Ad Hoc Networks 000 (2015) 1–11

Contents lists available at ScienceDirect

Ad Hoc Networks

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

Estimating online vacancies in real-time road traffic monitoring with traffic sensor data stream

Feng Wang[a], Liang Hu[a], Dongdai Zhou[b], Rui Sun[a], Jiejun Hu[a], Kuo Zhao[a,*]

[a] College of Computer Science and Technology, Jilin University, Changchun 130012, China
[b] College of Software, Northeast Normal University, Changchun 130012, China

ARTICLE INFO

Article history:
Received 16 December 2014
Revised 5 July 2015
Accepted 8 July 2015
Available online xxx

Keywords:
Data stream
Road traffic monitoring
Real time
Big data

ABSTRACT

Real-time road traffic monitoring is widely considered to be a promising traffic management approach in urban environments. In the smart cities scenario, traffic trajectory sensor data streams are constantly produced in real time from probe vehicles, which include taxis and buses. By exploiting the mass sensor data streams, we can effectively predict and prevent traffic jams in a timely manner. However, there are two urgent challenges to processing the massive amounts of continuously generated trajectory sensor data: (1) the inhomogeneous sparseness in both spatial and temporal dimensions that is introduced by probe vehicles moving at their own will, and (2) processing stream data in real time manner with low latency. In this study, we aim to ameliorate the aforementioned two issues. We propose an online approach to addresses the major defect of inhomogeneous sparseness, which focuses on employing only real-time data rather than mining historical data. Furthermore, we set up a real-time system to process trajectory data with low latency. Our tests are performed using field test data sets derived from taxis in an urban environment; the results show that our proposed method lends validity and efficiency advantages for tackling the sparseness, and our real-time system is viable for low latency applications such as trafficmonitoring.

1. Introduction

As motor vehicles continue to be a transportation method of choice in developed countries, transportation infrastructure continues to be overwhelmed by the number of cars on the road, leading to traffic jams and congestion in most major metropolises worldwide. Consequently, road traffic monitoring has become an essential and vital element for providing efficient and safe road transport. Real-time road traffic monitoring has received considerable attention and is considered to be a promising approach because it offers the opportunity to employ mitigation measures–such as changing the timings of traffic lights or advising commuters to take alternate routes–in real time. However, it is a significant challenge to process the massive amounts of real-time traffic sensor data that are continuously generated in large cities.

Traditional traffic monitoring technologies include magnetic loops [1], camera-based systems [2], microwave radar [3], laser-based systems [4,5], infrared detectors [6], and ultrasonic detectors [7]. These are roadside technologies that detect passing vehicles and provide precise and stable traffic information about a specific location where they are installed. The major disadvantage of these technologies is the high cost of deployment and maintenance. For example, a magnetic loop sensor can cost hundreds of dollars, and daily maintenance can cost much more. It is infeasible to install these expensive technologies densely enough to provide data on a city's entire road network. Several studies [3,8] have

* Corresponding author. Tel.:+86 0431 85168716; fax. +86 0431 85166494.
E-mail addresses: wangfeng12@mails.jlu.edu.cn (F. Wang), hul@jlu.edu.cn (L. Hu), ddzhou@nenu.edu.cn (D. Zhou), ruisun14@mails.jlu.edu.cn (R. Sun), hujc12@mails.jlu.edu.cn (J. Hu), zhaokuo@jlu.edu.cn (K. Zhao).

http://dx.doi.org/10.1016/j.adhoc.2015.07.003
1570-8705/

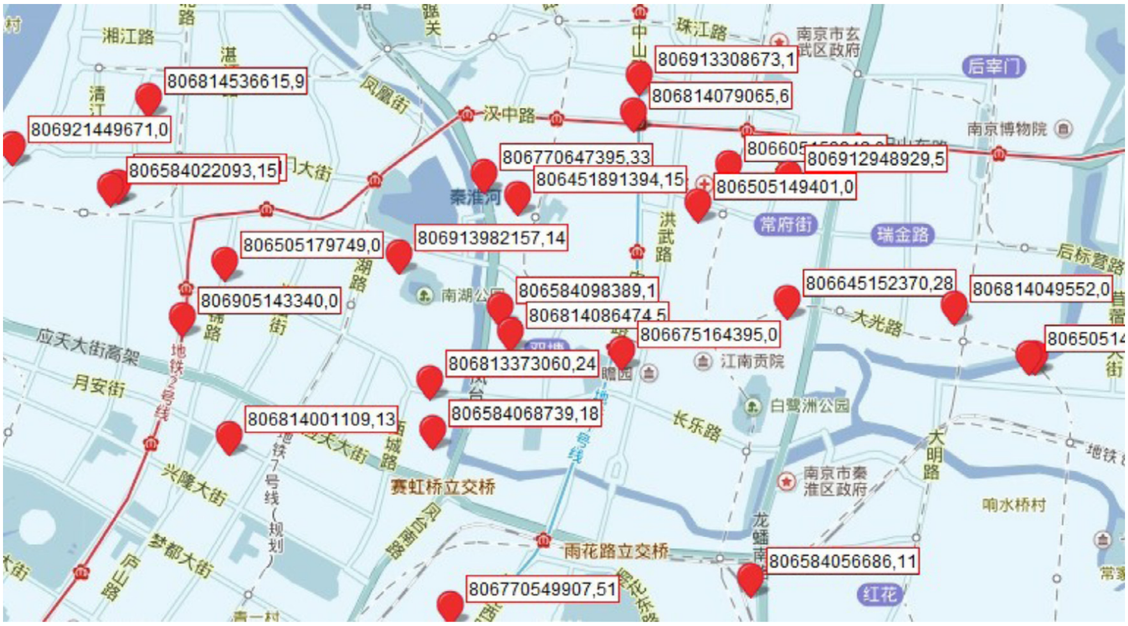

Fig. 1. A distribution snapshot of vehicles over the partial study region (red icons are probe taxis).

been published that have explored ways to overcome the limitations of coverage with traditional technologies by making a trade-off between precision and scalability. Some studies proposed using high-resolution satellite imagery to make up for coverage shortages [9–11]. However, the availability of the proposed approaches is limited by weather and other factors that impact the precision and timeliness of the data provided by these traditional technologies.

In this paper, we present a real-time road traffic monitoring system that uses global positioning system (GPS) data collected through wireless communication in probe vehicles, such as taxis, to monitor the real-time traffic scenario. However, obtaining traffic monitoring information directly from the raw reports of GPS is still a significant challenge. First, the data include spatio-temporal vacancies because the probe vehicles move at their own will. The random distribution of probe vehicles inevitably leads to inhomogeneous sparseness in data, which is an obstacle in acquiring the real-time traffic state in the entire study region. Second, computing infrastructure continues to be overwhelmed by the massive amounts of continuously generated trajectory sensor stream data.

In this study, we attempt to circumvent those obstacles. We present an online approach to addresses the major defect of inhomogeneous sparseness, which focuses on employing only real-time data rather than mining historical data. Furthermore, we set up a real-time system to process trajectory streaming data with Apache Storm which is an open-source distributed real-time computation framework.

To validate the system, we experiment with a field test data set from an urban environment, which contains one-day trajectories of 7,648 taxis. The total number of points in this data set is about 18 million. Fig. 1 shows a snapshot of vehicle distribution over the partial study region. Every red icon is a probe taxi, and the labels indicate the taxi number and its velocity. Velocity information is not collected in the data set and is computed in real time on the basis of longitude, latitude, and time. The results show that our proposed method lends validity and efficiency advantages for tackling the sparseness, and the real-time system is viable for low latency applications such as traffic monitoring.

The rest of the paper is organized as follows: Related work is presented in Section 2. The problem is formally defined and an algorithm is detailed in Section 3. The implementation of the proposed system is described in Section 4. The validation results are presented in Section 5. Finally, we conclude our paper and suggest avenues for future work in Section 6.

2. Related work

Traditionally, radar sensors, cameras, and similar equipment are static and are placed in roadside positions. For instance, traffic cameras were deployed and magnetic induction coils were installed under road surfaces in monitored areas. Through consecutive images captured by traffic cameras or electromagnetic signals, it was possible to easily measure vehicle speed and frequency of passing. However, these kinds of monitoring equipment cannot be installed at the density needed to monitor traffic in real time, particularly in large urban areas. This presents an unprecedented opportunity to harness traffic monitoring by developing some other approaches.

Some research [12–15] focuses on the potential of crowdsourcing, such as smartphones and mobile cellular networks, to facilitate the collection of vast amounts of traffic management data from probe vehicles and pedestrians throughout a city. In [16] and [17], the authors present crowd-based route recommendation systems for urban transportation. However, the energy and capability of mobile phones can become a problem, as [18] discusses in a review of the feasibility of utilizing smartphones as sensors to gather and disseminate location-relevant information to build a global view of a monitored area while considering energy and everyday use of phones.

Vehicular wireless sensor networks have also been developed to monitor traffic conditions in recent years. Several studies [19–22] have discussed in detail the diversification of urban traffic monitoring systems based on wireless sensor networks. Others [23–25] have produced surveys of vehicular cloud computing. Other issues with the use of wireless sensor networks are coverage and cost, in part because of the need for infrastructure such as roadside base stations. Timeliness may also be an issue; if a vehicle is far from a roadside station, traffic condition data cannot be transmitted to the server in real time.

Another method of monitoring urban traffic that has received significant attention is the use of vehicles equipped with GPS. The GPS may be a GPS module embedded in the vehicle or an on-board smartphone. Map-matching is an active research area in utilization of GPS trajectories. [26] proposes an online map-matching algorithm based on the Hidden Markov Model (HMM) that is robust to noise and sparseness. In addition, GPS trajectories data that result from pervasive probe vehicles has some intrinsic limitation due to vacancies that is introduced by sparse probe vehicle. To addresses such constraints, a number of recent work propose to estimate the vacancies. Many compressive sensing-based algorithms have been proposed to solve signal reconstruction or data compression problems with this method. [27] gathers GPS data from a number of historical probe vehicles and then, employs a compressive sensing [28] to handle vacancies in the traffic matrix. [29] presented a spatial-temporal method based on multiple linear regression models to calculate the traffic speed of the segments without sensor data. Nevertheless if those solutions are employed, the historical data is imperative and we have to explore related segments to vacancy segments by data mining ahead of time. It is incapable of estimation for vacancies in real time manner by reason of the scale of historical trajectory data from large urban increasing.

Unlike aforementioned research, our study presents a realtime computation system to process real-time streaming traffic data to enable estimation online,in which we focus on employing only real-time data rather than mining historical data to circumvent this obstacle .

3. Problem statement and algorithm design

Our proposed research angle is inspired by multiple linear regression to estimate traffic conditions and address the issues derived from inhomogeneous sparseness constraints. We employ geographical hash to characterize regions of the earth surface, and study the speed of the central region by means of the speeds of eight neighbor regions with multiple linear regression approach. The concrete solution is detailed in the remainder of this section. The Table 1 is the illustration of symbols in this section.

Table 1
Illustration of notations in this section.

N	The amount of vehicles
V	The probe vehicle set
i	The assigned id of probe vehicle
T_i	The set of timestamps at which vehicle i reports its states
S_i	The set of reports of vehicle i at each timestamp in T_i
r_m	The assigned id of region
t_m	The timestamp
d_m	The direction
$R(r_m, t_m, d_m)$	The set of reports from the neighborhoods of region m
$\hat{v}$	The average speed
C_{TCM}	The traffic condition matrix
E_{TCM}	The estimation matrix
$R_{nb}(i)$	The neighbor region set around region r_i
v_{it}	The traffic condition in region r_i at timestamp t
$\hat{v}_{it}$	The estimation for v_{it}
e_t	Error

3.1. Problem statement

Probe vehicles are deployed to acquire traffic conditions on roads. There are N probe vehicles on the road, denoted by a set of $V = \{1, 2, \ldots, N\}$. Every probe vehicle is assigned a number denoted by i. Each vehicle moves along a specified road, gathers traffic information, and makes a periodic report. The intervals between reports are not the same for a vehicle. The reported states at time are a four tuple: “no, time, longitude, latitude”, such as “806770783592,2010-09-02 17:50:51.000,118.756151,32.045157”. Let T_i denote the set of timestamps at which vehicle i reports its states $\mathrm{T}_i = \{t_1^i, t_2^i, \ldots, t_k^i\}$ and vehicle i forms a report set, $S_i = \{s_i(t) | t \in T_i\}$.

This study adopts average speed as the metric for quantifying traffic condition, since a good traffic condition always allows higher speed and larger throughput.

Definition 3.1 (Traffic condition)**.** In a certain region, the traffic condition of this region at time t_0 is defined as the average speed $\hat{v}$ of the probe vehicles in the region r_0 and with the same direction d_0. To compute the average speed $\hat{v}$, we need to collect reports that reside in the neighborhood of (r_0, t_0, d_0). Let $R(r_0, t_0, d_0)$ denote the set of reports from the neighborhood.

$$R(r_0, t_0, d_0) = \{S_i(t) | r(r, r_0) < \Delta r \wedge |t - t_0| < \Delta t \wedge d = d_0\} \quad (1)$$

Then, the average speed $\hat{v}$ is computed as

$$\hat{v}(r_0, t_0, \mathrm{d}_0) = \frac{1}{N} \sum\nolimits_{S_i(t) \in R(\mathrm{r}_0, t_0, d_0)} S_i(t) \cdot v, \text{ where } N = |R(r_0, t_0, d_0)| \quad (2)$$

The number of reports, N, is usually a random value, because of the random distribution of probe vehicles that results from vehicles traveling at their own will. N is also influenced by the selection of Δr and Δt. In this paper, Δt is a small value that approaches zero in real-time processing. It is obvious that N is large and $\hat{v}$ would be close to the real traffic condition in terms of the central limit theory. In the worst case, in which sensor data are inhomogeneously sparse in both spatial and temporal dimensions, N is zero, so that there is no way to compute $\hat{v}$.

As shown in Fig. 2, the urban traffic network is divided into grids. It is difficult to obtain an integral traffic condition when there are a lot of vacancies with no reports, such as R_0. There is no guarantee of an adequate number of reports being generated for every region for every time period.

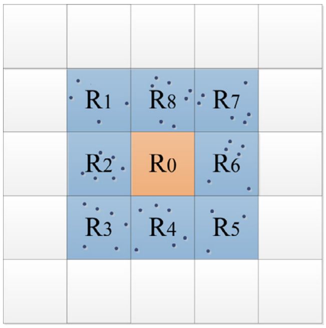

Fig. 2. Vacancy grid for urban traffic network.

To overcome the constraints of inhomogeneous sparseness, we can estimate the traffic condition by leveraging the states of neighboring regions using only real-time data rather than historical data. By considering urban traffic network connectivity, we can find an enclosed neighboring area to compensate for inhomogeneous sparseness. Fig. 2 shows the making of an estimate for R_0 by utilizing the states of $\{R_0, R_1, \ldots, R_8\}$.

Definition 3.2 (Vacancy estimation problem). Given a set of regions R in a set of time series T in the direction d_0, where $R_n = \{r_0, r_1, \ldots, r_{n-1}\}$, $\mathrm{T}_m = \{t_0, t_1, \ldots, t_{m-1}\}$:

If $N = |R(r,t,d)| \neq 0$, we compute a traffic condition matrix (TCM), denoted by $C_{TCM} = (v_{ij})_{m \times n}$ where v_{ij} is the average speed of $R(r, t, d)$.

If $N = |R(r,t,d)| = 0$, we estimate TCM, denoted by $E_{TCM} = (v_{ij})_{m \times n}$where v_{ij} is an average speed estimation of $R(r, t, d)$.

For system validation, we compute estimates on the assumption of some specified experiment regions $N = |R(r,t,d)| = 0$ and thus, approximate the estimation matrix E_{TCM} to the original traffic matrix C_{TCM} as closely as possible

3.2. The multiple linear regression approach

Linear regression models contain more than one predictor variable. In our system, we assume that there are m neighboring regions available around the estimation region r_i, denoted as $r_1, \ldots, r_m$, and the neighbor node set denoted as $R_{\mathrm{nb}}(i) = \{r_1, r_2, \ldots, r_m\}$. Because regions are geographically close resources, not only do they correlate spatially with every neighboring region,r_j, $r_j \in R_{nb}(i)$, but there are also spatial correlations among neighboring regions, denoted as r_j is spatial correlation with r_k , where $\forall r_j, r_k \in R_{nb}(i)$. Therefore, all neighboring regions are considered a whole in the estimation, since using a single region will introduce random errors. We establish multiple regression models for representing spatial correlations between r_i and its neighboring regions at time t, denoted as

$$v_{it} = \beta_0 + \beta_1 v_{1t} + \beta_2 v_{2t} + \cdots + \beta_m v_{mt} + \mu_t \tag{3}$$

where v_{it} is the traffic condition in region r_i at time t , v_{kt}, $k = \{1, 2, \ldots, m\}$ is the traffic condition in region r_k, $r_k \in R_{\mathrm{nb}}(i)$ at time t, β_k are the partial correlation coefficients for v_{kt}, and μ_t is the random error term.

In Eq. (3), v_{it} will be considered an explanatory variable, and partial correlation coefficients β_k represent the impact of v_{kt} to v_{it}, $k = \{1, 2, \ldots, m\}$. Obviously, we can estimate $\hat{v}_{it}$ by using Eq. (3) when it is a vacancy in the region. We will sample h, $(h - m \geq 2)$ items to establish multiple linear regression models for computing the estimates $\hat{\beta}_k$ for β_k. β_k is substituted for $\hat{\beta}_k$ in Eq. (3) and can be further derived by Eq. (4).

$$\hat{v}_{it} = \hat{\beta}_0 + \hat{\beta}_1 v_{1t} + \hat{\beta}_2 v_{2t} + \ldots + \hat{\beta}_m v_{mt} \tag{4}$$

where $\hat{v}_{it}$ is an estimation for v_{it} and v_{kt} is the true value in the region $k, k = \{1, 2, \ldots, m\}$ at time t. Eq. (4) is called the vacancy estimate equation, and the error between the estimation and the true value is called the residual, which is denoted as e_t, $e_t = v_{it} - \hat{v}_{it}$. During the computation of correlation coefficients β_k, we employ a vector to represent the traffic condition collected from h samples, denoted as $V = (v_{i1}, v_{i2}, \ldots, v_{ih})^T$, if there are m neighboring regions and they construct the matrix X denoted as

$$X = \begin{bmatrix} 1 & v_{11} & v_{21} & \cdots & v_{m1} \\ 1 & v_{12} & v_{22} & \cdots & v_{m2} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ 1 & v_{1h} & v_{2h} & \cdots & v_{mh} \end{bmatrix} \tag{5}$$

Similarly, we can rewrite the estimation of correlation coefficients β_k as follows:

$$\hat{\beta} = (\hat{\beta}_0, \hat{\beta}_1, \hat{\beta}_2, \ldots, \hat{\beta}_m)^T = (X^T X)^{-1}(X^T V) \tag{6}$$

3.3. Algorithm design

To solve the traffic vacancy estimate problem, we leverage the multiple linear regression approach to compute the estimates for the original matrix. In the multiple linear regression approach, we sample items to establish multiple linear regression models for computing the estimate correlation coefficients in Eq. 3. Then, correlation coefficients are employed to calculate the estimates in Eq. (4). The sample is that only leverages real-time data instead of historical data. To circumvent this obstacle, we design the sample approach based on spatial correlation, since traffic conditions are interrelated in a closed and connected region. Meanwhile, the smaller the range of region is with the greater the impact on the same direction. The concrete solution is detailed in Algorithm 1.

In steps 1 and 2 of Algorithm 1, our purpose is to find all neighbors around a specified probe vehicle in a fixed area and time window. This common requirement is called the fixed-radius near neighbors problem, which is a variant of the KNN problem. We employ the geohash method, which divides points on the earths surface into grids. This approach makes iterative binary divisions with the longitude [-180,180] and latitude [-90, 90] until it is possible to spot the given location with the required precision. The algorithm of obtaining a geohash from a location is common. A binary string will be returned in the division procedure, which is just geohash. It is usually represented in base32, but not in this article. In fact, because the geohash is stored in Redis' sorted set, the geohash is transformed into a float number so that the sorted set can be automatically sorted by the float number. The mapping from the geohash in a binary string to a float number is designed as follows:

Many bits at the beginning of the binary string should be the same, because the points are in the same city, and there

Algorithm 1 Traffic vacancy estimate.

Input:
$location_i$: include longitude and latitude
$scope_i$: range of region
d_i : direction of traffic condition estimation
Δt_i : time window
t_i : time of traffic condition
Output:
E_{TCM} : traffic condition estimation matrix
1: $r(r_i, t_i, d_i) \leftarrow geohash(\, longitude, latitude), scope_i, \Delta t_i$
2: $R_{\text{nb}}(i) \leftarrow SearchNeighboringr(r_0, t_0, d_0), |R_{\text{nb}}(i)| = 8$
3: **for** $K = 1$ to $|R_{\text{nb}}(i)|$ **do**
4: $R^k(r_i, t_i, d_i) = \{S_i^k(t) | r(r, r_i) < \Delta r \wedge |t - t_i| < \Delta t_i \wedge d = d_i\}$
5: $\hat{v}^k(r_i, t_i, \mathrm{d}_i) = \frac{1}{N} \sum_{S_i^k(t) \in R(\mathrm{r}_i, t_i, d_i)} S_i^k(t) \cdot v$
where $N = |R(r_i, t_i, d_i)|$
6: $C_{TCM}^k(d_i) \leftarrow \hat{v}^k(r_i, t_i, \mathrm{d}_i)$
7: **end for**
8: **while** $d_i == d_{estimate}$ **do**
9: $X \leftarrow C_{TCM}^k(d_i)$ //sample from two flanks
10: $\hat{\beta} = (\hat{\beta}_0, \hat{\beta}_1, \hat{\beta}_2, \ldots, \hat{\beta}_m)^T = (X^T X)^{-1}(X^T V)$
11: $\hat{v}_{it} = \hat{\beta}_0 + \hat{\beta}_1 v_{1t} + \hat{\beta}_2 v_{2t} + \ldots + \hat{\beta}_m v_{mt}$ //sample in the same direction
12: **end while**
13: $E_{TCM} \leftarrow \hat{v}_{it}$
14: **return** E_{TCM}

are at least 10 common bits in our data set. It is not necessary to store these common bits. Therefore, we drop these common bits and move the effective bits right of the decimal point, and a float number is obtained. As pseudo code was used for the implementation, this procedure can be expressed as follows:

```
long    bits = geohash . long Value();
double    score = ((bits << c) >>> (64 − n + c))
    ∗ 2^(c − n);
```

where “geohash” is a class and “score” is the float number to be stored.

In order to simplify the calculation, the region segmentation is assigned into nine rectangular areas as show in Fig. 2, and there are eight neighboring regions around the estimation region. In step 9 of Algorithm 1, two groups are sampled from two flanks around the estimation region, and each sample group incorporates three rectangular areas. In step 11, the estimation is made by the two rectangular areas that are in the same direction as the estimation region.

4. System design

4.1. System requirements

The proposed real-time streaming data processing system is implemented using Apache Storm. Apache Storm is a free, open-source distributed real-time computation framework that makes it easy to reliably process unbounded streams of data. It has been used in a variety of applications and domains, including real-time analytics, online machine learning, continuous computation, and distributed remote procedure call (RPC). The underlying primitives in Storm provide for stream transformations that are “spouts” and “bolts.” A spout is a source of streams. For example, a spout may read tuples off of a sequence and emit them as a stream. A bolt consumes any number of input streams, does some processing, and possibly emits new streams. Networks of spouts and bolts are packaged into a “topology,” which is the top-level abstraction that you submit to Storm clusters for execution. A topology is a graph of stream transformations where each node is a spout or bolt, which is distributed over a cluster. Applications in Storm take the form of a Storm topology consisting of a set of interconnected processing nodes. In this environment, streaming data processing is scalable and capable of processing unbounded sequences of tuples in real time within sub-millisecond latencies.

The proposed real-time streaming data processing system inherits the following properties from the Storm model:

- Scalability: A cluster can be sized to handle arbitrary volumes of streaming data.
- Fault tolerance: In case of partial worker’s node failure, Storm will try to restart, and the worker will be restarted on another node.
- Low latency: Data streams are processed within milliseconds. Data exchanging between nodes is stored in an in-memory database but not persistent on disk, which represents a different trade-off in which a high write and read speed is achieved with the limitation of data sets. In addition, expanding distributed computing clusters will take the adaptive latency, which fulfills specific application constraints, such as adaptive traffic lights control in response to changes in practice.
- Flexibility: The topology-based programming model allows the incorporation of new operations for data streams by modifying the partial nodes.

4.2. Framework description

The architecture of the proposed real-time streaming data processing system is illustrated in Fig. 3.

The following are brief descriptions of the system components:

- Data set and message queue: This study experiments with a taxi trajectory data set that contains one-day trajectories of 7,648 taxis. The total number of points in this data set is about 18 million. For system validation, the study was set in an urban environment. To simulate the real-time traffic sensory data flow, the records in each data set are sent to the message queue in chronological order. By adjusting the send rate, we can evaluate the system performance.
- Storm topologies: This study design includes one kind of spout and three types of bolts, which are denoted as Spout A, Bolt B, Bolt C, and Bolt D in the figure. The Spout A reads traffic sensory data one after the other from the message queue in redis and its parallelism is three. Every raw sensory datum is a piece of text that is separated and serialized in the Bolt B and its parallelism is three. The velocity and vacancies estimations, including speed and samples, are computed in the Bolt C and its parallelism is

Please cite this article as: F. Wang et al., Estimating online vacancies in real-time road traffic monitoring with traffic sensor data stream, Ad Hoc Networks (2015), http://dx.doi.org/10.1016/j.adhoc.2015.07.003

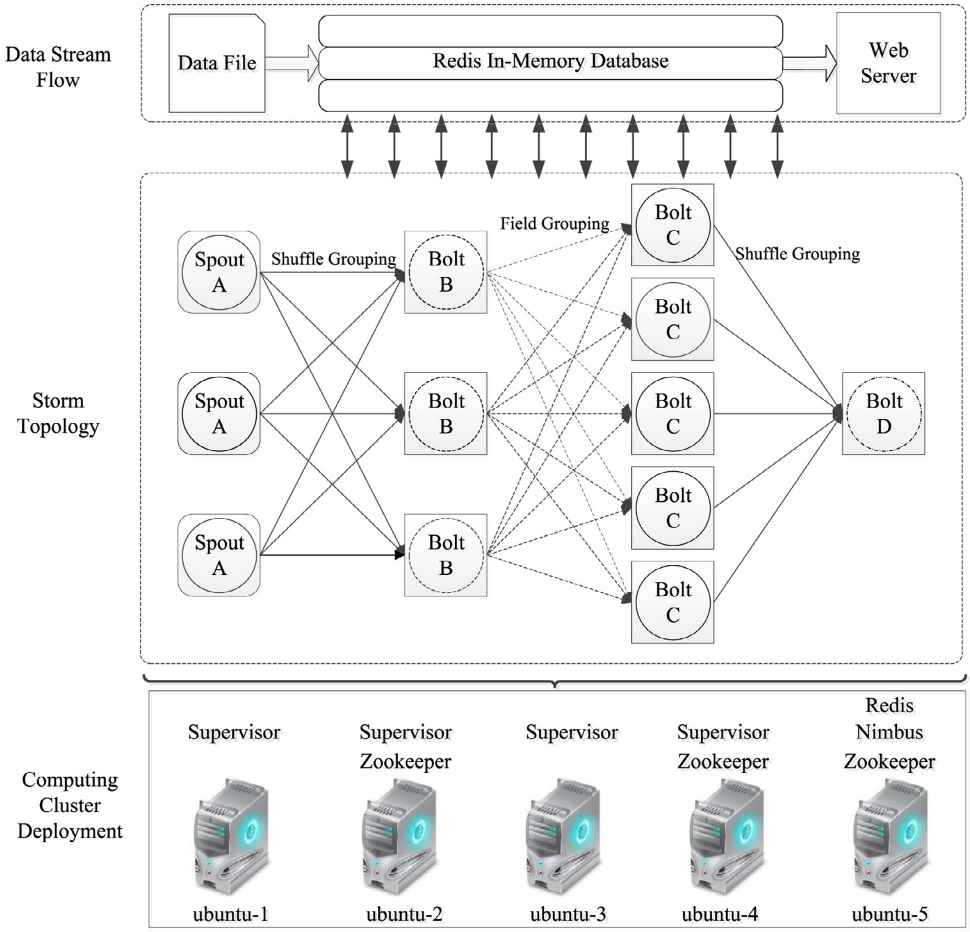

Fig. 3. Architecture of the proposed real-time streaming data processing system.

five. Global aggregation operation is executed in the Bolt D and its parallelism is one. The tuples between Spout A and Bolt B are split using shuffle grouping, which sends each tuple emitted by each source to a randomly chosen bolt, confirming that each consumer will receive the same number of tuples. The tuples between Bolt B and Bolt C are split by field grouping, which guarantees that a given set of values for a combination of fields is always sent to the same blot. The parallelism for each spout and bolt is also shown in Fig. 3.

- In-memory database: Real-time sensory data are collected at a rate of thousands of items per second, which is impossible with an on-disk SQL database. To handle a large write-heavy load with limited hardware and satisfy the requirements of low latency in real time computing, we leverage an in-memory database. Redis is an in-memory remote database that offers high performance, replication, and a unique data model. It can support five different types of data structures. Redis cluster allows convenient scaling to prototype a system of up to hundreds of gigabytes of data and millions of requests per second.

We deploy five PCs in the cluster, and an Ubuntu server operating system is installed on them. Finally, a Web-based output server provides the detailed results of real-time computing.

5. Experimental evaluation

5.1. Experimental results

The following experiments are conducted to evaluate the performance of the proposed algorithm for online vacancy estimation in real-time road traffic monitoring with a traffic sensor data stream. We first present the hardware environment and then introduce the algorithm results. Lastly, the performance results are presented and discussed.

The experiment cluster includes five ordinary PCs, which are 2-core 2.4-GHz desktop computers with 1-GB RAM and Ubuntu 14.04 LTS operating system. Although they are generic computers, they satisfy the requirements. The proposed real-time processing system depends not on sophisticated hardware but on the fine distributed computing framework and a good algorithm design, as the results demonstrate.

The trajectory data set contains one-day trajectories of 7,648 taxis. The total number of points in this data set is about 18 million. For system validation, the system was

	ID	CompanyID	VehicleSimID	GPSTime	GPSLongitude	GPSLatitude	CreateDate
1	131749	NULL	806584008859	2010-09-01 00:12:13.000	118.86234	31.940298	2010-09-01 00:12:13.030
2	151416	NULL	806584008859	2010-09-01 00:13:35.000	118.871376	31.941831	2010-09-01 00:13:35.217
3	171339	NULL	806584008859	2010-09-01 00:14:58.000	118.872408	31.941977	2010-09-01 00:14:58.653
4	192657	NULL	806584008859	2010-09-01 00:16:28.000	118.872409	31.941958	2010-09-01 00:16:28.843
5	243248	NULL	806584008859	2010-09-01 00:20:01.000	118.87242	31.941978	2010-09-01 00:20:01.827
6	263130	NULL	806584008859	2010-09-01 00:21:25.000	118.87241	31.941957	2010-09-01 00:21:25.797
7	283652	NULL	806584008859	2010-09-01 00:22:51.000	118.87243	31.941937	2010-09-01 00:22:51.467
8	303106	NULL	806584008859	2010-09-01 00:24:13.000	118.8724	31.941953	2010-09-01 00:24:13.000
9	322768	NULL	806584008859	2010-09-01 00:25:36.000	118.87245	31.941926	2010-09-01 00:25:36.217
10	343898	NULL	806584008859	2010-09-01 00:27:04.000	118.872452	31.942007	2010-09-01 00:27:04.623
11	362793	NULL	806584008859	2010-09-01 00:28:24.000	118.872421	31.941987	2010-09-01 00:28:23.997
12	474966	NULL	806584008859	2010-09-01 00:36:15.000	118.836111	31.933494	2010-09-01 00:36:15.170
13	549205	NULL	806584008859	2010-09-01 00:41:28.000	118.822078	31.93826	2010-09-01 00:41:28.780
14	571341	NULL	806584008859	2010-09-01 00:43:02.000	118.821087	31.945379	2010-09-01 00:43:02.263
15	591737	NULL	806584008859	2010-09-01 00:44:28.000	118.81844	31.945241	2010-09-01 00:44:28.280
16	611728	NULL	806584008859	2010-09-01 00:45:53.000	118.821129	31.946613	2010-09-01 00:45:53.170

Fig. 4. The sample trajectory data set.

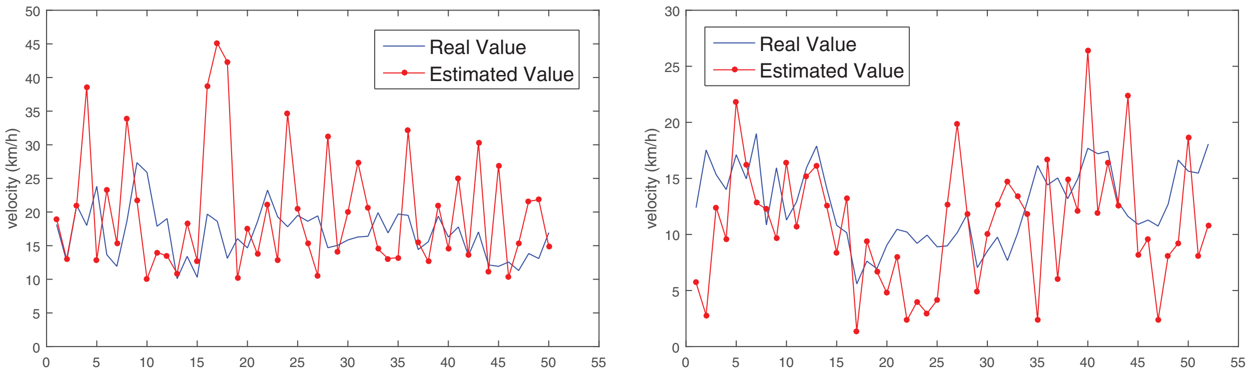

Fig. 5. Experiment results of region A in the north direction.

deployed in an urban environment. Every taxi sends an item of GPS information at intervals of 3–5 min. The sample trajectory data set is shown in Fig. 4.

Three sample regions (A, B, and C) are selected. Region A is in the suburbs and Region B and Region C are downtown areas. In Figs. 5–10 the abscissa axis represents the sample ID from the selected region, and the ordinate axis represents the average speed (in kilometers per hour) in the region. The solid line denotes the real values of average speed in the region and the solid line with dots denotes the estimation values. Every two figures form a set for comparison; they are samples from the same location but with different sizes of region segmentation. The first figure in a set has small region segmentation, and the second has region segmentation that is quadrupled the size of previous one.

In practice, the sensory data usually include error data as a result of obstructions to wireless communications such as canyons and underpasses. Therefore, we set abnormal data filtering in the experiment if the taxi speed is greater than a set threshold of a certain kilometer-per-hour speed. The reason for this is because the taxis are also not driving in accordance with the actual traffic conditions and the taxis at higher speed will better represent the actual traffic conditions. At any given time, the algorithm needs a sufficient number of samples to execute, so only the sample points that satisfy the algorithm conditions are listed in the figures.

Table 2
Average error rate.

Sample regions	A north	B north	B east	B west	C south	C east
Small scale	0.4347	0.2391	0.2493	0.2242	0.3666	0.2649
Big scale	0.3672	0.1256	0.1781	0.1671	0.1422	0.1524

Furthermore, we employ the "error rate" to envaluate the deviation of the result between the estimated values and the real ones. We define "error rate" as $\frac{|estimate-real|}{real}$, and we calculate the average error rates of studied regions which are depicted in Fig. 11 and Table 2. We can draw some conclusions from the result of the experiment. First, since Region A is in the suburbs and the other regions are downtown areas, more data are collected in Region B and Region C than in Region A. The result is a better description of regional traffic conditions, avoiding any errors associated with a single sample having an impact on the whole region. Second, the larger segmentation results in better estimates–again because larger segmentation results in the collection of more samples. Third, there is some relationship in traffic conditions among the closed areas. In our paper, we attempt to employ the multiple linear regression method to depict this relationship and obtain a better result in real time. The following section details the real-time performance in our experiment.

Please cite this article as: F. Wang et al., Estimating online vacancies in real-time road traffic monitoring with traffic sensor data stream, Ad Hoc Networks (2015), http://dx.doi.org/10.1016/j.adhoc.2015.07.003

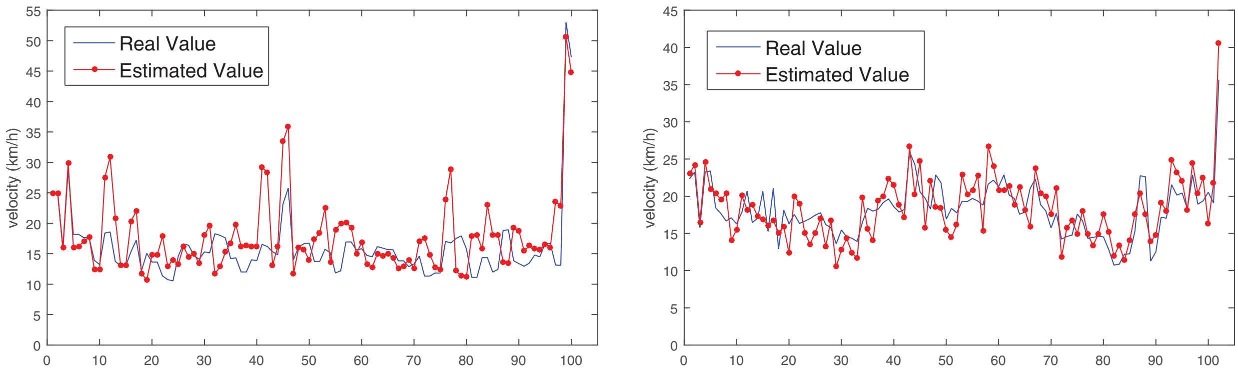

Fig. 6. Experiment results of region B in the north direction.

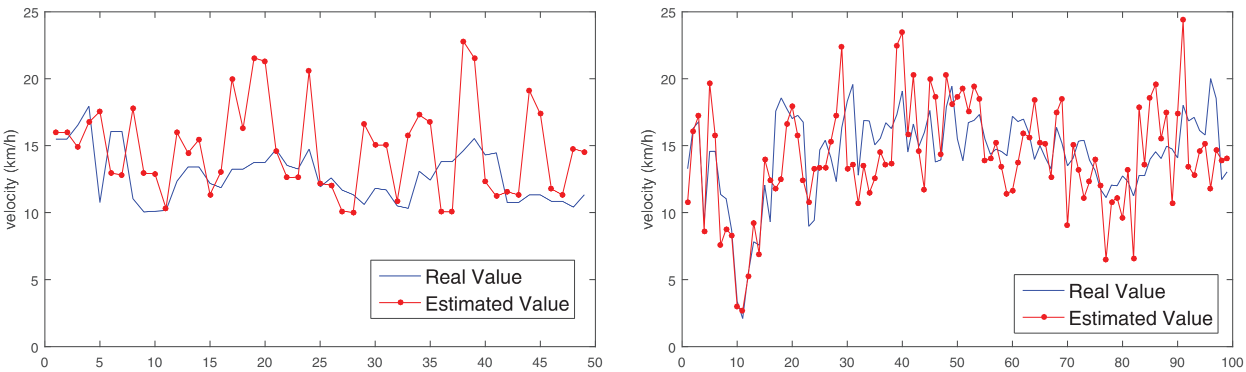

Fig. 7. Experiment results of region B in the east direction.

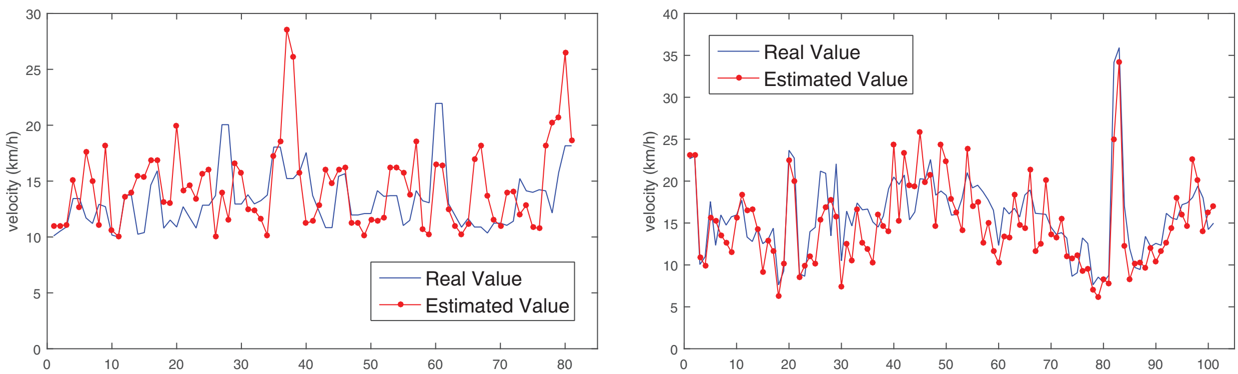

Fig. 8. Experiment results of region B in the west direction.

5.2. Performance analysis

In estimation calculation, we leverage the multiple linear regression approach. In a single estimation, since constant coefficients that are derived from sample segments and a final result are calculated, the time complexity is linear $O(n)$. The time complexity is still linear in terms of the speed of the data stream. Our approach improves computational complex from $O(n^2)$ to $O(n)$ in the phase of locating a set of neighboring nodes,which is compared to [29]. With respect to space complexity, the data of nine segments require storing for velocity estimation during one sampling. Therefore, the space complexity is a linear $O(n)$ in terms of sample segments as well. The space complexity will not increase during streaming, because the data of nigh segments do not need to stay in memory for long. For real-time estimation, the calculation must be done in a very short time. As the time of calculating the estimated value is constant and short, retrieving data remotely might be the only reason for a bottleneck. Therefore, we employ an in-memory remote database that offers high performance of hundreds of gigabytes of data and millions of requests per second.

In the experimental system, the real data in the data set are sent at an average speed of 1,500 items per second. The performance analysis of the system is detailed in Table 3.

Please cite this article as: F. Wang et al., Estimating online vacancies in real-time road traffic monitoring with traffic sensor data stream, Ad Hoc Networks (2015), http://dx.doi.org/10.1016/j.adhoc.2015.07.003

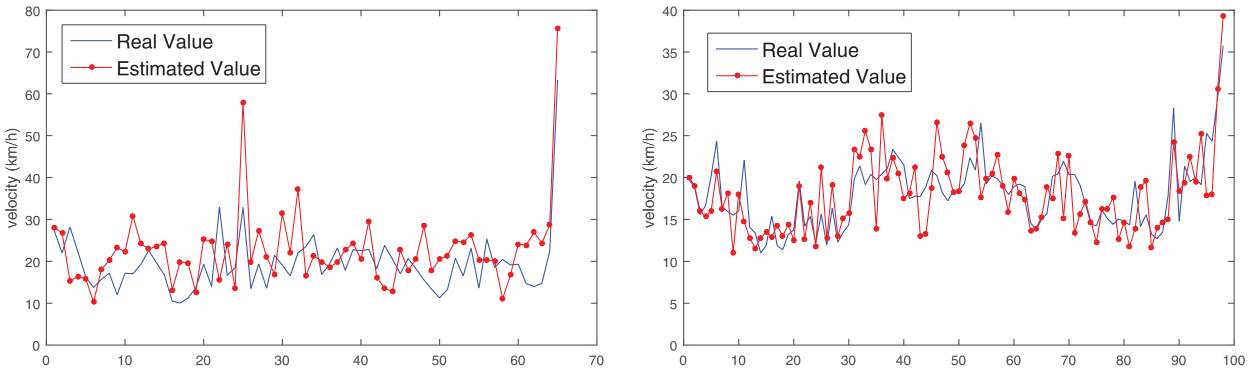

Fig. 9. Experiment results of region C in the south direction.

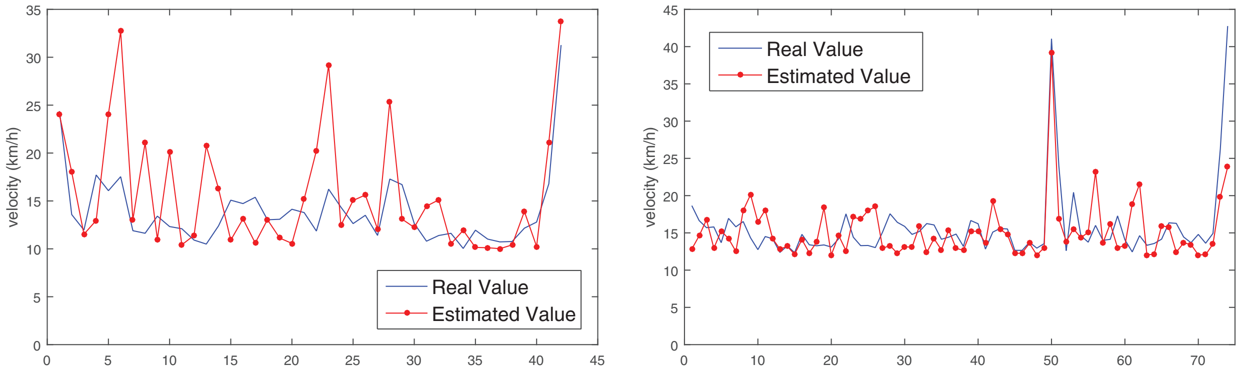

Fig. 10. Experiment results of region C in the east direction.

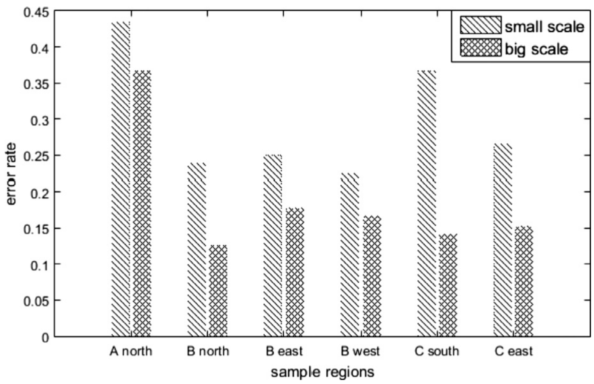

Fig. 11. Average error rate.

Table 3
Performance evaluation for real-time processing.

ID	Parallelism	Capacity	Execute latency(ms)	Process latency(ms)
Bolt B	3	0.177	0.341	0.327
Bolt C	5	0.442	2.322	2.305
Bolt D	1	0.151	0.317	0.304

Parallelism represents the number of executors assigned to the process ID. Capacity represents the metric of monitoring the performance to the process ID. It provides a panoramic overview of the percentage of time spent by the process ID in actually processing tuples in the last 10 min. If the value of capacity is close to 1, then the process ID is at capacity and the parallelism for the process ID must be increased. Otherwise, the at-capacity process ID will be a bottleneck for the system, because the source process emits the tuples at a faster rate, and most of them will time out, and the source process will need to re-emit the timed-out tuples. Process latency means the actual time (in milliseconds) taken to process a tuple. Execute latency is the sum of the processing time and the time consumed in sending the acknowledgment. In our tests, the proposed system can handle a thousand items of trajectory data per second with a cluster of five personal computers (PCs) with guarantee of low latency.As it is shown in Table 3, our delay is less than 1s. Our solution outperforms the method of [26] in which the average output delay is 82 s from the perspective of timeliness.

6. Conclusion and future work

This paper presents a new approach to traffic monitoring based on multiple linear regressions to estimate vacancies in real-time road traffic monitoring. The experiments for system validation are based on a probe vehicle (taxi) trajectory data set for an urban environment that contains one-day trajectories for over 7,000 taxis and results in about 18 million points.

The results suggest that the system can guarantee correctness and low latency with a cluster of five PCs, even if the traffic monitoring is in a large metropolis. The results also demonstrate that the proposed approach has effectively solved traffic condition estimation in cases where traffic trajectory sensor data streams exhibit inhomogeneous sparseness. This is performed by leveraging real-time data instead of historical data. Because the approach can still be effective even when the number of probe vehicles is not large, we hope that it will contribute to helping traffic managers acquire the newest traffic state by exploiting mass trajectory sensor data streams in real time.

We have identified two interesting avenues for future work: (1) extending our method to automatically select the scope of neighboring sample regions; and (2) enabling our method to summarize the different traffic patterns in different regions by applying online machine learning algorithms.

Acknowledgement

This work is funded by European Framework Program (FP7) under Grant No. FP7-PEOPLE-2011-IRSES, National Natural Science Foundation of China under Grant No. 61073009 & 61103197, National High Tech R&D Program 863 of China under Grant No. 2011AA010101, National Sci-Tech Support Plan of China under Grant No. 2014BAH02F03, National Sci-Tech Major Projects of China under Grant No. SinoProbe-09-01-03 & 2012ZX01039-004-04-3, Key Sci-Tech Program of Jilin Province of China under Grant No. 2011ZDGG007& 20150204035GX, and Fundamental Research Funds for Central Universities of China under Grant No. JCKY-QKJC46 & 2412015KJ005.

References

[1] L. Zhang, R. Wang, L. Cui, Real-time traffic monitoring with magnetic sensor networks, J. Inf. Sci. Eng. 27 (4) (2011) 1473–1486.
[2] T. Semertzidis, K. Dimitropoulos, A. Koutsia, N. Grammalidis, Video sensor network for real-time traffic monitoring and surveillance, Intell. Trans. Syst. IET 4 (2) (2010) 103–112, doi:10.1049/iet-its.2008.0092.
[3] D. Hounam, S. Baumgartner, K.H. Bethke, M. Gabele, E. Kemptner, D. Klement, G. Krieger, G. Rode, K. Wagel, An autonomous, non-cooperative, wide-area traffic monitoring system using space-based radar (tramrad), in: IEEE International Geoscience and Remote Sensing Symposium, IGARSS 2005, July 25 - July 29, vol. 4, Institute of Electrical and Electronics Engineers Inc., 2005, pp. 2917–2920, doi:10.1109/IGARSS.2005.1525679.
[4] N. Gallego, A. Mocholi, M. Menendez, R. Barrales, Traffic monitoring: Improving road safety using a laser scanner sensor, in: CERMA-Electronics Robotics and Automotive Mechanics Conference, September 22 - September 25, IEEE Computer Society, 2009, pp. 281–286, doi:10.1109/CERMA.2009.11.
[5] H. Zhao, C. Wang, W. Yao, J. Cui, H. Zha, Vehicle trajectory collection using on-board multi-lidars for driving behavior analysis, in: 7th International Conference on Intelligent Systems and Knowledge Engineering, ISKE 2012, December 15 - December 17, in: Advances in Intelligent Systems and Computing, vol. 214, Springer Verlag, 2012, pp. 659–671, doi:10.1007/978-3-642-37832-4_60. Compilation and indexing terms, Copyright 2014 Elsevier Inc. 20134917050959 Data processing systems Driving behavior Lidar sensing Moving object detection and tracking Processing approach Synchronized motion Trajectory extraction Vehicle trajectories
[6] Y. Iwasaki, M. Misumi, T. Nakamiya, Robust vehicle detection under various environmental conditions using an infrared thermal camera and its application to road traffic flow monitoring, Sensors (Switzerland) 13 (6) (2013) 7756–7773, doi:10.3390/s130607756.
[7] V. Agarwal, N.V. Murali, C. Chandramouli, A cost-effective ultrasonic sensor-based driver-assistance system for congested traffic conditions, IEEE Trans. Intell. Transp. Syst. 10 (3) (2009) 486–498, doi:10.1109/TITS.2009.2026671.
[8] K.H. Bethke, S. Baumgartner, M. Gabele, Airborne road traffic monitoring with radar, in: 14th World Congress on Intelligent Transport Systems, ITS 2007, October 9 - October 13, vol. 3, Curran Associates Inc., 2007, pp. 1895–1900.
[9] S.O. Larsen, A.B. Salberg, L. Eikvil, Automatic system for operational traffic monitoring using very-high-resolution satellite imagery, Int. J. Remote Sens. 34 (13) (2013) 4850–4870, doi:10.1080/01431161.2013.782708.
[10] S.M.M. Kahaki, M. Fathy, M. Ganj, Road-following and traffic analysis using high-resolution remote sensing imagery, in: 3rd International Workshop on Intelligent Vehicle Controls and Intelligent Transportation Systems - IVC and ITS 2009 In Conjunction with ICINCO 2009, July 1, INSTICC Press, 2009, pp. 133–142.
[11] M. Eslami, K. Faez, Automatic traffic monitoring using satellite images, in: 2nd International Conference on Computer Engineering and Technology, ICCET 2010, April 16, 2010 - April 18, 2010, vol. 6, IEEE Computer Society, 2010, pp. V6130–V6135, doi:10.1109/ICCET.2010.5486343.
[12] F. Calabrese, M. Colonna, P. Lovisolo, D. Parata, C. Ratti, Real-time urban monitoring using cell phones: A case study in rome, IEEE Trans. Intell. Transp. Syst. 12 (1) (2011) 141–151, doi:10.1109/tits.2010.2074196.
[13] N. Caceres, J.P. Wideberg, G. Benitez, Review of traffic data estimations extracted from cellular networks, IET Intell. Transp. Syst. 2 (3) (2008) 179–192, doi:10.1049/iet-its:20080003.
[14] J.C. Herrera, D.B. Work, R. Herring, X.G. Ban, Q. Jacobson, A.M. Bayen, Evaluation of traffic data obtained via gps-enabled mobile phones: The mobile century field experiment, Transp. Res. Part C-Emerging Technol. 18 (4) (2010) 568–583, doi:10.1016/j.trc.2009.10.006.
[15] A. Thiagarajan, L. Ravindranath, K. LaCurts, S. Madden, H. Balakrishnan, S. Toledo, J. Eriksson, Vtrack: Accurate, energy-aware road traffic delay estimation using mobile phones, in: 7th ACM Conference on Embedded Networked Sensor Systems, SenSys 2009, November 4 - November 6, Association for Computing Machinery, 2009, pp. 85–98, doi:10.1145/1644038.1644048.
[16] H. Su, K. Zheng, J. Huang, H. Jeung, L. Chen, X. Zhou, Crowdplanner: A crowd-based route recommendation system, in: Data Engineering (ICDE), 2014 IEEE 30th International Conference on, 2014, pp. 1144–1155, doi:10.1109/ICDE.2014.6816730.
[17] J. Yu, K.H. Low, A. Oran, P. Jaillet, Hierarchical bayesian nonparametric approach to modeling and learning the wisdom of crowds of urban traffic route planning agents, in: IEEE/WIC/ACM International Conference on Intelligent Agent Technology, IAT 2012, December 4 - December 7, vol. 2, IEEE Computer Society, 2012, pp. 478–485, doi:10.1109/WI-IAT.2012.216.
[18] R. Frank, M. Mouton, T. Engel, Towards collaborative traffic sensing using mobile phones (poster), in: 2012 IEEE Vehicular Networking Conference, VNC 2012, November 14- November 16, IEEE Computer Society, 2012, pp. 115–120, doi:10.1109/VNC.2012.6407419.
[19] U. Lee, M. Gerla, A survey of urban vehicular sensing platforms, Comput. Networks 54 (4) (2010) 527–544. http://dx.doi.org/10.1016/j.comnet.2009.07.011.
[20] J. Zhou, C.L. Philip Chen, L. Chen, A small-scale traffic monitoring system in urban wireless sensor networks, in: Proceedings of the IEEE International Conference on Systems, Man, and Cybernetics, SMC 2013, October 13, 2013 - October 16, 2013, IEEE Computer Society, 2013, pp. 4929–4934, doi:10.1109/SMC.2013.842.
[21] A. Hassanzadeh, A. Altaweel, R. Stoleru, Traffic-and-resource-aware intrusion detection in wireless mesh networks, Ad Hoc Networks 21 (2014) 18–41, doi:10.1016/j.adhoc.2014.04.009.
[22] G. Iannizzotto, F. La Rosa, L. Lo Bello, A wireless sensor network for distributed autonomous traffic monitoring, in: 3rd International Conference on Human System Interaction, HSI'2010, May 13, 2010 - May 15, 2010, IEEE Computer Society, 2010, pp. 612–619, doi:10.1109/HSI.2010.5514504.
[23] M. Whaiduzzaman, M. Sookhak, A. Gani, R. Buyya, A survey on vehicular cloud computing, J. Network Comput Appl. 40 (1) (2014) 325–344, doi:10.1016/j.jnca.2013.08.004.
[24] M. Gerla, Vehicular cloud computing, in: 11th Annual Mediterranean Ad Hoc Networking Workshop, Med-Hoc-Net 2012, June 19, 2012 - June 22, 2012, IEEE Computer Society, 2012, pp. 152–155, doi:10.1109/MedHocNet.2012.6257116.
[25] L. Gu, D. Zeng, S. Guo, Vehicular cloud computing: A survey, in: 2013 IEEE Globecom Workshops, GC Wkshps 2013, December 9, 2013 - December 13, 2013, IEEE Computer Society, 2013, pp. 403–407, doi:10.1109/GLOCOMW.2013.6825021.
[26] C.Y. Goh, J. Dauwels, N. Mitrovic, M.T. Asif, A. Oran, P. Jaillet, Online map-matching based on hidden markov model for real-time traffic sensing applications, in: Intelligent Transportation Systems (ITSC), 2012 15th International IEEE Conference on, 2012, pp. 776–781, doi:10.1109/ITSC.2012.6338627.
[27] Y. Zhu, Z. Li, H. Zhu, M. Li, Q. Zhang, A compressive sensing approach to urban traffic estimation with probe vehicles, IEEE Trans. Mobile Comput. 12 (11) (2013) 2289–2302, doi:10.1109/TMC.2012.205.

[28] D.L. Donoho, Compressed sensing, IEEE Trans. Inf. Theory 52 (4) (2006) 1289–1306, doi:10.1109/TIT.2006.871582. Compilation and indexing terms, Copyright 2014 Elsevier Inc. 2006169826702 Adaptive sampling Banana spaces Basis pursuit Compressed sensing Gel'fand n-widths Information-based complexity Minimum decomposition Optimal recovery Quotient-of-a-Subspace theorem Sparse solution
[29] Z. Shan, D. Zhao, Y. Xia, Urban road traffic speed estimation for missing probe vehicle data based on multiple linear regression model, in: Intelligent Transportation Systems - (ITSC), 2013 16th International IEEE Conference on, 2013, pp. 118–123. doi: 10.1109ITSC.2013.6728220.

Feng Wang received his B.E degree in Information Security in 2009 in Harbin Institute of Technology, followed by his M.S degree in Network and Information Security in 2012 in Jilin University. He is currently a PhD candidate in the College of Computer Science and Technology, Jilin University. His research interests are in computer networks, information security, the Internet of Things and Cyber Physical Systems.

Liang Hu received his M.S. and Ph.D. degrees in Computer Science from Jilin University in 1993 and 1999. Currently, he is a professor and doctoral supervisor at the College of Computer Science and Technology, Jilin University, China. His research areas are Network Security and Distributed Computing, including theories, models and algorithms of PKI/IBE, IDS/IPS, and Grid Computing. He is a member of the China Computer Federation.

Dongdai Zhou obtained his BEng and MSc, both from Changchun University of Science and Technology, China, in 1992 and 1997, respectively. He received his PhD degree from the Jilin University, China, in 2001. He is currently a professor at School of Software, Northeast Normal University, China. His research interests include software architecture and software code auto-generation, especially architecture design and codeless development of e-learning software.

Rui Sun received his B.E degree in Software Engineering in 2014 in College of Software, Jilin University. He is currently a post-graduate student in the College of Computer Science and Technology, Jilin University. His research interests are in big data, machine learning and artificial intelligence.

Jiejun Hu received his B.E degree in in 2012 in Changchun University of Technology. She is a PhD candidate of the college of College of Computer Science and Technology, Jilin University, China. His current research interests include cloud computing and distributed storage.

Kuo Zhao received his B.E degree in Computer Software in 2001 in Jilin University, followed by his M.S degree in Computer Architecture in 2004 and Ph.D. in Computer Software and Theory from the same university in 2008. He is currently an Assoicate Professor in the College of Computer Science and Technology, Jilin University. His research interests are in operating systems, computer networks and information security.