

Evaluation of Data Reduction Techniques for Vehicle to Infrastructure Communication Saving Purposes *

Luca Carafoli, Federica Mandreoli,
Riccardo Martoglia
DII - University of Modena and Reggio Emilia
Modena, Italy
<name.surname>@unimore.it

Wilma Penzo
DEIS - University of Bologna
Bologna, Italy
wilma.penzo@unibo.it

ABSTRACT

In this paper we investigate the employment of different data reduction techniques to minimize V2I communication in an Intelligent Transportation System (ITS). We consider the context of the PEGASUS Project, where vehicles are equipped with sensor-based devices able to compute and communicate to a Control Centre (CC) information like vehicle's position and speed. The CC relies on a general-purpose data management module that supports the execution of continuous queries as well as standard SQL one-time queries on the collected data to provide various infomobility services.

The paper explores two categories of data reduction techniques: independent techniques, where vehicles autonomously send data to the CC, and information-need techniques, where data is sent by taking into account additional data received from the CC. The paper discusses and implements the technical changes needed in the CC to support the required infomobility services under the reduced availability of data. All the investigated techniques have been extensively evaluated in a variety of traffic scenarios.

Keywords

Data reduction techniques, Intelligent Transportation Systems, Communication-saving, Technique evaluation and assessment, V2I

1. INTRODUCTION

One of the biggest emerging challenges that modern day cities have to face is traffic congestion in urban areas. This

*This work is partially funded by the Italian Council Industria 2015 PEGASUS Project, <http://pegasus.octotelematics.com>. We would also like to acknowledge Marcello Pietri for the initial system implementation and testing.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

IDEAS12 2012, August 8-10, Prague [Czech Republic]

Copyright ©2012 ACM 978-1-4503-1234-9/12/08 ...\$15.00.

is acknowledged as a crucial problem of modern society and many investments are going to be made in the direction of the so-called smart cities, whose objectives include the improvement of citizens' life-style by pushing innovation in urban mobility. The Horizon 2020 EU Programme specifically addresses sustainable urban transportation as a societal challenge to be tackled. The implementation of mechanisms to improve the efficiency of traffic flows requires expensive investments, mainly for the deployment of a communication infrastructure and for its maintenance. For these reasons, existing systems for collecting and disseminating traffic information tend to cover only selected areas (usually, highways).

To hold down the costs of infrastructure-based traffic monitoring systems, Floating Car Data (FCD) techniques [13] are employed as a much more economic alternative to the installation and maintenance of fixed-point traffic sensors, such as cameras and loop detectors. FCD assumes vehicles are equipped with sensor-based devices able to compute information like vehicle's position and speed, and that communicate this data to a server that processes it to update the current traffic situation. This updated information is then broadcasted to all vehicles to make them plan their travel routes at best. While providing a viable solution for traffic monitoring at a larger scale, FCD techniques pose new challenges as to managing data coming from a huge number of vehicles. Such heavy traffic load entails costs at various levels.

First, it strains Vehicle-to-Infrastructure (V2I) communication to the limit.

Second, the server hosts a database that have to provide for the real-time storage and processing of high volumes of traffic information. Since database management systems typically do not efficiently support high update rates, these systems may suffer from unbounded delay accumulation and monitoring accuracy decay.

Third, frequent sensing and processing of data, as well as the frequent communications to the server, definitely shorten the power autonomy of vehicles' devices.

Paper Contributions

In order to cope with these efficiency issues, a viable solution stands in employing data volumes reduction techniques, that could/should be conveniently exploited both to reduce the V2I payload, and, as a consequence, to lighten the server's burden, as well as to save battery-power at vehicles' de-

VICES. However, data acquisition policies should both meet the need of reducing the data to be transmitted and processed, and guarantee valuable, reliable, and timely infomobility services.

This paper presents the results achieved in this research field within the PEGASUS Project funded by the Italian Ministry of Economic Development within the Industria 2015 Program. The major objective of the Project is to provide an Intelligent Transportation System (ITS) that provides mobility solutions for an efficient and effective traffic management, with the aim of improving the safety and the efficiency of vehicles and goods flows, as well as to make transportation a smart experience.

In PEGASUS, vehicles are equipped with sensor-based devices, called On-Board Units (OBUs) that, beyond providing simple functionalities like a GPS service and GPRS communications, are enabled to do some computation on the data. Vehicles interact with a Control Centre (CC) that relies on a general-purpose data management module that supports the execution of continuous queries as well as standard SQL one-time queries on the collected data to provide various infomobility services, spanning from traffic monitoring to location-based ones [10]. Because of the project's peculiar requirements, a large amount of available data reduction techniques can not be applied tout-court, since these are tailored for specific purposes: either for traffic monitoring or for location-based services. Furthermore, these techniques are devised for, and tested in, typical US traffic scenarios, usually made up of high-flow roads with a low number of road intersections, like highways. Instead, the traffic scenarios envisioned in PEGASUS are the more disparate, ranging from highway roads to urban areas. Urban areas, which represent most European transport realities, are indeed characterized by opposite conditions: frequent vehicles' start-and-stop's and several road intersections. Under these multi-faceted conditions, one of the objectives of the PEGASUS Project is to arrange vehicles to implement data reduction techniques to limit the number of reports to be sent to the CC, while guaranteeing the provision of a variety of infomobility services, in a variety of traffic scenarios.

In this paper, we investigate a set of data reduction techniques with the aim of identifying those that perform better for a general-purpose service provision, as well as those that better fit specific CC's workloads. For this purpose, we consider two categories: independent techniques, where vehicles autonomously send data to the CC, and information-need techniques, where data is sent by taking into account additional data received from the CC. The former are attracting since they do not require additional communication costs due to CC's data transmission. The latter are interesting because they implement sophisticated mechanisms that are much more effective for specific traffic monitoring services. These could be profitably employed under specific CC's workloads. We consider two different kinds of data requests: aggregate-based information, like the average speed per road segment, used for instance to determine congested routes (example of traffic monitoring service), and item-based information, such as the vehicle's position at a given instant, needed for instance to support the tracking of vehicles (example of location-based service). The paper discusses and implements the technical changes needed at the CC to support the required infomobility services under the reduced availability of data. All investigated techniques

have been extensively evaluated in different traffic scenarios specifically designed with the traffic micro-simulator VIS-SIM¹.

The paper is organized as follows. Section 2 presents literature review with regard to current communication-saving approaches. The reference architecture envisioned in the PEGASUS Project is shown in Section 3, whereas Section 4 introduces the data reduction techniques we examined, together with the technical changes needed for their employment at server level. Section 5 gives details about the traffic scenarios used in the evaluation of the data reduction techniques, while experimental results are shown and discussed in Section 6. Finally, in Section 7 final discussion and conclusions are drawn.

2. RELATED WORKS

Several approaches have been proposed to reduce transmission costs in V2I communication systems.

Some research efforts have been made along the realization of Vehicle-to-Vehicle (V2V) communication systems in order to transmit aggregated data values rather than entire datasets [7]. These distributed solutions exploits the capability of vehicles to collect traffic information and to disseminate by communicating directly with other vehicles via WiFi link. These solutions, while exhibiting several advantages, like zero-additional infrastructure and wide coverage, present, however, in the context of the PEGASUS Project, severe limitations as to the stability of protocols, that fall short of expectations about reliability. Besides, the complete replacement of V2I with V2V communications, as proposed in most of the available literature [4, 6, 7], is still infeasible in a European traffic scenario because of the still low penetration of OBU-equipped vehicles that can not self-organize into significant clusters.

As to data reduction mechanisms employed by vehicles to minimize V2I communication, earlier basic techniques rely on time/space sampling of data [13, 5, 14]. These independent techniques are widely used in the literature as they are simple to implement and perform very well under different service requirements. In addition, unlike the scenarios envisioned in these works, vehicles in PEGASUS are enabled to store road network maps, and thus can exploit this information to employ map-based sampling techniques like the one we explored in our work.

With regard to the information-need techniques we investigated in this paper, the Deterministic Information Need Policy we implemented is an adaptation of the technique proposed in [8] where, in addition to vehicle reports notifying velocity variation over a given threshold, also an estimate of vehicle reports having velocity values under the threshold is taken into account to determine a more accurate computation of the average speed in a road segment. The randomized policy proposed in [12] is a refinement of [8], in that vehicles transmit their speed (if exceeding the given threshold) with some given transmission probability P . A research issue is how to determine P so that the server can achieve maximum accuracy on the broadcasted data with a minimum number of transmissions. However, both the techniques in [8] and [12] send an incomplete and potentially skewed version of data to the server. Furthermore, as asserted in [1], these policies send more data that

¹www.vissim.com

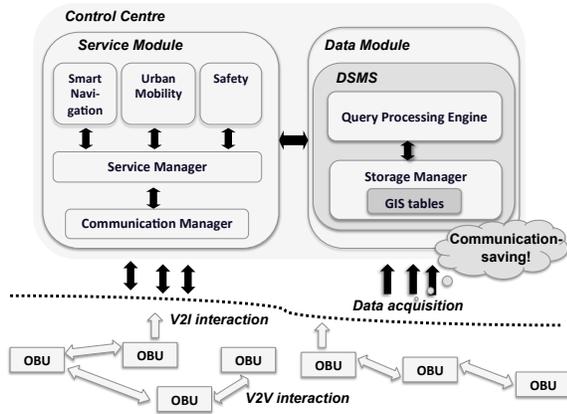


Figure 1: PEGASUS Architecture

actually needed. As to this point, in [1] authors observe that statistics can be used to determine the average speed within an error range with a specified confidence percentage. The paper presents a flow-based policy to compute the size of the sample that guarantees the given confidence. In this paper we have implemented also this technique as a valuable mechanism to be tested in the PEGASUS Project traffic scenarios.

Additional techniques to reduce V2I transmission costs consider the compression of trajectories [3, 11]. The work in [3] exploits server as well as client-side algorithms to reduce the number of updates for tracking moving objects. The underlying assumption is that precise locations of moving objects are not needed for most tracking location-based services, although a minimum accuracy should be guaranteed. The introduced techniques based on movement prediction, that is done on the clients and on the server, simultaneously. Whenever the difference between the actual position and the prediction done at the client exceeds a given threshold implied by the predefined accuracy, the client sends an update to the server. These techniques can be profitably employed in the specific context of location-based services.

Other approaches, orthogonal to the employment of data reduction techniques applied by vehicles, concern the adoption of server-side load-shedding techniques. In traffic monitoring applications, a huge number of FCD records sent by vehicles may exceed the processing capabilities of the system, thus increasing the processing delay and compromising the timeliness of the service. These techniques drop unprocessed records to reduce the system load when the demands can not be met by the system [9]. [2] presents algorithms that determine at what points in a query plan should load shedding be performed and what amount of load should be shed at each point in order to minimize the degree of inaccuracy introduced into query answers. An interesting research direction would be the adaptation of these techniques so as to be performed by vehicles, in a distributed fashion.

3. THE PEGASUS SCENARIO

The ITS scenario considered in the PEGASUS project is made up of On-Board Units (OBUs), i.e. sensor-based devices installed on vehicles, which interact with the Control Centre (CC), as shown in Figure 1. The CC and OBUs have the same map database that consists of road segments, each

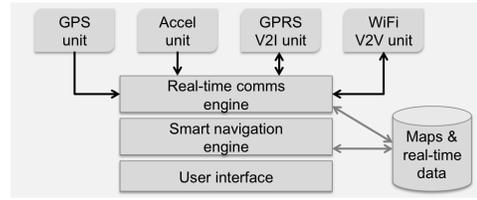


Figure 2: On-Board Unit (OBU) Architecture

identified by a unique id. In this section we will introduce their architectures and the kinds of data requests we address in this paper.

3.1 On-Board Units (OBUs)

Each OBU is a small and "intelligent" mobile computing device that acquires data through the GPS and the accelerometer unit and performs real-time GPRS or WIFI communications (see Figure 2). To this end, the I/O units interact with the *real-time communication engine*, which orchestrates all the data acquisition and communication operations. The device is also equipped with a flash-memory where road network maps are stored together with the real-time information acquired from the CC, as segment average speeds, traffic condition updates, and so on. Such storage is accessed by the smart navigation engine in order to provide the driver with the required services.

Each OBU is univocally identified by an identifier ID . Every time unit t OBU sensors report their data and a new data unit (ID, p, s, t, seg) is thus available for transmission, where $p = (x, y)$ is a point that represents the OBU position in a two dimensional Euclidean space, s is the speed and seg is the ID of the segment that contains the OBU's position p in the reference map.

Finally, disconnections are dealt with by network mechanism in such a way that the CC has always an almost correct picture of on-line OBUs.

3.2 The Control Centre (CC)

The main goal of the Control Centre is to collect and analyze the large amount of geo-located stream items coming from the OBUs and to provide end users with various services to enhance their mobility. It consists of two modules: the Data Module and the Service Module.

The Service Module interacts with the Data Module to obtain information which are used for satisfying service users' requests and with OBUs through the Communication Manager.

The Data Module is responsible for stream items acquisition, storage and manipulation. In the PEGASUS project, it must be able to manage both fresh data, which must be timely available, and historical data, which can be stored on disks and cached when needed. The querying support it offers concern both continuous queries (CQs) that are evaluated over the incoming data streams and standard SQL One-Time Queries (OTQs).

Most ITS services are based on one or more continuous queries. For example, the following CQ calculates the average speed of each segment every minute:

```
SELECT o.segment, AVG(speed)
FROM obu o[WINDOW INTERVAL "1" MINUTE]
GROUP BY o.segment
```

SAMPLE INTERVAL "1" MINUTE

where `obu(ID,x,y,speed,time,segment)` is the table that stores the vehicle reports received from OBUs, `WINDOW INTERVAL` is the sliding window size, i.e. the size of the temporal window that is used to capture the most recent portion of the `obu` table, and `SAMPLE INTERVAL` is the update rate, i.e. the time interval between subsequent query runs. Another sample of typical CQs is the following query that detects stopped cars in the road network:

```
SELECT DISTINCT ID
FROM obu o[WINDOW INTERVAL "1" MINUTE]
WHERE o.speed=0
SAMPLE INTERVAL "1" MINUTE
```

3.3 Info-mobility data requests

The Service Module usually interacts with the Data Module by submitting queries that falls within two main categories:

Aggregate-based information: the desired value is the aggregation of the stream items in a fixed sliding temporal window. The most common samples in this category are the average speed per segment and the number of vehicles per segment.

Item-based information the desired value is directly derivable from the item itself. Straightforward samples of this kind of requests are the position or the speed of a vehicle in a given instant.

In the following, we will focus on two popular data requests, one for each category.

The former is the average speed per segment, it represents a fundamental information for **Traffic-Monitoring Services (TMSs)**. Its relational algebraic expression on the `obu` table at time t is:

$$\bar{s}_{t,\tau} = \pi_{\text{segment,avg(speed)}}(\sigma_{\text{time} \in [t-\tau,t]}(\text{obu})) \quad (1)$$

where τ is the window size. When it is focused on a specific segment seg , the formula is computed on the seg tuples only and it is denoted as $\bar{s}(seg)_{t,\tau}$.

The latter is the current vehicle position, it represents the base information for **Location-Based Services (LBSs)**:

$$p(ID)_t = \pi_{ID,x,y}(\sigma_{\text{time}=t}(\text{obu})) \quad (2)$$

Similarly to the previous case, when it is focused on a specific OBU ID , the formula is computed on the ID tuples only and it is denoted as $p(ID)_t$.

4. A SURVEY OF DATA REDUCTION TECHNIQUES

In this section we introduce the data reduction techniques we considered in the ITS context for the purposes described above. All of them are OBU-side techniques that aim at reducing the number of data update communications. On one hand, they bring the well-known benefits of reducing the communications costs, the server side update costs and the client side costs. On the other hand, the CC receives fewer vehicle reports and it must implement specific solutions to

accurately solve service data requests also when the number of table updates is reduced.

Before discussing the specific techniques, we start distinguishing them in two main categories, the former can be used for both data requests while the latter for TMSs only:

Independent techniques the vehicle autonomously decides when to issue updates to the server. The techniques we considered in this context are the Simple Sampling, the Space Sampling and the Map Based Sampling.

Information-Need techniques the vehicle decides when to transmit on the basis of specific information made available by the CC. The techniques we considered in this category are the Deterministic Information Need and the Flow Information Need technique.

4.1 Independent Techniques

When an independent technique is adopted, the CC always assumes that the vehicle moves linearly and with the constant acceleration derivable from the most recent updates. In this case, TMS data requests are answered by using the position reports that have been sent to the CC only and actually stored in the `obu` table. LBS data requests, instead, are answered by predicting the current vehicle positions in the following way.

Let $p(ID)_{t_{n-1}}$ and $p(ID)_{t_{n-2}}$ be the last two vehicle positions for a given vehicle ID . In order to predict its position at time t_n , $p(ID)_{t_n}$, we first compute the angular coefficient m of the segment joining them. Then, by applying the motion equations we obtain the distance $dist$ between $p(ID)_{t_{n-1}}$ and $p(ID)_{t_n}$:

$$dist = s_{t_{n-1}} * (t_n - t_{n-1}) + \frac{1}{2} * a * (t_n - t_{n-1})^2 \quad (3)$$

where a is the acceleration derived from the last two speeds, $s_{t_{n-1}}$ and $s_{t_{n-2}}$. Then, the two coordinates are computed as: $p(ID)_{t_n}.x = dist * \cos(\arctan(m)) + p(ID)_{t_{n-1}}.x$ and $p(ID)_{t_n}.y = dist * \sin(\arctan(m)) + p(ID)_{t_{n-1}}.y$

The various policies are different as to the way they determine if an update should be sent to the CC. In this way, they provide different sets of scattered points on the road network thus inducing different vehicle movement predictions and average speed values.

4.1.1 Simple Sampling

Simple sampling is the simplest policy we considered and thus represent our baseline. According to this approach, a vehicle transmits a data report at a fixed time rate T (see Algorithm 1, *now* is the current timestamp). This policy has been considered in other ITS approaches [14, 5] for purposes different from data reduction.

Using this policy, the movement of the vehicle is represented as a constant time "jumping vector" that is completely independent from the actual vehicle movements and the road network. Moreover, it can cause several useless updates in different common situation such as stopping at a red light or traffic jam.

4.1.2 Space Sampling

This policy sends a position report to the CC when the vehicle covers a fixed distance D . Therefore, differently from simple sampling, a vehicle transmit only if it is actually traveling. This policy represents vehicle movements as constant

Algorithm 1 SimpleSampling(T)

```
1:  $t_{prev} = now$ ;
2: send();
3: while true do
4:   if ( $now == t_{prev} + T$ ) then
5:      $t_{prev} = now$ ;
6:     send();
7:   end if
8: end while
```

distance "jumping vectors". On the other hand, in a urban scenario where vehicles stop very frequently, the accuracy of the predicted positions can be very low. In order to improve accuracy, this policy allows vehicles to send a position report also when they stop or restart. The OBU-side cycle implementing this policy is shown in Alg. 2 (*stopDetection()* and *startDetection()* are system functions detecting whether the vehicle has just stopped or restarted, respectively).

Algorithm 2 SpaceSampling($D, stop$)

```
1:  $\triangleright stop$ : a flag to indicate whether the vehicle transmits
   when it stops or restarts
2:  $p_{prev} = detect(p)$ ;  $\triangleright$  The vehicle detects its current
   position
3: send();
4:  $dist=0$ ;
5: while true do
6:    $dist = dist + (detect(p) - p_{prev})$ ;
7:    $p_{prev} = detect(p)$ ;
8:   if  $dist \geq D \vee (stop \wedge (stopDetection() \vee$ 
    $startDetection()))$  then
9:     send();
10:  end if
11: end while
```

4.1.3 Map Based Sampling

Algorithm 3 Basic Functions

```
1: function ISSEND( $stop, turn, seg_{now}, seg_{prev}$ )
2:    $\triangleright$  Vehicle sets  $turn$  as true if it sends
   after a direction change;  $stop$  is a flag to indicate if the
   vehicle sends information after a stopDetection() and a
   startDetection().
3:   if  $stop \wedge (stopDetection() \vee startDetection())$  then
4:     return true;
5:   end if
6:   if  $turn \wedge isTurn()$  then
7:     return true;
8:   end if
9:   if  $!turn \wedge isChangeSeg(seg_{now}, seg_{prev})$  then
10:    return true;
11:  end if
12:  return false;
13: end function
```

The map based sampling is deeply different from previous techniques because it depends on the road network. The basic idea behind this approach is that the vehicle moves on map segments. Therefore, in its simplest version, the vehicle sends a position report when it reaches the end of a map

segment. This policy thus represents the vehicle movement as a set of road segments with positions on them. In this way it can be conceived as a simplified version of the segment-based policy, introduced in [3], where no interaction of CC to OBUs is required.

Similarly to space sampling, also map based sampling allows vehicles to inform the CC about their stops and restarts. Furthermore, instead of relying on map segments, the vehicle can choose to transmit when it reaches the end of a road segment. In particular road segment it's different from a map segment as the former is a straight stretch of road where as the latter is an arbitrary segment as provided by the map database.

To detect the end of a map or road segment and stops and restarts, we implements the ISSEND function, shown in Alg. 3. The function return *true* if the vehicle has just travelled on new map segment or, if *turn* is set *true*, when has just ended a road segment (line 6-11). Moreover, when the parameter *stop* is equal to *true*, it returns *true* both if it has just stopped and if it has just restarted after a stop (lines 3-5). To detect the change of map segment, we implement the function *isChangeSeg()* that returns *true* if the vehicle has changed the map segment; instead in line 8 *isTurn()* returns *true* if the vehicle has just reached the end of a road segment.

In Alg. 4, it is implemented the Map Based Sampling algorithm. Line 1 performs initialization, then the main cycle in lines 2-8 continues until the vehicle is switched off. These lines repeatedly detect the map segment where the vehicle is travelling and, if the vehicle state satisfies the sending conditions, it sends the position report to CC.

Algorithm 4 MapBasedSampling($stop, turn$)

```
1:  $seg_{prev} = detectSeg()$ ;
2: while true do
3:    $seg_{now} = detectSeg()$ ;  $\triangleright$  Vehicle detects its map
   segment at instant  $now$ 
4:   if ISSEND( $stop, turn, seg_{now}, seg_{prev}$ ) then
5:     send();
6:      $seg_{prev} = seg_{now}$ ;
7:   end if
8: end while
```

4.2 Information-Need Techniques

Information-need techniques can be used for TMS data requests only. To this extent, for each window period $[t + 1, t + \tau]$ of length τ the CC

- broadcasts the last computed average speed $\bar{s}_{t,\tau}$ for all segments at the begging $t + 1$ of the window period;
- receives multiple reports during the window period. Specifically, each report (ID, p, s, t, seg) comes from a vehicle that have reached the end of a segment seg and the speed s represents the current speed on the segment: It is computed by simply dividing the length of the road segment by the time it took the vehicle to traverse the segment;
- computes the average speed $\bar{s}_{t+\tau,\tau}$ for all segments by using both the received reports and $\bar{s}_{t,\tau}$ at the end $t + \tau$ of the sample period.

The two information-need policies we considered differ as to the OBU transmission rule and the average speed computation. However, they are both based on an estimate $N(seg)$ of the flow of vehicles through each road segment seg at each sample period. $N(seg)$ essentially approximates the number of messages the CC should receive when no data reduction policy is applied. $N(seg)$ is given by the formula $flow(seg) * \tau$ where $flow(seg)$ is computed by adopting the speed-flow model chosen in [1].

Algorithm 5 DeterministicInformationNeed(V)

```

1:  $seg_{prev} = seg_{now} = detectSeg()$ ;
2:  $\bar{s}(seg)_{t,\tau} = receive(seg_{now})$ ;  $\triangleright$  it receive from the CC
   the average speed of the current segment
3: while true do
4:    $seg_{now} = detectSeg()$ ;
5:   if ISSEND( $false, false, seg_{now}, seg_{prev}$ ) then  $\triangleright$  at
   the end of segment
6:      $s = CalculateS()$ ;  $\triangleright$  it calculates the current
   speed on the segment
7:     if ( $|s - \bar{s}(seg)_{t,\tau}| \geq V$ ) then
8:        $send()$ ;
9:     end if
10:     $\bar{s}(seg)_{t,\tau} = receive(seg_{now})$ ;
11:     $seg_{prev} = seg_{now}$ ;
12:  end if
13: end while

```

4.2.1 Deterministic Information Need Policy

The main idea of the deterministic information need policy is to prevent small differences in velocity from being transmitted to the CC [8]. To this end, a velocity threshold V is used and the vehicle sends its position report (ID, p, s, t, seg) when $|s - \bar{s}(seg)_{t,\tau}| \geq V$, as shown in lines 5-9 in Alg. 5. On the other side, at the end of the window period $[t, t + \tau]$, the Data Module computes a weighted average for each segment as follows:

$$\bar{s}(seg)_{t+\tau,\tau} = \bar{s}(seg)_{t+\tau,\tau} \frac{M(seg)}{N(seg)} + \bar{s}(seg)_{t,\tau} * \frac{N(seg) - M(seg)}{N(seg)} \quad (4)$$

where $M(seg)$ is the number of received updates for the segment seg in the sample period. The $\bar{s}(seg)_{t+\tau,\tau}$ will be received by the vehicle at the beginning of a new segment seg (see line 10 in Alg. 5).

Lines 1-2 in the algorithm implement the policy initialization: the vehicle detects the map segment where it is traveling and receives from the CC the $\bar{s}(seg)_{t,\tau}$. Similarly to other techniques, the main cycle in lines 3-13 continues until the vehicle is switched off.

4.2.2 Flow Information Need Policy

The flow information need policy performs data reduction by using randomization [1]. That means that every vehicle has an equal chance of transmitting to the CC, regardless of its speed. Specifically, the approach relies on k of messages the CC expects to receive for a given segment and each sample period.

The k value can be set in order to guarantee that the absolute difference between the computed speed average and the actual value is less then some error with some confidence percentage. Let X be a random variable representing the

Algorithm 6 FlowInformationNeed($turn$)

```

1:  $seg_{prev} = seg_{now} = detectSeg()$ ;
2:  $P = receive(seg_{now})$ ;  $\triangleright$  The Transmission Probability
   in the CC of the current segment
3: while true do
4:    $seg_{now} = detectSeg()$ ;
5:   if ISSEND( $false, turn, seg_{now}, seg_{prev}$ ) then
6:      $P_r = random\ probability$ ;
7:     if ( $P_r \leq P$ ) then
8:        $send()$ ;
9:     end if
10:     $P = receive(seg_{now})$ ;
11:     $seg_{prev} = seg_{now}$ ;
12:  end if
13: end while

```

speed of a car in the segment during the sample period with mean μ and variance σ^2 . μ is what the CC wishes to compute. Then, if a $100(1 - \alpha)\%$ confidence interval for the average speed μ is desired with maximum error of the estimate of ε then

$$k = \frac{z_{\alpha/2} \sigma^2}{\varepsilon^2} \quad (5)$$

where $z_{\alpha/2} = \Phi^{-1}(1 - \frac{\alpha}{2})$. Φ^{-1} is the inverse of the cumulative function of the normal distribution.

In this context, the transmission rule indicates that the vehicle that reaches the end of a segment sends the position report only if a random generated probability is less than the *transmission probability* received from the CC. To this end, at the beginning of every window period, the CC computes the transmission probability as $P = k/N$ and broadcasts P to all vehicles.

At the end of the window period, i.e. at the time instant $t + \tau$, the CC computes the speed average as the average of the received updates $\bar{s}(seg)_{t+\tau,\tau}$ for each segment seg for which it received at least k messages, as a weighted average otherwise:

$$\bar{s}(seg)_{t+\tau,\tau} = \begin{cases} \bar{s}(seg)_{t+\tau,\tau} & \text{if } M(seg) \geq k \\ \bar{s}(seg)_{t+\tau,\tau} \frac{M(seg)}{k} + & \\ + \bar{s}(seg)_{t,\tau} * \frac{k - M(seg)}{k} & \text{if } M(seg) < k \end{cases} \quad (6)$$

After initialization (lines 1-2) where the vehicle detects the current map segment and receives the P from CC, the vehicles repeatedly checks if ISSEND() is *true*, then it sends its position report if the transmission rule is satisfied and it receives from CC the *Transmission Probability* of the new segment (lines 6-12).

5. SCENARIOS

In order to thoroughly evaluate the effectiveness of the different data reduction techniques, we have tested them by means of simulations performed on actual road network scenarios. While the next section will be devoted to the presentation and analysis of the test results, first of all we will discuss the features of the four different environments we selected:

- Bologna (BO, for short): a portion of the city center of Bologna (Italy), i.e. a typical european urban scenario involving medium-width and narrow streets;

				Vehicles			Stops		
	Duration (sec)	Size (Km)	#Segs	#Vehicles	Avg life (sec)	Avg speed (Km/h)	%Vehic	#Stops	Avg dur (sec)
Bologna	600	10x10	1523	763	351.13	20.72	34.34%	3.02	6.8
Roma	600	2.2x0.7	1238	1739	75	51.5	21.62%	1.31	18.53
Toll Plaza	600	2.3x1.4	243	1281	114.25	50.19	45.98%	2.03	16.17
Beijing	600	1.2x1	147	3410	98.71	10.89	78.53%	1.58	67.7

Table 1: Specifications of the four scenarios

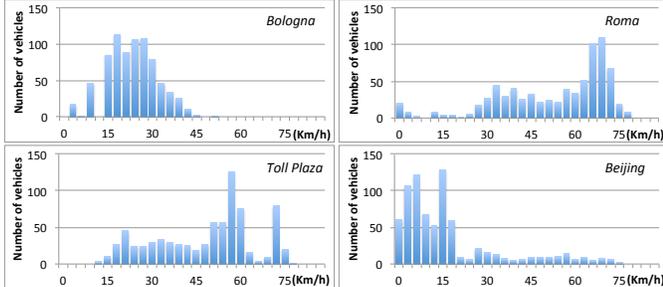


Figure 3: Detailed histograms on average speed for the four scenarios

- Roma (RO, for short): another european portion involving non-central junctions and crossroads in Rome (Italy), with fast and multi-way segments, and high traffic density;
- Toll Plaza (TP, for short): a portion of multi-way road junctions in Camden (New Jersey, USA), including a toll-payment station, with medium traffic density;
- Beijing (BJ, for short): a portion of Beijing traffic network, with very intense, congested and heterogenous (i.e. not only cars but also a significant number of motorbikes and bicycles) traffic conditions.

All the scenarios are ten minute long, have been simulated in the PTV Vision VISSIM software and recreated taking into account real data on the detailed topography, traffic volumes, vehicle speeds and flows representing typical traffic conditions in such environments.

Table 1 summarizes from a numerical point of view the peculiarities of the four scenarios. Besides duration and road network size, the table summarizes the number of segments and average travel time per segment, the number and behavior of the vehicles, including average simulation life and speed, and further details on the stops performed by the vehicles. In particular, from the average speed column, we can instantly see that the RO and TP scenarios are faster than BO and BJ. This is even more clear from the average speed histograms shown in Figure 3, where for RO and TP the peak is between 60 and 70 Km/h, as opposed BO (slower traffic, peak near 20Km/h) and BJ (even slower/congested situation, as can be also seen from the very large number of total vehicles from Table 1). Another crucial information to “understand” each scenary is the average number and duration of stops performed by the vehicles. The right part of Table 1 shows the percentage of vehicles performing at least one stop and, for such vehicles, the average number of

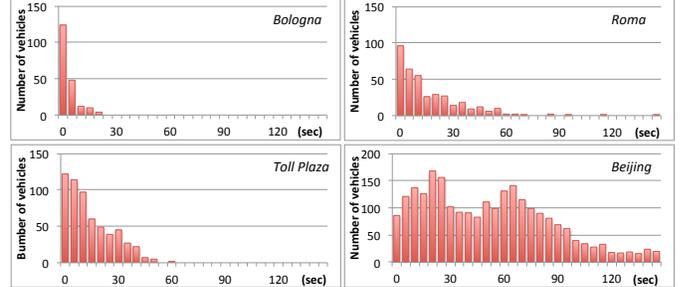


Figure 4: Detailed histograms on average cumulative stop duration for the four scenarios

stops and cumulative stop duration (which is also detailed by the histograms in Figure 4). As we can see, in BO and RO stops are quite short and infrequent. On the other hand, in TP nearly one vehicle out of two stops, and in BJ stops are frequent and long due to congested traffic.

6. EXPERIMENTAL EVALUATION

In this section, we will show the results of the experimental evaluation we performed on the different data reduction techniques acting on the four considered scenarios. In order to evaluate the actual effectiveness of each technique, we will consider the communication cost, i.e. the flow of information sent from the vehicles to the CC (expressed in messages sent per second) side by side with the average error produced by the CC in executing TMS and LBS data requests. The error is expressed in meters and as a percentage of the actual speed, respectively, and is computed as a mean of the “distances” between the values expected by the CC and the actual values (as available from the complete simulation data). Such distances were computed, for each vehicle and for each segment, respectively, at the end of each of the ten minutes of simulation, then the different figures were averaged. All techniques were implemented as described in the previous sections, and were tested with a large variety of values for their distinctive parameters (i.e. time period from 1.5 to 25 seconds for simple sampling, distance from 7 to 150 meters for space sampling, stop and/or turn variants for the affected techniques, and so on).

Let’s start by analyzing the performances achieved by the techniques for LBS data request: Table 2 summarizes the obtained results on communication cost and obtained errors for a selection of the input parameters (detailed in the first column) and in the four scenarios. Each row of the table represents a technique in a specific configuration and, for a meaningful comparison, the produced error should be compared with the results obtained by techniques showing a sim-

Bologna		Roma		Toll Plaza		Beijing		
Params	Sent/sec	Err (m)	Sent/sec	Err (m)	Sent/sec	Err (m)	Sent/sec	Err (m)
Simple sampling								
2 sec	0.5	3.21	0.5	7.84	0.5	9.69	0.5	1.99
5 sec	0.2	10.71	0.2	24.5	0.2	28.92	0.2	5.97
10 sec	0.1	26.11	0.1	54.03	0.1	61.26	0.1	11.61
15 sec	0.066	43.55	0.066	88.71	0.066	96.74	0.066	15.38
20 sec	0.05	60.78	0.05	119.14	0.05	127.52	0.05	19.15
Space sampling								
10 m	0.584	3.43	1.053	2.91	1.172	2.37	0.339	4.56
15 m	0.403	5.59	0.795	4.48	0.697	5.79	0.252	6.76
25 m	0.25	12.47	0.522	8.58	0.458	10.54	0.158	11.62
50 m	0.12	25.49	0.284	19.56	0.258	21.27	0.088	24.9
75 m	0.079	41.53	0.194	30.66	0.169	36.14	0.06	40.3
10 m, stop	0.586	3.35	1.057	2.61	1.181	1.78	0.354	2.22
15 m, stop	0.405	5.49	0.799	4.03	0.706	5	0.268	3.34
25 m, stop	0.251	10.35	0.526	7.74	0.468	9.03	0.174	5.43
50 m, stop	0.123	24.56	0.28	17.93	0.269	18.58	0.101	11.63
75 m, stop	0.082	40.19	0.193	27.52	0.18	30.64	0.075	16.64
Map-based sampling								
-	0.439	23.35	0.233	103.36	0.36	123.62	0.139	231.67
stop	0.443	22.74	0.239	87.97	0.392	109.25	0.169	24.38
turn	0.224	50.55	0.195	165.52	0.109	171.87	0.158	44.71
stop, turn	0.226	50.68	0.205	165.11	0.112	169.15	0.236	44.6

Table 2: Performances of the data reduction techniques for LBS data requests.

ilar amount of sent data. The performances offered by simple and space sampling are generally better than the ones offered by the map-based sampling technique, on all scenarios. Indeed, in slower scenarios (such as BO and BJ) map-based techniques achieve better results than in faster ones (such as RO and TP), even if, in all cases, simple and/or space sampling produce lower error. As to map-based variations, the turn option is, as one would expect, less precise than the standard one, since less data are sent to the CC (turns are not always associated with a segment change). Turning on the stop option brings some improvements to the results, especially for scenarios where vehicles arrest more frequently and for longer periods (e.g. BJ). As to the best performing techniques, in order to better visualize the differences, we plotted the achieved results (also including additional ones not shown in table) for all scenarios on a cartesian plane, where the sending rate is on the x axis and the achieved error on the y axis (Figure 2). Simple and space sampling achieve very similar results, with a (very) small preference for space sampling in TP and BJ, i.e. the scenarios where most vehicles stop. In particular, the congestions in BJ make us better appreciate the enhancement given in the space sampling performance by the stop variation.

Table 3 summarizes the obtained results for TMS data requests (traffic monitoring services) for some of the most significant parameter configurations. Together with the techniques we discussed for LBS data requests, in this case we also have information need ones. For such techniques, we tried different changes in the parameters (threshold V for the deterministic one, DIN for short, and ϵ for the flow-based one) in order to have a more complete evaluation: the specific parameter values are not explicitly reported in the table since they depend on the specific scenario. Also note that map-based communication costs are lower, parameters being equals, than for vehicle tracking, where two points had to be sent in order to inform the CC of the vehicle direction. For a better comparison of the best performing techniques, Figure 6 shows the graphic plot of the results. First of all, simple sampling achieved better results than space sampling

Bologna		Roma		Toll Plaza		Beijing		
Params	Sent/sec	Err (%)	Sent/sec	Err (%)	Sent/sec	Err (%)	Sent/sec	Err (%)
Simple sampling								
5 sec	0.2	18.64	0.2	28.21	0.2	16.28	0.2	24.76
10 sec	0.1	26.64	0.1	35.77	0.1	28.66	0.1	31.13
15 sec	0.066	32.67	0.066	41.98	0.066	35.02	0.066	37.44
20 sec	0.05	33.62	0.05	46.16	0.05	46.21	0.05	47.18
Space sampling								
50 m	0.12	29.51	0.284	30.43	0.258	31.55	0.088	127.62
100 m	0.056	43.65	0.146	25.09	0.131	39.81	0.064	199.29
50 m, stop	0.123	32.34	0.28	26.61	0.269	20.67	0.101	57.54
100 m, stop	0.06	44.11	0.148	27.97	0.142	34.32	0.064	90.78
Map-based sampling								
-	0.23	9.08	0.161	10.62	0.198	23.52	0.066	106.94
stop	0.233	8.52	0.162	6.4	0.215	10.52	0.081	51.62
turn	0.117	27.47	0.112	34.26	0.067	44.72	0.099	80.56
stop, turn	0.124	27.99	0.124	35.7	0.079	42.41	0.192	82.2
Deterministic information need								
low V	0.19	17.53	0.148	23.62	0.162	30.95	0.055	86.36
high V	0.14	20.02	0.104	25.9	0.123	34.69	0.05	119.84
Flow information need								
low eps	0.209	10.54	0.15	10.9	0.188	24.68	0.0634	110.43
med eps	0.176	11.21	0.137	11.2	0.162	26.04	0.0563	117.35
high eps	0.089	13.5	0.103	13.7	0.087	31.55	0.0378	115.11
low eps, turn	0.104	29.6	0.102	36.64	0.0592	43.45	0.0831	83.22
med eps, turn	0.096	29.9	0.089	34.08	0.057	44.93	0.0784	82.37
high eps, turn	0.072	30.33	0.058	40.92	0.051	49.76	0.061	121.47

Table 3: Performances of the data reduction techniques for TMS data requests.

for all scenarios; for this reason, and for a better readability, space sampling results are not reported in the graphs. Starting our analysis from the BO and RO scenario, we can see that the flow-based technique (without turn option) is able to produce the most satisfying performances, significantly decreasing the simple sampling error level. Indeed, the regularity/predictability of the traffic and the infrequent stops performed by the vehicles are well suited to such kind of techniques, while in TP the higher number of stops makes their effectiveness less evident. DIN is generally less effective than flow, with higher average errors (as also noticed in [1]), while map-based techniques (no turn), especially in the stop variant, offer interesting error figures, even if at higher communication costs. The turn option (available for flow-based and map-based techniques) provides lower costs but significantly higher errors and is generally not advisable. Finally, in BJ, which is a very complex scenario, with very irregular traffic flows (also due to the conspicuous presence of different vehicle types), the error levels of most advanced techniques are quite high: in this case, simple sampling appears the simplest and most effective option.

7. CONCLUDING REMARKS

In this paper we have investigated a set of data reduction techniques in different traffic scenarios with the aim of identifying those that perform better for a general-purpose service provision.

We have tested two kinds of data reduction techniques, both independent techniques, where CC doesn't affect the sending policy of vehicles, and information-need techniques, where vehicles send data taking into account additional data received from the CC. Tests have been done on disparate scenarios, ranging from highway roads to urban areas with frequent vehicle's start-and-stop's and several road intersection. Furthermore, the techniques have been tested for two different kinds of data requests, aggregate-based informa-

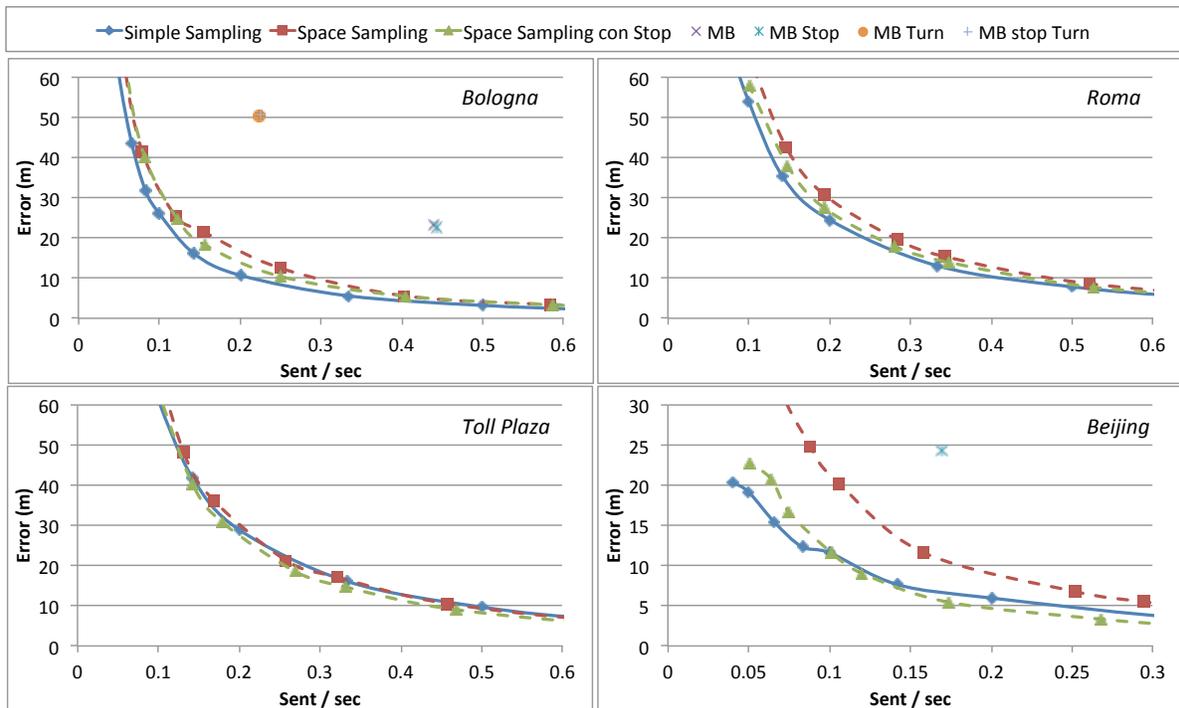


Figure 5: Performance detail graphs for LBS data requests.

tion and item-based information. The experimental results demonstrate that no technique performs definitely better than the others in all scenarios. This is because scenarios are very different under various aspects, as traffic, road network, and kind of vehicles. Under the global perspective of heterogeneity of both scenarios and data requests, the main aim of the performed experimental evaluation was to detect the technique with the best trade-off between communication cost and error. Tests have shown that simple and general techniques like Simple Sampling on average perform better than more complex techniques. As to our understanding, this is because more complex techniques try to exploit specific traffic or map peculiarities, thus losing their effectiveness in heterogeneous scenarios.

In our future work we plan to combine the different techniques dynamically, mainly depending on CC's workloads. Furthermore we will study how V2V communication combined with data reduction techniques can further reduce communication costs.

8. REFERENCES

- [1] D. Ayala, J. Lin, O. Wolfson, N. Rishe, and M. Tanizaki. Communication Reduction for Floating Car Data-based Traffic Information Systems. In *Proc. of Advanced Geographic Information Systems, Applications and Services*, pages 44–51, 2010.
- [2] B. Babcock, M. Datar, and R. Motwani. Load shedding for aggregation queries over data streams. In *Proc. of 20th Int. Conf. on Data Engineering (ICDE)*, pages 350 – 361, 2004.
- [3] A. Civilis, C. Jensen, J. Nenortaite, and S. Pakalnis. Efficient tracking of moving objects with precision guarantees. In *The 1st Annual Int. Conf. on Mobile and Ubiquitous Systems: Networking and Services (MOBIQUITOUS)*, pages 164 – 173, 2004.
- [4] R. Ding and Q. Zeng. A Clustering-based Multi-channel Vehicle-to-Vehicle (V2V) Communication System. In *Proc. of ICUFN*, pages 83–88, 2009.
- [5] Q. Doan, T. Berradia, and J. Mouzna. Vehicle speed and volume measurement using v2i communication. In *Proc. of 9th WSEAS Int. Conf. on Applied Informatics and Communications (AIC)*, pages 366–372, 2009.
- [6] S. Goel, T. Imielinski, and K. Ozbay. Ascertaining Viability of WiFi based Vehicle-to-Vehicle Network for Traffic Information Dissemination. In *Proc. of IEEE ITSC*, 2004.
- [7] O. Kennedy, C. Koch, and A. J. Demers. Dynamic Approaches to In-network Aggregation. In *Proc. of ICDE*, 2009.
- [8] B. Kerner, C. Demir, R. Herrtwich, S. Klenov, H. Rehborn, M. Aleksic, and A. Haug. Traffic state detection with floating car data in road networks. In *Intelligent Transportation Systems, 2005. Proceedings. 2005 IEEE*, pages 44 – 49, sept. 2005.
- [9] K. Liu, K. Deng, Z. Ding, M. Li, and X. Zhou. Moir/mt: monitoring large-scale road network traffic in real-time. *Proc. VLDB Endow.*, 2(2):1538–1541, 2009.
- [10] F. Mandreoli, R. Martoglia, W. Penzo, and S. Sassatelli. Data management issues for intelligent transportation systems. In *Proc. of SEBD*, pages 198–209, 2010.
- [11] A. Shinya, N. Satoshi, and T. Teruyuki. Research of compression method for probe data—a lossy compression algorithm for probe data. *IEIC Technical*

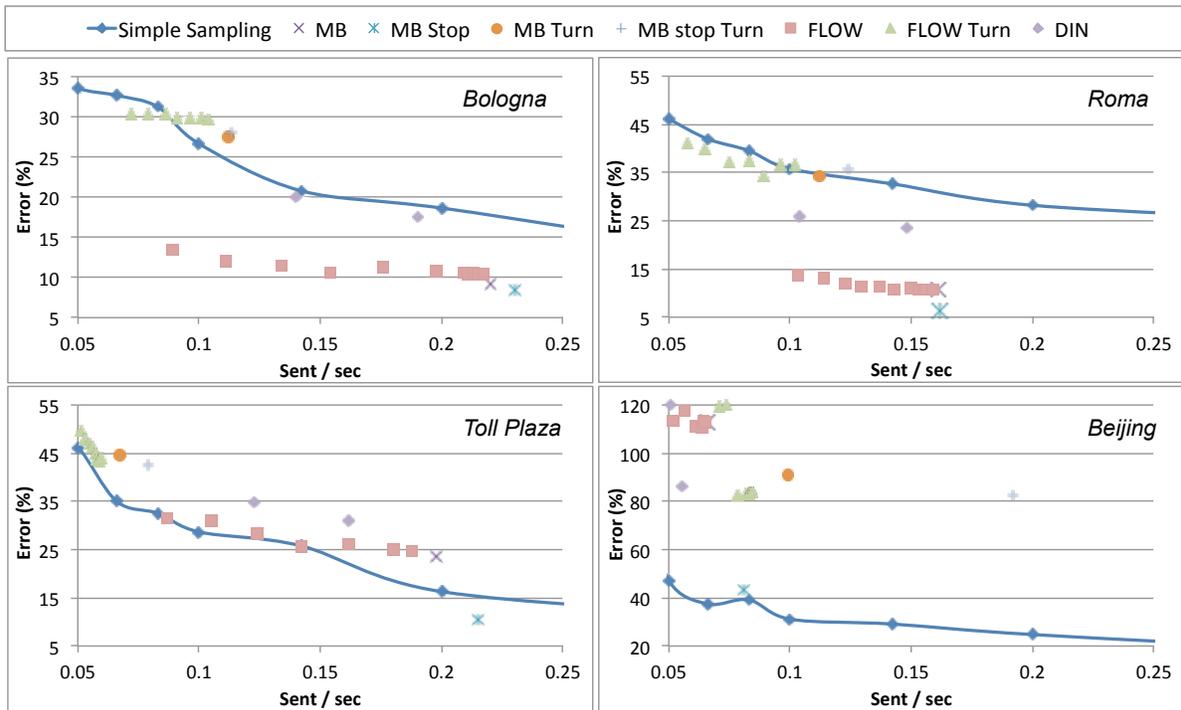


Figure 6: Performance detail graphs for TMS data requests.

Report (Institute of Electronics, Information and Communication Engineers), 104(762):13–18, 2005.

- [12] M. Tanizaki and O. Wolfson. Randomization in traffic information sharing systems. In *Proc. of the 15th annual ACM int. symp. on Advances in geographic information systems (GIS)*, pages 23:1–23:8, 2007.
- [13] S. Turksma. The various uses of floating car data. In

10th Int. Conf. on Road Transport Information and Control, pages 51–55, 2000.

- [14] Q. Xu, K. Hedrick, R. Sengupta, and J. VanderWerf. Effects of vehicle-vehicle/roadside-vehicle communication on adaptive cruise controlled highway systems. In *Proc. IEEE 56th of Vehicular Technology Conference*, volume 2, pages 1249–1253, 2002.