Highly Mobile Query Processing

Thierry Delot[1)], Sergio Ilarri[2)], Marie Thilliez[1)], Genoveva Vargas-Solar[3)], Mickael Bellengier[1)]

Abstract:

A number of wireless and small-sized devices have started to be embedded in modern cars in the form of on-board computers, GPS navigators, or even multimedia centers. Thus, the vehicles can carry useful information, acting as data sources for other vehicles. Recently, some works have addressed the problem of processing queries in such highly dynamic vehicular networks in order to share information between drivers. The proposed query processing techniques usually rely on a push model. Hence, each vehicle receives data from its neighbors and decides whether they are relevant enough to be stored in a local data cache. Then, the data may be used by a query processor to retrieve relevant data for the driver.

In this paper, we look at the problem from a broader perspective and discuss the interest of multi-scale query processing techniques in such context. The goal is to exploit, at the mobile device's level, different access modes (e.g., push, pull) and various data sources (e.g., data cached locally, data stored by vehicles nearby, remote Web services, etc.) to provide the users with results for their queries. We highlight the most important challenges and outline some possible approaches. We also present a prototype of a first query evaluator developed using the Microsoft LINQ API.

1 Introduction

Mobile devices and wireless networks are now widespread, and continuous technological advances are observed in this area. This motivates an intensive research effort to try to develop suitable data management strategies for mobile computing. Compared to the early works on mobile computing, today new elements play a key role [10]. Among them, we can highlight the availability of multiple data sources, multiple forms of network access, and the emergence of mobile P2P schemes. All these aspects have an important impact on the development of appropriate query processing techniques.

[1)]University Lille North of France, Lamih Fre UHVC/CNRS 3304, Valenciennes, France, {Thierry.Delot, Marie.Thilliez, Mickael.Bellengier}@univ-valenciennes.fr

[2)]University of Zaragoza, IIS Department, Zaragoza, Spain, silarri@unizar.es

[3)]CNRS, LIG-LAFMIA, Grenoble, France, Genoveva.Vargas-Solar@imag.fr

Particularly important is the fact that, when a query is submitted, it is highly probable that *multiple data sources* will need to be accessed in order to compose the final answer, including: local data stored on the mobile device, data stored on other devices nearby, remote databases, Web services, or even sensors or the Internet. Every data source will have different features in terms of access latency, economic cost, reliability, data quality or accuracy, etc. In some cases, it may also happen that some data could be available in more than one source. This heterogeneity and the distributivity of the resources bring several challenges: 1) the relevant data sources must be selected by considering the requirements of the user query, the availability of data, and the special features of each data source; 2) the query has to be decomposed into several local queries that are sent to the different data sources for processing; and 3) the data retrieved must be correlated to assemble the final answer.

We have also to consider that there are *multiple network connectivity options* (3G, WiFi, Bluetooth, UWB, etc.), with different features, and so it is a challenge to select the best communication option at a certain moment according to predefined user preferences. Finally, it is not only possible to access data through a fixed network infrastructure (like in the classical mobile computing environment) but also through direct interactions in an *ad hoc mobile P2P network*.

Therefore, new data management and query processing techniques are needed. The techniques proposed should take into account the elements mentioned above, among others, in order to decide appropriate query execution plans and retrieve the results. We will focus on the specific case of *vehicular networks* [8], where vehicles can exchange data among themselves to inform drivers about interesting events (e.g., available parking spaces, accidents, traffic congestions, etc.) [3, 11]. So, the purpose of this paper is to outline the challenges that arise for query processing in vehicular networks, with an emphasis on *multi-scale query processing* (any query processing that may need to access data sources of different types to obtain the answer to a query, generalizing the definition provided in [2]).

2 Models for Accessing Data

In this section, we present different models that can be exploited by mobile devices in vehicular networks to access data.

2.1 Access to Remote Services

Today, wireless networks allow mobile users to connect to various remote databases, Internet sites, Web services, or services available on remote computers. These last years, some research works have also considered query processing techniques that use distributed data stored on remote devices, some of which focus on the use of *mobile Web services* (i.e., Web services

available on mobile devices) [1]. One of the problems to use remote services in the evaluation of queries is the difficulty to have a uniform representation of the available services. Thus, although in a Web service context it is possible to use a *UDDI* registry (*Universal Description, Discovery and Integration*, `http://uddi.xml.org/`) and *WSDL* (*Web Services Description Language*, `http://www.w3.org/TR/wsdl`), in an heterogeneous context it is very difficult to have a standard representation of every available service. Another important problem is how to determine the services that are relevant for a given query.

An example of a remote service for drivers is *Waze* (`http://world.waze.com/`), which is a social mobile application which allows Waze users to publish and consume real-time maps and traffic information. Several maps can be provided to mobile users and traffic information can be retrieved dynamically by using a mobile telephone network.

2.2 Data Dissemination

Relying on a *push model* is a common approach for query processing in vehicular networks. With such an approach, each vehicle receives data from its neighbors and decides whether they are relevant enough to be stored in a local data cache or not [3]. Then, the data may be used locally by a query processor in charge of retrieving data relevant for the driver (e.g., the driver may submit a query to retrieve the five nearest parking spaces) [11]. Thus, a query can only find an answer to a query *opportunistically*, that is, when a vehicle with the required information has passed before nearby. The major difficulty is how to disseminate data in the vehicular network so that the vehicles receive the relevant information efficiently (timely and without unneeded overheads such as duplicate packets or irrelevant data).

Nevertheless, with a push model, we cannot imagine that every data item will be communicated to every vehicle, as this would consume too much bandwidth and lead to high communication and processing efforts on the vehicles. On the contrary, only data about events that are potentially interesting for a large set of vehicles (e.g., an emergency braking or a traffic congestion) are diffused among the vehicles that the system estimates as potentially interested. Information about other events will not be disseminated, and so it is not possible to share some particular information among a small set of interested vehicles, for example to build vehicular social networks. Moreover, the dissemination of a certain event is usually restricted to a spatio-temporal area where the event is estimated to be relevant. So, for example, a driver on a highway approaching a city would not be able to query which of the entries to the city she/he should take to avoid a traffic congestion.

Several interesting works have proposed a push model (e.g., [3, 7, 11]). In works such as [7, 11], an *opportunistic exchange* mechanism is proposed to exchange data among neighboring vehicles when they encounter each other. In [3], the concept of *Encounter Probability* is proposed to

efficiently share information among vehicles by using vehicle-to-vehicle communications.

2.3 Query Dissemination

Several works have considered a *pull model*, where a query is actually communicated to other vehicles in the vehicular network. This provides more flexibility in terms of the types of queries that can be considered, since a query could be diffused far away to retrieve remote data.

The basic idea, inspired by traditional Peer-to-Peer (P2P) systems, consists of diffusing the queries to different data sources either directly or using multi-hop relaying techniques. Then, each node can compute a partial query result based on its local data and deliver it to the destination node. However, since no fixed data server or any kind of infrastructure is necessarily available in vehicular ad hoc networks, new techniques to access data are needed. Indeed, the mobility of nodes makes the management of an indexing structure, used in traditional P2P systems to decide how to route queries, impossible (as indicated in works such as [4]). Therefore, some mechanism must be used to route the queries to the vehicles that could store relevant information, usually based on spatio-temporal criteria (more appropriate in this context than using flooding or a random or biased walk [4]).

These works must also face the problem of routing the query results back to the query originator. This is a challenge because the vehicle that issued the query can move in the meanwhile, and so routing the results based on simple geographic criteria may not be enough (it is difficult to know where the originator is currently located). Furthermore, it is not even possible to ensure that there is at that moment a communication path to the originator. A possibility to facilitate the routing process could be the use of mobile agent technology [9].

Some examples of works that have proposed a pull approach are [5, 6, 13, 12]. *FleaNet* [5] is a virtual market organized over a vehicular network, based on a *mobility assisted query dissemination* where the node that submitted the query periodically advertises it only to its one-hop neighbors. *PeopleNet* [6] is an infrastructure-based proposal for information exchange in a mobile environment. *Roadcast* [13] is a content sharing scheme for VANETs, such that a vehicle can only query other vehicles that it encounters on the way, and the scheme proposed tries to return the most popular content relevant to the query. Finally, in [12] a combination of pull and push is considered for *in-network query processing in mobile P2P databases*. In these works the queries and results are communicated only to the neighboring vehicles, and therefore the query/results routing problem does not appear (the query originator is always at one-hop distance). Besides, these proposals are not able to process some kinds of queries, since the query originator must move near the nodes which store the information needed.

3 Towards Multi-Scale Mobile Query Processing

In this section we show that, instead of focusing on a particular access model, it can be very interesting to consider a *multi-scale query processing*, which implies exploiting the available data sources whatever the access mode (e.g., push or pull). Thus, the possible use of various data sources can increase the number of potential results that can be retrieved and proposed to the user. It also increases the probability to provide at least one useful result. Finally, combining several types of data sources allows to compute results that could not be computed otherwise (or only with a lower quality). To illustrate our purpose, we propose in the following a couple of examples of mobile queries that may benefit from a multi-scale query evaluation.

Example 1: *"Retrieve all parking spaces in a radius of 2 Km"*. In this example, information about parking spaces can be extracted from a local data cache containing events received, through the use of short-range wireless networks, from neighboring vehicles leaving a parking space. Useful information about the location of parking lots can also be provided by different (Web) services accessed using mobile telephone networks (e.g., 3G). We can notice that both parts of the query (i.e., the local one and the remote one/s) can here be processed in parallel.

Example 2: *"Retrieve the list of petrol stations located in a radius of 10 Km around me and where fuel prices are less than 2$ (updating the result every 5 minutes)"*. Different steps can be distinguished for this continuous query. One possible query processing scheme implies retrieving first the "petrol stations located in a radius of 10 Km". For this, a local database containing the locations of various points of interest (e.g., train stations, airports, restaurants, petrol stations, etc.) can be queried. Once the list of petrol stations within the specified radius has been retrieved, it is possible to compute the join between this data set and one containing the list of petrol stations where the fuel prices are lower than 2$ (obtained elsewhere).

Thus, to evaluate multi-scale queries, it is necessary to decompose the user query into a set of sub-queries that have to be evaluated on different data sources (e.g., a local data cache, remote Web services, positioning services, etc.). Therefore, different alternative query execution plans may be generated for a given query. These plans contain the different evaluation steps that have to be performed, according to the data sources selected, to retrieve the query result.

For example, let us illustrate the decomposition of the query *"Retrieve the list of hotels with available rooms that I can reach in less than 30 minutes"*. To evaluate this query, the first step can be to determine the geographic location of the mobile client. To obtain this location, let us imagine that two services are available on the client's terminal: a GPS-based service (called *local service 1*) which provides the location in terms of geographic coordinates, and a WiFi-based service (called *local service 2*) which provides a *semantic/symbolic location* (street, city, etc.). Then, the second step is to retrieve the hotels that the user can reach in less than

30 minutes. For this sub-query, we can use a list of *POIs* (*Points of Interest*) stored locally on the client's device (called *local store 1*) to retrieve the subset of hotels based on the location of the user and her/his current speed. When the list of hotels has been retrieved, we can use a remote Web service (called *remote service 1*) that allows to obtain the subset of those hotels with available rooms (e.g., by transparently contacting appropriate services at the hotels). To retrieve the hotels that the user can reach in less than 30 minutes, if these data are not available locally, the query engine could also send a suitable sub-query with the location of the client to another Web service (called *remote service 2*), but this choice is only possible if the mobile client is willing to send her/his location to a remote Web service. So, to evaluate the query, different execution plans can be considered, involving different data sources. For example, a possible execution plan could use the local service 1, the local store 1, and the remote service 1. As another example, a different execution plan could consider the local service 2 and the remote service 2.

In this context, it is important to represent the properties of each data source or each service. It is also necessary to have precise information about the parameters needed to interact with each service or data source: the syntax of the query language supported, the format of the results provided, etc. Not only the query decomposition but also the query optimization is a challenge. Thus, besides the response time, several other aspects have a strong impact (e.g., the financial cost, the data quality of the sources, or the energy consumption). The user will probable need to express preferences about how the different costs considered have to be balanced, the context will need to be considered during the optimization (e.g., a different query processing strategy could be selected when the user's device is running out of battery), and only a few statistics may be available for query optimization (as the query processing may involve several remote independent data sources).

4 Experimental Evaluation

To validate our approach, we have developed a first prototype of a multi-scale mobile query processor for *VESPA* [3], which is a system designed for vehicles to share information (e.g., available parking spaces, accidents, emergency brakings, etc.) in inter-vehicle ad-hoc networks. In VESPA, continuous query processing techniques are applied on data received by the vehicles and stored in the local data caches. Moreover, to provide users the maximum number of relevant data items, we have extended our query processor with multi-scale access features, as described in this paper.

For the implementation, we have used *Microsoft .NET* and the *Microsoft LINQ API* . One of the benefits of LINQ is the possibility to query different types of data sources (a data structure, a Web service, a file system, or a database) which may be local or remote. One challenge during the implementation was to perform a continuous processing, which is basically not

supported by LINQ. Therefore, we decided to combine LINQ with the *Windows Presentation Foundation* (*WPF*). Indeed, thanks to the data binding capabilities provided by WPF, it is possible to implement arrival-based query processing mechanisms.

For the moment, we have developed two external LINQ providers. The first one is able to translate a LINQ query into an equivalent HTTP GET request (with the appropriate parameters) to the *JustaCote.com* Web server (`http://www.justacote.com/`), which provides information about services (e.g., petrol stations, cinemas, etc.) available in different cities in France. The second one is able to transform a LINQ query into a corresponding method call to a specific Web service using the SOAP protocol. The target Web service, called *MyDesertot*, is a service that we have developed and that enables searching for petrol stations near the current GPS coordinates of a vehicle.

As a specific application example, let us imagine a driver who wants to find an available parking space that is near a petrol station with a shop (e.g., she/he wants to gas up her/his car, buy a snack in the shop, and then park to have a rest). Solving this query involves performing a join between information about parking spaces and information about petrol stations, which can be performed by using a LINQ query with the *LINQ to Objects* model on two *IEnumerable*$<T>$ collections corresponding to the two types of data to combine. Data about available parking spaces, provided by VESPA, are available in the local data cache. Information about petrol stations can be obtained from JustaCote.com or from MyDesertot. Whereas MyDesertot stores information about whether the petrol stations have a shop or not, JustaCote.com does not have this information. So, even though there are in principle two possible execution plans, if we use JustaCote.com as a data source we can only answer a relaxed version of the query, which does not check whether the petrol station has a shop.

5 Conclusion and Perspectives

Vehicular networks open up new opportunities to develop different data services that provide drivers with interesting information. However, they also bring some challenges. In particular, new data management and query processing techniques are required. One interesting idea, which we call *multi-scale* query processing, implies to be able to combine different access modes to retrieve data (e.g., push and pull) from different data sources (e.g., data stored locally, data available in nearby vehicles, data provided by Web services, etc.). The data retrieved from the different data sources must then be combined to compose the final answer to the query. However, key research questions arise, for example from the point of view of the selection of the relevant data sources, the consideration of different alternative execution plans, and the correlation of the data retrieved. Moreover, some access modes are not so easy to carry out in the context of vehicular ad hoc networks, as it is the case for pull approaches, where query and result routing problems appear.

As future work, we will study in more detail some of the existing challenges, particularly regarding the definition of possible execution plans in a dynamic way and the different query optimization issues that must be considered.

Acknowledgments

This research work has been supported by the French research agency (ANR) in the scope of the OPTIMACS project, and by the CICYT project TIN2007-68091-C02-02.

References

[1] M. Adacal and A. Bener. Mobile web services: A new agent-based framework. *IEEE Internet Computing*, 10(3):58–65, 2006.

[2] Victor Cuevas-Vicenttin. Towards multi-scale query processing. In *IEEE 24th Int. Conf. on Data Engineering (ICDE'08) Workshops*, pages 137–144. IEEE, 2008.

[3] T. Delot, N Cenerario, and S Ilarri. Vehicular event sharing with a mobile peer-to-peer architecture. *Transportation Research - Part C (Emerging Technologies), Elsevier*, 18(4):584–598, 2010.

[4] Huilong Huang, John H. Hartman, and Terril N. Hurst. Efficient and robust query processing for mobile wireless sensor networks. *Int. Journal of Sensor Networks*, 2(1/2):99–107, 2007.

[5] Uichin Lee, Jiyeon Lee, Joon-Sang Park, and Mario Gerla. FleaNet: A virtual market place on vehicular networks. *IEEE Transactions on Vehicular Technology*, 59(1):344–355, 2010.

[6] Mehul Motani, Vikram Srinivasan, and Pavan S. Nuggehalli. PeopleNet: engineering a wireless virtual social network. In *11th Annual Int. Conf. on Mobile Computing and Networking (MobiCom'05)*, pages 243–257. ACM, 2005.

[7] S. Nittel, M. Duckham, and L. Kulik. Information dissemination in mobile ad-hoc geosensor networks. In *Int. Conf. on Geographic Information Science (GIScience'04)*, volume 3234 of *Lecture Notes in Computer Science*, pages 206–222. Springer, 2004.

[8] Stephan Olariu and Michele C. Weigle, editors. *Vehicular Networks: From Theory to Practice*. Chapman & Hall/CRC, 2009.

[9] O. Urra, S. Ilarri, T. Delot, and E. Mena. Mobile agents in vehicular networks: Taking a first ride. In *Eight Int. Conf. on Practical Applications of Agents and Multi-Agent Systems (PAAMS 2010)*, volume 70 of *Advances in Intelligent and Soft Computing*, pages 118–124. Springer, April 2010.

[10] G. Vargas-Solar, N. Ibrahim, C. Collet, M. Adiba, J.-M. Petit, and T. Delot. *Pervasive Computing and Communications Design and Deployment: Technologies, Trends, and Applications*, chapter "Querying Issues in Pervasive Environments". IGI Global, 2010.

[11] B. Xu, A. M. Ouksel, and O. Wolfson. Opportunistic resource exchange in inter-vehicle ad-hoc networks. In *Fifth Int. Conf. on Mobile Data Management (MDM'04)*. IEEE, 2004.

[12] Bo Xu, Fatemeh Vafaee, and Ouri Wolfson. In-network query processing in mobile P2P databases. In *ACM Int. Conf. on Advances in Geographic Information Systems (GIS'09)*, pages 207–216. ACM, 2009.

[13] Yang Zhang, Jing Zhao, and Guohong Cao. Roadcast: A popularity aware content sharing scheme in VANETs. In *29th Int. Conf. on Distributed Computing Systems (ICDCS'09)*, pages 223–230. IEEE, 2009.