Citation: Salamanis, A., Kehagias, D.D., Filelis-Papadopoulos, C.K., Tzovaras, D., Gravvanis, G.A. "Managing Spatial Graph Dependencies in Large Volumes of Traffic Data for Travel-Time Prediction," in Intelligent Transportation Systems, IEEE Transactions on (accepted for publication), doi: 10.1109/TITS.2015.2488593.

Managing Spatial Graph Dependencies in Large Volumes of Traffic Data for Travel Time Prediction

Athanasios Salamanis[1], Dionysios D. Kehagias[1], Christos K. Filelis-Papadopoulos[2], Dimitrios Tzovaras[1], *Senior Member, IEEE* and George A. Gravvanis[2]

***Abstract*—The exploration of the potential correlations of traffic conditions between roads in large urban networks, which is of profound importance for achieving accurate traffic prediction, often implies high computational complexity due to the implicated network topology. Hence, focal methods are required for dealing with the urban network complexity, reducing the performance requirements that are associated to the classical network search techniques (e.g. Breadth First Search). This paper introduces a graph theory-based technique for managing spatial dependencies between roads of the same network. In particular, after representing the traffic network as a graph, the local neighbors of each road are extracted using Breadth First Search graph traversal algorithm, and also a lower complexity variant of it. A Pearson product-moment correlation coefficient based metric is applied on the selected graph nodes for a prescribed number of level sets of neighbors. In order to evaluate the impact of the new method to the traffic prediction accuracy achieved, the most correlated roads are used to build a STARIMA model taking also into account the possible time delays of traffic conditions between the inter-related roads. The proposed technique is benchmarked using traffic data from two different cities: Berlin (Germany) and Thessaloniki (Greece). Benchmark results indicate not only significant improvement on the computational time required for calculating traffic correlation metric values, but also reveal that a different variant works better in different network topologies, after comparison to third party approaches.**

Index Terms—**Data processing, Graph theory, Time series analysis, Travel time prediction.**

[1] Athanasios Salamanis, Dionysios D. Kehagias, and Dimitrios Tzovaras are with Center for Research & Technology Hellas, Information Technologies Institute, P.O. Box 60361, GR 57001 Thermi-Thessaloniki, Greece, (e-mail: asal@iti.gr, diok@iti.gr, dimitrios.tzovaras@iti.gr).

[2] Christos K. Filelis–Papadopoulos, and George A. Gravvanis are both with the Department of Electrical and Computer Engineering, School of Engineering, Democritus University of Thrace, University Campus, Kimmeria, GR 67100, Xanthi, Greece, (e-mail: cpapad@ee.duth.gr, ggravvan@ee.duth.gr) .

I. Introduction

Nowadays optimizing the quality of transportation either for individuals or fleets of vehicles has emerged as an important need with significant social, economical and ecological impact, hence interest in developing intelligent transportation systems has grown significantly in respect to this. Predicting traffic, in terms of travel time, within multiple intervals ahead in urban networks is a challenging task with clear benefits, especially on the improvement of the accuracy of traveller information systems.

In order to quantify traffic and apply travel time prediction techniques in a transportation network, appropriate measurable descriptors or combinations of them are used as input data. Common traffic descriptors include traffic flow, density, occupancy etc., according to the source of input data, e.g. loop detectors provide occupancy measurements, whereas GPS-enabled devices provide speeds measurements, which are the traffic descriptors adopted in this paper.

Based on previous evidence [1], it is clear that the traffic volume of a particular road is affected by traffic in neighboring roads in the same network. Hence the discovery of potential dependencies and subsequent correlations in terms of traffic related parameters among roads in the same network can result in better prediction accuracy. As urban networks are expanding, the size of the available and constantly monitored traffic data is growing. Hence, the calculation of potential correlations within data becomes computationally expensive. To this end, new techniques for sufficiently managing large volumes of available traffic data in reasonable time are required for enabling efficient travel time prediction capabilities.

This paper presents a travel time prediction technique that deploys a novel algorithm for the reduction of the required time and the associated computational resources for the calculation of traffic dependencies in large volumes of traffic data. The approach is based on graph theory principles. In order to discover the aforementioned dependencies, the Coefficient of Determination (CoD) metric is computed, but only for a prescribed number of level-sets of neighbors. By applying a modified version of the Breadth First Search algorithm, not only the required time for computing the CoD is sufficiently reduced, but also the prediction accuracy is improved.

In brief, the main contributions of this paper are:

- The development of a new technique for the reduction of the time required to compute traffic correlations, between roads in big urban networks, by means of CoD.
- The improvement of travel time prediction accuracy leveraging on the previous technique and the deployment of a novel time series spatiotemporal model.

The rest of this paper is organized as follows. Section II reviews current research results associated with our work, whereas Section III gives a detailed description of the implemented model. Section IV presents the evaluation framework, introducing the two large volume datasets used for deploying and testing our model. The same section presents the benchmark traffic prediction algorithms, that were used for comparison to our model and presents the experimental results both in terms of prediction accura-

cy and computational complexity. Finally, Section V concludes the paper, reviewing the main contributions and suggesting future directions.

II. Related Work

The profound importance of travel time (or traffic) prediction on the design of intelligent transportation systems has lead to the development of various techniques that can be basically classified into the following four major categories: *naïve*, *parametric*, *non-parametric* and *hybrid*.

Naïve methods are characterized by the absence of any advanced mathematical model for the exploitation of traffic data. They are selected because of their minimum computational overhead and their ease of deployment, but they have significant drawbacks with respect to the achieved accuracy. The most common approach in this category is the computation of the historical average (HA).

Parametric techniques are based on specific models whose general structure has been defined in advance and only the exact values of a given set of parameters need to be determined. The learning procedure, through which these parameters are set is implemented heuristically on the basis of the available data that refer to the historical behavior of the system. The most common methods of this category are derived from time series analysis, i.e., Auto-Regressive Moving Average (ARMA), Auto-Regressive Integrated Moving Average (ARIMA) and Space-Time Auto-Regressive Integrated Moving Average (STARIMA). Stathopoulos and Karlaftis [2] developed a parametric state-space approach in order to model traffic flow whereas, for the same purpose, Kamarianakis and Prastacos [3] developed a STARIMA model with robust behaviour. An extension to this system is described by Min and Wynter [4] who target at dealing with various shortcomings of the classical STARIMA, such as the supposed stationarity of the system and the constant relationship among the neighboring links. The travel time prediction approach presented in this paper also falls into this category.

On the other hand, the term ‘non-parametric’ is not used in order to indicate the absence of parameters, but rather to underline the fact that this category of travel time prediction techniques does not presuppose a particular model structure. Hence, both the exact model structure and its parameters need to be specified during the training phase of the model. Non-parametric techniques can be classified into two broad categories: model-based and memory-based. Model-based methods exploit the available historical data only during the training procedure, in order to build the model and define its parameters. After the model is built no additional use of historical data is necessary. Common examples of model-based prediction techniques include Artificial Neural Networks (ANNs) [5], Random Forests (RFs) and Support Vector Machines (SVMs). Vlahogianni *et al.* [6] introduced the auto- and cross-correlated effect of traffic flow series in a neural network model [2] in the form of external information. The same approach was also adopted by various researchers that produced multi-step ahead predictions, like [7], [8] and [9] with acceptable levels of accuracy. Memory-based methods require retaining a database of historical samples, since besides training, the same

data is used for formulating the estimation of the model's future states. The most prominent memory-based prediction method is non-parametric regression mainly represented by the *k*-Nearest Neighbor (kNN) technique. Smith and Demetsky [10] introduced the kNN technique and proved that it outperforms both the naïve method of historical average and the parametric ARIMA model in terms of robustness against variable datasets. However, Williams *et al.* [11] showed that the results obtained through Seasonal ARIMA modelling exceed the precision of the *k*-Nearest Neighbor technique.

An important subset of traffic prediction techniques that has been expanded in the last years is the one that includes those algorithms that explore and exploit spatiotemporal correlations between roads of an urban network in order to increase prediction accuracy. In this context, Asif et al. [12] proposed a methodology for mining the spatial and temporal prediction performance trends in large scale urban networks by using unsupervised learning techniques such as k-means clustering, Principal Component Analysis (PCA), and Self Organizing Maps (SOM). Similarly Mitrovic et al. [13] use matrix decomposition techniques to exploit correlations among roads. Initially using a Singular Value Decomposition (SVD) based technique [14] a subset of road segments from a traffic network is created. Using the training traffic dataset of the selected roads and that of the whole network a relationship matrix is constructed and Support Vector Regression (SVR) models for the selected roads can be trained. At the evaluation step, travel time prediction is enabled only for the selected roads using the aforementioned SVR models, and the results are extrapolated to the whole network using the prefabricated relationship matrix. Finally, Kamarianakis et al. [15] propose a regime-switching space-time parametric approach, which leverages the Least Absolute Shrinkage and Selection Operator (LASSO), not only for solving an over-determined system of equations, but most importantly, for selecting the features of the model, i.e., those neighboring roads to select for predicting travel time of a specific road.

Finally, noticeable research effort has been dedicated on the development of hybrid techniques, which exploit the advantages of both parametric and non-parametric ones. In this respect, Zhang [16] proposed a methodology that combines ARIMA and ANN processes. In brief, an ARIMA model is initially applied, in order to analyze the system's linear constituents and subsequently an ANN is deployed to model the residuals from the ARIMA model, which are expected to contain all the information of traffic data non-linearity. Eventually, the feedback from the ANN is provided as input for defining the error terms of the ARIMA model. Furthermore, methods like [17] which combines fuzzy logic and ANN in order to build a Non-linear Autoregressive Moving Average with exogenous inputs (NARMAX) model or Quek's *et al.* [18] Fuzzy ANN also seem promising.

A key aspect of our work is the exploitation of graph theory tools and techniques for dealing with the computational complexity of traffic correlations exploration problem. Graphs serve as a mathematical model for network modelling and are used in many fields. As Easley and Kleinberg [19] pointed out, graphs can be used in order to model networks from several different fields such as: (i) communication networks, where nodes represent computers and edges direct links, through which messages can be transmitted, (ii) social networks, where nodes represent people and edges social interactions between them, and so on. Among

other application domains, graph theory is used in transportation as well, where according to [20] a graph is called *transportation network* when it is used to model the transportation of a commodity from one place to another. According to [21] there are mainly two graph representations of an urban structure: the primary and the dual. The former is based on the relations between junctions through their streets, where every node represents a junction and every edge a street or route between junctions. The latter describes the relationships between the roads of the network, in which every node represents a street and every edge a common point between any two streets. Both representations are used for analyzing complex urban networks [22][23].

III. Our Method

A. *Traffic correlations*

The discovery of traffic correlations between roads in an urban network can result in better prediction accuracy. Usually, the correlations between roads are estimated in terms of proximity. In this context, a distance metric, e.g. the Euclidean distance, is used rendering the most correlated roads for a road of interest among those, which are closer to it. This concept is expanded in [1], where the correlations are estimated in terms of traffic status, rather than proximity. For example, the high traffic of a highway at current time can affect the traffic of other roads that do not abut to it, in the future. Such correlations are explored using a Pearson product-moment correlation coefficient-based metric known as *Coefficient of Determination (CoD)*. The latter is applied between any pair of roads in the network resulting in "neighbors" that are not necessarily geographically close to each other, but share similar traffic patterns. CoD describes the correlation between the present (future) values of the traffic time series of a road of interest, and the past (present) values of the series of another road within the network. The *CoD* metric between two time series *x* and *y* at lag *k* is defined by the following equation:

$$CoD_{xy}(k) = 100\left\{\frac{E\left[(x_t - \mu_x)(y_{t+k} - \mu_y)\right]}{\sigma_x \sigma_y}\right\}^2, k = 0, \pm 1, \ldots \quad (1)$$

CoD is computed for all pairs of roads *(i, j)* of the network (except of *(i,i)*). This process is extremely computationally expensive for major urban networks with thousands of road segments. For example, for the downtown area of a medium sized urban traffic network with a few thousands of road segments and for one day period, the CoD computations may need more than 30 hours to be completed on a dual core Intel i5 CPU at 3.33GHz with 4GB RAM. Although the process runs offline, the computational time is prohibitive considering that we use traffic time series that correspond to only one day of the whole traffic dataset and exploring only the central area of a large urban network.

B. Graph representation

In order to deal with the computational complexity of the CoD computations, namely to reduce the computational time, the following principle is applied.

Since every road of the network is either one way or bidirectional, the *in* and *out* roads for a road of interest are defined. The *in* roads are those that end to the start or end point of the road of interest, whereas the *out* roads are those that start from these points. The *in* roads import traffic at the road of interest, whereas the *out* roads export traffic (Fig. 1). Based on this concept the urban network is represented as a geometric directed graph, in which every node represents a road and every edge a straight *in* or *out* connection. In particular, there is an edge between two nodes A, B if and only if B represents an *in* or *out* road of A (and vice versa).

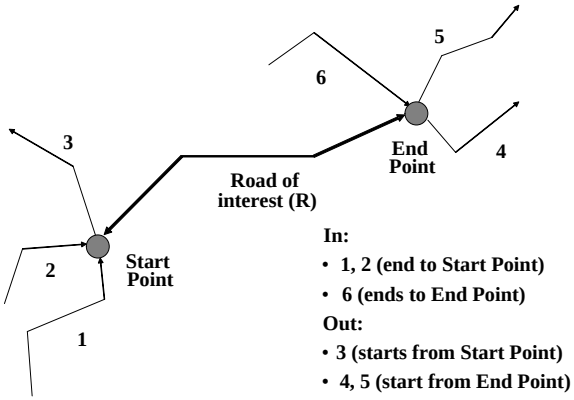

Fig. 1. In and out roads

The adjacency matrix *A* of the graph is given in Table I. The non-zero elements in row *i* of the adjacency matrix represent the *out* connections to a road *i* and those in the respective column the *in* connections. It should be noted that roads that allow traffic in both directions are considered both *in* and *out*. Consequently, the corresponding adjacency matrix of the graph represents all the first order connections (neighbors) between roads in the network.

TABLE I

ADJACENCY MATRIX OF THE GRAPH

			X		X
X		X	X		
X	X		X		
X	X	X		X	
			X		
		X			

The neighbors of a road (vertex) *i* at distance *d (hops)* can be found by accumulating all the neighboring vertices at distance *d*. These neighbors define an *adjacency class* which is by one smaller than distance *d* (*adjacency class* = *d*-1). It should be noted that adjacency class 0 (distance 1) for a specific vertex consists only of the vertex itself. The level sets of neighboring vertices are closely related to vertex *i*. Thus, CoD can be computed only for a prescribed number of level sets of neighbors, which are at maximum distance *d* from the vertex of interest. The neighbors at distance *d* can be computed by using the *in* roads, the *out* roads, or both, thus taking into account only the roads that redirect traffic to the road of interest, or the traffic is redirected to them from the road of interest, or both, respectively. The computation of the neighboring nodes is performed using a variant of the Breadth First Search (BFS) graph traversal algorithm (Fig. 3), which accumulates level sets of vertices up to the prescribed number. It should be noted that the BFS algorithm is one of the two most frequently used graph traversal algorithms in the literature due to its effectiveness. The other most used algorithm, i.e. Depth First Search (DFS), is not useful in this case because it searches the graph in depth and does not create a neighborhood of nodes as needed, but consecutive in depth connections of nodes. The process of accumulating neighboring vertices is schematically presented in Fig. 2.

The *Modified Breadth First Search (MBFS)* algorithm accumulates the neighbors with respect to *out* rows. In order to accumulate the neighboring vertices with respect to the *in* roads, the transpose adjacency matrix A^T is used. In the case where both *in* and *out* roads are needed the $A+A^T$ is used as the adjacency matrix. The adjacency matrix $A+A^T$ is not explicitly computed and stored, in order to reduce memory requirements, instead the rows of A and A^T are used to determine all the neighbors of a vertex. Thus, for the case of $A+A^T$ adjacency matrix, step 6 should be replaced by $B = B \cup \{A_v\} \cup \{A_v^T\}$.

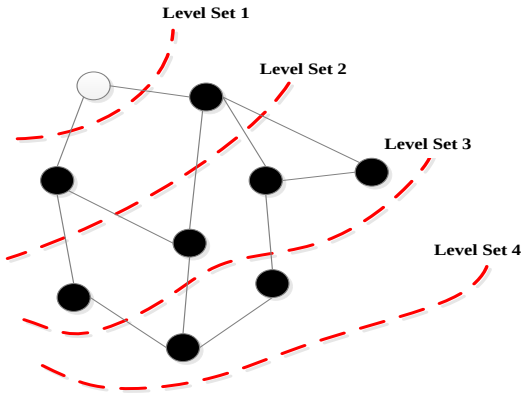

Fig. 2. Neighboring vertices accumulation

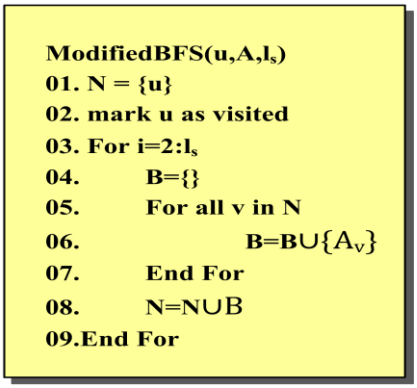

Fig. 3. MBFS algorithm

In step 6 of the MBFS algorithm where the union of sets is required, duplicate entries must be avoided. Thus, instead of accumulating all neighbors and sorting out the duplicates and the already visited vertices, a structure containing a dense vector and a list is used. The vector is of length equal to the total number of vertices and is initialized to zero. The list retains the positions of the non-zero elements in the vector. The candidate vertices to join the set *B* are only accepted if their respective position in the vector is zero and they have not been accumulated in *N* (step 8) during a previous iteration. Thus the *N* set is also implemented using the same strategy. When the algorithm finishes, the neighboring vertices are copied to a vector with length equal to the number of neighbors corresponding to the l_s level sets of the vertex *u*. As mentioned before, the neighboring vertices of a vertex *u* at distance one is the vertex *u* itself.

The computational complexity of the MBFS algorithm can be defined as $O(N\cdot(nnz(A)/N)^{Ap})$, where *N* is the number of nodes in the graph, *nnz(A)* is the number of non-zero elements of the adjacency matrix *A* of the graph and A_p the adjacency class until which neighbors are selected. This is $O(aN)$, $a > 1$, which is significantly improved compared to the naïve $O(N^2)$.

The memory requirements of the algorithm are limited to two dense vectors of order equal to the total number of vertices and two lists. The adjacency matrices are stored in the Matrix Market format [24]. Moreover, another double vector with size equal to the number of non-zero elements in the adjacency matrix is stored for future use, i.e. add weights to each road. Furthermore, since the adjacency matrices are sparse, the memory storage format used was the Compressed Sparse Row, which requires three vectors in order to store a sparse matrix [25].

C. Graph based Lag-STARIMA

The presented travel time prediction algorithm, namely *Graph based Lag STARIMA – GBLS*, is based on the classic parametric traffic prediction model STARIMA. The latter is based on *ARIMA(p, d, q)* model that was first introduced in [26] to predict the next values of a time series given its past values. The model takes into account the spatiotemporal relations between time series and is defined by (2):

$$\varphi_{p,\lambda}(B)\Phi_{P,\Lambda}(B^S)(1-B)^d(1-B^S)^D X_t = \theta_{Q,M}(B^S)e_t \tag{2}$$

where:

$$\begin{aligned}
\varphi_{p,\lambda}(B) &= 1-\sum_{k=1}^{p}\sum_{l=0}^{\lambda^k}\varphi_{k,l}W_l B^k \\
\Phi_{P,\Lambda}(B^S) &= 1-\sum_{k=1}^{P}\sum_{l=0}^{\Lambda^k}\Phi_{k,l}W_l B^{kS} \\
\theta_{q,m}(B) &= 1-\sum_{k=l}^{q}\sum_{l=0}^{m^k}\theta_{k,l}W_l B^k \\
\Theta_{Q,M}(B^S) &= 1-\sum_{k=1}^{Q}\sum_{l=0}^{M^k}\Theta_{k,l}W_l B^{kS}
\end{aligned} \tag{3}$$

The parameters of the STARIMA model take into account the spatial and temporal lags of the multiple time series. Hence, k and l denote the temporal and spatial lag respectively, while $\varphi_{k,l}$ and $\theta_{k.l}$ are the auto-regressive and moving average non-seasonal parameters. The neighboring matrix W is an $N \times N$ square matrix that contains the weights that generally define the *spatial* relationship of the terms. For instance, the weights w_{ij}, where i and j refer to two separate roads, might represent such metrics as the number of roads between i and j, the amount of public transportation available connecting the two roads, and even the flow rates upon these avenues. In the presented case, the W matrix is filled with values representing the *traffic* relationships between roads based on the CoD metric. Given two roads i and j, each element of the weight matrix W in the i-th row and j-th column is computed as follows:

$$w_{ij}^{\,l} = \begin{cases} \dfrac{1}{\sum_{j=1}^{N} w_{ij}}, & \text{if } \mathrm{CoD}_{ij}(l) \in M_l \\ 0, & \text{otherwise} \end{cases} \tag{4}$$

where M_l is the set with the largest CoDs for lag l. As in [1], two lag values were considered (for two time intervals before the current one, respectively) and the ten largest values of CoD for a specific lag are taken into account. Hence, the implemented model is formed as follows:

$$Z_{t+T} = \phi_{00}Z_t + \phi_{10}Z_{t-1} + \phi_{20}Z_{t-2} + \phi_{11}W_1Z_t + \phi_{12}W_2Z_t + \ldots \tag{5}$$

where Z_{t+l} ($l < 0$) represents speed with lag l, W_l is the traffic neighbor matrix of lag l and φ_{tl} are the parameters for roads of lag l at moment t.

The above model takes into account the current speed of the road, the speeds of the past two time intervals and the average speed of the most correlated roads that have a maximum time lag of 10 minutes. Naturally, using a training set it is attempted to approximate the φ_{ij} parameters and then to test the method by predicting the next Z_t values on a given test set. As noted in [27] the best estimates of the φ values, from many points of view, are the maximum likelihood estimates but since without *a priori* knowledge of their initial values, they cannot be exactly computed. Hence a close approximation of them is achieved via least

squares. In particular, for every training sample an equation like the one given by (5) is formed where φ_{ij} are the only unknown parameters. This leads to an over-determined equation system, in the form of $X\beta = y$, or written with the normal equations, as:

$$(X^T X)\hat{\beta} = X^T y \tag{6}$$

Using the linear least squares method, (6) has a solution in the form of (7).

$$\hat{\beta} = (X^T X)^{-1} X^T y \tag{7}$$

Since STARIMA models are non-linear in form, it is necessary to estimate the parameters of the model using any non-linear optimization technique. As a result, the Levenberg Marquardt (LM) algorithm [28] (also known as Damped Least Squares method) was used in order to solve the equation as a non-linear least square problem. The LM method actually solves a slight variation of (6), known as the augmented normal equation described by (8):

$$N\hat{\beta} = X^T y \tag{8}$$

,

where the off-diagonal elements of N are identical to the corresponding elements of $X^T X$ and the diagonal elements are given by (9):

$$N_{ii} = m + [X^T X]_{ii}, \quad m > 0 \tag{9}$$

The term m is called *damping term* and the rest of the LM algorithm, which was introduced in [29], remains unchanged.

Noticeably, the proposed method has inherent parallelism because the problem of computing the prediction for the traffic can be split into N independent sub-problems. Each one of them can be processed by a different CPU without synchronization points or intermediate communications, thus rendering the method suitable for parallel systems. At this point, this characteristic is not used at the computations and remains as future work.

IV. Evaluation

A. Traffic data

1) Available Datasets

In order to evaluate the prediction accuracy and the performance of the proposed scheme two datasets were used, thereafter referred as the Berlin dataset and the Thessaloniki dataset, with respect to the two cities. The reason for using datasets from two different cities was the need for exploring the behavior of the implemented model in two cities with different size and architec-

tural designs.

The Berlin dataset contains speed probes from GPS devices covering total period of two weeks, i.e. from March 18, 2012, to March 31, 2012, from the city of Berlin, Germany. The instantaneous speeds contained in the dataset were already matched to road segments before provided to us.

The Thessaloniki dataset used for the needs of this paper contains speed probes from GPS devices installed on taxis from the city of Thessaloniki, Greece and covers a total period of two months, i.e. from Feb 1, 2014 to Mar 31, 2014. It contains instantaneous speeds of each vehicle that were unmatched.

Both datasets provide a small portion of the data that can be available for serving the needs of an online traffic prediction system. Data production rates fall in the range of approximately 10kB/sec and 3kB/sec, for Berlin and Thessaloniki, respectively, resulting in several TBs, e.g., during one year data gathering cycle.

2) Pre-processing

Applying a prediction algorithm on those datasets is restrictive not only due to scalability issues, but also because of zero values, e.g. a road segment may have no data at different time instances. Thus a new 'higher level' interpretation of the network is necessary. To this end, the road segments were firstly combined to form roads. Each road was defined as a segment between two intersections, whereas road segments were defined as straight lines, thus a road contains an arbitrary number of road segments. In this way the 314,278 road segments of the Berlin network were transformed into 218,576 roads and the 91,926 road segments of the Thessaloniki network into 68,451 roads. The various characteristics of the two networks are shown in Table II.

As already mentioned, the Thessaloniki dataset consists of instantaneous speeds of vehicles at given coordinates, i.e. in an inappropriate format for travel time prediction. Therefore a process of matching instantaneous speed values to road segments on the network's map was required. The map-matching procedure that was used, is the one proposed in [30].

TABLE II

NETWORKS TOPOLOGICAL CHARACTERISTICS

Network characteristics		Berlin	Thessaloniki
1	Nodes	152010	40672
2	Road segments	314278	91926
3	Roads	218576	68451
4	Min. road segment length (km)	0.001	0.002
5	Max. road segment length (km)	5.777	19.525
6	Average road segment length (km)	0.097	0.l29
7	Min. road length (km)	0.001	0.002
8	Max. road length (km)	9.129	28.557
9	Average road length (km)	0.140	0.173
10	Min. road segments per road	1	1
11	Max. road segments per road	44	48
12	Average road segments per road	1.4364	1.34286

As a result of the previous pre-processing step both datasets contain individual records of instantaneous speeds per road segment and time instance. These records were aggregated into harmonic averages of 5-minute time intervals per road. Supposing n raw speed values recorded at a time interval t, the harmonic average $x_{i,t}$ of a road i at interval t given by (10) is preferred over the ordinary average, because it eliminates outliers in the form of too high speeds, thus resulting in more accurate prediction of travel times [31].

$$x_{i,t} = \frac{n}{\sum_{j=1}^{n} \frac{1}{x_j}} \tag{10}$$

Finally, for every road of the network a traffic time series was formulated, representing a day of traffic data for the specific road. In particular, every traffic time series consisted of harmonic average speeds for every 5-minute interval of a day, hence its size was $24 \cdot 60 / 12 = 288$.

B. Benchmark techniques

We compared against the following five benchmark techniques: (1) Historical Average (HA), (2) k-Nearest Neighbor (kNN), (3) Random Forests (RF), (4) Support Vector Regression (SVR), (5) Compressed Support Vector Regression (CSVR). All the benchmark methods are briefly described below.

1) Historical Average

As mentioned at Section II, HA is a naïve prediction method that is characterized by the absence of any advanced mathematical model and is usually used as a reference method. Given the traffic time series of a road of interest, the estimation of the *N+1* value of the time series (at interval *t+1*) is the average of its *N* previous values, i.e., from the beginning, up to the current interval *t*.

2) k-Nearest Neighbor

Predicting the next value of a traffic time series using kNN is based on the current state vector (a time interval *t*), which is defined as $x(t) = [V(t), V(t\text{-}1), V(t\text{-}2), V_h(t), V_h(t+1)]$, where $V(t)$ is the traffic descriptor (in this case speed) at time *t*, and $V_h(t)$ is the historical value of *V* at time *t*. The output is defined as $y(t)=V(t+l)$, where *l* is the number of intervals to predict ahead. The algorithm creates input vectors $x(t_1), \ldots, x(t_N)$ using the training set. When a prediction is requested, the *k* nearest vectors to the current vector (i.e. the vector x_p at present time where y_p is unknown) are calculated and an averaging technique is used to calculate the corresponding output value (y_p). In the experiments the value of *k* was set to 4.

3) Random Forests

An RF-based algorithm was implemented for our benchmark purposes, in order to predict the harmonic average of speed for a specific road and arbitrary time interval ahead of the current time. Each random forest corresponds to a road. Features from global statistics of the network for a 30-minute interval, such as the number of non-zero samples, the number of zero samples and the arithmetic mean of speeds were defined. A *Principal Component Analysis (PCA)* was applied to these features in order to isolate the most important of them, which in turn were given as global inputs to the algorithm. The inputs of the road to be predicted consist of these features, as well as the harmonic average of the road intervals beyond the 30-minute window. The latter were used as target output data in order to train the RF. Concerning both datasets, the number of trees per forest was set to 100, since it proved optimal for the generalization ability of the algorithm.

4) Support Vector Regression

SVR is the variant of the classic machine learning classification technique Support Vector Machines (SVM) for regression problems and is based on a straightforward idea: construct a hyperplane that sets apart the classes of data. Due to its high accuracy if trained with satisfactorily large dataset, SVR has been used for travel time prediction in various works such in [12].

5) Compressed Support Vector Regression

This technique was proposed in [13], and is based on the idea of training and running SVR models using the data from a subset of the road segments of a traffic network and then extrapolating the results to the whole network. At the training phase, the roads segments that will be used are chosen using a CUR decomposition technique [14] and the relationship matrix between the subset of road segments and the whole network is constructed. Also the method constructs SVR models for the chosen road seg-

ments. Then at the evaluation phase, the travel time is predicted for the selected road segments using the aforementioned SVR models, and the results are extrapolated to road segments of the whole network.

C. Evaluation framework

1) Setup of the evaluation framework

Concerning the Berlin dataset, training and testing of the algorithms were conducted using traffic data from one day of the first week, and the corresponding day of the next week, respectively, e.g. trained for Monday 19 March, 2012 and tested with data from Monday 26 March, 2012. In a similar manner, for the Thessaloniki dataset, the algorithms were trained and tested with data from one day of the first month and the corresponding day of the next month, respectively, e.g. trained with data from Friday 7 February, 2014 and tested with data from Friday 7 March, 2014. The prediction period includes up to twelve 5-minute intervals, i.e. 1 hour, beyond present time. The traffic descriptors used for quantifying traffic are the harmonic speed of an interval τ and the corresponding travel time (in min/km). In particular, the speed $S_{\tau,r}$ of a road *r* in a time interval *τ* is predicted, and the generated value is transformed into travel time $T_{\tau,r}$ ($T_{\tau,r} = 60 / S_{\tau,r}$).

The metric used for evaluating the performance of the implemented algorithms is the *Root Mean Square Error (RMSE)*, which was selected over other metrics, such as the *Mean Average Precision Error (MAPE)*, because of its robustness to near zero values. The RMSE for a specific prediction interval *τ* (e.g. 5-minute ahead prediction) for all the roads of the network is given by (11):

$$RMSE(\tau) = \sqrt{\frac{1}{N}\sum_{r=1}^{N}(T_{\tau,r} - T'_{\tau,r})^2} \tag{11}$$

where $T_{\tau,r}$ and $T'_{\tau,r}$ are the real and the predicted travel time values, respectively, for road *r* at prediction interval *τ*. The *average RMSE* is the mean RMSE value for all prediction steps and all roads of the network.

2) Selection of the optimal adjacency class

An interesting characteristic of the proposed method is its behavior with respect to the adjacency class, i.e. the maxi-mum *d* in hops that includes all neighboring roads that are taken into account for the calculation of CoD. The optimal adjacency class that should be used for the Berlin and Thessaloniki datasets was determined according to the following routine. We perform consecutive training and testing sessions using various CoD values that correspond to increased values of the adjacency class, i.e. from 1 to 20, in an iterative fashion. For each value of the adjacency class, we calculate the total computational time for CoD (Fig. 4) and the resultant average RMSE value (Fig. 5). Note that the numerical results presented at this section (and also at Section IV.C.3 and Section IV.C.4) refer to experiments for which no concern about the handling of missing data was taken into account, and so the predictions refer to all the roads of the two networks.

As shown in Fig. 4, and according to our expectation, the CoD computation time increases as the value of adjacency class increases too, because a larger number of computations are required for the calculation of CoD as more neighboring roads are taken into account.

An interesting case is revealed in Fig. 5. For the Berlin dataset, the average RMSE increases with the adjacency class. This means that as the number of neighbors that are taken into account for the computation of CoD increases, the quality of the prediction is harmed. On the other hand, concerning the Thessaloniki dataset, the average RMSE is reduced as the adjacency class increases. This implies that taking into account the traffic information of long distance roads, improves the quality of the prediction. This opposition between the two cities indicates different traffic patterns, which could be associated to their different urban architecture styles and topologies. The criteria for the selection of the optimal adjacency class are related to the achieved prediction accuracy that should be statistically better compared to the best of the benchmark methods, and to the complexity of the CoD calculation that should be managed in reasonable time.

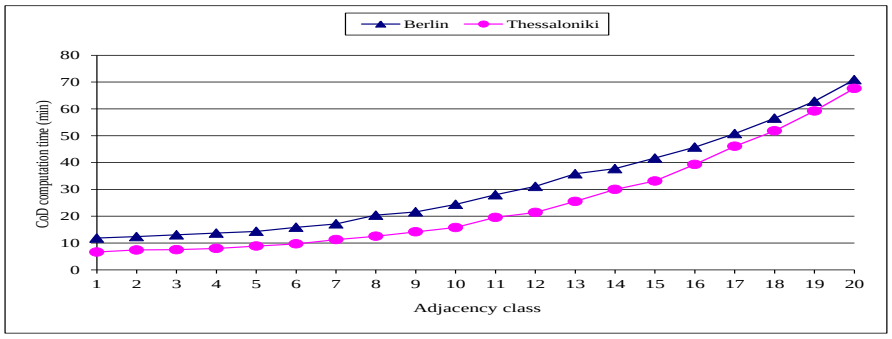

Fig. 4. CoD computation time versus adjacency class

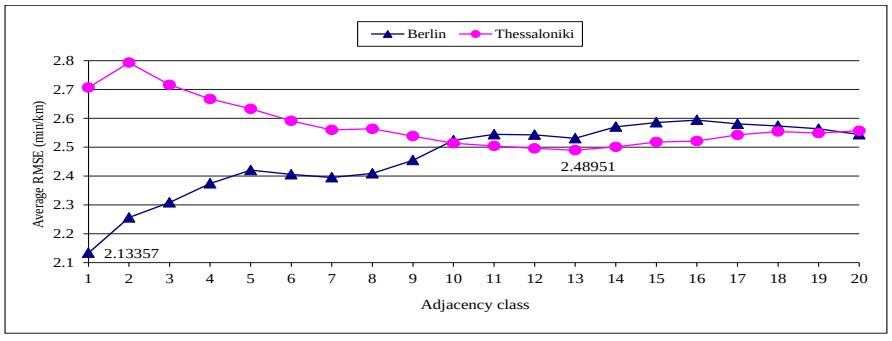

Fig. 5. Average RMSE versus adjacency class

Concerning the Berlin dataset, the benchmark method with the best prediction accuracy is the CSVR algorithm with average RMSE value of 2.55597 min/km. As shown in Fig. 5, the lowest average RMSE value (that corresponds to the optimal prediction accuracy) for the GBLS algorithm is the one that is derived for adjacency class 1 and its value is 2.13357 min/km (< 2.55597). In

order to make sure that the selected adjacency class leads to a better prediction accuracy which is statistically significantly different compared to the one of CSVR, for all prediction steps, the *Wilcoxon signed-rank test* was applied. This test showed that there is statistically significant difference between the compared accuracy values. Moreover, the required time for computing the CoD for the adjacency class 1 is 11.842 minutes, which is considered acceptable. For the aforementioned reasons the adjacency class of 1 was chosen as the optimal one for the Berlin dataset.

Regarding the Thessaloniki dataset the same approach was adopted. Again CSVR is the benchmark method that achieves the best prediction accuracy (average RMSE value of 2.51843 min/km). As shown in Fig. 5 the first value of the adjacency class for which our GBLS algorithm achieves an average RMSE value (2.51389 min/km) lower than the one of CSVR's, is 10. This is a candidate to be selected for the optimal adjacency class, but we finally select the value of 13 as it does not require significantly additional computational time (i.e. shorter than one minute) and it leads to even better accuracy. A Wilcoxon signed-rank test proves that there is statistically significant difference for all the prediction steps ahead in time between the prediction accuracy of the GBLS algorithm with adjacency class 13 and that of the CSVR algorithm. Hence, the adjacency class 13 was chosen as the optimal class for the Thessaloniki dataset.

3) Varying the time interval resolution

The time interval size used for aggregating traffic data and for defining the prediction step was 5 minutes. As one may consider, different interval resolutions (e.g. down to 1 minute) may lead to better prediction accuracy and smaller CoD computation times. This possibility was investigated by conducting focal experiments for benchmarking the GBLS method for different interval resolutions. In those tests we used the adjacency class values as calculated in Section IV.C.2. As shown in Fig. 6, increasing the time interval resolution to, e.g. 1 minute, leads to a decrease of the prediction accuracy for the Thessaloniki dataset, as opposed to the Berlin one, for which the accuracy increases. On the other hand, if we increase the length of the time interval, e.g. to 10 minutes or on, the prediction accuracy remains relatively constant for both cities.

However, as it is shown in Fig. 7, the CoD computational time increases significantly as the resolution of the time interval increases, i.e. its size becomes shorter. In the opposite case, it remains relatively constant, exhibiting a slight reduction for time intervals of length 10 or longer.

Hence, our choice of 5-minute long time intervals is justified because it comprises a fair trade-off between accuracy and performance, but also because it is the *de facto* standard in the relevant literature.

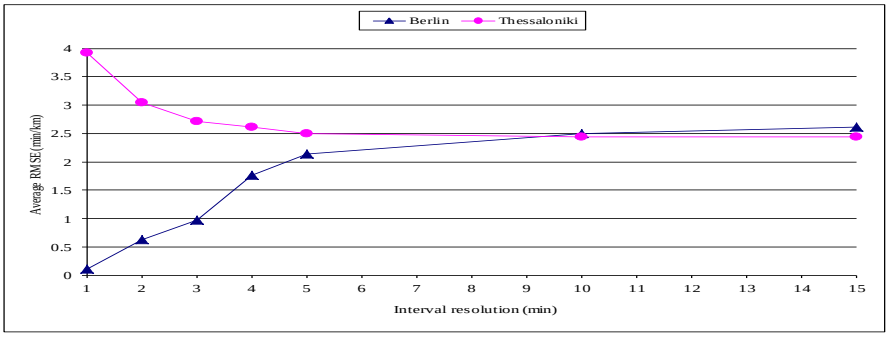

Fig. 6. Average RMSE versus interval resolution

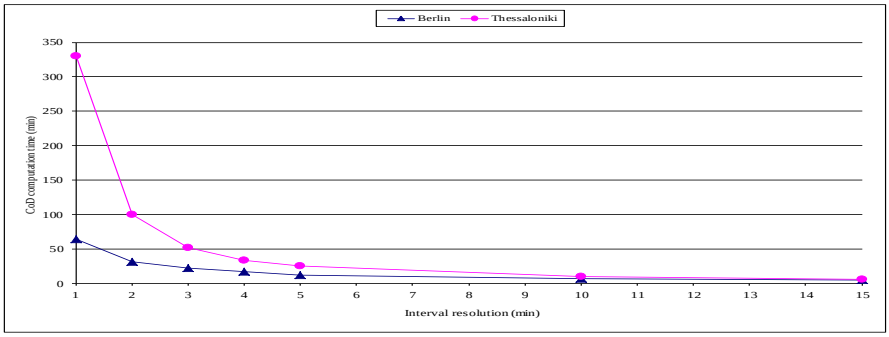

Fig. 7. CoD computation time versus interval resolution

4) GBLS variant

At the core of the GBLS lies the selection of the optimal adjacency class as it determines which neighbors of a road are taken into account for the CoD calculation. An alternative modification of interest would be the one that takes into account a fixed number of neighboring roads. In order to test the impact of this alternative way for selecting neighboring roads, we tested this GBLS variant with a range of fixed number of neighbors. The results are depicted in Fig. 8 and 9.

As shown in Fig. 8, the prediction accuracy remains almost constant for the Berlin dataset as the fixed number of neighbors increases, whereas in the Thessaloniki dataset we notice a slight deterioration in the prediction performance. However, in the case of Berlin the adoption of the variant leads to a higher achieved accuracy compared to the basic adjacency class selection method (Fig. 5). In terms of CoD computation times, we see that performance remains almost constant as the number of neighbors increases for both the Berlin and Thessaloniki datasets. In both cases the performance is significantly improved with respect to the basic adjacency class selection method.

The computational complexity of this variant of the proposed method is $O(N\cdot(nnz(A)/N)^{(x\cdot N/nnz(A))})$, where x is the explicitly selected number of neighbors. This complexity is similar to the one presented in Section III.B. The memory requirements remain these presented in Section III.B.

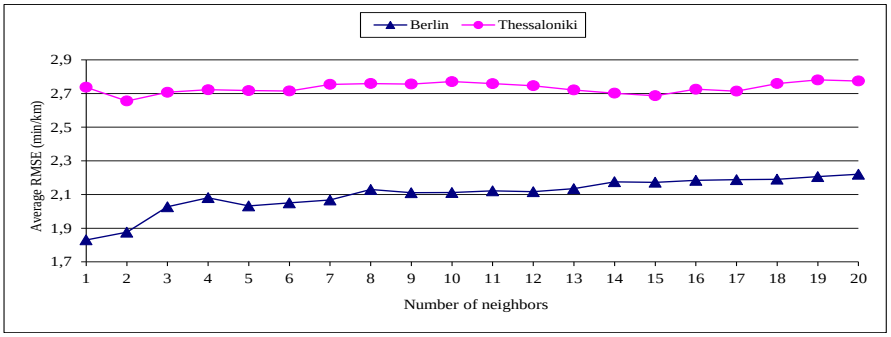

Fig. 8. Average RMSE versus number of neighbors

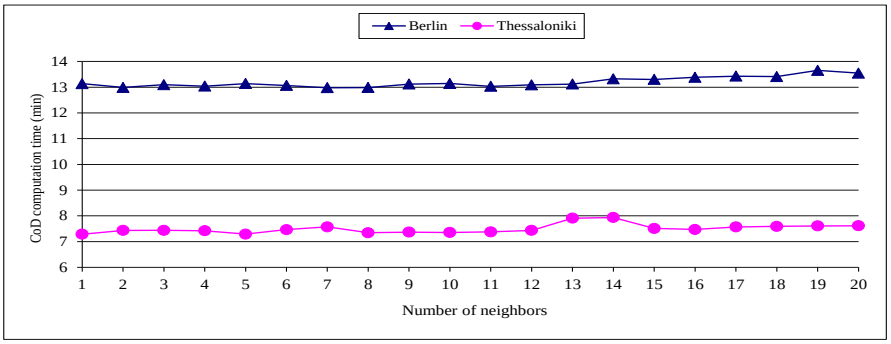

Fig. 9. CoD computation time versus number of neighbors

D. Experimental results

We tested our method against the benchmark techniques described in Section IV.B, with respect to prediction accuracy by means of RMSE. The prediction results presented here refer to network roads for which the corresponding time series, in both training and test datasets, were selected to have at most 50% and 70% missing data for the Berlin and Thessaloniki dataset, respectively. For the CoD calculations we use the GBLS variant presented in Section IV.C.4 for the Berlin dataset. For Thessaloniki we retain the initial GBLS algorithm (Fig. 3) with adjacency class set to 13. The above configurations correspond to the best achieved accuracy.

For all prediction steps ahead in time, GBLS outperforms all benchmark techniques in terms of prediction accuracy for both Berlin (Table III) and Thessaloniki (Table IV) dataset. From the above benchmark results, we notice that the RMSE of travel time for all benchmark techniques is less than 2.7 minutes in all cases. The quality of the prediction improves when the number of steps is low, whereas it becomes worse as the number of concerned steps increases. The average prediction performance, i.e. taking into account all prediction steps for each benchmark technique is depicted in Fig. 10.

Avoiding the computation of CoD for all roads significantly reduces computational time and RAM requirements. Fig. 11 shows the required calculation times when the value of the explicitly specified number of neighbors is 1 for the Berlin dataset, and the value of adjacency class is 13 for the Thessaloniki dataset. As shown in Fig. 11, the computational time reduced from

2,100 minutes (35 hours) to 13.1387 minutes for the Berlin dataset (i.e., 99.374% improvement), and from 1,200 minutes (20 hours) to 25.5433 minutes for the Thessaloniki dataset (i.e., 97.871% improvement) with respect to the naïve (all-to-all) approach for calculating CoD. It should be mentioned that all the experiments were conducted on a computer with a dual core Intel i5 CPU at 3.33 GHz with 4GB RAM.

TABLE III

RMSE VALUES FOR THE BERLIN DATASET

Step	HA	kNN	RF	SVR	CSVR	GBLS
1	2.075	2.138	1.849	1.103	0.827	0.378
2	2.044	2.108	1.856	1.145	0.859	0.781
3	2.020	2.090	1.916	1.158	0.875	0.851
4	2.009	2.067	1.984	1.147	0.870	0.817
5	2.029	2.085	1.989	1.132	0.860	0.746
6	2.045	2.099	2.095	1.109	0.839	0.722
7	2.051	2.104	2.058	1.119	0.855	0.654
8	2.111	2.137	2.125	1.148	0.885	0.716
9	2.124	2.150	2.018	1.110	0.850	0.718
10	2.169	2.211	2.058	1.102	0.837	0.710
11	2.117	2.159	2.141	1.151	0.887	0.755
12	2.170	2.230	2.108	1.203	0.936	0.700

TABLE IV

RMSE VALUES FOR THE THESSALONIKI DATASET

Step	HA	kNN	RF	SVR	CSVR	GBLS
1	2.596	2.593	2.569	2.662	2.338	2.039
2	2.601	2.595	2.573	2.663	2.338	2.255
3	2.605	2.600	2.586	2.664	2.339	2.264
4	2.609	2.607	2.588	2.669	2.341	2.272
5	2.614	2.609	2.571	2.672	2.344	2.279
6	2.613	2.613	2.576	2.671	2.343	2.283
7	2.612	2.615	2.588	2.668	2.340	2.283
8	2.613	2.613	2.583	2.666	2.338	2.285
9	2.615	2.615	2.576	2.667	2.338	2.289
10	2.614	2.611	2.582	2.666	2.336	2.300
11	2.613	2.606	2.591	2.663	2.334	2.309
12	2.616	2.619	2.600	2.664	2.335	2.299

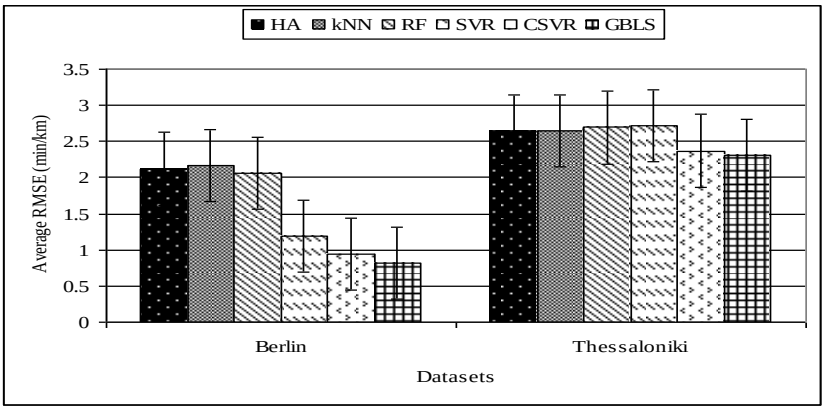

Fig. 10. Average RMSE values

V. CONCLUSION

Based on the benchmark results presented in Section IV, the newly introduced GBLS prediction method can be considered as a travel time prediction technique suitable for big traffic data, such as the ones from Berlin and Thessaloniki. In particular, it renders significant improved performance on the computation of the traffic correlations between roads of the same network, by means of CoD, compared to the basic naïve approach. Specifically, when the proposed technique is used the computational complexity reduces from $O(N^2)$ *to* $O(aN)$, with $a > 1$. Also the proposed technique outperforms benchmark methods from the litera-

ture in terms of prediction accuracy. The complexity of the urban network and the implicated topology are parameters that should be seriously considered for the selection of the appropriate GBLS variant. In the context of this paper we selected to apply those configurations that result in the best accuracy.

A practical implication of the GBLS approach is that it enables the exploration of traffic dependencies between roads in urban networks for improving traffic prediction maintaining computational performance, as opposed to the use of naïve techniques, which rendered the computation of correlation metrics, such as CoD, prohibitive for large networks. This enhancement introduced by GBLS is mainly related to the application of the MBFS. To this end, the overall contribution of GBLS can be seen as enabling the development of more accurate traveller information systems managing associated big traffic data in efficient time.

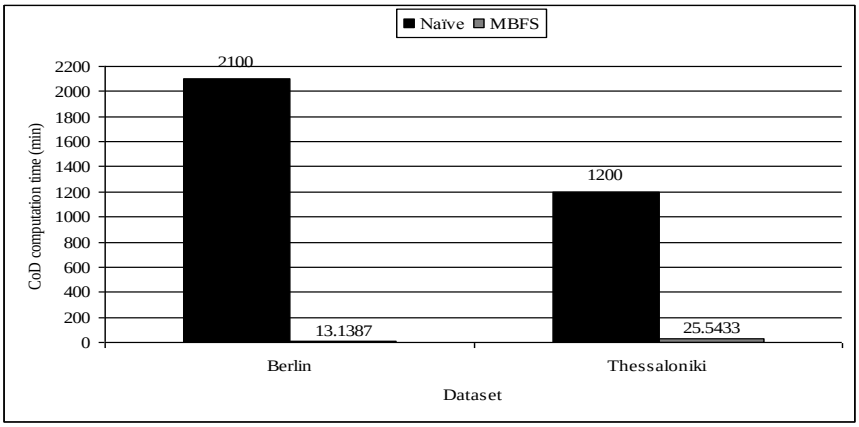

Fig. 11. Computational times against the naïve approach

ACKNOWLEDGMENTS

We would like to thank TomTom Corporation and especially Dr. Felix König, as well as the Hellenic Institute of Transport of the Centre for Research & Technology Hellas (CERTH/HIT) for providing us with anonymous traffic data from Berlin and Thessaloniki, respectively. This work was supported by the FP7 Collaborative Project MOVESMART (Grant Agreement no. 609026) funded by the European Commission.

REFERENCES

[1] T. Diamantopoulos, D. Kehagias, F. König and D. Tzovaras, "Investigating the effect of global metrics in travel time prediction," in *Proc. IEEE-ITSC*, The Hague, The Netherlands, 2013, pp. 412-417.

[2] A. Stathopoulos and M. G. Karlaftis, "A multivariate state-space approach for urban traffic flow modelling and prediction," *Transport. Res. C-Emer.*, vol. 11, no. 2, pp. 121-135, Apr. 2003.

[3] Y. Kamarianakis and P. Prastacos, "Space-time modelling of traffic flow," *Comput. Geosci.*, vol. 31, no. 2, pp.119-133, 2005.

[4] W. Min and L. Wynter, "Real-time road traffic prediction with spatio-temporal correlations," *Transport. Res. C-Emer.*, vol. 19, no. 4, pp. 606-616, Aug. 2011.

[5] Y. Kamarianakis and P. Prastacos, "Forecasting traffic flow conditions in an urban network: Comparison of multivariate and univariate approaches," *Transport. Res. Rec.*, vol. 1857, no.1, pp. 74-84, 2003.

[6] E. I. Vlahogianni, M. G. Karlaftis and J. C. Golias, "A multivariate neural network predictor for short term traffic prediction in urban signalized arterial," presented at the 10th IFAC Symposium on Control in Transportation Systems, Tokyo, Japan, Aug. 4-6, 2003.

[7] M. S. Dougherty and M. R. Cobbet, "Short term inter-urban traffic forecasts using neural networks," *Int. J. Prediction*, vol. 13, no.1, pp. 21-31, Mar. 1997.

[8] B. Abdulhai, H. Porwal and W. Recker, "Short term freeway traffic flow prediction using genetically optimized time delay based neural networks," presented at the 78th TRB Annual Meeting, Washington D. C., USA, 1999.

[9] S. Innamaa, "Short term prediction of traffic situation using MLP-neural networks," in *Proc. ITSWC7*, Turin, Italy, 2000, pp. 1-8.

[10] B. L. Smith and M. J. Demetsky, "Multiple interval freeway traffic flow prediction," *Transport. Res. Rec.*, vol. 155, no. 4, pp. 136-141, 1996.

[11] B. M. Williams, P. K. Durvasula and D. E. Brown, "Urban freeway traffic flow prediction. Application of seasonal autoregressive integrated moving average and exponential smoothing models," *Transport. Res. Rec.*, no. 1644, pp. 132-141, 1998.

[12] M. T. Asif, J. Dauwels, C. Y. Goh, A. Oran, E. Fathi, M. Xu, M. M. Dhanya, N. Mitrovic and P. Jaillet, "Spatial and temporal patterns in large-scale traffic speed prediction," *IEEE Trans. Intell. Transp. Syst.*, vol. 14, pp. 797-804, 2014.

[13] N. Mitrovic, M. T. Asif, J. Dauwels and P. Jaillet, "Low-dimensional models for compressed sensing and prediction of large-scale traffic data," conditionally accepted in *IEEE Trans. Intell. Transp. Syst.*

[14] N. Mitrovic, M. T. Asif, U. Rasheed, J. Dauwels and P. Jaillet, "CUR decomposition for compression and compressed sensing of large-scale traffic data," in *Proc. IEEE-ITSC*, The Hague, The Netherlands, 2013, pp. 1475-1480.

[15] Y. Kamarianakis, W. Shen and L. Wynter, "Real-time road traffic forecasting using regime-switching space-time models and adaptive LASSO," *Appl. Stoch. Models Bus. Ind.*, vol. 28, no. 4, pp. 297-315, Jul. 2012.

[16] G. P. Zhang, "Time series prediction using a hybrid ARIMA and neural network model," *Neurocomputing*, vol. 50, pp. 159-175, Jan. 2003.

[17] Y. Gao and M. J. Er, "Narmax time series model prediction: feed-forward and recurrent fuzzy neural network approaches," *Fuzzy Set. Syst.*, vol. 150, no. 2, pp. 331-350, Mar. 2005.

[18] C. Quek, M. Pasqueir and B. B. S. Lim, "Pop-traffic: a novel fuzzy neural approach to road traffic analysis and prediction," *IEEE T Int. Trans. Syst.*, vol. 7, no. 2, pp. 133-146, Jun. 2006.

[19] D. Easley and J. Kleinberg, "Graphs," in *Networks, Crowds and Markets: Reasoning about a Highly Connected World*, 1st ed., Cambridge University Press, 2010, ch. 2, pp. 23–46.

[20] S. G. Shirinivas, S. Vetrivel and N. M. Elango, "Applications of graph theory in computer science: An overview," *Int. J. Eng. Sci.*, vol. 2, no. 9, pp. 4610–4621, 2010.

[21] D. Volchenkov, "Analysis of urban complex networks," *Condens. Mattter Phys.*, vol. 11, no. 2, pp. 331–340, 2008.

[22] J. Anez, T. De La Barra and B. Perez, "Dual graph representation of transport networks," *Transport. Res. B-Meth.*, vol. 30, no. 3, pp. 209–216, Jun. 1996.

[23] S. Porta, P. Crucitti and V. Latora, "The network analysis of urban streets: A dual approach," *Physica A*, vol. 369, no. 2, pp. 853–866, 2006.

[24] F. R. Boisvert, R. Pozo, K. Remington, R. F. Barret and J. J. Dongarra, "Matrix Market: A web resource for test matrix collections," in *The Quality of Numerical Software: Assessment and Enhancement*, Chapman & Hall, 1997, pp. 125–137.

[25] Y. Saad, *Iterative methods for sparse linear systems*, New York, PWS Publishing, 1996.

[26] G. E. P. Box and G. Jenkins, "Time Series Analysis, Prediction and Control," Holden-Day, Inc., 1990.

[27] P. E. Pfeifer and S. J. Deutsch, "A three-stage iterative procedure for space-time modelling," *Technometrics*, vol. 22, no. 1, pp. 35-47, Feb. 1980.

[28] M. I. A. Lourakis, "A brief description of the Levenberg-Marquardt algorithm implemented by levmar," Ins. Com. Sci., Hellas, Tech. Rep., 2005.

[29] K. Levenberg, "A method for the solution of certain non-linear problems in least squares," *Appl. Math. Quart.*, vol. 2, no. 2, pp. 164-168, 1944.

[30] J. S. Greenfeld, "Matching GPS observations to locations on a digital map," in *Proc. 81th TRB Annual*, Washington D. C., USA, Jan. 13-17, 2002, pp. 164-173.

[31] M. Wojnarski, P. Gora, M. Szczuka, N. S. Hung, J. Swietlicka and D. Zeinalipour, "IEEE ICDM 2010 Contest: TomTom traffic prediction for intelligent GPS navigation," in *IEEE ICDM*, Warszawa, Poland, 2010, pp. 1372-1376.

Athanasios Salamanis received the Diploma degree in Electrical and Computer Engineering from the Aristotle University of Thessaloniki, Thessaloniki, Greece, in 2013. Since then he is working as a research assistant at CERTH/ITI. His main research interests include data mining, machine learning and pattern recognition.

Dionysios D. Kehagias received the Diploma degree in Electrical and Computer Engineering from the Aristotle University of Thessaloniki, Thessaloniki, Greece, in 1999 and the Ph.D. degree in Electrical and Computer Engineering from the same university in 2006. From 2000 to 2006 has worked as a research assistant in CERTH/ITI and since 2007 is a postdoctoral research associate. His research interests include time series data analytics and mining.

Christos K. Filelis Papadopoulos received his Dipl.-Eng. degree from the Electrical and Computer Engineering Department of the Democritus University of Thrace, Greece, in 2010 and the Ph.D. degree in Numerical Analysis and High Performance Scientific Computations from the same Department in 2014. His research interests include preconditioned iterative methods, multigrid and multilevel methods as well as parallel computing.

Dimitrios Tzovaras received the Diploma degree in Electrical and Computer Engineering from the Aristotle University of Thessaloniki, Thessaloniki, Greece, in 1992 and the Ph.D. degree in Electrical and Computer Engineering from the same university in 1997. His main research interests include visual analytics, 3D object recognition, search and retrieval, behavioral biometrics, assistive technologies, information and knowledge management, multimodal interfaces, computer graphics and virtual reality.

George A. Gravvanis received the Bachelor of Science with Joint Honours in Science from University of Salford (Manchester, UK) in 1982, the Degree of Mathematics from National and Kapodistrian University Athens in 1986, Master of Science in Numerical of Analysis from Brunel University (London, U.K.) in 1984 and the Doctoral Degree (PhD) in Computing Sciences from the Department of Informatics of Athens University of Economics and Business in 1992. His research interests include computational methods, mathematical modelling and applications, and parallel computations.