

Efficiently Monitoring Dynamic Data-Streams in Smart Cities

Hassan LOULOU^{1,2}, Sebastien SAUDRAIS², Hassan SOUBRA² and Cherif LAROUCI²

¹ Paris-Sud University, Paris, France

`Hassan.loulou@u-psud.fr`

² ESTACA-LAB, LAVAL, France

`sebastien.saudrais@estaca.fr`, `Hassan.SOUBRA@estaca.fr`, `Cherif.LAROUCI@estaca.fr`

Abstract

In the context of smart cities, data comes from heterogeneous sources (intelligent transportation devices, transportation social networks, autonomous vehicle sensors, etc). This continuously received information constructs data streams and has different representations and very big sizes. In order to obtain smarter transportation, the central monitoring system has to use this information in real-time, considering the uncertainty of the received information and its evolving structure. This article proposes an approach to improve the reasoning on dynamic data streams by suggesting alternative solutions at runtime and by predicting their costs. These alternative solutions will be exploited later to make the system fault tolerant. Our approach is applied on an illustration example: autonomous vehicles.

1 Introduction

The use of sensors is constantly growing in our daily lives, specially in the transportation sector. Data is coming continuously from these sensors allowing us to monitor the actual situation of the traffic in real-time. We are interested in the parts of the smart cities which could be critical and where the software system errors could have real dangerous impacts on the system. Monitoring cyber-physical systems in real-time is of high importance as it is the key to make the appropriate reactions in near real-time. Moreover, testing all the possible behaviors of a software system at design time is impossible, especially, when the system is large and has a certain degree of uncertainty at run-time. Therefore, we need to know continuously about the actual situation of these critical systems and to predict the consequence of events which could be the reason of their anomalies and their deviation from the planned situation to avoid them in the future. A modern vehicle is equipped with a great number of sensors and different technologies have appeared for connecting a vehicle to its environment. Moreover, many companies have developed advanced versions of autonomous vehicles which reached an important level of maturity and their future seems to be promising. In this context, we are facing the following problem: Could we decrease the overhead and the latency when we depend on a central monitoring system by replacing it with the local vehicle resources for local decisions? Is it possible to use the available local sensor data in the vehicle neighbors' sensors for the previous task? And could we predict the resource overload caused by these tasks allocations? Another important problem for us is: Could we change the strategy of the information system by applying adaptation at run-time in order to execute the system in the best safety conditions? for example when we have autonomous functionalities, specially in the context of autonomous vehicles. Thus, the proposed solution must be fault-tolerant and must propose alternative strategies when failures happen.

Using central systems for real-time monitoring and analyzing continuous big data streams (using cloud computing) suffers from high delays for reserving new nodes. Moreover, the system can not predict the fluctuations which require rapidly changing amounts of resources. There is also the possibility of losing the connection between the distributed and the central systems. Moreover, there is an increased overhead for extracting the contextual information of these distributed systems instead of delegating some tasks to the distributed processing units which would have probably all the needed information for processing specific encountered problem. However, even when the system delegates the processing responsibility to the distributed processing units (the vehicles) which have limited resources, the available resources would have high overload specially for the autonomous functionalities which may consume a lot of resources. Therefore, there is a need for a framework for analyzing the usage patterns of the resources according to the context and to the received information, in other words, it would be useful to monitor the software environment to predict its impacts on the internal situation of the software system. Young *et al.* [7] recently tried to find out this relation. However, this work is limited to a specific task allocated on the local resources of the vehicle. Moreover, our work will exploit the relation between the dynamically changing data streams and the best usage of the available distributed heterogeneous resources in order to make the best task distribution at runtime. Additionally, the strategic trajectory and the tactic motion planning generates multiple possible plans to the steer-by-wire system and each one has a specific cost. Thus, recalculating the trajectory (behavioral map) to find the one with lowest overhead is necessary. Unlike the previous work, our approach will consider the information coming from different sources to avoid the failure or the uncertainty of sensors' data by combining different alternative solutions and introduce the possibility to move among them at runtime.

The model@runtime [5, 6] in such systems can take the form of a collection of rules and it makes reactions to the inferred new internal or external conditions of the system. The collection of rules could change all over the time as new facts are always received. Some of the datasets could not be used for the real-time traffic analysis objectives, but they are used for estimation and prediction objectives [4]. Those data come from different sources not related directly to the vehicles such as the activities in the city. The way we receive data can be static or dynamic, Quasi stream or real stream. On the other hand, It could be in different formats such as CSV, XML, texts, photos, etc. as explained in [4]. Lecue *et al.* [4] do an off-line compiling of the previous diagnosis information into a state machine. They do not study the impacts of the on-line received information on the system overload. Moreover, this approach does not consider how the actual streaming data would impact the already extracted rules for the historical state machine. The real-time diagnosis is performed in their work only by analyzing the historical versions of the state machines with the actual circumstances to find the similarity between them. The work of Lecue *et al.* is similar to our first objective as they try to find out the potential explanation and the potential time of traffic congestion by analyzing the big-data streams. This is important for us to find out the relation between the incidents happening in the environment and the corresponding overhead of the system. Moreover, our work will anticipate the need for new resources by analyzing these historical state machines. Additionally, we are going to preprocess these data or even take a decision depending on the local data in the vehicles when it is possible.

Our methodology will decide the best alternative strategy at runtime depending on a runtime model of the available resources. This will help us to take the initiative when there is an anomaly for adapting the system to the safest strategy. This proactive approach depends on an on-line and off-line analysis of the big-data streams in order to figure out or to anticipate the circulation patterns in the city and to find out its relation with the software new requirements.

2 An Approach to Improve the Reasoning on Dynamic Data Streams

We are going to exploit the cyber-physical internal and external contextual data in order to estimate the needed resources and to adapt the proposed solutions in accordance with them and with the estimated future available resources (steps 1 and 2 of the figure 1). Therefore, we are building a reflective model with a causality connection with the system components. Thereafter, in step 3, we are going to figure out the possibility of using the existing vehicles' resources for data preprocessing before sending them to the server to avoid (latency, central system overhead, etc.) and to make proactive safety actions. However, this processing could cause a high overhead on the side of the vehicle. So, we may need to monitor the local computation resources in the vehicle side (as explained by the loop going back from the step 4 to the step 1) to avoid failures. Anyway, some information will be always processed on the central system especially for the strategic decisions.

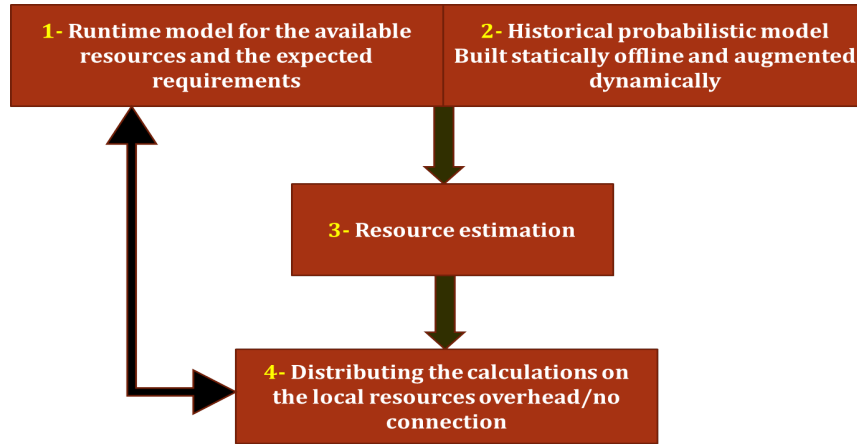


Figure 1: The proposed approach

2.1 Runtime Model for the Expected Available and Required Resources

Runtime models reflect the actual situation of the system when we receive new data causing reconfigurations at runtime [2]. It must help in capturing the new data but it must also reserve the previous snapshots of the system in order to reason about the new needed decisions. The representation of this information must help to make reasoning operations in near real-time. Moreover, we need to avoid the overhead caused by the previous operations. Considering the previous constraints, we are going to choose the most appropriate runtime model for our requirements. We are building a reflexive model for representing the runtime information of all the available resources to know how can we distribute the processing responsibilities on the available distributed resources. To this end, we are building a meta-model representing all kinds of available resources and their characteristics. Thereafter, we are going to exploit the possibility of making a decision about where to make the calculations, on the vehicle resources with Autosar operating system, on other systems attached to the vehicle's like android which offer an adaptive framework using Felix, OSGI for android [8, 1] using virtualization techniques

or on the cloud new reserved resources. Thus, we need to see if the information captured by the vehicle and by its contacts with the other vehicles it has met is sufficient. Moreover, we are examining if the available resources are sufficient to do the calculations locally. In this way, a complete solution to make a fault tolerant approach for tackling the peak overhead incidents in this critical system will be developed. The runtime model in step 1 has a feedback connection from the dynamic distribution component in order to make a new decision according to the new allocation of the resources. As a result, the runtime model is used for continuous planning. We will use a profiling framework introduced in [3] for capturing information from vehicle resources which is oriented to the autonomous vehicles systems. Depending on it, we will analyze the resource usage and (re-)configure the amount of resources allocated to tasks, redistribute them or recalculate their priority dynamically.

2.2 Historical Probabilistic Model

In this step, we try to answer the following three questions: 1- find out the best model representing rapidly changing big data. 2- how to estimate the overload of the on-line decision and the data-structure evolution? 3- How to decide the needed information for a specific context of this cyber-physical system? The probabilistic model must be built statically off-line and augmented dynamically on-line. We will analyze the information semantically to predict the potential resource consumption peaks and take the initiative when there is an anomaly. On the other hand, the Off-line process compile the previous diagnosis information into a state machine, This information could come from an open source available information such as those of linkgeodata.org which is available as RDF graph. For the On-line processing, we will solve the problem of adding the dynamic information efficiently to the complex big linked data-structure by reducing the amount of the required search for adding new information in the best place (a cluster, for example). For the preprocessing phase, we are going to process the information coming from different autonomous vehicles' sensors to eliminate the redundancy and link the different sources and extract a coherent meaning from them. This operation must be done also in real-time. The chosen data representation must be effective to ask questions such as: What is the congestion propagation on the nearby roads? It must be enriched with semantic city events and relates the past congestion conditions to the actual traffic conditions to make predictions.

2.3 The Resource Estimation

In this step, we aim at comparing the estimated available resources calculated by the runtime model and the estimated required resources calculated by the probabilistic model calculated in the step 2. This estimation changes continuously due to the fluctuating arrival rates and the impacts of these received data on the evolution of the used data-structure. In addition, the uncertainty about the data and the requirements make it difficult to estimate the load of the applied algorithms for all the active connections and queries asked either to the central monitoring or to the distributed systems. Anyway, we are going to optimize this estimation depending on the previous (runtime, real-time) and depending on the feedback information coming from the step 4.

2.4 Distributing the Calculation

This step will be built by supposing different solutions according to the context and by combining these different solutions with the ability to adapt at runtime to reflect the new system

requirements. Therefore, when the system encounters a problem or predicts its possibility, it takes proactive actions such as delegating the calculation to the vehicles, reserving new resources in advance or exchanging specific information with other vehicles when encountering sensors anomalies, for example. The decision of using an a specific distributed processing solutions will exploit the reflexive runtime model (using appropriate adaptive software solutions: OSGI and the variability models: feature models, or new adaptive solutions proposed for the vehicles internal system) and depending on the resource estimation step results.

3 Conclusion

In this paper, we proposed a general approach for monitoring the smart transportation system. Our approach aims at making estimations about the next system overhead peaks and the corresponding resource overloads. It also aims at proposing alternative solutions and reasoning about the best one at runtime. Currently, we are working on this model using the state of the art model@runtime concepts for representing the system internal information. Another runtime model will represent the cyber-physical systems external environment and it will predict the environment's future conditions to adapt the software system internal structure accordingly. Thereafter, we are going to validate our approach using a case study related to the autonomous vehicles. This system will take all the tactic and the strategic real-time data into consideration for choosing the best plan from the available ones, especially to fulfill the safety requirements.

References

- [1] Ming-Chiao Chen, Jiann-Liang Chen, and Teng-Wen Chang. Android/osgi-based vehicular network management system. *Computer Communications*, 34(2):169–183, 2011.
- [2] Holger Giese, Nelly Bencomo, Liliana Pasquale, AndresJ. Ramirez, Paola Inverardi, Sebastian Wtzoldt, and Siobhn Clarke. Living with uncertainty in the age of runtime models. In Nelly Bencomo, Robert France, BettyH.C. Cheng, and Uwe Amann, editors, *Models@run.time*, volume 8378 of *Lecture Notes in Computer Science*, pages 47–100. Springer International Publishing, 2014.
- [3] Hyoseung Kim, Junsung Kim, and Ragnathan Raj Rajkumar. A profiling framework in linux/rk and its application. *RTSS@ Work*, 2012.
- [4] Freddy Lécué, Simone Tallevi-Diotalle, Jer Hayes, Robert Tucker, Veli Bicer, Marco Sbodio, and Pierpaolo Tommasi. Smart traffic analytics in the semantic web with star-city: Scenarios, system and lessons learned in dublin city. *Web Semantics: Science, Services and Agents on the World Wide Web*, 27:26–33, 2014.
- [5] Brice Morin, Olivier Barais, J Jezequel, Franck Fleurey, and Arnor Solberg. Models@ run. time to support dynamic adaptation. *Computer*, 42(10):44–51, 2009.
- [6] Francisco Javier Acosta Padilla, Frederic Weis, and Johann Bourcier. Towards a model@ runtime middleware for cyber physical systems. In *Proceedings of the 9th Workshop on Middleware for Next Generation Internet Computing*, 2014.
- [7] Young-Woo Seo, Junsung Kim, and Ragnathan Rajkumar. Predicting dynamic computational workload of a self-driving car. In *2014 IEEE International Conference on Systems, Man, and Cybernetics, SMC 2014, San Diego, CA, USA, October 5-8, 2014*, pages 3030–3035, 2014.
- [8] Mussab Zneika, Hasan Loulou, Fatiha Houacine, and Samia Bouzefrane. Towards a modular and lightweight model for android development platforms. In *2013 IEEE International Conference on Green Computing and Communications (GreenCom) and IEEE Internet of Things (iThings) and IEEE Cyber, Physical and Social Computing (CPSCom), Beijing, China, August 20-23, 2013*, pages 2129–2132, 2013.