TransDB—GPS Data Management with Applications in Collective Transport*

Christian S. Jensen
Department of Computer Science
Aalborg University, Denmark
Selma Lagerlöfs vej 9220-DK
csj@cs.aau.dk

Dalia Tiešytė
Department of Computer Science
Aalborg University, Denmark
Selma Lagerlöfs vej 9220-DK
dalia@cs.aau.dk

ABSTRACT

Recent and continuing advances in geo-positioning, mobile communications, and computing electronics combine to offer opportunities for advanced and affordable collective transport services. As the roads in many parts of the world are facing increasing congestion, it becomes increasingly important to establish collective transport solutions, such as bus services, that are competitive in comparison to the use of private cars. One important ingredient in the provisioning of such solutions is an information system that is always aware of the current location and expected future locations of each bus and that is capable of utilizing this information in real time as well as off-line, e.g., for offering the users accurate arrival information and for creating safe, realistic, and environmentally friendly bus schedules. This paper introduces to an on-going project that explores the advanced data management techniques needed to create an efficient, accurate, and yet inexpensive information system for collective transport monitoring. Focus is on bus travel time prediction and the communication between the vehicles and their surrounding infrastructure.

1. INTRODUCTION

Improvements to the capabilities and affordability of geo-positioning technologies, mobile communication, and computing hardware combine to offer new opportunities for improved collective transport services. We are witnessing a global trend toward public bus services that include on-the-fly, real-time information about bus arrival times to the passengers, via desktop computers and mobile phones on the Internet, or via on-line displays at bus terminals and bus stops.

For example, the public buses in the Aalborg region are equipped with PCs, GPRS-based connectivity to a central server, and Global Positioning System (GPS) receivers for positioning. The system also includes on-line displays at the central bus terminal and at major bus stops. The current system is capable of providing real-time bus arrival information to the displays, and it also interacts with some traffic lights, to improve the timeliness of the bus services when congestion occurs. A similar system is operational in Copenhagen. These systems are typical examples of the bus information systems that are in operation in other countries as well.

As the roads are getting increasingly congested, it becomes increasingly important to convince drivers to leave private cars and instead use collective transport. This may be done by making collective transport services more attractive, specifically by making them reliable (on-time arrivals), fast (e.g., by using prioritized traffic lights), informational, safe, and comfortable. One important factor that reduces the travel times of the users and increases user satisfaction is accurate prediction of travel times of the buses. This accuracy reduces waiting times and the need for departing early in order to be guaranteed to arrive on time.

The travel times of buses are by nature more difficult to predict than, e.g., those of trains. This is so because buses are affected by external factors such as wait times at traffic lights, congestion, and accidents. While travel time prediction for buses is commercially available today, there is still a need for a better prediction that takes into account both historical and real-time automated vehicle location (AVL) data, and that is more general in the sense that it should not be dependent on multiple system-specific parameters. Better prediction leads to better passenger information, but may also be exploited for improving future bus schedules. In addition, better predictions may be exploited for reducing the cost of communication between the buses and the surrounding infrastructure, as with better prediction, less communication is needed.

This project's objective is to develop efficient software techniques that utilize position data for establishing a better correspondence between the status of buses, most importantly their current positions, and the information about this status as known by the surrounding infrastructure, e.g., the central server and the on-line, variable displays at bus stops. In particular, focus is on more accurate prediction of the arrival times of buses at bus stops and on the reduction of the communication needed between the buses and the infrastructure.

The project also explores techniques that enable efficient trajectory data management. As buses progress along their routes, the trajectories that capture the progress so far are recovered from tracking data and are stored together with other historical data. The trajectories are subsequently used by arrival time prediction algorithms (ATPs) that utilize techniques for efficient similarity search in databases of historical trajectories.

It is a central hypothesis that the use of historical and current position data will enable better prediction of bus arrivals at bus stops, in comparison to what is offered by existing techniques. Further, the project is faced with the challenge of ensuring low-cost communication between buses and the infrastructure, which is done by

*This research was funded (in part) by the Danish Research Agency's Programme Commission on Nanoscience, Biotechnology, and IT.

MobiQuitous 2008, July 21–25, 2008, Dublin, Ireland.

Digital Object Identifier: 10.4108/ICST.MOBIQUITOUS2008.3890
http://dx.doi.org/10.4108/ICST.MOBIQUITOUS2008.3890

sending updates from the buses only when needed to preserve accuracy guarantees of predictions, rather than at pre-defined times.

The techniques developed in this project are expected to be applicable in other areas of intelligent transport systems than those relating to buses. The proposed applications include the use of taxis for transportation of physicians and medical patients, children in rural areas, and the elderly. Increasingly efficient transport contributes to the realization of the European Union's strategy to make transportation safer and more environmentally friendly.

Four organizations participate to make the project possible. The participating researchers from Aalborg University have experience with data management in general, and with the management and uses of GPS data, in particular. Nordjyllands Trafikselskab (NT) (`http://www.nordjyllandstrafikselskab.dk`) are the providers of public bus services in Northern Jutland, including the Aalborg region. They supply actual GPS and bus schedule data for the project, pose requirements, and evaluate project results from the point of view of the user. The company, in the beginning of the project known as TNC Connect A/S, now part of Fara ASA (`http://www.fara.dk`), are suppliers of computing, positioning, and communication infrastructures for buses, and the company has delivered key components of the information system deployed by NT. In addition, Fara ASA are working more generally on the development of intelligent transport systems for collective buses. Their AVL systems are aimed at providing high-quality bus services. Finally, Center for Intelligent Transport Systems (CITS) (`http://www.cits.aau.dk`) is a research center that focuses on the development of intelligent transport systems. The center connects projects with national and international companies and organizations to achieve synergy and supply knowledge. ICTS are offering their networking support to the project.

The rest of the paper is outlined as follows. Section 2 covers related work in vehicle tracking, travel-time prediction, similarity search, and pattern mining from vehicle trajectories. Section 3 introduces the system model assumed on the project. Section 4 discusses the arrival-time prediction algorithms under study in the project. Section 5 summarizes the paper.

2. RELATED WORK

State-of-the-art vehicle location tracking techniques are capable of guaranteeing that the real-time position of a vehicle is known by the system with a pre-defined accuracy. Instead of continuously updating the current position, a prediction of the vehicle's movement is used. Once the actual position deviates from the predicted position by a certain distance, an update is issued. Vehicle tracking has been discussed in a number of papers, e.g., [24, 25, 26, 29, 30, 31]. We extend the existing approach by allowing communication not only from the vehicles to the server, but also from the server to the vehicles. This setting allows the server to change its predictions to presumably more accurate ones and share these with the vehicles.

Real-time prediction of bus arrival times have been studied for a couple of decades. Currently, automated vehicle location (AVL) systems have become an active research area within intelligent transport systems. Bus arrival-time prediction (ATP) is an important part of am AVL system. Commonly used AP methods include Kalman filtering (KF), e.g., [3, 8, 20], and artificial neural networks (ANNs), e.g., [6, 11, 14, 15, 18]. Other machine learning methods such as support vector machines [1] are also possible. A model that utilizes historical data and current conditions (including weather) has been proposed [7]. Some researchers use simple mathematic algorithms, e.g., prediction according to deviation from a schedule [17]. A prediction model that includes detection of routes and continuous queries on future arrival times has been proposed [19]. KF prediction is mainly based on real-time data, although historical data can also be used as an external influence parameter. KF works well when a short-term prediction is needed, but deteriorates when prediction is needed for more than one time step. ANNs and other learning methods are based on historical data. Their learning is computationally expensive; therefore, the ANNs cannot be updated in real time. In contrast, the techniques used for arrival time prediction in this project are data centric and data intensive, and they aim to exploit all the historical data available. The algorithms do not require any external data, such as real-time traffic information, which promises easier integration into existing systems of the developed methods.

Considerable research has been conducted on similarity-based retrieval on one-dimensional time series data, such as sensor data, e.g. [2], as well as 2-dimensional trajectories of moving objects. Some authors [27] compare only spatial coordinates, excluding the temporal component. Chen et al. [4, 5] measure edit distance with extensions. Vlachos et al. [28] propose Longest Common Subsequence search for vehicle trajectories. Works by Hwang et al. [12, 13] define times and points of interest that similar trajectories must traverse. In our representations of trajectories, we consider only the relative temporal component, while the positions are fixed according to pre-defined routes. Furthermore, the similarity measure has to be predictive: trajectories that are found to be similar in the past are expected to be similar in the future. To the best of our knowledge, similarity search in the context of collective transport has not been discussed yet.

Pattern mining from trajectories and the clustering of trajectories have been discussed in the literature [9, 10, 16, 27]. Existing works deal with movement of objects in a two-dimensional space. In our case, the vehicle is limited to movement along known routes, meaning that the space can be reduced to being one-dimensional.

This project extends existing works by proposing new approaches and placing them in the context of collective transport. It integrates tracking, similarity search in historical databases, and prediction using those data as well as real-time information into one interacting system, though the proposed techniques may also be applied in other areas, such as logistics, personal routes, or even sensor data. For example, measurements taken throughout a day can be seen as a trajectory, and the historical "trajectories" can be used to predict the up-coming values.

3. THE SYSTEM

We present the tracking and prediction system as it is viewed by the project, and we introduce to the modules that are being developed as party-independent parts of this system.

3.1 System Model

The tracking and prediction system is depicted schematically in Figure 1. The server has a *Central Tracker* as well as *Predictor*, *Similarity Search*, and *Trajectory Recovery* modules.

The *Central Tracker* module communicates with the vehicles and tracks their current positions. When an update is generated, the *Central Tracker* updates its state and informs the predictor module. The *Predictor* can also change its prediction for a vehicle independently of the vehicle. The *Central Tracker* then needs to communicate the new prediction to the vehicle. The tracking policies are discussed in more detail elsewhere [22].

The *Predictor* module predicts arrival times for all vehicles in the system. The *Predictor* obtains the current positions of the buses from the *Tracker*. It also has access to the database historical trajectories via the module *Similarity Search*. When more than one prediction method is enabled, the *Predictor* chooses adaptively the

Digital Object Identifier: 10.4108/ICST.MOBIQUITOUS2008.3890
http://dx.doi.org/10.4108/ICST.MOBIQUITOUS2008.3890

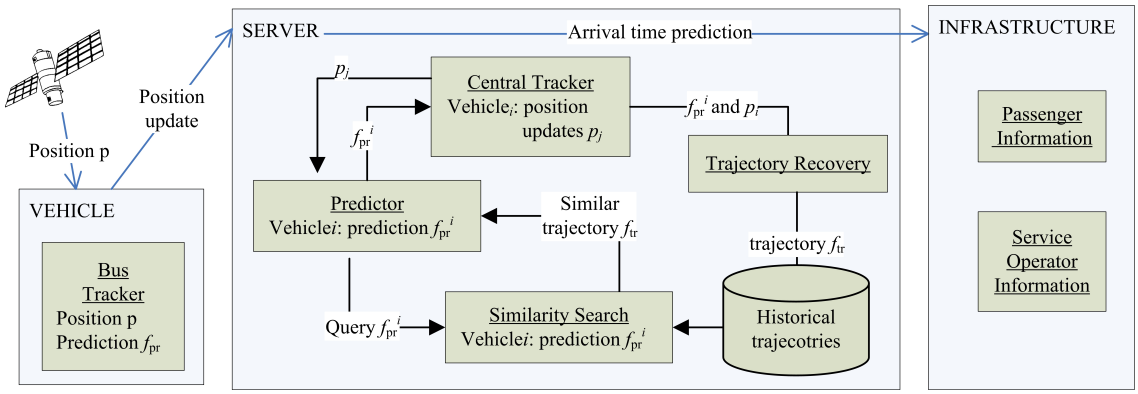

Figure 1: System architecture

most efficient prediction method.

The *Similarity Search* module provides means for efficient access to historical trajectories. We expect that large amounts of historical data are needed for accurate prediction. With very large such databases, it becomes infeasible to compare a current real-time trajectory with each trajectory in the database. An index is needed that would prune those trajectories that should not be considered as candidates. The module enables efficient search among historical data, including both historical trajectories and patterns derived from them.

The *Trajectory Recovery* module recovers trajectories from the tracking data, namely the positions that arrived as updates together with the prediction function that was used for tracking at the time. Such trajectories are stored in the database and can later used for prediction or data analysis. The method is presented elsewhere [23].

3.2 Data Representation

In this section we discuss how the data is represented in the system.

We use *polylines* for capturing the geography of a bus route. A polyline is a sequence of two-dimensional points: $GR = [(x_1, y_1), ..., (x_n, y_n)]$. Other points on the route are calculated using linear interpolation. The points used for route representation, we call vertices: $Ver_i = (x_i, y_i)$, $i = 0, ..., n$, where Ver_0 is the starting point of the route. The vertices are connected by segments. A segment is denoted by Seg_i and has length $d_{\mathrm{E}}(Ver_i, Ver_{i-1})$, where d_{E} is Euclidean distance.

Intuitively, using a higher number of vertices results in a more accurate route representation. The vertices should be chosen so that the actual shapes of the Seg_i are near linear. The fact that the buses travel along pre-defined routes allow us to disregard the geographies of the routes in the actual representation of the positions of buses used in the system. In particular, we use linear referencing for representing the positions of buses in one-dimensional space. Specifically, a route is a linear element: it has a well-defined starting point, and each point on a route can be referred to by its distance from the starting point of the route, as well as by its distance from another point on the route. Each vertex Ver_i is projected to a one-dimensional point: $\perp Ver_0 = 0$, and $\perp Ver_i = \perp Ver_{i-1} + d_{\mathrm{E}}(Ver_i, Ver_{i-1})$. An example is shown in Figure 2.

A route is then represented as a sequence of one-dimensional points, $LR = [p_0, ..., p_n]$, where the p_i are chosen points of interest, also termed timing points, which we are interested in arrival-time predictions for; these may be bus stops in a collective transport system. They do not need to be the same as the Ver_i. They are projected to one dimension by measuring their Euclidean distance from the nearest vertex.

The *raw trajectory* of the traversal of a route by a bus is a function from time to a point on the route, $f_{\mathrm{raw}} : T \rightarrow LR$, as it is best known by the system. This function is non-decreasing and monotone: the vehicle cannot go backwards on the route, and it cannot go off its route.

Due to these properties, and due to the positions $p \in LR$ being one-dimensional, the function f_{raw} is a bijection, with the inverse function is $f_{\mathrm{raw}}^{-1} : LR \rightarrow T$ that captures the travel time $t \in T$ from the start of the route to the position $p \in LR$ on the route. From this, we derive a new trajectory function, $f_{\mathrm{tr}} : LR \rightarrow \Delta T$, by setting the time $\Delta t \in \Delta T$ to be the travel time in-between two subsequent points, $\Delta t_i = f_{\mathrm{raw}}^{-1}(p_i) - f_{\mathrm{raw}}^{-1}(p_{i-1})$.

A prediction function is denoted as $f_{\mathrm{pr}} : LR \rightarrow T$, where the time $t \in T$ is the predicted arrival time for the position $p \in LR$. The prediction is available for future positions, i.e., positions beyond the current position $p_{cur} \in LR$. When $p < p_{cur}$, $f_{\mathrm{pr}}(p)$ returns historical predictions. When the function f_{pr} is updated, only the future predictions are changed, so that the historical predictions can be used for data analyses.

Prediction error $err(t, f_{\mathrm{pr}}, f_{\mathrm{tr}})$ is the difference between the actual travel times f_{tr} and the predicted travel times f_{pr} as predicted at some fixed time t. The error per trajectory is calculated as the average squared difference per point: $err(t) = \frac{1}{n} \sum_{i=cur+1}^{n} (|f_{\mathrm{pr}}(p_i) - f_{\mathrm{pr}}(p_{i-1})| - f_{\mathrm{tr}}(p_i))^2$. If prediction for some of the points p_i is not of interest, they are omitted. The error is calculated per travel time for a segment rather than comparing the absolute times. As a result, a prediction error is penalized for only once, locally where it occurs.

Figure 3 offers a diagram of the data types that are used for the communication in-between the vehicles and the server, with focus on the tracking module. The type *Vehicle* represents a vehicle moving along its route. On the server, each vehicle is represented by a separate data structure and is tracked individually. The *Vehicle* is aware of its *Route* that is represented as a sequence of *Vertex*es (that can be specialized into, e.g., *Stop*s and *Crossroad*s), and a sequence of *Segment*s (road segments).

These objects have certain properties, e.g., geometrical information. The type *VehicleTracker* on the vehicle side tracks the location of the vehicle and compares this with the server's prediction. It updates the server by sending a message *UpdateServer*. The server subsequently (or when needed) updates the infrastructure by sending *UpdateInf*. For example, in case of collective transport, each

Digital Object Identifier: 10.4108/ICST.MOBIQUITOUS2008.3890
http://dx.doi.org/10.4108/ICST.MOBIQUITOUS2008.3890

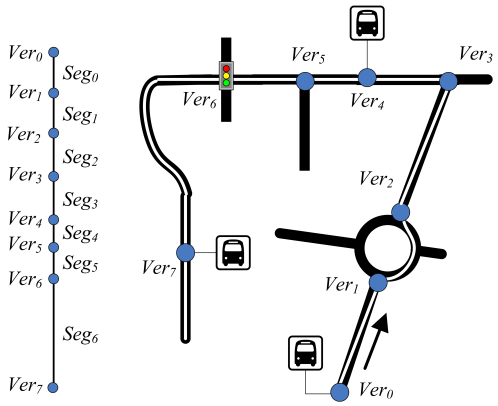

Figure 2: Route representation

Display, which is located at a bus stop and informs the passengers about expected arrival times, has to be updated. The server can also communicate to the vehicle by issuing a message *UpdateVehicle*.

4. ATP ALGORITHMS

The frequency of the communication in-between the vehicles and the server and the accuracy of ATPs are closely related. Accurate prediction helps to reduce communication costs, while position updates sent from the vehicle to the server allow to update the prediction in real time.

4.1 Cost-Effective Communication

We aim to reduce communication costs by minimizing the number of updates sent from the vehicles to the server, while still preserving the required accuracy guarantees for vehicle locations. A vehicle sends an update to the server only when the movement predicted for it deviates from its actual movement by an agreed-upon threshold. We analyze two policies: (1) update only when the actual position of the bus deviates from the predicted position by a distance threshold; (2) update only when the arrival time at the next timing point predicted by the bus deviates from the server's prediction by a time threshold. The server can also update its prediction independently, e.g., based on information received from another vehicle traveling on the same route, and send the new prediction to the vehicle.

4.2 History-Based Prediction

Data mining methods can be applied for pattern mining from historical trajectories. Data mining helps to extract previously unknown information from large databases. The process consists of the following steps: (1) selection of data, (2) preprocessing of the data, (3) transformation of the preprocessed data, (4) pattern recognition from the transformed data, (5) and knowledge extraction from the patterns [21]. The following types of classification algorithms can be considered: (1) template matching that measures the similarity between the trajectory and the template, (2) statistical approaches that calculates the probabilities that the trajectories belong to certain classes, (3) pattern matching where rules help to construct the sentences that represent the patterns, and (4) ANNs which are non-algorithmic methods that can learn complex non-linear relationships between the attributes. An ANN can use a number of available attributes as input, e.g., travel times of other vehicles on other routes, travel times on previous segments, type of a day (e.g., weekend), type of a day time (e.g., peak hour), etc.

We pay particular attention to template matching, considering two types of classification: (1) each historical trajectory represents its own class, and (2) trajectories are grouped into clusters, with their centers representing the templates that the real-time trajectories are matched to. In the first case, the database of historical trajectories is searched every time a new prediction has to be formed, or the old prediction has to be revised. The *Similarity Search* module uses an index structure that enables efficient access to historical data. Considering each trajectory as its own class is advantageous when no frequent patterns are recognized: similar situations may have occurred only a couple of times in the past. By matching the real-time trajectory to a similar trajectory in the past, we expect to recognize such specific situations. In the second case, clustering algorithms are used to recognize patterns in the historical data. The real-time trajectory is then matched against the centers of the clusters, and the most similar center is chosen for prediction.

Similarity (or distance) measures are used for both clustering and template matching. A trajectory is viewed as a k-dimensional point, $k \leq n$, where each dimension represents the already known travel time in-between two subsequent points p_{i-1}, p_i on the vehicle's route. The Euclidean distance has a number of desirable properties: it is a metrics, it contends well with outliers when used with the trajectory representation described above, and it is efficient to compute. This measure is used for clustering. When comparing a real-time trajectory to a historical pattern, a different distance measure is applied. For this purpose, we propose a weighted Euclidean (or other L^p) distance, where the more recent points have higher weights than the older points. This way, the similarity measure allows to adjust to a changing travel pattern while the vehicle is on its route.

The prediction is re-estimated in real time as the vehicle proceeds along its route. First, the prediction is adjusted every time an update from the vehicle arrives. The function $UpdatePR$ updates the shared prediction on both the server side and the vehicle side. Second, the prediction is revised periodically, e.g., when the vehicle arrives at a point of interest, and when the prediction error given by the template is too large. If a new template is chosen by the server, the vehicle is updated with the newly-formed prediction. A general algorithm is presented in Alg. 1. The function f_{tr} models the movement of the vehicle, and the function f_{pr} predicts the future arrival times. When the template is chosen, the relative time values Δt_i are transformed into absolute arrival times t_i. The parameter t_{h} controls how far into the history the algorithm looks when evaluating the prediction error.

4.3 Dynamic Choice of ATP

A number of factors have to be considered in the development of ATP algorithms. Different timing and accuracy requirements may be posed for rural versus urban areas, and different requirements are posed for short-term versus long-term prediction. For example, higher accuracy is expected of a short-term prediction than of a long-term prediction. We propose a flexible prediction system that allows to adjust dynamically to the changing environment. The central server has access to a library of prediction methods. It dynamically evaluates the accuracies of the available methods and chooses the one that offers the best prediction for the current situation. Such a system is easily extensible, as new prediction methods can be plugged in. Only those methods that give accurate prediction will be used for the actual prediction that is provided to the outside world. A general algorithm is presented in Alg. 2. It works similarly to the Alg. 1. The difference is in the updating of the prediction: the algorithm allows not only to revise the chosen template, but also to change the prediction method. The algorithm ATP is

Digital Object Identifier: 10.4108/ICST.MOBIQUITOUS2008.3890
http://dx.doi.org/10.4108/ICST.MOBIQUITOUS2008.3890

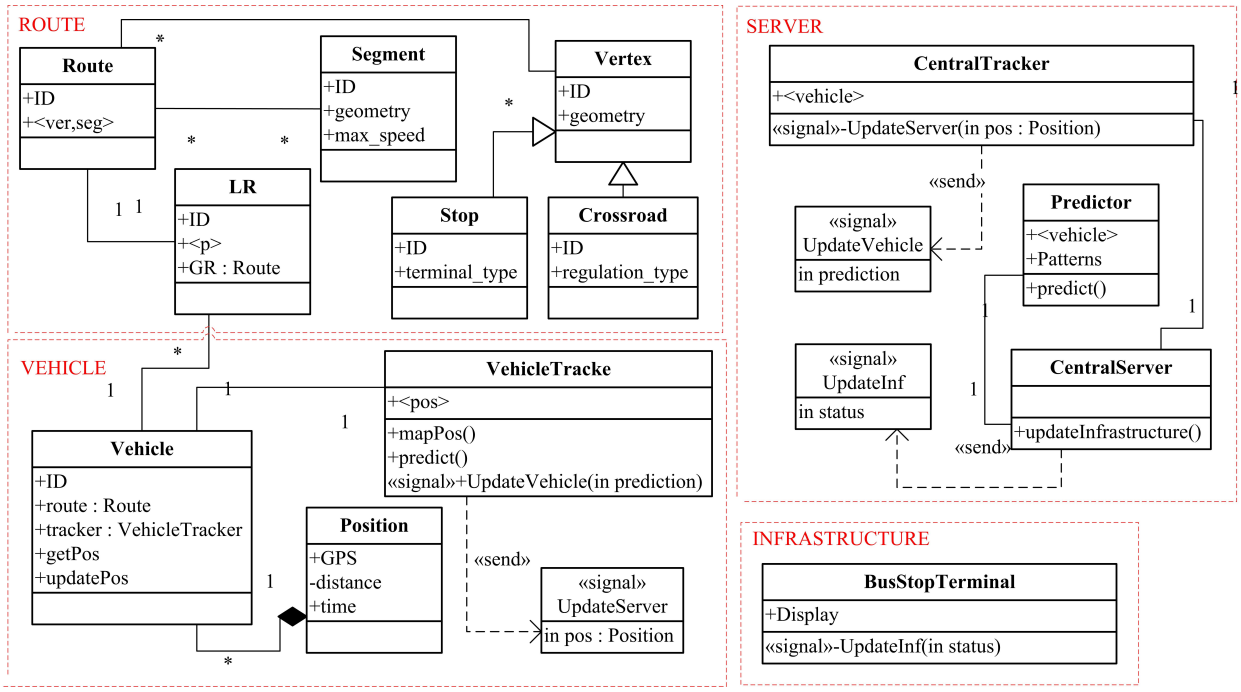

Figure 3: Data representation in the communication system

Algorithm 1: *ATPTemplate*: template-based continuous prediction

Input: Historical templates F_{tr}, prediction threshold thr_{pr}, distance function d.

$f_{\mathrm{pr}} \leftarrow template[t_i \leftarrow t_{i-1} + \Delta t_i]$
Initialize the vehicle and the infrastructure with f_{pr}
$f_{\mathrm{tr}}(0) \leftarrow 0$
while Vehicle is on the route **do**
  if Update (p_{cur}, t_{cur}) received **then**
    $f_{\mathrm{pr}} \leftarrow UpdatePR(f_{\mathrm{pr}})$ so that $f_{\mathrm{pr}}(p_{cur}) = t_{cur}$
    Update the infrastructure with f_{pr}
  end if
  if $f_{\mathrm{pr}}(t_{cur}) = p_i$ **then**
    $f_{\mathrm{tr}}(p_i) \leftarrow t_{cur} - f_{\mathrm{tr}}(p_{i-1})$
    if $err(t_{cur} - t_{\mathrm{h}}, f_{\mathrm{tr}}, f_{\mathrm{pr}}[t_{cur} - t_h, t_{cur}]) > thr_{\mathrm{pr}}$ **then**
      $template \leftarrow \mathrm{argmin}_{f'_{\mathrm{tr}} \in F_{\mathrm{tr}}}\ d(f_{\mathrm{tr}}, f'_{\mathrm{tr}})$
      $f_{\mathrm{pr}} \leftarrow template[t_i \leftarrow t_{i-1} + \Delta t_i]$
      Update the vehicle and the infrastructure with f_{pr}
    end if
  end if
end while

Algorithm 2: *ChoosePrediction*: dynamic choice of ATP

Input: A library of ATPs $ATPLIB$, prediction threshold thr_{pr}

$ATP \leftarrow$ initial prediction method from $ATPLIB$
$F_{\mathrm{pr}} \rightarrow$ predictions for each $ATP \in ATPLIB$
$f_{\mathrm{pr}} \leftarrow$ initial prediction $pr()$
Initialize the vehicle and the infrastructure with f_{pr}
$f_{\mathrm{tr}}(0) \leftarrow 0$
while Vehicle is on the route **do**
  if Update (p_{cur}, t_{cur}) received **then**
    $f_{\mathrm{pr}} \leftarrow UpdatePR(ATP, p_{cur}, t_{cur})$
    Update F_{pr}
    Update the infrastructure with f_{pr}
  end if
  if $f_{\mathrm{pr}}(t_{cur}) = p_i$ **then**
    $f_{\mathrm{tr}}(p_i) \leftarrow t_{cur} - f_{\mathrm{tr}}(p_{i-1})$
    if $err(t_{cur} - t_h, f_{\mathrm{tr}}, f_{\mathrm{pr}}[t_h, t_{cur}]) > thr_{\mathrm{pr}}$ **then**
      $ATP \leftarrow ATP \in ATPLIB$ such that corresponding $f_{\mathrm{pr}}[t_h, t_{cur}] \in F_{\mathrm{pr}}$ minimizes err
      $f_{\mathrm{pr}} \leftarrow ATP(f_{\mathrm{tr}}, t_{cur})$
      Update F_{pr}
      Update the vehicle and the infrastructure with (f_{pr}, ATP) if changed
    end if
  end if
end while

chosen from the library of algorithms $ATPLIB$.

5. SUMMARY

This paper has introduced a project called TransDB in which a dynamic, integrated system for managing position-related data of vehicles traveling along pre-defined routes is being developed. The system integrates a number of modules that each has an assigned task: communication; formation of vehicle trajectories from tracking data; similarity search and management of historical data; and real-time prediction of future travel times. The paper focuses on work in progress: the prediction of future travel times and the dynamic choice of ATP while a vehicle is moving along its route.

We expect that the proposed methods will be robust and adapt to changes in the environment. The framework that allows to choose the most appropriate ATP on the fly can be used both in a deployed system and for the evaluation purposes where ATPs are being evaluated by replaying historical traversals of routes. The system uses only historical data, thus allowing the methods to be implemented where external data, such as real-time traffic information, is not available. It does not require frequent communication, though at the same time it enables position accuracy guarantees.

Digital Object Identifier: 10.4108/ICST.MOBIQUITOUS2008.3890
http://dx.doi.org/10.4108/ICST.MOBIQUITOUS2008.3890

In the near future, the proposed algorithms are going to be implemented and tested in the collective transport management system provided by the project's industrial collaborator, Fara ASA.

6. REFERENCES

[1] Y. Bin, Y. Zhongzhen, and Y. Baozhen. Bus arrival time prediction using support vector machines. *Journal of Intelligent Transportation Systems*, 10(4):151–158, 2006.

[2] B. Bollobas, G. Das, D. Gunopulos, and H. Mannila. Time-series similarity problems and well-separated geometric sets. In *Proc. of the 13th Annual Symposium on Computational Geometry*, pp. 454–456, 1997.

[3] F. Cathey and D. Dailey. A prescription for transit arrival/departure prediction using automatic vehicle location data. *Transportation Research Part C*, pp. 241–264, 2003.

[4] L. Chen and R. Ng. On the marriage of edit distance and lp norms. In *Proc. of the 30th Intl. Conference on Very Large Data Bases*, pp. 792–803, 2004.

[5] L. Chen, M. T. Özsu, and V. Oria. Robust and fast similarity search for moving object trajectories. In *Proc. of the ACM SIGMOD Intl. Conference on Management of Data*, pp. 491–502, 2005.

[6] S. I.-J. Chien, Y. Ding, and C. Wei. Dynamic bus arrival time prediction with artificial neural networks. *Transportation Engineering*, 128(5):429–438, 2002.

[7] E.-H. Chung and A. Shalaby. Expected time of arrival model for school bus transit using real-time global positioning system-based automatic vehicle location data. *Journal of Intelligent Transportation Systems*, 11(4):157–167, 2007.

[8] D. Dailey, S. Maclean, F. Cathey, and Z. Wall. Transit vehicle arrival prediction: an algorithm and a large scale implementation. *Transportation Research Record, Transportation Network Modeling*, pp. 46–51, 2001.

[9] Z. Fu, W. Hu, and T. Tan. Similarity based vehicle trajectory clustering and anomaly detection. In *Proc. of the IEEE Intl. Conference on Image Processing*, pp. 602–607, 2005.

[10] F. Giannotti, M. Nanni, F. Pinelli, and D. Pedreschi. Trajectory pattern mining. In *Proc. of the 13th ACM SIGKDD Intl. Conference on Knowledge Discovery and Data Mining*, pp. 330–339, 2007.

[11] J. R. Hee and L. R. Rilett. Bus arrival time prediction using artificial neural network model. In *Proc. of the 7th Intl. IEEE Conference on Intelligent Transportation Systems*, pp. 988–993, 2004.

[12] J. Hwang, H. Kang, and K. Li. Spatio-Temporal Similarity Analysis between Trajectories on Road Networks. In *Proc. of the ER Workshop on Perspectives in Conceptual Modeling*, pp. 280–289, 2005.

[13] J. Hwang, H. Kang, and K. Li. Searching for Similar Trajectories on Road Networks Using Spatio-temporal Similarity. In *Proc. of the 10th East European Conference on Advances in Databases and Information Systems*, pp. 282–295, 2006.

[14] R. H. Jeong. *The Prediction of Bus Arrival Time Using Automatic Vehicle Location Systems Data*. Doctoral dissertation, Texas A&M University, 2004.

[15] R. Kalaputapu and M. J. Demetsky. Modeling schedule deviations of buses using automatic vehicle location data and artificial neural networks. *Transportation Research Record*, 1497:44–52, 1995.

[16] J.-G. Lee, J. Han, and K.-Y. Whang. Trajectory clustering: A partition-and-group framework. In *Proc. of the ACM SIGMOD Intl. Conference on Management of Data*, pp. 593–604, 2007.

[17] W.-H. Lin and J. Zeng. An experimental study on real time bus arrival time prediction with GPS data. *Journal of Transportation Research Board*, 78:101–109, 1999.

[18] T. Park, S. Lee, and Y.-J. Moon. Real time estimation of bus arrival time under mobile environment. In *Proc. of Intl. Conference on Computational Science and Its Applications*, pp. 1088–1096, 2004.

[19] B. Predic, D. Stojanovic, S. Djordjevic-Kajan, A. Milosavljevic, and D. Rancic. Prediction of bus motion and continuous query processing for traveler information services. In *Proc. of the 11th East European Conference on Advances in Databases and Information Systems*, pp. 234–249, 2007.

[20] A. Shalaby and A. Farhan. Prediction model of bus arrival and departure times using AVL and APC data. *Journal of Pubublic Transport*, 7:41–61, 2004.

[21] B. Son, H. J. Kim, C.-H. Shin, and S.-K. Lee. Bus arrival time prediction method for its application. In *Proc. of the 8th Intl. Conference on Knowledge-Based Intelligent Information and Engineering Systems*, pp. 88–94, 2004.

[22] D. Tiešytė and C. Jensen. Efficient Cost-Based Tracking of Scheduled Vehicle Journeys. In *9th Intl. Conference on Mobile Data Management*, pp. 9–16, 2008.

[23] D. Tiešytė and C. S. Jensen. Recovery of vehicle trajectories from tracking data for analysis purposes. In *Proc. of the Sixth European Congress and Exhibition on Intelligent Transport Systems and Services*, 12 pages, 2007.

[24] D. Tiešytė and C. S. Jensen. Challenges in the tracking and prediction of scheduled-vehicle journeys. In *Proc. of the IEEE PerCom Workshops*, pp. 407–412, 2007.

[25] A. Čivilis, C. S. Jensen, J. Nenortaite, and S. Pakalnis. Efficient tracking of moving objects with precision guarantees. *Proc. of the First Annual Intl. Conference on Mobile and Ubiquitous Systems: Networking and Services*, 164–173, 2004.

[26] A. Čivilis, C. S. Jensen, and S. Pakalnis. Techniques for efficient road-network-based tracking of moving objects. In *IEEE Transactions on Knowledge and Data Engineering*, 17(5):698–712, 2005.

[27] M. Vlachos, D. Gunopulos, and G. Kollios. Robust similarity measures for mobile object trajectories. In *Proc. of the 13th Intl. Workshop on Database and Expert Systems Applications*, pp. 721–726, 2002.

[28] M. Vlachos, G. Kollios, and D. Gunopulos. Discovering similar multidimensional trajectories. In *Proc. of the 18th Intl. Conference on Data Engineering*, pp. 673–684, 2002.

[29] O. Wolfson, S. Chamberlain, S. Dao, L. Jiang, and G. Mendez. Cost and imprecision in modeling the position of moving objects. In *Proc. of the 14th Intl. Conference on Data Engineering*, pp. 588–596, 1998.

[30] O. Wolfson, A. P. Sistla, S. Chamberlain, and Y. Yesha. Updating and querying databases that track mobile units. *Distributed and Parallel Databases*, 7(3):257–387, 1999.

[31] O. Wolfson and H. Yin. Accuracy and resource consumption in tracking moving objects. In *Proc. of the Symposium on Spatial and Temporal Databases*, pp. 325–343, 2003.

Digital Object Identifier: 10.4108/ICST.MOBIQUITOUS2008.3890
http://dx.doi.org/10.4108/ICST.MOBIQUITOUS2008.3890