A Quality-Oriented Data Collection Scheme in Vehicular Sensor Networks

Wendi Nie, Victor C. S. Lee, Member, IEEE, Dusit Niyato, Fellow, IEEE, Xiaoxin Duan, Kai Liu, Member, IEEE, and Sarana Nutanong, Member, IEEE

Abstract—Considerable research attention has been dedicated to vehicular sensor networks (VSNs) because of its great potential in traffic monitoring. By taking advantage of sensors embedded in vehicles, a VSN harvests data while vehicles are traveling along the roads and then updates the collected data to the infrastructure to support the intelligent transportation system (ITS) applications. To meet the data collection requirements of different ITS applications, a huge number of update packets are generated, which may exhaust the available wireless communication bandwidth. To improve the efficiency of utilization of wireless bandwidth, in this study, we propose a quality-oriented data collection scheme, which aims to effectively support both the accuracy and real-time requirements stipulated by ITS applications while reducing communication overhead. We formulate a minimized communication overhead (MCO) problem and propose two algorithms, mixed-integer linear programming (MILP) and deviation-detection (DD), to solve the MCO problem. MILP can obtain the optimal solution by having all the data collected by every vehicle, while DD could achieve an efficient solution without this impractical assumption. We conducted extensive experiments by using SUMO to simulate vehicle traces in freeway and downtown environments. The experimental results have demonstrated the effectiveness of the proposed solutions.

Index Terms—Vehicular sensor network, intelligent transportation system, data collection, quality of information.

I. INTRODUCTION

INTELLIGENT Transportation Systems (ITS) have received increasing research interests in view of its ability to improve safety, mobility and efficiency of contemporary transportation systems [1]–[6]. The Federal Intelligent Transportation Systems program of U.S. Department [7] has provided an overview of ITS applications, e.g., Arterial Management, Freeway Management, Road Weather Management and Traffic Incident Management. In the past decade, Vehicular Sensor Networks (VSNs) have emerged as one of the most promising techniques supporting ITS applications [8]–[12]. VSN is built on top of the Vehicular Ad-hoc Network (VANET). It first makes use of different kinds of sensors embedded in vehicles to harvest data while vehicles are traveling along the roads and then updates the collected data to the infrastructure via the communication capabilities provided by VANET such as vehicle-to-vehicle and vehicle-to-infrastructure communications. Since vehicles are the principal parts of a transportation system, the data collected from vehicles are more accurate and wide-ranging than that from traditional stationary sensors [13]. However, the unique characteristics of VSN such as high mobility, varying vehicle density, dynamic network topology, limited bandwidth, unlimited battery and storage, constrained mobility patterns and speed limits of the road network, the existing data collection schemes designed for Wireless Sensor Networks (WSNs) cannot be applied directly to VSN [14]–[16]. Therefore, a data collection scheme has to be tailor-made for VSN.

In this paper, we propose a Quality-oriented Data Collection (QDC) scheme which aims to provide high quality data to ITS applications. Here, the term “quality” refers to the Quality of Information (QoI) which is widely used in sensor network [17]–[19] to evaluate the quality of data collected by data collection schemes. QoI is a set of attributes including cost, accuracy and timeliness. Among these attributes, cost is the most essential one for evaluating the efficiency of a data collection scheme. In a VSN, the dominant cost of data collection is the communication overhead, because of the limited wireless communication bandwidth. A sheer amount packets could be generated by vehicles to meet the data collection requirements of different ITS applications. For instance, a study by Hitachi has shown that there is about 25 gigabytes of data uploaded to the cloud by a vehicle every hour [20]. This large number is generated because of the following factors. First, there is a wide variety data, such as vehicle speed, traveling time, deceleration, acceleration, temperature, air quality and even acoustic data, to be collected by ITS applications of different purposes. Second, the typical scale of a VSN can be in thousands of vehicles. Third, most ITS applications need data that can reflect the current state of the world, which requires frequent updates from vehicles. The sheer amount packets may exhaust the available wireless communication bandwidth, results in network congestions and delays [21].
To reduce communication overhead, QDC makes use of spatial locality and temporal locality. First, for spatial locality, we take advantage of the fact that the data such as outdoor temperature collected by vehicles in the vicinity of each other would be similar so that only a proportion of vehicles can be selected to submit their updates. As for temporal locality, we utilize the fact that certain types of data may not change rapidly to reduce the update frequency. Therefore, by reducing the update frequency and the proportion of update vehicles, QDC could reduce the communication overhead.

The main challenge of QDC is how to reduce the communication overhead, while maintaining an accuracy level required by ITS applications. We make use the accuracy, which is generally defined as the similarity between the value observed from collected data to the real value [22], to quantify the accuracy requirement specified by ITS applications. Different ITS applications may have different accuracy requirements. For example, Congestion Level Detection [23] that categorizes the congestion into levels, e.g., slight, moderate and severe, does not have a stringent requirement on accuracy. Therefore, QDC aims to strike an optimal balance between communication overhead and accuracy to minimize communication overhead.

In a VSN, the real values, which are required to evaluate accuracy, can be obtained by collecting data from every vehicle in every sampling period. However, obtaining real values in VSNs may not be possible in practice because the vehicles which have the interested data may have left the Area of Interest (AoI) or fail to find a route to deliver the collected data to the server [24]. Furthermore, since the situation on the road changes from time to time, it is impossible to fix a certain update frequency and a certain proportion of update vehicles that can satisfy the accuracy requirement of ITS applications all the time. To address these issues, two algorithms, namely MILP and DD, are proposed for QDC to dynamically adjust the update frequency and the proportion of update vehicles to satisfy the accuracy requirements stipulated by ITS applications. MILP is able to obtain the optimal solution owing to its assumption of having real values while DD does not rely on this impractical assumption.

Measure of timeliness is another widely used factor to evaluate the quality of data collected by a data collection scheme. Timeliness can be characterized by update packets that are made available to ITS applications within certain timing constraints. Different ITS applications may have different real-time requirements. For instance, an incident detection application may need to process the collected data every few seconds in order to detect an incident in a timely manner while a congestion level detection application may only need to process the collected data every few minutes to report the congestion level. Therefore, QDC is designed to support real-time data collection and allows ITS applications to specify their own real-time requirements.

Our contributions are as follows:

i) We propose a QDC scheme which incorporates cost, accuracy and timeliness in QoI, to support quality-oriented data collection in VSN.

ii) We define a Minimized Communication Overhead (MCO) problem to formulate the QDC scheme.

iii) An MILP formulation is proposed to achieve an optimal solution to the MCO problem.

iv) We propose a practical heuristic algorithm, called the DD algorithm, to arrive at an efficient solution for the MCO problem.

v) Extensive experiments are conducted by using SUMO [25], to demonstrate the efficiency and the effectiveness of the proposed algorithms.

The rest of the paper is organized as follows. In Section II, we discuss related work. The basic principle of using QDC to collect data in VSNs is introduced in Section III. In Section IV, we define the MCO problem to formulate the QDC scheme. Two algorithms, MILP and DD are proposed in Section V. Results from our experimental studies are discussed in Section VI. Finally, we conclude our work in Section VII.

II. RELATED WORK

Data collection in a mobile environment has attracted increasing research attentions. By leveraging the information collected from mobile users, a wide range of services, such as real-time navigation and air quality monitoring, can be provided to users. Providing high quality services requires the collection of high quality data. Numerous efforts have been made to improve the quality of the collected data from different perspectives. From users’ perspective, the authors in [32]–[34] focus on motivating mobile users to provide high quality data by taking account the willingness of mobile users to participate into data sharing. The authors in [14]–[16], [26], [27] focus on collecting high quality data from mobile users from the network perspective by taking network parameters (e.g., communication overhead) into consideration.

Quality of Information (QoI) [26], [27] is a widely used term while accessing the quality of collected data in WSNs. QoI has been defined as a set of attributes including cost, accuracy and timeliness. Sachidananda et al. [26] formulate and solve a constrained optimization problem to find an optimal tradeoff between sensing accuracy and transport reliability to minimize the energy consumption in terms of the number of total retransmissions. Sensing accuracy is defined as the sampling accuracy perceived by the application. Transport reliability is defined as the success rate of one sample from one specific sampling node to reach the sink node. Viet et al. [27] formulate an integer-programming problem to minimize the communication cost in terms of energy consumption by adjusting the number of samples collected from sensors, while satisfying the user-specified QoI in terms of accuracy. The two aforementioned works focus on how to reduce the energy consumption in order to prolong the lifetime of WSNs. Furthermore, they assume that the number of sensors is fixed and the mobility of sensors is not considered.

In a VSN, vehicles are regarded as mobile sensors that collect data while traveling. Since the number of vehicles joining the VSN may change over time, the existing data collection schemes in WSN cannot be applied directly to VSN [14]–[16]. Therefore, data collection schemes must be specially designed to provide high quality data collection in VSNs.

Haddadou et al. [28] propose an Advanced Diffusion Protocol of Classified Data (ADCD). ADCD first classifies the data collected from VSN into several classes by importance. It specifies the targeted diffusion area and period of validity.
for each class. Then an adaptive broadcasting strategy is used to limit the broadcast of a packet according to the predefined class to reduce the communication overhead caused by flooding. Lee et al. [9] propose MobEyes to support VSN-based proactive urban monitoring applications. In MobEyes, every vehicle in the VSN stores a chunk which includes the sensed data in its local storage every 2 to 10 seconds. Each 2 to 10 minutes, every vehicle aggregates all the chunks during this period to obtain a summary chunk which is spread into VSN via periodic “single-hop” broadcasting for a given period of time. The authority nodes in VSNs collect all the summaries and proactively build a low-cost distributed index to serve for queries. The aforementioned works focus on reducing the communication overhead during data collection. However, the issue of the accuracy of the collected data is simply ignored.

Reducing the update frequency and the proportion of update vehicles is an effective way of decreasing the communication overhead. Hong et al. [29] propose an analytical statistical model based on the Nyquist sampling theorem to derive the bounds of vehicle’s sampling period and sample size. Ferman et al. [30] propose an analytical model which examines the relationships between key system parameters such as the traffic volume and the reporting interval to analyze how the penetration levels and the proportion of update vehicles impact the opportunity for providing timely and flexible data. These works have provided theoretical analyses of the update frequency and the proportion of update vehicles. However, these analyses are based on the assumption of several pre-determined crucial parameters and hence are difficult to be applied in the time-varying environments arising in the real situations.

Timeliness is also a widely-used factor in evaluating data quality. Baiocchi et al. [11] propose two sampling protocols, the SAmpled Measurement Estimation (SAME) protocol and the Timer-based Ordered Measurement Estimation (TOME) protocol, to collect real-time vehicular traffic measurement. SAME is able to collect traffic data from vehicles moving along the road between two Road Side Units (RSUs). For each time interval, the RSU at one end of the road issues a Measurement Collection (MC) message. The MC message is passed over from one vehicle to another vehicle until it reaches the RSU at the other end of the road. At each hop, the relay vehicle adds its own traffic data including coordinates, direction and velocity to the MC message. In TOME, an RSU triggers a round of traffic data collection from vehicles every period of time with an MC message. Instead of taking a sample each hop in SAME, each vehicle will schedule a message copy including position and speed. The two data collection protocols could provide real-time traffic data collection. However, communication overhead and accuracy are not considered.

The problem of reducing the communication overhead as well as that of providing an accurate data collection is considered by Hu et al. [31]. Specifically, they propose an adaptive approach that can dynamically adjust the reporting rates of vehicles to balance monitoring accuracy and message overhead in a VSN. However, this approach is designed specifically for the use of monitoring the concentration of CO2 and does not consider the proportion of update vehicles adjustment. In this work, we propose the QDC scheme to support quality-oriented data collection in VSN. QDC considers three QoI attributes namely cost, timeliness and accuracy and aims to provide high quality data to ITS applications.

III. THE QUALITY-ORIENTED DATA COLLECTION SCHEME

In this section, we introduce the QDC scheme and describe its role in a VSN as depicted in Fig. 1. The procedure of the QDC scheme can be divided into the following three steps: initiation step, control packet dissemination step, and data collection step. In Fig. 1, these are denoted by dashed line arrows, dotted line arrows and solid line arrows, respectively. When QDC begins, the input parameters of QDC are initialized by ITS applications in the initiation step. Then time is divided into signal alive periods, each of which is a period of time (e.g., 5 seconds) corresponding to the real-time requirement of the ITS application. In each signal alive period, QDC disseminates a control packet to vehicles via the infrastructure. Upon receiving the control packet, selected vehicles send the respective collected data in update packets to the central server and QDC via the infrastructure. The control packet dissemination step and the data collection step are repeated in every signal alive period. The details of these three steps are given below.

**Initiation step:** Consider an ITS application running on a central server and attempting to collect data from an AoI. As shown in Fig. 1, to capitalize on QDC, the ITS application has to input a set of parameters to the QDC running at the same server. The set of parameters includes the real-time requirement, the accuracy requirement, the aggregation function, and the road segments that the ITS application needs to monitor in the AoI. QDC makes use of signal alive period to satisfy the real-time requirement specified by the ITS application. In every signal alive period, there should be at least one update packet received from vehicles on every road segment. The accuracy is defined as the similarity between the observed value and the real value. The aggregation function is applied to the collected data to obtain observed values and real values. Formal definitions of these parameters will be given in Section IV.

**Control packet dissemination step:** In this step, QDC disseminates one control packet, which contains the update frequency and the proportion of update vehicles of every road segment, to the vehicles in the VSN of AoI. Here an update vehicle is a vehicle that is selected to send its collected data to the central server.
at a certain frequency. To reduce the communication overhead while meeting the accuracy requirement, QDC adjusts the update frequency and the proportion of update vehicles for each road segment in each signal alive period. In the first signal alive period, the update frequency and the proportion of update vehicles on every road segment are set at the sampling frequency, which is the maximum possible frequency, and 100%, respectively, so as to instruct every vehicle to send one update packet to the server in every sampling period. In subsequent signal alive periods, QDC determines the update frequency and the proportion of update vehicles for every road segment based on the data received in the previous signal alive period. Specifically, QDC obtains the observed value of every road segment by applying the aggregation function to the data received in the previous signal alive period. Based on whether the observed value meets the accuracy requirement, QDC adjusts the update frequency and the proportion of update vehicles on every road segment. Next, the update frequency and the proportion of update vehicles on every road segment are packed into a control packet. The control packet is finally delivered to the infrastructure for subsequent diffusion into VSN via the downlink broadcast channel.

Data collection step: In this step, vehicles deliver the collected data to the ITS application and QDC according to the update frequency and the proportion of update vehicles specified in the control packet. The number of update vehicles chosen has to comply with the proportion of update vehicles specified in the control packet on every road segment. And every update vehicle sets its update frequency at the value specified in the control packet according to the road segment. We assume that each vehicle is equipped with a GPS device and can readily identify the road segment in which it is currently residing. One way to select update vehicles is to take advantage of cluster-based dissemination protocols [35]–[38], which are commonly used in VANETs. In cluster-based dissemination protocols, vehicles are divided into clusters. In each cluster, there is a cluster head which acts as a gateway and can communicate with every cluster member in its cluster. Thus the cluster head is able to select update vehicles in its cluster according to the proportion of update vehicles. Next, every update vehicle delivers one update packet in accordance with its update frequency, which stores the latest collected data, the time when this data was collected and the location where this data was collected, to the infrastructure via VSN. Then the update packets received from update vehicles are delivered to the ITS application via the infrastructure. At the same time, one replica of every update packet is sent to QDC for adjusting the update frequency and the proportion of update vehicles for the next signal alive period.

From the above description, the communication overhead of QDC is that one control packet is required to be diffused into VSN to direct the update activities of vehicles in every signal alive period. Compared with the number of update packets reduced by QDC, this overhead is negligible. In addition, an efficient broadcast strategy could be used to further reduce the overhead of control packets diffusion (e.g., one-hop broadcast) [39], [40]. As can be seen, the objective of QDC is to minimize the communication overhead while maintaining the accuracy in every signal alive period. To achieve this objective, we define an MCO problem for the QDC scheme in the next section.

IV. THE MINIMIZED COMMUNICATION OVERHEAD PROBLEM

In this section, we formally define our MCO problem as part of the QDC scheme. The MCO problem aims to minimize the communication overhead while maintaining the accuracy in every signal alive period as described in Section III. To facilitate the understanding of the MCO problem, definitions and the mathematical constraints of the frequently used notations are given in Table II.

Let $T_a$ denote the sampling period of a sensor and $T_a$ denote the signal alive period. Therefore, the number of sampling periods in one signal alive period $T_a$ is

$$K = \left\lfloor \frac{T_a}{T_s} \right\rfloor.$$  

(1)

Suppose there are $N$ vehicles in total in the VSN of AoI. Each vehicle is denoted as $n$, where $1 \leq n \leq N$. The data collected by vehicle $n$ at the $k$-th sampling period in one signal alive period $T_s$ is represented by $V_{k,n}$, where $1 \leq k \leq K$. There are $E$ road segments in the AoI. Each road segment is denoted as $e$, where $1 \leq e \leq E$. We assume that each vehicle is equipped with a GPS device, so each vehicle can easily know on which road segment it is currently located. We introduce a binary variable $L_{k,n,e}$ to indicate whether vehicle $n$ is located in road segment $e$ at the $k$-th sampling period, $\forall k, k \in [1, \ldots, K], \forall n, n \in [1, \ldots, N]$.
TABLE II
NOTATIONS AND DEFINITIONS

<table>
<thead>
<tr>
<th>Notations</th>
<th>Definitions</th>
</tr>
</thead>
<tbody>
<tr>
<td>$T_a$</td>
<td>signal alive period</td>
</tr>
<tr>
<td>$T_s$</td>
<td>a fixed sampling period</td>
</tr>
<tr>
<td>Accu</td>
<td>accuracy requirement</td>
</tr>
<tr>
<td>$K$</td>
<td>the number of sampling periods in one signal alive period</td>
</tr>
<tr>
<td>$k$</td>
<td>the $k$-th sampling period in one $T_a$, $1 \leq k \leq K$</td>
</tr>
<tr>
<td>$N$</td>
<td>the number of vehicles in the VSN of AoI</td>
</tr>
<tr>
<td>$n$</td>
<td>the id of a vehicle, $1 \leq n \leq N$</td>
</tr>
<tr>
<td>$V_{k,n}$</td>
<td>the data collected by vehicle $n$ at the $k$-th sampling period in one signal alive period</td>
</tr>
<tr>
<td>$B$</td>
<td>the number of road segments in the AoI</td>
</tr>
<tr>
<td>$e$</td>
<td>the id of a road segment, $1 \leq e \leq E$</td>
</tr>
<tr>
<td>$L_{k,n,e}$</td>
<td>a binary variable, which indicates whether vehicle $n$ is located in road segment $e$ at the $k$-th sampling period</td>
</tr>
<tr>
<td>$\alpha_e$</td>
<td>the update frequency of road segment $e$</td>
</tr>
<tr>
<td>$\beta_e$</td>
<td>the proportion of update vehicles on road segment $e$</td>
</tr>
<tr>
<td>$q_e$</td>
<td>the update period during which an update packet is sent to the infrastructure by every update vehicle on segment $e$</td>
</tr>
<tr>
<td>$u_e$</td>
<td>the number of sampling periods in an update period of road segment $e$</td>
</tr>
<tr>
<td>$C_{k,n}$</td>
<td>the communication overhead of vehicle $n$ to deliver an update packet to the infrastructure at the $k$-th sampling period</td>
</tr>
<tr>
<td>$c_{\text{total}}$</td>
<td>the total communication overhead in one signal alive period</td>
</tr>
<tr>
<td>$b_{k,n}$</td>
<td>a binary variable, which represents whether vehicle $n$ sends an update packet at the $k$-th sampling period</td>
</tr>
<tr>
<td>$v_e$</td>
<td>the observed value on road segment $e$</td>
</tr>
<tr>
<td>$EX_e$</td>
<td>the real value obtained by every vehicle updated its collected data to the infrastructure with a maximum update frequency</td>
</tr>
<tr>
<td>ValueRange</td>
<td>the range of value of the data to be collected</td>
</tr>
</tbody>
</table>

and $\forall e, e \in [1, \ldots, E]$,

$$L_{k,n,e} = \begin{cases} 1 & \text{if vehicle } n \text{ is located in road segment } e \text{ at the } k\text{-th sampling period}, \\ 0 & \text{otherwise}. \end{cases}$$

If vehicle $n$ is selected to be an update vehicle, it packs the latest collected data $V_{k,n}$, the time when this data was collected and the location where this data was collected into an update packet and sends it to the infrastructure.

Let $\alpha_e$ and $\beta_e$ denote the update frequency and the proportion of update vehicles located in road segment $e$, respectively. In every $T_a$, QDC packs $\alpha_e$ and $\beta_e$ for all road segments into a control packet. As shown in Fig. 1, the control packet is first delivered to the infrastructure, and then diffused into the VSN. Our objective is to determine $\alpha_e$ and $\beta_e$ for each road segment $e$, which can minimize the communication overhead in every $T_a$ while maintaining the accuracy specified by the ITS application.

Let $C_{k,n}$ denote the communication overhead of vehicle $n$ in delivering an update packet to the infrastructure at the $k$-th sampling period. For simplicity, in this work, communication overhead is measured in terms of the number of update packets delivered to the infrastructure. The total communication overhead $c_{\text{total}}$ in one $T_a$ for all road segments is given by

$$c_{\text{total}} = \sum_{k=1}^{K} \sum_{n=1}^{N} (C_{k,n} \cdot b_{k,n}),$$

where $b_{k,n}$ is a binary variable, which represents whether vehicle $n$ sends an update packet at the $k$-th sampling period, $\forall k, k \in [1, \ldots, K], \forall n, n \in [1, \ldots, N]$.

$$b_{k,n} = \begin{cases} 1 & \text{if vehicle } n \text{ sends an update packet to the infrastructure at the } k\text{-th sampling period, otherwise}. \end{cases}$$

To satisfy the real-time requirement during each $T_a$, there should be at least one update packet received from vehicles on every road segment. Therefore, $\forall e, e \in [1, \ldots, E]$, the inequality

$$\sum_{k=1}^{K} \sum_{n=1}^{N} (b_{k,n} \cdot L_{k,n,e}) \geq 1$$

should be satisfied.

Let $q_e$ denote the update period, during which an update packet should be sent to the infrastructure by every update vehicle on the road segment $e$. $q_e$ can be calculated by

$$q_e = \frac{1}{\alpha_e}.$$  \hspace{1cm} (6)

The number of sampling periods in an update period $q_e$ is denoted as $u_e$, which is given by

$$u_e = \frac{q_e}{T_s}.$$  \hspace{1cm} (7)

Since the update period $q_e$ cannot be larger than the signal alive period, $u_e$ should satisfy the inequality, $\forall e, e \in [1, \ldots, E]$,

$$1 \leq u_e \leq K.$$  \hspace{1cm} (8)

The proportion of update vehicles in every update period $q_e$ is denoted as $\beta_e$, $\forall e, e \in [1, \ldots, E]$,

$$0 \leq \beta_e \leq 1.$$  \hspace{1cm} (9)

Since vehicles are moving, it is possible that the number of vehicles in every sampling period on a road segment are changing. For simplicity, we use the average number of vehicles in a sampling period to calculate the proportion of update vehicles in every update period. In each update period, the proportion of update vehicles on the road segment $e$ should be $\beta_e$ specified in the control packet, which is given by

$$\frac{\sum_{n=1}^{N} \sum_{i=1}^{u_e} (b_{k,n} \cdot L_{k,n,e})}{\sum_{n=1}^{N} \sum_{i=1}^{u_e} L_{k,n,e}} / u_e = \beta_e,$$  \hspace{1cm} (10)

where $i$ is the index of the update period in one $T_a$, and $1 \leq i \leq \lceil T_a / \alpha_e \rceil$.

After receiving the update packets from vehicles in the current $T_a$, an aggregation function (e.g., $Max$, $Min$ and $Average$), which is specified by the ITS application, is used to obtain the observed value $v_e$. The observed value $v_e$ is obtained by aggregating all the collected data received on the segment $e$ in the current signal alive period. For instance, when the aggregation function is $Average$, for each segment $e$, $v_e$ can be
calculated by
\[
v_e = \frac{\sum_{n=1}^{N} \sum_{k=1}^{K} (V_{k,n} \cdot b_{k,n} \cdot L_{k,n,e})}{\sum_{n=1}^{N} \sum_{k=1}^{K} (b_{k,n} \cdot L_{k,n,e})}. \tag{11}
\]
QDC uses observed values to decide whether the accuracy requirement \textit{Accu} specified by the ITS application is satisfied.

Let \(EX_e\) denote the real value which can be obtained when every vehicle updates its collected data to the infrastructure with a maximum update frequency (every sampling period). Consider the aggregation function \textit{Average}, for each segment \(e\), \(EX_e\), which could be calculated by
\[
EX_e = \frac{\sum_{n=1}^{N} \sum_{k=1}^{K} (V_{k,n} \cdot L_{k,n,e})}{\sum_{n=1}^{N} \sum_{k=1}^{K} (b_{k,n} \cdot L_{k,n,e})}. \tag{12}
\]
Note that assuming the availability of \(EX_e\) is for the sake of MILP to derive the optimal solution to the MCO problem. In fact, this assumption is unrealistic and is not required in our practical algorithm DD to be presented in a later section.

To satisfy the accuracy requirement \(\textit{Accu}\), for each segment \(e\), the inequality
\[
|EX_e - v_e| \leq \textit{ValueRange} \cdot (1 - \textit{Accu}) \tag{13}
\]
should be satisfied. \(\textit{ValueRange}\) is the range of value of the data to be collected from AoI. For instance, if the valid range of outdoor temperature is \((-40, +40)\), i.e., \(\textit{ValueRange} = 80\), and the accuracy requirement of the ITS application is 95%, i.e., \(\textit{Accu} = 0.95\), the maximum deviation of the observed value from the real value has to be 4 degree to satisfy the accuracy requirement.

Considering all the above settings and assumptions, we define the Minimized Communication Overhead Problem as follows: \textit{Given an Area of Interest including \(E\) road segments, the signal alive period \(T_{a}\), the accuracy requirement \(\textit{Accu}\) and an aggregation function, decide the update frequency \(\alpha\) and the proportion of update vehicles \(\beta\) for each road segment \(e\), such that the communication overhead used to collect data is minimized.} Formally, our optimization model can be expressed as follows:

\[
\min \quad \text{Equation (3)}
\]
\[
\text{subject to} \quad \text{Equations (1), (2), (4) – (13)}
\]

V. METHODOLOGY

In this section, we first propose an MILP formulation of the MCO problem to derive the optimal solution. Then we introduce a practical heuristic algorithm, the DD algorithm, to obtain an efficient solution for the MCO problem.

A. The Mixed-Integer Linear Programming Formulation of the MCO Problem

In this section, we introduce the MILP formulation of the MCO problem. MILP is a mathematical programming, that is able to solve a wide range of optimization problems [41]–[43]. An MILP formulation has the following restrictions:
- some of the variables must take integer values;
- the objective function and the constraints must be linear.

First, we give all the variables and the objective function of the MILP formulation. Then, we transfer all the non-linear constraints of the MCO problem to be linear constraints. Finally, the commercial tool called Gurobi [44] is used to solve the MILP and obtains the optimal solution to the MCO problem.

The variables of the MILP formulation are as follows:
- input variables:
  - accuracy: \(\textit{Accu}\);
  - the number of sampling periods in one \(T_{a}\): \(K\);
  - aggregation function: \textit{Average};
  - number of segments: \(E\);
  - number of vehicles: \(N\);
  - the value of data collected by vehicle \(n\) at the \(k\)-th sampling period: \(V_{k,n}\);
  - communication overhead: \(C_{e,n}\);
  - the real value of segment \(e\): \(EX_e\);
  - the real value of segment \(e\): \(v_e\);
- auxiliary variables:
  - \(i_{k,n,e}^1, i_{k,n,e}^2, i_{j,k,e}^3, i_{j,k,n,e}^4, i_{j,k,n,e}^5, i_{j,k,n,e}^6, p_{j,k,e}, u_e, b_{k,n}\);
- output variables:
  - the number of sampling periods in each update period on segment \(e\): \(u_e\);
  - the proportion of update vehicles on segment \(e\): \(\beta_e\).

All the input variables are given. To obtain an optimal solution to the MCO problem, it is assumed that MILP knows the data collected by every vehicle in every sampling period. Because of the linearity required by MILP, only aggregation function \textit{Average} is supported. Therefore, the value of \(EX_e\) is an input variable which could be computed by (12). The auxiliary variables are used to form the MILP formulation. \(i_{k,n,e}^1, i_{k,n,e}^2, i_{j,k,e}^3, i_{j,k,n,e}^4, i_{j,k,n,e}^5, i_{j,k,n,e}^6, p_{j,k,e}\) and \(u_{e}\) are introduced to avoid the non-linearities, where \(i_{j,k,n,e}^1, i_{j,k,n,e}^2, i_{j,k,n,e}^3, i_{j,k,n,e}^4\) are binary variables. For instance, \(i_{k,n,e}^1\) is introduced to avoid the non-linearities in (11). The output variables \(u_e\) and \(\beta_e\) are the solution to the MCO problem.

The objective of the MILP formulation is

\[
\min \quad \text{Equation (3)}
\]
subject to the constraints listed in Table III, where \(G\) is a large constant (e.g., 9999). Comparing with the MCO problem which is subject to (1), (2), (4)–(13), the MILP formulation is only subject to (5), (8)–(11) and (13). First, (1), (2), (4) and (12) are already included in the variables of the MILP formulations. In addition, we use \(u_e\) to substitute for \(\alpha_e\) as the solution of the MILP formulation and \(\alpha_e\) can be obtained directly from \(u_e\) by (6) and (7). Therefore, (6) and (7) are not needed in the MILP formulation of the MCO problem.

Obviously, (10) and (11) are non-linear. In order to obtain the MILP formulation of the MCO problem, we need to transform (10) and (11) into linear constraints.

We first introduce how to transform (11) to be linear. In Table III, it could be found that the constraints C5.1 to C5.3 are transformed from (11). Intuitively, (11) can be transformed as
\[
\sum_{n=1}^{N} \sum_{k=1}^{K} (v_e \cdot b_{k,n} \cdot L_{k,n,e}) = \sum_{n=1}^{N} \sum_{k=1}^{K} (V_{k,n} \cdot b_{k,n} \cdot L_{k,n,e}). \tag{14}
\]
TABLE III
EQUATION AND THE CORRESPONDING CONSTRAINTS OF THE MIXED-INTEGER LINEAR PROGRAMMING

<table>
<thead>
<tr>
<th>Equation</th>
<th>Corresponding constraints</th>
</tr>
</thead>
<tbody>
<tr>
<td>Equation 5: $\sum_{k=1}^{N} \sum_{n=1}^{K} (b_{k,n} \cdot L_{k,n,e}) \geq 1$</td>
<td>$C1: \sum_{k=1}^{N} \sum_{n=1}^{K} (b_{k,n} \cdot L_{k,n,e}) \geq 1$;</td>
</tr>
<tr>
<td>Equation 8: $1 \leq v_e \leq K$</td>
<td>$C2: v_e \leq K$;</td>
</tr>
<tr>
<td>Equation 9: $0 \leq \beta_e \leq 1$</td>
<td>$C3: 0 \leq \beta_e \leq 1$;</td>
</tr>
<tr>
<td>Equation 10: $\sum_{k=1}^{N} \sum_{n=1}^{K} (l_{k,n,e}) = \beta_{e}$</td>
<td>$C4.1: \frac{n_{k,n,e}}{\beta_e} \leq i_{j,k,n,e} \leq \frac{n_{k,n,e}}{\beta_e} + 1$;</td>
</tr>
<tr>
<td></td>
<td>$C4.2: f \cdot (k-1) \geq \beta_e \cdot (j-1)$, then $\frac{n_{k,n,e}}{\beta_e} \leq i_{j,k,n,e} \leq \frac{n_{k,n,e}}{\beta_e} + 1$, else $\frac{n_{k,n,e}}{\beta_e} \leq i_{j,k,n,e} \leq \frac{n_{k,n,e}}{\beta_e} + 1$;</td>
</tr>
<tr>
<td></td>
<td>$C4.3: \frac{n_{k,n,e}}{\beta_e} \leq i_{j,k,n,e} \leq \frac{n_{k,n,e}}{\beta_e} + 1$:</td>
</tr>
<tr>
<td></td>
<td>$C4.4: \delta \leq \frac{n_{k,n,e}}{\beta_e} \leq \delta$;</td>
</tr>
<tr>
<td></td>
<td>$C4.5: \beta_e + (p_{j,k,e} - 1) \cdot G \leq \delta_{j,k,n,e} \leq p_{j,k,e} \cdot G$;</td>
</tr>
<tr>
<td></td>
<td>$C4.6: \delta_{j,k,n,e} \leq \frac{n_{k,n,e}}{\beta_e} \leq \frac{n_{k,n,e}}{\beta_e} + 1$;</td>
</tr>
<tr>
<td>Equation 11: $v_e = \sum_{k=1}^{N} \sum_{n=1}^{K} (V_{k,n} \cdot L_{k,n,e}) / \sum_{k=1}^{N} \sum_{n=1}^{K} (b_{k,n} \cdot L_{k,n,e})$</td>
<td>$C5.1: 0 \leq i_{j,k,n,e} \leq v_e$;</td>
</tr>
<tr>
<td></td>
<td>$C5.2: v_e \leq \frac{1}{i_{j,k,n,e}} \cdot \leq b_{k,n} \cdot G$;</td>
</tr>
<tr>
<td></td>
<td>$C5.3: \sum_{n=1}^{K} \sum_{l=1}^{E} (i_{j,k,n,e} \cdot \cdot L_{k,n,e} - \sum_{n=1}^{K} \sum_{l=1}^{E} (V_{k,n} \cdot b_{k,n} \cdot \cdot L_{k,n,e})$;</td>
</tr>
<tr>
<td>Equation 13: $</td>
<td>EX_e - v_e</td>
</tr>
</tbody>
</table>

It is observed that the non-linearity in (14) results from the multiplication of two variables $v_e$ and $b_{k,n}$. We introduce a variable $i_{k,n,e}$, which is given by

$$i_{k,n,e} = v_e \cdot b_{k,n}.$$  \hspace{1cm} (15)

Since $b_{k,n}$ is a binary variable, if $b_{k,n}$ equals 1, $i_{k,n,e}$ is $v_e$, otherwise $i_{k,n,e}$ is 0. We introduce the constraint C5.1 and the constraint C5.2 to formulate this relationship. If $b_{k,n}$ equals 1, $i_{k,n,e}$ is $v_e$ since it should satisfy the constraints

$$\begin{cases}
0 \leq i_{k,n,e} \leq v_e \\
v_e + (b_{k,n} - 1) \cdot G \leq i_{k,n,e} \leq b_{k,n} \cdot G.
\end{cases}$$  \hspace{1cm} (16)

Similarly, it can be inferred that $i_{k,n,e}$ is 0 when $b_{k,n}$ equals 0. Therefore, (11) can be transformed into the constraint C5.3.

The constraints C4.1 to C4.11 correspond to (10). Equation 10 is a periodic function, and it requires that in each update period, the proportion of update vehicles on segment $e$ should be $\beta_e$. Let $p_{j,k,e}$ denote the periodicity, $\forall j, \in [1, \ldots, K], \forall k, k \in [1, \ldots, K]$ and $\forall e, e \in [1, \ldots, E]$, we assume

$$p_{j,k,e} = \begin{cases}
1 & \text{if } 1 + (j - 1) \cdot u_e \leq k \leq j \cdot u_e \\
0 & \text{otherwise},
\end{cases}$$  \hspace{1cm} (17)

We introduce constraints C4.1 to C4.3 to formulate the relationship in (17), the detailed explanation of the transformation is given in Appendix A.

Then $\forall j, \in [1, \ldots, K]$ and $\forall e, e \in [1, \ldots, E]$, (10) can be transformed to

$$\sum_{n=1}^{N} \sum_{k=1}^{K} (\beta_e \cdot L_{k,n,e} \cdot p_{j,k,e}) = \sum_{n=1}^{N} \sum_{k=1}^{K} (b_{k,n} \cdot L_{k,n,e} \cdot p_{j,k,e} \cdot u_e).$$  \hspace{1cm} (18)

It is observed that the non-linearity of (18) results from the multiplication of $\beta_e \cdot p_{j,k,e}$ and $b_{k,n} \cdot p_{j,k,e} \cdot u_e$. Since $p_{j,k,e}$ and $b_{k,n}$ are binary variables, by using the same method described above, we introduce auxiliary variables $i_{j,k,n,e}^4$, $i_{j,k,n,e}^5$, $i_{j,k,n,e}^6$ and constraints C4.4 to C4.11 to transform (18) into linear constraints.

Although MILP could achieve an optimal solution to the MCO problem, it is impractical because of the following reasons.

1) In real-world environments, it is impossible to have the data collected by every vehicle in every sampling period;
2) The complexity of MILP is non-polynomial;
3) MILP only supports aggregation function which could be transformed into linear formulation (e.g., Average).

Therefore, to address the above problems, we propose a practical heuristic algorithm, the DD algorithm.

B. The Deviation-Detection Algorithm

In this section, we propose the DD algorithm. Unlike MILP, DD can be applied to any aggregation function (e.g., Max) to obtain an observed value from collected data, according to the specification of ITS applications. Moreover, it does not rely on the availability of real values. To exploit spatial locality, DD employs the standard deviation of the data collected on the same road segment in the same $T_a$ for computing the proportion of update vehicles $\beta_e$. The standard deviation could measure the...
Algorithm 1: Deviation-Detection Algorithm.

Input: Accu, K, aggregation function, E, ValueRange, stdRange.  
Output: u_e and β_e.
1: reportInterval ← 0  
2: for each e in E do  
3:     list U_e ← NULL  
4:     list BETA_e ← NULL  
5:     list V_e ← NULL  
6:     array preReportInterval[e] ← 0  
7: end for  
8: for every T_a do  
9:     reportInterval ← reportInterval+1  
10: for each e in E do  
11:     if no update packets on segment e then  
12:         v_e ← -1  
13:     else  
14:         obtain v_e and stdValue from collected data in the update packets received on segment e  
15:     end if  
16:     u_e ← computeU(Accu, K, V_e, U_e, reportInterval, preReportInterval[e], v_e, ValueRange)  
17:     β_e ← computeBETA(v_e, stdValue, stdRange, Accu)  
18:     U_e[reportInterval] ← u_e  
19:     BETA_e[reportInterval] ← β_e  
20:     V_e[reportInterval] ← v_e  
21:     if v_e != -1 then  
22:         preReportInterval[e] ← reportInterval  
23:     end if  
24: end for  
25: end for

Algorithm 2: ComputeU algorithm.

Input: Accu, K, list V_e, list U_e, reportInterval, preReportIntervalId, v_e, ValueRange.  
Output: u_e.  
1: if reportInterval == 1 or (v_e = -1 and preReportIntervalId == 0) then  
2:     return 1  
3: end if  
4: if v_e = -1 then  
5:     return U_e[preReportIntervalId]  
6: else  
7:     deviation = |V_e[preReportIntervalId] - v_e|  
8: end if  
9: if deviation ≤ ValueRange · (1 - Accu) then  
10:     if deviation ≤ 1  
11:         if U_e[preReportIntervalId] < K then  
12:             return U_e[preReportIntervalId]+1  
13:         end if  
14:     else if 1  
15:         return U_e[preReportIntervalId]  
16:     end if  
17: else if ValueRange · (1 - Accu) < deviation then  
18:     if deviation ≤ 1  
19:         if U_e[preReportIntervalId] > 1 then  
20:             return U_e[preReportIntervalId] - 1  
21:         end if  
22:     else if 1  
23:         return 1  
24:     end if  
25: end if

variation of data collected by vehicles on the same road segment. When the collected data are similar, the standard deviation is small, and therefore the proportion of update vehicles can be small. To exploit temporal locality, DD monitors the deviation of observed values obtained from two consecutive T_a. Based on the accuracy requirement imposed by the ITS application, DD defines three degrees of deviation. Next, DD adjusts the update frequency of the next T_a according to the degree of deviation. If the deviation is small, DD decreases the update frequency to further reduce the communication overhead. If the deviation is moderate, it maintains the same update frequency. Otherwise, it increases the update frequency to track the real state of the road more closely.

The variable reportInterval, lists U_e, BETA_e, V_e and an array preReportInterval are used to record the historical information during the execution of the DD algorithm. The variable, reportInterval, represents the index of the current T_a. Lists U_e, BETA_e and V_e record u_e, β_e and observed value v_e for every reportInterval on segment e, respectively. The length of the array preReportInterval is E. preReportInterval[e] records the index of the previous T_a in which update packets have been received from segment e. When DD begins, reportInterval is set to be 0 and lists U_e, BETA_e and V_e are set to be empty for every segment e. It is observed from line 8 to line 15, in every T_a, if there are update packets received from segment e, DD derives the observed value v_e and the standard deviation stdValue from the data collected on segment e in the current T_a. The observed value v_e is obtained by aggregation function specified by ITS applications. Based on the data collected in the current T_a, from line 16 to line 20, u_e, which is the number of sampling periods in each update period of segment e, and the proportion of update vehicles β_e in the next T_a are first calculated by using Algorithms 2 and 3, and then are stored in the corresponding lists U_e and BETA_e, respectively. From line 21 to line 23, if there are update packets received from segment e in the current T_a, the index of the current T_a is stored in preReportInterval[e].

The update frequency u_e is determined by the deviation of the observed values obtained from two consecutive T_a. To be consistent with MILP, we use u_e instead of α_e. Recall that the update vehicle should send an update packet to the infrastructure for every u_e sampling periods. And α_e could be obtained from u_e by using (6) and (7). From line 1 to line 3 of Algorithm 2, when DD starts (e.g., reportInterval == 1) or there is no update packets received from segment e since DD starts (e.g., v_e == -1 and preReportIntervalId == 0), the update frequency
Algorithm 3: ComputeBeta algorithm.

Input: \( v_e, stdValue, stdRange, Accu \)
Output: \( \beta_e \)

1: if \( v_e \leq -1 \) then
2: \( \beta_e \) = 1
3: else
4: \( p = \frac{stdValue}{stdRange} \)
5: \( p = p + p \cdot (1 - p)^{1 + \frac{1}{stdRange}} \)
6: if \( p \leq 0.1 \) then
7: \( \beta_e \) = 0.1
8: else
9: \( \beta_e \) = \( p \)
10: end if
11: return \( \beta_e \)
12: end if

is set to be maximum. Therefore, \( u_e \) is set to be 1, which means that the update vehicle should send an update packet to the infrastructure every sampling period. From line 4 to line 8 of Algorithm 2, if there are no update packets received on segment \( e \) (e.g., there is no vehicle passing by segment \( e \)) in the current \( T_a \), then DD maintains the update frequency of the previous \( T_a \), in which update packets have been received from this segment. Otherwise, DD calculates the difference of values observed from two consecutive \( T_a \).

DD uses a safety boundary of \( \frac{1}{3} ValueRange \cdot (1 - \text{Accu}) \) to classify the derived deviation. From (13), to satisfy the accuracy requirement, the deviation of the observed value to the real value should be less than or equal to \( ValueRange \cdot (1 - \text{Accu}) \). From line 9 to line 16, when the deviation of values observed from two consecutive \( T_a \) is less than or equal to \( ValueRange \cdot (1 - \text{Accu}) \), DD considers that the update frequency is sufficient to satisfy the accuracy requirement. Then the derived deviation is classified into two cases. If the deviation is less than or equal to \( \frac{1}{3} ValueRange \cdot (1 - \text{Accu}) \), DD considers it safe to decrease the update frequency. Therefore, \( u_e \) is increased by 1. Otherwise, if the deviation is less than or equal to \( ValueRange \cdot (1 - \text{Accu}) \), but is larger than \( \frac{1}{3} ValueRange \cdot (1 - \text{Accu}) \), although the current update frequency could satisfy the accuracy requirement, it is not safe to decrease the update frequency. Therefore, DD maintains the current \( u_e \). From line 17 to line 25, when the deviation is larger than \( ValueRange \cdot (1 - \text{Accu}) \), which means that the current update frequency no longer satisfies the accuracy requirement. Therefore, DD increases the update frequency to collect more data. At this point, the derived deviation can be classified into two cases. If the deviation is larger than \( ValueRange \cdot (1 - \text{Accu}) \), but less than or equal to \( \frac{5}{12} ValueRange \cdot (1 - \text{Accu}) \), DD decreases \( u_e \) by 1 to increase the update frequency. Otherwise, if the deviation is larger than \( \frac{5}{12} ValueRange \cdot (1 - \text{Accu}) \), DD sets the update frequency to be maximum to track the real state on the road. Therefore, \( u_e \) is set at 1.

Algorithm 3 is used to calculate \( \beta_e \) for the next \( T_a \). \( \beta_e \) is decided by standard deviation \( stdValue \), which represents the variation of the collected data received during the current \( T_a \). From line 1 to line 2 of Algorithm 3, if there is no vehicle passing by segment \( e \) in the current \( T_a \), then \( \beta_e \) is set at 1. This is to require all the vehicles on segment \( e \) to send update packets to the server in the next \( T_a \). \( \beta_e \) is calculated between line 4 and line 11 of Algorithm 3. In line 4, DD first makes use of \( stdRange \) to measure the deviation of the collected data updated by individual vehicles. Here \( stdRange \) represents the maximum standard deviation of the collected data updated by individual vehicles. Therefore, \( p \in [0, 1] \). For instance, if the valid range of outdoor temperature is \((-40, +40)\), we can obtain \( stdRange \) by setting the upper bound at 40 and the lower bound at -40, which is 56.5685425. Then DD weights \( p \) by the accuracy requirement \( Accu \) to derive the proportion of update vehicle \( \beta_e \) in line 5. A higher accuracy requirement leads to a larger \( \beta_e \). Finally, if \( \beta_e \) is smaller than or equal to 0.1, it is set at 0.1 to maintain a minimum proportion of update vehicles.

For every signal alive period \( T_a \) of the DD algorithm, the computational complexity is \( O(N \cdot K) \). In every \( T_a \), for every segment \( e \), DD first computes the observed value \( v_e \) and the standard deviation \( stdValue \) from the data in the update packets collected from segment \( e \). And the computational complexity of this computation is \( O(N \cdot K) \) because the maximum number of update packets received from all the segments is \( N \cdot K \), which means that all the vehicles send an update packet to the infrastructure in every sampling period. Then DD calculates the \( u_e \) and \( \beta_e \) values of the next signal alive period \( T_a \) by using Algorithm 2 and Algorithm 3, respectively. The computational complexity of this step is \( O(E \cdot (1 + 1)) \) since the computational complexity of Algorithm 2 and Algorithm 3 are \( O(1) \) and \( O(1) \), respectively. Therefore, for every signal alive period of the DD algorithm, the computational complexity of DD is \( O(N \cdot K + E \cdot (1 + 1)) \), that is \( O(N \cdot K) \).

By exploiting spatial locality and temporal locality, communication overhead could be reduced significantly by DD when the observed values change smoothly and the collected data are similar. However, if the observed value changes greatly from \( T_a \) to \( T_a \) and there is a considerable fluctuation of the data collected in the vicinity, the communication overhead needed by DD would have to be increased to maintain accuracy.

VI. EVALUATION

The goal of the MCO problem is to minimize communication overhead while providing a quality-oriented data collection in VSN. Therefore, in this section, we evaluate our solutions in terms of Percentage of the reduced communication overhead and Percentage of the Accuracy-satisfied signal alive periods. The percentage of reduced communication overhead is the reduction in communication overhead while obtaining the observed values with respect to the amount of communication overhead in obtaining the real values. The percentage of accuracy-satisfied signal alive periods is defined as the number of signal alive periods that satisfy the accuracy requirement divided by the total number of signal alive periods. In this section, we first introduce the experimental settings. Next we conduct experiments in two typical environments, freeway and downtown, to give comprehensive studies of the effectiveness of the proposed solutions.

A. Experimental Settings

Consider an ITS application (e.g., Incident Detection) that aims to collect average speed of vehicles from AoI. Two types
TABLE IV
DEFAULT EXPERIMENT PARAMETERS

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sampling period ($T_s$)</td>
<td>1 second</td>
</tr>
<tr>
<td>Signal alive period ($T_a$)</td>
<td>5 seconds</td>
</tr>
<tr>
<td>Accuracy requirement ($\text{Accu}$)</td>
<td>95%</td>
</tr>
<tr>
<td>Aggregation function</td>
<td>Average</td>
</tr>
<tr>
<td>Simulation time</td>
<td>1000 seconds</td>
</tr>
<tr>
<td>Segment length on downtown</td>
<td>200 meters</td>
</tr>
<tr>
<td>Segment length on freeway</td>
<td>300 meters</td>
</tr>
<tr>
<td>Vehicle arrival rate on downtown</td>
<td>10 vehicles per second</td>
</tr>
<tr>
<td>Vehicle arrival rate on freeway</td>
<td>1 vehicle per second</td>
</tr>
</tbody>
</table>

Fig. 2. Chosen road segments of Tsing Long Freeway.

Fig. 3. Number of vehicles on chosen segments of Tsing Long Freeway.

of AoI, which are freeway and urban road in downtown area, are extracted from Hong Kong’s roads by using JOSM [45]. We employ SUMO [25], which is a popular road traffic simulator, to generate traces of moving vehicles on the extracted road map.

The parameters and their default values used in this experiment are given in Table IV. In order to evaluate the quality of the observed values obtained by MILP and DD, we make use of Expected-speed, which is the real mean speed of all vehicles that can be obtained only when every vehicle updates its collected data in every sampling period. In every signal alive period, we obtained the observed value of DD by randomly choosing update vehicles according to the proportion of update vehicles derived by DD. To obtain more representative experimental results, we repeated the random experiment 50 times and showed the mean observed value. The standard deviations are also shown in the experimental results.

B. Freeway

Tsing Long Freeway is an expressway from North West Tsing Yi Interchange on Tsing Yi Island to Yuen Long, in Hong Kong. We extract 900 meters of Tsing Long Highway from south to north, the screenshot of which is presented in Fig. 2. The chosen road is divided into three segments, and the length of each segment is 300 meters. There are three lanes on the chosen segments and the speed limitation of which is 100 km/h. We use the “duarouter” tool provided by SUMO to generate vehicle traces in the chosen segment for 1000 seconds. A vehicle is supposed to enter the chosen road every second.

In order to give a more representative experimental results, we simulate an incident on the chosen road. The incident happened at 270 seconds and the location of the incident is marked with a red star in Fig. 2. Because of the incident, all lanes were blocked until 400 seconds. The number of vehicles on each segment is depicted in Fig. 3. It is observed that when the incident happened at 270 seconds, the number of vehicles on Segment 1 kept increasing until the incident was clear. The number of vehicles on Segment 2 first increased and then became unchanged since all the lanes are blocked because of the incident. The number of vehicles on Segment 3 first decreased dramatically, and then became zero during the incident. When the block was cleared at 400 seconds, the number of vehicles on Segments 1 and 2 decreased greatly while the number of vehicles on Segment 3 increased. Since the incident occurred in the middle of Segment 2, we choose the experimental results from 250 seconds to 450 seconds on Segment 2 to present our experimental results.

First, we study the proposed algorithms on Segment 2, the results of which are shown in Fig. 4. There are 40 signal alive periods from 250 seconds to 450 seconds. The length of each signal alive period is 5 seconds. Fig. 4(a) depicts the mean speed of every signal alive period obtained by MILP and DD. In Fig. 4(a), MILP always met the accuracy requirement. As for DD, the observed values of 94.65% signal alive periods satisfied the accuracy requirement, which is given in Table V. The communication overhead needed to obtain the observed values of every signal alive period is depicted in Fig. 4(b). As we can see, MILP achieved a minimum communication overhead all the time since it is under the assumption of having real values. When no incident occurs, the mean speed on the freeway is stable (e.g., from 250 seconds to 270 seconds). When the mean speed is stable, DD effectively reduces the communication overhead. When the incident occurred, the mean speed decreased rapidly from 270 seconds to 300 seconds and the number of vehicles on Segment 2 kept increasing, the communication overhead required by DD increased significantly. When all the lanes were blocked because of the incident from 330 seconds to
In order to have a comprehensive understanding of DD, we show performance of DD in terms of Percentage of the Accuracy-satisfied signal alive periods and Percentage of the reduced communication overhead. The performance of DD between 50 seconds and 850 seconds is shown in Table V. The simulation time is divided into four time intervals. It could be observed from Fig. 3 that, in the first time interval 50–250, the incident did not occur, and thus the number of the vehicles on Segment 2 was relatively stable. DD could achieve a reduction of 93.68% communication overhead while the observed values of 99.20% signal alive periods could meet the accuracy requirement. In the second time interval 250–450, an incident happened and blocked all the lanes for about two minutes. Because of deceleration and acceleration caused by the incident, the mean speed of vehicles fluctuated and consequently decreased the performance of DD. In the third time interval 450–650, although the incident was over, it took time for the traffic to resume normal, which could be observed from the fact that the number of vehicles still fluctuated. But the fluctuation of the number of vehicles is smaller than that of the second time interval. Therefore, the third time interval achieved a better performance than that of the second time interval, but still worse than that of the first time interval. The number of the vehicles on Segment 2 became stable again in the fourth time interval 650–850, and

### Table V

<table>
<thead>
<tr>
<th>Time (s)</th>
<th>Percentage of the reduced communication overhead (Standard Deviation) (%)</th>
<th>Percentage of the Accuracy-satisfied signal alive periods (Standard Deviation) (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>[50,250]</td>
<td>93.68 (±0.52)</td>
<td>99.20 (±1.27)</td>
</tr>
<tr>
<td>[250,450]</td>
<td>78.29 (±1.49)</td>
<td>94.65 (±3.46)</td>
</tr>
<tr>
<td>[450,650]</td>
<td>85.16 (±1.49)</td>
<td>95.05 (±3.95)</td>
</tr>
<tr>
<td>[650,850]</td>
<td>93.29 (±0.38)</td>
<td>99.10 (±1.30)</td>
</tr>
</tbody>
</table>

390 seconds, although the number of vehicles on Segment 2 reached the maximum value, a considerable communication overhead has been reduced by DD since the mean speed became relatively stable. For the simulation period from 250 seconds to 450 seconds, compared with the communication overhead required to obtain Expected-speed, DD could reduce 78.29% communication overhead.
TABLE VI
THE PERFORMANCE OF DD FOR DIFFERENT ACCURACY REQUIREMENTS ON SEGMENT 2 ON TSING LONG FREEWAY

<table>
<thead>
<tr>
<th>Accuracy requirement (%)</th>
<th>Percentage of the reduced communication overhead (%) (Standard Deviation)</th>
<th>Percentage of the Accuracy-satisfied signal alive periods (%) (Standard Deviation)</th>
</tr>
</thead>
<tbody>
<tr>
<td>85</td>
<td>86.52 (±1.20)</td>
<td>98.05 (±2.08)</td>
</tr>
<tr>
<td>90</td>
<td>82.99 (±1.36)</td>
<td>97.95 (±1.71)</td>
</tr>
<tr>
<td>95</td>
<td>78.29 (±1.49)</td>
<td>94.65 (±3.46)</td>
</tr>
</tbody>
</table>

the performance of DD became similar to the performance in the first time interval.

In Table VI, we vary the accuracy requirement among 85%, 90% and 95% to study the impact of accuracy requirement on the performance of DD. As we can see, if the accuracy requirement is slightly relaxed, both the performance and the communication overhead of DD can be improved, because both the update frequency and the proportion of update vehicle are dependent on the accuracy requirement in DD. A strict accuracy requirement leads to a small safety boundary, thus reducing the possibility of decreasing the update frequency. Also, the proportion of update vehicle is weighted by the accuracy requirement. An increase of the accuracy requirement leads to an increase of the proportion of update vehicle, therefore increasing the communication overhead.

C. Urban Road in Downtown Area

Mong Kok is one of the major shopping areas in Hong Kong. We extract a 1 km × 1 km road network from Mong Kok, the screenshot of which is presented in Fig. 5. We use the tool “randomTrips.py” provided by SUMO to generate a set of random trips in the road network. The vehicle arrival rate is 10 vehicles per second, which means in every second, 10 vehicles enter the extracted road network. We choose Lai Chi Kok Road, which is an arterial road in Mong Kok. The speed limit is 70 km/h.

The chosen road is divided into three segments from south to north, marked with red lines in Fig. 5. The length of each segment is roughly 200 meters. The number of vehicles on the chosen road map and the number of vehicles on the each of the chosen road segments are depicted in Figs. 6 and 7, respectively. It is observed from Fig. 6 that, as time progresses, the number of vehicles on the road map keeps increasing. We choose Segment 2, which is in the middle of the chosen road, to demonstrate our experimental results.

In order to present our experimental results more clearly, we extract the results collected in the range from 300 seconds to 500 seconds in Fig. 8 for discussing the performance of the proposed solutions.

In Fig. 8, there are 40 signal alive periods, from 300 seconds to 500 seconds. The length of each signal alive period is 5 seconds. Fig. 8(a) depicts the mean speed of every signal alive period obtained by MILP and DD. MILP always met the accuracy while for DD, the observed values of 81.45% signal alive periods satisfied the required accuracy, which is given in Table VII. The communication overhead required to obtain the observed value of every signal alive period is depicted in Fig. 8(b). As we can see, MILP dramatically reduced the communication overhead all the time. When the mean speed fluctuated dramatically and the number of vehicles on Segment 2 kept increasing (e.g., from 345 seconds to 375 seconds), DD incurred a relative large communication overhead. When the mean speed became relatively stable (e.g., from 375 seconds to 400 seconds), even the number of vehicles on Segment 2 increased, DD greatly reduced
the communication overhead. For the simulation period from 300 seconds to 500 seconds, compared with the communication overhead required to obtain Expected-speed, DD could reduce 72.68% communication overhead.

The performance of DD between 300 seconds and 900 seconds is shown in Table VII. The simulation time is divided into three time intervals. The length of each time interval is 200 seconds. We can observe that, the performance of the three time intervals is relatively stable. In other words, DD compensated approximate 20% accuracy for 70% communication overhead reduction in every time interval. This is because although in Fig. 6, the number of vehicles in AoI kept increasing, we could observe in Fig. 7 that the traffic on the Segment 2 was relative stable. Specifically, since there was no incident happened during the simulation of Mong Kok, the number of vehicles on Segment 2 changed regularly according to the traffic light. When the traffic light, which is located at the intersection between Segment 1 and the Segment 2, turned green (e.g., 455 seconds), the number of vehicles on Segment 2 started to decrease. When the traffic light between Segment 1 and Segment 2 turned red or the traffic light between Segment 2 and Segment 3 turned green, the number of vehicles on Segment 2 started to increase (e.g., 420 seconds).

In Table VIII, we vary the accuracy requirement among 85%, 90% and 95% to study the impact of accuracy on the performance of DD. Similar to the freeway area, if the accuracy requirement is slightly relaxed, both the performance and the communication overhead of DD can be improved.

To sum up, for urban roads and freeways, MILP always incurs a minimum communication overhead while maintaining a good result accuracy level. DD achieves a decent performance on freeway. Due to rapid accelerations and decelerations of individual vehicles caused by traffic lights, the performance of DD on urban road is not as good as that on the freeway, but it could still significantly reduce communication overhead.
In this paper, we have introduced a QDC scheme to deal with real-time data collection in VSNs. QDC is able to maintain the accuracy while minimizing the communication overhead. It also offers flexibility by supporting ITS applications to specify their own real-time and accuracy requirements. An MCO problem is defined to formulate the QDC scheme. Two algorithms, MILP and DD are proposed to solve the MCO problem. MILP is capable of achieving an optimal solution of the MCO problem under the assumption of having the data collected by every vehicle in every sampling period while DD could obtain an efficient solution without this impractical assumption. We have conducted extensive experiments using SUMO in two typical environments, urban roads in downtown area and freeway, to demonstrate the effectiveness of the proposed algorithms.

APPENDIX

EXPLANATION OF TRANSFORMATION FOR (17)

In this section, we give a detailed explanation for linear transformation of (17).

From (17), we can get two inequalities,
\[
\frac{k}{j} \leq u_e \leq \frac{k-1}{j-1}
\]  
(19)
and
\[
u_e \leq \frac{K}{j}.
\]  
(20)
It can be simply approved that \(\frac{k}{j}\) is always smaller than \(\frac{k-1}{j-1}\) and \(\frac{K}{j}\). Therefore, if \(\frac{k-1}{j-1}\) is smaller than \(\frac{K}{j}\), (19) and (20) equals (19), otherwise, the two equations can be expressed as
\[
\frac{k}{j} \leq u_e \leq \frac{K}{j}
\]  
(21)
In other words, if \(j \cdot (k - 1) \leq K \cdot (j - 1)\), then
\[
\begin{align*}
\frac{k}{j} & \leq u_e \leq \frac{k-1}{j-1} \\
p_{j,k,e} & = 1
\end{align*}
\]  
(22)
otherwise
\[
\begin{align*}
\frac{k}{j} & \leq u_e \leq \frac{K}{j} \\
p_{j,k,e} & = 1
\end{align*}
\]  
(23)
We introduce auxiliary binary variables \(v_{j,k,e}^2\) to \(v_{j,k,e}^3\) and constraints C4.1 to C4.3 to model the relationship in (22) and (23).

TABLE VIII

<table>
<thead>
<tr>
<th>Accuracy requirement (%)</th>
<th>Percentage of the reduced communication overhead (Standard Deviation) (%)</th>
<th>Percentage of the Accuracy-satisfied signal alive periods (Standard Deviation) (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>85</td>
<td>82.04 (±3.39)</td>
<td>94.85 (±3.76)</td>
</tr>
<tr>
<td>90</td>
<td>79.81 (±32.03)</td>
<td>92.55 (±4.11)</td>
</tr>
<tr>
<td>95</td>
<td>72.68 (±4.13)</td>
<td>81.45 (±6.25)</td>
</tr>
</tbody>
</table>

VII. CONCLUSION

We introduce auxiliary binary variables \(v_{j,k,e}^2\) to \(v_{j,k,e}^3\) and constraints C4.1 to C4.3 to model the relationship in (22) and (23).

Victor C. S. Lee (M’92) received the Ph.D. degree in computer science from the City University of Hong Kong, Hong Kong, in 1997. He is currently an Assistant Professor with the Department of Computer Science, Chongqing University, Chongqing, China. His research interests include data dissemination in vehicular networks, real-time databases, and performance evaluation. He is a member of the Association for Computing Machinery and the IEEE Computer Society. He was the Chairman of the IEEE Hong Kong Section Computer Chapter in 2006–2007.

Dusit Niyato received the B.Eng. degree from the King Mongkut’s Institute of Technology Ladkrabang, Bangkok, Thailand, in 1999, and the Ph.D. degree in electrical and computer engineering from the University of Manitoba, Winnipeg, MB, Canada, in 2008. He is currently an Associate Professor at the School of Computer Science and Engineering, Nanyang Technological University, Singapore. His research interests include energy harvesting for wireless communication, Internet of Things, and sensor networks.

Yaoxun Duan received the Graduate degree in automatic control from the Huazhong University of Science and Technology, Wuhan, China, in 2006, and the M.S. degree in computer science from Chongqing University, Chongqing, China, in 2015. He is currently working toward the Ph.D. degree in computer science with the City University of Hong Kong, Kowloon, Hong Kong. His current research interests include multicore architecture optimization, high-performance parallel computing, wireless networks, and wireless localization.

Kai Liu (S’07–M’12) received the Ph.D. degree in computer science from the City University of Hong Kong, Hong Kong, in 2011. He is currently an Assistant Professor with the College of Computer Science, Chongqing University, Chongqing, China. From 2010 to 2011, he was a Visiting Scholar with the Department of Computer Science, University of Virginia, Charlottesville, VA, USA. From 2011 to 2014, he was a Postdoctoral Fellow with Nanyang Technological University, Singapore; the City University of Hong Kong; and Hong Kong Baptist University, Hong Kong. His research interests include mobile computing, intelligent transportation systems, and vehicular cyber-physical systems.

Sarana Nutanong received the Ph.D. degree in computer science from the University of Melbourne, Australia, in 2010. He is currently an Assistant Professor with the Department of Computer Science, City University of Hong Kong (CityU), Kowloon, Hong Kong. Before joining CityU, he was a Postdoctoral Research Associate at the University of Maryland Institute for Advanced Computer Studies between 2010 and 2012 and held a research faculty position with the Johns Hopkins University from 2012 to 2013. His research interests include scientific data management, data-intensive computing, spatial-temporal query processing, and large-scale machine learning.