High-Throughput Infrastructure for Advanced ITS Services: A Case Study on Air Pollution Monitoring

José M. Cecilia, Isabel Timón, Jesús Soto, José Santa, Fernando Pereñíguez and Andrés Muñoz

***Abstract*—Novel cooperative Intelligent Transportation Systems (ITS) serve as the basis for the provision of a number of services for drivers, occupants and third parties. The vast amount of information to be collected, especially in vehicle-to-infrastructure (V2I) communication services, requires new algorithms and hardware platforms to cope with real-time requirements, however, this combination is not properly addressed in the literature. In this paper we introduce a high-throughput hardware-software infrastructure to gather information from vehicles and efficiently process it to provide novel ITS services. We propose a parallelization approach of a fuzzy clustering technique on heterogeneous servers based on CPU and several GPUs, tailored to classification problems in V2I. The infrastructure is empirically tested to offer a geo-located pollution information service through the periodical collection of both vehicle's position and status data. We offer a real service that correctly identifies highly polluting traffic areas and drivers. The results indicate a good performance of the system under high loads, and our scalability analysis reveals a good operation in real ambitious deployments, thanks to the use of both CPU and multiple GPUs, showing that our proposal can efficiently host cooperative services involving high processing in the ITS context.**

***Index Terms*—Pollution Monitoring, Fuzzy Clustering, HPC, Heterogeneous Computing, V2I, Intelligent Transport Systems**

I. Introduction

ADVANCES in information and communication technologies have enabled the integration of novel services in road transportation. Intelligent Transportation Systems (ITS) pioneer contributions towards the application of information and communication technologies in vehicles to improve safety by means of sensors, actuators and embedded computers. In the last decade, advances in communication technologies have fostered cooperative systems. Cooperative ITS allow vehicles to communicate with other vehicles (V2V) and with the infrastructure (V2I), opening the development of a broad set of novel services [1]. The era of Internet of Things (IoT) has been recently assumed by the cooperative ITS community as a way to achieve a smarter exchange of information by connecting different entities such as cars, physical devices, roads or buildings with computing servers, sensors and actuators [2].

Within the cooperative ITS scenario, vehicles can generate valuable information to enable different services to improve the quality of traffic, driver's experience and even provide tools for environment protection. A clear example of the latter would be the provision of pollution parameters to third-party services and authorities to assure the air quality in cities. Indeed, this is a hot topic nowadays as indicated in a recent study made by top scientific advisers of the European Commission [3]. However, these ITS services require a communication and processing infrastructure capable to analyze the large amount of information generated by traffic entities. Therefore, cooperative ITS infrastructures are called to be redesigned and even be rethought in order to offer Big Data analytics in real-time.

Machine learning [4] has become essential for the predictive analysis of data deluge, but high performance computing (HPC) plays an equally important role, particularly when real-time response is crucial [5]. Machine learning methods can provide good solutions in a reasonable time frame, which is a key added value to common vehicle-to-infrastructure ITS services. Moreover, most of the machine learning techniques are inherently parallel by definition and therefore they are well-suited for parallelization on current HPC architectures [6]. HPC computing architectures are based on two different approaches: low-power processors for embedded and on board devices with limited computing power, and high throughput computing servers that rely on heterogeneous computing, mainly based on the CPU and accelerators like Graphics Processing Units (GPUs). Indeed, the intersection between big data algorithms and HPC is crucial when large and complex data sets need to be computed, stored and analyzed in a timely manner under highly scalable environments.

This paper presents a novel platform to efficiently gather information from vehicles and process it through a paralleled approach. It merges machine learning and parallel computing techniques to cover the ITS field and further support V2I services with recommendation and classification capabilities. As a proof of concept, in this paper we deploy the infrastructure to cover a geo-located air quality service that identifies polluting traffic areas and driving behaviors with a high carbon footprint. In addition, the generic nature of the solution would also allow the integration of other services requiring the processing of high volumes of data, such as traffic jam avoidance, dynamic route planning, parking and, in general, other crowd sensing schemes with the aim of providing smart mobility capabilities. Hence, the major contributions of this paper are:

1) A hardware-software infrastructure able to cope with the big data processing needs of high performance computing V2I services in real-time.
2) A machine learning method for fuzzy classification on heterogeneous computing architectures has been adapted to be deployed on multiple CPUs and GPUs by using

José M. Cecilia, Isabel Timón, Jesús Soto and Andrés Muñoz are with Escuela Politécnica, Universidad Católica de Murcia (UCAM), 30107, Guadalupe, Murcia, Spain e-mail: jmcecilia@ucam.edu.

José Santa is with Dept. Information and Communications Engineering, University of Murcia.

Fernando Pereñíguez is with Dept. Sciences and Informatics, University Centre of Defence at the Spanish Air Force Academy.

OpenMP and CUDA at ITS back offices.
3) Several load balancing strategies are evaluated to pursue the performance when using high-demanding services in a cooperative ITS scenario.
4) Two pollution-related services have been implemented and evaluated for vehicle fuel-consumption monitoring and an air quality assurance, including a data gathering campaign to collect diagnosis data.
5) Finally, a performance evaluation analyses the scalability of the solution when increasing data flows, thus involving a high numbers of vehicles.

The rest of the paper is structured as follows. Section II provides a background about the main areas covered by this work. Related works are reviewed in Section III. Section IV introduces the general infrastructure to provide high-performance computational services in ITS, while Section V exemplifies its application to implement a traffic-related air pollution monitoring as a case study. Section VI describes the experimental methodology along with performance and quality evaluations. Finally, Section VII includes conclusions and future directions.

II. Background

A. Cooperative ITS

Services in cooperative ITS are usually categorized in the next groups [7]: safety, involving services intended to reduce accidents and safeguard vehicle occupants and pedestrians; traffic efficiency, dealing with the improvement of the road network capacity and reduce the travel time; and infotainment, mainly oriented to provide value-added comfort services, Internet access and multimedia.

A recent service set closely related to traffic efficiency is *green transport*, given the attention that pollution is receiving lately. For this reason, novel ITS services have started to consider the collection of pollution parameters to better inform authorities of high levels of COx and NOx gases. At the moment, air measurements are taken by weather stations installed at key points in cities, but future advances consider collecting pollution information from vehicles in real-time. Some contributions in this field bet on the integration of new sensors and special hardware to measure NO and CO gases, among other parameters, such as the works in [8], [9]. On the contrary, another research line focuses on the usage of currently available features of cars to gather pollution parameters. This is the case of using an On-Board Diagnosis (OBD) interface to obtain such data, as can be seen in the works [10], [11]. In [12], this strategy is combined with the use of digital maps to geo-locate pollution issues, addressing the same objective than our case of study later discussed.

In scenarios such as pollution monitoring, where large volumes of data are collected, there are performance issues that should be considered, involving efficient (vehicular) communications, new-age data processing algorithms adapted to deal with imprecise and incomplete data, and high performance computing platforms to cover real-time requirements.

B. ITS Communication Architecture

The development of ITS services has received a decisive impulse thanks to the definition of a common ITS communications architecture. This architecture is intended to be valid for a variety of communication scenarios (vehicle-based, roadside-based and Internet-based) through a diversity of access technologies (802.11p, infra-red, 2G/3G, satellite, etc.) and for a variety of application types. This common communication architecture is known as the *ITS Station Reference Architecture* (ITS-SRA) and is specified by ISO in [13] and by ETSI in [14]. The ITS-SRA establishes both the types of entities and an unified communications stack.

Regarding the entities participating in the implementation of vehicular services, the ITS-SRA identifies three main categories of *ITS Stations* (ITS-S): *vehicle ITS-S*, *roadside ITS-S* and *central ITS-S*. The last one identifies all the equipment located in the road operator's network infrastructure back-end and receives a particular attention in this paper. With respect to the communications stack, as depicted in Fig. 1, the ITS-SRA includes four horizontal layers surrounded by two vertical layers. On the top of the stack the *application layer* implements ITS application services like those related to avoiding collisions or achieving green transport. Middleware is included in the *facilities layer*, while networking modules are embedded in the *networking & transport layer*. Finally, the *access layer* brings together adaptation and medium access. Here it is worth mentioning 802.11p, an extension to the basic 802.11 that represents the facto solution for short-range vehicular communications. Additionally, internal processes and secure operations are supported by the *management* and *security* layers, respectively. This stack is implemented partially or totally by ITS stations, depending on its role and requirements.

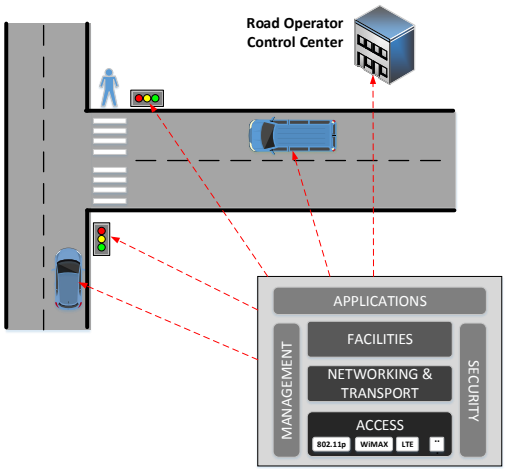


Fig. 1. ITS Station Reference Architecture (ITS-SRA)

C. Processing large data-sets for ITS services

Emergent ITS-based applications are characterized by processing large data sets collected from different sources such as vehicles, citizens, sensors or even information generated by companies or individuals on the Internet. The analysis of these data within an administrative domain like a city has become increasingly relevant to identify new insights and offer novel

solutions for upcoming problems [15]. Indeed, these large data sets must be usually processed with real-time requirements. Therefore, High Performance Computing (HPC) is becoming mandatory at the early stages of the infrastructure development to deal with such requirements.

The HPC market is witnessing the steady transition from homogeneous to heterogeneous computing systems [16], where nodes combine traditional multicore architectures (CPUs) and accelerators (mostly represented by GPU computing movement or Intel Xeon Phi cards). In heterogeneous computing systems, programmers play a fundamental role, as they might have to redesign applications to maximize performance with parallelism as a mandatory ingredient. In a few years, the GPGPU (General Purpose application on GPUs) field has grown very fast and evolve into one of the best ways of achieving high performance computing from novel heterogeneous platforms. Indeed, Compute Unified Device Architecture (CUDA) is a leading exemplar of GPGPU, offering a platform for GPUs that comprises both software and hardware.

Even with those great advances in the HPC fields, the amount of data to be processed grows exponentially each year in the big data era. Therefore, data analysis need to be performed with techniques that do not require large computational times. Actually, ITS-based applications do not require precision in many scenarios, but they may include imprecision by definition, since it depends on several factors such as communication and sensors reliability. Soft Computing methods are algorithms that include imprecision into the calculation on one or more levels [17]. The imprecision is targeted by these techniques either by decreasing the problem granularity or by "softening" the goal of optimization at some stage. Soft Computing uses natural processes as a role model and, at the same time, aims to formalization of these tasks like humans perform cognitive decisions in their daily activities. Moreover, soft computing algorithms are inherently parallel by its definition, and the combination of HPC and Soft Computing is mandatory to provide new applications for smart environments, and particularly, to successfully develop ITS services.

III. Related Work

Given the broad spectrum of potential services and technologies involved in a system like the one presented in this paper, related works must be considered from different viewpoints. Starting from the technological issues when collecting data from vehicles, we must take into account that this problem is not new. For instance, the work in [18] proposes a system to gather OBD information from vehicles by using 3G networks. This work also includes an intelligent system to identify potential mechanical problems in vehicles on the basis of the previous experience. In [19], the problem of vehicle monitoring is solved using IoT application protocols. In [20], pollution information is gathered from especial sensors installed in vehicles with Radio Frequency Identification (RFID) support. However, these works have in common that the computational resources required for the operation of these services under real settings are not considered. The system proposed in the current paper gathers status information from vehicles (including OBD data) but, as key differences with the works cited in this section, it includes a communication link based on recent advances in vehicular networks, a generic intelligent classification algorithm for ITS applications, and a parallelism strategy to overcome the computational problems of processing huge amounts of data.

Regarding proposals aligned with the aim of processing traffic data to provide ITS services, in [21] authors tackle the performance problem of route guidance and traffic forecasting. To this end, a parallel version of the A* algorithm and a propagation model are used, and experiments involving several threads are performed obtaining significant gains with respect to the sequential executions. Xia et. al [22] propose a framework to offer real-time traffic parameters. To achieve this, authors propose a model based on DempsterShafer (D-S) and rough set theories to fuse the heterogeneous data gathered, and two types of parallelization are used: algorithm-centric and data-centric. While this work focuses on the efficient fusion of data from traffic sensors, our proposal only uses vehicle sensors without the need of external infrastructure. In [23], the authors introduce a statistical framework that uses the cumulative acoustic signal from a roadside microphone to classify the vehicular traffic density state. They use different classification algorithms, such as Support Vector Machine (SVM) and Bayes, but they focus on evaluating the cost and accuracy, and not on the overall system performance.

There are several works in the literature that combine the multi-agent paradigm with high performance computing techniques to offer real-time ITS services. Thus, Murueta et al. [24] propose a multi-agent framework that takes into account multiple sources of data to recommend the best routes. Agents implement a parallel version of the K shortest path algorithm outperforming the sequential execution. In [25] a multi-agent architecture is presented to offer cooperative routing services. However, data for the evaluation of the architecture is simulated, both for traffic sensors and for OBD data. In [26] it is introduced the use of a cloud environment as an alternative for achieving high computational performance in a multi-agent and multi-layer framework. Following the big data paradigm, authors use the Infrastructure as a Service (IaaS) approach and resources are provided on-demand according to the computational requirements of agents, although no performance evaluation is shown in the paper. While all these works rely on distributing the data and the algorithms among agents, our proposal gathers the data following a client-server model and applies parallelization techniques in the server, also obtaining real-time and efficient ITS services.

Especially focused on traffic pollution, the work in de Penning et al. [27] applies deep learning and symbolic reasoning to learn drivers' behavior and help them to reduce $C0_2$ emissions. To this end, their system gathers data from the vehicle's CAN-bus such as speed, gear, steering angle, etc. Ambiguity and errors in the data are dealt with a Bayesian inference model. The work in [28] extends the platform presented in [29] about vehicular sensing, but now considering a cloud and edge computing approach. One of the services evaluated is pollution monitoring and real prototypes for vehicular on-board units are

presented and evaluated in detail. However, no computing performance nor parallelization techniques are taken into account. Ir our case we bet on a FC clustering approach, and we apply parallelization to overcome the computing of high volume of data in the back-end system. The advantages of parallelizing this kind of algorithms can be checked in [30], where the system is checked to work precisely while achieving a speedup of x4 approximately compared with the sequential equivalent. This justifies our approach towards the parallelization of the ITS data classification subsystem.

IV. A PARALLEL INFRASTRUCTURE FOR GENERAL ITS SERVICES

This section introduces the overall hardware-software infrastructure proposed in this work to deal with ITS services. First, the main hardware entities and the communication platform are described. Then, the GPU-based fuzzy clustering algorithm used is presented.

A. Overall infrastructure

There are three main hardware entities in our proposal, as depicted in Fig. 2: *Vehicle ITS-S*, *roadside ITS-S* and *Central ITS-S*. The base communication architecture presented in [31] has been used, which is compliant with the standardized ISO/ETSI reference architecture specifications explained in section II-B. This stack has been augmented with additional IPv6-based protocols to support the mobility of nodes while maintaining Internet connectivity, and it is a perfect frame for infrastructure services.

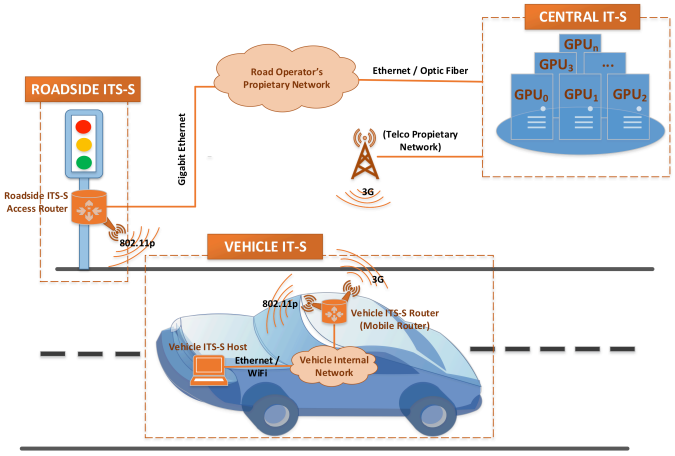


Fig. 2. Proposed architecture for high-performance ITS services

The *Vehicle ITS-S* entity integrates the on-board networked nodes for accessing the whole network. The functionality in the vehicle is split into two nodes: *vehicle ITS-S host* and *vehicle ITS-S router* (also known as Mobile Router - MR). MR hides networking tasks to in-vehicle hosts and it is responsible for providing connectivity to in-vehicle hosts through a vehicle internal network. An unlimited number of hosts could connect to the vehicle internal network through a common WiFi access (e.g. IEEE 802.11a/b/g) or an Ethernet connection. To maintain external communication with roadside equipment and the road operator control center, we consider the use of 802.11p, to communicate with Roadside ITS stations located in the surroundings, and 3G as backup communication technology for those cases where the 802.11p connectivity is not available. Additionally, GPS enables the vehicle to be geo-located. The MR offers IPv6 mobility of the whole in-vehicle network, while the Vehicle ITS-S Host executes final applications that could access remote services. The *Roadside ITS-S* entity provides local wireless connectivity, acting as network attachment point for vehicles using short/medium-range communication technologies (e.g. 802.11p). As discussed in [31], when using 802.11p, it is obtained peaks of 5 Mbps of bandwidth, less than 10 ms of two-way delay and less than 15 % of packet losses. With 3G (using HSPA), these values decrease to 1 Mbps and 200 ms respectively, but similar packet losses are detected.

The *Central ITS-S* entity brings together all the computing resources available for processing our algorithms to offer novel ITS services. This entity collects data from vehicles to offer valuable services, such as our reference pollution service. It is based on a heterogeneous cluster with multiple CPUs and Nvidia GPUs that are CUDA compatible. It is noteworthy to remark that the *Central ITS-S* is heterogeneous in two different senses; first it includes different types of processing units, i.e. CPUs and multiple Nvidia GPUs; and second, these units may have different computing capabilities or improved instruction set architectures. Thus, our algorithms need to leverage such heterogeneous systems to get optimal performance.

B. The fuzzy clustering algorithm

The software side of our infrastructure is based on a fuzzy clustering algorithm. This is a data-analysis technique aimed at organizing a collection of data-points in a multidimensional space into *clusters* (a.k.a groups). These clusters contain those data-points that are similar to each other depending on a particular metric [32]. There are different data clustering algorithms in the literature that have been applied in different scientific applications [33]. Of particular interest to us is the *Fuzzy Minimals* (FM) clustering algorithm [34] and its parallelization approach, called *Parallel Fuzzy Minimals* (PFM) [35]. FM is an unsupervised algorithm that does not require data-sets to be identified in advance. Besides, the FM algorithm allows clusters to overlap to each other, which is actually an important issue in many data-sets. Bearing this in mind, we now briefly introduce the FM algorithm that was first described by us in [35].

The FM algorithm is a fixed-point iteration algorithm that minimizes an objective function given by Equations (1) and (2). Algorithm 1 shows the sequential baselines of the FM algorithm. Two input values are included in the computation. ε_1 establishes the error degree committed in the minimum estimation; and ε_2 shows the difference between potential minimums. Next, the r factor for the targeted data-set is calculated. Then, the prototypes are figured out by minimizing the Equation (1).

$$J_{(v)} = \sum_{x \in X} \mu_{xv} \cdot d_{xv}^2, \tag{1}$$

where

$$\mu_{xv} = \frac{1}{1 + r^2 \cdot d_{xv}^2}, \tag{2}$$

Algorithm 1 The sequential baselines of Parallel Fuzzy Minimals Algorithm:

```
1: LoadDataSet()
2: Choose ε1 and ε2 standard parameters.
3: r=FactorRCalculation(dataset)
4: Prototypes=CalculatePrototypes(dataset, r)
```

The r factor measures the disruption of the homogeneity and isotropy of the sample by a set of factors affecting the Euclidean distance in each group [36]. Actually, clusters are created when the isotropy and/or homogeneity are broken. Equation (3) shows the r factor calculation in a non-linear expression

$$\frac{\sqrt{|C^{-1}|}}{nr^F} \sum_{x \in X} \frac{1}{1 + r^2 d_{xm}^2} = 1, \tag{3}$$

where $|C^{-1}|$ is the determinant of the inverse of the covariance matrix, m is the mean of the sample X, d_{xm} is the Euclidean distance between x and m, and n is the number of elements of the sample.

Algorithm 2 *CalculatePrototypes* function of Fuzzy Minimals algorithm, where *n* is the size of the data-set. *V* is the algorithm output that contains the prototypes found by the clustering process. *F* is the dimension of the vector space.

```
1: Initialize V = { } ⊂ R^F.
2: for k = 1; k < n; k = k + 1 do
3:    v_(0) = x_k, t = 0, E_(0) = 1
4:    while E_(t) ≥ ε1 do
5:       t = t + 1
6:       μ_xv = 1/(1+r^2·d_xv^2), using v_(t−1)
7:       v_(t) = (Σ_{x∈X} (μ_xv^(t))^2 · x) / (μ_xv^(t))^2
8:       E_(t) = Σ_{α=1}^{F} (v_(t)^α − v_(t−1)^α)
9:    end while
10:   if Σ_α^F (v^α − w^α) > ε2, ∀w ∈ V then
11:      V ≡ V + {v}.
12:   end if
13: end for
```

Algorithm 2 summarizes the prototype calculation procedure. FM looks for the number of prototypes using a data-structure which is actually modeled by r factor in FM's objective function (see Equations (1) and (2)). Equation (2) is the membership function that establish the degree of membership for a given element x to a particular cluster where v is the prototype. Indeed, these prototypes are the FM's output representing the minimum values of the objective function by the classification process.

C. Parallelization approach

The parallelization of this fuzzy clustering algorithm relies on the fact that FM does not compare clusters among them to minimize the objective function like other fuzzy clustering techniques. Therefore, our algorithm can take advantage of this feature to divide the original data-set into different subsets, in which FM is performed independently. This subset does not lose information about global properties for the classification, since each subset is classified using an objective function that includes r factor to provide information about the overall data-set. Indeed, the r factor is a global parameter that provides information about the whole data-set, therefore this factor is used in all subsets even though they are executed in parallel, keeping the main clustering characteristics of FM unaltered.

The r factor calculation does not require too much execution time compared to the prototype calculation as it will be justified in Section VI. Then, depending on the data-set size and the number of computational devices available on the Central ITS-S, our algorithm divides the data-set into a different number of subsets to enable prototype calculation on the GPUs. Each subset is therefore offloaded to a GPU, although not all GPUs may be used at once. If the data-set is not big enough, the prototype calculation could be developed on the CPU to avoid additional overheads.

With that scenario in mind the computation proceeds following a MapReduced strategy [37]. Firstly, the r factor is calculated using the whole dataset. Then, the algorithm decides the number of computational devices that are eventually involved in the computation. In case of multiple GPUs are required, several CPU threads are created using OpenMP technology to manage each GPU context. In a *homogeneous distribution*, the input data-set is equally distributed among the GPUs that are involved in the computation (map stage). Then, the FM algorithm proceeds on each subset in parallel to obtain a set of prototypes. Those prototypes are merged in the CPU to obtain the final set of prototypes (reduce stage).

The Central ITS-S may have different kinds of devices or even devices within the same family with different computing capabilities as previously explained. Therefore, the execution time of each independent task may differ, as it depends on the underlying GPU each subset runs on, which is actually unknown at compile-time. Given that the slowest GPU will determine the overall execution time, our mission is to make use of the idle time offered by the most powerful GPUs. Then, we propose a *heterogeneous distribution* where each OpenMP thread calculates the subset size to be classified, whose size is given on the basis of the Dif parameter (see Equation (4)). Here, several runs of our algorithm are performed to empirically determine the performance differences between all computational devices in the Central ITS-S. This is only performed once, and this information is used for the following executions. Section VI provides more details about this experimentation with real evaluations. The main output of this stage are the following parameters: the threshold data-set size for offloading to a GPU, the threshold data-set size to scale to several GPUs, and the computational differences between GPUs. Importantly, at this stage, the algorithm is not

trying to solve the problem in any meaningful sense, but these runs allow us to calculate the performance differences between GPUs. According to Equation (4), the slowest GPU will have $Dif = 1$; a GPU two times faster than the slowest GPU would have $Dif = 0.5$, and so on. Finally, the optimum number of threads is also experimentally obtained at this stage to increase the level of parallelism and to maximize processor occupancy.

$$Dif = \frac{Ex.time_{actualGPU}}{Ex.time_{slowestGPU}} \quad (4)$$

V. A CASE STUDY: TRAFFIC-RELATED AIR POLLUTION MONITORING

The generic ITS infrastructure-based service described in Section IV is evaluated in real services related to the traffic pollution monitoring. Two services have been developed for this infrastructure and a testing campaign has been carried out to assess the operation of the whole system and, especially, the intelligent classification algorithm executed in the parallel computing architecture.

A. Traffic-related air pollution services

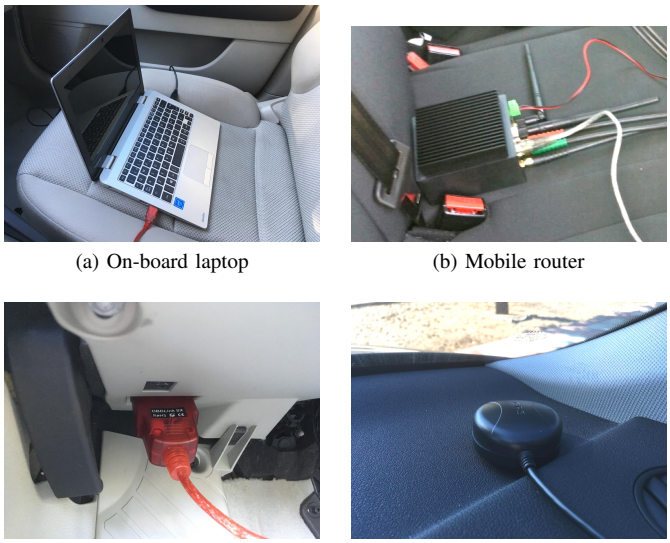
Our study evaluates two different traffic-related air pollution services:

(S1) *Fuel-consumption monitoring*. This service monitors the fuel-consumption for a single car. Once the vehicle arrives to its destination, our service provides information about the fuel-consumption level based on the vehicle's speed, RPM, engine temperature and fuel rate variables. Thus, it is an offline service where the deferred calculation can be accessed by the user.

(S2) *Air-pollution monitoring*. This is a real-time service in the area of green transport, with the aim of ensuring air quality in cities. Based on real-time crowdsourcing information from the fuel-consumption of cars within the same area, our service detects highly polluted areas in the city at the moment.

As a proof of concept, we have implemented both services to identify pollution areas in Murcia (Spain). Traffic-related air pollution is monitored within different locations using data collected from four vehicles. These cars send information, including GPS location (latitude and longitude) and fuel-consumption related variables previously described for service S1. Cars are classified by location first to identify targeted areas, and then by fuel-consumption level (as in service S1) to determine the pollution level of each of those areas.

Calculating the carbon footprint or, in general, the air pollution of vehicles implies the estimation of both: (1) the fuel flow, which is carried out using OBD-II data in our case; and (2) the emission factor per unit of fuel consumed. As used in [38], we consider that the emission factor is focused on the CO_2 gas, implying 2.64 Kg of CO_2 per one liter of diesel fuel. CO_2 comprises the most important part of the greenhouse gases emitted (around 77%) according to [38]. For the case of the regular gasoline, this ratio is 2.33 Kg of CO2 per 1 liter of fuel, although in our particular case we are using diesel cars.



(a) On-board laptop (b) Mobile router

(c) OBD-II interface (d) GPS receiver

Fig. 3. Equipment used in the testbed

B. Deployed Infrastructure

The main entities of the infrastructure described in Section IV are deployed in a testbed where the data collected from vehicles are used to detect pollution areas in Murcia (Spain). The deployment comprises the next set of nodes:

- A Toshiba Satellite CL10-C-102 laptop, depicted in Fig. 3a, with processor Intel Celeron N3050, 2 GB RAM, and Linux Ubuntu as operating system. This computer operates as Vehicle ITS Station Host.
- A Laguna LGN-20 from Commsignia, depicted in Fig. 3b, operating as Vehicle ITS Station Router. This unit provides mobile IPv6 connectivity to the laptop through a WiFi network, and it connects the vehicle network with the Central ITS Station through 3G during the tests.
- A computational back-end acting as Central ITS Station that is described in detail in the next part.

The laptop is connected with the car CAN bus through an OBD-II interface. The OBD scan tool used is OBDLink SX with USB interface, and it is connected with a proper connector available in the car, as showed in Fig. 3c. In the car used (Audi A3 1600 CC TDI) this connector is under the steering wheel. An external USB GPS has been used to obtain the position of the vehicle, model Adopt SkyTraq Venus 8. It is a mouse model, and it is placed near the windscreen to improve coverage.

OBD data collected from the vehicle comprise a set of all supported parameter identifiers (PID) by the car. Among the most important parameters we can find the Mass Air Flow (MAF) and the Calculated Engine Load ($LOAD_{CALC}$), since they allow us to compute the Fuel Flow (FF), measured in liters per hour. As indicated in [39], the model used to compute

the FF is as indicated in equation 5.

$$FF = \frac{CMAF \times 3600}{CAFR \times FD}$$
$$CMAF = MAF \times \frac{LOAD_{CALC}}{100} \quad (5)$$

$CAFR$ is the Corrected Air to Fuel Ratio, which is considered as an optimum (ideal) air to fuel ratio of 14.5g (14.7 g of air to 1 g of fuel), given that the OBD correction parameter to apply to this factor is not available in all cars. FD is the fuel density, which is 832 g/l.

All data are collected from the OBD-II interface and the GPS receiver by using the OBD GPS Logger software, and then they are sent over the IPv6 network by using one UDP datagram per data record gathered at a 1 Hz frequency. GPS information includes time, latitude, longitude and altitude, while the OBD parameters considered comprise 26 numerical values. This record is sent in comma-separated value (CSV) format.

C. Central ITS-S description

The computational back-end (Central ITS-S) we have used to run the GPU-based fuzzy algorithm algorithm has four Intel Xeon X7550 processors running at 2 GHz and plugged into a quad-channel motherboard endowed with 128 Gigabytes of DDR3 memory. Moreover, two NVIDIA GPUs are connected through PCI-Express bus. A GPU NVIDIA Tesla Kepler K40c with 2880 CUDA cores (15 Streaming Multiprocessors and 192 Streaming Processors per Multiprocessor) running at boost clock of 0.88 GHz, giving a raw processing power of up to 5068 GFLOPS, and having up to 12 GB of GDDR5. A GPU NVDIA Fermi GeForce GTX 580 with 512 CUDA cores (16 Streaming Multiprocessors and 32 Streaming Processors per Multiprocessor) running at boost clock of 1.54 GHz, giving a raw processing power of up to 1581 GFLOPS, and having up to 1.5 GB of GDDR5.

In both platforms, gcc 4.8.2 with the -O3 flag was used for compilation on the CPU, and the CUDA toolkit version 8.0 was used for compilation on the GPU.

VI. Evaluation

This section shows the results obtained testing our parallel infrastructure by targeting the traffic-related air-pollution monitoring described above. The main objective of these experiments is two-fold. Firstly, we analyze the quality evaluation for the two different services described in Section V-A: fuel-consumption monitoring (S1) for one car, and air-pollution monitoring (S2) for several areas. Secondly, we discuss the scalability for the communication infrastructure. Finally, we evaluate the performance of the platform when processing the collected data and consider ambitious scenarios to analyse the expected scalability of the platform.

A. Classification results

1) Service S1: Fuel-consumption monitoring: To verify the proper operation of this service, we analyze the results obtained with respect to fuel-consumption level from one vehicle. Firstly, the FM algorithm is applied to 843 samples containing data on car's speed, RPM, engine temperature and fuel rate. As a result, it is obtained 68 prototypes, i.e. 68 different groups. These groups are refined through a dendogram (see Fig. 4), resulting in eight final groups determined by the threshold represented as a red dash line (separation value of 120). This threshold has been found taking into account the number, size and distance among groups. Next, it is calculated the mean value for the fuel rate variable in each of the eight groups and they are sorted in ascending order, giving as a result the following mean values for each group:{0.4297, 0.4758, 0.8315, 0.9145, 1.3228, 2.0262, 3.0798, 4.7392}. We use then a color scale from light blue to red to identify each group, labeling the less fuel-consuming group as light blue and the most fuel-consuming one as red.

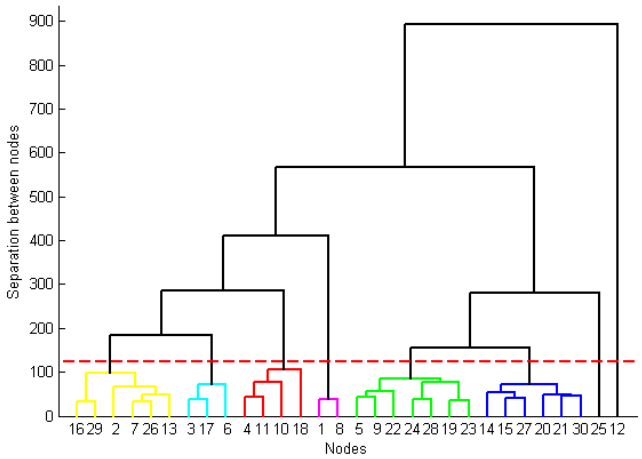


Fig. 4. Dendogram representing the final eight groups (nodes) out from the 68 initial groups obtained from the FM algorithm.

Once the groups for fuel-consuming patterns are identified, we can now add this information to any route of our vehicle. Fig. 5 shows a real example of a journey where each localization in the route is colored according to the classified fuel-consuming group. Note that less fuel is consumed at the starting and ending point (leftmost and rightmost parts of Fig. 5, respectively), whereas more fuel consumption is made in the central segment. In this segment, we identify two reductions in fuel consumption near longitude values -0.9 and -0.85, due to stops caused by traffic lights and crossroads.

A complementary view of this information is shown in Fig. 6. Here, each data sample in the X-axis is classified into one of the eight fuel-consumption groups along with the specific fuel rate in the Y-axis. The first 300 samples matches the starting segment in Fig. 5. Samples from 300 to 700 match the central segment, where more fuel consumption is made, and samples from 700 match the ending segment. It is worth mentioning the low fuel rate values (and fuel consumption levels) around samples 500 and 650, corresponding with the two fuel reductions in longitude values -0.9 and -0.85 observed in Fig. 5.

2) Service S2: Air-pollution monitoring: The correct operation of this service has been evaluated through the analysis of 4100 samples obtained from four vehicles in order to

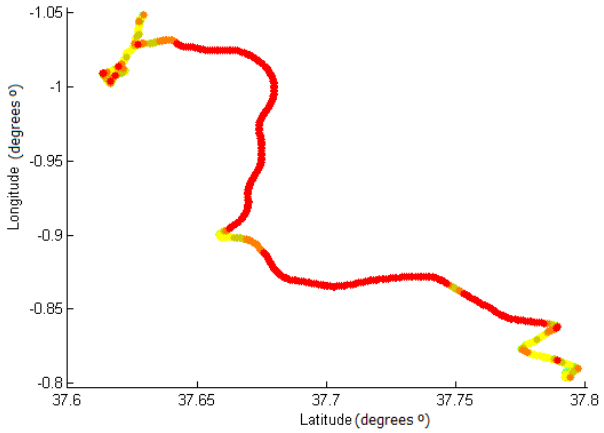


Fig. 5. Fuel consumption information during a real car journey, starting at longitude -1.01 and latitude 37.62, and ending at longitude -0.8 and latitude 37.79.

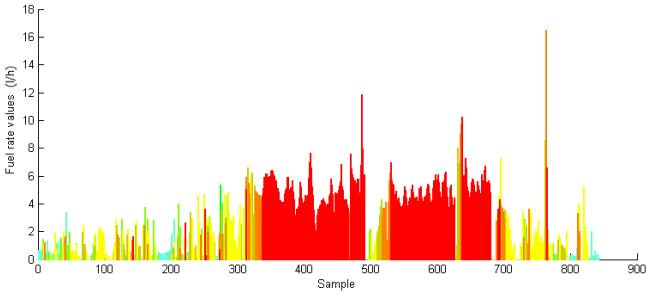


Fig. 6. Fuel rate values for each sample, classified in colors according to its fuel consumption group identified by the FM algorithm.

monitor traffic-related air-pollution in some specific areas. These samples contain the same data obtained from the vehicle as in service S1, namely vehicle's speed, RPM, engine temperature and fuel rate, along with GPS coordinates (latitude and longitude).

Firstly, the FM algorithm is applied to the streaming of 4100 samples to automatically determine the areas to be monitored according to the GPS data. Once an area is determined, a second execution of the FM algorithm is applied to the fuel-consumption related variables as in the first scenario, but in this case to the data of all vehicles within each different area. In this manner, different number of fuel-consumption levels are detected in each area and then they are converted to traffic-related air-pollution levels as explained in Section V, using the same color scale as in the first scenario. In particular, eight air-pollution levels were detected in Area A, six levels in Area B and seven levels in Area C.

Regarding Area A (see Fig. 7a), there are three lanes with similar air-polluting levels. There is an increase of this level near the roundabout (in the central segment of the area, between longitude values -1.1155 and -1.116), where a larger vehicle concentration is expected. On the other hand, in Area B (see Fig. 7b) the most interesting information is located at the bottom left corner, representing the area entrance and exit lanes. The entrance lane shows the higher air-pollution level in the area, due to slow incorporation and slow traffic flow. In contrast, the exit lane does not present this problem and no high fuel consumption is detected. Finally, in Area C (see Fig. 7c) we find one-direction lane with high pollution levels in the central segment (between latitude values 38.0035 and 38.0055), due to a crossroad causing a slow traffic flow and therefore a larger concentration of vehicles.

B. Scalability issues in the communication infrastructure

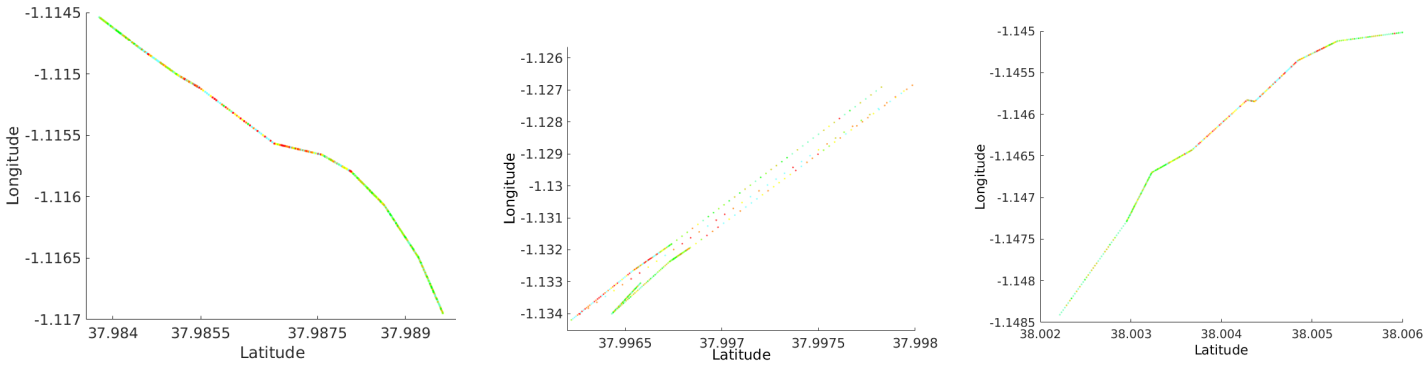
Our system is able to combine several wireless technologies in order to maximize the performance in terms of quality of service and cost. 3G/4G networks have evolved in the last years and they are not considered as a potential bottleneck in V2I applications like the one supported by the presented platform. Currently, operators are able to serve hundreds of users in a limited space, by increasing the density of base stations. A more challenging scenario is presented by the incipient 802.11p technology, when used exclusively by many cars within the coverage of a sole base station. According to the experimental work in [40], ten nodes transmitting 200-byte data packets at a frequency of 50 Hz could collapse the medium and obtain packet delivery ratios of less than 20%. Bearing in mind our pollution monitoring scenario, with a data rate of 1 Hz, we could estimate that about 500 nodes could be supported under the same base station. This value should be adjusted by the extra size of our packets of 293 bytes (including our 26 numerical fields in CSV format), which could cause a slight reduction in performance, and the mobility conditions, which would imply more losses, as experienced in our previous works [31].

This rationale envisages a good potential value of vehicle density, which by far exceeds the real density that could be achieved in a real scenario considering experimental coverage ranges of 500 meters, assuming a mean vehicle length of 5 meters and the worst case where the 802.11p station is covering a road intersection. Hence, it could be stated that even when exclusively using the 802.11p vehicular communication technology, the network infrastructure is not considered an issue for the operation of the system presented here, supposing a proper deployment of base stations. Although the 802.11p medium were used by other applications at the same time, the low requirements of the pollution monitoring services presented would allow its correct operation.

C. Performance evaluation under high loads

This section introduces the performance evaluation and scalability of our GPU-based fuzzy algorithm running on the heterogeneous CPU-GPU Central ITS-S previously described in Section V-C. To deeply analyze the behavior of our implementation, we use synthetic data-sets that contain the same type of information previously described in Section V, but adding random numbers to either increase the number of rows or number of columns for the input data-set. In this manner, the performance behavior of our infrastructure in hypothetical scenarios can be analyzed.

Algorithm 1 shows that the main functions of the FM algorithm are $FactorRCalculation()$ and $CalculatePrototypes()$. There are up to three different



(a) Area A pollution level (b) Area B pollution level (c) Area C pollution level

Fig. 7. Pollution levels detailed for the three areas.

parameters that affect performance of these functions, all of them related to the data to be classified.

- The *number of rows* that determines the number of inputs to be classified. For instance, in the fuel-consumption scenario, one car is submitting (or storing in a buffer to submit afterwards) information depending on recording of the OBD-II interface and the GPS receiver frequency (1 Hz in our case). In the air-pollution scenario, each row represents a car location, that is sent along with OBD-II information, and therefore there would be as many rows as cars submitting information to our infrastructure.
- The *number of columns* that determines the number of variables to be considered for classification. As previously explained in Section V, all data is collected from the OBD-II interface and the GPS receiver by using the OBD GPS Logger software. GPS information includes up to four different variables, namely time, latitude, longitude and altitude, while the OBD variables reach up to 26 numerical values including the Mass Air Flow (MAF) and the Calculated Engine Load ($LOAD_{CALC}$).
- Finally, the last performance parameter is the *number of prototypes* found by our GPU-based fuzzy algorithm. Our fuzzy algorithm performs a blind clustering search, not depending on the input information; indeed, it depends on the shape of data and on the input error values (ε_1 and ε_2, see Algorithm 1 and 2).

With that in mind, we now analyze these main performance factors to let the reader know about the infrastructure's scalability.

Fig. 8 shows the execution times of the $FactorRCalculation()$ and $CalculatePrototypes()$ functions for input data-sets ranging from 900 to 28000 data records from cars (i.e., rows). The $FactorRCalculation()$ is executed on the CPU and the $CalculatePrototypes()$ is executed on the GPU as it is data-parallel by its definition. In this case, the number of rows for the input data-set is increased to analyze the scalability. We leave the number of columns (i.e. the variables to be classified) set to two in this experiment. Performance results reveals that $CalculatePrototypes()$ is the main computational bottleneck, as its execution time increases exponentially along with the problem size. It is worth mentioning that $CalculatePrototypes()$ execution time does not depend only on the number of rows or columns like $FactorRCalculation()$, but also on the number of prototypes found in each data-set, which is unknown at runtime. The number of prototypes in the simulation shown in Fig. 8 for the different data-sets are 22, 45, 48, 54, 55, 80, 77 and 101.

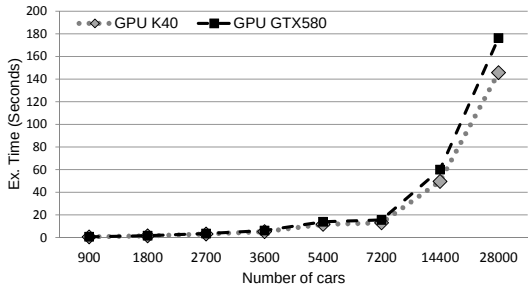


Fig. 8. Execution time in seconds of Fuzzy Minimals algorithm main functions ($FactorRCalculation()$ and $CalculatePrototypes()$) for different input data-sets, varying the number of Cars. We consider the processing time of our algorithm of up to 28000 cars sending information

Fig. 9 shows the execution times of both functions for different input data-sets, but now varying the number of variables considered in the classification algorithm (i.e., columns). We leave the number of rows set to 5400 in this experiment. In this case, performance results raise different conclusions than in the previous experiment. Increasing the number of variables to be classified affects more the $FactorRCalculation()$ in terms of performance. Let us remind that the r factor calculation includes the determinant of the inverse of the covariance matrix calculation, which is actually of size $number\_of\_columns \times number\_of\_columns$. Up to six variables the $FactorRCalculation()$ is still less time consuming than $CalculatePrototypes()$. However, from that point the $FactorRCalculation()$ execution time grows drastically. Although the classification of more than six variables is not trivial at all, here we would recommend a preliminary analysis like Principal Component Analysis (PCA) that uses an orthogonal transformation to transform a set of possibly correlated

variables into a set of values of uncorrelated variables called principal components.

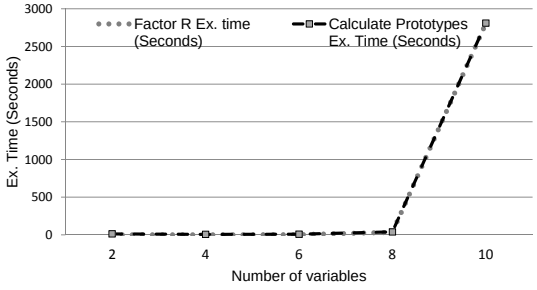


Fig. 9. Execution time in seconds of Fuzzy Minimals algorithm main functions ($FactorRCalculation()$ and $CalculatePrototypes()$) for different input data-sets, varying the number of variables to be classified

The previous performance analysis reveals that our solution could further improve results by using extra nodes computing together, bearing in mind that our algorithm allows extra division of the data-set into more subsets to perform $CalculatePrototypes()$ function in parallel. However, this distribution introduces several overheads that should be considered. With such problem division, if we are using several computational devices, at the installation stage it should be performed a load balancing strategy to get peak performance. Fig. 8 reports up to 20% speed-up factor between Tesla k40 and Geforce GTX580 for the $CalculatePrototypes()$ function. The installation stage is only performed once and, for the set of benchmarks we are running, it takes around five minutes. This will provide knowledge about the platform in which we are running our experiments and, hence, obtain an indication for load balancing in the platform.

Finally, going into details with the most challenging scenario that we have evaluated, our infrastructure executes the PFM algorithm with 8 variables and 28,000 data rows in 145.57 seconds using only a Nvidia GPU (see Figures 8 and 9). Next, we determine the potential bottleneck of the back-end, where data from hundred or even millions of vehicles could be received. To conduct this evaluation, we will assume a challenging scenario where our air pollution monitoring services are deployed in Madrid, the most crowded city in Spain. According to the reports published by the Spanish Directorate-General of Traffic [41], in 2016 about 4.5 million of vehicles are registered in Madrid. In the worst case where all vehicles are on the road at the same time in the city and using our traffic-related pollution services, the back-end infrastructure will be receiving a traffic of 10.54 Gbps. This value takes into consideration the analysis performed in section VI-B where we have confirmed the capacity of the communication infrastructure to afford our traffic-related pollution services. Recorded variables from cars are 32-bit floats, thus at a 10.54 Gbps data rate, the back-end would receive 329,375,000 variables per second. Since each data row received from a car is composed of 8 variables, this gives us 41,171,875 8-variables data-sets per second. Assuming a system response threshold of 145.57 seconds for our previous worst scenario of 28,000 cars with a single GPU, our system would need 1471 GPUs to provide an 8-variables data-set classification in that time. For instance, the current Top-3 fastest supercomputer, Piz Daint include more than 5320 Nvidia Tesla P100 GPUs (see [42]). This means we are at the technology level to really deploy our infrastructure in a real scenario like Madrid where we would be able to provide air monitoring with a delay of less than three minutes in the worst scenario. This would enable an air pollution monitoring system for a city like Madrid with a refresh rate under 5 minutes.

VII. Conclusions and future work

Cooperative Intelligent Transportation Systems have been placed as an emergent field where many novel services will be provided to final users. These services will be based on the analysis of large data-sets generated by a great number of cars. In this paper we propose a high-throughput support infrastructure for advanced ITS services, combining both hardware and software techniques, to provide novel mechanisms that enable exascale data-analysis within this context. All infrastructure layers aimed at enabling a real ITS service based on high performance fuzzy clustering are discussed, including data collection from vehicles, ITS communication architecture to connect vehicles with the infrastructure, and heterogeneous central ITS-S based on CPU and multiple GPUs. An air-pollution monitoring service is tailored to this infrastructure showing that effective information can be provided to drivers and authorities in an effort to reduce the environmental impact.

Our results reveal that realistic services such as pollution monitoring can be developed and operated efficiently within this framework, offering a good performance of the system under high loads, thanks to the use of both CPU and multiple GPUs with appropriated configuration through load balancing techniques. Moreover, our scalability analysis estimates a realistic operation of the framework in ambitious deployments where data coming from millions of vehicles is collected and processed.

ITS services are at a relatively mature stage, however, hardware features included in upcoming heterogeneous processor generations may require different run-time strategies and algorithm redefinition to obtain peak performances in V2I applications. It is expected to get even higher time-response ratios on GPUs whenever new hardware features come along to boost performance into unprecedented gains where parallelism is called to play a decisive role. Indeed, other off-the-shelf big data frameworks like Hadoop or Spark will also offer a good scalability that are well-suited to be combined with our approach. Moreover, we may anticipate that the benefits of our approach would be also combined with other soft computing techniques like deep learning to make it extensive to other ITS services that require data-intensive analysis.

Acknowledgment

This work has been sponsored by the Spanish Ministry of Economy and Competitiveness through the project HETEROLISTIC (contract TIN2016-78799-P (AEI/FEDER, UE)).

REFERENCES

[1] M. Annoni and B. Williams, *The History of Vehicular Networks*. Cham: Springer International Publishing, 2015, pp. 3–21.

[2] O. Kaiwartya, A. H. Abdullah, Y. Cao, A. Altameem, M. Prasad, C. T. Lin, and X. Liu, "Internet of vehicles: Motivation, layered architecture, network model, challenges, and future aspects," *IEEE Access*, vol. 4, pp. 5356–5373, 2016.

[3] J. Bujnicki, P. Dykstra, E. Fortunato, R. Heuer, J. Slingo, C. Villani, and H. Wegener, "Closing the gap between light-duty vehicle real-world CO2 emissions and laboratory testing," European Comission, Directorate-General for Research and Innovation, Brussels, Belgium, Tech. Rep. 1/2016, November 2016.

[4] I. H. Witten, E. Frank, M. A. Hall, and C. J. Pal, *Data Mining: Practical machine learning tools and techniques*. Morgan Kaufmann, 2016.

[5] K. Kambatla, G. Kollias, V. Kumar, and A. Grama, "Trends in big data analytics," *Journal of Parallel and Distributed Computing*, vol. 74, no. 7, pp. 2561–2573, 2014.

[6] J. M. Cecilia, J. M. García, A. Nisbet, M. Amos, and M. Ujaldón, "Enhancing data parallelism for ant colony optimization on gpus," *Journal of Parallel and Distributed Computing*, vol. 73, no. 1, pp. 42–51, 2013.

[7] European Telecommunications Standards Institute, "Intelligent Transport Systems (ITS); Vehicular Communications; Basic Set of Applications; Definitions," ETSI TR 102 638, European Telecommunications Standards Institute, June 2009.

[8] J. May, D. Bosteels, and C. Favre, "An assessment of emissions from light-duty vehicles using pems and chassis dynamometer testing," in *SAE World Congress 2014*, 2014.

[9] "Use of portable emissions measurement system (PEMS) for the development and validation of passenger car emission factors," *Atmospheric Environment*, vol. 64, pp. 329 – 338, 2013.

[10] "Real-world fuel consumption and CO2 (carbon dioxide) emissions by driving conditions for light-duty passenger vehicles in china," *Energy*, vol. 69, pp. 247 – 257, 2014.

[11] A. Alessandrini, F. Filippi, and F. Ortenzi, "Consumption calculation of vehicles using obd data," in *20th International Emission Inventory Conference*, 2012.

[12] R. K. Ganti, N. Pham, H. Ahmadi, S. Nangia, and T. F. Abdelzaher, "Greengps: A participatory sensing fuel-efficient maps application," in *Proceedings of the 8th International Conference on Mobile Systems, Applications, and Services*, ser. MobiSys '10. New York, NY, USA: ACM, 2010, pp. 151–164.

[13] ISO TC 204, "Intelligent transport systems - Communications Access for Land Mobiles (CALM) - Architecture," ISO 21217, International Organization for Standardization, april 2013.

[14] ETSI TC ITS, "Intelligent Transport Systems (ITS); Communications Architecture," ETSI EN 302 665, European Telecommunications Standards Institute, September 2010.

[15] Y. Lv, Y. Duan, W. Kang, Z. Li, and F.-Y. Wang, "Traffic flow prediction with big data: a deep learning approach," *IEEE Transactions on Intelligent Transportation Systems*, vol. 16, no. 2, pp. 865–873, 2015.

[16] K. Roy, B. Jung, D. Peroulis, and A. Raghunathan, "Integrated systems in the more-than-moore era: designing low-cost energy-efficient systems using heterogeneous components," *IEEE Design & Test*, vol. 33, no. 3, pp. 56–65, 2016.

[17] L. A. Zadeh, "Fuzzy logic, neural networks, and soft computing," *Communications of the ACM*, vol. 37, no. 3, pp. 77–85, 1994.

[18] S. H. Chen, J. F. Wang, Y. Wei, J. Shang, and S. Y. Kao, "The implementation of real-time on-line vehicle diagnostics and early fault estimation system," in *2011 Fifth International Conference on Genetic and Evolutionary Computing*, Aug 2011, pp. 13–16.

[19] R. Dhall and V. K. Solanki, "An iot based predictive connected car maintenance approach," *International Journal of Interactive Multimedia and Artificial Intelligence*, vol. 4, no. 3, pp. 16–22, 03/2017 2017.

[20] S. Manna, S. S. Bhunia, and N. Mukherjee, "Vehicular pollution monitoring using iot," in *International Conference on Recent Advances and Innovations in Engineering (ICRAIE-2014)*, May 2014, pp. 1–5.

[21] A. Attanasi, E. Silvestri, P. Meschini, and G. Gentile, "Real World Applications Using Parallel Computing Techniques in Dynamic Traffic Assignment and Shortest Path Search," in *IEEE 18th International Conference on Intelligent Transportation Systems*. IEEE Computer Society, 2015, pp. 316–321.

[22] Y. Xia, X. Li, and Z. Shan, "Parallelized Fusion on Multisensor Transportation Data: A Case Study in CyberITS," *International Journal of Intelligent Systems*, vol. 28, no. 6, pp. 540–564, 2013.

[23] V. Tyagi, S. Kalyanaraman, and R. Krishnapuram, "Vehicular traffic density state estimation based on cumulative road acoustics," *IEEE Transactions on Intelligent Transportation Systems*, vol. 13, no. 3, pp. 1156–1166, 2012.

[24] P. O. P. Murueta, C. R. C. Prez, and J. A. R. Uresti, "A Parallelized Algorithm for a Real-Time Traffic Recommendations Architecture Based in Multi-agents," in *13th Mexican International Conference on Artificial Intelligence*. IEEE Computer Society, 2014, pp. 211–214.

[25] M. Fiosins, B. Friedrich, J. Görmer, D. Mattfeld, J. P. Müller, and H. Tchouankem, "A Multiagent Approach to Modeling Autonomic Road Transport Support Systems," in *Autonomic Road Transport Support Systems*, T. L. McCluskey, A. Kotsialos, J. P. Müller, F. Klügl, O. Rana, and R. Schumann, Eds. Springer International Publishing, 2016, pp. 67–85.

[26] L. Heilig, R. R. Negenborn, and S. Voß, "Cloud-Based Intelligent Transportation Systems Using Model Predictive Control," in *International Conference on Computational Logistics*, F. Corman, S. Voß, and R. R. Negenborn, Eds. Springer International Publishing, 2015, pp. 464–477.

[27] L. de Penning, A. S. d'Avila Garcez, L. C. Lamb, A. Stuiver, and J. J. C. Meyer, "Applying Neural-Symbolic Cognitive Agents in Intelligent Transport Systems to reduce CO2 emissions," in *International Joint Conference on Neural Networks (IJCNN)*. IEEE Computer Society, 2014, pp. 55–62.

[28] R. Tse and G. Pau, "Deployment of vehicular edge clouds: lessons and challenges," in *Eighth International Conference on Digital Image Processing (ICDIP 2016)*, August 2016, pp. 1–6.

[29] G. Pau and R. Tse, "Challenges and opportunities in immersive vehicular sensing: Lessons from urban deployments," *Signal Processing: Image Communication*, vol. 27, no. 8, pp. 900 – 908, 2012, special issue on: pervasive mobilemultimedia.

[30] M. Vadiveloo, R. Abdullah, M. Rajeswari, and A. A. Abu-Shareha, "Image segmentation with cyclic load balanced parallel fuzzy c-means cluster analysis," in *Imaging Systems and Techniques (IST), 2011 IEEE International Conference on*. IEEE, 2011, pp. 124–129.

[31] P. J. Fernandez, J. Santa, F. Bernal, and A. F. Skarmeta, "Securing vehicular ipv6 communications," *IEEE Transactions on Dependable and Secure Computing*, vol. 13, no. 1, pp. 46–58, Jan 2016.

[32] P. Tan, M. Steinbach, and V. Kumar, "Introduction to data mining, addison wesley publishers," 2006.

[33] A. K. Jain, "Data clustering: 50 years beyond k-means," *Pattern recognition letters*, vol. 31, no. 8, pp. 651–666, 2010.

[34] A. Flores-Sintas, J. Cadenas, and F. Martin, "A local geometrical properties application to fuzzy clustering," *Fuzzy Sets and Systems*, vol. 100, no. 1, pp. 245–256, 1998.

[35] I. Timón, J. Soto, H. Pérez-Sánchez, and J. M. Cecilia, "Parallel implementation of fuzzy minimals clustering algorithm," *Expert Systems with Applications*, vol. 48, pp. 35–41, 2016.

[36] A. Flores-Sintas, J. M. Cadenas, and F. Martin, "Detecting homogeneous groups in clustering using the euclidean distance," *Fuzzy Sets and Systems*, vol. 120, no. 2, pp. 213–225, 2001.

[37] J. Dean and S. Ghemawat, "Mapreduce: simplified data processing on large clusters," *Communications of the ACM*, vol. 51, no. 1, pp. 107–113, 2008.

[38] H. Hilpert, L. Thoroe, and M. Schumann, "Real-time data collection for product carbon footprints in transportation processes based on obd2 and smartphones," in *2011 44th Hawaii International Conference on System Sciences*, Jan 2011, pp. 1–10.

[39] V. Ribeiro, J. Rodrigues, and A. Aguiar, "Mining geographic data for fuel consumption estimation," in *16th International IEEE Conference on Intelligent Transportation Systems (ITSC 2013)*, Oct 2013, pp. 124–129.

[40] K. Mori, O. Shagdar, S. Matsuura, M. Tsukada, T. Ernst, and K. Fujikawa, "Experimental Study on Channel Congestion using IEEE 802.11p Communication System," in *IPSJ Technical Workshop on Mobile Computing and Ubiquitous Communications*, 2013.

[41] "Directorate-General of Traffic. Vehicle fleet annual directory. 2016," http://www.dgt.es/, 2017, [Online; accessed October 10, 2017].

[42] "Top 500 supercomputer site," http://www.top500.org/, 2017, [Online; accessed November 15, 2017].

José M. Cecilia received his B.S. in Computer Science from the University of Murcia in 2005, his M.S. in Computer Science from the University of Cranfield in 2007, and his Ph.D. in Computer Science from the University of Murcia in 2011. He is Associate Professor at the Catholic University of Murcia. His research interests include heterogeneous architecture and bio-inspired algorithms.

Isabel Timón graduated in 2011 in Mathematics at the University of Murcia. She is a PhD candidate at Catholic University of Murcia in the area of high performance fuzzy clustering techniques for Big data.

Jesús Soto graduated in 1993 in Mathematics at the University of Murcia. He holds a PhD in Applied Mathematics from the University of Alicante in 2005. He is Associate Professor at the Catholic University of Murcia (UCAM). His research interests include fuzzy clustering.

José Santa received an MSc in Computer Engineering and an MSc in Advanced Information and Telematics Technologies in 2004 and 2008, respectively, and his PhD in Computer Science in 2009, all from University of Murcia. Currently, he is Senior Research Fellow at the same university. His research interests include ITS, mobile services and networks.

Fernando Pereñíguez received an MSc in Computer Engineering in 2007, and his PhD in Computer Science in 2011, both from University of Murcia. Currently, he is Assistant Professor at University Centre of Defence at the Spanish Air Force Academy. His research interests include security, privacy and handoff management in mobile networks.

Andrés Muñoz obtained his PhD in Computer Science in 2011 at the University of Murcia. He is a senior lecturer at the Catholic University of Murcia. His main research interests include multi-agent systems, applications in Intelligent Systems, Semantic Web technologies and Ambient Intelligence.