Design and Analysis of An Intelligent Sensor Code Dissemination System On-The-Move in Smart Cities

Salman Naseer[a*], Iqra Zadar[b], Muhammad Mudasar Ghafoor[a], M. Qamar Khan[c]

[a]Department of Information Technology, University of the Punjab (Gujranwala Campus), Gujranwala, Pakistan.
[b]Punjab College for Women, Punjab Group of Colleges, Gujranwala, Pakistan.
[c]Pervaiz Elahi Institute of cardiology, Wazirabad, Pakistan

*Corresponding Author: salman@pugc.edu.pk

Abstract

In the era of smart cities, the Internet of Things (IoT) has led to the widespread deployment of sensor-based devices for real-time data collection and environmental monitoring. However, the challenge of remotely updating software on these devices—often dispersed, low-powered, and delay-tolerant—remains a key barrier to their adaptive functionality. This paper proposes an **Optimal Code Forwarding Scheme (OCFS)**, a novel strategy to disseminate software updates to smart sensors (SSs) using vehicular opportunistic communication. By leveraging the mobility of smart vehicles as Mobile Nodes (MNs) and strategically positioned Road Side Units (RSUs), OCFS optimizes code delivery without requiring changes to vehicle trajectories. A multi-objective optimization model is introduced to maximize code coverage and minimize transmission delays, transforming RSU and MN selection into a minimum set coverage problem. The proposed system was evaluated using real-world GPS traces from 400 taxis in Beijing, demonstrating that OCFS achieves up to **98.72% code coverage** with significantly reduced dissemination time compared to fixed and random deployment schemes. These results validate the feasibility of scalable, low-cost, delay-tolerant code dissemination in large-scale smart city environments.

Keywords: Smart Cities, Spatial Data Mining, Grid Clustering, Big Data, Delay Tolerant Network, Sensor Networks, GPS Traces, Internet of Things, Intelligent Transportation System.

Introduction

The Internet of Things (IoT) relies on a vast number of sensor-based devices to facilitate efficient and convenient data collection. It introduces an innovative approach to addressing complex sensing challenges in various domains, such as telemedicine systems, intelligent traffic management systems, and environmental monitoring systems. These sensors can be deployed extensively across target areas to gather real-time data tailored to specific application requirements (Ali et al., 2022). As the number of deployed sensors increases, the volume of generated data grows exponentially. However, a large volume of data alone does not necessarily meet the precise needs of an application. To overcome this limitation, the concept of *Smart Data* has emerged. Smart Data focuses on eliminating irrelevant or noisy information and retaining valuable, meaningful insights, thereby enabling more effective planning, operation, monitoring, control, and intelligent decision-making within IoT environments (Ahmad et al., 2022).

However, acquiring Smart Data from IoT systems remains a significant challenge. One viable and effective solution is to integrate data filtering algorithms directly into sensor-based devices. By enabling these devices to locally process and filter the data they generate before transmitting it to the network, more meaningful information can be extracted, and the overall network load can be substantially reduced (Naseer, Liu, Sarkar, Shafiq, & Choi, 2021). With the rapid advancement of artificial intelligence (AI), new AI algorithms are being developed and introduced at an unprecedented pace. In this context, a key challenge is how to efficiently disseminate program code to sensor devices, enabling them to perform intelligent functions and support Smart Data processing within IoT environments (Naseer & Chaudhry, 2011).

However, disseminating program code to deployed sensor devices presents a significant challenge. In earlier stages, sensor-based networks typically consisted of a small number of devices deployed over limited areas, often requiring dedicated wired connections to the Internet. Due to the high cost of deployment, these networks were used only in critical or high-priority applications. Today, with the rapid advancement of the Internet of Things (IoT), the landscape has changed dramatically (Benkirane et al., 2023). A wide range of buildings and infrastructure are now equipped with embedded sensor devices serving diverse functions. These devices can monitor their operational state as well as environmental conditions (Velusamy et al., 2021).

Statistically, the number of embedded sensor devices now far exceeds the human population. Beyond those integrated during manufacturing, many sensor devices are added post-production to support specific applications (Naseer et al., 2018). For instance, smart sensing devices are commonly embedded in buildings, streetlights,

garbage bins, and even advertising displays to monitor operational status (Arsalan, Burhan, Naseer, & Rehman, 2022). Smart garbage bins can detect and report fill levels to data centers, while sensor-equipped streetlights can measure parameters such as brightness and operational time. Such applications are becoming increasingly common across Smart City initiatives (Kaur et al., 2022).

Sensor devices typically face several common challenges:
(a) These devices are generally compact and easily embedded in the monitored targets. However, their small size often results in limited energy capacity and short communication ranges.
(b) In most scenarios, these devices lack built-in networking capabilities, making it impractical to transmit upgrade codes via wired networks.
(c) Furthermore, these devices are often deployed in large numbers, widely distributed, unattended, and may be relocated frequently (Ahmad, Manzoor, Naseer, Ghaffar, & Hussein, 2021). This makes fixed network deployment unrealistic. For instance, smart sensor-based garbage bins are designed for flexible deployment and are positioned based on real-time needs (Abrar et al., 2021). During peak tourist seasons, additional smart bins may be required in remote or hazardous areas to accommodate increased usage (Naseer & Mahmood, 2015). As a result, deploying dedicated network infrastructure for such facilities becomes both impractical and cost-prohibitive (Sandhu, Haider, Naseer, & Ateeb, 2011).

A key characteristic of such applications is their tolerance for delays in code dissemination to sensor-based devices. Similarly, the data collected by these devices typically does not need to be transmitted to the data center in real time (Arshad et al., 2023). As a result, the network formed by these devices can be classified as a delay-tolerant network (DTN) (Satti et al.). Generally, updated sensor devices remain compatible with those that have not yet received the update, allowing both versions to operate concurrently without disruption. This eliminates the need for simultaneous updates across an entire city (Naseer, Ghafoor, bin Khalid Alvi, & ul Islam, 2022). During the code update process, devices running both the old and new versions coexist—often for extended periods, ranging from several days to even months, depending on the specific application (Naseer, Ghafoor, bin Khalid Alvi, Zafar, & Murtaza, 2023).

This delay tolerance also applies to data collection. For example, in ecologically sensitive environments, sensor devices may periodically measure variables such as temperature and humidity, store the data locally, and wait for an appropriate opportunity to transmit it to the data center (Naseer, 2021). In such cases, data can be stored locally for days or even a month without compromising the application's effectiveness (Khan et al., 2023). These applications typically involve long-term monitoring, where uncovering macro-level patterns may require data accumulated over decades. In other scenarios—such as garbage bin fill-level monitoring, bridge

deformation tracking, or urban greenery assessment—a delay of several hours or even 1–2 days in data transmission is considered acceptable (Zafar, Naseer, & Ghafoorb).

The key contributions in this paper are listed below,

1. We propose an optimal code forwarding scheme (OCFS) for software update code transmission from end-user to smart sensors (SSs), where optimal mobile nodes (MNs) and roadside units (RSUs) are selected to solve a multi-objective optimization problem that manages a balance between coverage enhancement and minimize time duration for this transmission.
2. Finally, we evaluated the proposed OCFS using real taxi traces of Beijing city. Our results show that OCFS enhances the code transmission coverage rate and reduce the time duration for this dissemination.

The rest of the paper is organized is as follow, in section II Optimal code forwarding scheme is introduced, whereas in section III we introduced the simulation parameter and in section IV a detailed analysis is given. Final the paper is concluded in section V.

Optimal code forwarding scheme (OCFS)

Smart sensors are smart devices and their functionality can be updated according to need. They will receive their software update code from MNs and update themselves automatically. A huge number of SS are densely deployed throughout the smart city closed to the road network. The working of these sensors can be adjusted according to application requirements. e.g, a temperature sensor can detect the temperature in a fire-prone area; a sensor can sense the filling level of garbage in the garbage can; for weather prediction wind sensors are required and a sensor can be used to determine building or road deformation structure. Suppose we have a set of sensors $S_{ss} = \{S_1, S_2, \ldots S_k\}$. To update these sensors suppose we have a set of codes $S_{codes} = \{c_1, c_2 \ldots, c_m\}$. RSUs have a consistent power supply, are directly connected to CC and data center DC. They will collect the updated code from CC and forward it to SS by using MN. We have a set of RSU $S_{RSU} = \{r_1, r_2, \ldots r_n\}$.

In this scheme, MNs are smart vehicles moving around the city, having storage, communication, GPS, and processing facility. These MNs get updated code for SSs from RSUs, work in a store carry forward manners and transfer the code to SS whenever they are in their wireless range. In general, these mobile nodes can be used without any change. They are not specifically responsible to forward the code to SSs. In other words, the trajectory of these MNs doesn't need to change by our proposed code forwarding scheme but it upon the destination of the rider. This non-scheduled and non-regular code forwarding communication between SSs and RSUs by using MNs is called vehicular opportunistic communication. The following are the main steps, how this scheme will work, as shown in Figure 1.

1. Whenever a user needs to update a set of sensors, he will send his request to CC, along with GPS locations of SS, updated code, and time interval D, the delay-tolerant indicator for task completion.

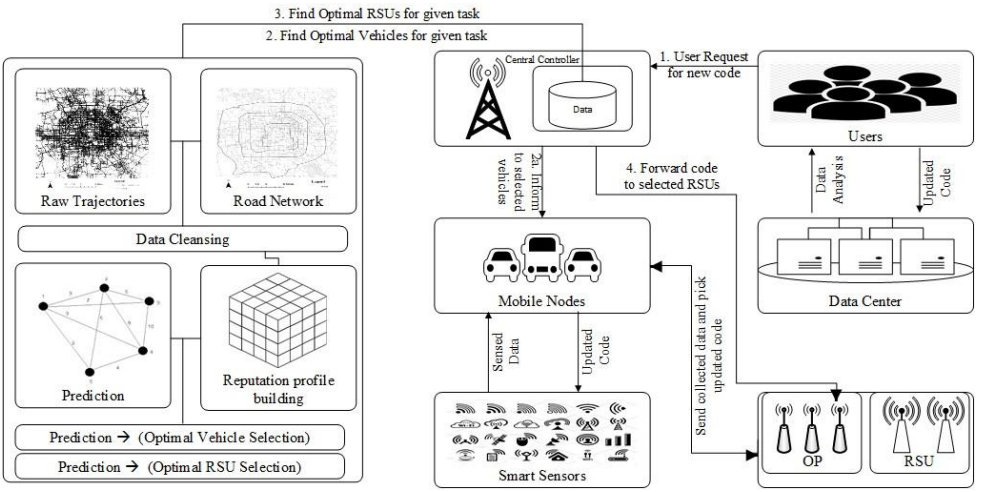

Figure 1: System overview.

2. CC finds the optimal set of vehicles for this task by using the vehicles trajectory history. Vehicle participating in this proposed system send their reputation profile to CC periodically for optimal decision making in vehicle selection. This reputation profile is based on the frequency of visits in different clusters.
3. Based on the optimal set of vehicles CC also calculates the optimal set of RSUs. CC finds which RSU or cluster in existing set of RSUs have maximum visits of these optimal vehicles.
4. CC forward the updated code to optimally selected RSUs.
5. All vehicles from the optimal vehicle selection set will start execution of the given task as follow
    a. The updated code can be received from RSUs. MNs will get this code whenever they will be in the wireless range of any RSU.
    b. After getting the code MNs transfer this code to required SSs wherever they are in their wireless range.
    c. After each task completion, MN will inform to CC.
6. After delay-tolerant interval D, CC will inform to MN to stop executing the tasks.

A. Problem Statement

In our proposed scheme the trajectory of MN is not adjusted for code forwarding. Therefore, it is unpredictable that all sensors receive the updated code. Hence, we need to address some concerns:

How to forward the code to maximum possible SSs. How to forward the code more quickly to SSs in a given time interval D. The problem definition is given in detail to solve these issues.

a. Code forwarding coverage rate

The quality of service of the proposed scheme depends upon the task completion rate. Code forwarding coverage rate (CFCR) is the ratio of the number of SSs who receive the updated code to the given number of SSs in a time interval T. Higher ratio will show the batter output of the proposed scheme. This ratio can be calculated by the following equation.

$$CFCR = \frac{SS_{Finish}}{SS_{Total}} \quad (1)$$

Where SS_{Total} is the given number of smart sensors need to update and SS_{finish} is the updated SS in time T.

b. Code forwarding time duration

It is a total time from start to end of the proposed scheme to forward the updated code. The completion time will reach when CFCR gives 100% coverage or it doesn't grow well after a certain time period. The system will outperform with a lower value of CFTD. It can be calculated by the following equation.

$$T_{total} = t_2 - t_1 \quad (2)$$

Where t1 is the starting time and t2 is the finishing time of the proposed scheme. To get better performance and service quality of the proposed scheme we need to maximize the CFCR and minimize the CFTD as follow.

$$CFCR = \mathbf{max}\left(\frac{SS_{Finish}}{SS_{Total}}\right) \quad (3)$$

$$CFTD = \mathbf{min}(t_2 - t_1) \quad (4)$$

B. Design of proposed scheme

1. Motivation

To recognize SC, a huge number of SSs are need to install throughout the city.

As stated earlier, the sensing requirements of different applications may change from time to time which leads to software code update in SSs. Many of the SSs in SC cannot be connected to DC clouds directly because of the high cost of infrastructure network and power resources. This is the reason, managing software update code transmission to SSs from DCs in traditional ways is challenging.

On the other hand, there exists a displacement of a huge number of mobile vehicles in SC. These vehicles travel on streets and roads almost everywhere and at any time. Many vehicles are equipped with storage, processing, and communication facilities. We can use this huge displacement of vehicles as mobile nodes to forward updated code to SSs.

The other important thing is, this displacement of vehicles have a certain similarity in trajectories in continuous periods of time. To examine the similarity, the Beijing taxi trajectory data set is used in this work. We use the following equation to measure the similarity of the vehicle displacement in the consecutive periods.

$$Similarity = \frac{|\lambda(x) \cap \lambda(y)|}{|\lambda(x) \cup \lambda(y)|} \quad (5)$$

This equation calculates the similarity between the periods x and y. Infect, this is a ratio of vehicles visited in common areas, to all areas visited by vehicles in time x and time y. We calculate the similarity of a given set of taxi trajectories, we divide the time into intervals of 24 hours each. Figure 2 shows that there exists a strong similarity in taxi displacement of taxi trajectories in an area of 25 ×25 m2.

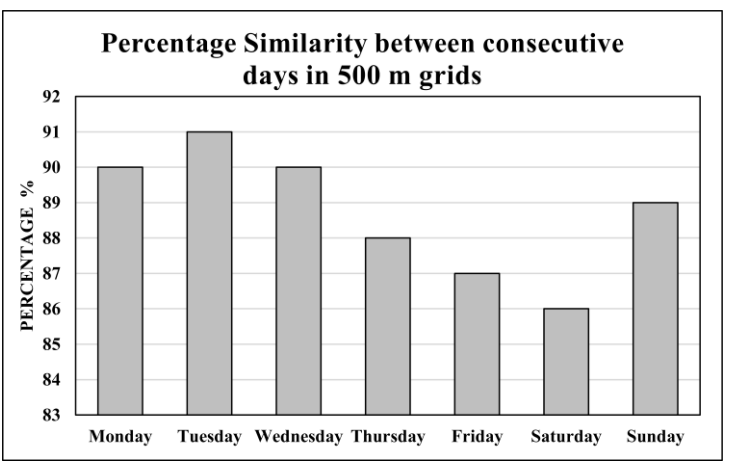

Figure 2. Similarity in consecutive days

2. Overview

We use mobile vehicles as mobile nodes MN in this proposed scheme of code

transmission to SSs in SC. This scheme manages a huge number of SSs and saves the infrastructure cost. In this scheme, SSs are scattered throughout the city at different locations to manage different sensing applications and they are installed near road networks. Although traditional cities have many sensors installed once they are installed, it is difficult to change there working status. But SSs can update themselves wirelessly after their installation and in many scenarios, it is necessary to update their working status. In many situations, environmental changes that are monitored by SSs, e.g high temperature in summer can become the reason to cause a fire. To monitor the fire cause more quickly in the summer season, the temperature of SSs can be adjusted for easy detection of fire by summer updated code. But the problem is how to transfer updated code to SSs without the internet. Our scheme uses smart vehicles as mobile nodes MN to forward this updated code to SSs. The following are the three main components used in this scheme.

1. **Mobile Node**

Mobile nodes in this work are the smart vehicles, which are extensively dispersed and frequently moving in the entire city. These vehicles are equipped with multiple wireless interfaces, storage, and processing units, so many of them can be used in this data communication between sensors and roadside units without any real modifications. We can say that the trajectory of the mobile node is not affected by our proposed model, but it depends upon the final destination of the rider is going to. This non-scheduled and non-regular communication between SSs and RSUs can be called as store carried and forward communication or opportunistic communication. Whenever a MN visits a cell area it updates its reputation parameter and profiles its reputation concerning encounters in different areas as shown in Figure 6. If MN is in transmitting mode then it sends a transmission request to SS/RSU, it transfers the data if the other device is ready to receive the data. If a device wants to send data to MN then it converts its mode to receiving mode and receive the data. It can transfer updated code from RSUs to SSs and sensed data from SSs to RSUs. To build the profile of a MN and forwarding code or data between RSUs and SSs, we use algorithm 1.

Algorithm 1: Algorithm at Mobile Node

```
while True do
    Enter Detecting Mode
    if Detect an RSU then
        R_RSU(aid) ← ++value
    else
        R_SS(aid) ← ++value
    end
    if MN in transmitting mode then
        Transmit a send request to RSU/SS
        receive a responce from RSU/SS
        if RSU/SS is ready then
            Transmit code/data and add log
            if finishing transmission then
                clear internal memory
            else
                attempt to transmit next time
            end
        else
            do nothing
        end
    else
        Switch to detecting mode
    end
    if MN in receiving mode then
        if RSU/SS is ready to send then
            MN receives the code/data packets
            if data is critical then
                upload to DC by using LTE connection
            else
                Wait for RSU/SS location
            end
        else
            do nothing
        end
    else
        Switch to detecting mode
    end
end
```

2.Smart Sensor

Smart devices and actuators installed in the infrastructure of the entire smart city are termed as the smart sensor in this work. These smart sensors are used for data collection and controlling of different applications. These sensors can be controlled and administrated by the central authority e.g by using software-defined networking. They can upgrade their IOS after receiving regular software updates from data centers and can become more intelligent. They are low powered devices, to save the cost; these devices are not directly attached to

cellular networks. These smart sensors can communicate with MNs whenever they are in the wireless range of each other, by using different wireless technologies like, IEEE 802.11n, IEEE 802.11p, IEEE 802.16 and IEEE 802.11 ay, etc. After getting the data from control centers, these sensors can update their settings. For example, sensors that are previously working for data collection of temperature can be reset to gather data for humidity in the vegetation field. The sensors controlling the working period of street lights can be reset with foggy weather conditions. These sensors can forward the collected data to RSUs by using MNs. SSs use algorithm 2 to perform its operations.

Algorithm 2: Algorithm at Smart Sensor

```
while True do
    Enter Detecting Mode
    if SS detects MNs and have data to send then
        switch to transmission mode
        if m.ttl is not expired then
            1. Broadcast request for reputation index for each MN
            2. MNs send their reputation index
            3. Find the MN with highest reputation value
            if R_Value ≥ thresholdvalue then
                transfer data packets
            else
                wait for next comming MNs
            end
        else
            do nothing
        end
    else
        do nothing
    end
    if SS detects a MN having update code from S_MN^optimal then
        1. Switch to receiving mode.
        2. Receive the code from MN.
        3. update its function by received code.
    else
        do nothing
    end
    Collect data from srounding environment.
end
```

3. Road Side Units

They are controlled by the central authority, data center cloud, or control center. Data from all the sensors of the city is collected here with the help of mobile nodes. They are also responsible for downloading software updates from CC, for different sensors and forward to them by using MNs. Moreover, these RSUs have continuous power supply and connection with the core network. RSU uses algorithm 3 to perform its functionality

Algorithm 3: Algorithm at RSU

Input: S_{SS} ← set of smart sensors, Code ← updated code from DC, $S_{MN}^{optimal}$ ← set of selected vehicle for code transmission

```
while True do
    Enter Detecting Mode
    if RSU detect a MN having data from sensors. then
        if MN send a data forwarding request then
            1. Switch to receiving mode.
            2. Receive the data from MN.
            3. Upload the data to DC.
        end
    else
        do nothing
    end
    if RSU detects a MN from S_MN^optimal for code transmission then
        1. Switch to transmission mode.
        2. Forward the received code to MN.
        3. Update DC.
    else
        do nothing
    end
    update RT_MN[tid][aid] ← ++value // update reputation value of vehicle at DC
    Switch to detecting mode
end
```

C. Code deployment

We use the RSU on the road network to transmit updated code to SSs by using MNs. All RSUs are attached to CC by using the internet. CC needs to select optimal RSUs among all, to transfer the updated codes and to maximize the coverage rate and minimize time and cost. We turned the RSU selection problem into two problems of minimum set coverage. 1) Based on historical trajectories, find the optimal set of vehicles that can cover all the SSs of the given task. 2) based on optimal vehicle selection, find a minimum set of areas, where all selected vehicles travel through at least one selected area. This problem of set coverage is a distinctive NP-hard problem and our proposed algorithms can obtain a realistic solution set. The following are two algorithms are used for this selection.

a. Optimal vehicle selection

In the motivation section, the similarity between future and past vehicle trajectories have been observed. This similarity will build the reputation of vehicles w.r.t. area visit. Therefore, the reputation of vehicles has certain implications for future trajectories that can be used for code transmission. If we have an updated code for a given set of SSs, then we can use historical data to find the optimal set of vehicles that cover all areas of SSs by using algorithm 4 and RFA(MNi) function will calculate the reputation of mobile node i with

respect to area visits, how many areas are covered by MNi in the history and number of SSs covered in the given requirement, where they are located in a given set of clusters as follows.

$$RI(MN_i) = \sum_{i=1}^{n} | M(MN_i, aid) | \quad (6)$$

Where n is total number of clusters.

$$RFA(MN_i) = \{MN_i : RI(MN_i) > RI(MN_j) \therefore for\ all\ SSs\} \quad (7)$$

For all given set of SSs the CC calculate the set of optimal vehicles by 7 with respect to maximum coverage.

Table 1. Symbols used in this scheme

Symbol	Meaning
S_{MN}	Set of Mobile Nodes
S_{SS}	Set of available Smart Sensors
S_{RSU}	Set of road side units
$Best_{MN}$	The MN covered the highest number of SSs
$S_{SS}^{bestcovered}$	Set of SSs that $Best_{MN}$ passed
$S_{MN}^{optimal}$	Set of optimally selected MNs
$S_{optimalMN}^{bestcovered}$	Set of optimal mobile nodes passed by $Best_{RSU}$
S_{Area}	Set of Areas where $S_{MN}^{optimal}$ passed
$Best_{RSU}$	The Area where the highest number of MNs passed in SArea
$RI(MN_i)$	Reputation index of mobile node MN_i,
$RI(RSU_i)$	Reputation index of cluster RSU_i,
$RFA(MN_i)$	Reputation function, can find the set of areas that the MN_i passed
$RFM(RSU_i)$	Reputation function, can find the set of MNs passed by RSU_i

Algorithm 4: Optimal set of vehicles for code forwarding

Input: $S_{SS} \leftarrow$ set of smart sensors to update, $S_{MN} \leftarrow$ set of all mobile nodes.
Output: $S_{MN}^{optimal} \leftarrow$ set of optimal vehicles

```
while S_SS is not Φ do
    Best_MN = = NULL
    S_SS^bestcovered = Φ
    for MN_i ∈ S_MN do
        S_covered^MN_i = RFA(MN_i) ⋂ S_SS
        if S_covered^MN_i > S_SS^bestcovered then
            S_SS^bestcovered = S_covered^MN_i
            Best_MN = MN_i
        end
    end
    S_MN^optimal ⊕ Best_MN
    S_SS ⊖ S_SS^bestcovered
    S_MN ⊖ Best_MN
end
return S_MN^optimal
```

b. RSU Selection.

To verify the performance of our proposed scheme, we following two more schemes for RSUs selection and compare the results with optimal RSU selection scheme. Here are the three schemes for RSU selection.

1. Optimal RSU Selection

After getting the set of optimal vehicles, we need to find the areas where the selected vehicles travel through at least one time. Algorithm 5 is used to select the optimal set of RSUs for code transmission from CC to RSUs. RFM(RSUi) function will help to get the clusters where the mobile nodes in the set of optimal vehicles visited at least one time in the history. That code will be further forwarded to SSs by using the optimal set of MNs. RI(RSUi) will calculate the reputation index of RSUi, how many vehicles passed in a given cluster. RFM(RSUi) calculate the optimal set of RSUi with respect to given set of optimal vehicles.

$$RI(RSU_i) = \sum_{i=1}^{m} | M(aid, MN_i) | \tag{8}$$

$$RFM(RSU_i) = \{RSU_i : RI(RSU_i) > RI(RSU_j) \therefore for\ all\ MNs\} \tag{9}$$

Where m is total number of vehicles. For the given set of optimal vehicles CC calculates the optimal RSUs by 9 in algorithm 5.

2. Adjacent Fixed RSU Selection.

By closely observing the heat table ?? in work 3, of vehicle trajectories, we have selected an area of adjacent cells, for RSU deployment, where the frequency of vehicle is very high.

3. Random Selection

In random selection, we have selected some areas randomly for RSU deployment that fulfil certain criteria.

Algorithm 5: b. Optimal RSU selection for code forwarding

Input: $S_{MN}^{optimal} \leftarrow Algorithm$ x, Set of Road Side Units = S_{RSU}
Output: $S_{RSU}^{optimal} \leftarrow$ set of optimal RSUs
while S_{RSU} *is not* Φ **do**
  $Best_{RSU} = \text{NULL}$
  $S_{optimalMN}^{bestcovered} = \text{NULL}$
  for RSU_i *in* S_{RSU} **do**
    $S_{covered}^{RSU_i} = RFM(RSU_i) \cap S_{MN}^{optimal}$
    if $S_{covered}^{RSU_i} > S_{optimalMN}^{bestcovered}$ **then**
      $S_{optimalMN}^{bestcovered} = S_{covered}^{RSU_i}$
      $Best_{RSU} = RSU_i$
    end
  end
  $S_{RSU}^{optimal} \oplus Best_{RSU}$
  $S_{MN}^{optimal} \ominus S_{optimalMN}^{bestcovered}$
  $S_{RSU} \ominus Best_{RSU}$
end
return $S_{RSU}^{optimal}$

Performance analysis of OCFS

a. Area Selection

In this analysis, we use 1000 m2 cell size, and there are 625 cells in our selected area. The parameters used in this analysis are shown in table 2.

To verify the performance of the proposed code transmission scheme we use the T-Drive taxi data set from the city of Beijing. For VADD contact estimation, we have selected an area of 25×25 km2 having longitude between 116.24◦ E to 116.5335◦ E and latitude between 39.8125◦ N to 40.04◦ N, as shown in figure 3 [5]. The exact measurements of all lengths between these longitudes and latitudes are calculated from a web page Moveable Type Scripts [6]. The number of vehicles visit from a set of 400 random selected vehicles is shown in figure 3, in each cell of 1000 m2.

Table 2. Parameter settings for code transmission analysis.

Parameters	**Value**
Dataset	Beijing Taxi Traces (China)
Area	25×25Km2
Area division (Cell size)	1000 m2
Number of cells	625
Wireless Technology	IEEE 802.11p
Number of GPS points reported	5,051,513
Longitude	116.24 ◦ E -116.5335◦ E
Latitude	39.8125◦ N-40.04◦ N
Number of Taxies	400
Average update time of Taxies	30 seconds
Number of smart sensors	1000,1200,1400,1600,1800,2000
Number of groups	6
Delay tolerant interval	12 hours, 24 hours
Number of repetitions for each group	5 times
Maximum execution time	1800 minutes

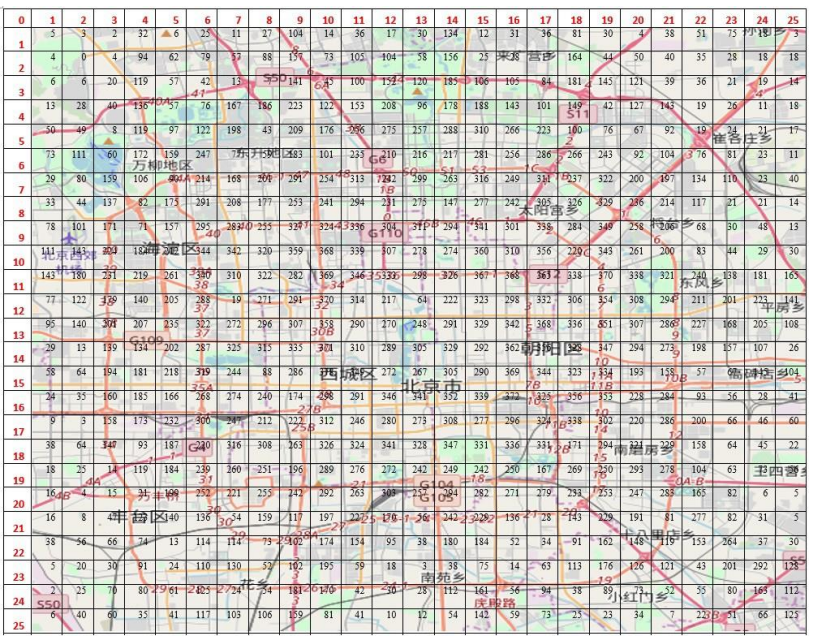

Figure 3. Number of vehicles visits in each cell of 1000 m2.

b. Data filtration

The remaining GPS locations are filtered out other than the selected area. To reduce the error, we filtered more points, deviated from the typical trajectory of each taxi. Figure 4 depicts the invalid points of the taxicab having ID 3015, with red color. The vehicles are broadcasting wrong GPS locations near the areas,

vulnerable to GPS errors, like tunnels, areas having high buildings skyscrapers, etc. For this purpose, we calculate the average speed and remove those points where speed is much greater than the average speed. After this data filtration, we purify the data set that is shown in figure 5

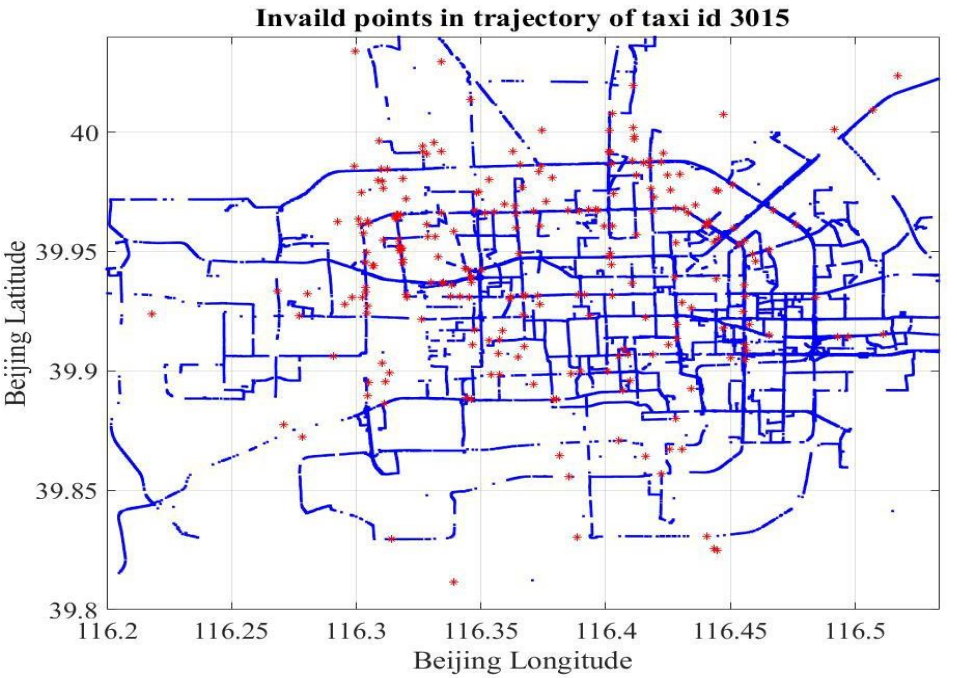

Figure 4. Trajectory with invalid points.

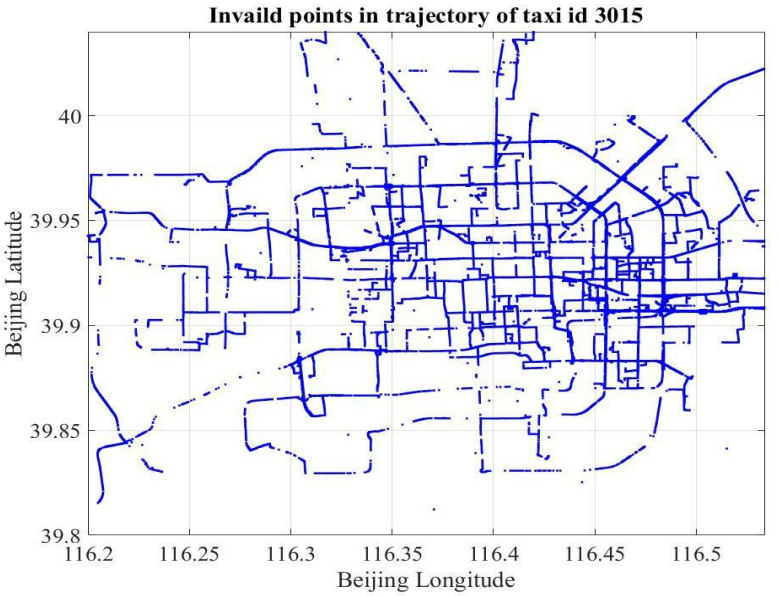

Figure 5. Trajectory with valid points.

c. Profile building

After filtration of data for each taxi, we got 5,051,513 GPS trace points. We divided the data set into two parts, 70% of the data we have used for a historical time period T1 and the rest of the trajectory data has been used for code transmission in time period T2. In T1 each taxi will build his reputation profile as shown in figure 6. Each vehicle will maintain its trajectory history with respect to area visit and periodically update to CC with its heat table. Then for each given request CC can calculate r index with respect to number of SSs covered by each vehicle. A vehicle having highest index will be selected as an optimal vehicle. The heat table shows the movement of taxi ID:6275 in all cells of 1000 m of selected area 25×25 km . Similarly, it can build its profile for 100 m , 250 m , and 500 m cell sizes. Figure 6 shows that this taxi has more visits to the down-town area as compared to the peripheral areas. Similarly, all the taxies will build their profiles.

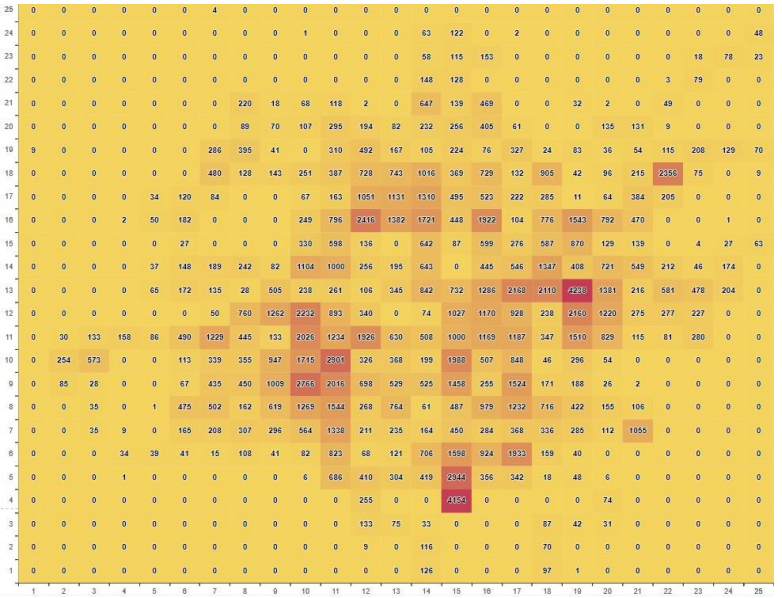

Figure 6. Reputation of taxi id 6275 in each cell of 1000 m^2.

d. Smart sensors locations

To verify the proposed scheme, we need to find the locations of SS and RSUs. At first, we selected a set of 1000 SSs and find 1000 random locations for SSs near the road network of 100 m2 cells of the selected area. To minimize the contingency of the experiment, each group of comparison experiments was performed 5 times for each set of SSs, for a total of 30 comparison experiments. The random locations of these 1000 SSs for 1st experiment is shown in figure 7. In our proposed scheme, we use algorithm 4, to find the optimal MNs which can complete the largest number of tasks to transfer the updated code to these SSs.

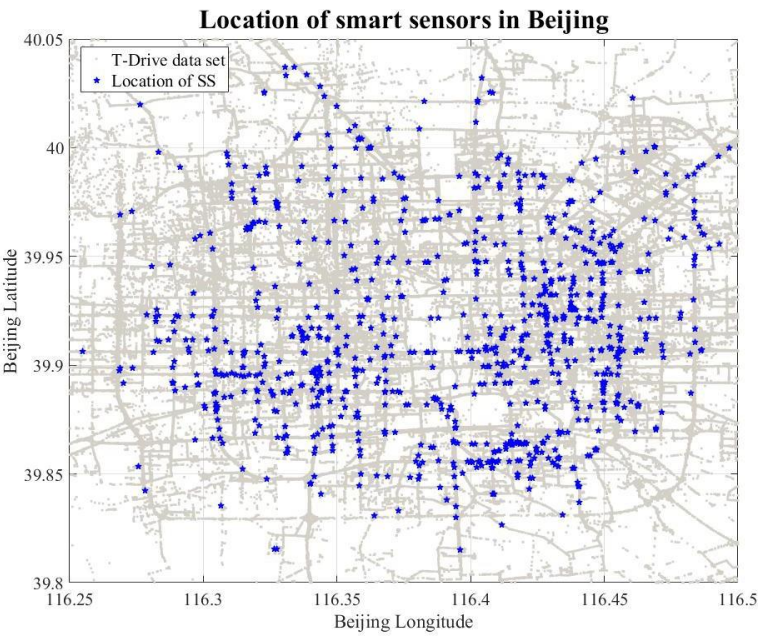

Figure 7. Location of SS in Beijing City.

e. RSU deployment

Our next task is to find the locations for RSUs. In our proposed scheme, we use algorithm 3 to find the optimal locations of RSUs and we got 36 locations of RSU to implement this task. For comparison, we use the same number of RSU in fixed RSU selection, and Random RSU selection. For fixed RSU selection we selected 36 consecutive cells in the down-town area having the high number of taxi visits and for Random RSU selection we selected 36 random cells having taxi visits greater than 100. The RSU deployment by all three schemes is shown in figure 8.

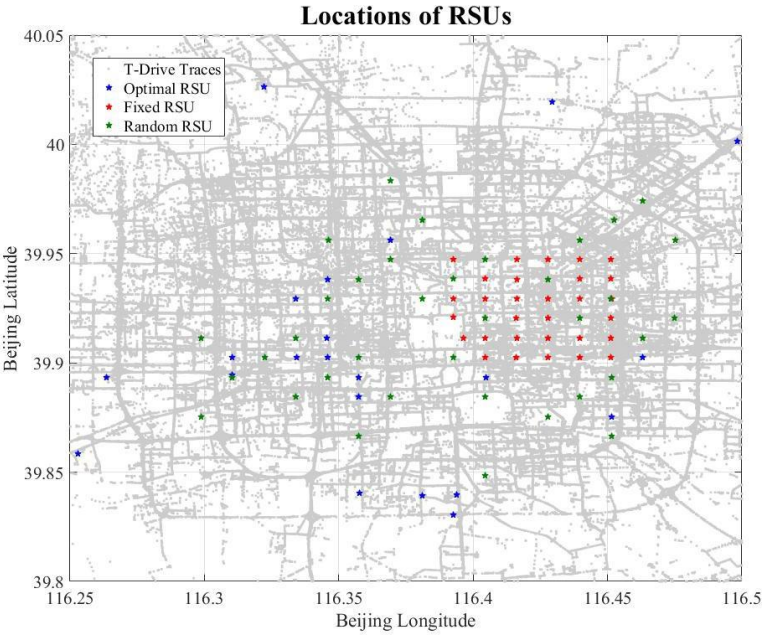

Figure 8. Location of RSU in Beijing City

Results and discussion

For the 1st experiment, we execute the scheme for a delay-tolerant interval of 12 hours and compare the CFCR for all deployments. We got the results as shown in figure 9. Our proposed scheme outperforms in all 5 cases and obtains a CFCR rate of up to 85%. The fixed deployment scheme has also obtained batter CFCR it is because these locations have the highest frequency of MNs visits. On the other hand, random deployment obtains the highest value up to 70% in all 5 experiments. In this experiment set, we increase the delay-tolerant interval equal

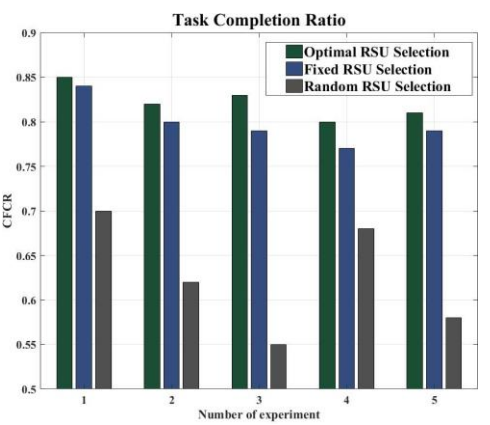

Figure 9. Delay tolerant interval = 12 hours and number of SSs = 1000.

to 24 hours and calculate the CFCR rate for all three schemes. When we increase the delay-tolerant interval

then the CFCR rate also increases in all three cases. Our proposed scheme got the highest completion rate up to 97 %, whereas fixed and random deployment got the highest completion rate up to 90% and 85% as shown in figure 10.

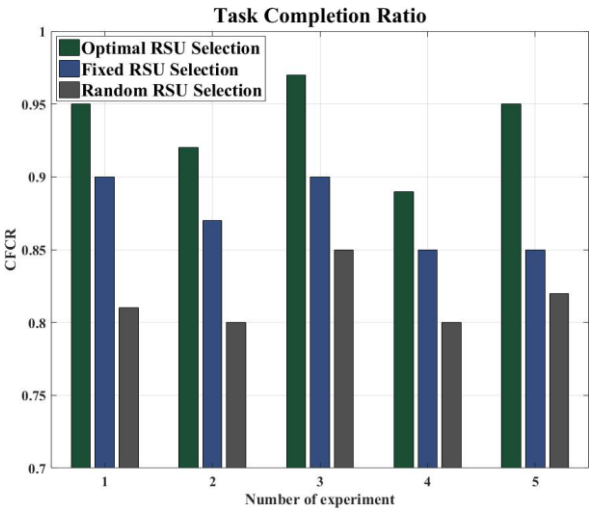

Figure 10. Delay tolerant interval = 24 hours and number of SSs = 1000.

The rest of the experiments are executed with 1200, 1400, 1600, 1800, and 2000 smart sensors. The results of the CFCR rate for all 30 experiments are shown in figure 11. our proposed scheme outperforms and got the highest completion rate as compared to fixed and random deployment.

Now we assume that there is no limit on delay-tolerant interval and execute code dissemination for all their schemes for 1000 SSs. The results obtained are shown in figure 12, it compares the code dissemination rate CFCR over CFTD in minutes. The fixed deployment scheme got results closer to our proposed scheme of RSU deployment, after 1800 minutes.

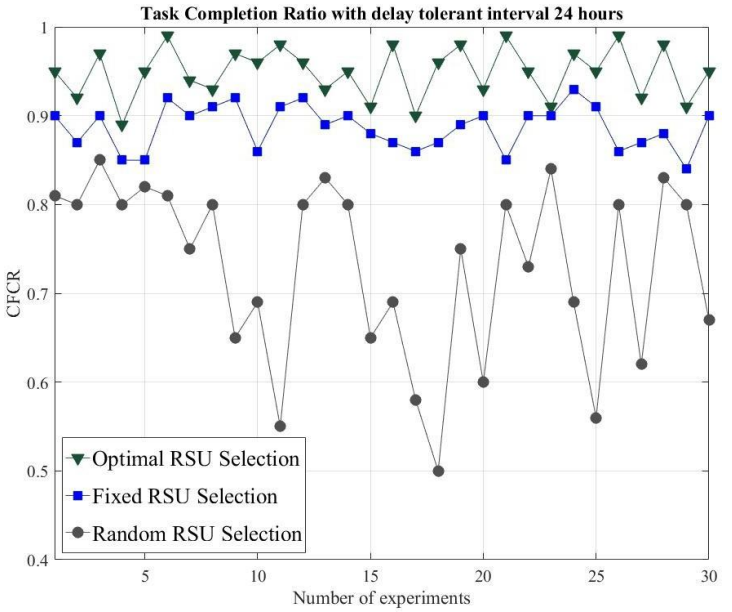

Figure 11. Task completion ratio for delay tolerant interval of 24 hours. (G1=1000 SSs→ experiment number1-5, G2=1200 SSs→ experiment number 6-10, G3=1400 SSs→ experiment number 11-15, G4=1600 SSs→ experiment number 16-20, G5=1800 SSs→ experiment number 21-25, G6=2000 SSs→ experiment number 26-30).

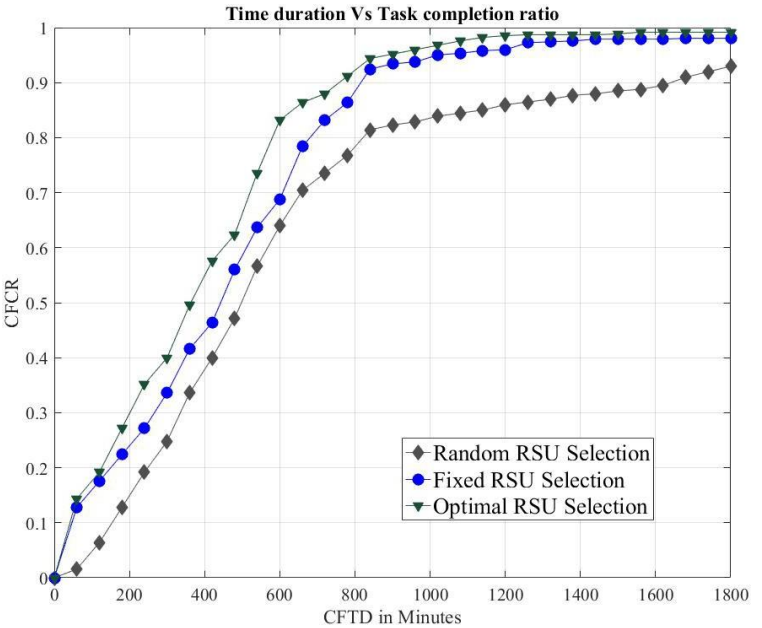

Figure 12. CFCR Vs CFTD in minutes for 400 vehicles in 1000 m^2 clusters.

Table 3 shows that our proposed code dissemination scheme obtains a 98.72% CFCR code completion rate in 1260 minutes. The figure shows that there is no change in the completion rate after this time. By observing traces it is found that a few MNs passed through these points but at that time they didn't visit any RSU location, i.e these MNs have no updated code when they visited these SSs locations. After 1800 minutes fixed and random deployment code dissemination archive 97.88% and 90% respectively. The number of sensors who didn't get the updated code is the smallest in our proposed scheme, which is 1.2% after 1260 minutes. The results in the case of fixed RSU are closer and obtain in 1800 minutes, whereas in the case of random RSU selection after the same time 10% of the sensors didn't get the code. Our proposed OCFS achieves the highest code transmission coverage rate in less time.

Table 3. Results of three RSUs deployments schemes.

RSU Selection	CFCR	CFTD (in minutes)	Un updated sensors
Optimal RSU	98.72%	1260	1.28%
Fixed RSU	97.88%	1800	2.12%
Random RSU	90%	1800	10%

Conclusion

In this paper, we proposed data and code forwarding schemes for our proposed VADD framework. To evaluate the data dissemination scheme, we develop a case scenario in Auckland city by using the real data of road traffic count and to

verify the effectiveness of the code transmission scheme we use the real traces of Beijing city.

In this work, we developed delay models for sustainable data dissemination to be used to tackle typical SC daily data traffic. As a proof-of-concept of our work, we presented Auckland City case scenario where delay tolerant data is delivered to data centers. We found that our proposed system can effectively use the daily vehicle mobility of Auckland City for enormous data transmission from data sources to data centers. The results obtained show that the proposed system can offer up to four times better data transfer rate than the dedicated core network with a data rate of 1 Gbps.

In the 2nd part of this work, we proposed an optimal code forwarding scheme OCFS. In this strategy, we used thousands of vehicles as the mobile nodes to transfer code from the data center to SSs. RSUs collect the codes from CC and forward this code to optimally selected MNs. These MNs transfer the collected code to SSs whenever they are in their wireless range in store carry forward opportunistic communication. The huge displacement of vehicles in a city has a large number of vehicles that lead to high coverage of code. OCFS is low cost because we need to install the RSUs only, the other components MNs and SSs are already available in the proposed system. In OCFS we propose multi-objective optimization of an NP-hard problem, the minimum set coverage of the optimal selection of MNs, and RSUs. This optimal solution enhances the coverage ratio and reduces the time for this code transmission. The results of the experiments validated the proposed scheme of code forwarding from the users to SSs in SC.

In this work, the data and code forwarding schemes are proposed and in the next work, we discuss the usefulness of the proposed VADD framework in terms of energy consumptions and carbon emissions. In this paper, we purposed an alternate network to collect delay tolerant data from a massive number of wireless devices in a smart city that does not required expensive infrastructure to be deployed. It is possible to offload collected data on vehicles for further delivering data at data centers in smart cities. We investigated the data dissemination potential of the proposed system in terms of network coverage and capacity. The analysis of network capacity and coverage is a function of the selected wireless technology range. We evaluated our proposed system using the microscopic movement of taxicabs and their GPS locations in Beijing. We applied the area division algorithm on big temporal-spatial data to reduce the complexity of data for analysis of the proposed system. We evaluated the vehicle density in the given area, which is further used to find the potential locations for data offloading. Results obtain have shown that relatively small subsets of taxi

fleets operating in a very large area of Beijing (25×25 Km2) can even achieve more than 90% coverage in 24 hours when the grid size is greater than 500 m2. Our proposed system can be used to collect a significant amount of data that can be offloaded to vehicles. The system can collect more than 1.4 PB data on a daily basis using the IEEE 802.11n standard. We found that the proposed system is suitable for delay-tolerant data delivery applications as it can reduce the network load further by sharing the burden of congested networks. The data dissemination case study showed that the system can perform well whenever there is a loose time of requirement.

References

Abrar, U., Yousaf, A., Jaffri, N. R., Rehman, A. U., Ahmad, A., Gardezi, A. A., . . . Choi, J.-G. (2021). Analysis of Complex Solid-Gas Flow under the Influence of Gravity through Inclined Channel and Comparison with Real-Time Dual-Sensor System. Electronics, 10(22), 2849.

Ahmad, S., Cherif, N., Naseer, S., Ijaz, U., Faouri, Y. S., Ghaffar, A., & Hussein, M. (2022). A wideband circularly polarized CPW-fed substrate integrated waveguide based antenna array for ISM band applications. Heliyon, 8(8).

Ahmad, S., Manzoor, B., Naseer, S., Ghaffar, A., & Hussein, M. (2021). A Flexible Broadband CPW-Fed Circularly Polarized Biomedical Implantable Antenna With Enhanced Axial Ratio Bandwidth.

Ali, H., Batool, K., Yousaf, M., Islam Satti, M., Naseer, S., Zahid, S., . . . Choi, J.-G. (2022). Security Hardened and Privacy Preserved Android Malware Detection Using Fuzzy Hash of Reverse Engineered Source Code. Security & Communication Networks.

Arsalan, A., Burhan, M., Naseer, S., & Rehman, R. A. (2022). Efficient Interest Packet Forwarding Solution for NDN enabled Internet of Underwater Things. Paper presented at the 2022 17th International Conference on Emerging Technologies (ICET).

Arshad, M., Karim, A., Naseer, S., Ahmad, S., Alqahtani, M., Gardezi, A. A., . . . Choi, J.-G. (2023). Detecting Android Botnet Applications Using Convolution Neural Network. Computers, Materials & Continua, 77(2).

Benkirane, S., Guezzaz, A., Azrour, M., Gardezi, A. A., Ahmad, S., Sayed, A. E., . . . Shafiq, M. (2023). Adapted Speed System in a Road Bend Situation in VANET Environment. CMC-COMPUTERS MATERIALS & CONTINUA, 74(2), 3781-3794.

Kaur, P., Nand, P., Naseer, S., Gardezi, A. A., Alassery, F., Hamam, H., . . . Shafiq, M. (2022). Ontology-Based Semantic Search Framework for Disparate Datasets. Intell. Autom. Soft Comput, 32, 1717-1728.

Khan, M., Ali, I., Khurram, S., Naseer, S., Ahmad, S., Soliman, A.-T., . . . Choi, J.-G. (2023). ETL Maturity Model for Data Warehouse Systems: A CMMI Compliant Framework. Computers, Materials \& Continua, 74(2), 3849--3863. Retrieved from http://www.techscience.com/cmc/v74n2/50194

Naseer, S. (2021). Vehicle Assisted Energy-Efficient Data Dissemination Framework. Auckland University of Technology,

Naseer, S., & Chaudhry, S. (2011). Lr-wpan formation topologies using ieee 802.15. 4. International Journal of Computer Science Issues (IJCSI), 8(6), 39.

Naseer, S., Ghafoor, M., bin Khalid Alvi, S., & ul Islam, H. S. (2022). Denial of Services (DoS) Attack: Implementation in Wireless LAN and Countermeasures. Pakistan Journal of Multidisciplinary Research, 3(2), 1-13.

Naseer, S., Ghafoor, M. M., bin Khalid Alvi, S., Zafar, I., & Murtaza, G. (2023). Wrapper Extraction and Integration using GNN. Pakistan Journal of Multidisciplinary Research, 4(1), 66-92.

Naseer, S., Liu, W., Sarkar, N. I., Chong, P. H. J., Lai, E., Ma, M., . . . Qadir, J. (2018). A sustainable marriage of telcos and transp in the era of big data: Are we ready? Paper presented at the International Conference on Smart Grid Inspired Future Technologies.

Naseer, S., Liu, W., Sarkar, N. I., Shafiq, M., & Choi, J.-G. (2021). Smart city taxi trajectory coverage and capacity evaluation model for vehicular sensor networks. Sustainability, 13(19), 10907.

Naseer, S., & Mahmood, R. (2015). Intrusion detection techniques in mobile adhoc networks: A review. Lecture Notes on Information Theory Vol, 3(1), 52-55.

Sandhu, U. A., Haider, S., Naseer, S., & Ateeb, O. U. (2011). A study of the novel approaches used in intrusion detection and prevention systems. International Journal of Information and Education Technology, 1(5), 426.

Satti, M. I., Ahmed, J., Muslim, H. S. M., Gardezi, A. A., Ahmad, S., Sayed, A. E., . . . Shafiq, M. Ontology-Based News Linking for Semantic Temporal Queries.

Velusamy, P., Rajendran, S., Mahendran, R. K., Naseer, S., Shafiq, M., & Choi, J.-G. (2021). Unmanned Aerial Vehicles (UAV) in precision agriculture: applications and challenges. Energies, 15(1), 217.

Zafar, I., Naseer, S., & Ghafoorb, M. Mobile Data offloading Techniques Using Licenced and Un-licenced Frequency Band: A Comprehensive Survey.