HERMES: Aggregative LBS via a Trajectory DB Engine

Nikos Pelekis, Elias Frentzos, Nikos Giatrakos and Yannis Theodoridis

Department of Informatics, University of Piraeus,

80 Karaoli-Dimitriou St, GR-18534 Piraeus, Greece

{npelekis, efrentzo, ngiatrakos, ytheod}@unipi.gr

ABSTRACT

We present HERMES, a prototype system based on a powerful query language for trajectory databases, which enables the support of aggregative Location-Based Services (LBS). The key observation that motivates HERMES is that the more the knowledge in hand about the trajectory of a mobile user, the better the exploitation of the advances in spatio-temporal query processing for providing intelligent LBS. HERMES is fully incorporated into a state-of-the-art Object-Relational DBMS, and its demonstration illustrates its flexibility and usefulness for delivering custom-defined LBS.

Categories and Subject Descriptors: E.0 Data, General

General Terms: Management.

Keywords: Trajectories, Moving Object Databases, Location-Based Services

1. INTRODUCTION

Moving Object Databases (MOD) [4] being at the core of spatio-temporal database research, have emerged due to the explosion of mobile devices and positioning technologies. A MOD is the basic component of any LBS-oriented application. However, although LBS are already in the air for some years, the services currently provided are rather naïve, not exploiting the current software capabilities and the recent advances in MOD research field. We argue that one of the reasons for this is due to the common practice in existing approaches, which provides services to mobile users by just taking into account the current location-time and velocity information, arriving at the MOD server as a sequence of updates [10]. Given this model and the fact that LBS applications need to handle huge volumes of data, it rationally arises that performance is a significant problem; therefore, efficient query processing and indexing techniques should be applied. Moreover, this model has limited applicability in real-world applications, since safe estimations about future positions should involve past positions as well.

For this purpose, we adopt the idea of maintaining a MOD consisting of trajectories, i.e., a sequence of 3D points $<(x_{i-1}, y_{i-1}, t_{i-1}), \ldots, (x_{i-n}, y_{i-n}, t_{i-n})>$ for each mobile user o_i. The construction of the future trajectory of a user can be achieved by using the

distribution of the traffic patterns along a road network, the user's origin (i.e., the location of the LBS request) and destination (i.e., the target location) [12]. Then, the trajectory can be deducted by using a time-dependent extension of the Dijkstra's algorithm. Thus, instead of maintaining a MOD storing only current information, we invest on maintaining a so-called Trajectory Database (TD), which allows us to support more efficient and intelligent LBS [2].
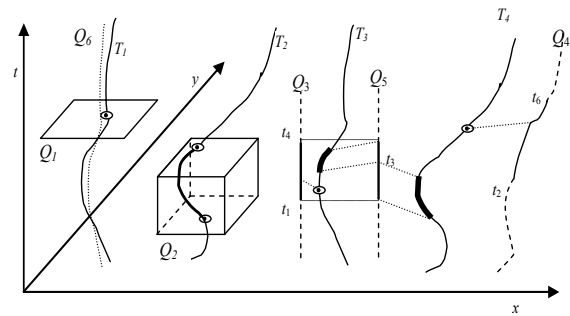

Figure 1. Querying trajectory databases

Technically speaking, maintaining a TD has several advantages. First of all, this allows us to utilize specialized access methods for trajectories [9], which not only achieve lower update costs, but, more important, provide us with a strong query processing background (i.e., support of trajectory-related queries [9], see Fig. 1) that can be the source of inspiration for advanced LBS. Second, it supports advanced querying on trajectories, including coordinate-based queries (e.g. range, nearest neighbor queries), trajectory-based queries (e.g. topological, navigational, similarity-based queries) or combinations [9], [11].

HERMES adopts the above ideas, proposes solutions, and demonstrates their implementation taking advantage of extensible ORDBMS software. Our contribution can be summarized as follows:

- We describe the architectural aspects of our server-side trajectory database engine, as well as the interface for building advanced LBS applications on top this framework.
- We investigate the power, flexibility and efficiency of the proposed query language for supporting aggregative LBS.
- We present the capability of HERMES for enabling LBS by demonstrating a desktop as well as a mobile application, which show the potential functionality of the prototype.

To the best of our knowledge, HERMES is the first work that presents a complete set of state-of-the-art query processing algorithms for TD, such as [1], [3], [6], [8], [9], which have been packaged into a state-of-the-art ORDBMS, namely Oracle 10*g*.

2. HERMES ARCHITECTURE

The main goal of HERMES is to support modeling and querying of continuously moving objects. Taking advantage of the extensibility interfaces provided by modern ORDBMS, HERMES DB engine [7], [8] is developed as a system extension that provides trajectory functionality to Oracle ORDBMS. More specifically, HERMES defines a trajectory data type and a collection of operations [1], [3], [6] as an Oracle data cartridge, which is further enhanced by a special trajectory preserving access method, namely the TB-tree [9]. Figure 2 illustrates HERMES three-tier architectural framework.
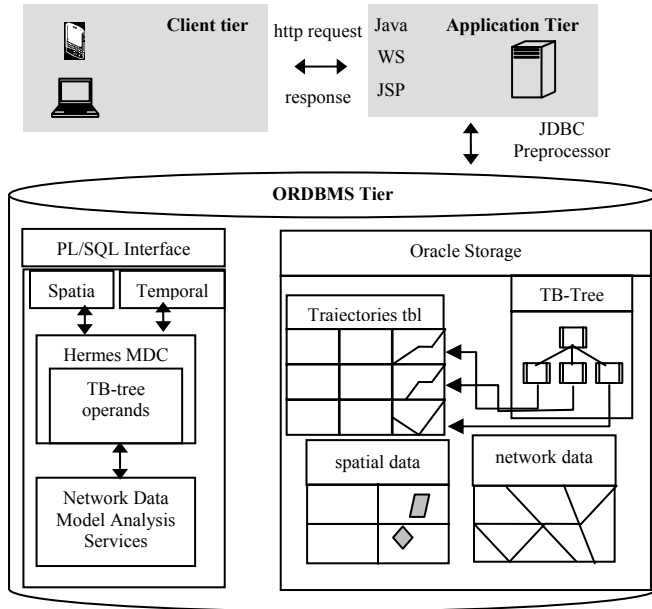

Figure 2. HERMES architecture

ORDBMS Tier: The Oracle ORDBMS Server enhanced with trajectory data storage and query capabilities serves as the infrastructure for LBS support. HERMES Moving Data Cartridge (HERMES-MDC) is the heart of this tier. In order to implement such a framework in the form of a data cartridge we exploit a set of standard data types together with (static) spatial data types offered by Oracle10g Spatial and appropriate temporal types [8]. Embedding this functionality offered by HERMES-MDC in Oracle's DML [5], one obtains an expressive and easy to use query language for moving objects.

Application Tier: To take advantage of HERMES-MDC and, thus, provide efficient LBS support, new applications should be written and registered in a middleware tier. Incorporating new LBS is straightforward since new functions implementing desired operations are embodied in JSP/Java code and published in the form of JSP pages, wsdl documents, executable jars etc. We adopt Oracle AS, which constitutes a representative example of this tier and can serve as a wap gateway, a web server and/or a wireless platform. Furthermore, AS MapViewer module enables data visualization.

Client Tier: Depending on the form that an LBS has been made available, client software involves desktop/mobile browsers, installed java application clients or registered J2ME stubs (mobile browsers and J2ME stubs refer to PDAs, smartphones etc.). These components enable the client side to post http requests and receive corresponding responses.

Recalling Fig. 1, we discuss some indicative examples that show how advanced query processing techniques may enhance existing LBS applications. Let us assume a scenario with four taxi drivers who have requested for routing services (i.e., trajectories T_1, ..., T_4, represented as solid lines in Fig.1, are maintained in the TD). HERMES users may pose various requests, which can be treated as spatio-temporal queries. Consider, for example, the fleet management operator posing a request of the form *"find all taxis located within a given area during a certain time interval"*, illustrated as Q_2 in Fig.1. Such a request is a typical range query, while a request of the type *"find all taxis' locations within a given area at a certain time instance"*, is a timeslice query (Q_1 in Fig.1). Another useful query type that can be directly used in order to enable, e.g., car-pooling services, is the nearest neighbor query, in the form *"find the nearest taxi to a reference object during a certain time interval"*, where the reference object could be a 2D static point (Q_3 in Fig.1) or a trajectory (Q_4 in Fig.1) [1].

Another useful type of trajectory queries is the result of the so-called *trajectory similarity problem*, which aims to find similar trajectories of moving objects. As an example, reference trajectory Q_6 in Fig.1, retrieves T_1 as the most similar trajectory [3]. Other similarity-based queries according to space, time and as well as derivations of motion (such as speed or direction) [6] could support a variety of LBS such as those based on a pattern of motion (e.g. *"find taxis that follow a direction similar to a given direction pattern, e.g. NE during the first half of the route and subsequently W"*) or based on sub-trajectory matching (e.g. *"find the most similar portions between two, in general, dissimilar trajectories"*).

To sum up the previous discussion, HERMES power is not only on supporting advanced LBS; more than that, it is designed to serve the need that a user may construct his/her own service and pose or even register it into the system.

3. DEMO SPECIFICATIONS

Throughout the demonstration, users will be able to test the system using either a real dataset of a fleet of trucks or a synthetic dataset simulating a fleet of taxis (for producing very large datasets), both moving in Athens metropolitan area, Greece. Taking the latter case, the scenario is the following: Each taxi is located in a randomly selected location and requests routings to an, also randomly generated destination. When it completes its planned journey, i.e., it reaches its destination, the system randomly generates a new destination point, routes the object there, and so on. During each phase, the trajectory proposed by the network analysis component is inserted into the TD; an additional component is responsible for deleting the expired parts of trajectories, i.e., those being in the far past, as time evolves. Then users can pose either typical benchmark queries over trajectories [11], following the classification of queries with respect to the type of data / reference objects (static vs. moving objects) [2], or even build their own query on either past, current or estimated (future) locations of taxis. Queries are posed in a GUI that provides essential capabilities including query predicate selection and results projection. Graphical map interaction for predicate definition is supported where appropriate.
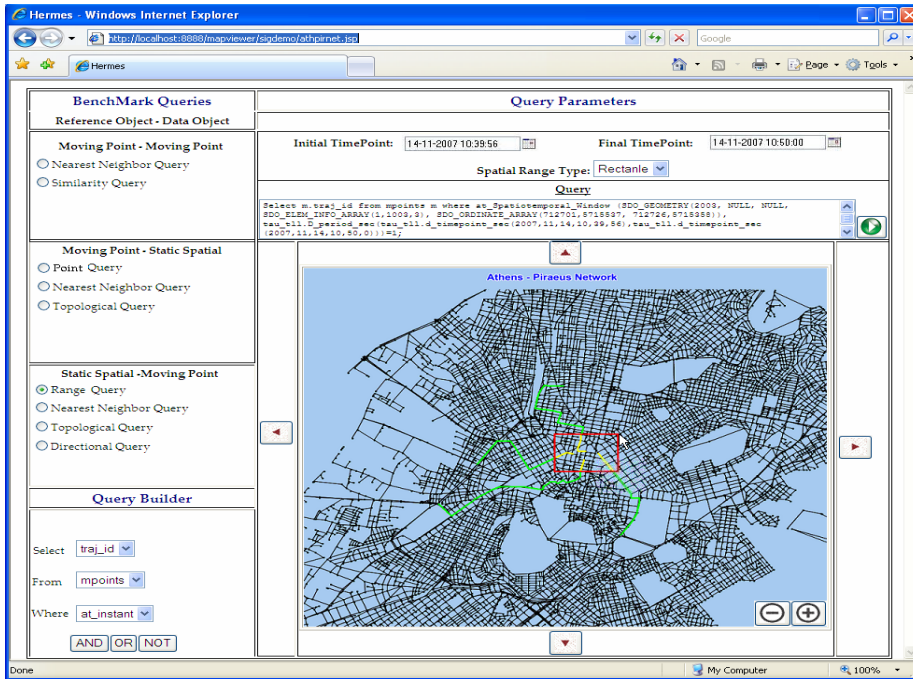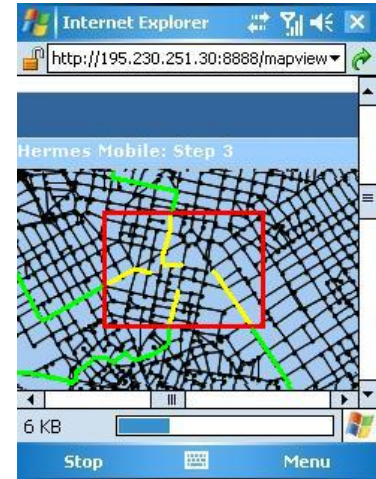
(a) HERMES desktop	(b) HERMES mobile

Figure 3. Screenshots of HERMES user interface

Fig. 3(a) and 3(b) are two representatives snapshots of HERMES (desktop and mobile, respectively) GUI, which illustrate the process of a range query, where the reference object (the data objects) is a static spatial (are moving points, respectively) – see the user selection at the left part of the interface in Fig. 3(a)). Having selected this query type, the user is able to define query parameters (a temporal period and a spatial range, in this query) using a graphical interface, if applicable).

HERMES demonstration consists of three basic stages:

1. **Query Specification:** This stage involves the selection of one among predefined queries [11] or the formation of a custom one using the query builder. The corresponding parameters are presented to be filled in by the user.

2. **Results Presentation:** After forming the query, the system processes the query and presents the results on the map.

3. **Query Refinement:** To facilitate interaction, the query can be further refined or exploited by selecting among the visualized results and building a new query.

This setting motivates the users to have an energetic apperception of the power and advantages of our approach.

ACKNOWLEDGMENTS

REFERENCES

[1] Frentzos, E., Gratsias, K., Pelekis, N., and Theodoridis, Y. Algorithms for Nearest Neighbor Search on Moving Object Trajectories. *Geoinformatica*, 11:159–193, 2007.

[2] Frentzos, E., Gratsias, K., and Theodoridis, Y. Towards the Next Generation of Location-based Services. *Proceedings of W2GIS,* 2007.

[3] Frentzos, E., Gratsias, K., and Theodoridis, Y. Index-based Most Similar Trajectory Search. *Proceedings of ICDE,* 2007.

[4] Guting R. H. and Schneider M. *Moving Object Databases*. Morgan Kaufmann Publishers, CA, 2005.

[5] Oracle 10g. URL: http://otn.oracle.com/pls/db10g/

[6] Pelekis N., Kopanakis I., Ntoutsi I., Marketos G., Andrienko G., and Theodoridis Y.. Similarity Search in Trajectory Databases. *Proceedings of TIME*, 2007.

[7] Pelekis N., Theodoridis Y., Vosinakis S., and Panayiotopoulos T.. HERMES - A Framework for Location-Based Data Management. *Proceedings of EDBT*, 2006.

[8] Pelekis N., Theodoridis Y. Boosting Location-Based Services with a Moving Object Database Engine. *Proceedings of MobiDE*, 2006.

[9] Pfoser, D., Jensen, C. S., Theodoridis Y. Novel Approaches to the Indexing of Moving Object Trajectories. *Proceedings of VLDB*, 2000.

[10] Sistla A. P., Wolfson O., Chamberlain S., and Dao S. Modeling and querying moving objects. *Proceedings of ICDE*, 1997.

[11] Theodoridis, Y. Ten Benchmark Database Queries for Location-based Services. *The Computer Journal*, 46, 6, 713-725, 2003.

[12] Wolfson O., Cao H., Lin H., Trajcevski G., Zhang F., and Rishe N. Management of dynamic location information in domino. *Proceedings of EDBT*, 2002.