Fuzzy decision method to improve the information exchange in a vehicle sensor tracking system

Guillermo Cueva-Fernandez, Jordán Pascual Espada, Vicente García-Díaz, Ruben Gonzalez-Crespo

A R T I C L E   I N F O

Article history:
Received 20 September 2014
Received in revised form 22 January 2015
Accepted 23 January 2015
Available online xxx

Keywords:
Vehicle sensor
Fuzzy logic
Fuzzy decision making
Tracking
Resource management
Bandwidth

A B S T R A C T

Internet of Things is based on the identification of real-world objects in a unique way that interconnects them by means of communication interfaces. Such a simple idea allows the emergence of a huge number of new applications in almost any domain of knowledge. One of the most prominent areas of application is road vehicles, which currently have, on average, more than 50 sensors inside and which information can be accessed through a standard protocol. With this, vehicles have become real smart objects that can interact with other objects or any software system. To allow that, our previous work focused on proposing and developing the Vitruvius platform, where users with no programming knowledge can design and quickly generate Web applications based on the real-time data consumption from interconnected vehicles. The problem is that the sending of such information is out of control, being unable to filter out when the best time to send the information is and what information should be sent at any moment in order to minimize the resource consumption of the mobile device that acts as a bride between the vehicle and the database in which all the information is stored. Thus, in this work we propose a fuzzy algorithm that allows to optimize the resources that are used by real-time applications that constantly send data while maintaining data quality, contextualized in vehicle sensor tracking systems and the applications that can be built above them.

© 2015 Elsevier B.V. All rights reserved.

1. Introduction

Internet of Things (IoT) is a concept proposed by Ashton [1] that tries to identify real-world objects in a unique way, creating a graph-like structure with all the related information. The main idea is to integrate several technologies such as identification and tracking methods, wired and wireless sensors and actuators, communication protocols or distributed intelligence [2], being the “smart” objects the building blocks for the IoT [3]. Those objects can be used to create a large amount of different applications such as for example: (1) helping people with disabilities with assistance and support to achieve a good quality of life [4]; (2) efficient smart homes with ubiquitous services [5]; (3) therapy management in ambient assisted living taking into account different parameters such as illness, treatments, physical and psychological stress, drugs, etc. [6]; or (4) efficient logistics management [7]. Smart objects could be any classic real-world object with a digital component equipped with sensors, actuators and communication interfaces, usually Wi-Fi [8], Bluetooth or IrDA. Sensors and actuators are very important since they are used to gather information from the real world for different tasks of monitoring and react according to the information obtained.

An industry in which the sensors are becoming of vital importance is the transport industry. Modern vehicles have a large number of sensors that show important information about the current state of the vehicle (e.g., vehicle speed, fuel pressure, throttle position, engine RPM, engine coolant temperature, etc.). For that, the OBD-II (On Board Diagnosis) standard port is used [9], at least in all new passenger road vehicles in EU and USA. It provides a set of common mandatory sensors, opening the possibility of including brand-specific sensors. It makes it possible to directly join together the IoT and vehicles (e.g., He et al. worked on a multi-layered vehicular data cloud platform to provide services such as intelligent parking [10]).

Traditional logic deals with binary sets of values and is used in a huge amount of applications. However it is not always the best solution since the values can only be two, true or false. Thus, fuzzy logic [11] deals with reasoning what is approximate rather than
exact, by means of a truth value that ranges on a scale between 0 (completely false) and 1 (completely true). This idea is widely used in many fields such as artificial intelligence and control systems when there is no clear separation between what is true and what is false. The same idea could be applied to determine if it might be a good time to transmit data, and what data, through a computer system. As an example, with traditional logic we could say: "If there is new data and speed is high then send information to database". However, the real world is not usually so easy to manage and decide what is new, or what is a high value. It is not usually a binary issue since it can depend on various factors and, at the same time, in different degrees.

The goal of this work is to propose a fuzzy algorithm to reduce the resources that are used by real-time applications that constantly send data through any interface. This is expected to optimize their operation. The focus is to know when the best moment occurs to send the information and what information should be sent. We think that our previous work, the Vitruvius platform, is a clear example of a scenario in which this proposal could be useful since it is based on a real-time applications in which data (currently each vehicle has an average of more than 60 sensors on board) is constantly sent (between one and four seconds) through a resource-constrained device as the smartphone of any user.

The rest of this paper is organized as follows: Section 2 includes the background of related works, Section 3 describes the proposal and the approach followed to meet the goal, Section 4 shows the results and Section 5 discloses the conclusions.

2. Background

There is a growing concern about the amount of resources spent by computer systems [12]. Thus, new concepts like green computing progressively arise [13]. It refers to the study and use of environmentally sustainable computing and is a broad field that tries to efficiently and effectively manage computers, servers, communication systems or devices such as monitors, printers, disks, etc. It is such an important need today that many government agencies have implemented standards and regulations that encourage green computing (e.g., Energy Star program includes efficiency requirements for computer equipment since 2006 [14]).

However, it also includes the design and programming of good algorithms [15] to reduce the use of resources such as battery consumption, bandwidth of the network or CPU cycles. All of this leads to energy and monetary savings. As an example, an average Google search releases between 0.2 [16] and 7 [17] grams of carbon dioxide, depending on the source consulted.

There are a large number of works in which different algorithms have been applied to achieve resource waste reduction. For example, Mezmaz et al. [18] investigated the problem of scheduling precedence-constrained parallel applications on heterogeneous computing systems, proposing a parallel genetic algorithm that takes into account the energy consumption of the system. Other authors like Berl et al. [19] identified some of the research challenges that need to be addressed regarding cloud computing environments for energy efficient processing. However, the focus of this work is on resources for mobile devices which have some specific needs, and are greatly related to the search for energy savings.

From the point of view of mobile computing, there are special challenges [20] such as: (1) short duration of the batteries, which although they have increasingly greater capacity, they also have higher consumption because the resources used by devices (e.g., due to the increasing size of the screens); (2) data connections are usually much more constrained and limited than data wiring connections in a fixed place; and (3) total computational resources that are available are generally much lower than those available in a laptop or desktop computer. The following sections will deepen in those topics.

2.1. Focusing on battery usage

Batteries and their low duration have always been one of the weaknesses of mobile devices. Thus, to compensate the increasing power of such devices, batteries have been increasing their size too. However, that does not seem to be enough and various additional electronic gadgets have appeared to help keep active the devices for more time. For example, adapters to increase the battery capacity, and therefore the size of the mobile device, or external battery chargers that can be carried to charge the mobile device even where there is no power supply in which to be connected. However, that does not seem to be a global solution to the low duration problem of batteries and is one of the reasons why there are multiple research works focusing on mitigating the effects of the limitation of current batteries for mobile devices.

For example, Pathak et al. [21] claim there are lots of different types of bugs faced by millions of devices (energy bugs) that caused an unexpected amount of high energy consumption. Perucci [22] says that mobile phones should switch the network in dependency of the service (e.g., 3G networks need more energy for text messaging and voice services but 2D networks need more for downloading large amount of data) to save energy. Others like Lin et al. [23] designed an adaptive location service that helps to reduce the high battery drain caused by GPS receivers.

2.2. Focusing on network traffic

Although mobile devices have progressively bettered, with faster and with higher capacities, Internet connections (e.g., in many cities there are free WiFi areas for people), etc., we are still far from achieving full and unlimited connectivity. We depend on the rates offered by operators, which tend to be expensive and with many limitations from the point of view of the bandwidth and the amount of data that can be sent/received, especially when compared to the connections that can exist in fixed locations such as offices or homes. This makes reducing the amount of data that is sent and received a priority in many scientific papers. Although it is not always necessarily true, it is important to note that reducing the amount of data that is sent and the CPU cycles that are needed are likely to reduce the battery used.

Some interesting works include the one by Pering et al. [24] in which they propose a method to enable a device to automatically switch between multiple radio interfaces such as WiFi and Bluetooth in order to optimize the transmission. Marsan & Meo [25] evaluate the amount of energy that can be saved by using all networks when required, but progressively disconnecting networks when traffic decreases. Bojic et al. [26] proposed an agent-based mechanism for auto-tuning mobile networks with the aim of achieving energy savings when users try to access them. Ma et al. [27] created a tool for refactoring traditional websites to be adapted to smartphones taking into account, among other things, that under limited networks users might just want to load their desired contents, to reduce fee and speed up loading time.

2.3. Focusing on CPU cycles

Related to all the above aspects and many others is the CPU consumption of mobile devices. Currently, these devices are full-fledged computers with very powerful features but, mainly due to the cost of small components, they are far from what other more conventional computers could provide. This means that we have to be careful with application designs to get applications running.
quickly while leaving resources for other applications that could be running simultaneously on the device. Thus, the fewer and faster CPU cycles consumed by an application, the better. For this, different works focus on designing efficient and optimized algorithms, how to avoid executing instructions unless absolutely necessary, and how to properly schedule the execution of CPU instructions.

Thus, Yao et al. [28] explained a model of job scheduling to capture some important aspects of energy consumption and execute each task according to the computation given by different proposed algorithms and heuristics. Hsu and Kremer [29] designed a compiler algorithm that optimizes programs for energy usage using dynamic voltage scaling in order to identify regions in which the CPU can be slowed down to save energy consumption and with negligible performance loss. Or, for example, Vallina-Rodrigez & Crowcroft [30] propose a mobile operating system (ErdOS) with two main components: (1) a proactive resource management system that predicts the future resource demands; (2) an opportunistic access to computing resources available in nearby devices.

2.4. Fuzzy-based solutions adoption

In addition to the techniques and methods used in the previous studies, fuzzy logic has also been used with the aim of saving energy when mobile devices are used. Thus, for example, Chamod-rakas and Martakos [31] use a fuzzy set representation method that solves the inconsistency related to decision criteria in order to be connected to a network in the best possible way in terms of QoS performance and energy consumption. Larios et al. [32] propose a fuzzy algorithm to obtain the position of the device, obtaining less localization errors, better accuracy and avoiding the consumption of high amounts of energy needed by some elements like the GPS. Another example is Bagchi [33] who proposed a fuzzy adaptive buffering algorithm to try to maintain QoS of playback and, at the same time, saving energy consumption of the mobile devices when they are executing multimedia streaming. In any case, the use of fuzzy algorithms is highly employed in works that try to make decisions based on parameters that are not always clearly stated (e.g., [34,35]).

There are huge amount of works aimed at saving energy in the use of mobile devices. Lots of them use fuzzy logic to achieve their goals, especially when it is necessary to make choices between different possibilities from which there is no clear choice a priori. It therefore seems that the use of fuzzy techniques in our work may be a solution to the problem of high energy consumption required by our proposed Vitruvius platform when there is no control over the data that are sent in each cycle. However, although we could use some of the outlined solutions to save energy, not surprisingly, none of the previous work provides a concrete solution adapted to the domain-specific language used by our platform. The next sections therefore deepen in our domain-adapted solution.

Our previous work focused on proposing and developing the Vitruvius platform [36]. It is a Web environment that provides a domain-specific language for users to easily design, create and deploy rich Web applications that are automatically generated using standard Web technologies. Using Vitruvius the workflow is as follows: (1) a user defines an application using the Web interface; (2) a Web application is automatically generated based on the definitions made by the user; (3) the Web application can be accessed only by authorized users from any Web browser; (4) there is a loop in which vehicles send information from their OBD2 port to a database through the use of a mobile device acting mainly as a bridge. The defined Web application obtains the required information for the application from the database and finally, the Web application shows the requested information on a map.

Although the tests we have done with relatively few data sent during not very long periods of time worked properly, the platform is not very efficient in terms of resources consumption. Inefficient resource consumption produces a reduced scalability. The problems we detected are located on the mobile device that acts as a bridge between the vehicle and the database. It may become a bottleneck for the system because of: (1) high battery usage; (2) high network traffic; and (3) high number of CPU cycles executed. The reason for this problem is that the system tries to always send information every one to four seconds regardless of whether there is really new data or whether new data can provide valuable information for client applications. The continuous transmission of all data from sensors of the car to the database causes the mobile device to: (1) drain a large amount of battery, given that reduced battery life is one of the weaknesses of current mobile devices; (2) expend a great amount of network data in an environment where it is difficult to find Wi-Fi networks available; and (3) execute too many CPU cycles on the device, causing a reduction in performance.

3. Application of the proposed approach

Fuzzy logic can be used to solve some problems that can be very difficult or impossible to solve property with absolute rules. In this section, we present the evolution of the Vitruvius platform which applies a novel Fuzzy inference system designed especially to improve the performance in the mobile clients of the real-time systems.

3.1. Introduction to Vitruvius platform

Vitruvius is a platform which allows users to generate real time applications using road vehicle sensors. The applications generated with Vitruvius range from simple sensor notification applications to multi-sensor, multi-vehicle systems for road hazards. Vitruvius is a complete solution. It contains all the necessary elements at all levels to make fully working applications.

Vitruvius is managed by a web interface where users can design, generate, and deploy their own applications. Also as part of the platform, Vitruvius Car clients get in charge of sensor data gathering from vehicles.

Vitruvius web consists of an application editor that allows users to design sensor based applications. Users can choose from a wide range of sensors, such as, speed, lateral G’s, engine coolant temperature, throttle position, air pressure, and etc. This information can be obtained in real time or be retrieved from past records. This data can be used to generate applications to create routes, alarms or notifications to other vehicles. An assistant wizard is available to help the development of applications. Users can just click on the kind of application, vehicles or sensors that they want to add to their application. Once the design is completed, the user can generate the application. The resultant application is a HTML5 with JavaScript application that gets instantly deployed in the Vitruvius server. The application will automatically start running and will be ready to be used. Applications can be public or shared with other authorized Vitruvius users.

In order to collect vehicle data and feed Vitruvius applications, Vitruvius Car can be installed to Android devices and can be connected to many different sensors in vehicles. These connections are done via Bluetooth to OBD2 compatible devices that can be connected to the service port from vehicles. Data is collected and depending on the sensor values, data is immediately sent or stored to be sent later to the Vitruvius centralized database.

3.2. Fuzzy inference systems

Fuzzy logic can be applied to solve a great number of problems and could be combined with a range of different techniques and algorithms. The approach is said to be pure when only IF-THEN rules
Fig. 2. Fuzzify functions for the input variables: time_from_last_data_send, variation_position, variation_important_values, variation_all_values.

- Can wait – this is the unique case in which postponing sending the information is acceptable. The changes in the values are not important and the delivery of information can be merged with the next information sent.

DEFUZZIFY send
TERM yes := (0,1) (2,1) (4,0);
TERM recommended := (3,0) (5,1) (7,0);
TERM canwait := (6,0) (7,1) (10,1);
METHOD : COG; // Use 'Center Of Gravity' defuzzification method
DEFAULT := 0; // Default value is 0 (if no rule activates defuzzifier)
END_DEFUZZIFY

The mobile client will send the data to the server when the output parameter will be Yes, or Recommended > 0.5.

At the final end of the process, after the expert’s advice, simulations, and real tests, the set of fuzzy rules selected are the follows:

RULEBLOCK No1
AND : MIN; // Use 'min' for 'and' (also implicit use 'max' for 'or' to fulfill DeMorgan's Law)
ACT : MIN; // Use 'min' activation method
ACCU : MAX; // Use 'max' accumulation method

RULE 1:
IF time_from_last_data_send
   IS t_few
THEN send IS canwait;

RULE 2:
IF time_from_last_data_send
   IS t_few
   AND variation_position
   IS a_enough
THEN send IS recommended;

RULE 3:
IF time_from_last_data_send
   IS t_few
   AND variation_important_values
Fig. 3. Fuzzify functions for the output variable.
Please cite this article in press as: G. Cueva-Fernandez, et al., Fuzzy decision method to improve the information exchange in a vehicle sensor tracking system, Appl. Soft Comput. J. (2015), http://dx.doi.org/10.1016/j.asoc.2015.01.066

These sets of rules force data delivery when the vehicle data experience a significant change between the current data and the last data sent. These rules also enable the data merger when the changes in the vehicle data are light or only affect minor vehicle variables.

**RULE 4:**
IF time_from_last_data_send
IS t_few
AND variation_all_values
IS a_enough
THEN send IS recommended;

**RULE 5:**
IF time_from_last_data_send
IS t_few
AND variation_all_values
IS a_enough
AND variation_position
IS a_enough
AND variation_important_values
IS a_enough
THEN send IS yes;

**RULE 6:**
IF time_from_last_data_send
IS t_few
AND variation_position
IS a_enough
THEN send IS yes;

**RULE 7:**
IF time_from_last_data_send
IS t_enough
AND variation_important_values
IS a_enough
THEN send IS yes;

**RULE 8:**
IF time_from_last_data_send
IS t_enough
AND variation_all_values
IS a_many
THEN send IS yes;

**RULE 9:**
IF time_from_last_data_send
IS t_many
THEN send IS yes;

**RULE 10:**
IF variation_position
IS a_many
THEN send IS yes;

**RULE 11:**
IF variation_important_values
IS a_many
THEN send IS yes;

**END RULEBLOCK**

4. Evaluation scenario and result

We evaluate the performance of the mobile client application in three real environments. We did three urban travels on city and highway roads. We collected the information from the vehicles using the Vitruvius system and stored the data in an evaluation database. We developed a new mobile client that instead of using the real data of the car uses the data from the database. There were two types of mobile clients:

- (1) **Mobile client**. The first mobile application is the old Vitruvius mobile client. This application works in a standard way and sends the vehicle data to the server every second.
- (2) **Mobile client, Fuzzy method optimization**. The mobile application client is basically the first application, although it integrates the fuzzy module that determines when the data will be sent to the central server.

We run both applications with the same data in three trips of different durations: 20 min, 30 min, and 60 min. We execute the two applications for the same datasets that were obtained in real trips. To analyze the performance we monitored several of the most relevant factors: memory and CPU usage (Fig. 4) and the network traffic (Fig. 5).

The Smartphone used to run the applications was a Galaxy Nexus. The device has the following technical features: OS – Android 4.2.2, Bluetooth v3.0 with A2DP, CPU – dual-core 1.2 GHz...
Cortex-A9, Memory – 1 GB RAM. The tools used to monitor the performance parameters during the use of the applications were Advance Task Manager and SystemPanel.

The result set obtained shows the expected effects on the application (2) Mobile Client. Fuzzy method optimization can reduce the data network in some scenarios in a very significant way. In these cases the input data was reduced between 51.54% and 53.6% and the output data between 56.1% and 58%. This is a very big reduction in the use of the mobile network.

The fuzzy method included in the application (2) uses more CPU to analyze the data. Although we tried to reduce the impact in the CPU usage of the phone, the application (2) increases the CPU usage between 28.8% and 29.4%. This consumption of ram memory during the execution of the application was stable. It even improved the consumption of the application (1) by 1% although that is a slight reduction and not very significant.

5. Conclusions and future work

In this research we present a fuzzy method that can be used to improve the performance in the mobile clients of the real-time systems. We integrated the proposed algorithm in the mobile client of the platform Vitruvius. The mobile clients of the Vitruvius platform are Android applications that send all the vehicle and sensor data to a centralized server. The information is used in the Vitruvius generated applications. The Vitruvius generated applications can alert and show in real time information about the connected network.

Our approach is based on some cases where information can be merged, especially if the information is not critical in the current moment. Instead of sending a petition every second, an intelligent system can decide to merge particular petitions. With this approach we can significantly improve the performance of mobile clients. This is one of the most critical parts of the Vitruvius system as in many of other real time systems in very different domains.

To reduce the data sent from the Vitruvius mobile clients to the server, we use a set of fuzzy rules that were mainly defined by Vitruvius experts. This way the experts defined rules that are very useful to optimize the number of petitions that the mobile client sends to the server. With the expert knowledge, it is possible to allow clients to only send the information that is needed to the Vitruvius platform. Vitruvius applications can run properly, even if they postpone the delivery of some non-priority data. According to the evaluations performed in real scenarios the proposed algorithm achieved reduction of the data network input data between 51.54% and 53.6%, and the output data between 56.1% and 58%. It shows a very big reduction of the use of the mobile network. On the other hand the computational cost of the execution of the fuzzy method in the mobile client provoked an increase in the CPU usage between 28.8% and 29.4%. This is relatively high but acceptable in respect to the significant improvements achieved in the network traffic reduction.

The final conclusion based on the results is that it can be possible to improve the performance of mobile software applications that send real data to servers repeatedly. To achieve this aim we can use the knowledge of the platform experts combined with analysis and test processes to refine and optimize the system.

Fuzzy algorithm could be applied with many other different sensors. Vehicles with special requirements, such as trucks delivering special goods or dangerous materials, can also benefit from this application. The application could easily accommodate data from extra sensors and be added as part of the input. Adding a new subset of rules would help prioritize the delivery of data.

In addition, not only road vehicles can apply the same principle. Other means of transportation, such as small aircrafts and boats can benefit from the same algorithm. These vehicles can have severe network limitations occur due to the nature of the environment in which they travel. Usually Internet connection can be provided by satellite. Satellite Internet connection method is very expensive and usually users are charged by traffic. In these cases, the algorithms applied in the Vitruvius client’s application could improve the optimization of network resources without deteriorating the monitoring quality. It would allow tracking systems to work more efficiently by taking in consideration only events that require an urgent communication to the server.

References


Please cite this article in press as: G. Cueva-Fernandez, et al., Fuzzy decision method to improve the information exchange in a vehicle sensor tracking system, Appl. Soft Comput. J. (2015), http://dx.doi.org/10.1016/j.asoc.2015.01.066


