Challenges in the Tracking and Prediction of Scheduled-Vehicle Journeys

Dalia Tiešytė and Christian S. Jensen
Department of Computer Science
Aalborg University, Denmark
E-mail: {dalia|csj}@cs.aau.dk

Abstract

A number of applications in areas such as logistics, cargo delivery, and collective transport involve the management of fleets of vehicles that are expected to travel along known routes according to fixed schedules. Due to road construction, accidents, and other unanticipated conditions, the vehicles deviate from their schedules. At the same time, there is a need for the infrastructure surrounding the vehicles to continually know the actual status of the vehicles. For example, anticipated arrival times of buses may have to be displayed at bus stops. It is a fundamental challenge to maintain this type of knowledge with minimal cost.

This paper characterizes the problem of real-time vehicle tracking using wireless communication, and of predicting the future status of the vehicles when their movements are restricted to given routes and when they follow schedules with the best effort. The paper discusses challenges related to tracking, to the prediction of future travel times, and to historical data analysis. It also suggests approaches to addressing the challenges.

1 Introduction

This paper concerns the challenges inherent in maintaining, at minimum cost, up-to-date information on the status of each vehicle belonging to a fleet of vehicles traveling on scheduled journeys. For example, the fleet may consist of public buses, and the status information may be used in the surrounding infrastructure that encompasses on-line arrival time displays at bus stops and controllable traffic lights. As other examples, the fleet may consist of public service taxis or delivery trucks. In general, real-time vehicle status information is useful for the managers and the users of the fleets. Such information allows these actors to observe the movement of a vehicle, to anticipate future travel times, and to estimate the arrival times at scheduled timing points (e.g., bus stops or delivery points), to plan and reschedule journeys, etc.

Systems are already available that enable this type of monitoring. For example, some existing systems [1] employ PCs on-board the vehicles, GPRS and WiFi for data communication, and GPS and tag readers for positioning. The key challenge is to accomplish the monitoring accurately and efficiently.

We consider scenarios where the vehicles follow their routes and time schedules with the best effort. A route is defined as a sequence of road segments in a digital road network, and a schedule consists of points along a route, where each point has an associated (scheduled) arrival and departure time (these are termed timing points).

When a vehicle travels along public roads, it is inherently difficult to predict the progress of the vehicle, as its progress is affected by external factors such as waiting times at traffic lights, congestion, road construction, weather conditions, and accidents. The infrastructure that surrounds a vehicle must therefore be informed on a continual basis about the status of the vehicles in order to ensure an appropriate degree of consistency between the actual status of the vehicles and the knowledge of this status in the infrastructure. However, frequent updates introduce high communication costs, and server-side updates easily become a bottleneck. Efficient tracking techniques are then needed that reduce the numbers of updates sent from the vehicles to the server that represents the infrastructure, and from the server to the vehicles, while maintaining sufficiently accurate vehicle status information in the infrastructure.

In an existing approach to tracking, the server predicts a vehicle's near-future position and shares this prediction with the vehicle (e.g., [2, 3, 4, 5, 6]). The vehicle issues an update to the server with its current position-related status when its actual position deviates from the predicted position by more than an agreed-upon threshold. A new prediction is then formed, and the procedure repeats itself. We extend this general prediction sharing scheme. In particular, we propose to extend the update policies to allow for complex prediction functions and cost-driven decisions.

The efficiency of the tracking depends on the ability to accurately predict travel times, which, as stated, is challenging since travel times depend on many external conditions. Records of historical vehicle traversals of routes may be utilized in prediction algorithms. Thus, similar journeys from the past can be identified, matched against an ongoing journey, and used to predict the future progress of that journey. To enable the utilization of historical data, definitions of journey similarity and of which criteria should be used to identify past journeys that match the current journey are needed.

Public bus services is an application area where tracking and prediction techniques are already being applied, and where such techniques face stringent quality requirements. Accurate prediction of bus travel times is important because it has the potential for reducing the travel times of the passengers and for increasing their satisfaction. In particular, real-time information is provided to the passengers at bus stops, on the Internet, and via mobile devices. This information reduces waiting times and the need for departing early in order to arrive on time. However, buses pose challenges: the travel times are inherently difficult to predict; knowledge of their positions must be maintained with high accuracy; and large fleets of buses introduce high communication and update processing costs.

The predominant travel-time prediction techniques in the area of public transportation are Kalman filtering (KF), e.g., [7, 8, 9], and artificial neural networks (ANNs), e.g., [10, 11, 12]. The KF algorithms utilize real-time data; studies suggest that they perform well for short-term prediction. The ANNs utilize historical data. Their learning algorithms are computationally expensive, meaning that they cannot be updated in real-time. We propose to extend the existing models by combining real-time algorithms with knowledge extracted from historical journey data.

This paper covers key challenges being addressed in an on-going research project (www.cs.aau.dk/TransDB). The industrial partner in this project, TNC Connect, has automatic vehicle location and real-time passenger information as its business area and supplies intelligent transport systems for public bus operations. One of its customers also takes part in the project. The paper more specifically covers challenges in the development of efficient tracking techniques and the prediction of future travel times of scheduled-vehicle journeys. Efficient tracking relies on accurate predictions. Patterns, derived from historical journeys, are utilized for the travel time prediction.

The remainder of the paper is outlined as follows. Section 2 introduces the assumed system architecture and presents a framework for the efficient vehicle tracking. Section 3 discusses challenges faced by tracking and prediction algorithms for scheduled-vehicle journeys. Section 4 concludes the paper.

2 Tracking System Architecture

The components of a transportation system encompass moving vehicles, a central tracking server, and additional, surrounding infrastructure components that are connected to the system either wirelessly or via a wired network. The server and other infrastructure require information related to the current status of the vehicles, and the server can provide the vehicles with relevant information. In the tracking context, we focus on position-related vehicle status, although other data (e.g., error messages) can also be considered part of the vehicle's status. The vehicles are equipped with computing, positioning, and data communication capabilities.

The main purpose of tracking in a transportation system is to maintain some required degree of consistency between the real status of the vehicles and the record to these on the server. The required consistency may be more or less stringent, which results in higher or lower costs of maintaining this dynamic, distributed system in a consistent state.

The diagram in Figure 1 describes the communication between the *central server* (CS), a *vehicle* (VH), and the

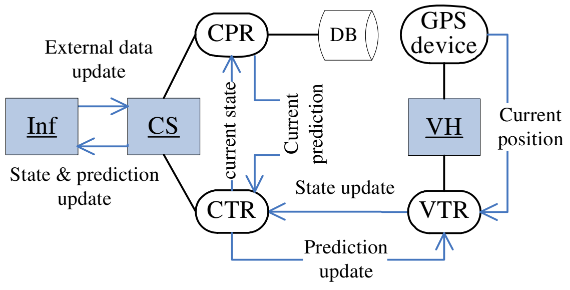

Figure 1. System architecture

surrounding infrastructure (Inf). The CS has a *Central Tracker* (CTR) and a *Central Predictor* (CPR) module. The VH has a *Vehicle Tracker* (VTR) module and a GPS receiver. The CTR module is responsible for controlling the communication. The VTR receives GPS positions continually and updates the CTR with its position-related status when necessary in order to maintain the required consistency. The CPR module predicts the future travel times of the vehicle. It obtains the current vehicle status from the CTR, and it has access to a database with schedules, historical journey data, and other relevant data. Updates of other information (e.g., traffic volumes) are also possible. The CTR updates each vehicle with a new prediction when needed. The infrastructure can be updated by request (e.g., from mobile users), or when the system state changes.

Figure 2 describes a tracking-and-prediction scenario. The server recomputes its prediction when triggering events occur: such events include changes in external data, an update from a VTR, or the start of a vehicle on a jour-

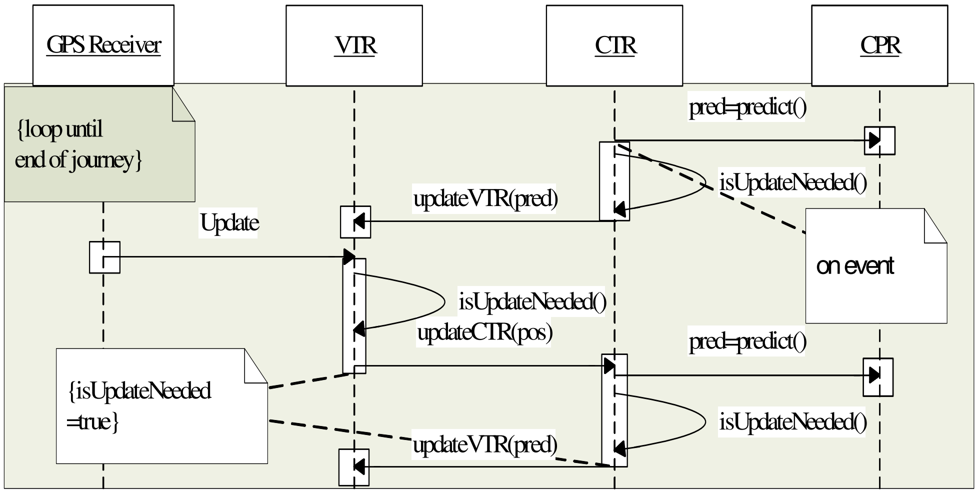

Figure 2. Sequence diagram representing vehicle tracking

ney. At the start of a journey, the CTR calls function $predict$ in the CPR module to predict the journey pattern. The CTR examines whether the VTR needs to be updated, and possibly updates it with an initial prediction by calling function $updateVTR$. When the VTR receives an $Update$ from the GPS receiver, the update policy decides whether the CTR must be updated (function $isUpdateNeeded$). If yes, the CTR is updated by calling function $updateCTR$, and the VTR recomputes the tracking function (function $updateVTR$). The CPR then recomputes the server's prediction (function $predict$). The CTR decides whether the VTR has to be updated with the new predictions, and it then possibly calls function $updateVTR$. The loop is repeated throughout the journey.

3 Tracking and Prediction Challenges

3.1 Accuracy Guarantees

The tracking system is often required to guarantee a certain server-side accuracy of a continuous variable related to a vehicle. In this paper, we consider two variables, namely the current position of a vehicle and the arrival time at a given timing point (e.g., a bus stop). The accuracy guarantee is that the actual values of these do not deviate from the values assumed by the server and infrastructure by more than a global threshold thr.

Two scenarios are possible. In the first, the vehicle and the server always use the same prediction function and have the same data available for prediction. In addition, the vehicle knows its actual, current status. It updates the server when the shared prediction deviates by thr from the actual status.

In the second scenario, the server has more data available for prediction than has the vehicle. The server can then update the vehicle when it updates its prediction for the vehicle. In this scenario, the threshold thr is split between the vehicle and the server. The threshold assumed by the vehicle is $thr_{\mathrm{v}} \leq thr$, and the threshold on the server is $thr_{\mathrm{c}} = thr - thr_{\mathrm{v}}$. The server then has to update the vehicle when its actual prediction deviates by thr_{c} from the prediction that is assumed by the vehicle. Furthermore, the server is able to update the vehicle at any time—this may reduce the number of future vehicle and server updates, in this case reducing the overall cost associated with the journey. The second scenario generalizes the first, and we proceed to consider it more closely.

If the server changes its prediction for a vehicle due to external data only infrequently, most of threshold thr should be assigned to the vehicle, i.e., to thr_{v}. In general, the quality of the server's predictions, the changes in external data, and the threshold values used all influence the tracking costs. The following generic cost function estimates the total cost of updates issued by the server and vehicle from the current time and until the end of the ongoing journey, given the current server state s_{cur} that captures the information currently available to the server: $cost_{\mathrm{journey}}(s_{\mathrm{cur}}) = c_{\mathrm{v}} \sum_j P(up_{\mathrm{v}} = j | s_{\mathrm{cur}}) j + c_{\mathrm{c}} \sum_j P(up_{\mathrm{c}} = j | s_{\mathrm{cur}}) j$, where up_{c}, c_{c}, up_{v}, and c_{v} are the numbers of updates and the costs of a single update from the server to the vehicle and from the vehicle to the server, respectively. The function sums up the products of the costs of one update, the probabilities that the vehicle and server issue j updates, and j, where j varies from 1 to the maximum possible number of updates.

The cost until the end of the journey can be difficult to estimate; it is more feasible to estimate the cost of a part of the journey until some time t_k, assuming that at that time, the server is in a state s_k that is independent of the current decision on whether or not to update the vehicle. Denote the state of server s_- if the decision d_- of not to update is taken, and denote it s_+ if the decision d_+ to update is taken and the update is issued. Denote the transition from state s_i to s_k as $s_i \rightarrow s_k$. Then, the server should issue an update, if $cost_{\text{journey}}(s_- \rightarrow s_k) > cost_{\text{journey}}(s_+ \rightarrow s_k) + c_c$.

Support for accuracy guarantees is faced with several challenges: (1) The threshold thr can be set manually. If a limited budget for the tracking costs (wireless communication and server-side update costs) is given, it can be set by the system. Different thresholds can then be assigned to different vehicles and to different parts of a route so that the budget is used optimally. (2) The appropriate settings of many parameters, including thr_{c}, thr_{v}, and the parameter that captures the quality of the server's predictions, depend on external conditions. The system should be able to maintain these settings itself under changing conditions. (3) The probabilities in the cost function depend on the function itself, which renders the cost function recursive and difficult to specify. (4) The thresholds, the parameters for the cost function, and the prediction accuracy depend on each other.

3.2 Tracking Algorithms

The tracking algorithms currently employed in public transportation systems often rely on timing points. When a vehicle arrives at and departs from a timing point, it updates the server with its current status. Such tracking has several drawbacks: (1) The server is unaware of actual travel pattern deviations when the vehicle is traveling in-between timing points. This renders it difficult to accurately predict near-future travel times. (2) Only relatively poor accuracy guarantees may be derived. (3) Unnecessary costs are incurred when the vehicle is traveling as expected.

We propose to employ tracking algorithms that are based on journey pattern prediction, as described earlier. The two algorithms that we discuss next track a continuous variable, which is either the position of a vehicle or a vehicle's predicted arrival time at a timing point.

Position threshold-based tracking The continuous variable being tracked is the current position of a vehicle. The server and each vehicle share a function $f_{\text{c}} : T \rightarrow P$ that predicts the vehicle's position at time t. The vehicle monitors the distance $|f_{\text{a}}(t) - f_{\text{c}}(t)|$ between its actual and predicted positions, and when this distance reaches the threshold thr_{v}, it updates the server with its status.

Similarly, the server monitors the distance $|f_{\text{pr}}(t) - f_{\text{c}}(t)|$ between its own prediction f_{pr} and the prediction f_{c} it shares with the vehicle, and it updates the vehicle when this distance reaches position threshold thr_{c}. The server can also update the vehicle at other times if this is believed to reduce subsequent costs. The shared prediction f_{c} is recomputed each time the vehicle or the server issues an update. This framework is applicable when the position of a vehicle is to be known by the server and infrastructure by some specified accuracy.

Arrival time threshold-based tracking The continuous variable being tracked is now a vehicle's arrival time at a future timing point. The server and the vehicle share function $g_{\text{c}} : T \rightarrow T$ that predicts the vehicle's arrival time at the timing point as of time t. When, at time t_{up}, the vehicle's prediction $g_{\text{v}}(t)$ deviates from $g_{\text{c}}(t)$ by threshold thr_{v}, the vehicle updates the server, and a new shared prediction g_{c} is computed based on the position $f_{\text{a}}(t_{\text{up}})$ of the vehicle at the time of the update. The vehicle may recompute g_{v} due to $f_{\text{a}}(t)$, which it monitors. It is expected that the predictions by g_{v} are at least as accurate as those by g_{c}. As in position-based tracking, the server monitors the distance $|g_{\text{pr}}(t) - g_{\text{c}}(t)|$ between the value of its prediction function g_{pr} and the shared prediction, and it updates the vehicle when the threshold thr_{c} is reached, or when a cost function suggests that this will reduce subsequent cost.

In some cases, an update from the vehicle to the server can be eliminated even though the threshold thr_v is exceeded. For example, this is possible if g_{v} predicts an early arrival at the timing point and the prediction by g_{c} is that the vehicle will arrive at least thr_{c} time units early, assuming that only departure information is relevant to the problem domain. In this case, the vehicle knows that the server's function g_{pr} also predicts an early arrival.

Using position tracking is not optimal when the objective is to predict arrival times. With position tracking, if a vehicle stops within distance thr_{v} of a timing point, it is possible for the server to believe that the vehicle is at the timing point, which may yield incorrect information in the infrastructure. This does not happen with time prediction tracking. Also, updates are not required where the vehicle's actual position deviates considerably from the server's prediction if this is expected to not delay the arrival. Actual arrival time accuracy guarantees cannot be provided; however, position accuracy guarantees may be derived given all prediction functions.

Challenges Development of tracking algorithms is faced with several challenges: (1) Accuracy guarantees have to be maintained while minimizing the status update costs. (2) The cost of tracking depends on the quality of predictions—it is a challenge to minimize the effects of inaccurate predictions. The estimation of prediction uncertainties may help evaluate prediction algorithms and control tracking parameters. (3) System errors, communication delays, and measurement inaccuracies must be taken into account.

3.3 Prediction Algorithms

Accurate predictions of the movements of a vehicle must be provided to the surrounding infrastructure, and such predictions are also exploited by the tracking algorithms. The quality of a prediction is estimated by means of the notion of prediction uncertainty. Both the average prediction error (i.e., the average difference between the predicted and the actual value) and the variance of the prediction error must be minimized. Large, random errors are usually more frustrating for the users than small, frequent errors, and they impact tracking efficiency more adversely.

It is important that accurate arrival times are predicted as early as possible. The scheduled arrival times are not always the best predictions even at the start of a journey. A large number of more or less predictable factors may affect a journey. Movement patterns derived from historical journey data may be utilized in conjunction with real-time GPS data to predict arrival times.

Predictions are continuously adjusted in real time during the course of a journey. The Kalman filter is considered to be a valuable predictor for short-term prediction when using reasonable criteria. A Kalman filter can exploit available data such as speed, position, and measurement accuracy. Historical data should carry more weight than real-time data in long-term predictions.

The challenges inherent in the prediction of a vehicle's movement are several: (1) The movement of a vehicle is affected by factors that range from being quite predictable to being unpredictable, which makes prediction difficult and calls for continuous prediction. (2) Prediction techniques rely on identification of past journeys (or patterns among such journeys) that match the current journey. (3) It is attractive to be able to match historical journey patterns to an ongoing journey as early as possible.

3.4 Analysis of Historical Vehicle Journeys

One approach to the discovery of patterns in historical journey data is to apply clustering to these using, e.g., k-means clustering [13] and to perform statistical evaluations on the clusters [14]. This approach does not rely on prior knowledge of external factors that affect vehicle movement. Correlations between factors and journeys may be detected by means of the clustering—e.g., the journeys in one cluster may all have occurred during morning rush hour.

Another approach to obtaining patterns is by means of a variety of data mining techniques. Such techniques have been studied in the past, however, in different settings. For example, Mamoulis et al. [15] propose a framework for mining, indexing, and querying spatio-temporal data. They analyze trajectories of mobile objects in 2-dimensional space. Other existing approaches also focus on 2-dimensional space in mobile environments [16], or they focus on general time series databases [17, 18]. The vehicle trajectories discussed in this paper are constrained in the temporal and spatial dimensions: vehicles move on routes that are defined as sequences of road segments, and they have to follow time schedules with the best effort. This setting may allow us to simplify the existing algorithms, but may also pose higher requirements to the efficiency of the data mining techniques.

The position updates sent from the vehicles to the server are often the only position-related data available for subsequent, off-line analysis of historical journeys. These data and the associated position prediction functions that were employed during the original tracking should be utilized for representing the trajectories in a format that is well suited for subsequent analysis.

During the tracking of a vehicle, a new prediction function is created every time an update from the vehicle or from the surrounding infrastructure arrives. The function $h^i_{\rm pr} : T \rightarrow P$, $i = 0, ..., k + 1$, is the predicted trajectory after update up_{i-1} at time $t^{\rm up}_{i-1}$, and $h^0_{\rm pr}$ is the initial prediction. The predicted trajectory $h_{\rm pr} : T \rightarrow P$ is defined based on the predictions $h^i_{\rm pr}$:

$$h_{\rm pr}(t) = \begin{cases} h^0_{\rm pr}(t) & \text{if } t < t^{\rm up}_0 \\ h^i_{\rm pr}(t) & \text{if } t \in [t^{\rm up}_{i-1}, t^{\rm up}_i),\ i = 1, ..., k \\ h^{k+1}_{\rm pr}(t) & \text{if } t \geq t^{\rm up}_k \end{cases}$$

This function captures the predictions in effect during the journey. The challenge is then to create a continuous non-decreasing function, preferably with a compact representation, that is as close as possible to the actual trajectory, and it is never further than thr from the actual journey.

For example, such a function can be specified by a feature vector that, for each timing point on the route considered, contains the travel time from the previous timing point (points other than the timing points may also be used).

The analysis of historical journey data faces several challenges: (1) Patterns found in the data should represent the actual distributions of the trajectories, and the patterns should be updated by the system as more data becomes available. (3) The algorithms should efficiently utilize the spatial and temporal restrictions that apply to the scenario considered, and they should enable the analysis of large volumes of historical data. (4) Appropriate representations of vehicle trajectories must be recovered based on sparse tracking data.

3.5 Similarity Measures in Journey Patterns

During the data analysis of historical journeys, a notion of similarity between vehicle trajectories is needed that is efficient to compute.

An appropriate similarity function should satisfy at least these properties: (1) delays of equal duration should yield the same dissimilarity independently of when they occur during a journey, (2) matching of sub-journeys should be possible, (3) it should be robust to "dirty" data, and (4) it should be a metric (i.e., satisfy the identity, symmetry, and triangle inequality properties). Although analysis techniques can accommodate non-metric similarity functions, these adversely impact efficiency.

Dynamic Time Warping (DTW) [19] is a commonly used similarity measure for time series. But this measure is computationally expensive. Further the stretching and shifting allowed may not be appropriate in our setting. A simple measure that (to some degree) satisfies our requirements is the (squared) Euclidean distance. The distance between two journeys is then the sum of the squared differences between the pairs of travel times in the two feature vectors that represent them. Various weights may also be introduced.

An appropriate similarity function must possess the following qualities: (1) it must enable the classification of journeys in an intuitive manner, (2) it must be computationally efficient, (3) it must enable the identification of correspondences between influencing factors (e.g., accidents, weather conditions) and the trajectories, and (4) it must enable the matching of an ongoing journey against historical journeys.

4 Conclusions

This paper describes the problem of tracking and prediction in the context of scheduled-vehicle journeys, and it identifies key challenges. The focus of past work has been on the tracking of the positions of vehicles traveling along unknown routes. Although the tracking of scheduled-vehicle journeys has the potential for being more efficient than the tracking of vehicles that travel along unknown routes, the requirements are often also higher. The same holds for the accuracy of travel-time prediction.

The key tracking and prediction challenge can be formulated in two ways: (i) given a limited budged for the tracking cost, achieve the maximum accuracy of the observed and anticipated system status; (ii) given requirements for the accuracy, achieve these requirements with the minimum costs. Both variants aim to achieve the maximum gain at the minimum cost. The primary challenges associated with these can be summarized as follows: (1) prediction algorithms must contend with the influence of external factors that are only predictable to varying degrees, (2) update policies must be optimized, taking into account the problem domain, and (3) all system parameters must be maintained and updated dynamically by the system itself in response to the current system behavior.

References

[1] "Nordjyllands Trafikselskab (NT)," http://www.nordjyllandstrafikselskab.dk/.

[2] A. Čivilis, C. S. Jensen, J. Nenortaitė, and S. Pakalnis, "Efficient tracking of moving objects with precision guarantees," in *MOBIQUITOUS*, 2004, pp. 164–173.

[3] A. Čivilis, C. S. Jensen, and S. Pakalnis, "Techniques for efficient road-network-based tracking of moving objects," *IEEE TKDE*, 17(5):698–712, 2005.

[4] O. Wolfson, S. Chamberlain, S. Dao, L. Jiang, and G. Mendez, "Cost and imprecision in modeling the position of moving objects," in *ICDE*, 1998, pp. 588–596.

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

[6] O. Wolfson and H. Yin, "Accuracy and resource consumption in tracking moving objects," in *SSTD*, 2003, pp. 325–343.

[7] F. Cathey and D. Dailey, "A prescription for transit arrival/departure prediction using automatic vehicle location data," *Transp. Res. C*, pp. 241–264, 2003.

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

[9] A. Shalaby and A. Farhan, "Prediction model of bus arrival and departure times using AVL and APC data," *Journal of Pub. Transp.*, 7(1):41–61, 2004.

[10] S. I.-J. Chien, Y. Ding, and C. Wei, "Dynamic bus arrival time prediction with artificial neural networks," *Transp. Engrg.*, 128:429–438, 2002.

[11] J. R. Hee and L. R. Rilett, "Bus arrival time prediction using artificial neural network model," in *IEEE ITSC*, 2004, pp. 988–993.

[12] T. Park, S. Lee, and Y.-J. Moon, "Real time estimation of bus arrival time under mobile environment," in *ICCSA*, 3043:1088–1096, 2004.

[13] T. Kanungo, D. M. Mount, N. S. Netanyahu, C. D. Piatko, R. Silverman, and A. Y. Wu, "An efficient k-means clustering algorithm: Analysis and implementation," *IEEE TPAMI*,24(7):881–892, 2002.

[14] R. Tibshirani, G. Walther, and T. Hastie, "Estimating the number of clusters in a dataset via the gap statistic," *J. Roy. Statist. Soc. B*, vol. 63, pp. 411–423, 2001.

[15] N. Mamoulis, H. Cao, G. Kollios, M. Hadjieleftheriou, Y. Tao, and D. W. Cheung, "Mining, indexing, and querying historical spatiotemporal data," in *ACM SIGKDD*, 2004, pp. 236–245.

[16] W.-C. Peng and M.-S. Chen, "Mining user moving patterns for personal data allocation in a mobile computing system," in *ICPP*, 2000, pp. 573–580.

[17] J. Han, Y. Yin, and G. Dong, "Efficient mining of partial periodic patterns in time series database," in *ICDE*, pp. 106–115, 1999.

[18] J. Yang, W. Wang, and P. S. Yu, "Mining asynchronous periodic patterns in time series data," *IEEE TKDE*, 15(3):613–628, 2003.

[19] "Dynamic Time Warping (DTW)," http://en.wikipedia.org/wiki/Dynamic_time_warping.