbols used in this paper are listed in Table 1 and Table 2 respectively.

2. Related work

Geospatial data analysis is an emerging research challenge due to its large volume and computational intensity of processing the geospatial data. Access to geospatial data is required for various applications such as mobility prediction, regional weather forecasting, reservoir water level forecasting [17], land-use/land-cover (LULC) distribution [18], wild fire spread estimation [19], providing assistance in natural disaster using volunteered geographic information [20,21], urban growth prediction [22]. Mobility prediction needs access to geospatial big data which can be captured using GPS of the users mobile device [23,24]. Regional weather forecasting also needs access to the weather data e.g. temperature, humidity, along with the geospatial data of the region for which the weather data has been collected and accessed [25–27].

Various researches have focused on the challenges of geospatial applications [1,11–14] and the use of cloud computing to address them. However, the use of only cloud-based system may not be efficient in terms of accessing the data at low latency, low energy consumption. Moreover, the load on the cloud data centre is also very high. Use of fog computing for distribution of the voluminous data has not been explored much. In this section, we have discussed on the geospatial information and the query resolution along with the existing approaches on geospatial cloud computing and fog computing.

Geospatial information refers to the data with respect to a geographical location in terms of geographic coordinates, collected using Geographic Information System (GIS) [28]. Remote sensing images are recognized as important geospatial data. Geospatial data are of two types: Raster data and Vector data. Indexing, scoring, and ranking of remote sensing images have been discussed

Table 1
Abbreviations and it's meaning.

Abbreviation	Meaning
CSW	Catalog Service for the Web
EPSG	European Petroleum Survey Group
GB	Gigabyte
GCP	Google Cloud Platform
GHz	Gigahertz
GIS	Geographic Information System
GPS	Global Positioning System
GQ	Geospatial Query
GQ1	Geospatial Query 1
GQ2	Geospatial Query 2
HDD	Hard Disk Drive
ICMP	Internet Control Message Protocol
ID	Identification Number
LULC	Land-Use Land-Cover
MATLAB	Matrix Laboratory
Mbps	Megabits per Second
MB	Megabyte
MGL	Meghamala
MSM	Mobile Station Modem
OGC	Open Geospatial Consortium
QGIS	Quantum Geographical Information System
QoS	Quality of Service
RAM	Random Access Memory
sec	Second
TB	Terabyte
TCP	Transmission Control Protocol
UDP	User Datagram Protocol
vCPU	Virtual Central Processing Unit
VM	Virtual Machine
XBR+	External Balanced Regular Plus
W	Watt
WMS	Web Map Service
WCS	Web Coverage Service
WFS	Web Feature Service
WPS	Web Processing Service

Table 2
Notations used in the paper.

Notation	Description
a	Sub-area
z_a	Zone under sub-area a
r_{z_a}	Region under zone z_a
$S_{r_{z_a}}$	Geospatial data of region r_{z_a}
f	Fog device of one region
g	Fog device of another region
M	Mobile device
N	Number of Geospatial Query
GQ	Geospatial Query
T_{GQ}	Response time of a geospatial query
RT_N	Average system response time

in [29]. Geospatial dataset is multidimensional [1]. Various researches have been performed on geospatial big data analysis [11–15]. The use of cloud computing to deal with the large volume of geospatial data has been illustrated in [11,12]. The use of fog computing for the big geospatial data analysis has been discussed in [15].

A method on location based service provisioning has been discussed in [30]. This method focuses on the creation, store, manipulation and display of location based data. A unique location identifier has been used which considers latitude, longitude along with a time stamp or sequence number. This identifier has been used as a key and for geospatial data analysis to give geographic context. Distributed location-sensitive hashing has been used for privacy-preserving and scalable service recommendation in [31]. For location based services, scalable data management method has been discussed in [32,33]. Dual data cache for managing spatial and temporal locality has been proposed in [34]. For improving the locality of references, array layout has been suitable selected in [35].

Query resolution for spatiotemporal database has been discussed in [36]. Query processing for the geospatial database has been

discussed in [37]. Usually Quadtree [38] and R-tree [39] are popular for geospatial query processing [37]. However, R^+-tree [40], R^*-tree [41] and PMR Quadtree [42] are also used for geospatial query processing [37]. The authors in [37] have used XBR^+-tree for geospatial query processing, which is a dynamic, disk-resident and balanced Quadtree-based index structure. Multiple query optimization for relational database has been highlighted in [43]. For geospatial query evaluation, a method has been discussed in [1], where multi-dimensional data set has been considered. A predictive and exploratory analysis over high volume multi-dimensional data set in a distributed environment has been performed in [44]. For land cover classification a geospatial web service has been designed in [45]. For disaster-oriented regions, a geospatial service-oriented framework has been proposed in [21]. For storage and analysis of high volume geospatial data, cloud data centres have been used [11,12,46]. For high-resolution geospatial interpolation, cloud computing has been used in [47]. In the Geospatial Cloud framework, the application tier is used for geospatial services, such as Web Map Service (WMS) [48], Web Coverage Service (WCS), Web Feature Service (WFS) [49], Catalog Service for the Web (CSW) and Web Processing Service (WPS) [50]. Though cloud computing has been largely used for geospatial data analysis, use of long distant cloud servers for geospatial query resolution increases delay and energy. Use of fog computing for geospatial data analysis has been highlighted in [15,16,51]. Geospatial query processing in edge devices has been discussed in [16]. However, geospatial data is of large amount. Hence, the processing of voluminous data requires high-end processing, which the edge or fog devices may not be able to deal with. However, if this huge volume of data is stored in a distributed manner in fog devices and cloud servers, so that the high-end processing will be executed by the cloud, there a trade-off will be maintained between communication and computation costs.

In fog based frameworks, the fog devices are used for preliminary processing of data. Fog devices usually offload the computational tasks, rather than permanently storing massive volume of data. In the existing methods, the fog devices perform preliminary processing before forwarding the data to the cloud, for example, overlay analysis or compression on the geospatial data is performed inside the fog devices, and then the data is forwarded to the cloud [15,16,51]. However, from the perspective of geospatial data analysis, the use of fog computing is promising due to the data intensity as well as the computational intensity of the geospatial data. Hence, there should be a framework which will deal with decision making regarding the distribution of geospatial data, load balancing, and geospatial query resolution with energy-efficiency and delay optimization. The users will generate queries related to relational regional geospatial data, and the system has to respond to it. The objective of using fog computing in our work is to process a user-generated regional geospatial query at an optimum delay and power consumption of the client device. Towards the solution of this, we have proposed a cloud-fog based distributed framework in this paper. For geospatial query processing in the geospatial relational database, parse tree [52] is used in our work. The communication and computation costs in terms of delay and power consumption of the user device in our framework are measured and compared with the existing systems on geospatial data analysis and services in cloud and fog environment.

Comparison with existing schemes:

As our work is based on geospatial query processing in cloud-fog framework, we have compared the proposed model with the existing strategies on geospatial data analysis and services in cloud and fog environment. Table 3 compares the contributions of the proposed work with the existing schemes on geospatial data analysis and services. In the existing cloud-based systems, the geospatial data storage and analysis take place inside the cloud servers [1,11,12,46]. In the existing fog based system, the fog devices are used to perform only the preliminary processing such as compression, on the geospatial data [15]. In our system, the data to be contained by the fog devices are categorized regionally, with an objective to optimize the delay and power consumption of the client device during geospatial query resolution, as well as to reduce the load on cloud data centres. This is the uniqueness of the proposed model with respect to the existing frameworks [1,11,12,15,46]. From Table 3, it is also observed that in our system, we have not only analysed

Table 3
Comparison with existing schemes on geospatial query resolution.

Feature	Existing strategies					Proposed strategy
	Fast, ad hoc query evaluations over multi-dimensional geospatial datasets [1]	Geospatial data analysis using cloud [11]	Spatial cloud computing [12]	FogGIS [15]	Geospatial services using cloud [46]	Proposed Spatio-Fog
Contribution	A novel indexing to model the correlations among heterogeneous datasets is proposed for approximate query resolution.	The use of cloud computing for big geospatial data storage and analysis has been discussed.	The use of cloud computing for spatial data search, access and utilization has been discussed.	Fog computing is used for preliminary processing on geospatial data.	An architecture for geospatial data processing and data acquisition services has been proposed.	Region-specific fog computing paradigm for geospatial query resolution by data analysis has been proposed.
Fog computing is used	X	X	X	✓	X	✓
Delay is calculated	X	X	X	X	X	✓
Power is calculated	X	X	X	X	X	✓
Response time is calculated	✓	X	✓	X	X	✓

the data for geospatial query resolution but also calculated the delay, power consumption, and system response time.

It is observed that most of the existing approaches on geospatial data analysis and geospatial query resolution are based on cloud computing, and the fog computing has been used only for partial computation on the geospatial data before forwarding to the cloud. However, none of them has focused on the use of fog nodes for distributing the storage and processing of the high volume geospatial data in order to reduce the load on the cloud as well as for better utilization of the fog devices' resources. We have focused on this, and as a solution towards this challenge, we have proposed a fog-cloud based collaborative framework in Section 4. But before that, we have discussed on the geospatial data analysis and different types of services required for geospatial query processing in Section 3. The use of the proposed architecture for resolving these types of queries has been discussed in Section 4.

3. Geospatial data analysis

Geospatial data analysis [53] is a collection of techniques and models which are used to determine the spatial relationships among geospatial data. In cartographic modelling, a map is represented by processed geospatial data. In mathematical modelling, results depend upon the spatial interaction between the objects of the model. Geospatial data analysis improves the development and application of statistical techniques. Differences between relational database query and geospatial query [54] are there are no fixed set of operators for geospatial query evaluation, and databases deal with a large volume of geospatial data, computationally expensive due to spatial predicates.

The spatial data analysis comprises of two major operations: *single layer operations*, and *multi-layer operations* or *topological overlay*. Former operation is associated with individual feature dataset. The variations of geospatial data operate on a single data layer. The geospatial query, like buffer creation over a point or layer, filtering a feature from various features of a single layer, falls into the single layer operation category. Whereas, multi-layer operations are associated with several layers spatial data manipulation. Two or more spatial layers combine together and generate new layer through topological overlays. More information are generated from the combined layer which are not present in the individual layer. We have illustrated these two operations below with two examples GQ1, GQ2. Moreover, Open Geospatial Consortium (OGC) compliant geospatial services help to process and publish the results of the geospatial queries [55]. They offer the following capabilities:

- Web Feature Service(WFS) helps to extract the featured data from the large volume of geospatial data.
- Web Processing Service(WPS) helps to process like buffer, intersection operations over the geospatial data.
- Web Map Service(WMS) helps to generate a map from the geospatial data according to its coordinates.
- Web Coverage Service(WCS) helps to access multi-dimensional geospatial data through the Internet for a precise geospatial query. It can be considered as a feature service for raster data. It handles complex geospatial query and returns query result with detailed information of the specific coverage area. This feature of WCS influences complex geospatial modelling and analysis.
- Catalog Service for Web(CSW) helps to maintain a catalog for geospatial records in xml format over the Internet. It provides an interface for both service providers and service consumers. Service providers publish their services here and service consumers discover, browse, and query metadata.

Two examples of the geospatial query are provided as follows.

Geospatial Query 1 (GQ1): List the 'one_Way' roads of Bengaluru, India.

There are varied types of road segments in the underlying road network dataset such as 'Metal', 'Non-Metal', 'One way', 'Two way'. Any location-based service-provider may require to extract a specific type of roads in a particular spatial region. For instance, through GQ1, a user wants to know about one way roads in Bengaluru city. This is an example of single layer operated geospatial query. In this query, only road layer of the Bangaluru city has been considered. 'One_way' feature has been extracted from the road layer. For GQ1 the query command takes the following form.

```
SELECT B.road_name
FROM Bengaluru_Road B
WHERE B.road_type = 'One_Way';
```

Query parsing tree for GQ1 is presented in Fig. 1. A 'filter' spatial operation is done over the road network shapefile of Bengaluru. For resolving GQ1, the following geospatial services are required.

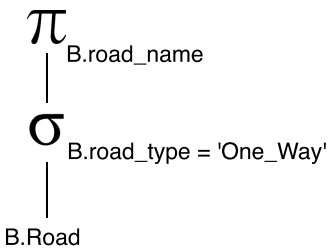

Fig. 1. Query Parsing Tree of GQ1.

- WFS *getFeature* service is used for filtering the 'One_Way' roads from Bengaluru road network.
- WMS *GetMap* service is used to display the 'One_Way' road map of Bengaluru.

Geospatial Query 2 (GQ2): List of areas which are present within 10 km of each hospital in Bankura district.

There are several thematic layers namely LULC, Road Network, River, Facilities (Hospitals, ATMs, Schools, Universities, Oil Pumps, Post Office, Markets, Rail, and Bus Stations) available in a geospatial dataset. GQ2 is an example of multi-layer operation query. In this query, two layers (LULC and Hospital) are considered. While two layers are individual, the user is not able to determine the distance or shortest path between a point of an area and a hospital. But after the overlaying of these two layers, user can easily determine the shortest path or distance of a specific point and the hospital. Through GQ2, the user wants to know the name of areas of Bankura district covered by each hospital. The distance of the areas should be within 10 km from hospitals. For this geospatial query, the following geospatial query command is generated.

```
SELECT B.area_name
FROM Bankura B
WHERE Overlap (area.shape, Buffer (hospital.shape, 10))=1
GROUP BY hospital_name
ORDER BY area desc;
```

Here,'.shape' stands for shapefile. A shapefile[1] is a simple, nontopological format for storing the geometric location and attribute information of geographic features. Geographic features in a shapefile can be represented by points, lines, or polygons (areas). The workspace containing shapefiles may also contain database tables, which can store additional attributes that can be joined to a shapefile's features.

Query parsing tree for GQ2 is presented in Fig. 2. In GQ2, first, select the hospitals of Bankura and create a 10-kilometer buffer of each Hospital. Now, make an 'overlap' spatial operation between buffered hospital shapefile and area shapefile of Bankura. From there, we obtain area names of Bankura which are within 10 km of hospitals. For resolving GQ2, the following geospatial services are required.

- WPS *BufferFeatureCollection* service is used for the creation of 10 km buffer of the hospitals.
- WPS *IntersectionFeatureCollection* service is used for the overlap operation.
- WMS *GetMap* service is used to display the final area map of Bankura.

In this section, we have discussed on the geospatial data analysis and geospatial query processing. In the next section, we have proposed the Spatio-Fog architecture for geospatial data storage and geospatial query processing. These types of geospatial queries will be processed by the fog devices or cloud based on the geospatial area for which the geospatial query has been made.

4. Proposed Spatio-Fog architecture

The fog based system architecture for geospatial query processing is presented in Fig. 3. The total geographical area is divided into sub-areas, e.g., countries. Each geographical sub-area has coverage which is decomposed into several zones, e.g., in a country, there are a number of states. Each zone is decomposed into several regions, e.g., each state contains a number of districts. In our system, we are considering a hierarchical architecture (see Fig. 4), where level 1 is a sub-area, e.g., country, which is then decomposed into several zones, e.g., states, in level 2. Each zone, e.g., the state is then decomposed into several regions, e.g., districts, in level 3.

Let a is a sub-area and z_a is a zone under sub-area a, then $z_a \in a$. Let r_{z_a} is a region under zone z_a, then $r_{z_a} \in z_a$. The geospatial data of region r_{z_a} is denoted by $S_{r_{z_a}}$. If f is a fog device located in r_{z_a}, then f will contain $S_{r_{z_a}}$.

In our system, fog devices located in each region contains the geospatial information of that region. Usually, switch, gateway, an indoor base station with storage and computation ability [56] are used as fog devices. In our system, the intermediate devices between the end node and cloud servers, possessing high storage and computation ability are considered as fog devices, because they will store the geospatial data of the region as well as process the data for geospatial query resolution. When a user generates one or multiple queries to retrieve information related to a region where he/she is currently present, then instead of accessing the cloud servers the fog device accesses its geospatial data and responds accordingly, e.g., the geospatial query resolution is performed by the fog device instead of the cloud servers. The advantages of this framework are:

- As the fog device stores the geospatial data of the region instead of the cloud data centres, the propagation and communication delays are reduced, which in turn helps to reduce the total delay.
- As the fog device is containing the region-specific geospatial data instead of the cloud, the load on cloud data centre will be reduced. Moreover, an extra layer for regional data privacy will be maintained. However, the geospatial data of the regions to be frequently accessed, are stored in the cloud, which will not compromise with the delay over remote cloud if a geospatial query is generated regarding such regions other than the current region.

[1] http://desktop.arcgis.com/en/arcmap/10.3/manage-data/shapefiles/what-is-a-shapefile.htm .

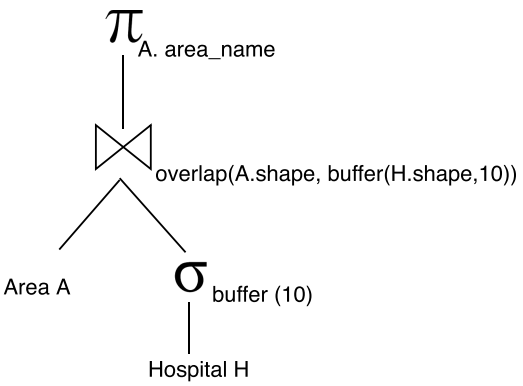

Fig. 2. Query Parsing Tree of GQ2.

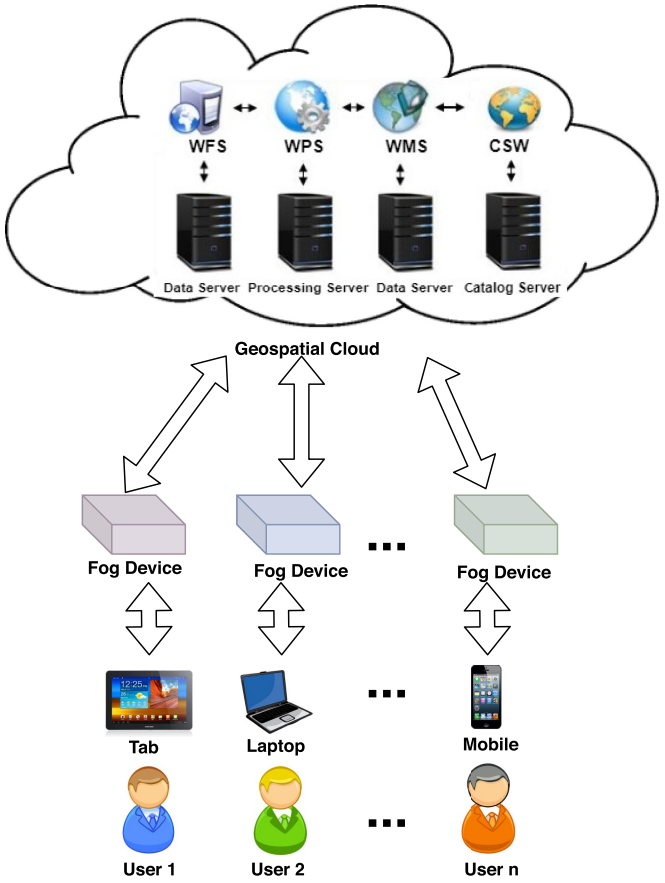

Fig. 3. Architecture of fog based system for geospatial query resolution.

4.1. Geospatial query generation from mobile user

A mobile device located in a region is connected with a fog device. The mobile device generates a geospatial query and sends to the fog device.

Let a mobile device M currently located in region r_{z_a} is connected with a fog device f of the region and the geospatial data stored by f is $S_{r_{z_a}}$. M generates a geospatial query GQ and sends to f.

4.2. Geospatial query resolution by fog device

The fog device of a region contains the geospatial data of that region. The geospatial data storage and processing are performed by

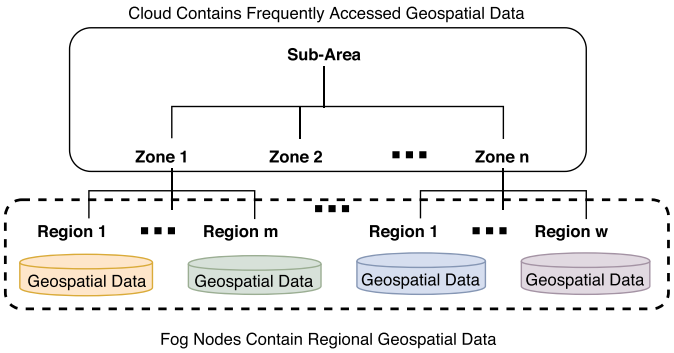

Fig. 4. Hierarchy of the proposed Spatio-Fog framework.

the fog device. When the fog device receives a geospatial query from the mobile device, first it verifies whether the geospatial query is related to the current region where both the mobile device and fog device are located. If the geospatial query is related to the current region, the fog device analyses its geospatial data and responds accordingly. Otherwise, the fog device forwards the geospatial query to the cloud servers. If the geospatial query *GQ* received by the fog device *f* is related to $S_{r_{za}}$, *f* responds after analysing the data. Otherwise, *f* forwards *GQ* to the cloud.

4.3. Cloud servers

The fog devices are connected with the cloud servers. The cloud servers contain the geospatial data of the regions for which the users frequently generate queries. When the cloud receives a geospatial query, it checks whether it has the intended geospatial data to solve the geospatial query. If so, it analyses the data and sends the result. Otherwise, the cloud servers identify the region related to the geospatial data and forward the request to the fog device of that region.

If the geospatial query *GQ* received by the cloud is related to the geospatial data which it contains, it resolves the geospatial query and sends the result to the fog device, which further forwards it to the mobile device. Otherwise, the cloud forwards *GQ* to a fog device *g* which contains geospatial data related to *GQ*. Here, fog devices *f* and *g* belong to two different regions. To find out the sub-area, zone, and region of a geospatial query and then finding out the respective fog device, area registry is maintained inside the cloud servers. The registry is containing the sub-area IDs under the area and the zone IDs under the sub-areas. The zone IDs are referring to the region IDs of the respective zones. The region IDs are referring to the fog device IDs containing the geospatial data of the respective regions. The registry for area *a* is presented in Fig. 5. There are *as* sub-areas present under the area *a*. Each sub-area has several zones, which are pointing to the regions under them. The regions are pointing to the fog devices containing the geospatial data of those regions. By accessing the area registry, the cloud can find out the fog device containing the geospatial data related to a particular region.

4.4. Case study

In Section 3, we have discussed on the queries and the services for processing those queries. Now, the mobile user when generates such queries, then the proposed framework resolves them based on the location for which the geospatial query has been made. For

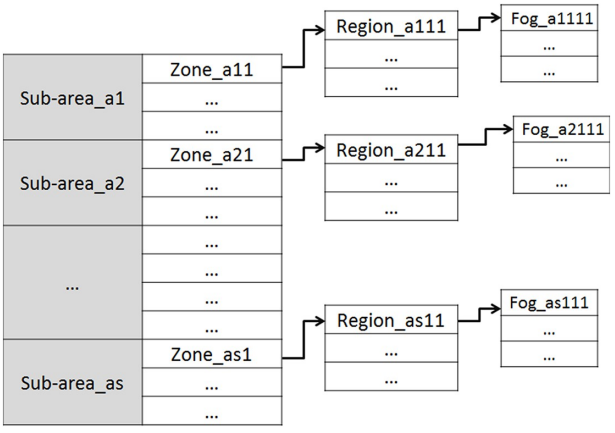

Fig. 5. Area registry maintained inside the cloud.

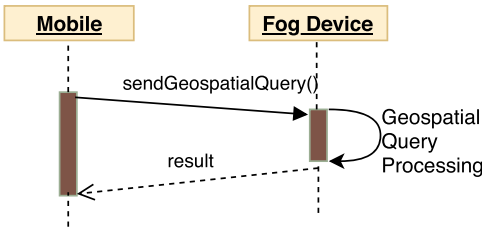

Fig. 6. Case 1- Sequence diagram of geospatial query resolution by fog device of current region.

resolving these queries, the WPS, WFS, and WMS services are used according to the type of operation required. Based on the location for which the geospatial query is made, the following three cases come into the scenario:

- Geospatial query resolution by fog device of the current region
- Geospatial query resolution using cloud servers
- Geospatial query resolution using fog device of another region

These three cases are discussed as follows.

4.4.1. Case 1: Geospatial query resolution by fog device of current region

In this case, the mobile device sends a geospatial query to the connected fog device of its region. The geospatial query is related to the geospatial data of the current region. Thus the fog device analyses its geospatial data and sends the result to the mobile device. The corresponding sequence diagram is presented in Fig. 6.

4.4.2. Case 2: Geospatial query resolution using cloud servers

In this case, the mobile device sends a geospatial query to the connected fog device of its region. The geospatial query is related to the geospatial data of the other region, whose data the cloud servers contain. The fog device forwards the geospatial query to the cloud servers. The cloud servers analyse the intended geospatial data to resolve the geospatial query and send the result to the fog device. The fog device then sends the result to the mobile device. The corresponding sequence diagram is presented in Fig. 7.

4.4.3. Case 3: Geospatial query resolution using fog device of another region

In this case, the mobile device sends a geospatial query to the connected fog device of its region. The geospatial query is related to the geospatial data of the other region, whose data is not available inside the cloud servers. The fog device forwards the geospatial query to the cloud servers. The cloud accesses the area registry and finds out the fog device containing the corresponding geospatial data. After that, the cloud forwards the request to that fog device along with the ID of the requesting fog device and the geospatial query ID. The corresponding fog device analyses its geospatial data to resolve the geospatial query and sends the result to the requesting fog device directly. The fog device then sends the result to the mobile device. The corresponding sequence diagram is presented in Fig. 8.

4.5. Computation offloading

In the previous sections, we have discussed on geospatial query resolution. When a mobile device of a region raises a geospatial query, then using the fog device or the cloud the query is resolved. However, if a mobile device requests for offloading a computation, then also the fog device or cloud will be used. The computation offloading method has been stated in Algorithm 1.

When a mobile device has to offload a computation, it will send a request to the nearby fog device for offloading the computation.

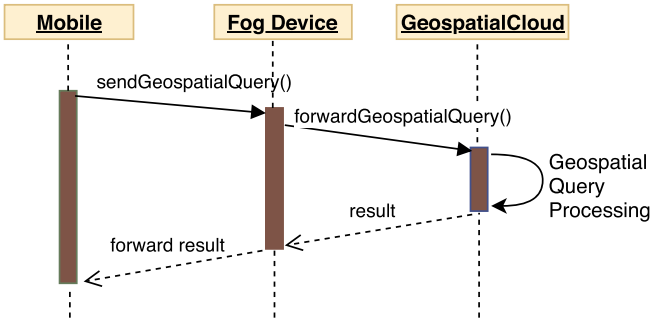

Fig. 7. Case 2- Sequence diagram of geospatial query resolution using cloud servers.

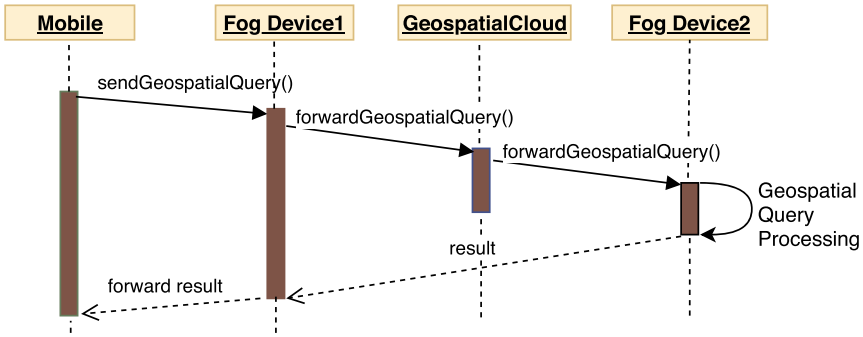

Fig. 8. Case 3- Sequence diagram of geospatial query resolution using fog device of other region.

If the fog device is able to execute the computation based on its processing capability, current load, memory, it will send the result to the device after execution. Otherwise, it will forward the request to the cloud along with the ID of the requesting mobile device. The cloud sends the result to the mobile device after execution.

4.6. Delay and power consumption in geospatial query resolution

The parameters used in delay and power calculation are defined in Table 4.

The sum of propagation delay [5,10], communication delay [5,10], processing delay [5,10,24] and queuing delay [5] is the total delay while resolving geospatial query generated from a mobile device. The probabilities of occurrences of case 1 i.e. query resolution by fog device of present region, case 2 i.e. query resolution by cloud and case 3 i.e. query resolution by fog device of another region are denoted by p_1, p_2 and p_3 respectively. Hence, the values of p_1, p_2 and p_3 are calculated as (N_{case1}/T_{req}), (N_{case2}/T_{req}) and (N_{case3}/T_{req}) respectively, where N_{case1}, N_{case2} and N_{case3} are the number of requests arrived for case 1, case 2 and case 3 respectively, and T_{req} is the total number of requests. Therefore the values of p_1, p_2 and p_3 are considered as less 1 i.e. $p_1 \leq 1$, $p_2 \leq 1$, $p_3 \leq 1$ and $p_1 + p_2 + p_3 = 1$.

In the first case, the propagation delay is (d_{prop1}/S_p) as the fog device of current region resolves the geospatial query. However, in the second case, the cloud resolves the geospatial query. Therefore, the propagation delay is (d_{prop2}/S_p). In the third case, the fog device of another region resolves the geospatial query. Hence, the propagation delay is (d_{prop3}/S_p). If all the three cases are considered with their probability of occurrences (p_1, p_2, p_3), then the propagation delay is given as,

$$D_p = (p_1{*}d_{prop1} + p_2{*}d_{prop2} + p_3{*}d_{prop3})/S_p \tag{1}$$

where $p_1 \leq 1$, $p_2 \leq 1$, $p_3 \leq 1$ and $p_1 + p_2 + p_3 = 1$.

In the first case, the fog device of current region resolves the geospatial query. Hence, for the first case, the uplink and downlink communication delays between mobile device and fog device are considered. The communication delay from mobile device and fog device is given as,

$$D_{up_1} = (D_{mf}/Up_{mf}){*}(1 + fu_{mf}) \tag{2}$$

The communication delay from fog device to mobile device is given as,

$$D_{dw_1} = (D_{fm}/Dw_{mf}){*}(1 + fd_{mf}) \tag{3}$$

In the second case, the cloud resolves the geospatial query. Hence, for the second case, the uplink and downlink communication delays between fog device and cloud are taken into account along with the delays in the first case. The communication delay from fog device to cloud is given as,

$$D_{up_2} = (D_{fc}/Up_{fc}){*}(1 + fu_{fc}) \tag{4}$$

The communication delay from cloud to fog device is given as,

$$D_{dw_2} = (D_{cf}/Dw_{fc}){*}(1 + fd_{fc}) \tag{5}$$

In the third case, the fog device of another region resolves the geospatial query. Hence, for the third case, the uplink and downlink communication delays between two fog devices are taken into account along with the delays in the first case. As the fog device responds to the geospatial query using fog device of another region, the communication delay from requesting to serving fog device is given as,

$$D_{up_3} = ((D_{fc}/Up_{fc}){*}(1 + fu_{fc})) + ((D_{cfq}/Dw_{fc}){*}(1 + fd_{fc})) \tag{6}$$

In this case, the geospatial query is sent from the requesting fog device to the serving fog device through the cloud. However, after resolving the geospatial query the serving fog device directly delivers the result to the requesting fog device. Therefore, the

Input : Computation to be offloaded
Output: Result after execution
start;
the mobile device sends a request to the nearby fog device for offloading the computation;
if *the fog device is able to execute the computation* **then**
    the fog device executes the computation;
    the fog device sends the result to the mobile device;
else
    the fog device forwards the request to the cloud along with the mobile device ID;
    the cloud executes the computation;
    the cloud sends the result to the mobile device;
end
end;

Algorithm 1. Computation offloading.

Table 4
Notations used in delay and power calculation.

Parameter	Definition
d_{prop1}	Distance to be covered during propagation while fog device of current region resolves geospatial query
d_{prop2}	Distance to be covered during propagation while cloud resolves geospatial query
d_{prop3}	Distance to be covered during propagation while fog device of other region resolves geospatial query
S_p	Propagation speed
D_f	Data amount processed for resolving geospatial query
S_f	Data processing speed of fog device
S_{cl}	Data processing speed of cloud
D_{mf}	Data amount transmission from mobile device to fog device while sending geospatial query
D_{fm}	Data amount transmission from fog device to mobile device while sending result
D_{fc}	Data amount transmission from fog device to cloud while sending geospatial query
D_{cfq}	Data amount transmission from cloud to fog device while sending geospatial query
D_{cf}	Data amount transmission from cloud to fog device while sending result
D_{ff}	Data amount transmission from serving fog device to requesting fog device while sending result
Up_{mf}	Data transmission rate from mobile device to fog device
Dw_{mf}	Data transmission rate from fog device to mobile device
Up_{fc}	Data transmission rate from fog device to cloud
Dw_{fc}	Data transmission rate from cloud to fog device
Dw_{ff}	Data transmission rate from serving to requesting fog device
fu_{mf}	Link failure rate from mobile device to fog device
fd_{mf}	Link failure rate from fog device to mobile device
fu_{fc}	Link failure rate from fog device to cloud
fd_{fc}	Link failure rate from cloud to fog device
fd_{ff}	Link failure rate from serving fog device to requesting fog device
D_w	Queuing delay
P_t	Power consumption of mobile device for data transmission per unit time
P_r	Power consumption of mobile device for data reception per unit time
P_m	Power consumption of mobile device in idle mode per unit time
N_{case1}	Number of requests arrived for case 1
N_{case2}	Number of requests arrived for case 2
N_{case3}	Number of requests arrived for case 3
T_{req}	Total number of requests arrived
p_1	Probability of case 1
p_2	Probability of case 2
p_3	Probability of case 3

communication delay from serving to requesting fog device is given as,

$$D_{dw_3} = (D_{ff}/Dw_{ff})*(1 + fd_{ff}) \tag{7}$$

After considering the probabilities of occurrences (p_1, p_2, p_3) of all the three cases, the uplink communication delay is therefore given as,

$$\begin{aligned} D_{up} &= p_1*D_{up_1} + p_2*(D_{up_1} + D_{up_2}) + p_3*(D_{up_1} + D_{up_3}) \\ &= D_{up_1} + p_2*D_{up_2} + p_3*D_{up_3} \end{aligned} \tag{8}$$

where $p_1 \leq 1$, $p_2 \leq 1$, $p_3 \leq 1$ and $p_1 + p_2 + p_3 = 1$.

Accordingly, The downlink communication delay is given as,

$$\begin{aligned} D_{dw} &= p_1*D_{dw_1} + p_2*(D_{dw_1} + D_{dw_2}) + p_3*(D_{dw_1} + D_{dw_3}) \\ &= D_{dw_1} + p_2*D_{dw_2} + p_3*D_{dw_3} \end{aligned} \tag{9}$$

where $p_1 \leq 1$, $p_2 \leq 1$, $p_3 \leq 1$ and $p_1 + p_2 + p_3 = 1$.

The communication delay as a sum of the uplink and downlink communication delays, is therefore given as,

$$D_{com} = D_{up} + D_{dw} \tag{10}$$

In the first and third cases, fog device processes the geospatial query. The data processing delay of fog device is given as,

$$D_{procf} = D_f/S_f \tag{11}$$

In the second case, cloud processes the geospatial query. The data processing delay of cloud is given as,

$$D_{proccl} = D_f/S_{cl} \tag{12}$$

After considering the probabilities of occurrences of all the three cases, the processing delay is given as,

$$D_{proc} = p_1*D_{procf} + p_2*D_{proccl} + p_3*D_{procf} \tag{13}$$

where p_1 is the probability of occurrence of first case where fog device of current region processes the geospatial query, p_2 is the probability of occurrence of second case where cloud processes the geospatial query, p_3 is the probability of occurrence of third case where fog device of other region processes the geospatial query, $p_1 \leq 1$, $p_2 \leq 1$, $p_3 \leq 1$ and $p_1 + p_2 + p_3 = 1$.

The total delay as the sum of the propagation, communication, processing and queuing delays is given as,

$$D_{tot} = D_p + D_{com} + D_{proc} + D_w \tag{14}$$

During propagation the mobile device's idle mode is considered [5]. Hence, the power consumption of mobile device during propagation is given as,

$$P_p = P_m{*}D_p \tag{15}$$

where P_m is the power consumption of mobile device per unit time if it is in idle mode.

During uplink communication, when data transmission takes place from the mobile device, the device's power consumption per unit time in data transmission mode is considered [5]. Otherwise, the device's power consumption per unit time in idle mode is considered [5]. Hence, the power consumption of mobile device during uplink communication is given as,

$$P_{mu} = P_t{*}D_{up_1} + p_2{*}P_m{*}D_{up_2} + p_3{*}P_m{*}D_{up_3} \tag{16}$$

where P_m is the power consumption of mobile device per unit time if it is in idle mode and P_t is the power consumption of mobile device per unit time if it is in data transmission mode.

During downlink communication, when data reception takes place by the mobile device, the device's power consumption per unit time in data reception mode is considered [5]. Otherwise, the device's power consumption per unit time in idle mode is considered [5]. Hence, the power consumption of mobile device during downlink communication is given as,

$$P_{md} = P_r{*}D_{dw_1} + p_2{*}P_m{*}D_{dw_2} + p_3{*}P_m{*}D_{dw_3} \tag{17}$$

where $p_2 \leq 1$, $p_3 \leq 1$ and $p_1 + p_2 + p_3 = 1$, P_m is the power consumption of mobile device per unit time if it is in idle mode and P_r is the power consumption of mobile device per unit time if it is in data reception mode.

Therefore, the power consumption of mobile device during communication is given as,

$$P_{mcom} = P_{mu} + P_{md} \tag{18}$$

The power consumption of mobile device during data processing by fog device or cloud, is given as [5],

$$P_{proc} = P_m{*}D_{proc} \tag{19}$$

where P_m is the power consumption of mobile device per unit time if it is in idle mode.

The power consumption of mobile device during queuing period is given as [5],

$$P_{mqu} = P_m{*}D_w \tag{20}$$

where P_m is the power consumption of mobile device per unit time if it is in idle mode.

Therefore the total power consumption of the mobile device as a sum of the power consumption during propagation, communication, geospatial query processing (inside the fog device/cloud) and queuing periods, is given as,

$$P_{tot} = P_p + P_{mcom} + P_{proc} + P_{mqu} \tag{21}$$

In the next section, using this mathematical model we have determined the delay in resolving a mobile user-generated geospatial query using the proposed Spatio-Fog framework, and the power consumption of the user device during geospatial query resolution period, and compared with the existing cloud-based geospatial query resolution framework. After that we have validated the results of the empirical study with the experimental results obtained by carrying out different types of geospatial query resolution using the proposed framework in the laboratory.

5. Performance evaluation

In this section, we have performed empirical study using MATLAB and experimental analysis using OpenStack Cloud and Google Cloud Platform. We have compared our framework with the existing query processing system in terms of delay and power consumption of the mobile device. We have also performed a comparison between the results obtained from the empirical study and experimental analysis in this section.

5.1. Empirical study

In this section, the delay and power consumption during geospatial query resolution using proposed fog based system and existing cloud-based system are compared [11,46]. MATLAB 2015 is used for empirical study. For empirical study, the total amount of geospatial data is considered 1 to 10 TB. It is assumed that $p_1 < 1$, $p_2 < 1$, $p_3 < 1$ and $p_1 + p_2 + p_3 = 1$. For empirical study the data amount transmitted in uplink and downlink is assumed 1.25 to 2.50 MB and 15 to 27.5 MB respectively, the uplink and downlink data transmission rate are assumed 5 to 10 Mbps and 10 to 30 Mbps respectively, and the power consumption of mobile device per unit time in transmit, receive and idle modes are assumed 0.01 to 0.1 W, 0.005 to 0.05 W, and 0.005 W respectively. Fig. 9 shows the

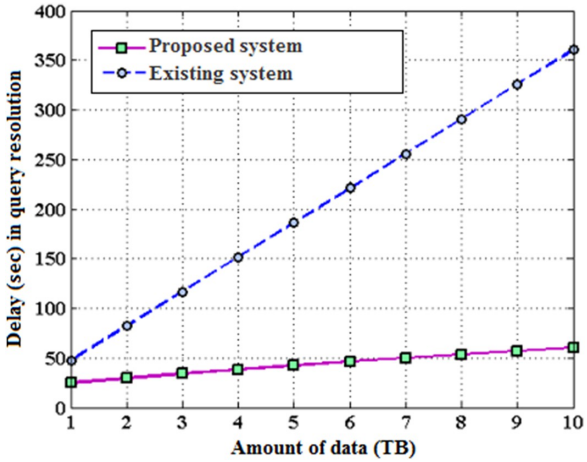

Fig. 9. Delay in geospatial query resolution using proposed Spatio-Fog and existing system.

delay in geospatial query resolution using proposed fog based system Spatio-Fog, calculated using Eq. (14) and using existing cloud-based system [11,46]. The simulation results show that Spatio-Fog reduces the delay by approximately 47–83% than the existing system.

Fig. 10 shows the power consumption of mobile device during geospatial query resolution using proposed fog based system Spatio-Fog, calculated using Eq. (21) and using existing cloud-based system [11,46]. The simulation results show that Spatio-Fog reduces the power by approximately 43–47% than the existing system.

If the geospatial query size and the size of the result to be sent to the device are large, then communication delay will be high. The processing delay depends on the volume of data has to be processed to resolve the geospatial query. Hence, the reduction in delay and power consumption while using the proposed architecture will differ for large, medium, or small data size.

5.2. Analysis of the experimental results

The configurations of the devices used in the experimental analysis are presented in Table 5. Two mobile phones are used, which generate geospatial queries. The first mobile phone has 3 GB RAM, 32 GB HDD, and Qualcomm MSM8940 Octa Core processor. The second mobile phone has 2 GB RAM, 8 GB HDD and 64-bit 1.2 GHz Qualcomm Snapdragon 410 Quad Core processor. The operating system of both the mobile phones is Android 6.0.1. Two laptops with 4 GB RAM, 250 GB HDD, and Intel Core i5 processor have been used as fog devices. The operating system of these two devices is Windows 7 Professional and Ubuntu. For local cloud environment, we have used OpenStack Cloud Platform of IIT Kharagpur, referred as Meghamala (MGL), where we have taken a regular VM (2 vCPU, 4 GB RAM). For remote cloud environment, we have used Google Cloud Platform (GCP), where we have created one Virtual Machine (VM) instance (zone: asia-south1-c). The machine type is n1-standard-1 (1 vCPU, 3.75 GB RAM). The protocols used are TCP, UDP, ICMP. The mobile devices are connected through Wi-Fi Access points. The data transmission rate is 10–50 Mbps.

Mobile phone 1 and mobile phone 2 are connected with fog device 2 and fog device 1 respectively. In this work, we have used the

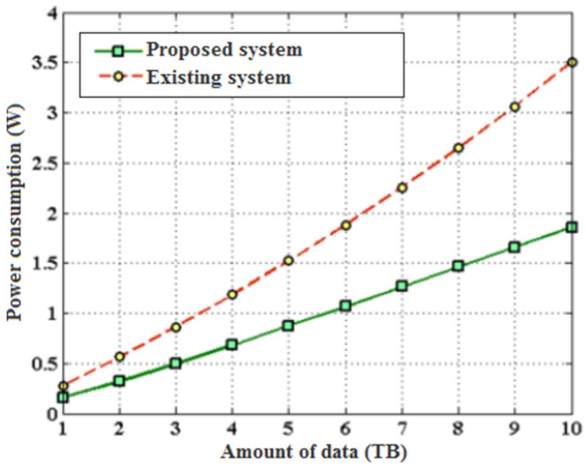

Fig. 10. Power consumption by mobile device during geospatial query resolution using proposed Spatio-Fog and existing system.

Table 5
Configurations of devices used in experiment.

Device	RAM	HDD	Processor	Operating system
Mobile phone 1	3 GB	32 GB	Qualcomm MSM8940 Octa Core	Android 6.0.1
Mobile phone 2	2 GB	8 GB	64-bit 1.2 GHz Qualcomm Snapdragon 410 Quad Core	Android 6.0.1
Fog device 1	4 GB	250 GB	Intel Core i5	Windows 7 Professional
Fog device 2	4 GB	250 GB	Intel Core i5	Ubuntu
VM instance in Google Cloud Platform (Remote Cloud referred as GCP)	3.75 GB	250 GB	1vCPU	Windows Server 2012 R2 Datacenter
IITKGP_regular VM OpenStack (Local Cloud referred as MGL)	4 GB	45 GB	2 vCPU	Ubuntu 14.04

spatial reference system, EPSG:32645. We have considered the geospatial data related to road, land area and railway track of Purulia, India, road network of Mumbai, India, road network of Delhi, India, and forest and road network of Raipur, India. Geospatial data are of two types: vector data and raster data. In this experiment, we have considered vector data. Two mobile phones send geospatial queries related to geospatial data of Purulia, Delhi, Mumbai and Raipur. Fog device 1 and fog device 2 are containing the data of Purulia and Raipur respectively. The data of Mumbai and Delhi are stored in the cloud. The two mobile devices generate total six geospatial queries which come under the three case studies, discussed in Section 4. Concerning the three cases, the analysis of generated geospatial queries from two mobile devices is discussed in the Subsections 5.2.1–5.2.3.

We have used QGIS (version 2.18.28) for analysing the geospatial queries. The time complexity for buffer creation is $O(N)$, for insertion and retrieval of event points is $O(\log N)$, for intersection is $O(N^2)$, where N is the number of edges for a geometric buffer. The results of data analysis are presented in Figs. 11–16. The total delay in geospatial query resolution are presented in Table 6. In this case we have measured the round trip delay (difference between the time stamp of sending request and receiving response). The power consumption of the mobile phones during this period is presented in Table 6. To determine the power consumption we have multiplied the measured delay with the power consumption of mobile device per unit time. The results are compared with geospatial query resolution using the existing remote cloud-based system [11,46], and local cloud-based system.

5.2.1. Experimental results of geospatial query analysis by fog device of current region

The user of mobile phone 1 generates the first geospatial query, where he asks for an information related to Raipur, and the geospatial query is sent to Fog device 2, with which it is connected. As fog device 2 is containing the geospatial data of Raipur, it responds. The user of mobile phone 2 generates the second geospatial query, where he asks for an information related to Purulia, and the geospatial query is sent to Fog device 1, with which it is connected. As fog device 1 is containing the geospatial data of Purulia, it responds.

Result of geospatial query 1 resolution: In Fig. 11, the result of geospatial query 1 is displayed. The geospatial query is to find the

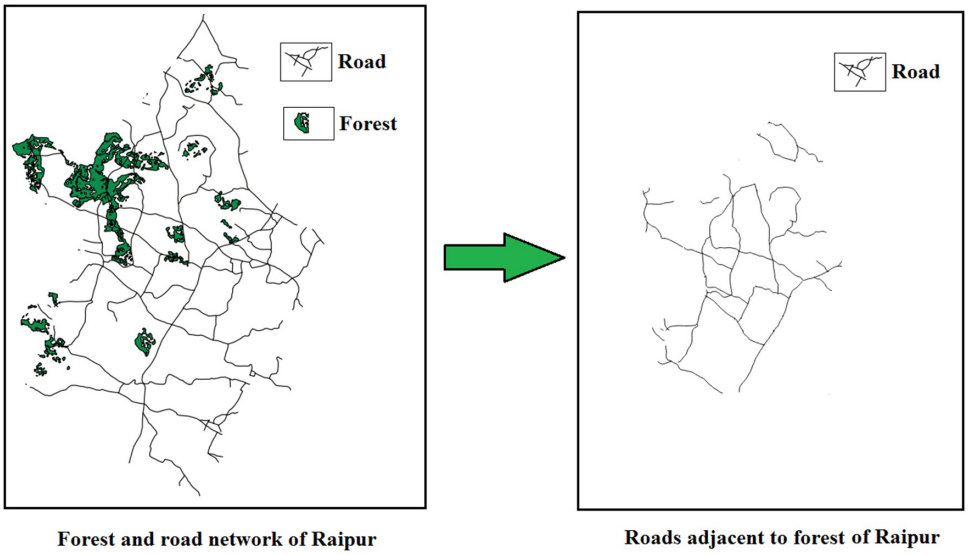

Fig. 11. Result of Geospatial Query 1: connecting roads to forest in Raipur.

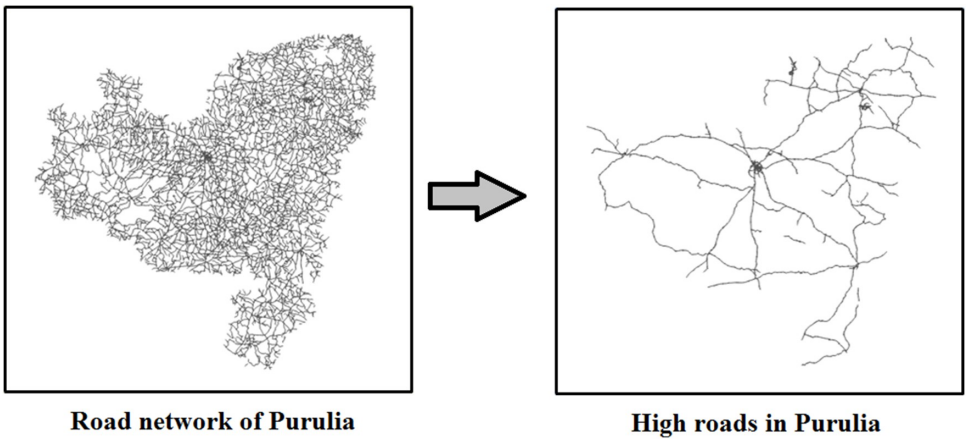

Fig. 12. Result of Geospatial Query 2: high roads in Purulia.

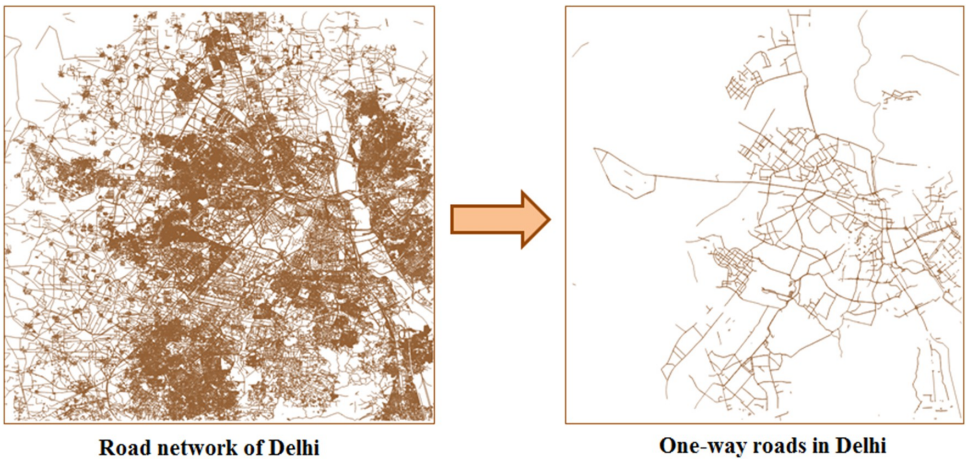

Fig. 13. Result of Geospatial Query 3: one-way roads in Delhi.

road adjacent to the forest of Raipur. The forest area data and road network data of Raipur are analysed by fog device 2 to resolve the geospatial query. The following three geospatial services used to resolve this geospatial query:

- WPS *BufferFeatureCollection* service is for 1 km buffer creation around Raipur forest shape file.
- WPS *IntersectionFeatureCollection* service is for the overlap of Raipur road and buffered forest shape file.
- WMS *GetMap* service is used to display the resultant map of geospatial query 1.

The amount of data transmission, delay and power consumption of the user device are presented in Table 6. The amount of data transmission between consecutive nodes, in this case, is 1.98 MB. The delay in geospatial query resolution using the proposed model, only GCP (remote cloud VM) and only MGL (local cloud VM) are 1.72 s, 4.71 s, and 3.31 s respectively. The power consumptions of the user device during this period are 0.19 W, 0.52 W, and 0.365 W respectively for geospatial query resolution using the proposed model, only GCP, and only MGL. The experimental results show that the proposed model reduces the delay in geospatial query resolution by 63% and 48% approximately than only GCP and only MGL respectively. The experimental results also show that using the proposed model the power consumption of the user device is reduced by 63% and 48% approximately than only GCP and only MGL respectively.

Result of geospatial query 2 resolution: In Fig. 12, the result of geospatial query 2 is displayed. The geospatial query is to find the high roads in Purulia. The geospatial data of road of Purulia is analysed by fog device 1 to resolve the geospatial query. The following two geospatial services are used to resolve this geospatial query:

- WFS *getFeature* service is used for filtering the 'High Roads' from Purulia road network.
- WMS *GetMap* service is used to display the 'High Roads' map of Purulia.

The amount of data transmission, delay, and power consumption of the user device are presented in Table 6. The amount of data

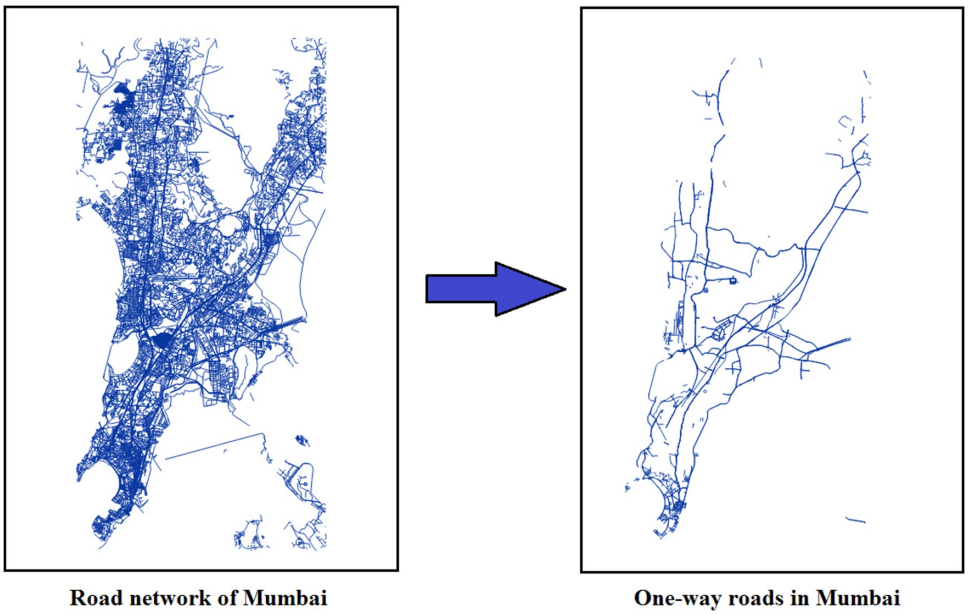

Fig. 14. Result of Geospatial Query 4: one-way roads in Mumbai.

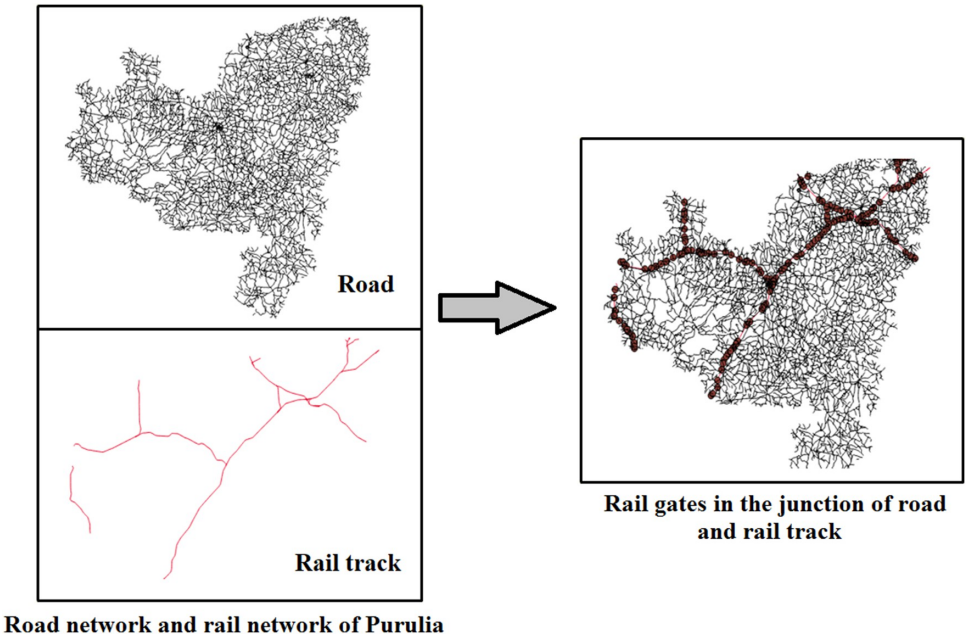

Fig. 15. Result of Geospatial Query 5: rail gates in the junction of road and rail track in Purulia.

transmission between consecutive nodes, in this case, is 1.56 MB. The delay in geospatial query resolution using the proposed model, only GCP (remote cloud VM) and only MGL (local cloud VM) are 1.13 s, 3.61 s, and 2.21 s respectively. The power consumption of the user device during this period are 0.124 W, 0.397 W, and 0.243 W respectively for geospatial query resolution using the proposed model, only GCP, and only MGL. The experimental results show that the use of the proposed model reduces the delay in geospatial query resolution by 71% and 49% approximately than only GCP and only MGL respectively. The experimental results also show that using the proposed model, the power consumption of the user device is reduced by 68% and 49% approximately than only GCP, and only MGL respectively. Hence, it is observed that the proposed framework has $\sim$ 60% less power consumption of user device and $\sim$ 60% less delay than the existing remote cloud only system [11,46].

5.2.2. Experimental results of geospatial query analysis by cloud

The user of mobile phone 1 generates the third geospatial query, where he asks for an information related to Delhi, and the

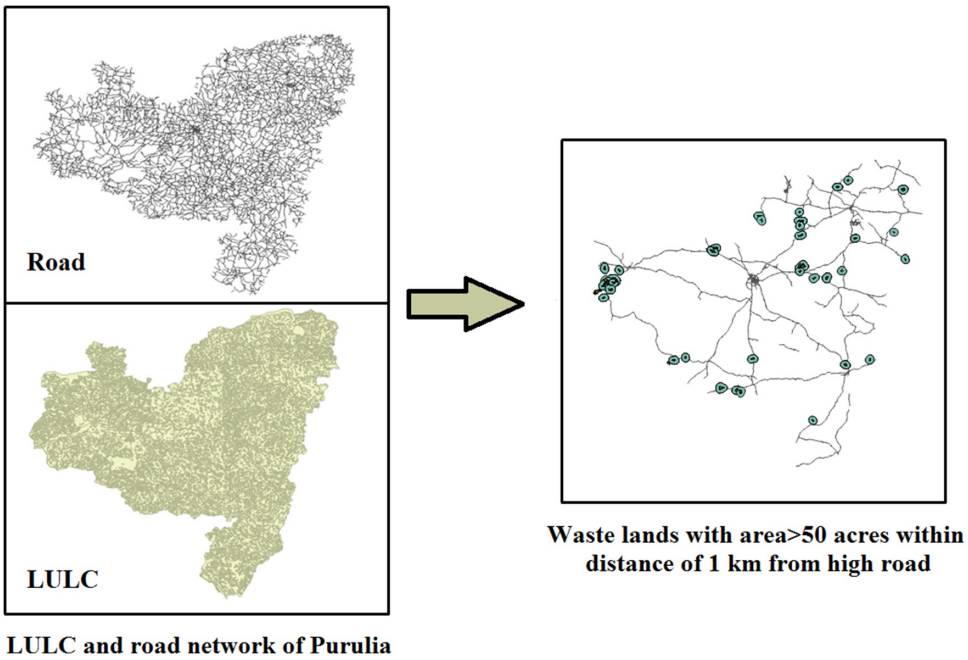

Fig. 16. Result of Geospatial Query 6: waste lands with area of greater than 50 acres within the distance of 1 km from high road in Purulia.

geospatial query is sent to Fog device 2. Nevertheless, fog device 2 does not have the geospatial data of Delhi. Hence, it forwards the request to the cloud. The cloud has the geospatial data of Delhi. Thus, it processes the geospatial query and then sends the result to fog device 2. Fog device 2 forwards the result to mobile phone 1. The user of mobile phone 2 generates the fourth geospatial query, where he asks for an information related to Mumbai, and the geospatial query is sent to Fog device 1. However, fog device 1 is not containing the geospatial data of Mumbai. Hence, it forwards the request to the cloud. The cloud has the geospatial data of Mumbai, and it sends the result to fog device 1 after processing the geospatial query. Fog device 1 forwards the result to mobile phone 2.

Result of geospatial query 3 resolution: In Fig. 13, the result of geospatial query 3 is displayed. The query is to find one-way roads in Delhi. The road network data of Delhi is analysed by the cloud to resolve the geospatial query. The following two geospatial services are used to resolve this geospatial query:

- WFS *getFeature* service is used to filter out the 'One_Way' roads from the Delhi road network.
- WMS *GetMap* service is used to exhibit the geospatial query 3 result.

The amount of data transmission, delay, and power consumption of the user device are presented in Table 6. The amount of data transmission between consecutive nodes, in this case, is 2.28 MB. The delay in geospatial query resolution using the proposed model, only GCP (remote cloud VM), and only MGL (local cloud VM) are 2.85 s, 4.91 s, and 4.45 s respectively. The power consumption of the user device during this period are 0.31 W, 0.54 W, and 0.49 W respectively for geospatial query resolution using the proposed model, only GCP, and only MGL. The experimental results show that the use of the proposed model reduces the delay in geospatial query resolution by 41% and 35% approximately than only GCP and only MGL respectively. The experimental results also show that using the proposed model, the power consumption of the user device is reduced by 42% and 36% approximately than only GCP, and only MGL respectively.

Result of geospatial query 4 resolution: In Fig. 14, the result of geospatial query 4 is displayed. The geospatial query is to find one-way roads in Mumbai. The road network data of Mumbai is analysed by cloud to resolve the geospatial query. The following two geospatial services are used to resolve this geospatial query:

- WFS *getFeature* service is used to filter the 'One_Way' roads from the Mumbai road network.
- For displaying the result of geospatial query 4, WMS *GetMap* service is used.

The amount of data transmission, delay, and power consumption of the user device are presented in Table 6. The amount of data transmission between consecutive nodes, in this case, is 2.32 MB. The delay in geospatial query resolution using the proposed model, only GCP (remote cloud VM) and only MGL (local cloud VM) are 2.91 s, 5.22 s, and 4.45 s respectively. The power consumption of the user device during this period are 0.32 W, 0.57 W, and 0.49 W respectively for geospatial query resolution using the proposed model, only GCP, and only MGL. The experimental results show that the use of the proposed model reduces the delay in geospatial query resolution by 44% and 35% approximately than only GCP and only MGL respectively. The experimental results also show that using

Table 6
Delay and power consumption results obtained from experiment.

No.	Geospatial Query	Transmitted data between consecutive nodes	Delay	Power consumption of mobile device	Reduction using proposed framework
1	SELECT road_id FROM Raipur WHERE Overlap(road.shape, Buffer (forest.Shape,1));	1.98 MB	1.72 s using Spatio-Fog, 4.71 s using GCP, 3.31 s using MGL	0.19 W using Spatio-Fog, 0.52 W using GCP, 0.365 W using MGL	Delay: 63% than GCP, 48% than MGL, Power: 63% than GCP, 48% than MGL
2	SELECT road_name FROM Purulia WHERE road = 'High Road' ORDER BY area desc;	1.56 MB	1.13 s using Spatio-Fog, 3.61 s using GCP, 2.21 s using MGL	0.124 W using Spatio-Fog, 0.397 W using GCP, 0.243 W using MGL	Delay: 71% than GCP, 49% than MGL, Power: 68% than GCP, 49% than MGL
3	SELECT road_name FROM Delhi WHERE road_type = 'One_Way';	2.28 MB	2.85 s using Spatio-Fog, 4.91 s using GCP, 4.45 s using MGL,	0.31 W using Spatio-Fog, 0.54 W using GCP, 0.49 W using MGL	Delay: 41% than GCP, 35% than MGL, Power: 42% than GCP, 36% than MGL
4	SELECT road_name FROM Mumbai WHERE road_type = 'One_Way';	2.32 MB	2.91 s using Spatio-Fog, 5.22 s using GCP, 4.45 s using MGL,	0.32 W using Spatio-Fog, 0.57 W using GCP, 0.49 W using MGL	Delay: 44% than GCP, 35% than MGL, Power: 43% than GCP, 34% than MGL
5	SELECT point FROM Purulia.rail Rl, Purulia.road Rd WHERE Cross(Rl.Shape, Rd.Shape)=1 ORDER BY area desc;	2.12 MB	2.76 s using Spatio-Fog, 3.94 s using GCP, 3.32 s using MGL	0.31 W using Spatio-Fog, 0.45 W using GCP, 0.38 W using MGL	Delay: 30% than GCP, 18% than MGL, Power: 31% than GCP, 17% than MGL
6	SELECT area_name FROM Purulia WHERE *area* > 50 and road = 'High Road' and Overlap (road.shape, Buffer(area.shape,1)) ORDER BY area desc;	3.15 MB	5.25 s using Spatio-Fog, 7.66 s using GCP, 6.35 s using MGL	0.577 W using Spatio-Fog, 0.842 W using GCP, 0.69 W using MGL	Delay: 31% than GCP, 17% than MGL, Power: 31% than GCP, 16% than MGL

the proposed model, the power consumption of the user device is reduced by 43% and 34% approximately than only GCP, and only MGL respectively. Hence, the proposed framework has ∼ 40% less power consumption of user device and ∼ 40% less delay than the existing remote cloud only system [11,46].

5.2.3. Experimental results of geospatial query analysis by fog device of another region

The user of mobile phone 1 generates the last two geospatial queries, where he asks for information related to Purulia, and the geospatial query is sent to Fog device 2. However, fog device 2 is not containing the geospatial data of Purulia. Hence, it forwards the fifth and sixth geospatial queries to the cloud. But the cloud is also not containing the data of Purulia, hence forwards it to fog device 1 that is containing the data of Purulia. Fog device 1 then processes the fifth and sixth geospatial queries and sends the result directly to fog device 2. Fog device 2 forwards the result to mobile phone 1.

Result of geospatial query 5 resolution: In Fig. 15, the result of geospatial query 5 is displayed. The geospatial query is to find the rail gates in the junction of road and rail track in Purulia. The geospatial data of rail track and road of Purulia are analysed by fog device 1 to resolve the geospatial query. The following two geospatial services are used to resolve this geospatial query:

- WPS *LineIntersectionFeature* service is used for performing the cross operation on the road network and rail tracks of Purulia.
- WMS *GetMap* service is used to display the final map of geospatial query 5.

The amount of data transmission, delay, and power consumption of the user device are presented in Table 6. The amount of data transmission between consecutive nodes, in this case, is 2.12 MB. The delay in geospatial query resolution using the proposed model, only GCP (remote cloud VM) and only MGL (local cloud VM) are 2.76 s, 3.94 s, and 3.32 s respectively. The power consumption of the user device during this period are 0.31 W, 0.45 W, and 0.38 W respectively for geospatial query resolution using the proposed model, only GCP, and only MGL. The experimental results show that the use of the proposed model reduces the delay in geospatial query resolution by 30% and 18% approximately than only GCP and only MGL respectively. The experimental results also show that using the proposed model, the power consumption of the user device is reduced by 31% and 17% approximately than only GCP and only MGL respectively.

Result of geospatial query 6 resolution: In Fig. 16, the result of geospatial query 6 is displayed. The geospatial query is to find wastelands with an area greater than 50 acres within the distance of 1 km from the high road in Purulia. The geospatial data of road and land use land cover (LULC) of Purulia are analysed. The amount of data transmission, delay and power consumption of the user device are presented in Table 6. The following four geospatial services are used to resolve this geospatial query:

- WFS *getFeature* service is used twice to filter the 'High Road' roads from Purulia road network and to filter areas which are greater than 50 acres from Purulia LULC.
- To create 1 km buffer of each filtered area using WPS *BufferFeatureCollection* service.
- WPS *IntersectionFeatureCollection* service is used to make intersection Land buffer with filtered Roads.
- Finally, WMS *GetMap* service is used to display the resultant map of the geospatial query 6.

The amount of data transmission between consecutive nodes in this case is 3.15 MB. The delay in geospatial query resolution using the proposed model, only GCP (remote cloud VM) and only MGL (local cloud VM) are 5.25 s, 7.66 s and 6.35 s respectively. The power consumption of the user device during this period are 0.577 W, 0.842 W and 0.69 W respectively for geospatial query resolution using the proposed model, only GCP and only MGL. The experimental results show that use of the proposed model reduces the delay in geospatial query resolution by 31% and 17% approximately than using only GCP and only MGL respectively. The experimental results also show that using fog device the power consumption of the user device is reduced by 31% and 16% approximately than using only GCP and only MGL respectively.

Hence, from this two experimental studies it is observed, the proposed framework has ∼ 30% less power consumption and ∼ 30% less delay than the existing remote cloud only system [11,46].

The experimental results in Table 6 illustrate that the use of the proposed model reduces delay and power than existing cloud-based system. In the first two experimental studies the fog device connected with the requesting mobile phone is containing data of the region for which the geospatial queries are made. Hence, the reduction in delay and power consumption of user device is ∼ 60% than the existing remote cloud-based system [11,46]. In the next two experimental studies the fog device connected with the requesting mobile phone is not containing data of the region for which the geospatial queries are made, whereas the cloud is containing the data. Hence, the communication cost in terms of delay and power consumption is same as in the existing cloud-based system. However, in the existing system the cloud has to access a huge volume of data to respond to the geospatial query, as the geospatial data of all the regions is contained by the cloud. As a result the computational cost is higher than the proposed system where cloud is containing only the frequently accessed geospatial data. Consequently the total cost (sum of communication and computation) is higher in the existing system. It is observed that use of cloud for geospatial query resolution in case of the proposed Spatio-Fog architecture achieves ∼ 40% less delay and power consumption of user device than the existing remote cloud-based system [11,46]. In the last two experimental studies the third case is considered i.e. the use of another fog device for geospatial query resolution. In this case the proposed model has higher communication cost than the existing cloud-based system. Nevertheless, the computational cost in case of the remote cloud-based system is much higher (as discussed earlier) than the proposed model. As a result the total cost becomes less in the proposed framework. It is observed that use of fog device of another region for geospatial query resolution in case of the proposed Spatio-Fog architecture achieves ∼ 30% less delay and power consumption of user device than the existing remote

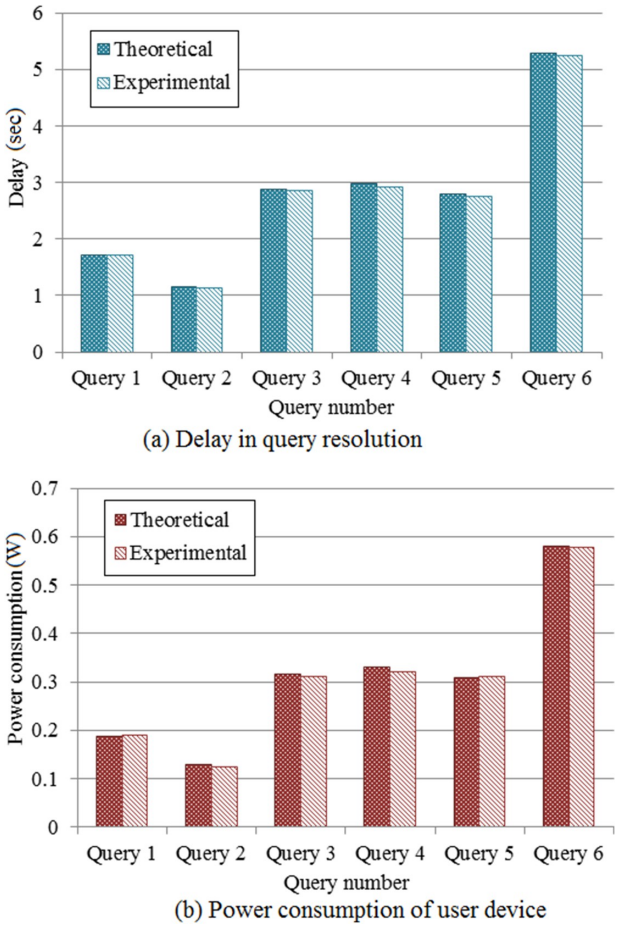

Fig. 17. Comparison between the results of empirical study and experimental analysis of geospatial query resolution.

cloud-based system [11,46]. Hence, from the six experimental studies, it is observed that the proposed system provides 30% − 60% less power consumption of user device and 30% − 60% less delay than the existing system.

5.3. Comparative study

In the empirical study, we have considered large data size, whereas in the experimental analysis, we have a comparatively small amount of data. Hence the results differ. However, we have performed the empirical study also for the six geospatial queries discussed above. Using the mathematical model, the delay in geospatial query resolution and power consumption of user device during that period have been calculated and compared with the experimental results, which is graphically presented in Fig. 17. It is observed from Fig. 17 that the empirical study and corresponding experimental results are approximately the same. Experimentally it is already noted that the proposed framework reduces the power consumption and delay than the existing frameworks. In the proposed framework, the use of fog device in geospatial query resolution reduces the delay and consequently the power consumption of the mobile device during the time of query resolution. Moreover, the storage of regional geospatial data inside the fog devices instead of putting the entire geospatial data inside the cloud reduces the probability of accessing the cloud for geospatial query resolution. This in turn reduces the power consumption of the cloud for spatial query resolution. Further, the network load to the cloud servers via internet is also reduced. Hence, it is concluded that the proposed fog computing system Spatio-Fog is a low power, i.e., green and delay-sensitive system. Here, green refers to power-efficiency. Usually a green system refers to a low power system. As the proposed Spatio-Fog has low power consumption, it is referred as a green i.e. power-efficient framework.

5.4. System response time

System response time refers to the time difference between receiving a request and sending the corresponding response by a system i.e. the sum of waiting time and service time is the response time [12]. If a system receives N geospatial queries and the response time of a geospatial query GQ is T_{GQ}, then the average system response time is given as, $RT_N = \frac{\sum_{GQ=1}^{N} T_{GQ}}{N}$. The average system

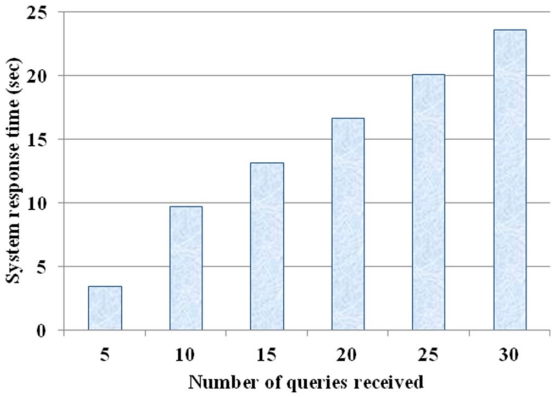

Fig. 18. System Response Time in proposed Spatio-Fog system.

response time with respect to the number of geospatial queries received are presented in Fig. 18. From the figure it is observed that the average system response time for the proposed Spatio-Fog system is < 25 s for the number of received geospatial queries 5–30.

6. Conclusions and future work

In this paper, a fog computing based architecture, namely Spatio-Fog, has been proposed for geospatial query resolution. The fog devices, located in the different regions, contain geospatial data of the respective regions. When a geospatial query is received from a mobile device related to the current region, the fog device resolves the geospatial query after analysing the data and responds to the mobile device. Otherwise, if the geospatial query is related to other regions, the fog device responds using cloud servers or fog device of the corresponding region. The empirical study shows that the proposed framework reduces the power consumption and delay by approximately 43–47% and 47–83% respectively than the existing system. The experimental results illustrate that the proposed framework reduces the power consumption and delay by 30–60% approximately than the existing query resolution system. Thus, we conclude that the proposed framework is a green and delay aware framework.

The availability of the network connectivity is a challenge if the mobile client is present at a remote location. Hence, the client needs the service with minimal network facility even in offline mode. If such a mobile user has a geospatial query, then to resolve it dew computing can be a solution. Thus, creation, update, and access to raw geospatial data set for resolving the geospatial query for the devices connected in offline mode, use of dew computing will be a promising future research scope of this work.

Acknowledgements

This research is partially supported by the Department of Science and Technology, Government of India through Geospatial Chair Professorship research project, Indian Institute of Technology Kharagpur. This work is also partially supported by Melbourne-Chindia Cloud Computing (MC3) Research Network.

References

[1] M. Malensek, S. Pallickara, S. Pallickara, Fast, ad hoc query evaluations over multidimensional geospatial datasets, IEEE Trans. Cloud Comput. 5 (1) (2017) 28–42.
[2] J. Das, A. Dasgupta, S.K. Ghosh, R. Buyya, A geospatial orchestration framework on cloud for processing user queries, Proceedings of International Conference on Cloud Computing in Emerging Markets (CCEM), IEEE, 2016, pp. 1–8.
[3] A. Mukherjee, D. De, Femtolet: a novel fifth generation network device for green mobile cloud computing, Simul. Model. Pract. Theory 62 (2016) 68–87.
[4] M. Satyanarayanan, P. Bahl, R. Caceres, N. Davies, The case for VM-based cloudlets in mobile computing, IEEE Pervasive Comput. 4 (2009) 14–23.
[5] A. Mukherjee, D. De, D.G. Roy, A power and latency aware cloudlet selection strategy for multi-cloudlet environment, IEEE Trans. Cloud Comput. 7 (1) (2016) 141–154.
[6] A.V. Dastjerdi, R. Buyya, Fog computing: helping the internet of things realize its potential, Computer 49 (8) (2016) 112–116.
[7] D. Tychalas, H. Karatza, A scheduling algorithm for a fog computing system with bag-of-tasks jobs: Simulation and performance evaluation, Simul. Model. Pract. Theory 98 (2020) 101982.
[8] M. Forcan, M. Maksimović, Cloud-fog-based approach for smart grid monitoring, Simul. Model. Pract. Theory (2019) 101988.
[9] F. Bonomi, R. Milito, P. Natarajan, J. Zhu, Fog computing: a platform for internet of things and analytics, Big Data and Internet of Things: A Roadmap for Smart Environments, Springer, 2014, pp. 169–186.
[10] A. Mukherjee, P. Deb, D. De, R. Buyya, C2OF2N: a low power cooperative code offloading method for femtolet-based fog network, J. Supercomput. 74 (6) (2018) 2412–2448.
[11] C. Yang, M. Yu, F. Hu, Y. Jiang, Y. Li, Utilizing cloud computing to address big geospatial data challenges, Comput. Environ. Urban Syst. 61 (2017) 120–128.
[12] C. Yang, M. Goodchild, Q. Huang, D. Nebert, R. Raskin, Y. Xu, M. Bambacus, D. Fay, Spatial cloud computing: how can the geospatial sciences use and help shape cloud computing? Int. J. Digit. Earth 4 (4) (2011) 305–329.
[13] S. Li, S. Dragicevic, F.A. Castro, M. Sester, S. Winter, A. Coltekin, C. Pettit, B. Jiang, J. Haworth, A. Stein, et al., Geospatial big data handling theory and methods: a review and research challenges, ISPRS J. Photogramm. Remote Sens. 115 (2016) 119–133.
[14] J.-G. Lee, M. Kang, Geospatial big data: challenges and opportunities, Big Data Res. 2 (2) (2015) 74–81.
[15] R.K. Barik, H. Dubey, A.B. Samaddar, R.D. Gupta, P.K. Ray, FogGIS: fog computing for geospatial big data analytics, Proceedings of International Conference on

Electrical, Computer and Electronics Engineering (UPCON), 2016 IEEE Uttar Pradesh Section, IEEE, 2016, pp. 613–618.
[16] R.K. Barik, A. Tripathi, H. Dubey, R.K. Lenka, T. Pratik, S. Sharma, K. Mankodiya, V. Kumar, H. Das, MistGIS: optimizing geospatial data analysis using mist computing, Progress in Computing, Analytics and Networking, Springer, 2018, pp. 733–742.
[17] M. Das, S.K. Ghosh, V. Chowdary, A. Saikrishnaveni, R. Sharma, A probabilistic nonlinear model for forecasting daily water level in reservoir, Water Resour. Manag. 30 (9) (2016) 3107–3122.
[18] S. Bhattacharjee, M. Das, S.K. Ghosh, S. Shekhar, Prediction of meteorological parameters: an a-posteriori probabilistic semantic kriging approach, Proceedings of the 24th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems, ACM, 2016, p. 38.
[19] F. Gu, On-demand data assimilation of large-scale spatial temporal systems using sequential monte carlo methods, Simul. Model. Pract. Theory 85 (2018) 1–14.
[20] A. Dasgupta, S.K. Ghosh, P. Mitra, A technique for assessing the quality of volunteered geographic information for disaster decision making, International Conference on Computational Science and Its Applications, Springer, 2018, pp. 589–597.
[21] O. Chakraborty, J. Das, A. Dasgupta, P. Mitra, S.K. Ghosh, A geospatial service oriented framework for disaster risk zone identification, International Conference on Computational Science and Its Applications, Springer, 2016, pp. 44–56.
[22] S. Pal, S. Ghosh, A large-scale data-oriented intelligent system for urban growth simulation, Geospatial Infrastructure, Applications and Technologies: India Case Studies, Springer, 2018, pp. 143–154.
[23] S. Ghosh, S.K. Ghosh, Modeling of human movement behavioral knowledge from GPS traces for categorizing mobile users, Proceedings of the 26th International Conference on World Wide Web Companion, International World Wide Web Conferences Steering Committee, 2017, pp. 51–58.
[24] S. Ghosh, A. Mukherjee, S.K. Ghosh, R. Buyya, Mobi-IoST: mobility-aware cloud-fog-edge-iot collaborative framework for time-critical applications, IEEE Trans. Netw. Sci. Eng. (2019), https://doi.org/10.1109/TNSE.2019.2941754.
[25] S.M. Vicente-Serrano, S. Beguería, L. Gimeno, L. Eklundh, G. Giuliani, D. Weston, A.E. Kenawy, J.I. López-Moreno, R. Nieto, T. Ayenew, et al., Challenges for drought mitigation in africa: the potential use of geospatial data and drought information systems, Appl. Geogr. 34 (2012) 471–486.
[26] A.H. Weerts, J. Schellekens, F.S. Weiland, Real-time geospatial data handling and forecasting: examples from delft-fews forecasting platform/system, IEEE J. Sel. Top. Appl. Earth Obs. Remote Sens. 3 (3) (2010) 386–394.
[27] N.N. Kussul, B.V. Sokolov, Y.I. Zyelyk, V.A. Zelentsov, S.V. Skakun, A.Y. Shelestov, Disaster risk assessment based on heterogeneous geospatial information, J. Autom. Inf. Sci. 42(12) 32–45.
[28] C.-R. Shyu, M. Klaric, G.J. Scott, A.S. Barb, C.H. Davis, K. Palaniappan, GeoIRIS: geospatial information retrieval and indexing system content mining, semantics modeling, and complex queries, IEEE Trans. Geosci. Remote Sens. 45 (4) (2007) 839–852.
[29] J.-H. Hong, Z.L.-T. Su, E.H.C. Lu, A recommendation framework for remote sensing images by spatial relation analysis, J. Syst. Softw. 90 (2014) 151–166.
[30] D. Warren, Method and apparatus for providing location based data services, 2003, US Patent 6,611,751.
[31] L. Qi, X. Zhang, W. Dou, Q. Ni, A distributed locality-sensitive hashing-based approach for cloud service recommendation from multi-source data, IEEE J. Sel. Areas Commun. 35 (11) (2017) 2616–2624.
[32] S. Nishimura, S. Das, D. Agrawal, A.E. Abbadi, MD-HBase: a scalable multi-dimensional data infrastructure for location aware services, 2011 IEEE 12th International Conference on Mobile Data Management, 1, IEEE, 2011, pp. 7–16.
[33] S. Nishimura, S. Das, D. Agrawal, A.E. Abbadi, MD-HBase: design and implementation of an elastic data infrastructure for cloud-scale location services, Distrib. Parallel Databases 31 (2) (2013) 289–319.
[34] A. González, C. Aliagas, M. Valero, A data cache with multiple caching strategies tuned to different types of locality, International Conference on Supercomputing, Citeseer, 1995, pp. 338–347.
[35] M. Kandemir, A. Choudhary, J. Ramanujam, N. Shenoy, P. Banerjee, Enhancing spatial locality via data layout optimizations, European Conference on Parallel Processing, Springer, 1998, pp. 422–434.
[36] P. Gerolymatos, S. Sioutas, N. Nodarakis, A. Panaretos, K. Tsakalidis, SMaRT: a novel framework for addressing range queries over nonlinear trajectories, J. Syst. Softw. 105 (2015) 79–90.
[37] G. Roumelis, M. Vassilakopoulos, A. Corral, Y. Manolopoulos, Efficient query processing on large spatial databases: a performance study, J. Syst. Softw. 132 (2017) 165–185.
[38] H. Samet, The quadtree and related hierarchical data structures, ACM Comput. Surv. (CSUR) 16 (2) (1984) 187–260.
[39] I. Kamel, C. Faloutsos, Hilbert R-tree: an improved R-tree using fractals, Tech. rep. (1993).
[40] T. Sellis, N. Roussopoulos, C. Faloutsos, The R+-Tree: A dynamic index for multi-dimensional objects, Tech. rep. (1987).
[41] N. Beckmann, H.-P. Kriegel, R. Schneider, B. Seeger, The R*-tree: an efficient and robust access method for points and rectangles, ACM Sigmod Record, ACM, 1990, pp. 322–331. 19
[42] G.R. Hjaltason, H. Samet, Speeding up construction of PMR quadtree-based spatial indexes, Int. J. Very Large Data Bases 11 (2) (2002) 109–137.
[43] T. Dokeroglu, M.A. Bayir, A. Cosar, Robust heuristic algorithms for exploiting the common tasks of relational cloud database queries, Appl. Soft Comput. 30 (2015) 72–82.
[44] M. Malensek, S. Pallickara, S. Pallickara, Analytic queries over geospatial time-series data using distributed hash tables, IEEE Trans. Knowl. Data Eng. 28 (6) (2016) 1408–1422.
[45] K. Karantzalos, D. Bliziotis, A. Karmas, A scalable geospatial web service for near real-time, high-resolution land cover mapping, IEEE J. Sel. Top. Appl. Earth Obs. Remote Sens. 8 (10) (2015) 4665–4674.
[46] K. Evangelidis, K. Ntouros, S. Makridis, C. Papatheodorou, Geospatial services in the cloud, Comput. Geosci. 63 (2014) 116–122.
[47] A. Rezgui, Z. Malik, C. Yang, High-resolution spatial interpolation on cloud platforms, Proceedings of the 28th Annual ACM Symposium on Applied Computing, ACM, 2013, pp. 377–382.
[48] J. Wenjue, C. Yumin, G. Jianya, Implementation of OGC web map service based on web service, Geo-Spatial Inf. Sci. 7 (2) (2004) 148–152.
[49] P.A. Vretanos, Web feature service implementation specification, Open Geospatial Consort. Specif. 1325 (2005) 04–094.
[50] P. Schut, A. Whiteside, OpenGIS web processing service, Version 1.0. 0, OGC 05-007r7, Open Geospatial Consortium, Inc 87.
[51] R. Barik, H. Dubey, S. Sasane, C. Misra, N. Constant, K. Mankodiya, Fog2fog: augmenting scalability in fog computing for health GIS systems, Proceedings of the Second IEEE/ACM International Conference on Connected Health: Applications, Systems and Engineering Technologies, IEEE Press, 2017, pp. 241–242.
[52] G. Buehrer, B.W. Weide, P.A. Sivilotti, Using parse tree validation to prevent SQL injection attacks, Proceedings of the 5th international workshop on software engineering and middleware, ACM, 2005, pp. 106–113.
[53] R.P. Haining, R. Haining, Spatial Data Analysis: Theory and Practice, Cambridge University Press, 2003.
[54] T. Brinkhoff, H.-P. Kriegel, B. Seeger, Efficient Processing of Spatial Joins Using R-Trees, ACM, 1993. Vol. 22
[55] J. Das, A. Dasgupta, S.K. Ghosh, R. Buyya, A learning technique for VM allocation to resolve geospatial queries, Recent Findings in Intelligent Computing Techniques, Vol. 1, Springer, 2019, pp. 577–584.
[56] A. Mukherjee, D. De, R. Buyya, E2R-F2N: energy-efficient retailing using a femtolet-based fog network, Software 49 (3) (2019) 498–523.