Travel Cost Inference from Sparse, Spatio-Temporally Correlated Time Series Using Markov Models

Bin Yang Chenjuan Guo Christian S. Jensen
Department of Computer Science, Aarhus University, Denmark
{byang, cguo, csj}@cs.au.dk

ABSTRACT

The monitoring of a system can yield a set of measurements that can be modeled as a collection of time series. These time series are often sparse, due to missing measurements, and spatio-temporally correlated, meaning that spatially close time series exhibit temporal correlation. The analysis of such time series offers insight into the underlying system and enables prediction of system behavior. While the techniques presented in the paper apply more generally, we consider the case of transportation systems and aim to predict travel cost from GPS tracking data from probe vehicles. Specifically, each road segment has an associated travel-cost time series, which is derived from GPS data.

We use spatio-temporal hidden Markov models (STHMM) to model correlations among different traffic time series. We provide algorithms that are able to learn the parameters of an STHMM while contending with the sparsity, spatio-temporal correlation, and heterogeneity of the time series. Using the resulting STHMM, near future travel costs in the transportation network, e.g., travel time or greenhouse gas emissions, can be inferred, enabling a variety of routing services, e.g., eco-routing. Empirical studies with a substantial GPS data set offer insight into the design properties of the proposed framework and algorithms, demonstrating the effectiveness and efficiency of travel cost inferencing.

1. INTRODUCTION

When monitoring a complex system, the resulting measurements can often be modeled as a collection of time series. These reflect the internal dynamics of the systems and hold the potential for understanding and predicting aspects of the system's behavior.

While this general scenario applies to a wide variety of settings, we consider the setting of a road network. Figure 1 shows two travel time series obtained from GPS records collected from two adjacent road segments in a road network. Each time series records the average cost associated with traversing its segment during different time intervals. The two time series are correlated: (i) the peak on road 2 in the 28-th interval may result in the peak on road 1 in the 29-th interval; (ii) the peak on road 1 in the 35-th interval may cause the peak on road 2's time series in the 36-th interval.

Proceedings of the VLDB Endowment, Vol. 6, No. 9

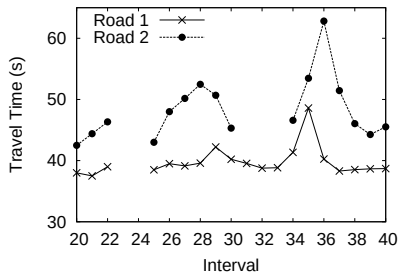


Figure 1: Spatio-Temporally Correlated Sparse Time Series

With a travel-cost time series available (e.g., derived from GPS data) for each road in a road network, a mathematical model of the network can be instantiated and then used for predicting the future travel cost behavior associated with the road network. Thus, it becomes possible to obtain effective routing services in the underlying road network that minimize travel time (e.g., [4]) or GHG emissions, termed eco-routing [5]. However, achieving this is nontrivial, as it is necessary to contend with three challenging characteristics of the time series.

Sparse: Travel-cost time series are built from GPS data obtained from probe vehicles that cannot cover every time interval. As a result, the time series are sparse, meaning that during some intervals, no cost measurements are available for some road segments. For example, both time series in Figure 1 lack travel times in the 23-rd and 24-th intervals, and road 2 also lacks travel times in intervals 31–33. Thus, the time series we consider are fundamentally sparse, which is different from regular time series [2, 13], where each interval has a value.

Dependent: The travel costs associated with a road segment are temporally dependent. For example, congestion on a road segment disappears only gradually (see intervals 28–30 of road 2 in Figure 1). Likewise, time series are spatially correlated. For example, congestion on one road segment may correlate with congestion on spatially adjacent road segments. This is illustrated by the 28-th interval on road 2 and the 29-th interval on road 1. When modeling the overall travel cost behavior of a road network, it is important to capture the dependencies within time series and the correlations among different time series.

Heterogenous: Different road segments may exhibit very different behaviors. For example, some segments may have clear morning and afternoon peak hours, while others may not. We model an individual time series as the output generated by a state transition machine (e.g., a Hidden Markov Model, HMM [2,13]) that operates on a set of states. For instance, the traffic on a road segment may

change between two states, e.g., *clear* and *congested*, and each state may output different travel costs. A prerequisite of using HMMs is that the cardinality of the state set that the underlying process operates on is given a priori. Due to the heterogeneity across time series, it is not feasible to use the same number of states for every time series. Rather, each time series must have its own state set.

We model multiple travel-cost time series using a *Spatio-Temporal Hidden Markov Model* (*STHMM*) while taking into account the above three challenging characteristics. We compress a sparse time series into a compact time series and identify a state set for each compact time series. The dependencies within a time series and the correlations among different time series are modeled by connecting pertinent states of the road segments while considering the topology of the road network. Smoothing techniques are applied to reduce the effects of data sparsity. Finally, future travel costs are inferred based on the learned STHMM.

We believe that this is the first study to provide general techniques, including state formulation and parameter learning, that contend with sparse, heterogenous, and spatio-temporally correlated time series in a combined manner. The paper makes four contributions. First, it proposes a general framework that is capable of modeling the traffic behavior of a road network and thereby enables routing services that optimize different travel-related costs. Second, a spatio-temporal hidden Markov model is formalized to model the traffic behavior of a road network. Third, learning algorithms are proposed to obtain individual state sets for all road segments and to determine the parameters needed to configure an STHMM. Fourth, comprehensive experiments are conducted to elicit the design properties of the proposed framework and algorithms.

The remainder of this paper is organized as follows. Section 2 covers preliminaries. Section 3 defines an STHMM, and Section 4 describes the learning algorithms needed for determining an STHMM. Section 5 reports on the empirical evaluation. Finally, related work is covered in Section 6, and conclusions and research directions are offered in Section 7.

2. PRELIMINARIES

2.1 Road Network Model

A road network is modeled as a directed, labeled graph $G = (V, E, W)$, where V and $E \subseteq V \times V$ is a vertex set and an edge set, respectively, and $W : E \times \mathbb{T} \rightarrow \mathbb{R}^+$ captures time dependent edge weights. A vertex $v_i \in V$ models a road intersection or an end of a road. An edge $e_k = (v_i, v_j) \in E$ models a directed road segment, indicating that travel is possible from its source v_i to its destination v_j. We use the notation $e_k.s$ and $e_k.t$ to denote the source and destination of edge e_k. Figure 2 shows a road network with 5 vertices and 6 edges.

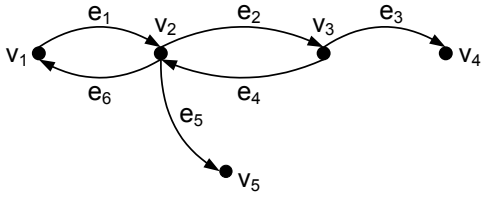


Figure 2: A Simple Road Network

2.2 Travel-Cost Time Series

We construct time series from GPS data obtained from probe vehicles. We use GPS data instead of road side sensor data because GPS data is becoming available in increasing volumes, is less costly to obtain, and provides good coverage of a road network. However, the proposed techniques also apply to time series constructed from road-side sensor data, as they share the same characteristics.

A trajectory $\mathcal{T} = \langle p_1, p_2, \ldots, p_X \rangle$ is a sequence of GPS records pertaining to a particular trip, where each record p_i specifies a $(location, time)$ pair of the vehicle. With the help of map matching [10], a GPS record can typically be mapped to a specific location in the underlying road network.

Map matching transforms a trajectory $\mathcal{T}$ into a sequence of cost records $\langle l_1, l_2, \ldots, l_m \rangle$, where each cost record l_j is of the form $(e, t_s, cost)$, where e is an edge traversed by trajectory $\mathcal{T}$, t_s is the time at which traversing edge e starts, and $cost$ is the travel cost of traversing edge e. Some travel costs, notably travel times, can be obtained directly from the GPS records, while other travel costs, e.g., GHG emissions, can be derived from the GPS records [5].

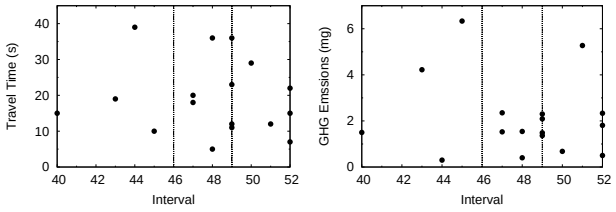
Next, we introduce a parameter α that specifies the finest interval to be used in defining time series. Given a GPS data set collected during Z days, we then have $T = \lceil \frac{Z \cdot 24 \cdot 60}{\alpha} \rceil$ intervals. For example, we may use $\alpha = 15$ minutes because this is typically the finest time granularity used in the transportation area [16]. When $Z = 30$ days, we then have $T = 2,880$ intervals.

We define the travel-cost time series on edge e_i as

$$\mathcal{TS}_i = \langle C_i^{(1)}, C_i^{(2)}, \ldots, C_i^{(T)} \rangle,$$

which is a sequence of T travel cost sets. Travel cost set $C_i^{(t)} = \{l_j.cost | l_j.e = e_i \wedge l_j.t_s \in [(t-1) \cdot \alpha, t \cdot \alpha)\}$, contains the travel costs observed in the t-th interval $[(t-1) \cdot \alpha, t \cdot \alpha)$. If no GPS records are collected on e_i during the t-th interval, $C_i^{(t)}$ is empty.

Figure 3 shows travel-time based and GHG-emissions based time series for the same edge, where α is set to 15 minutes. Both time series are sparse: for example, the 46-th interval contains no data. Figure 3 also illustrates that an average value for an interval [13] cannot capture the traffic behavior during the interval, which may vary considerably throughout the interval. In the example, the travel time in the 49-th interval varies considerably. In the proposed STHMM, the travel cost distribution in the t-th interval is modeled based on $C_i^{(t)}$.



(a) Travel Time (b) GHG Emissions

Figure 3: Examples of Travel-Cost Time Series

The distributions of travel time and GHG emissions may be quite different (e.g., see the 49-th intervals in Figures 3(a) and (b)). A general technique that is able to support different travel costs must handle such variation. Section 5 shows that the proposed STHMM is able to predict both travel time and GHG emissions.

2.3 Framework Overview

In short, we aim to update time-dependent edge weights in G based on real-time travel cost information. Figure 4 gives an overview of the system, which has three major components: offline learning, online inference, and routing services.

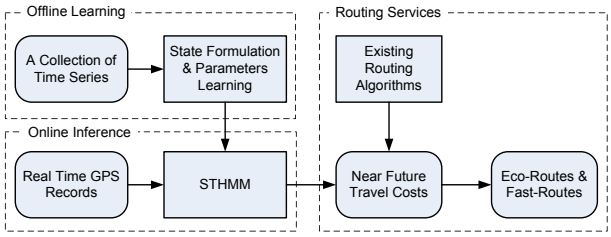


Figure 4: Framework Overview

In offline learning, a state formulation and parameter learning module takes as input a collection of time series that are obtained from historical GPS trajectories. It outputs an STHMM. As real-time GPS records stream in, the online inference component infers near-future travel costs by using the learned STHMM. The time-dependent edge weights in road network G can be updated based on the inferred travel costs. In the routing services, any routing algorithms that support time-dependent edge weights [4, 6, 8] can be applied to road network G to enable various kinds of routing such as eco-routing or time-based routing.

3. TRAVEL COST MODELING

We employ HMMs to capture the temporal dynamics of the travel costs on individual edges; then we define a *Spatio-Temporal Hidden Markov Model (STHMM)* to model the spatio-temporal correlation of travel costs among all edges in a road network.

3.1 Modeling Temporal Dependence

Without loss of generality, we assume that the travel costs of an edge are affected by the underlying traffic on the edge and vary as the traffic changes over time.

Consider a particular edge e_i. A state set $S_i = \{s_i^{(x)}\}$ models all possible traffic conditions on edge e_i. For example, states $s_i^{(1)}$ and $s_i^{(2)}$ may model congestion and clear, respectively. We use a state sequence $\mathcal{Q}_i = \langle q_i^{(1)}, q_i^{(2)}, \ldots, q_i^{(T)} \rangle$ to model the transition of the traffic condition on e_i from one state to another over the T intervals. State $q_i^{(t)} \in S_i$ $(1 \leqslant t \leqslant T)$ models the traffic condition in the t-th interval, and it depends only on its previous state $q_i^{(t-1)}$, i.e., the traffic condition in the $(t-1)$-st interval. In Figure 5, $q_i^{(t)} = s_i^{(1)}$ indicates that edge e_i is in state congestion in the t-th interval, and it is dependent on the state in the $(t-1)$-st interval $q_i^{(t-1)} = s_i^{(1)}$, which is also in congestion.

We use a travel-cost time series $\mathcal{TS}_i = \langle C_i^{(1)}, C_i^{(2)}, \ldots, C_i^{(T)} \rangle$ to model the variation of the travel costs on edge e_i during the T intervals. Here, $C_i^{(t)}$ contains the travel costs observed in the t-th interval and is dependent on the corresponding traffic conditions, i.e., state $q_i^{(t)}$. Figure 5 shows that the travel costs in $C_i^{(t)}$ generally exceed those in $C_i^{(t+1)}$. This corresponds to the traffic in the t-th interval being congested, while the traffic is clear in the $(t+1)$-st interval.

Based on the above, we use a Hidden Markov Model (HMM) [11] to model the relationship between traffic conditions and travel costs. An HMM is a statistical model for modeling a time series, e.g., the travel-cost time series $\mathcal{TS}_i$ on edge e_i, which is generated by a Markov process.

A Markov process operates on a set of hidden states (e.g., a traffic state set S_i) to generate a state sequence (e.g., $\mathcal{Q}_i$). A state in the state sequence depends only on its previous state, and a state generates an output, thus resulting in the time series. In our setting,

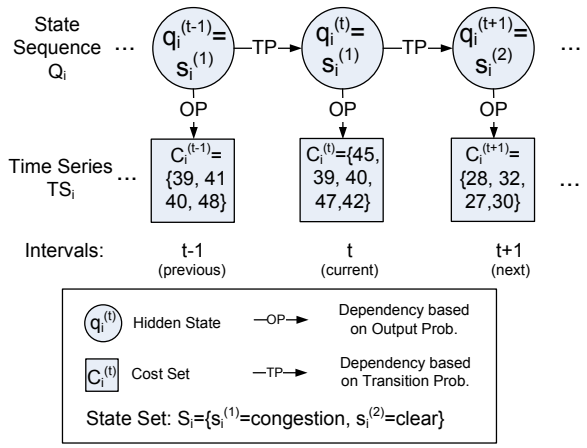


Figure 5: Traffic Conditions and Travel Costs on Edge e_i

the traffic condition in the t-th interval only depends on the traffic conditions in the $(t-1)$-st interval, and travel costs in the t-th interval depend on the traffic conditions in the t-th interval.

An HMM is defined in terms of three parameters PI, A, and B:

1. Initial probabilities $PI = \{\pi^{(x)}\}$, $1 \leqslant x \leqslant |S_i|$, where $\pi^{(x)} = \mathbf{P}(q_i^{(1)} = s_i^{(x)})$ is the probability that state sequence $\mathcal{Q}_i$ starts with state $s_i^{(x)}$.

2. Transition probabilities $A = \{a^{(x,y)}\}$, $1 \leqslant x, y \leqslant |S_i|$, where $a^{(x,y)} = \mathbf{P}(q_i^{(t)} = s_i^{(y)} | q_i^{(t-1)} = s_i^{(x)})$ is the probability that state $s_i^{(x)}$ transits to state $s_i^{(y)}$ in the next step.

3. Output probabilities $B = \{b^{(x)}(c)\}$, $1 \leqslant x \leqslant |S_i|$, where $b^{(x)}(c) = \mathbf{P}(c \in C_i^{(t)} | q_i^{(t)} = s_i^{(x)})$ is the probability that cost value c is observed given the state is $s_i^{(x)}$.

Note that only the time series $\mathcal{TS}_i$ is observable from the available GPS records, while the states are hidden and cannot be observed directly. The formulation of state set S_i is detailed in Section 4.1. The state sequence $\mathcal{Q}_i$ and the time series $\mathcal{TS}_i$ are interrelated. Given the travel costs $C_i^{(t)}$ in the t-th interval, the corresponding state $q_i^{(t)}$ can be inferred using output probabilities. The state in the next interval $q_i^{(t+1)}$ can then be predicted from $q_i^{(t)}$ and the transition probabilities. The travel costs at the next interval, $C_i^{(t+1)}$, can in turn be inferred from $q_i^{(t+1)}$ and output probabilities. The derivation of these probabilities is detailed in Section 4.2.

3.2 Modeling Spatio-Temporal Dependence

The state of an edge influences not only its own next state, but also the next states of nearby edges. For example, an edge is more likely to be congested if its neighboring edges are congested. To model the spatio-temporal dependencies of travel costs over an entire road network, we define an STHMM.

3.2.1 N-th Order Neighbors

To define an STHMM, we need the concept of the n-th order neighbors of an edge e_i, denoted as $L_i^{(n)}$. Given an edge e_i, its 1-st order neighbors $L_i^{(1)}$ is defined in Equation 1.

$$L_i^{(1)} = \{e_i\} \bigcup \{e_j | (e_j.t = e_i.s \vee e_j.s = e_i.t) \wedge \neg(e_j.t = e_i.s \wedge e_j.s = e_i.t)\} \quad (1)$$

Here, $L_i^{(1)}$ includes edge e_i itself and the edges that share a source or target vertex with e_i, but excludes the oppositely directed edge that corresponds to the same physical road segment as e_i. Considering Figure 2, $L_2^{(1)} = \{e_1, e_2, e_3\}$, since edges e_2 and e_1 share v_2, and edges e_2 and e_3 share v_3. Edge e_4 is not in $L_2^{(1)}$ as it is the oppositely directed edge.

The n-th order ($n > 1$) neighbors of edge e_i are defined recursively as $L_i^{(n)} = \bigcup_{e_j \in L_i^{(n-1)}} L_j^{(1)}$. For example, since $L_1^{(1)}$=$\{e_1, e_2, e_5\}$, $L_1^{(2)}$=$L_1^{(1)} \cup L_2^{(1)} \cup L_5^{(1)}$= $\{e_1, e_2, e_5\} \cup \{e_1, e_2, e_3\} \cup \{e_1, e_4, e_5\}$ = $\{e_1, e_2, e_3, e_4, e_5\}$.

3.2.2 Spatio-Temporal Hidden Markov Model

To model the traffic evolution in an entire road network, we need to consider the interactions among the traffic on all edges. A naive model is to treat different edges' traffic evolutions as independent: the traffic on an edge evolves by itself and does not interact with the traffic on other edges. This yields $|E|$ individual HMMs, each of which models the traffic on a single edge. This naive model fails to capture any interactions among the traffic on different edges and is unable to properly model global traffic changes.

We proceed to define an STHMM that couples the traffic of n-th order neighbors to model the interactions among these edges. An n-th order STHMM couples the HMM of an edge with the HMMs of its n-th order neighbors. In an n-th order STHMM, edge e_i's current state $q_i^{(t)}$ is not only dependent on its previous state $q_i^{(t-1)}$, but also on the previous state of each of its n-th order neighbors, i.e., all the states $q_j^{(t-1)}$ where $e_j \in L_i^{(n)}$.

The 1-st order STHMM of the road network in Figure 2 is shown in Figure 6, where only states are shown and the time series are omitted. Taking edge e_1 as an example, since $L_1^{(1)}$=$\{e_1, e_2, e_5\}$, state $q_1^{(t)}$ is dependent on states $q_1^{(t-1)}$, $q_2^{(t-1)}$, and $q_5^{(t-1)}$. Thus, states $q_1^{(t-1)}$, $q_2^{(t-1)}$, and $q_5^{(t-1)}$ are connected to $q_1^{(t)}$ in Figure 6.

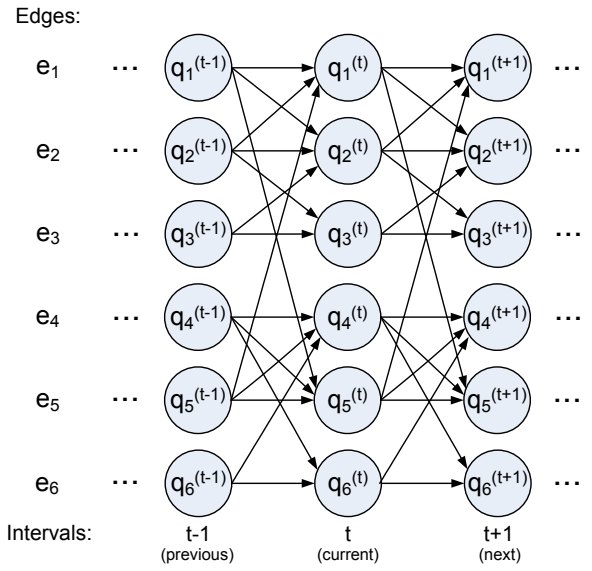


Figure 6: 1-st Order Spatio-Temporal Hidden Markov Model

3.2.3 Parameter Space

In an STHMM, edge e_i's transition probability is a conditional probability conditioned on the previous states of its n-th order neighbors $e_{i_j} \in L_i^{(n)}$. Thus, we have $\mathbf{P}(q_i^{(t)}|q_{i_1}^{(t-1)}, \ldots, q_{i_k}^{(t-1)})$, where $q_{i_j}^{(t-1)} (1 \leqslant j \leqslant k$ and $k = |L_i^{(n)}|)$ is the previous states of edge e_{i_j}. Considering the STHMM in Figure 6, the transition probability factors of edges e_1 and e_6 are $\mathbf{P}(q_1^{(t)}|q_1^{(t-1)}, q_2^{(t-1)}, q_5^{(t-1)})$ and $\mathbf{P}(q_6^{(t)}|q_4^{(t-1)}, q_6^{(t-1)})$, respectively.

An STHMM is governed by parameters TAU, H, and D:

1. Initial probabilities: $TAU = \{\tau_i^{(x)}\}$, $1 \leqslant i \leqslant |E|$, $1 \leqslant x \leqslant |S_i|$, where $\tau_i^{(x)}$=$\mathbf{P}(q_i^{(1)} = s_i^{(x)})$ is the probability that edge e_i starts with state $s_i^{(x)}$.
2. Transition probabilities: $H = \{h_{i,i_1,\ldots,i_k}^{x_i,x_{i_1},\ldots,x_{i_k}}\}$, $1 \leqslant i \leqslant |E|$, $1 \leqslant x_i \leqslant |S_i|$, $e_{i_j} \in L_i^{(n)}, 1 \leqslant x_{i_j} \leqslant |S_{i_j}|$, $1 \leqslant j \leqslant k = |L_i^{(n)}|$. Further, $h_{i,i_1,\ldots,i_k}^{x_i,x_{i_1},\ldots,x_{i_k}} = \mathbf{P}(q_i^{(t)} = s_i^{(x_i)}|q_{i_1}^{(t-1)} = s_{i_1}^{(x_{i_1})}, \ldots, q_{i_k}^{(t-1)} = s_{i_k}^{(x_{i_k})})$ is the probability that the current state of edge e_i is $s_i^{(x_i)}$ given the previous states of its n-th order neighbors $e_{i_1}, \ldots, e_{i_k}$ being $s_{i_1}^{(x_{i_1})}, \ldots, s_{i_k}^{(x_{i_k})}$.
3. Output probabilities: $D = \{d_i^{(x)}(c)\}$, $1 \leqslant i \leqslant |E|$, $1 \leqslant x \leqslant |S_i|$, where $d_i^{(x)}(c) = \mathbf{P}(c \in C_i^{(t)}|q_i^{(t)} = s_i^{(x)})$ is the probability of observing output c from edge e_i given that its state is $s_i^{(x)}$.

4. LEARNING AN STHMM

To obtain an STHMM instance that appropriately models the dynamics of traffic in a road network, we must formulate hidden states for all edges, and we must instantiate the parameters that govern the STHMM. Figure 7 gives an overview of the whole process.

The process of learning an STHMM starts with the state formulation phase (Section 4.1). For each edge $e_i \in E$, its sparse time series $\mathcal{TS}_i$ is compressed into a compact time series $\overline{\mathcal{TS}}_i$. The state formulation module clusters the travel cost sets in compact time series $\overline{\mathcal{TS}}_i$ into a set of clusters, where each cluster indicates a state. Thus, state set S_i for edge e_i is obtained. In addition, the output probabilities are determined.

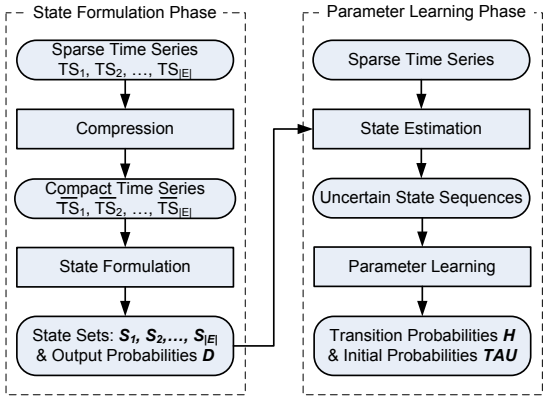


Figure 7: Overview of the STHMM Learning Process

The next phase is parameter learning (Section 4.2). For each edge e_i, based on the obtained state set S_i, a state estimation module estimates possible states for each interval in the sparse time series $\mathcal{TS}_i$, thus obtaining an uncertain state sequence $\mathcal{Q}_i$. Coupling the uncertain state sequences according to their n-th order neighbors, the transition and initial probabilities of the STHMM are determined. Finally, we describe how to use a learned STHMM to infer travel costs (Section 4.3).

4.1 State Formulation

4.1.1 Compact Time Series

The time series $\mathcal{TS}_i = \langle C_i^{(1)}, \ldots, C_i^{(T)} \rangle$ on edge e_i may be sparse because many or some of the $C_i^{(t)}$ may contain no or few travel costs due to the lack of GPS records in the corresponding intervals. To contend with this sparsity, we compress $\mathcal{TS}_i$ into a compact time series $\overline{\mathcal{TS}}_i$ in which each interval contains relatively more travel costs, which renders subsequent analysis (detailed in Sections 4.1.2 and 4.1.3) easier and more effective.

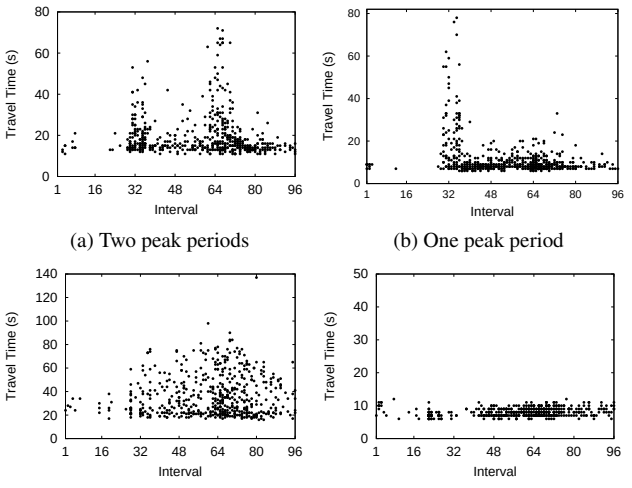
Recall that $\mathcal{TS}_i$ records travel costs on e_i during Z days or $T = \lceil \frac{Z \cdot 24 \cdot 60}{\alpha} \rceil$ intervals. Its compressed version $\overline{\mathcal{TS}}_i$ consolidates these costs into a period of interest P that contains $M = \lceil \frac{P}{\alpha} \rceil$ intervals. In the example in Section 2.2, $Z = 30$ days, $\alpha = 15$ minutes, and $T = 2,880$ intervals. With P being a 24-hour period, $\overline{\mathcal{TS}}_i$ contains $M = 96$ intervals.

Next, we define a compact time series $\overline{\mathcal{TS}}_i$ as follows.

$$\overline{\mathcal{TS}}_i = \langle \overline{C}_i^{(1)}, \overline{C}_i^{(2)}, \ldots, \overline{C}_i^{(M)} \rangle, \text{ where } \overline{C}_i^{(x)} = \bigcup_{k \bmod M = x} C_i^{(k)}$$

Intuitively, set $\overline{C}_i^{(1)}$ in $\overline{\mathcal{TS}}_i$ contains all travel costs observed on edge e_i in the interval $[0{:}00, 0{:}15)$ during each of the Z days. Thus, set $\overline{C}_i^{(1)}$ is the union of $C_i^{(1)}, C_i^{(97)}, C_i^{(193)}, \ldots, C_i^{(2785)}$ in $\mathcal{TS}_i$.

Figure 8 shows four compact travel time series obtained from four different edges. The travel times in Figure 8(a) exhibit clear morning and afternoon peaks, while the times shown in Figure 8(b) have only a clear morning peak. The travel times in Figure 8(c) do not display a clear trend of variation, but vary significantly, from 16 seconds to 137 seconds. The travel times in Figure 8(d) remain almost constant at around 8 seconds.



(a) Two peak periods (b) One peak period

(c) No clear peak periods (d) No obvious variations

Figure 8: Compact Travel Time Series

Figure 8 suggests that the traffic on different edges evolves quite differently. In order to use an STHMM to model the traffic dynamics, it is thus important that each edge e_i can be given its own state set S_i. To achieve this, we propose a *Time Sensitive Gaussian Mixture* (*TSGM*) method to formulate each S_i, where the obtained states satisfy two properties: (i) in a state, the traffic on an edge behaves similarly, thus making the travel cost follow a distribution; (ii) a state is dependent on its previous state. As shown in Algorithm 1, the *TSMG* method consists of two steps, *cost clustering* and *time-cost clustering*, detailed below.

Algorithm 1: TSMG

Input : double: λ; CompTimeSeries: $\overline{\mathcal{TS}}_1 \ldots \overline{\mathcal{TS}}_{|E|}$;
Output: State sets for all edges: $S_1, S_2, \ldots, S_{|E|}$;
1 **for** *each edge* e_i *in the road network* **do**
2 $\quad GMM_i \leftarrow CostClustering(\overline{\mathcal{TS}}_i)$;
3 $\quad S_i \leftarrow TimeCostClustering(\overline{\mathcal{TS}}_i, GMM_i, \lambda)$;

4.1.2 Cost Clustering

For each edge e_i, the cost clustering step identifies a probability density function (pdf) that describes the distribution of all travel costs in $\overline{\mathcal{TS}}_i$, regardless of the interval in which they are observed. As a Gaussian Mixture Model (GMM) is able to approximate any complex pdf [2], the cost clustering step identifies a GMM, denote as GMM_i, to describe the distribution of all travel costs observed on e_i (line 2 in Algorithm 1).

Specifically, GMM_i is determined by grouping all the cost values in $\overline{\mathcal{TS}}_i$ into K_i clusters. Each cluster indicates a representative group of travel costs, whose distribution is described by a single Gaussian distribution, termed a Gaussian component. Equation 2 gives the formal definition.

$$GMM_i(c) = \sum_{k=1}^{K_i} m_{i,k} \cdot \mathcal{N}(c|\mu_{i,k}, \delta_{i,k}^2) \tag{2}$$

Here, $m_{i,k}$ is the k-th mixing coefficients of the k-th Gaussian components, and these satisfy $\sum_{k=1}^{K_i} m_{i,k} = 1$; and the k-th Gaussian component $\mathcal{N}(c|\mu_{i,k}, \delta_{i,k}^2)$ has mean $\mu_{i,k}$ and variance $\delta_{i,k}^2$.

If K_i is given in advance, basic clustering algorithms, e.g., K-Means, can be applied directly. However, deciding an appropriate K_i before the step starts is difficult, and the obtained clusters may not be optimal. An overly small K_i may not fully capture all the representative travel costs on the edge, thus resulting in under-fitting; and an overly large K_i may capture the travel costs over-specifically, yielding over-fitting [2].

We apply the procedure given in Algorithm 2 to select an appropriate K_i. It starts with $K_i = 1$ and then increments K_i by 1 until the benefit (e.g., likelihood) of using K_i is smaller than that of using $K_i - 1$.

Algorithm 2: CostClustering

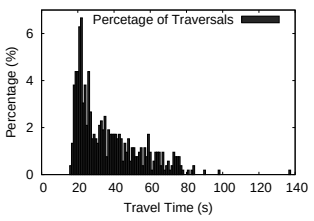
Input : CompTimeSeries: $\overline{\mathcal{TS}}_i = \langle \overline{C}_i^{(1)}, \ldots, \overline{C}_i^{(M)} \rangle$;
Output: GaussianMixtureModel: GMM_i;
1 GMM $preGMM \leftarrow null$, $newGMM \leftarrow null$;
2 double $preLH \leftarrow -\infty$, $newLH \leftarrow -\infty$; int $k \leftarrow 0$;
3 $CC \leftarrow \bigcup_{x=1}^{M} \overline{C}_i^{(x)}$;
4 Split CC into f equal subsets $cc[1], \ldots, cc[f]$;
5 **repeat**
6 $\quad preGMM \leftarrow newGMM$;
7 $\quad preLH \leftarrow newLH$; $newLH \leftarrow 0$;
8 $\quad k \leftarrow k + 1$;
$\quad$ /* f-fold likelihood evaluation */
9 $\quad$ **for** $j = 1 \ldots f$ **do**
10 $\quad\quad train \leftarrow CC \setminus cc[j]$; $test \leftarrow cc[j]$;
11 $\quad\quad newGMM \leftarrow EstimateGMM(train, k)$;
12 $\quad\quad newLH \leftarrow newLH + EvaLH(test, newGMM)$;
13 **until** $newLH \leqslant preLH$;;
14 $GMM_i \leftarrow preGMM$;
15 **return** GMM_i;

All the cost values in $\overline{\mathcal{TS}}_i$ are recorded in CC, and CC is split into f equal subsets (lines 3–4). In the k-th iteration (lines 5–13), a GMM $newGMM$ with $K_i = k$ Gaussian components and the likelihood $newLH$ of using $newGMM$ are obtained.

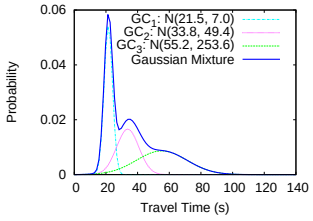
The likelihood of using k Gaussian components is evaluated using f-fold cross validation (lines 9–12). Each of f subsets is used as a testing set $test$ once, and each of the remaining $f-1$ subsets is used as a training set $train$ (line 10). The parameters of a Gaussian mixture model $newGMM$ with k Gaussian components are estimated based on the training set $train$ using a classical Expectation-Maximization algorithm [2] ($EstimateGMM(train, k)$, line 11). The likelihood that the test data $test$ is generated by the GMM $newGMM$ is evaluated ($EvaLH(test, newGMM)$ in line 12).

Consider the edge shown in Figure 8(c), where travel costs range from 0 to 140. Figure 9(a) shows the percentage of traversals (on the y-axis) of the edge with each cost value (on the x-axis). For instance, the highest bar indicates that 6.7% of the traversals took 22 seconds. By applying cost clustering, it is found that a GMM with $K_i = 3$ Gaussian components best describes the travel-time distribution—see Figure 9(b).

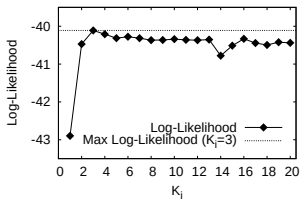
By using 10-fold cross validation, the likelihoods of choosing different numbers of K_i are reported in Figure 9(c), which indicates that $K_i = 3$ is the optimal choice. This example also suggests that although it is infeasible to model travel-cost distributions by a single pdf (e.g., Gaussian, uniform or exponential distribution), properly chosen GMMs can describe such distributions very well.



(a) Percentage of Traversals



(b) GMM with $K_i = 3$



(c) Likelihood vs Varying K_i

GC1, GC2, GC3, KL
s_i^1: [0.35, 0.30, 0.35, 0.031]
s_i^2: [0.33, 0.45, 0.22, 0.047]
s_i^3: [0.64, 0.35, 0.01, 0.602]
s_i^4: [0.15, 0.62, 0.23, 0.306]
s_i^5: [0.31, 0.08, 0.61, 0.132]
s_i^6: [0.16, 0.29, 0.55, 0.076]
s_i^7: [0.62, 0.01, 0.37, 0.093]
s_i^8: [0.53, 0.18, 0.29, 0.066]

(d) Final States

Figure 9: State Formulation

4.1.3 Time-Cost Clustering

For each edge e_i, given the GMM GMM_i with K_i Gaussian components, time-cost clustering (line 3 in Algorithm 1) identifies the state set S_i on e_i. First, each $\overline{C}_i^{(x)}$ in the compact time series $\overline{\mathcal{TS}}_i$ is transformed into a $K_i + 1$ dimensional point. Such a point captures both travel-cost distribution in an interval and the temporal dependency with its previous interval. Thus, $\overline{\mathcal{TS}}_i = \langle \overline{C}_i^{(1)}, \ldots, \overline{C}_i^{(M)} \rangle$ is transformed into M K_i+1 dimensional points $\mathbf{f}_i^{(1)}, \ldots, \mathbf{f}_i^{(M)}$. Second, these points are clustered, such that each cluster refers to a state on edge e_i.

Transforming Compact Time Series to Points: In cost clustering, we identified K_i Gaussian components describing K_i representative travel-cost groups on edge e_i. The travel costs in a particular interval on the edge should also follow the K_i representative travel cost groups. Thus, the distribution of the travel costs in the x-th interval $\overline{C}_i^{(x)}$ is estimated with a new GMM $GMM_i^{(x)}$ with the K_i Gaussian components determined in the cost clustering phase, as defined by Equation 3.

$$GMM_i^{(x)}(c) = \sum_{k=1}^{K_i} \hat{m}_{i,k}^{(x)} \cdot \mathcal{N}(c|\mu_{i,k}, \delta_{i,k}^2) \tag{3}$$

Here, $\mathcal{N}(c|\mu_{i,k}, \delta_{i,k}^2)$ is the k-th Gaussian component, as in Equation 2. However, the $\hat{m}_{i,k}^{(x)}$ are new mixing coefficients that satisfy $\sum_{k=1}^{K_i} \hat{m}_{i,k}^{(x)} = 1$.

Although $GMM_i^{(x)}$ and GMM_i share the same K_i Gaussian components, the mixing coefficients in the two GMMs are typically different because the travel costs observed in each interval typically differ. For example, consider the edge in Figure 9(b). $GMM_i^{(x)}$ may have a larger coefficient for Gaussian component $\mathcal{N}(21.5, 7.0)$ for an offpeak interval, but a smaller coefficient for a peak interval.

Next, each $\overline{C}_i^{(x)}$ is transformed into a $K_i + 1$ dimensional point:

$$\mathbf{f}_i^{(x)} = (\hat{m}_{i,1}^{(x)}, \ldots, \hat{m}_{i,K_i}^{(x)}, KL_i^{(x)})$$

The first K_i coordinates are the new mixing coefficients, and the last coordinate $KL_i^{(x)}$ measures the differences between the distribution of the costs in the x-th interval $\overline{C}_i^{(x)}$ and the distribution of the costs in its previous interval $\overline{C}_i^{(x-1)}$, which reflects the temporal dependency of the distributions of costs in two adjacent intervals. In particular, $KL_i^{(x)}$ is evaluated based on Kullback-Leibler divergence [2], as defined in Equation 4.

$$KL_i^{(x)} = \int_{-\infty}^{+\infty} GMM_i^{(x-1)}(c) \cdot \ln \frac{GMM_i^{(x-1)}(c)}{GMM_i^{(x)}(c)} \, \mathrm{d}c \tag{4}$$

Algorithm 3 transforms a compact time series to a set of points by identifying a point for each $\overline{C}_i^{(x)}$ in $\overline{\mathcal{TS}}_i$. First, new mixing coefficients in $GMM_i^{(x)}$ are initialized as the coefficients in GMM_i that is obtained from cost clustering (lines 2–3).

If the x-th interval is *sparse*, i.e., $\overline{C}_i^{(x)}$ contains fewer than a threshold of $mCounts$ cost values, the procedure skips re-estimating the new mixing coefficients. Rather than using a new GMM re-estimated based on the few cost values in a sparse interval, it is better to use the original GMM_i, which captures the common behavior over the whole period of interest P, to describe the distribution of costs in the sparse interval. This avoids over-fitting to the cost values in sparse intervals.

For example, the 3-rd interval of the edge shown in Figure 8(c) contains only one cost value. Mixing coefficients estimated based on this single value may be over-fitted to this value, which is not optimal. Sparse intervals typically occur during periods with low traffic, for which the value very much depends on the particular driver and therefore can vary considerably.

New mixing coefficients are estimated for the remaining *non-sparse* intervals (lines 4–16). In each iteration, the mixing coefficients are updated based on normalized likelihood values of the mixing coefficients obtained from the previous iteration.

Finally, Kullback-Leibler divergence is evaluated for each interval, and the point for each interval is formulated (lines 17–20). Function $EvaKL(\cdot, \cdot)$ returns a KL divergence value that is normalized to the unit range $[0, 1]$, e.g., by dividing by the largest KL divergence value on edge e_i.

Algorithm 3: CompactTimeSeriesToPoints

Input : CompTimeSeries: $\overline{\mathcal{TS}}_i = \langle \overline{C}_i^{(1)}, \ldots, \overline{C}_i^{(M)} \rangle$;
GMM: GMM_i;
Output: Point Set: F_i;

1 **for** $x = 1 \ldots M$ **do**
2 **for** $k = 1 \ldots GMM_i.K_i$ **do**
3 $GMM_i^{(x)}.\hat{m}_{i,k}^{(x)} \leftarrow GMM_i.m_{i,k}$;
4 **if** $|\overline{C}_i^{(x)}| > mCounts$ **then**
5 **repeat**
6 Initialize an array $sum[GMM_i.K_i]$;
7 **for** *each cost value* $c \in \overline{C}_i^{(x)}$ **do**
8 Initialize an array $lh[GMM_i.K_i]$;
9 **for** $k = 1 \ldots GMM_i.K_i$ **do**
10 $lh[k] \leftarrow GMM_i^{(x)}.\hat{m}_{i,k}^{(x)} \cdot \mathcal{N}(c|\mu_{i,k}, \delta_{i,k}^2)$;
11 **for** $k = 1 \ldots GMM_i.K_i$ **do**
12 $lh[k] \leftarrow \frac{lh[k]}{\sum_{k=1}^{K_i} lh[k]}$;
13 $sum[k] \leftarrow sum[k] + lh[k]$;
14 **for** $k = 1 \ldots GMM_i.K_i$ **do**
15 $GMM_i^{(x)}.\hat{m}_{i,k}^{(x)} \leftarrow \frac{sum[k]}{|\overline{C}_i^{(x)}|}$;
16 **until** *converged*
17 **for** $x = 2 \ldots M$ **do**
18 $KL_i^{(x)} \leftarrow EvaKL(GMM_i^{(x-1)}, GMM_i^{(x)})$;
19 $\mathbf{f}_i^{(x)} \leftarrow (\hat{m}_{i,1}^{(x)}, \ldots, \hat{m}_{i,K_i}^{(x)}, KL_i^{(x)})$;
20 $F_i \leftarrow \mathbf{f}_i^{(1)} \cup \ldots \cup \mathbf{f}_i^{(M)}$;
21 **return** F_i;

Clustering Points: In Section 3.1, we assumed that (i) the traffic in the same hidden state behaves similarly, and (ii) the current state depends on its previous state. Thus, we consider both *travel cost distance* and *temporal dependency distance* when clustering points $\mathbf{f}_i^{(1)}, \ldots, \mathbf{f}_i^{(M)}$.

Dis_C and Dis_T measure the travel-cost distance and the temporal dependency distance between a point $\mathbf{f}_i^{(x)}$ and the center of a cluster U_n, respectively. Recall that the first K_i coordinates of a point $\mathbf{f}_i^{(x)}$ refers to the travel-cost distribution in the x-th interval and that the KL value (i.e., the last coordinate) captures the temporal dependency of the travel cost distributions in the $(x-1)$-st and x-th intervals. Thus, Dis_C computes a distance using the first K_i coordinates of $\mathbf{f}_i^{(x)}$ and of the center of U_n, and it relates to assumption (i); and Dis_T computes a distance using the KL value of $\mathbf{f}_i^{(x)}$ and of the center of U_n, and it relates to assumption (ii). Specifically, we have:

$$\begin{aligned} Dis_C(\mathbf{f}_i^{(x)}, U_n) &= \textstyle\sum_{k=1}^{K_i} m_{i,k} \cdot (\hat{m}_{i,k}^{(x)} - \bar{\hat{m}}_{i,k}^{(n)})^2 \\ Dis_T(\mathbf{f}_i^{(x)}, U_n) &= (KL_i^{(x)} - \bar{KL}_i^{(n)})^2 \end{aligned}$$

Here, $\bar{\hat{m}}_{i,k}^{(n)}$ and $\bar{KL}_i^{(n)}$ are the coordinates of the center of cluster U_n, which is equal to the average value of the k-th coordinates, and the KL value of the points in cluster U_n, respectively.

Based on the above, the distance between a point $\mathbf{f}_i^{(x)}$ and the center of a cluster U_n is defined in Equation 5 as a weighted (using parameter λ), linear combination of Dis_C and Dis_T.

$$Dis(\mathbf{f}_i^{(x)}, U_n) = \lambda \cdot Dis_C(\mathbf{f}_i^{(x)}, U_n) + (1-\lambda) \cdot Dis_T(\mathbf{f}_i^{(x)}, U_n) \quad (5)$$

The time-cost clustering procedure is described in Algorithm 4. K-Means clustering [2] that uses the distance function defined in Equation 5 is applied. The algorithm calls K-Means with $k = 1$ and increments k until a termination criterion is satisfied.

Algorithm 4: TimeCostClustering

Input : CompTimeSeries: $\overline{\mathcal{TS}}_i$; GMM: GMM_i; double λ;
Output: A state set: S_i;

1 $F_i \leftarrow$CompactTimeSeriesToPoints($\overline{\mathcal{TS}}_i$, GMM_i);
2 int $k \leftarrow 0$;
3 **repeat**
4 $k \leftarrow k + 1$;
5 ClusterSet $U \leftarrow K - Means(F_i, k, \lambda)$;
6 $PreviousDis \leftarrow PreviousDis \cup Dis(U)$;
7 **until** $TermHeur(PreviousDis)$;
8 StateSet $S_i \leftarrow U$;
9 **return** S_i;

The algorithm chooses an appropriate number of clusters, each corresponding to a hidden state. In one extreme, if a state is created for each point, the states have the best quality because each state has a unique travel cost distribution and time dependency. However, the parameter space (notably the transition probabilities) of an STHMM increases exponentially with the number of states, thus rendering parameter learning expensive or infeasible. Towards the other extreme, if choosing only one or a small number of clusters, the STHMM is unlikely to capture adequately the traffic dynamics, thus decreasing the travel cost inference accuracy.

To choose the number k of clusters, a termination heuristic (line 7) is applied. We track the overall distance value when having k clusters $U = U_1, \ldots, U_k$, as defined in Equation 6.

$$Dis(U) = \sum_{n=1}^{k} \sum_{\mathbf{f}_i^{(x)} \in U_n} Dis(\mathbf{f}_i^{(x)}, U_n) \quad (6)$$

For most edges, as k increases, the overall distances initially drop quickly, but then subsequently drop only slowly. If the distance decrease is low for several consecutive steps, the process terminates, and the k obtained prior to the onset of the low distance decrease is chosen.

Finally, each cluster is regarded as a state, and a state is represented by the coordinates of its center. For example, eight clusters are identified for the edge shown in Figure 8(c). The coordinates of the cluster centers are shown in Figure 9(d).

Discussion: In time-cost clustering, instead of estimating distinct GMMs with different Gaussian components for the travel costs in different intervals, we use the K_i Gaussian components identified during cost clustering. This has two benefits. First, it provides reliable travel cost distributions for sparse intervals, which reduces the effects of the data sparsity. Second, it allows transformation of the travel cost distribution in each interval into a $K_i + 1$ dimensional point. Clustering of points using weighted Euclidean distance (based on Equation 5) is much more efficient than clustering distributions. Measuring the similarity between two distributions accurately and reliably typically requires expensive sampling and the use of many sampling points.

4.2 Parameter Learning

Having obtained sets of states for all edges, the parameters that specify an STHMM as defined in Section 3.2.3 need to be identified. We determine output, transition, and initial probabilities.

4.2.1 Output Probabilities

Since a state $s_i^{(x)} \in S_i$ of edge e_i is a cluster of points containing mixing coefficients of GMMs, the output probability of the state is also defined as a GMM. The output probability $d_i^{(x)}(c)$ of state $s_i^{(x)}$ on edge e_i is defined as follows.

$$d_i^{(x)}(c) = \sum_{k=1}^{K_i} \bar{m}_{i,k}^{(x)} \cdot \mathcal{N}(c|\mu_{i,k}, \delta_{i,k}^2)$$

The Gaussian component $\mathcal{N}(c|\mu_{i,k}, \delta_{i,k}^2)$ is as defined in Equation 2. The k-th mixing coefficient is the average of all the k-th coordinates of the points in state $s_i^{(x)}$: $\bar{m}_{i,k}^{(x)} = \sum_{\mathbf{f}_i^{(y)} \in s_i^{(x)}} \hat{m}_{i,k}^{(y)} / |s_i^{(x)}|$.

4.2.2 Transition Probabilities

The original, sparse time series are used for learning transition probabilities, as they provide more observations of state transitions than do the compact time series and thus yield more accurate learning. Given a sparse time series, the possible states of each interval are estimated based on the cost values in the interval. Thus, sparse time series are transformed into uncertain state sequences, based on which transition probabilities are determined using maximum likelihood estimation.

State Estimation: Given a set of travel costs $C_i^{(t)}$ during the t-th interval on edge e_i, the probability that the corresponding state is $s_i^{(x)} \in S_i$, denoted as $\mathbf{P}(q_i^{(t)} = s_i^{(x)}|C_i^{(t)})$, is estimated using Equation 7 according to the Bayes' theorem.

$$\mathbf{P}(q_i^{(t)} = s_i^{(x)}|C_i^{(t)}) = \frac{\mathbf{P}(C_i^{(t)}|q_i^{(t)} = s_i^{(x)}) \cdot \mathbf{P}(q_i^{(t)} = s_i^{(x)})}{\sum_{x=1}^{|S_i|} \mathbf{P}(C_i^{(t)}|q_i^{(t)} = s_i^{(x)}) \cdot \mathbf{P}(q_i^{(t)} = s_i^{(x)})} \tag{7}$$

Here $\mathbf{P}(C_i^{(t)}|q_i^{(t)} = s_i^{(x)})$ is the probability that costs in $C_i^{(t)}$ are observed in state $s_i^{(x)}$ (i.e., output probabilities), and thus we have:

$$\mathbf{P}(C_i^{(t)}|q_i^{(t)} = s_i^{(x)}) = \prod_{c \in C_i^{(t)}} \mathbf{P}(c|q_i^{(t)} = s_i^{(x)}) = \prod_{c \in C_i^{(t)}} d_i^{(x)}(c)$$

If $C_i^{(t)}$ is empty due to data sparsity, $\mathbf{P}(C_i^{(t)}|q_i^{(t)} = s_i^{(x)})$ is omitted from Equation 7, i.e., we set $\mathbf{P}(C_i^{(t)}|q_i^{(t)} = s_i^{(x)})$ to 1.

Further, the prior $\mathbf{P}(q_i^{(t)} = s_i^{(x)})$ is decided based on the temporal context, i.e., the t-th interval. Assume that the t-th interval of edge e_i's sparse time series corresponds to the t'-th interval of its compact time series. If the corresponding point in the t'-th interval $\mathbf{f}_i^{(t')}$ is assigned to a state $s_i^{(\gamma)}$ (i.e., $\mathbf{f}_i^{(t')} \in s_i^{(\gamma)}$) in the state formulation phase then $s_i^{(\gamma)}$ is used as the default state.

Although the default state $s_i^{(\gamma)}$ is treated as the most probable state of e_i during the t-th interval, this does not mean that the remaining states $s_i^{(\gamma')}$ ($\gamma' \neq \gamma$ and $s_i^{(\gamma')} \in S_i$) are not possible at all. Motivated by the notion of additive smoothing [9], we smooth the probabilities among every possible state by assigning a small probability ϵ to each non-default state and assigning a big probability $1 - \epsilon \cdot (|S_i| - 1)$ to the default state. Formally, we have:

$$\mathbf{P}(q_i^{(t)} = s_i^{(x)}) = \begin{cases} 1 - \epsilon \cdot (|S_i| - 1) & s_i^{(x)} = s_i^{(\gamma)} \\ \epsilon & s_i^{(x)} \neq s_i^{(\gamma)} \end{cases} \tag{8}$$

Having estimated the possible states for each interval in a sparse time series $\mathcal{TS}_i$, an uncertain state sequence is formulated. The t-th element in the uncertain state sequence contains the probabilities for each state in S_i, namely $\mathbf{P}(q_i^{(t)} = s_i^{(x)}|C_i^{(t)})$ for $1 \leqslant x \leqslant |S_i|$.

Figure 10 shows uncertain state sequences for edges e_1, e_2, and e_5. The costs during the 1-st, 2-nd, and 3-rd intervals are only observed on edges e_1 and e_5, edges e_1 and e_2, and edge e_1, respectively. Thus, possible states on $q_1^{(1)}$, $q_5^{(1)}$, $q_1^{(2)}$, $q_2^{(2)}$, and $q_1^{(3)}$ can be obtained based on the corresponding cost values using Equation 7, while all the remaining states are obtained using Equation 8 (where $\epsilon = 0.01$ as an example).

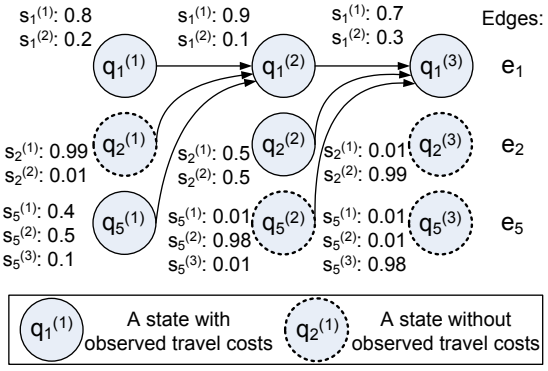


Figure 10: Learning Transition Probabilities

Learning Transition Probabilities: Transition probabilities are learned from the obtained uncertain state sequences using maximum likelihood estimation. The state transition probability of edge e_i, $h_{i,i_1,\ldots,i_k}^{x_i,x_{i_1},\ldots,x_{i_k}}$, is defined as the probability of e_i being in state $s_i^{(x_i)}$ given that the previous states of e_i's n-th order neighbors being $s_{i_1}^{x_{i_1}} \ldots s_{i_k}^{x_{i_k}}$, respectively. It is estimated using Equation 9.

$$h_{i,i_1,\ldots,i_k}^{x_i,x_{i_1},\ldots,x_{i_k}} = \frac{\sum_{t=2}^{T} \mathbf{P}(q_i^{(t)} = s_i^{(x_i)}) \cdot \prod_{j=1}^{k} \mathbf{P}(q_{i_j}^{(t-1)} = s_{i_j}^{(x_{i_j})})}{\sum_{t=1}^{T-1} \prod_{j=1}^{k} \mathbf{P}(q_{i_j}^{(t)} = s_{i_j}^{(x_{i_j})})} \tag{9}$$

An example of determining transition probabilities for edge e_1 is shown in Figure 10. Using Equation 9, $h_{1,1,2,5}^{(1,1,2,3)}$, i.e., the probability of e_1 being $s_1^{(1)}$ given that the previous states of its neighbors e_1, e_2, and e_5 are $s_1^{(1)}$, $s_2^{(2)}$, and $s_5^{(3)}$, is estimated as $h_{1,1,2,5}^{(1,1,2,3)} = \frac{0.8 \cdot 0.01 \cdot 0.1 \cdot 0.9 + 0.9 \cdot 0.5 \cdot 0.01 \cdot 0.7}{0.8 \cdot 0.01 \cdot 0.1 + 0.9 \cdot 0.5 \cdot 0.01} = 0.730$

4.2.3 Initial Probabilities

The initial probability describes the first state of each edge's traffic evolution. As only one uncertain state sequence is identified for each edge, only one training instance is available, which is insufficient to accurately estimate an edge's initial probability.

To derive a meaningful initial probability for each edge, its uncertain state sequence is split based on the period of interest P that was used to obtain the compact time series. Thus, a group of uncertain state sequences is obtained for each edge, which provides more training instances for the estimation. Figure 11 shows an example, where P is a day. The full uncertain state sequence is split into short uncertain state sequences, each corresponding to a day.

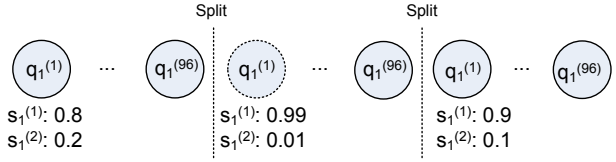


Figure 11: Learning Initial Probabilities

Since initial probabilities are independent among different edges, an edge's initial probability can be learnt by only considering the first states in the short uncertain state sequences of the edge. Given edge e_i, assume that the first states in all the short uncertain state sequences are in set SS_i. Then the initial probability of edge e_i is defined by Equation 10. For example, using the short uncertain states sequences of edge e_1 shown in Figure 11, the probability of the initial state being $s_1^{(1)}$ is: $\tau_1^{(1)} = (0.8+0.99+0.9)/3 = 0.897$.

$$\tau_i^{(x)} = \frac{\sum_{q_i^{(1)} \in SS_i} \mathbf{P}(q_i^{(1)} = s_i^{(x)})}{|SS_i|} \quad (10)$$

4.3 Travel Cost Inference

An STHMM with learned parameters enables travel cost inference. Since the inference is similar to the inference on classical HMMs [11], we omit the details and only briefly discuss two cases.

The first concerns the beginning of travel cost inference, where no real-time GPS data is available. In this case, initial probabilities are applied to infer the next states.

The second case occcurs when GPS data streams into the system. Here, the current state is estimated using Equation 7 based on the travel costs from the latest interval. Based on the learned transition probabilities, the next states are inferred. Using the inferred next states and the learned output probabilities, the distributions of travel costs in the next intervals are also available. For example, the expected values of the estimated distribution can be used to assign near-future edge weights to road network G.

5. EMPIRICAL STUDY

We report on a study that aims to elicit design properties of the proposed framework and algorithms.

5.1 Experimental Setup

Data Set: We use 181 million GPS records collected at 1 Hz (i.e., one GPS record per second) in North Jutland, Denmark during week days from April 2007 to March 2008. The data is from an experiment where young drivers start out with a rebate on their car insurance and then are warned if they speed and are penalized financially if they continue to speed.

The GPS data is map matched [10] to OpenStreetMap's road network for Denmark[1], where 34%, 29%, 15%, 9%, and 13% of the data occurs on tertiary, secondary, residential, motorway, and other roads[2], respectively. Most of the data is from urban and suburban regions. The data set is divided into a training set (for learning an STHMM), and a testing set (for testing the accuracy of inferred travel costs). By default, we use the first half year (April to September) of data for training, and the remaining half year for testing.

Since the data only covers part of Denmark and some covered edges have little data, we consider the subset of the road network that is composed of edges that have at least 500 cost records, denoted as E. If an edge has less than 500 records in a year (i.e., less than 2 records per day), the edge is either unlikely to have much traffic and traffic variation, or the vehicles in the GPS data set do not cover the edge. Such edges are not of interest to us. The resulting, smaller road network contains $|E| = 1,916$ edges. Thus, the proposed STHMM learns $1,916$ correlated time series. We also note that our study with $1,916$ correlated time series exceeds the sizes of existing studies by two orders of magnitude [3, 7, 12, 17]. In fact, existing studies consider at most 10 correlated time series.

[1] http://www.openstreetmap.org

[2] According to OpenStreetMap road categories: http://wiki.openstreetmap.org/wiki/Highway_tag_usage.

Period of Interest P: It is common to focus on "hot" periods when studying traffic behavior [7] because such periods are most interesting. In contrast, travel-time estimation during midnight is of relatively little interest. Here, we consider P = [*6:00, 20:00*), since the GPS data is collected primarily in P. Although we consider a "hot" period and "hot" edges, the obtained time series remain quite sparse. Table 1 reports the percentage of intervals that do not have any costs in the time series obtained using different α.

α (minutes)	15	30	60
Sparsity	89.2%	79.1%	60.2%

Table 1: Sparsity of Traffic Time Series

Travel Costs: We consider travel time (*TT*) and GHG emissions (*GE*). Travel times are obtained as the difference between the corresponding time points of the last and first GPS records on an edge. We use the VT-micro model [1] to estimate the GHG emissions based on instantaneous velocities and accelerations, which are derived from the available GPS records. A recent benchmark [5] indicates that VT-micro is appropriate for this purpose.

Parameters: We vary the parameters α, λ, and n, which are used in the state formulation and parameter learning steps, according to Table 2, where default values are shown in bold. Default values are used unless stated otherwise. Parameter f used in cost clustering (in Algorithm 2) is set to 10, i.e., using 10-fold cross validation. Threshold $mCounts$, used in Algorithm 3, is set to 5. Smoothing parameter ϵ, used in Section 4.2.2, is set to 0.02.

Parameters	Values
α	**15**, 30, 60 (minutes)
λ	0, 0.1, 0.2 , **0.3 (for *TT*)**, 0.4, **0.5 (for *GE*)**, 0.6, 0.7, 0.8, 0.9, 1
n	0, **1**, 2, 3, 4, 5, 6, 7, 8, 9, 10

Table 2: Parameter Settings

Accuracy Measurements: When we infer travel costs for an edge, the intervals that have been traversed at least once are of interest to us because they have ground-truth travel costs, which enables us to measure inference accuracy. We call these *test intervals*, and we consider only these in the experiments.

We quantify the effectiveness of the estimated travel costs using average sum of squared loss ($ASSL$). The $ASSL$ value of edge e_i is defined as $ASSL(e_i) = \frac{1}{M_i} \cdot \sum_{j=1}^{M_i} (EST_j - GT_j)^2$, where M_i is the total number of test intervals of edge e_i and EST_j and GT_j are the estimated and the ground truth costs for the j-th test interval, respectively. In the j-th test interval, the STHMM estimates a state that describes the cost distribution during the interval, so we use the expected cost as EST_j. We compare the STHMM with a baseline method, where the EST_j is the average of the cost values observed during the same interval from the training set. The average of all cost values observed in the j-th test interval is used as the ground truth GT_j. The overall accuracy is the average $ASSL$ value of every edge, where $ASSL = \frac{1}{|E|} \cdot \sum_{e_i \in E} ASSL(e_i)$.

Implementation Detail: All algorithms are implemented in Java using JDK 1.7. To ease the management of Gaussian mixture models, the jEMF package[3] is applied. Some clustering algorithms are implemented based on Weka[4]. A computer with Windows 7 Enterprise, a 3.40GHz Intel Core i7-2600 CPU, and 16 GB main memory is used for all experiments.

[3] http://www.lix.polytechnique.fr/~nielsen/MEF/

[4] http://www.cs.waikato.ac.nz/ml/weka/

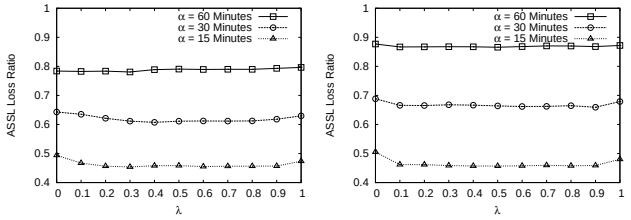
5.2 Effects of α and λ

To observe the effects of parameters α and λ, we plot the *ASSL loss ratio* (i.e., $\frac{ASSL_{STHMM}}{ASSL_{Baseline}}$) while varying α and λ in Figure 12. Recall that λ controls the relative importance of travel cost distance and temporal dependency distance in time-cost clustering.

Given a fixed λ, the STHMM is always superior to the baseline method for both *TT* and *GE*. For example, for *TT* and $\alpha = 15$, the best ratio is around 45%, meaning that the $ASSL$ of the STHMM is 45% of the baseline's $ASSL$. This clearly demonstrates the effectiveness of our STHMM approach.

The STHMMs with finer-grained intervals (i.e., smaller α) produce better $ASSL$ loss ratios. It is expected that the traffic between two consecutive 15-minute intervals is more inter-dependent than between two consecutive 60-minute intervals. Longer intervals lower the traffic dependencies between consecutive intervals, yielding a worse $ASSL$ loss ratio.

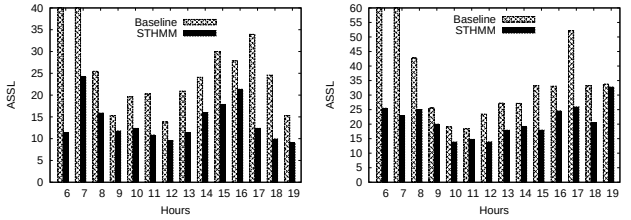
We fix α and study the effect of varying λ. If we only consider cost similarity ($\lambda = 1$) or only consider temporal similarity ($\lambda = 0$), the $ASSL$ loss ratios on both *TT* and *GE* are higher than when both similarities are considered, as seen in Figure 12. We also see that the loss ratio is not very sensitive to λ, which makes it easy to choose an appropriate λ value. We use $\lambda = 0.3$ and $\lambda = 0.5$ as defaults for subsequent experiments on *TT* and *GE*, respectively, as these values produce the best results.



(a) Travel Time (b) GHG Emissions

Figure 12: Effects of α and λ

Next, to observe the effectiveness of the STHMM across time, we plot the $ASSL$ values for the baseline and the STHMM with α=15 at an hourly granularity in Figure 13. For both *TT* and *GE*, the STHMM always has a lower $ASSL$ for each hour. The results suggest that the STHMM models traffic evolution during finer-grained intervals much more effectively than the baseline method and is able to provide much more accurate dynamic edge weights.



(a) Travel Time (b) GHG Emissions

Figure 13: Hourly Inference, $\alpha = 15$

The total run time of learning an STHMM and the average run time of performing *TT* inference per test interval on the learned STHMM are shown in Figure 14 (results on *GE* are similar and are thus omitted). Recall that both state formulation and parameter learning are conducted off-line and are not time critical. The figure shows that as α increases, the total run time of both phases decreases. Given a training period, a smaller α yields more intervals, thus creating more data to be considered in both phases. For $\alpha = 15$, more intervals with varying travel-cost distributions need to be handled during time-cost clustering, yielding state sets with higher cardinalities. Higher cardinalities increase the parameter spaces of the transition probabilities, which makes parameter learning take longer than when $\alpha = 30$ or 60. However, the longest total run time of both phases is below 92 minutes, which is acceptable for an off-line computation.

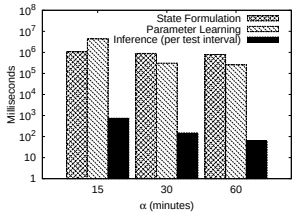


Figure 14: Efficiency, *TT*

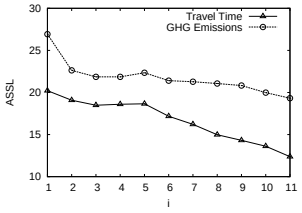


Figure 15: Training Size

It is also important to note that both state formulation and parameter learning are parallelizable. State formulation for an edge only uses the compact time series from the edge, and parameter learning for an edge only employs a group of uncertain state sequences from the edge's n-th order neighbors. By distributing the compact time series and pertinent groups of uncertain state sequences to different computer nodes, both state formulation and parameter learning for different edges can be conducted in parallel. Frameworks such as MapReduce are capable of distributing pertinent data to different nodes and thus enable parallel state formulation and parameter learning. This way, the total run time can be further reduced using known techniques.

Travel cost inference is conducted on-line, meaning that as real-time GPS data streams in, near-future travel costs are to be inferred with little delay. Thus, travel cost inference is time critical. The "Inference (per test interval)" columns in Figure 14 suggest that travel cost inference is quite efficient and is capable of supporting real-time inference.

5.3 Effects of Training Data

To observe **the effect of the sizes of training sets**, we use data from the first i months for training, and the remaining data for testing, where $1 \leqslant i \leqslant 11$. For example, when $i = 3$, data from April–June 2007 is used for training. The $ASSL$ values using different training sets are reported in Figure 15. As more data is used for training, the inference accuracy for both *TT* and *GE* also increases (i.e., lower $ASSL$ values).

Note that the STHMM always outperforms the baseline, especially when the training set is small. In particular, when $i = 1$, the $ASSL$ values of the STHMM is only 19% and 27% of those of the baseline for *TT* and *GE*, respectively, indicating that the STHMM works well when the training set is small and sparse.

It is of interest to **incorporate recent data** and learn an up-to-date STHMM periodically. Assume that we learn a new STHMM every month. We consider two strategies. When predicting travel costs for a new month (e.g., September), *Strategy 1* considers the data collected from all previous months (e.g., April–August) to learn a new STHMM, while *Strategy 2* only considers the data collected from the two most recent months (e.g., July–August). We compare these with a static approach (that always uses an STHMM learned from data collected from April–May).

The results on *TT* are shown in Figure 16. The periodically learned STHMMs outperform the static one, especially when the

test months are further away from the training months (e.g., from October 2007 to March 2008). The two strategies are almost equally effective (i.e., similar $ASSL$ values), but have quite different efficiencies. As time passes, *Strategy 1* uses more and more training data and thus takes longer and longer time (up to around 2 hours). *Strategy 2* always uses only two months of data for training and takes from 25–35 minutes, depending on the numbers of GPS records in the different months. Thus, *Strategy 2* is preferable.

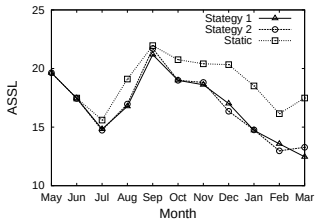


Figure 16: Recent Data, *TT*

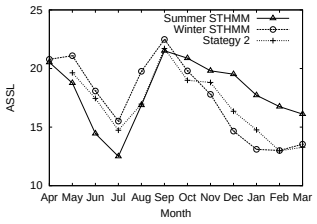


Figure 17: Seasonality, *TT*

As summer and winter may have different traffic conditions, it is of interest to study the **effect of the seasonality**. We learn a summer STHMM with data from June and July and a winter STHMM with data from December and January. The results on *TT* shown in Figure 17 indicate that the summer STHMM achieves better accuracy for summer months (e.g., May and August), while the winter STHMM is best for winter months (e.g., November, February, and March). For the spring and fall months (e.g., April, September, October), both STHMMs perform similarly. The findings are consistent with domain knowledge of summer/winter traffic in Denmark. We also include *Strategy 2* in Figure 17: it offers a reasonable all-year accuracy.

We suggest that (i) if a region has different summer and winter traffic, using separate summer and winter STHMMs yield the best results; otherwise, (ii) using *Strategy 2* is also effective. The effects of incorporating recent data and the seasonality on *GE* are similar, and thus are omitted.

5.4 Effects of n-th Order Coupling

We proceed to consider the effects of different coupling levels, i.e., using different n-th order STHMMs. When $n = 0$, the STHMM degrades to the naive model that is composed of $|E|$ independent HMMs (refer to Section 3.2.2). The naive model can be regarded as an improved version of the state-of-the-art approach [14] that applies a Markov model to predict future travel times. It formulates states using travel time clusters. In contrast, in our naive model, a state is a cluster represented by a Gaussian mixture model describing a travel cost distribution, and a KL divergence value representing its temporal dependency w.r.t. its previous state. Also, the state-of-the-art approach is only tested on travel time, not on GHG emissions.

We choose a subset of edges $E' \subset E$ with state set cardinalities that exceed 1 to observe the effect of varying n. The reason for using E' is twofold: if an edge's state set cardinality is 1, (i) the edge stays in its single state, regardless of the states of its n-th neighbors, and (ii) the edge also cannot effect its neighbor edges' traffic because it is always in the single state. Thus, if the cardinality is 1, the n-th ($n \geqslant 1$) order STHMM should yield the same results as the naive model.

The chosen E' contains 653 edges for *TT* and 723 edges for *GE*. As the distributions of *TT* and *GE* can be quite different (cf. the 49-th interval in Figure 3) even when both are derived from the same GPS data, the state set cardinalities on the same edge are also quite different for *TT* and *GE*. Thus, the E' contains different edges for *TT* and *GE*.

We plot the average and maximal cardinalities of n-th order neighbors derived from E' in Figure 18. When n is large (> 4), some edges have many neighbor edges, causing an exponential increase in the parameter spaces of transition probabilities in the STHMM. For instance, when $n = 5$, an edge may have 28 neighbor edges. If each edge has 4 states, the transition probability of the edge needs to maintain $4 \cdot 4^{28}$ entries, which causes prohibitively high computation and storage overheads. Fortunately, an approximation algorithm (AA) [3, 17] is able to reduce the parameter space from exponential to linear. We use AA to learn the transition probabilities in the STHMM for n up to 10. The details of AA are omitted for brevity.

When $n \leqslant 4$, we use the proposed learning algorithms to study the effectiveness of the n-th order STHMM based on 50 randomly chosen edges from E'. We do not include all edges in E' because parameter learning on some edges is quite time consuming when $n = 3$ or 4. The *couple ratio* $= \frac{ASSL_{n\text{-th order STHMM}}}{ASSL_{Naive}}$ is shown in Figure 19.

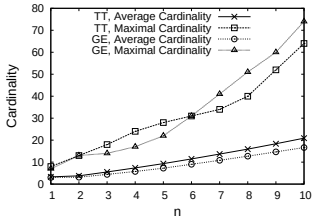


Figure 18: Cardinality

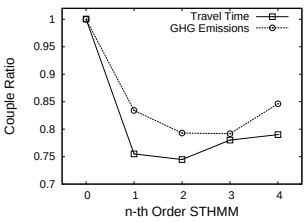


Figure 19: Couple Ratio

The n-th ($n \geqslant 1$) order STHMM outperforms the naive model. In particular, the benefit of using the 1-st order STHMM is most significant for inferring both *TT* and *GE* (see the two steep slopes from $n = 0$ to $n = 1$). The 2-nd and 3-rd order STHMMs yield the best results for *TT* and *GE*, respectively. When $n > 1$, the benefit of using a higher-order STHMM is not prominent, and for $n = 4$ the couple ratios even increase. The results suggest that the traffic interactions among different edges are relatively localized and that a fully coupled model [3, 7, 12, 17] is not well suited for road network traffic modeling.

The average run time of parameter learning per edge and the average run time of inference per test interval are shown in Figure 20. Figure 14 gives the run time of state formulation, which does not change when varying n. As n increases, the parameter space of the STHMM also increases, thus increasing the average run time of parameter learning. The 1-st order STHMM and the naive model have similar run times, and the n-th order STHMM is expensive to learn when $n > 1$. Considering the effectiveness shown in Figure 19, the 1-st order STHMM stands out as offering a high effectiveness gain while limiting the additional run time cost.

Next, we apply AA to all edges in E' to study the effectiveness and efficiency of n-th order STHMMs for $1 \leqslant n \leqslant 10$. The *Appr Ratio*$= \frac{ASSL_{n\text{-th order STHMM AA}}}{ASSL_{1\text{-st order STHMM AA}}}$ is shown in Figure 21, and the average run time of AA is also shown in Figure 20. AA is much more efficient, but also less effective. Higher order ($n > 5$) STHMMs almost keep the same effectiveness, suggesting that traffic on further-away edges seldom affect the traffic of the edge of interest.

We recommend the use of a lower order ($n < 3$, especially $n = 1$) STHMMs for travel cost inference in road networks. These provide significant benefits at low training times.

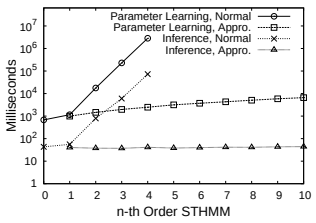


Figure 20: Efficiency, TT

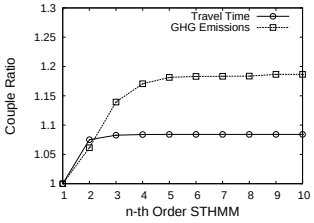


Figure 21: Appr. Ratio

6. RELATED WORK

Time Series Analysis: Hidden Markov Models (HMMs) [2, 11] and coupled HMMs [3, 7, 12, 17] are proposed to model individual time series and to model the interactions among multiple time series, respectively. The proposed STHMM has several unique characteristics. First, while the existing approaches consider regular, non-sparse time series (i.e., one value per interval), the heterogenous, sparse time series that we consider may have multiple or no value(s) in an interval.

Second, HMMs and coupled HMMs assume the state set of an individual time series is given a priori, e.g., using domain knowledge. This assumption does not hold in our setting where identifying a distinct state set for each traffic time series is an important challenge. A recent approach, pHMM [13], segments a regular time series into line segments and clusters these line segments to obtain states. Although pHMM is capable of computing state sets for regular time series without relying on prior knowledge, it is inapplicable in our setting because linear transformation of regular time series does not apply to our sparse and heterogenous time series. Rather, the STHMM identifies a distinguishable state set as a set of Gaussian mixture models for each sparse time series.

Third, parameter learning for the STHMM differs from classical HMM parameter learning, e.g., using the Baum-Welch algorithm [11], which needs to estimate output probabilities while estimating initial and transition probabilities. In contrast, the STHMM identifies the output probabilities in the state formulation phase, which simplifies the estimation for initial and transition probabilities in the parameter learning phase. Further, coupled HMMs couple each time series with all the other time series and thus have huge parameter spaces. Existing studies [3, 17] focus on proposing approximate models along with randomized learning algorithms to avoid inaccurate or inefficient inferencing as much as possible. Our STHMM considers the structure of the underlying road network to couple only those time series that need coupling, thus reducing the parameter space substantially.

Travel Cost Inference: Most existing work on travel cost estimation focuses on travel time estimation, and only few studies consider GHG emissions. Further, most estimation approaches are quite static in nature. Although a recent approach [15] learns time-dependent travel times and GHG emissions based weights for roads (even for the roads without any GPS data), the weights remain static and do not consider real-time traffic data. The current state-of-the-art proposal for travel time inference, which is also the most related to ours, uses a Markov model to update the travel time on a road based on real-time traffic data from the road. However, the real-time support considers each road segment in isolation, does not exploit the correlation of travel times among road segments, and does not contend with sparsity and heterogeneity. Section 5.4 includes an empirical comparison with an improved version of this approach. One study [7] uses coupled HMMs to model dynamic travel times using loop detector data, which assumes that the state set of each time series is given in advance. Further, since loop detectors can constantly report travel time, sparsity is not considered.

7. CONCLUSION AND OUTLOOK

We study real-time travel cost inferencing from multiple correlated, sparse time series based on GPS records from probe vehicles. A spatio-temporal hidden Markov model is formalized to model multiple correlated traffic time series, while considering sparsity, dependency, and heterogeneity in an integrated manner. An empirical study considering travel time and GHG emissions demonstrates that the framework and algorithms are effective and efficient.

As the volumes of GPS data increase, it becomes possible to study even larger numbers of correlated traffic time series. The resulting STHMM parameter space increase renders the learning more difficult. First, exact learning may not be feasible, calling instead for approximate sampling based learning. Second, scalable learning algorithms are of particular interest when we face large numbers of correlated time series.

Acknowledgments

This work was partially supported by the Reduction project that is funded by the European Commission as an FP7-ICT-2011-7 STREP project, contract number 288254.

8. REFERENCES

[1] K. Ahn, H. Rakha, A. Trani, and M. Van Aerde. Estimating vehicle fuel consumption and emissions based on instantaneous speed and acceleration levels. *Journal of Transportation Engineering*, 128(2):182–190, 2002.

[2] C. Bishop. *Pattern recognition and machine learning*. Springer, 2006.

[3] M. Brand. Coupled hidden Markov models for modeling interacting processes. Technical report, MIT Media Lab.

[4] B. Ding, J. X. Yu, and L. Qin. Finding time-dependent shortest paths over large graphs. In *EDBT*, pp. 205–216, 2008.

[5] C. Guo, Y. Ma, B. Yang, C. S. Jensen, and M. Kaul. Ecomark: Evaluating models of vehicular environmental impact. In *GIS*, pp. 269–278, 2012.

[6] E. Kanoulas, Y. Du, T. Xia, and D. Zhang. Finding fastest paths on a road network with speed patterns. In *ICDE*, p. 10, 2006.

[7] J. Kwon and K. Murphy. Modeling freeway traffic with coupled HMMS. Technical report, U. California, Berkeley, 2000.

[8] N. Malviya, S. Madden, and A. Bhattacharya. A continuous query system for dynamic route planning. In *ICDE*, pp. 792–803, 2011.

[9] C. Manning, P. Raghavan, and H. Schütze. *Introduction to information retrieval*, vol. 1. Cambridge University Press, 2008.

[10] F. Pereira, H. Costa, and N. Pereira. An off-line map-matching algorithm for incomplete map databases. *European Transport Research Review*, 1(3):107–124, 2009.

[11] L. Rabiner. A tutorial on hidden Markov models and selected applications in speech recognition. *Proceedings of the IEEE*, 77(2):257–286, 1989.

[12] I. Rezek, P. Sykacek, and S. Roberts. Learning interaction dynamics with coupled hidden Markov models. *IEE Proceedings-Science, Measurement and Technology*, 147:345, 2000.

[13] P. Wang, H. Wang, and W. Wang. Finding semantics in time series. In *SIGMOD Conference*, pp. 385–396, 2011.

[14] J. Yuan, Y. Zheng, X. Xie, and G. Sun. Driving with knowledge from the physical world. In *KDD*, pp. 316–324, 2011.

[15] B. Yang, M. Kaul, and C. S. Jensen. Using Incomplete Information for Complete Weight Annotation of Road Networks. In *TKDE*, 2013.

[16] W. Zheng, D. Lee, and Q. Shi. Short-term freeway traffic flow prediction: Bayesian combined neural network approach. *Journal of Transportation Engineering*, 132(2):114–121, 2006.

[17] S. Zhong and J. Ghosh. A new formulation of coupled hidden Markov models. Technical report, U. Texas, Austin, 2001.