ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

PERFORMANCE STUDY IN AUTONOMOUS AND CONNECTED VEHICLES A INDUSTRY 4.0 ISSUE

[1]ALESSANDRA PIERONI, [2]NOEMI SCARPATO AND [3] MARCO BRILLI
[1]Guglielmo Marconi University, Dept. of Innovation and Information Engineering, Rome, Italy
[2]San Raffaele Roma Open University, Dept. of Human sciences and Promotion of the quality of life, Rome, Italy
[3]Marco Brilli, Sinergia s.c.a.r.l., Rome, Italy
E-mail: [1]a.pieroni@unimarconi.it, [2]noemi.scarpato@unisanraffaele.gov.it, [3]brilli.m@tiscali.it

ABSTRACT

Industry 4.0 represents an opportunity. The term 4.0 is used to specify the current industrial revolution, not-only from the technological point of view but also from the economical, sociological and strategical point of view. Indeed, this revolution involves different sectors: from manufacturing to healthcare. Its disruptive diffusion is due to several enabling technologies, such as Internet of Things (or Internet of Everything or Industrial Internet of Things), and, as said, it is a vision rather than a technological step forward. In this evolutionary process, the Connected and Autonomous Vehicles (CAVs) represent the perfect connection between technology and society world, an issue that stands actually in the center of Industry 4.0.

This article intends to extend a previous work of the authors, in which a new and non-conventional approach to manage the great amount of data that is generated by the CAVs has been proposed. In particular, the validity of the approach will be demonstrated by means of performance indices specifically defined for this case study.

Keywords: *Autonomous Vehicle, Connected Vehicle, Big Data, Graphdb, Internet Of Things, Industry 4.0, Performance Indices.*

1. INTRODUCTION

Industry 4.0 is characterized by innovative enabling technologies that provide the instruments through which new collaborative environment is achieved in which humans and systems may heavily interact, taking advantages from self-organizing and optimizing in real-time (see fig.1).

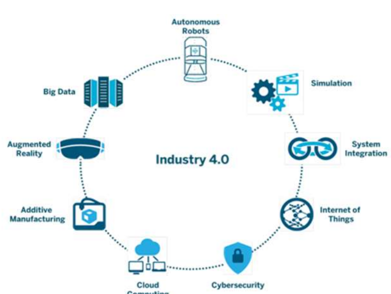

Figure 1: Industry 4.0 Enabling Technologies.

The development of the processes belonging to the so-called fourth industrial revolution are driven by innovations in the areas of IT, embedded systems, production, automation technique and mechanical engineering. The aim consists of creating new factories able to manage much more complex systems as we know today [1]. Smart products and smart production equipment are connected with the network and will overview the entire process, from the product idea, the product design, the supply chain and manufacturing. This approach allows to obtain more efficient results in all the value-chain production. Smart production also covers the delivery of products to the end users, after sales services and product recycling. The connection of all elements within the value chain in real time is considered the basis of Internet 4.0.

Even the automotive industry is experimenting new challenges and frontiers with the so-called autonomous and connected vehicles, which are becoming “smart” and totally connected with the rest of the world through Internet technologies. Indeed, the improvements in modern technologies

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

have allowed the comprehensive integration of essential systems and data for autonomous vehicle operations. This allows the vehicles to process decisions based on defined criteria informed by actual conditions, such as:

- Global Position System (GPS) – Satellite-based global location and time reference of vehicles for an accurate and constant position tracking.
- Inertial Navigation System (INS) – Monitors and calculates positioning, direction and speed of vehicles, assisted by sensors on-board.
- Laser Illuminated Detection and Ranging (LIDAR) – Laser detection sensors to identify surrounding objects.

On the other hands, collectively all the synchronization mechanisms provide the decision-making data that are necessary for the vehicle to be aware of position, traffic conditions and possible movements.

Currently, it is possible to identify two different categories of smart vehicle: AV (Autonomous Vehicle - vehicles that perform all driving functions with or without the "human driver", also called vehicles without driver) and CAV (Connected Autonomous Vehicle - vehicles provided with advanced communication technologies toward other vehicles or infrastructure). Both of them will lead new investment in urban infrastructure in order to reinforce features (especially of wireless communication) between the vehicle and the infrastructure at the edge of the road (e.g. smart lamppost) to transmit in real time an increasing amount of bidirectional data between vehicle and the urban infrastructure.

In a previous paper [2], a non-conventional approach to manage the huge amount of bidirectional data has been presented, being based on the use of Graph DB technology.

The aim of the current paper, instead, consists of demonstrating the validity of the approach, defining a set of performance indices in order to measure the advantages provided from the use of Graph DB technology.

To perform the above described goals, the communication between CAV and the neighboring urban infrastructures is considered and simulated through a case study that will be illustrated in the following sections.

In particular, next sections illustrate a brief overview on the Connected Autonomous Vehicle and a comparison has performed between the two considered methodologies able to threat with the big data generated from the sensors on-board vehicle and from the urban digital infrastructure. Being the first methodology based on a RDBMS conventional approach and the second based on a non-conventional GraphDB approach methodologies.

2. THE TECHNOLOGICAL IMPACT OF CAVS

Several vehicles have been designed and developed to be tested in various markets around the globe [3].

Figure 2 illustrates the conceptual evolution vision of a smart, integrated, dynamic and connected society in which the diffusion of connected autonomous vehicles is growing. Focusing the attention on the bidirectional flow of data between vehicles and infrastructure, a non-negligible aspect that has to be considered concerns the regulation about the privacy of the exchanged of data.

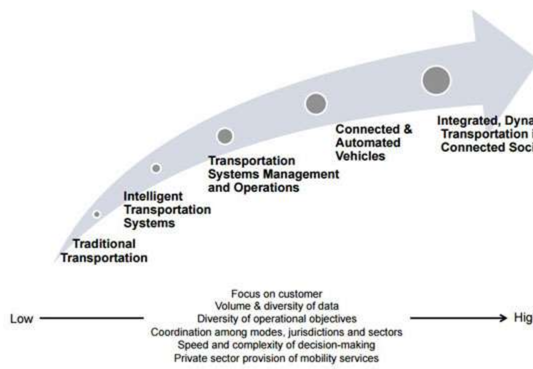

Figure 2: US Department of Transportation's (USDOT) CAV evolution vision.

Regulations exist, varying from state to state, that govern this issue which include the privacy [4] of the collection, the preservation, and the use of data in the connected society. Several protocols have been implemented to collect data from vehicles to infrastructure (V2I) and between vehicles (V2V). The former is better defined as an infrastructure communication technology, while the latter as a cooperative communication technology.

Obviously, in the V2I protocol the infrastructure plays a coordination role by gathering global or local information on traffic and road conditions and then suggesting or imposing certain behaviors on a group of vehicles. The V2V protocol, instead, is more difficult to realize because of its decentralized structure, and aims at organizing the interaction among vehicles and possibly developing collaborations among them [5].

Progress in on-board computational technology has introduced "adaptive" intelligence, in other words vehicle change its behavior according to environmental conditions (for example, traffic condition). This processing must be performed in real-time, considering at the same time the speed of the vehicle, the obstacles present in the roadway and the traffic conditions, in order to have the possibility to make appropriate changes keeping the vehicle in the lane and suggesting alternative itineraries.

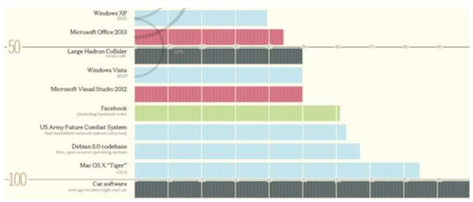

Figure 3: Complexity of modern software [4]

Figure 3 illustrates an estimation of the complexity (in MoC - Million-of-line-Code) of the software modules that are provided for a vehicle totally connected compared to the software on board the vehicle with other existing functionalities.

It is possible to observe that the software embedded in a totally-connected vehicle can reach and exceed 100 millions of lines of code [6]. Even if the measure of line-of-code is not widely recognized, it can represent a valid indicator of the software complexity in this type of vehicle.

For all the cited reasons, it is mandatory reduce the computational time, the complexity of the embedded software while maintaining at the same time the availability and the reliability of software itself.

This software complexity requires the availability of processors with increasing performances. In the literature, this issue has been faced with different approaches. An interesting solution is based on the use of mixed architectures composed by a main microprocessor and a hardware accelerators[7][8][9][10]. On the other hand, due to the inherently complexity of the studied systems, distributed simulation approach may be considered in order to study the performance of this kind of systems [11][12][13][14].

As a consequence, the complexity of the vehicle embedded software is increasing, not only for the above-mentioned growing-up of the existing interactive functionalities, but also for the development of the orchestration layer that allows the communication between the various software modules and the on-road infrastructure. Next sections illustrate the conventional approach to manage the big data originated by urban infrastructural sensors and by the sensor on-board and the non-conventional proposed approach using GraphDB methodology.

3. TYPICAL BIG DATA INFRASTRUCTURE

In literature the term Big Data is not clearly defined [15], in general, we can affirm that big data is a large set of heterogeneous information. Many attempts are been provided to realize collaborative environment for information management via artificial intelligent technology[16][17] these attempts should been applied in very different field such as legal domain [18][19], earth observation [20], Cultural Heritage [21].

In order to exploit Big Data, we need to use the so-called Big Data Technology and Analytic Methods instead of traditional data mining methods.

We can find Big Data Technology in a very different research areas such as Medicine [22][23][24], IoT [4] and Internet of Vehicles [2][25].

For example, in the automotive sector, the next-generation PW1000G family of engines (manufactured by Pratt & Whitney, United Technologies division) installed on new Airbus A320 aircraft, count up to five thousand sensors that transmit data approximately 10 GB / Sec for each single engine, this brings the total amount of data generated, in one hour flight, to one TB. By comparison, one Formula 1 Williams car, with its 200 sensors installed, generates about a thousand streams for a total of 250 GB data for each trial session (weekend).

Another example of this is the experiences gained by the Ford® manufacturer, who, utilizing the Big Data massively to having the best advantage from it, as well as other major automotive manufacturers.

Therefore, it is natural to use Big Data when we need to manipulate a lot of data came from motion sensors and electronic control boards on board vehicle. Otherwise, this amount of data could become unmanageable.

On the strictly technical side, the example of Google ™ Car is impressive: from field tests, with the set of active sensors, the Google ™ car generates a data volume of approximately 1 GB / sec. Considering an average (for the United States alone), people drive a single vehicle about 600

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

hours per year, producing a data volume of more than 2 PB (approximately 2000 TB or two million GB). Since worldwide, the number of cars is near to one billion units, it is difficult to imagine the amount of data involves. In this context, then, the number of information provided by the various vehicles is intended to increase sharply, with a consequent increase in the complexity of the management of the same. One of the possible solutions to deal with the management of these moles of data is the appeal to cloud infrastructure;

In this context, the number of information provided by the various vehicles is expected to increase dramatically, resulting in augmented complexity in managing them. One of the possible solutions to deal with these amount of data is the employ of the cloud infrastructure because this type of infrastructure is based on "Commodity Hardware" where you can scale the number of nodes in the order of thousands of units. In this way, there is a high storage capacity, typically accomplished through the use of local hard disks that then constitute a distributed file system.

The presence of a network typically at high speed completes the infrastructure. The use of these resources requires that the customer (user), pay only a predetermined fee that corresponds to an hourly cost, or to the number of nodes occupied, or storage used.

The advantages, from an operational point of view, are:

- Low initial costs: these costs are reduced to a minimum, since no initial investment is required; there is only one annual fee for service delivery.
- Backup Procedures: These procedures allow data to be saved (in aggregate or raw form) and stored for a possible recovery in case of hardware or software failure.
- Updates (Operating Systems and Databases): Systems are regularly updated to avoid the malfunctioning (often software) that occurs under certain circumstances.
- Scalability: This issue can be easily solved by requiring additional power (computing, storage, networking) to its provider. The service provider will then be responsible for providing the appropriate resources.

Type of structure adopted for the operating platform can be classified as:

- Infrastructure as a Service (IaaS): the services provider supplies the customer with basic infrastructure for the various segments (computing, storage, network) represented by individual components such as CPU RAM and network adapters. This solution is very flexible because it gives the customer the possibility to customize the entire infrastructure according to their needs. With this methodology, for example, the customer can install how many virtual servers they want, with the Operating Systems with which they intend to create and manage resources.
- Platform as a service (PaaS): the service provider defines the entire infrastructure by choosing, in agreement with the client, the type of resources and the associated systems. The customer needs only to define its applications, in order to optimize the costs according to the initial needs.
- Software as a Service (SaaS): the service provider customizes also the standard applications (such as mail, CRM and web services).

4. DEFINITION OF THE CASE STUDY

In this section, the considered case study will be detailed. First of all, the so-called *road queue event* will be defined. Furthermore, in order to consider a case study as representative as possible, some assumptions are made:

- Geo-localization features are considered available on vehicle
- A set of sensors must be considered to collect data from vehicles.
- Pattern search will be used
- The length of the road under exam (at least three geospatial coordinates)
- At least five vehicles are involved, in a specific time interval, within the section under examination.

In addition to these "static" assumptions, “dynamic” parameters of each vehicle are taken into account, such as:

- *Accelerator_pedal_position*: indicates the position of the pedal at the instant of detection. If the driver is not exerting any pressure, then the value of this parameter is set to 0 (zero);
- *engine_speed*: indicates the instantaneous speed of engine rotation. If the engine is switched off (see *ignition_status* parameter)

then the value of this parameter is set to 0 (zero);

- *ignition_status*: indicates the status of the vehicle (*"run"* or *"off"*). The "run" value indicates that the vehicle has been started, while the "off" value indicates that the vehicle has been stopped;

- *Latitude*: indicates the latitude of the vehicle at the time of detection;

- *Longitude*: indicates the longitude of the vehicle at the time of detection. Together with the previous parameter, it provides the basis for geolocation of the vehicle on a map;

- *Vehicle_speed*: indicates the instantaneous speed at the time of detection. If the vehicle is stationary, then the value of this parameter is 0 (zero).

In addition to this set of dynamic parameters, two additional indicators are required to be stored in the database:

- *vehicle_id*: a primary key that uniquely identifies a vehicle

- *Timestamp*: that represents the instant at which a specific sensor has detected the relevant parameter (measurement).

In order to made the case study as representative as possible of reality, in addition to the above-described information, the so-called *road queue event* will be considered if:

- In the observed area, there are at least five vehicles

- The sensors of the vehicles indicate an engine speed greater than zero (the vehicles aren't switched off), accelerator pedal with a value of zero and vehicle speed of zero.

The above values are for at least 3 different latitude and longitude values, all of which inherent to the selected area.

5. POSSIBLE APPROACHES TO CASE STUDY

In this section, two possible approaches to manage the above-described case study are presented. The former proposed solution is related to the conventional Relational Solution (see fig.4) and the latter is related to the innovative use of Graph DB (see fig. 7).

5.1 Relational Solution

As shown in figure 4, the data sent from Connect Vehicle has received from Ingestion Server that provides a set of Ascii files to be load into existing Relational Tables. Obviously, only structured data regarding sensor values will be treated. In this way, after the loading process, the SQL queries retrieve the number of vehicle that lying in the queue of the road artery.

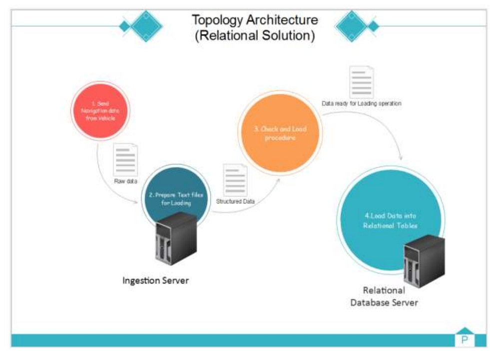

Figure 4: Architecture - Relational Solution

In order to represent the model described in figure 4, seven tables have been defined. Five of these tables represent the sensors that are in the vehicle (one table for each sensor). In addition, has been defined a table, called *demographics of the sensors*, which contains: a progressive number, the number of identification of the vehicles and their description. The latest table contains the information about the geo-referencing, this information is utilized to perform the simulated path of vehicles.

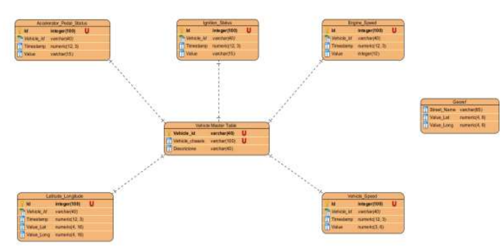

Figure 5: RDBMS Model

The master table is linked, through specific referential integrity constraints, with each table representing the individual sensor; in this way, it will be possible to navigate the data, by searching by the specific characteristics of interest.

Journal of Theoretical and Applied Information Technology
28th February 2018. Vol.96. No 4

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

Figure 5 shows whole described elements and their relationships.

As discussed in [2], RDBMS Approach has two main benefits:

- The management of multi-vehicle: this management is available already at the beginning. Indeed, in case of management of a fleet of vehicles, just need to enter in the vehicle identifier in the master table and the sensor data (even not complete) in the related tables.
- The possibility, without enhance the model complexity, to add a new sensor (or type of sensors) to face with new requirements that may arise. It is sufficient, indeed, to create a new table, with the specific fields relating to: the timestamp, the value of the data provided and the key-field that binds the latter to the master table.

However, the RDBMS approach leads three main problems:

- Non-linear increasing of response times that implies grows the complexity of navigation.
- Data Management only for structured data.
- Impossibility of the real-time case implementation.

For these reasons, in [2] has been proposed a Graph DB approach to manage this scenario.

5.2 Graph DB Solution

In this subsection, a description of the Graph DataBase solution is illustrated, as represented in figure 6. In this case the start set file is the same but some differences are regarding, for example, the loading methodology. The data are loaded into GraphDB scheme, and as a consequence, node entities and relationships that will be stored in the same main area (RAM typically) will be generated.

Following is shown the script for the loading of the dataset containing the considering road (i.e. Way), the corresponding waypoints that contain the geospatial coordinates and its relationships:

```
//
// CREATING (:Way) nodes
// CREATING (:Waypoints) nodes
//     CREATING    (:Way)-[:HAS_WAYPOINT]->(:Waypoints) relationships
//
USING PERIODIC COMMIT
LOAD   CSV   file   with   headers   from
'file:///Florence-station-ways.csv' as row
WITH row. The Way' as streetName,
     ToFloat(row. The Latitude') as lat,
     ToFloat(row. The Longitude') as lon
       '('  +  lon  +  ','  +  lat  +  ')'  as
waypointDesc
CREATE (way:Way { name: streetName })
CREATE  (Waypoints:Waypoints  {  lon:  lon,
lat: lat, desc: waypointDesc })
CREATE (way)-[:HAS_WAYPOINT]->(waypoint);
```

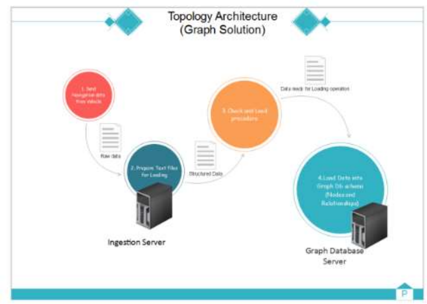

Figure 6: Architecture – Graph Database Solution

The second step of the proposed approach consists of creating the following additional three types of nodes:

- *Vehicle*: each vehicle is univocally represented by a specific node.
- *Trip*: Every time a vehicle is "Started" (by setting the parameter *ignition_status* as 'run'), it generates a *Trip* type node, this node represents the started vehicle travel. The timestamp associated with the sending of the data of the sensor *ignition_status* is stored in the '*start*' attribute of the node. When the sensor sends a "*stopped*" signal (*ignition_status* = ”*off*”) to the vehicle, will be added an attribute of type '*end*', filled with the respective timestamp, to represent the fact that the vehicle travel is finished.
- *TripData*: this node contains, for each Trip, all data received from the following sensors: *accelerator_pedal_position*, *engine_speed*, *latitude*, *longitude*, and *vehicle_speed*. The transmissions sequence is managed as a linked list.

In addition, has been introduced the following two types of relations (see fig. 7):

- *HAS_COVERED*: is a relation between a specific vehicle and each of its related trips;
- *NEXT*: is a relation able to constitute the concatenated list between the individual data transmissions (TripData), relating to the specific travel (Trip).

In order to determine the position of the vehicle in the road surface, it is necessary to associate to each TripData node, characterized with the attributes latitude and longitude, the closer Waypoint node (filled with the same attributes). Actually, the geospatial coordinates sent by the vehicle does not coincide exactly with the static

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

geospatial coordinates that represent the streets, therefore, it is necessary to exploit geospatial algorithms and in particular the function "closest", which provides precisely the closest node using appropriate Cypher command.
The following statement shows the geospatial index creation:

```
 //
// CREATING GEOSPATIAL INDEX 'roads' FOR
(:Waypoint) NODES
//
CALL      spatial.addPointLayerXY('roads',
'lon', 'lat');
//
// ADDING (:Waypoint) NODES TO GEOSPATIAL
INDEX 'roads'
//
MATCH (wp:Waypoint)
WITH collect(wp) as wps
CALL spatial.addNodes('roads', wps);
//
// CREATING (:TripData)-[:IS_LOCATED_IN]-
>(:Waypoint)
//
MATCH (tripData:TripData)
CALL spatial.closest('roads',
      {    lon:    tripData.lon,    lat:
tripData.lat }, 0.002) yield node
WITH  tripData,  head(collect(node))  as
closest
CREATE       (tripData)-[:IS_LOCATED_IN]-
>(closest);
```

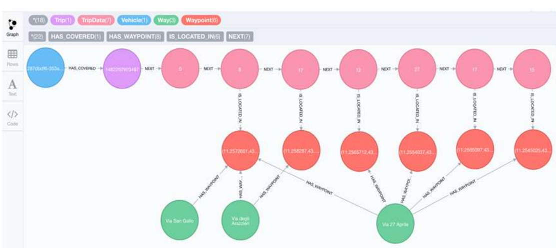

Figure 7: Coupling between the route taken by the vehicle and the geospatial map representing the road

Given the above-described environment, it is possible query the graphical database Neo4j® with Cypher language. Following query display the road queue event (subdivided into sections (1-6) in order to facilitate reading):

```
// [CQL.1]
WITH {
       minEngineSpeed: 500, now:
1482253953497,
       minQueueMinutes: 8,
minQueueWaypoints: 3, minQueueCars: 5
     } as params
// [CQL.2]
MATCH (w:Way)-[:HAS_WAYPOINT]-
>(wp:Waypoint)
        <-[:IS_LOCATED_IN]-(td:TripData)<-
[:NEXT*]-(t:Trip)
          <-[:HAS_COVERED]-(v:Vehicle)
// [CQL.3]
WHERE td.speed = 0
AND td.acceleratorPedalPosition = 0
AND td.engineSpeed >=
params.minEngineSpeed
AND td.timestamp <= params.now
AND coalesce(t.end, params.now) >=
params.now
// [CQL.4]
WITH params, w, wp, v,
     (max(td.timestamp) -
min(td.timestamp)) / 60000 AS queueMinutes
WITH params, w, count(distinct wp) AS
queueWaypoints,
     count(distinct v) AS queueCars,
     avg(queueMinutes) AS
averageQueueMinutes
// [CQL.5]
WHERE averageQueueMinutes >=
params.minQueueMinutes
AND queueWaypoints >=
params.minQueueWaypoints
AND queueCars >= params.minQueueCars
// [CQL.6]
RETURN w AS way, queueWaypoints,
queueCars, averageQueueMinutes
ORDER BY w.name
```

The result produced is shown in the following figure:

way	queueWaypoints	queueCars	averageQueueMinutes
name Via delle Porte Nuove	3	5	8.6

Started streaming 1 record after 89 ms and completed after 90 ms.

Figure 8: Query results

In “Via delle Porte Nuove” street there are 5 vehicles for which the average stay time (queue_time) is 8.6 minutes. Note that the execution time for this complex cypher query was initially 89 ms.
In order to improve the performance of the query, in terms of execution time, the use of particular structures (indexes) is considered in such a way that the cypher query optimizer takes advantage of the indexes. As shown in the figure 9, simple indixes (not composite) are created for each type of node involved in the main cypher query.

INDEX ON :Trip(start)	ONLINE	node_label_property
INDEX ON :TripData(timestamp)	ONLINE	node_label_property
INDEX ON :Vehicle(id)	ONLINE	node_unique_property
INDEX ON :Way(name)	ONLINE	node_unique_property
INDEX ON :Waypoint(desc)	ONLINE	node_unique_property

Figure 9: List of indexes for each node

The results lead to a drastic reduction in terms of execution times, from the initial 89 ms up to 25 ms (see fig. 10).

Figure 10: Query's execution time

These performances are achieved since all the elementary structures (such nodes and relationships) involved are always resident in RAM. The details of how these results were achieved should be specified in more detail.
Through the EXPLAIN function, Graph Db extracts the information, by means of a special algorithm, on how it performs the search for the extraction of the final data. Figure 11 shows the entire search path. The EXPLAIN function allows to obtain a "logical" view on the data path. Another function, closer to the data, is the PROFILE function that provides information about the "physical" access to the data, e.g. how aggregations are performed, how many records involved, the sort operations of the data obtained (see fig. 12).

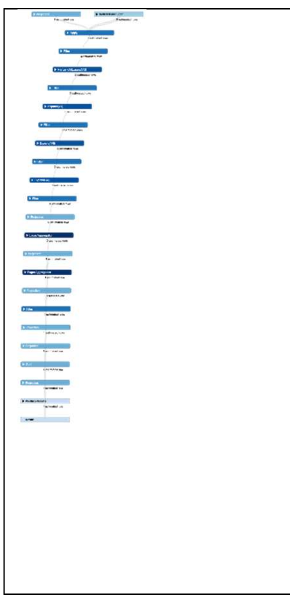

Figure 11: EXPLAIN function show the choice of path

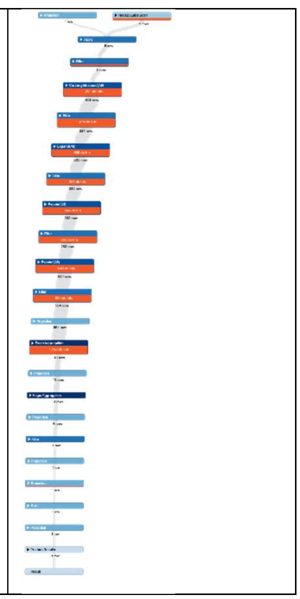

Figure 12: PROFILE function indicates the amount of involved data

By using just these two indicators, it is possible to evaluate the reactivity (in terms of response time) of the database when requested are submitted. The behavior, given the memorization of both nodes and relationships, is completely different from the relational approach. This non-negligible advantage becomes more essential when it comes to providing near real-time answers that involve huge amounts of data.

6. GRAPHDB SOLUTION VS RDBMS SOLUTION

The relational nature of GraphDB makes possible the execution of complex queries based on the nodes graph.
As example, exploiting these queries is possible to determine, more or less in real time the shortest path between two locations; in this code fragment however, is not consider the possibility that one or more ways are busy.
Following, the statement Cypher able to calculate the shortest path:

```
MATCH    p    = shortestPath((start:Way)-
[:HAS_WAYPOINT|:NEXT*]-(end:Way)
WHERE start.name = 'Via degli Arazzieri'
AND end.name = 'Via Pietro Toselli'
RETURN p
```

The results of this statement is: "*to arrive in Via Pietro Toselli, starting from Via degli Arazzieri, the shortest path passes by 27 April, Piazza dell'Indipendenza, Via Giuseppe Dolfi, Viale Filippo Strozzi, Viale Belfiore and finally to Via Delle Porte Nuove (congested)*".
This is possible exploiting the shortest path algorithm to determine an alternate path, in case the original path is congested by the traffic. In order to avoid the queue will be sufficient to require the system to calculate the shortest route, excluding however *Via Delle Porte Nuove* street. The previous query should therefore be amended by adding this restriction, as follows:

```
MATCH p = shortestPath((start:Way)-
[:HAS_WAYPOINT|:NEXT*]-(end:Way)
WHERE start.name = 'Via degli Arazzieri'
AND end.name = 'Via Pietro Toselli'
AND NOT 'Via Delle Porte Nuove' in
Extract( way in nodes(p) | way.name )
RETURN p
```

The result of the query is represented by the alternative route shown in figure 13, useful to avoid the queue.

Figure 13: Alternate shortest path performed by system to avoid "Via Delle porte Nuove" queue.

The algorithm has been executed with benchmarks, the experiment has confirmed the capacity to take

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

over 50 million crossings per second. It is clear that this type of structure can be easily used when we consider a large number of objects and response times near real-time.

It is clear that the advantages of this approach are:

- *The GraphDB representation*: is easily understandable and can be automated with dedicated tools.
- *The data modeling*: data modeling process is more accurate respect to RDBMS. GraphDB representation allows users to describe domain with a complex graph model able to represent the whole domain, its concepts, and their relations.
- *The GraphDB query language*: the query language provides many native commands able to highlight the shortest path or, as shown in the case study, an alternative path easily; this gives a good level of abstraction and is independent of the knowledge of the data structures, differently from the relational context. Statistically, an optimized query for graph database contains fewer errors respect to the equivalent realized in SQL language.
- *The implementation*: the GraphDB technology allows storing both the node that the reports which it has with other neighboring nodes (mechanism of index-free adjacency), for this reason it no need of complex join operations to be carried out at the time of execution. Due this feature the above-mentioned solution is better respect to RDBMS solution in terms of scalability and efficiency.
- *Performance*: the GraphDB query performance is near to real-time. Using GraphDB, the complex join (i.e. the so-called join pain that is a typical issue of RDBMS) are converted into the crossings of the graph, thus maintaining the performance in the range of milliseconds

7. CONCLUSIONS

In this paper, the validity of the proposed Graph DB solution for this specific category of problems (i.e. those involving paths of vehicles totally connected) respect to RDBMS solution has been demonstrated.

In particular, this article has presented a brief overview of the Autonomous and Connected Vehicles and validate the advantages that may be obtained using the proposed GraphDB approach to manage the great amount of data generated by the sensors disseminated inside the vehicle.

To prove the validity of the proposed approach, respect the conventional one, a simple case study has been illustrated, in which the information of the sensors has been attached to the geolocation of the vehicle, making possible to discover in time *near* real-time the queue event. This is of great help to potential stakeholders such as traffic control structures and assurance agencies.

In order to demonstrate the advantages that may be obtained using the GraphDB it has been executed the proposed algorithm with specific benchmarks, and the experiment has confirmed a capacity to perform over 50 million crossings per second.

Concluding, the proposed methodology provides the following advantages: the easily representation of the model; the accuracy of the data modeling; the usability of the GraphDB query language; the innovation of implementation, in terms of mechanism of index-free adjacency, and the temporal complexity performance, *near* to real-time.

REFERENCES:

[1] J. LAKA and M. GONZALEZ RODRIGUEZ, “INDUSTRIA 4.0,” *DYNA Ing. E Ind.*, vol. 90, no. 3, pp. 16–17, 2015.

[2] A. Pieroni, N. Scarpato, and M. Brilli, “Industry 4.0 Revolution in Autonomous and Connected Vehicle A non-conventional approach to manage Big Data,” *Journal of Theoretical and Applied Information Technology*, vol. 96, no. 1, 2018, E-ISSN 1817-3195 / ISSN 1992-8645.

[3] “Baidu opens up driverless car tech to challenge Google, Tesla.” [Online]. Available: https://www.cnbc.com/2017/04/19/baidu-driverless-cars-open-source-technology.html. [Accessed: 04-Oct-2017].

[4] F. Guadagni *et al.*, “Personal and Sensitive Data in the e-Health-IoT Universe,” in *Internet of Things. IoT Infrastructures: Second International Summit, IoT 360{\textdegree} 2015, Rome, Italy, October 27-29, 2015, Revised Selected Papers, Part II*, B. Mandler, J. Marquez-Barja, M. E. Mitre Campista, D. Cagá\vnová, H. Chaouchi, S. Zeadally, M. Badra, S. Giordano, M. Fazio, A. Somov, and R.-L. Vieriu, Eds. Cham: Springer International Publishing, 2016, pp. 504–514.

[5] K. C. Dey, A. Rayamajhi, M. Chowdhury,

P. Bhavsar, and J. Martin, "Vehicle-to-vehicle (V2V) and vehicle-to-infrastructure (V2I) communication in a heterogeneous wireless network - Performance evaluation," *Transp. Res. Part C Emerg. Technol.*, vol. 68, pp. 168–184, 2016.

[6] "Million Lines of Code — Information is Beautiful." [Online]. Available: http://www.informationisbeautiful.net/visualizations/million-lines-of-code/. [Accessed: 04-Oct-2017].

[7] G. C. Cardarilli, L. Di Nunzio, R. Fazzolari, and M. Re, "Algorithm acceleration on LEON-2 processor using a reconfigurable bit manipulation unit," in *2010 8th Workshop on Intelligent Solutions in Embedded Systems*, 2010, pp. 6–11.

[8] G. C. Cardarilli, L. Di Nunzio, R. Fazzolari, and M. Re, "TDES cryptography algorithm acceleration using a reconfigurable functional unit," in *2014 21st IEEE International Conference on Electronics, Circuits and Systems (ICECS)*, 2014, pp. 419–422.

[9] G. C. Cardarilli, L. Di Nunzio, R. Fazzolari, and M. Re, "Fine-grain Reconfigurable Functional Unit for embedded processors," in *2011 Conference Record of the Forty Fifth Asilomar Conference on Signals, Systems and Computers (ASILOMAR)*, 2011, pp. 488–492.

[10] G. C. Cardarilli, L. Di Nunzio, R. Fazzolari, M. Re, and R. B. Lee, "Butterfly and Inverse Butterfly nets integration on Altera NIOS-II embedded processor," in *2010 Conference Record of the Forty Fourth Asilomar Conference on Signals, Systems and Computers*, 2010, pp. 1279–1283.

[11] A. D'Ambrogio, D. Gianni, G. Iazeolla, and A. Pieroni, "Distributed simulation of complex systems by use of an HLA-transparent simulation language," in *2008 Asia Simulation Conference - 7th International Conference on System Simulation and Scientific Computing*, 2008, pp. 460–467.

[12] A. D'Ambrogio, G. Iazeolla, A. Pieroni, D. Gianni, "A model transformation approach for the development of HLA-based distributed simulation systems," in *SIMULTECH 2011 - Proceedings of 1st International Conference on Simulation and Modeling Methodologies, Technologies and Applications*, 2011, pp. 155–160.

[13] G. Iazeolla, A. Pieroni, A. D'Ambrogio, and D. Gianni, "A Distributed Approach to Wireless System Simulation," in *2010 Sixth Advanced International Conference on Telecommunications*, 2010, pp. 252–262.

[14] G. Iazeolla and A. Pieroni, "Energy saving in data processing and communication systems.," *ScientificWorldJournal.*, vol. 2014, p. 452863, Jun. 2014.

[15] A. De Mauro, M. Greco, and M. Grimaldi, "A formal definition of Big Data based on its essential features," *Libr. Rev.*, vol. 65, no. 3, pp. 122–135, 2016.

[16] M. T. Pazienza, N. Scarpato, A. Stellato, and A. Turbati, "Din din! The (Semantic) Turkey is served!," in *CEUR Workshop Proceedings*, 2008, vol. 426.

[17] M. T. Pazienza, N. Scarpato, A. Stellato, and A. Turbati, "Semantic Turkey : A Browser-Integrated Environment for Knowledge Acquisition and Management," *Semant. Web*, vol. 0, pp. 1–14, 2011.

[18] M. Bianchi, M. Draoli, G. Gambosi, M. T. Pazienza, N. Scarpato, and A. Stellato, "ICT tools for the discovery of semantic relations in legal documents," in *CEUR Workshop Proceedings*, 2009, vol. 582.

[19] M. T. Pazienza, N. Scarpato, and A. Stellato, "STIA*: Experience of semantic annotation in Jurisprudence domain," in *Frontiers in Artificial Intelligence and Applications*, 2009, vol. 205, pp. 156–161.

[20] F. Fallucchi, M. T. Pazienza, N. Scarpato, A. Stellato, L. Fusco, and V. Guidetti, "Semantic bookmarking and search in the earth observation domain," in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 2008, vol. 5179 LNAI, no. PART 3, pp. 260–268.

[21] A. R. D. Accardi and S. Chiarenza, "Musei digitali dell ' architettura immaginata : un approccio integrato per la definizione di percorsi di conoscenza del patrimonio culturale Digital museums of the imagined architecture : an integrated approach," *DISEGNARECON*, vol. 9, 2016.

[22] P. Ferroni *et al.*, "Risk Assessment for Venous Thromboembolism in Chemotherapy-Treated Ambulatory Cancer Patients," *Med. Decis. Mak.*, vol. 37, no. 2, pp. 234–242, 2016.

[23] P. Ferroni, F. M. Zanzotto, N. Scarpato, S.

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

Riondino, F. Guadagni, and M. Roselli, “Validation of a machine learning approach for venous thromboembolism risk prediction in oncology,” *Dis. Markers*, vol. 2017, pp. 1–7, 2017.

[24] F. Guadagni *et al.*, “RISK: A random optimization interactive system based on kernel learning for predicting breast cancer disease progression,” in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 2017, vol. 10208 LNCS, pp. 189–196.

[25] M. Gerla, E. K. Lee, G. Pau, and U. Lee, “Internet of vehicles: From intelligent grid to autonomous cars and vehicular clouds,” in *2014 IEEE World Forum on Internet of Things, WF-IoT 2014*, 2014, pp. 241–246.

[26] G.C. Cardarilli, L. Di Nunzio, R. Fazzolari, M. Re, R.B. Lee, "Integration of butterfly and inverse butterfly nets in embedded processors: Effects on power saving", *in (2012) Conference Record - Asilomar Conference on Signals, Systems and Computers, art. no. 6489268, pp. 1457-1459.*

[27] Gianni, D., D'Ambrogio, A., Iazeolla, G., Pieroni, A., "Producing simulation sequences by use of a java-based generalized framework", *in 2008 Proceedings - EMS 2008, European Modelling Symposium, 2nd UKSim European Symposium on Computer Modelling and Simulation, art. no. 4625266, pp. 171-176.*