Geoinformatica (2017) 21:57–88
DOI 10.1007/s10707-016-0272-z

Enabling time-dependent uncertain eco-weights for road networks

Jilin Hu[1] · Bin Yang[1] · Christian S. Jensen[1] · Yu Ma[2]

Received: 7 July 2015 / Revised: 24 July 2016 /
Accepted: 21 September 2016 / Published online: 29 September 2016

Abstract Reduction of greenhouse gas (GHG) emissions from transportation is an essential part of the efforts to prevent global warming and climate change. Eco-routing, which enables drivers to use the most environmentally friendly routes, is able to substantially reduce GHG emissions from vehicular transportation. The foundation of eco-routing is a weighted-graph representation of a road network in which road segments, or edges, are associated with eco-weights that capture the GHG emissions caused by traversing the edges. Due to the dynamics of traffic, the eco-weights are best modeled as being time dependent and uncertain. We formalize the problem of assigning a *time-dependent, uncertain eco-weight* to each edge in a road network based on historical GPS records. In particular, a sequence of histograms is employed to describe the uncertain eco-weight of an edge at different time intervals. Compression techniques, including histogram merging and buckets reduction, are proposed to maintain compact histograms while retaining their accuracy. In addition, to better model real traffic conditions, virtual edges and extended virtual edges are proposed in order to represent adjacent edges with highly dependent travel costs. Based on the techniques above, different histogram aggregation methods are proposed to accurately estimate time-dependent GHG emissions for routes. Based on a 200-million GPS record data set collected from 150 vehicles in Denmark over two years, a comprehensive empirical study

✉ Bin Yang
byang@cs.aau.dk

Jilin Hu
hujilin@cs.aau.dk

Christian S. Jensen
csj@cs.aau.dk

Yu Ma
yu@cs.au.dk

[1] Department of Computer Science, Aalborg University, Aalborg, Denmark

[2] Department of Computer Science, Aarhus University, Aarhus, Denmark

is conducted in order to gain insight into the effectiveness and efficiency of the proposed approach.

Keywords Road network · Eco-routing · Time-dependent uncertain edge weight

1 Introduction

The greenhouse effect is due to the concentration of greenhouse gases (GHG) in the Earth's atmosphere, which prevents heat from escaping into space. The combustion of fossil fuel results in GHG emissions, and transportation is a prominent fossil fuels burning sector. Thus, reducing the GHG emissions from transportation is crucial in combating global warming.

Eco-routing is an easy-to-employ and effective approach to reducing GHG emissions from transportation. Given a source-destination pair, eco-routing returns the most environmentally friendly route, i.e., the route that produces the least GHG emissions [2, 8]. The literature reports that eco-routing can yield 8–20 % reductions in GHG emissions from road transportation [6].

Neither the shortest nor the fastest routes generally have the least environmental impact [2]. Figure 1 shows an example of the shortest route, the fastest route, and the eco-route between source A and destination D.

Vehicle routing generally relies on a weighted-graph representation of a road network, where the vertices and edges represent road intersections and road segments, respectively. The key to enabling effective eco-routing is to assign eco-weights to the edges that accurately capture the environmental costs (i.e., GHG emissions or fuel consumption) of traversing the edges. Based on the resulting weighted graph and the types of weights, e.g., single-value weights, time-dependent weights, or uncertain weights, existing routing algorithms [12, 17, 23, 25, 26] can be applied to enable eco-routing.

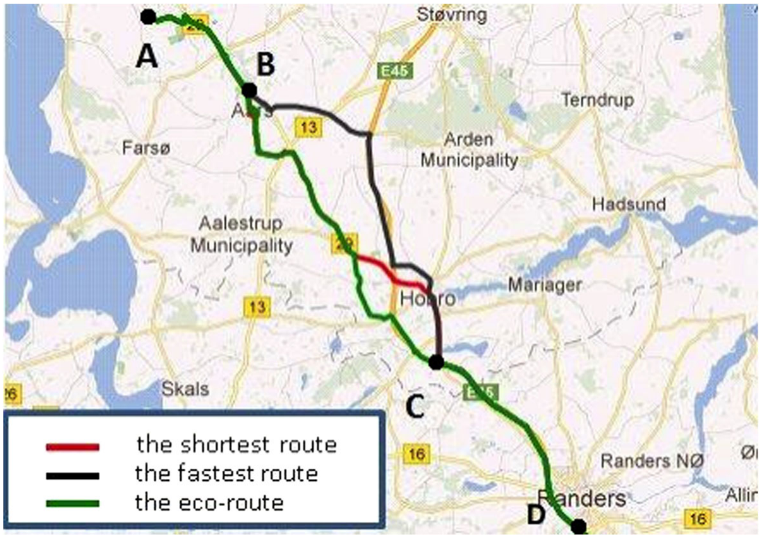


Fig. 1 Eco-Route, Fastest Route, and Shortest Route

A single-valued edge weight typically cannot fully capture the environmental cost of traversing an edge. For instance, while traversing an edge, aggressive drivers may generate more GHG emissions than average drivers. Thus, emissions vary across drivers, and an uncertain eco-weight that records the distribution of the cost of traversing an edge captures reality better [22]. Further, eco-weights are generally time dependent, due to the temporal variation in traffic. For instance, during peak hours, traversing an edge normally produces more GHG emissions than during off-peak hours. As an example, Fig. 2a shows GHG emissions cost values observed on an edge in our road network during off-peak and peak hours, and Fig. 2b shows the corresponding uncertain edge weights of the edge during off-peak and peak hours as histograms.

According to a recent benchmark on vehicular environmental impact models [6, 7], environmental costs of traversing edges can be derived from GPS data using vehicular environmental impact models. Based on such models, a previous study [16] offers a preliminary attempt at assigning time-dependent, uncertain eco-weights to edges. That study derives a time-dependent histogram for each edge to represent the eco-weight of each edge. That study makes the assumption that the histograms on different edges are independent and proposes a method to estimate the GHG emission distributions of a route at a given time using the eco-weights.

This paper makes three main contributions to extend and enhance the previous study [16]. First, by introducing virtual edges and extended virtual edges, we make it possible to capture the dependence among eco-weights of adjacent edges. Second, we propose several histogram aggregation methods that are able to estimate GHG emissions of routes based on the eco-weights of edges, virtual edges, and extended virtual edges. Third, experiments are conducted on a comprehensive GPS data set that provide insight into the efficiency and accuracy of the paper's proposals.

The remainder of the paper is organized as follows. Section 2 reviews related work, and Section 3 covers preliminaries and formalizes the problem. In Section 4, the methods for building and using an Eco Road Network with histogram-based eco-weights is proposed, and Section 5 describes how to estimate GHG emissions using the Eco Road Network. Section 6 reports on the experimental results, and Section 7 concludes.

2 Related work

Although much work has been conducted to enable time-dependent (e.g., [4, 5]) and stochastic (e.g., [9]) routing services in different application scenarios, existing proposals

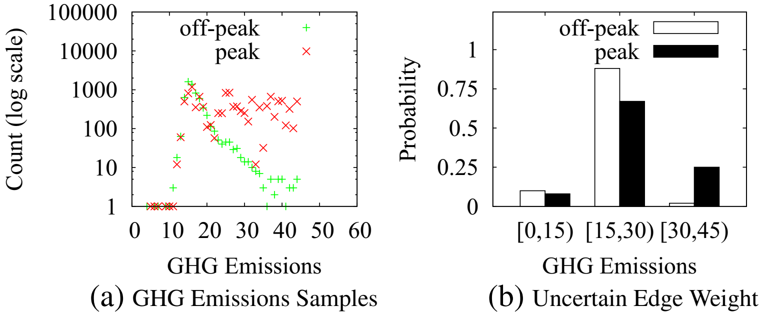


Fig. 2 Time-Dependent Uncertain Edge Weight of An Edge

do not provide a detailed description of how to obtain the time-dependent and uncertain weights, and most studies simply rely on synthetically generated weights.

T-drive [28] is the most relevant work to our study. T-drive aims at providing a fastest-route service based on travel-time weights learned from GPS records obtained from taxis. Our study differs from T-drive in several respects. First, due to its different focus, T-drive does not assign weights to road-network edges, but identifies so-called landmarks and assigns weights to landmark edges that connect pairs of landmarks. This setup makes the T-drive approach inappropriate for general-purpose routing because T-drive suggests sequences of landmark edges, not sequences of road-network edges because landmark edges typically correspond to multiple paths in the road network. In contrast, our study assigns weights to edges and thus provides a general-purpose foundation for stochastic routing. Second, T-drive uses a sequence of histograms as the weight of a landmark edge, where each histogram represents the distribution of travel times during an interval. When doing so, T-drive uses the same buckets for all histograms on a landmark edge. In contrast, our proposal is able to assign different buckets (based on distinct distributions) to the histograms on an edge during different intervals. For instance, more buckets are used for representing more complex distributions, e.g., the GHG emissions distributions during peak hours. Third, T-drive assumes that adjacent landmark edges are independent. In contrast, we consider the travel cost dependencies between adjacent edges, which better captures reality. The experimental studies confirm that our approach achieves better travel cost estimations for routes. Fourth, T-drive does not compute a travel cost histogram for a sequence of landmark edges, but instead computes a single value based on individual drivers' optimism indices. Going beyond the T-drive study, we propose different histogram aggregation methods to aggregate the histograms of the edges in a route such that the travel cost histogram of a route can be computed. Thus, our approach provides a foundation for stochastic routing. Finally, we consider also eco-weights, not only time-weights.

A few recent studies also consider how to derive weights using GPS data. One study [27] covers the estimation of single-valued eco-weights for edges with infrequent or no GPS records. However, it is unable to capture time-dependence and uncertainty. Orthogonal to this study, we assume a setting where edges have considerable amounts of GPS records, and we focus on capturing detailed GHG emissions distributions during different intervals for the edges. Another recent study [21] concerns the update of near-future (e.g., the next 15-min or 30-min) eco-weights based on incoming real-time GPS data. In contrast, we capture time-dependent GHG emissions for longer periods, e.g., a day, a week, or a month, based on historical GPS data. As a result, our work is complementary to that study. Further, some studies propose methods to enable time-dependent weights [13, 14] using various traffic sensor data, but they do not consider the uncertainty of the weights.

3 Problem setting and definition

We proceed to cover the problem setting and to formalize the problem.

3.1 Time-dependent histograms

Given a multiset of cost values C, the range of the cost values $Range(C)$ is the set of non-duplicated values that occur in C [11]. The *data distribution* of the cost values in C, denoted as $DD(C)$, is a set of $(val, prob)$ pairs, where val indicates a value in $Range(C)$, and $prob$ is the number of occurrences of the value in C divided by the total number of values in C.

An example is shown as follows, where multiset C contains GHG emission values observed from an edge.

$$\begin{aligned} C &= \{\{5, 8, 10, 20, 15, 10, 20, 20, 34, 28\}\}; \\ Range(C) &= \{5, 8, 10, 15, 20, 28, 34\}; \\ DD(C) &= \{(5, 0.1), (8, 0.1), (10, 0.2), (15, 0.1), \\ &\quad (20, 0.3), (28, 0.1), (34, 0.1)\}. \end{aligned}$$

In particular, a histogram $H = \langle (b_1, p_1), \ldots, (b_n, p_n) \rangle$ is a vector of (*bucket*, *probability*) pairs, where a *bucket* $b_i = [f_i, l_i)$ indicates a range of cost values, where f_i and l_i indicate the starting and ending values of the range. The buckets are disjoint, i.e., $b_i \cap b_j = \emptyset$ if $i \neq j$; and all elements in $Range(C)$ belong to the union of the buckets, i.e., $b_1 \cup \ldots \cup b_n$. The width of a bucket is defined as $|b_i| = l_i - f_i$. If every bucket in a histogram has the same width, the histogram is an *equi-width histogram.* A *probability* p_i records the percentage of the cost values that are in the range indicated by b_i. The sum of all probabilities is 1, i.e., $\sum_{i=1}^{n} p_i = 1$.

Next, two equi-width histograms H_1 and H_2 are *isomorphic* if they have the same number of buckets representing the same data range. However, the probabilities of corresponding pairs of buckets may still be different.

Given a time interval of interest *TI*, a *Time Dependent Histogram* is a vector of (*period*, *histogram*) pairs, where the time interval *TI* is partitioned into *periods*. Specifically, in a time dependent histogram $tdh = \langle (T_1, H_1), \ldots, (T_m, H_m) \rangle$, *period* T_i is a period in *TI*, and histogram H_i is the histogram of the cost values observed in period T_i. The periods partition the time interval of interest, i.e., $T_1 \cup \ldots \cup T_m = TI$.

3.2 Road networks and trajectories

An *Eco Road Network* (*ERN*) is a weighted, directed graph $G = (V, E, F)$, where V and E are vertex and edge sets. A vertex $v_i \in V$ models a road intersection or the end of a road, and an edge $e_k = (v_i, v_j) \in E$ models a directed road segment that enables travel from vertex v_i to vertex v_j. Function $F : E \rightarrow TDH$ in G assigns time-dependent and uncertain eco-weights to edges in E; and TDH is the set of all possible time dependent histograms.

A *Trajectory* $trj = \langle r_1, r_2, \ldots, r_x \rangle$ is a sequence of GPS records. Each GPS record r_i specifies the location (typically with latitude and longitude coordinates) and velocity of a vehicle at a particular time $r_i.t$. Furthermore, the GPS records in a trajectory are ordered based on their timestamps. Given the road network where the trajectories occurred, a GPS record in a trajectory can be mapped to a specific location on an edge in the road network using some map-matching algorithm [19].

3.3 Computing GHG emissions

GHG emissions are computed using vehicular environmental impact models [6, 7]. Specifically, such models take as input a vehicle's instantaneous speeds, average speeds, instantaneous accelerations, travel distance, road grades [24], and vehicle type (e.g., passenger car, truck), and output GHG emissions for the vehicle and trip considered. Most of the input such as instantaneous speeds, average speeds, instantaneous accelerations, and travel distance, can be derived from GPS trajectories, and the vehicle type is typically obtained from the meta-data of the GPS data.

In the transportation research literature, a dozen different vehicular environmental impact models have been presented. In this study, we use the VT-micro model to compute the GHG emissions because a recent benchmark study indicates that it is the most accurate model [6]. Essentially, VT-micro is a polynomial function of instantaneous speeds and accelerations that returns GHG emissions. Specifically, VT-micro takes as input the instantaneous velocity v_t (km/h) and acceleration a_t $(km/h/s)$, and it estimates GHG emissions f_t (mg/s) at time point t as follows.

$$f_t = \begin{cases} \exp(\sum_{i=0}^{3}\sum_{j=0}^{3}(K_{i,j} \cdot v_t^i \cdot a_t^j)) & \text{if } a_t \geqslant 0 \\ \exp(\sum_{i=0}^{3}\sum_{j=0}^{3}(L_{i,j} \cdot v_t^i \cdot a_t^j)) & \text{if } a_t < 0, \end{cases}$$

where $K_{i,j}$ and $L_{i,j}$ are model coefficients for accelerating ($a_t \geqslant 0$) and decelerating ($a_t < 0$) conditions, respectively. The coefficients are calibrated based on vehicle types, etc. [29].

Although we use VT-Micro in the study, the paper's proposal on enabling time-dependent and uncertain eco-weights does not dependent on the specific choices of vehicular environmental impact model, and users can choose any vehicular environmental impact model that fits their needs to compute GHG emissions and thus enable the ERN.

Since different types of vehicles emits significantly different amounts of GHG, it is necessary to build different ERNs for different types of vehicles. In the experiments, all the GPS trajectories are collected from normal passenger cars, and the resulting ERN is applicable only to passenger cars, not to, e.g., trucks. However, the paper's proposal can be applied straightforwardly to generate the ERN for trucks provided that GPS data from trucks is available.

3.4 Problem definition and solution framework

Given a set *TRJ* of map matched trajectories in a road network $G' = (V, E, null)$, the paper studies how to obtain the corresponding Eco-Road Network $G = (V, E, F)$. Specifically, the key task is to determine $G.F$, which assigns time dependent histograms to edges, based on trajectory set *TRJ*.

An overview of the framework that determines $G.F$ is shown in Fig. 3. The pre-processing module transforms the map matched trajectories into a set *TRR* of *traversal records* of the form $trr = (e, t_s, tt, ge, trj_j)$. A traversal record r indicates that edge e is traversed by trajectory trj_j starting at time t_s. The travel time and the GHG emissions of the traversal are tt and ge, respectively. The travel times can be derived directly from the GPS records as the difference between the times of the first and last GPS records. Different vehicular environmental impact models [7] can be applied to compute the GHG emissions from the GPS records, as discussed in Section 3.3.

After pre-processing, each edge e_i is associated with a set of traversal records $TRR_i = \{trr \in TRR | r.e = e_i\}$.

The ERN construction module builds initial time-dependent histograms for edges based on their traversal records. Maintaining the time-dependent histograms of all edges in a large road network may incur a large storage overhead. To reduce the overhead, approximation and compression techniques are employed to reduce both the number of (*period*, *histogram*) pairs in the time-dependent histograms and the number of the buckets in individual histograms. Specifically, *histogram merging* and *bucket reduction* are applied to obtain a compact representation of an ERN.

Next, *virtual edge and extended virtual edge generation* module identifies virtual edges (i.e., pairs of adjacent edges whose GHG emissions are dependent) and extended virtual

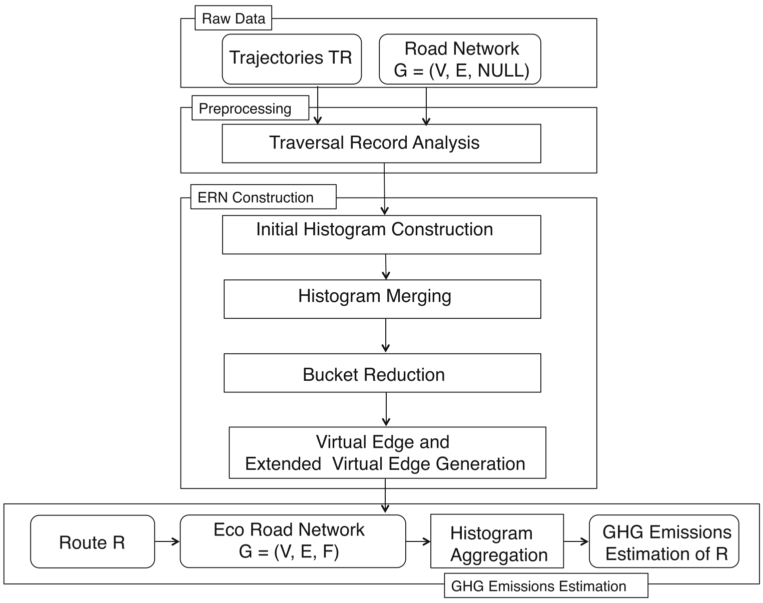

Raw Data
Trajectories TR
Road Network G = (V, E, NULL)
Preprocessing
Traversal Record Analysis
ERN Construction
Initial Histogram Construction
Histogram Merging
Bucket Reduction
Virtual Edge and Extended Virtual Edge Generation
Route R
Eco Road Network G = (V, E, F)
Histogram Aggregation
GHG Emissions Estimation of R
GHG Emissions Estimation

Fig. 3 Framework Overview

edges (i.e., sequences of adjacent edges whose GHG emissions are dependent). The module also assigns time-dependent histograms for virtual edges and extended virtual edges based on the trajectories occurred on them. Finally, the Eco-Road Network G is returned, where the function $G.F$ assigns compact, time-dependent, uncertain eco-weights to edges, virtual edges, and extended virtual edges.

Based on the obtained ERN, the GHG emissions distribution of any given route R can be estimated based different histogram aggregation methods.

4 ERN construction

We propose methods to generate an *Eco-Road Network* from traversal records.

4.1 Initial time dependent histograms

An initial time dependent histogram is built for every edge $e_i \in E$ based on the traversal records associated with e_i, i.e., R_i. Given a time interval of interest *TI*, e.g., a day or a week, and the finest temporal granularity α, e.g., 15 minutes or one hour, *TI* is split into $\lceil \frac{TI}{\alpha} \rceil$ periods, where the j-th period T_j is $[(j-1) \cdot \alpha,\ j \cdot \alpha)$. For each T_j, an equi-width histogram H_j is built based on the traversal records that occurred in the period, i.e., $R_i^{(j)} = \{r | r.e = e_i \wedge r.t_s \in T_j\}$.

To ease the following histogram compression operations, we make sure the initial histograms on edge e_i are isomorphic. The initial histograms share the same range $[l, u]$ where l and u are the lowest and highest GHG emissions (or travel times) observed in R_i. Further, the same number of buckets N_{bucket} is used for all histograms, where N_{bucket} is an configurable parameter. Thus, $\lceil \frac{TI}{\alpha} \rceil$ isomorphic histograms are obtained for each edge, where each histogram has N_{bucket} buckets.

Assuming α is set to 1 hour, Fig. 4a shows two isomorphic histograms during periods [8 a.m., 9 a.m.) and [9 a.m., 10 a.m.) for an edge in North Jutland, Denmark. The high similarity of the two histograms motivates us to compress them into one histogram with little loss of information. We proceed to show how to compress the histograms using histogram merging and bucket reduction. Our methods are configurable so that histogram accuracy can be controlled.

4.2 Histogram merging

If two temporally adjacent histograms H_i and H_{i+1} represent similar data distributions, it is potentially attractive to merge the two histograms [16] into one histogram $\overline{H}$ that represents the data distribution for the longer period $\overline{T} = T_i \cup T_{i+1}$.

Given two distributions, several techniques exist to measure their similarity, such as cosine similarity, the K-S test, and the χ-square test. The simplicity and efficiency of computing cosine similarity makes it appropriate for evaluating the similarity of two histograms.

To facilitate the use of cosine similarity, we treat a histogram as a vector of probabilities. A histogram $H = \langle (b_1, p_1), \ldots, (b_n, p_n) \rangle$ has the vector $V(H) = \langle p_1, \ldots, p_n \rangle$. Since the initial histograms are isomorphic and equi-width, they have the same number of buckets, and each bucket in a corresponding pair has the same range. Thus, all the vectors are isomorphic, meaning that they have the same number of dimensions, with each dimension representing the same entity, i.e., the probability in a particular sub-range. The *similarity* between two histograms is defined by Eq. 1.

$$sim(H_i, H_j) = \frac{V(H_i) \odot V(H_j)}{\|V(H_i)\| \cdot \|V(H_j)\|}, \tag{1}$$

where $\odot$ indicates dot product between two vectors, $\cdot$ indicates the product between two reals, and $\|V\|$ indicates the magnitude of vector V.

When the similarity of adjacent isomorphic histograms H_i and H_{i+1} exceeds a threshold T_{merge}, they are merged into a new histogram $\overline{H}$. The weight of H_i is $W_i = \frac{H_i.c}{H_i.c + H_{i+1}.c}$, where $H_i.c$ is the total number of cost values that are used to derive H_i, which is equivalent to the number of traversal records in the i-th period. The probability value for the k-th bucket in $\overline{H}$ is given by Eq. 2.

$$\overline{H}.p_k = H_i.p_k \cdot W_i + H_{i+1}.p_k \cdot W_{i+1}, \forall k \in [1, n] \tag{2}$$

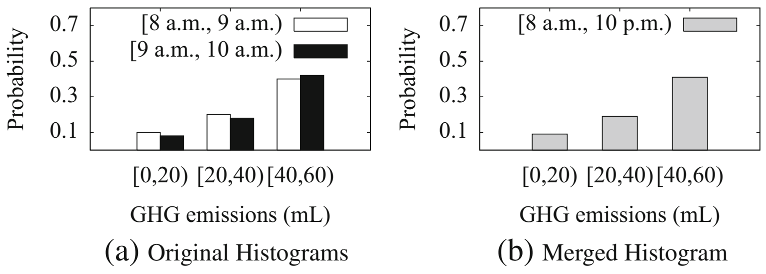


Fig. 4 Histogram Merging Example

When merging two isomorphic histograms, the obtained histogram $\overline{H}$ is isomorphic to H_i and H_{i+1}. The probability given by the k-th bucket in $\overline{H}$ is not just the average of $H_i.p_k$ and $H_{i+1}.p_k$. As the number of traversal records in the k-th buckets of histograms H_i and H_{i+1} may be different, we use the weighted average to construct the data probability of the k-th bucket in $\overline{H}$, as shown in Eq. 2. We also maintain $\overline{H}.c = H_i.c + H_{i+1}.c$ so that the count of traversal records in $\overline{H}$ is available for subsequent merging steps.

Given an initial time-dependent histogram *tdh* for an edge, a corresponding merged time-dependent histogram $\overline{tdh}$ is computed iteratively. In each iteration, a pair of adjacent histograms with the highest histogram similarity is identified. If the similarity exceeds a user-defined threshold T_{merge}, the two histograms are merged according to Eq. 2, and the union of the two argument histograms' periods becomes the period of the new histogram. The iteration terminates when T_{merge} exceeds the highest histogram similarity. For example, the two histograms for adjacent periods shown in Fig. 4a are merged into the single histogram shown in Fig. 4b.

Assume that an initial time-dependent histogram has m histograms and each histogram has n buckets, where $m = \lceil \frac{TI}{\alpha} \rceil$, TI is the time interval of interest, and α is the finest temporal granularity. The worst case asymptotic run-time of histogram merging is then $O(m^2 \cdot n)$.

4.3 Bucket reduction

Histogram merging reduces the numbers of histograms. Bucket reduction reduces the sizes of individual histograms, which is orthogonal to histogram merging.

Bucket reduction transforms a histogram H into a new histogram $\widehat{H}$ that approximates the data distribution represented by H using fewer buckets [10, 11, 16]. $\widehat{H}$ is not necessarily equi-width, meaning that different buckets may have different widths. Histogram regression is conducted by merging two adjacent buckets. The range of the new bucket is the union of the range of two original buckets, and the probability of the buckets is the sum of the probabilities of the two original buckets. Thus, given a histogram $H = \langle (b_1, p_1), \ldots, (b_i, p_i), (b_{i+1}, p_{i+1}), \ldots, (b_n, p_n) \rangle$, after merging buckets b_i and b_{i+1}, the new histogram is $\widehat{H} = \langle (b_1, p_1), \ldots, (b_{i-1}, p_{i-1}), (b_x, p_x), (b_{i+2}, p_{i+2}), \ldots, (b_n, p_n) \rangle$, where $b_x = b_i \cup b_{i+1}$ and $p_x = p_i + p_{i+1}$.

The sum of squared error (*SSE*) is employed to measure the discrepancy between the original histogram H and the histogram after bucket reduction $\widehat{H}$. Since the error is only introduced by the buckets we merge and we merge only two adjacent buckets b_i and b_{i+1}, the error introduced by this operation is given by Eq. 3.

$$SSE(H, \widehat{H}) = \left(\frac{|H.b_i|}{|H.b_i| + |H.b_{i+1}|} \widehat{H}.p_x - H.p_i \right)^2 + \left(\frac{|H.b_{i+1}|}{|H.b_i| + |H.b_{i+1}|} \widehat{H}.p_x - H.p_{i+1} \right)^2, \quad (3)$$

where $|H.b_i|$ is the range of the i-th bucket in histogram H, and $H.p_i$ is the probability of the i-th bucket in histogram H. The accuracy of the histograms is defined to be the deviation of the distribution described by the histograms from the distribution of the original data, and our goal is to achieve small deviations, which indicate small accuracy losses. Moreover, a smaller *SSE* indicates that $\widehat{H}$ achieves a smaller accuracy loss compared to the original histogram H.

We consider a scenario where a storage budget (i.e., a number of buckets) for an edge is given, and where we need to decide how to merge the buckets in the histograms that represent the GHG emissions for different periods so that we maximize the overall accuracy. For example, rather than distributing the buckets uniformly, we may use higher (lower) bucket budgets for histograms representing peak hours (off-peak hours).

Given a merged time dependent histogram $\overline{tdh}$ of an edge and a reduction threshold T_{red} indicating the total number of buckets available for the edge, Algorithm 1 describes how to obtain a time dependent histogram that meets the storage budget while achieving least accuracy loss. Note that for different edges, the reduction threshold T_{red}, i.e., the bucket budget, may be different. A simple heuristic is to assign higher bucket quotas to edges that have many merged histograms in their time dependent histograms after histogram merging. The number of buckets used for an edge e is proportional to the number of histograms associated with e.

Algorithm 1 HistogramBucketReduction

```
Input:
 1: Merged time-dependent histogram of edge e: $\overline{tdh}$ =
 2: $\langle(\overline{T}_1, \overline{H}_1), \ldots, (\overline{T}_{m'}, \overline{H}_{m'})\rangle$;
 3: Bucket reduction threshold: $T_{red}$;
Output:
 4: Final time-dependent histogram of edge e: $\overline{tdh}$
 5: while total buckets in $\overline{tdh}$ exceeds $T_{red}$ do
 6:     $minSSE \leftarrow \infty$;
 7:     for each histogram $\overline{H}_i$ do
 8:         for each adjacent buckets $\overline{H}_i.b_j$ and $\overline{H}_i.b_{j+1}$ in $\overline{H}_i$ do
 9:             Generate candidate histogram $H_i'$ by merging $b_j$ and $b_{j+1}$ in $\overline{H}_i$;
10:             if $SSE(\overline{H}_i, H_i') < minSSE$ then
11:                 Record the pair buckets in MinPairBuckets;
12:                 $minSSE \leftarrow SSE(\overline{H}_i, H_i')$;
13:             end if
14:         end for
15:     end for
16:     Merge the buckets pair in MinPairBuckets;
17: end while
18: return $\overline{tdh}$;
```

Algorithm 1 works iteratively. For each iteration, it linearly scans all adjacent buckets pairs and finds the pair that achieves the smallest *SSE* to merge (lines 2–9). Note that to identify two buckets that need to be merged, every histogram in the given $\overline{tdh}$ has to be checked. This process terminates when the total number of buckets for the edge is below the reduction threshold T_{red}.

Alternatively, a priority queue Q can be used, where an element is an adjacent bucket pair and the priority of the element is the *SSE* value of merging the two adjacent buckets. We do not use such a priority queue because maintaining the priority queue is complex when a pair of adjacent buckets are merged into one bucket. For example, assuming that the highest-priority element is $p = \langle b_1, b_2\rangle$, after merging b_1 and b_2 into a new bucket b', if there exist elements containing b_1 or b_2 in Q, i.e., $e_1 = \langle b_0, b_1\rangle$ and $e_1 = \langle b_2, b_3\rangle$, these should be updated to $e_1' = \langle b_0, b'\rangle$ and $e_1' = \langle b', b_3\rangle$.

Assume that a merged time-dependent histogram $\overline{tdh}$ consists of m' histograms each with n buckets. The worst case run-time of bucket reduction is then $O(m'^2 \cdot n^2)$.

5 GHG emissions estimation

After histogram merging and bucket reduction, we obtain an ERN where each edge is associated with a compact time dependent histogram. Here, we study how to use the obtained ERN to estimate the GHG emissions of traversing a route.

Rather than estimating a single value for a traversal of a route, e.g., the expected GHG emissions, we estimate the GHG emissions distribution using histograms. This yields much more detailed information than a single value and is useful in many applications, e.g., stochastic route planing [15, 25] and probabilistic threshold-based routing [9].

The distribution of the GHG emissions of a traversal on a route is estimated based on the ERN. In particular, the distribution of the GHG emissions of the traversal, also represented by a histogram, is achieved by aggregating pertinent (w.r.t. the traversal time) histograms of *sub-routes* of the route, where a sub-route is an edge or a sequence of adjacent edges if the edges have highly dependent GHG emissions. In particular, if the starting traversal time of a sub-route with time-dependent histogram $tdh = \langle (T_1, H_1), \ldots, (T_m, H_m) \rangle$ is t, histogram H_i is selected for histogram aggregation if $t \in T_i$.

5.1 Modeling dependence among adjacent edges

When aggregating the histograms of the edges in a route, we need to consider the dependencies of the GHG emissions distributions of adjacent edges. A route $R = \langle e_1, e_2, \ldots, e_n \rangle$ is a sequence of edges where the ending vertex of e_i is the starting vertex of e_{i+1}, for $1 \leqslant i \leqslant n-1$. The adjacent edges in a route are the edges e_i and e_{i+1}, where $1 \leqslant i \leqslant n-1$.

5.1.1 Dependence analysis

Most existing studies [3, 15, 18, 20] assume that the travel costs (e.g., travel times) of adjacent edges are independent. To evaluate this assumption, we conducted an empirical study on a collection of frequently traversed adjacent edge pairs. Specifically, we identified 82 edge pairs that each is traversed by at least 1,000 trajectories.

The GHG emissions distributions of two adjacent edges are modeled as two random variables X and Y, and the normalized mutual information $\mathbf{NMI}(X, Y)$ is applied to quantify the dependency between X and Y, as defined in Eq. 4.

$$\begin{aligned} \mathbf{MI}(X, Y) &= \sum_{y \in Y} \sum_{x \in X} p(x, y) \cdot \log\left(\frac{p(x,y)}{p(x)p(y)}\right) \\ \mathbf{NMI}(X, Y) &= 2 \cdot \frac{\mathbf{MI}(X,Y)}{ET(X)+ET(Y)}, \end{aligned} \tag{4}$$

where $\mathbf{MI}(X, Y)$ is the mutual information between X and Y, which quantifies the mutual dependency between the two random variables, $ET(X)$ denotes the entropy of random variable X, and $\mathbf{NMI}(X, Y)$ is the normalized mutual information between X and Y. We use mutual information to represent the degree of GHG emissions dependency between two edges, as this enable us to identify both linear dependency and non-linear dependency. Further, the use of normalized mutual information values makes it easier to evaluate and visualize the degree of dependency between two random variables. The **NMI** values are normalized to the range [0, 1]. An **NMI** value of 0 means that the two edges are independent, and the larger the **NMI** value, the higher the dependency.

Figure 5 shows the percentage of edge pairs w.r.t. different ranges of **NMI** values, it shows the **NMI** values for both the scenarios where time dependence is considered (with time) and not considered (without time). It suggests that most adjacent edges tend to be

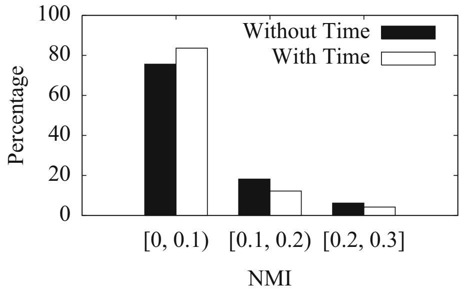


Fig. 5 NMI Study

independent, which complies with the independence assumption. However, some adjacent edges do have non-negligible dependencies, as indicated by the last two buckets in Fig. 5.

It is also of interest to further investigate the dependency between two edges when taking the times of the traversals into consideration. For each edge pair, we partition the time domain according to the intervals obtained from both edges' corresponding time dependent histograms. For each partition, we compute **NMI** using only the trajectories that occurred in the partition. The results, also shown in Fig. 5, suggest that the GHG emissions dependencies between adjacent edges are reduced when we take into account the traversal times. However, some adjacent edges still have relatively high dependencies (shown by the last two buckets).

5.1.2 Virtual edges

The above study shows that the independence assumption does not always hold. To better model dependencies and thus improve on the state-of-the-art, we introduce the notion of a *virtual edge* for each pair of dependent adjacent edges (i.e., edges whose **NMI** exceeds the dependency threshold T_{dep}). It is worth noting that using a smaller T_{dep} results in more virtual edges being generated in the ERN and also results in more preprocessing. As for normal edges, time dependent histograms can be obtained that represent the distributions of GHG emissions and travel times for virtual edges.

Figure 6 shows an example with a simple road network, along with numbers of trajectories in the road network, shown in Table 1. For example, 200 trajectories traversed e_4 (the 5-th line in Table 1). After traversing e_4, 140 trajectories continued on e_5 (the 6-th line in Table 1).

Table 2 shows the GHG emissions dependencies between adjacent edges. For example, to check whether edges e_4 and e_5 are dependent, we use the traversal records from the 140 trajectories that traversed both e_4 and e_5 to compute the **NMI**. As shown in the last line of Table 2, e_4 and e_5 have a high **NMI** value (the default dependency threshold T_{dep} is 0.2), and thus they are considered as dependent. A virtual edge e_v representing e_4 and e_5 is created.

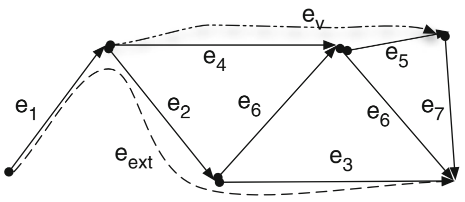


Fig. 6 Edges, Virtual Edges, and Extended Virtual Edges

Table 1 Number of Trajectories

Trajectory	Trajectory count
$\langle e_1, e_2\rangle$	40
$\langle e_1, e_3\rangle$	60
$\langle e_1, e_2, e_3\rangle$	5
$\langle e_4\rangle$	200
$\langle e_4, e_5\rangle$	140
$\langle e_4, e_6\rangle$	50

Our empirical study in Section 6 investigates the effect of varying the T_{dep} value. The time dependent histograms of the virtual edge e_v is generated based on the 140 trajectories that traversed both e_4 and e_5. In contrast, we use all 200 trajectories that traversed e_4 to generate the time dependent histograms of edge e_4.

5.1.3 Extended virtual edges

Next, it is possible that more than two adjacent edges exist such that each adjacent edge pair is dependent. We use an *extended virtual edge* to represent such edges. For instance, e_1 and e_2 are dependent, and so are e_2 and e_3. Thus, an extended virtual edge e_{ext} is used to represent e_1, e_2, and e_3.

Intuitively, it is possible to compute a time dependent histogram for an extended virtual edges based on the trajectories that use the extended virtual edges. However, this approach becomes unattractive due to two reasons. First, as the number of edges in an extended virtual edge increases, the number of trajectories that use the extended virtual edge decreases quickly. Based on an analysis on more than 200 million GPS records collected in Denmark over two years, we show that the average trajectory counts that cover n (where $1 \leq n \leq 5$) consecutive edges in Fig. 7.

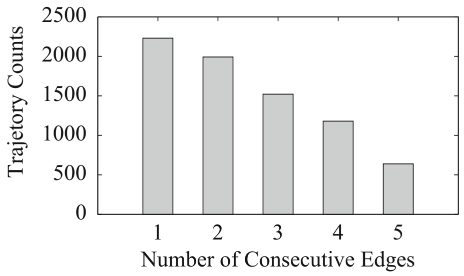
If we compute time dependent histograms of extended virtual edges using the corresponding trajectories, we risk to obtain biased distributions and then perform over-fitting to small amounts of trajectories. For example, if we compute a time dependent histogram for e_{ext} in Fig. 6, we may be over-fitting to the 5 trajectories (the third line in Table 1). Second, it is impractical to store the time dependent histograms for all extended virtual edges as this calls for excessive storage.

To contend with this problem, we propose a method that estimates GHG emissions for an extended virtual edge $e_{ext} = (e_1, e_2, \ldots, e_N)$, based on all the virtual edges in e_{ext}. Based on the trajectories that use a virtual edge $e_v = (e_1, e_2)$ in the extended virtual edge,

Table 2 GHG Emissions Dependency

Edge pairs	Dependency
(e_1, e_2)	0.2
(e_2, e_3)	0.25
(e_4, e_6)	0.05
(e_1, e_4)	0.03
(e_4, e_5)	0.23

Fig. 7 Trajectory Counts



the **joint probability** of e_1 and e_2 is used to represent the correlation between the GHG emissions on e_1 and e_2, denoted as $f_{e_1,e_2}(b_1^i, b_2^j) = Pr[G_{e_1} = b_1^i, G_{e_2} = b_2^j] = Pr[G_{e_2} = b_2^j, G_{e_1} = b_1^i]$, where b_i^j is the j-th bucket in the histogram of the i-th edge e_i, i.e., H_i; thus, $G_{e_i} = b_i^j$ indicates that the GHG emissions on e_i are within the width of the i-th bucket in e_i's histogram.

The **conditional probability** of GHG emissions on edge e_1 yields b_1^i given the GHG emissions on edge e_2 is b_2^j is given in Eq. 5.

$$f_{e_1|e_2}(b_1^i|b_2^j) = \frac{f_{e_1,e_2}(b_1^i, b_2^j)}{f_{e_2}(b_2^j)} \tag{5}$$

The estimated GHG emissions of a route $R = \langle e_1, \ldots, e_n \rangle$ is considered as a range and is computed using the estimation of GHG emissions ranges of all edges in R, and R_m represents the first m edges in R. By using one bucket from each of the n histograms, we get a **bucket sequence** $BS = (b_1^i, b_2^j, \ldots, b_{n-1}^p, b_n^q)$. Specifically, operator $+$ computes the union of two ranges, i.e., given two ranges $b_1 = [l_1, r_1)$ and $b_2 = [l_2, r_2)$, $b_1 + b_2 = [l_1 + l_2, r_1 + r_2)$. Thus, the range of BS can be computed as $|BS| = b_1^i + b_2^j + \ldots + b_{n-1}^p + b_n^q$, which is the sum of all the buckets in one bucket sequence. The probability of traversing R that yields $|BS|$ is represented as $f_R(|BS|)$. Formally, $f_R(|BS|)$ is given by Eq. 6.

$$\begin{aligned}
f_{R_n}(|BS|) &= f_{R_n}(b_1^i, b_2^j, \ldots, b_{n-1}^p, b_n^q) \\
&= f_{e_n|e_{n-1}}(b_n^q|b_1^i, b_2^j, \ldots, b_{n-1}^p) \cdot f_{R_{n-1}}(b_1^i, b_2^j, \ldots, b_{n-1}^p) \\
&= f_{e_n|e_{n-1}}(b_n^q|b_{n-1}^p) \cdot f_{R_{n-1}}(b_1^i, b_2^j, \ldots, b_{n-1}^p) \\
&= \frac{f_{e_n,e_{n-1}}(b_{n-1}^p, b_n^q)}{f_{e_{n-1}}(b_{n-1}^p)} \cdot f_{R_{n-1}}(b_1^i, b_2^j, \ldots, b_{n-1}^p) \\
&= \ldots \\
&= \frac{\prod_{i=1}^{n-1} f_{e_i,e_{i+1}}(b_i^j, b_{i+1}^k)}{\prod_{i=2}^{n-1} f_{e_i}(b_i^k)}
\end{aligned} \tag{6}$$

To get the GHG emissions histograms of n adjacent edges with dependence, we have to enumerate every bucket sequence using one bucket from each edge's histogram and compute the GHG emission of each sequence. If each of these n histograms has M buckets, the number of bucket combinations is M^n. Using the lower bound L and the upper bound U of the result histogram H_R, which are the sum of the lower bounds and the sum of the upper bounds of n histograms, to construct the result histogram H_R as an equi-width histogram of

Table 3 Sample GHG Emissions

Edge	GHG emissions: (bucket: probability)	
	Bucket 1	Bucket 2
e_1	[20,30):0.2	[30,40):0.8
e_2	[20,30):0.5	[30,40):0.5
e_3	[15,25):0.3	[25,35):0.7
e_4	[10,20):0.4	[20,30):0.6

M buckets, we distribute the probabilities obtained from all the M^n bucket unions into the buckets in H_R. Equation 7 shows how the probabilities of bucket unions are distributed to a bucket B in H_R.

$$f(B) = \sum \frac{|B \cap BS|}{|BS|} \cdot f_R(BS), \forall BS\ (|B| \cap |BS| \neq \emptyset) \tag{7}$$

Consider the route $R = \langle e_1, e_2, e_3 \rangle$ in the road network shown in Fig. 6. The marginal GHG emissions on all the edges are shown in Table 3. The joint GHG emissions probabilities of (e_1, e_2) and (e_2, e_3) are shown in Table 4.

Table 5 shows the bucket sequences of R together with their GHG emission ranges and probabilitie, where the resulting histogram H_R is an equi-with histogram that consists of two buckets B_1 and B_2 whose ranges are [55, 85) and [85, 115).

Take B_1 as an example. Its probability to get its range can be computed as follows.

$$\begin{aligned} f_R(|B_1|) &= \frac{|[55, 85) \cap [55, 85)|}{|[55, 85)|} \cdot 0.04 + \frac{|[55, 85) \cap [65, 95)|}{|[55, 85)|} \cdot 0.06 \\ &+ \frac{|[55, 85) \cap [65, 95)|}{|[55, 85)|} \cdot 0.02 + \frac{|[55, 85) \cap [75, 105)|}{|[55, 85)|} \cdot 0.08 \\ &+ \frac{|[55, 85) \cap [65, 95)|}{|[55, 85)|} \cdot 0.08 + \frac{|[55, 85) \cap [75, 105)|}{|[55, 85)|} \cdot 0.32 \\ &+ \frac{|[55, 85) \cap [75, 105)|}{|[55, 85)|} \cdot 0.08 \\ &= \frac{30}{30} \cdot 0.04 + \frac{20}{30} \cdot 0.06 + \frac{20}{30} \cdot 0.02 + \frac{10}{30} \cdot 0.08 + \frac{20}{30} \cdot 0.08 \\ &+ \frac{10}{30} \cdot 0.32 + \frac{10}{30} \cdot 0.08 \\ &= 0.31 \end{aligned}$$

5.1.4 Sub-routes

A route R can be split into a set of sub-routes, as described in Algorithm 2, where a sub-route is an edge, a virtual edge, or an extended virtual edge. Since we model adjacent edges

Table 4 Joint GHG Emissions

(a) $f_{e_1,e_2}(b_1^i, b_2^j)$			(b) $f_{e_2,e_3}(b_2^i, b_3^j)$	
	[20,30)	[30,40)	[15,25)	[25,35)
[20, 30)	0.1	0.1	0.2	0.3
[30, 40)	0.4	0.4	0.1	0.4

Table 5 GHG Emissions for Bucket Sequences

Bucket sequence	GHG emissions range	Probability
[20, 30), [20, 30), [15, 25)	[55, 85)	0.04
[20, 30), [20, 30), [25, 35)	[65, 95)	0.06
[20, 30), [30, 40), [15, 25)	[65, 95)	0.02
[20, 30), [30, 40), [25, 35)	[75, 105)	0.08
[30, 40), [20, 30), [15, 25)	[65, 95)	0.08
[30, 40), [20, 30), [25, 35)	[75, 105)	0.32
[30, 40), [30, 40), [15, 25)	[75, 105)	0.08
[30, 40), [30, 40), [25, 35)	[85, 115)	0.32

with (high) dependencies as virtual edges and extended virtual edges, each adjacent sub-route pair is independent. The next step is to describe how to estimate the GHG emissions for a route given an ERN.

Algorithm 2 RouteToSubRoutes

```
Input:
 1: A route $R = (e_1, e_2, \ldots, e_N)$;
 2:
Output:
 3: A sequence of independent sub-routes $S$;
 4: $i \leftarrow 1$;
 5: while $i \leq N$ do
 6:     if $i < N$ then
 7:         $j \leftarrow i + 1$;
 8:     end if
 9:     if $i = N$ or $e_i$ is not dependent on edge $e_j$ then
10:         Insert $e_i$ into $S$;
11:         $i \leftarrow i + 1$;
12:     else
13:         while $j \leq N$ and $e_{j-1}$ and $e_j$ are dependent do
14:             $j \leftarrow j + 1$;
15:         end while
16:         if $|\langle e_i, \ldots, e_{j-1} \rangle| = 2$ then
17:             Insert the virtual edge $e_v$ that represents edges $\langle e_i, \ldots, e_{j-1} \rangle$ into $S$;
18:         else
19:             Insert the extended virtual edge $e_{ext}$ that ¡represents edges $\langle e_i, \ldots, e_{j-1} \rangle$
    into $S$;
20:         end if
21:         $i \leftarrow j$;
22:     end if
23: end while
24: Return $S$;
```

5.2 Histogram aggregation

Histogram Aggregation takes two histograms H_1 and H_2 and yields a histogram H' that represents the aggregated GHG emissions for traversing both edges. The aggregation of H_1

and H_2 is also a histogram and is denoted as H_{agg}. Algorithm 2 shows how to perform a form of histogram aggregation that we call **bucket-wise histogram aggregation** because it operates on all combinations of two bucktes from each input histogram iteratively. Recall that operator $+$ denotes the union of two ranges, i.e., $[f_1, l_1) + [f_2, l_2) = [f_1 + f_2, l_1 + l_2)$. The buckets in H_{agg} are constructed by combining all bucket pairs from H_1 and H_2, using one bucket from each histogram (Lines 2–4). As there might be overlaps between buckets in H_{agg}, we rearrange the buckets in H_{agg} to combine two buckets with the same data range, and we split a data range if it contains the range of another bucket (line 5); thus, we obtain an equi-width histogram as the final result that contains buckets without overlap and duplicates.

Algorithm 3 BucketWiseAggregate

```
 Input:
     Independent histograms $H_1$ and $H_2$;
Output:
     Aggregated Histogram $H_{agg}$;
 1. for all buckets $H_1[i]$ in $H_1$ do
 2.     for all buckets $H_2[j]$ in $H_2$
 3.         newBucket ← $(H_1[i].bH_2[j].b,\ H_1[i].p \cdot H_2[j].p)$
 4.         add newBucket to $H_{agg}$
 5. Rearrange the generated buckets;
 6. Return $H_{agg}$;
```

To illustrate, consider two histograms $H_1 = \langle([0, 2), 0.2), ([2, 4), 0.8)\rangle$ and $H_2 = \langle([0, 2), 0.4), ([2, 4), 0.6)\rangle$. Before being rearranged, the aggregated result is $H_{agg} = \langle([0, 4), 0.08), ([2, 6), 0.32), ([2, 6), 0.12), ([4, 8)0.48)\rangle$; after being rearranged, the result is $H_{agg} = \langle([0, 2), 0.04), ([2, 4), 0.26), ([4, 6), 0.46), ([6, 8), 0.24)\rangle$.

Alternatively, two other histogram aggregation methods, namely **point-wise aggregation** and **median aggregation**, can be used.

Point-Wise Aggregation: A histogram H is first transformed into a probability mass function, which is represented as a list L of $(val, prob)$ pairs that reflect the data distribution (see Section 3.1) in histogram H. The values here are in the range between the minimum and maximum data values in H. The resolution of the resulting values is customized by changing the histogram aggregate parameter T_{agg} that defines the finest granularity.

For example, consider H_1: if $T_{agg} = 1$, the transformed list is $\langle(0, 0.1), (1, 0.1), (2, 0.4), (3, 0.4)\rangle$; and if T_{agg} is 2, the transformed list is $\langle(1, 0.2), (3, 0.8)\rangle$. After obtaining the lists L_1 and L_2 from histograms H_1 and H_2, the list L' for the aggregation of H_1 and H_2 is given by Eq. 8.

$$L'[v] = \sum p_i.prob \cdot p_j.prob, (p_i \in L_1 \wedge p_j \in L_2 \wedge p_i.val + p_j.val = v) \tag{8}$$

Algorithm 4 shows how point-wise aggregation aggregates two histograms. To illustrate, consider lists $L_1 = \langle(1, 0.2), (3, 0.8)\rangle$ and $L_2 = \langle(2, 0.4), (4, 0.6)\rangle$ as obtained from H_1 and H_2 when $T_{agg} = 2$. Following all the iterations in Algorithm 4 (Lines 2–6), the result list L' is $\langle(3, 0.08), (5, 0.44), (7, 0.48)\rangle$. Initially, $(1, 0.2)$ in L_1 means that the probability of range $[1, 1 + T_{agg}) = [0, 2)$ is 0.2 and $(2, 0.4)$ in L_2 means that the probability of $[2, 4)$ is 0.4; so after aggregation, $(3, 0.08)$ in L' means that the probability of $[3, 7)$ is 0.08. Based on the probability derived from L', the result histogram H' is $\langle([3, 5), 0.04), ([5, 7), 0.26), ([7, 11), 0.7)\rangle$.

Algorithm 4 PointWiseAggregate

```
Input:
    Independent histograms $H_1$ and $H_2$;
    Aggregate resolution: $T_{agg}$;
Output:
    Aggregated histogram $H'$;
1. $L_1 \leftarrow Trans(H_1, T_{agg})$, $L_2 \leftarrow Trans(H_2, T_{agg})$;
2. for all $p_i$ in $L_1$
3.     for all $p_j$ in $L_2$ do
4.         $L'[p_i.val + p_j.val] \leftarrow L'[p_i.val + p_j.val] + p_i.prob \cdot p_j.prob$;
5. Generate the result histogram $H'$ that approximates the data distribution derived from
   $L'$;
6. Return $H'$;
```

Median aggregation The performance of point-wise aggregation deteriorates quickly as the number of edges in a route grows, which makes it unattractive for long routes. By making changes to point-wise aggregation, we obtain an alternative method with better performance and reasonable accuracy. When we transform a histogram into a list of $(val, prob)$ pairs, median aggregation uses the median point of the bucket, instead of using all points in the bucket. For example, H_1 and H_2 are transformed into $L_1 = \langle(2, 0.2), (4, 0.8)\rangle$ and $L_2 = \langle(3, 0.4), (5, 0.6)\rangle$. The remaining steps of median aggregation are identical to those in point-wise aggregation. Thus, the result list L' is $\langle(5, 0.08), (7, 0.44), (9, 0.48)\rangle$. Similar to point-wise aggregation, $(2, 0.2)$ in L_1 means that the probability of range $[1, 3)$ is 0.2 and $(3, 0.4)$ in L_2 means that the probability of $[2, 4)$ is 0.4; and after aggregation, $(5, 0.08)$ in L' means that the probability of $[3, 7)$ is 0.08. Thus, the final histogram H' derived from L' is $\langle([3, 5), 0.04), ([5, 9), 0.72), ([9, 11), 0.24)\rangle$.

5.3 GHG emissions estimation for routes

Assume that a traversal uses a particular route at a particular time. The distribution of GHG emissions of this traversal is estimated by aggregating the pertinent histograms (w.r.t. the traversal time) of the independent sub-routes in the route. To choose the pertinent histogram for a sub-route *sr*, we need to know when the traversal of *sr* starts. Since the travel times of the sub-routes before *sr* are uncertain, the traversal starting time of *sr* is also uncertain.

Take a traversal on route $R = \langle e_1, e_4\rangle$ in Fig. 6 as a running example. Figures 8 and 9 show the distributions of the two edges' GHG emissions and travel times, respectively. When starting at 8:58 a.m., the confidences that the traversal arrives at e_4 before and after 9 a.m. are both 0.5 according to Fig. 8a. As the GHG emissions distributions on edge e_4 are different before and after 9 a.m., as shown in Fig. 9b, histogram aggregation is done separately for the two periods. Thus, different arrival times at e_4 may result in the need to consider different GHG emissions histograms.

Based on the travel time on e_1, the aggregated GHG emissions histograms for traversing $R = \langle e_1, e_4\rangle$ is represented as $H_1 = \langle([10, 30), 0.1), ([30, 50), 0.35), ([50, 70), 0.4), ([70, 90), 0.15)\rangle$ with confidence 0.5 (entering e_4 before 9 a.m.), and $H_2 = \langle([10, 30), 0.15), ([30, 50), 0.4), ([50, 70), 0.35), ([70, 90), 0.1)\rangle$ with confidence

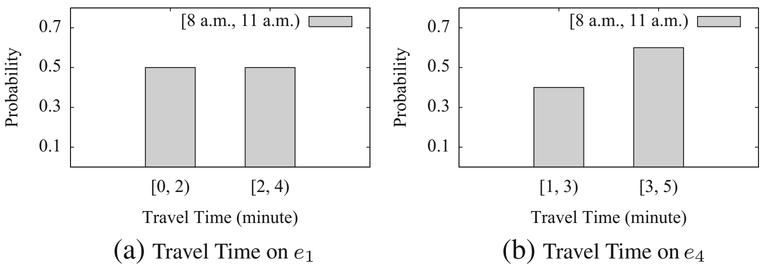


Fig. 8 Travel Time Distribution

0.5 (entering e_4 after 9 a.m.) using histogram aggregation. Finally, we merge H_1 and H_2 into the single histogram $\langle([10, 30), 0.1125), ([30, 50), 0.3675), ([50, 70), 0.3875), ([70, 90), 0.1375)\rangle$. Algorithms 4 and 6 show in detail how we get the final result GHG emissions histogram.

To explain how time-dependent histogram aggregation is done, we introduce the notions of **arrival time period** and **arrival confidence**. Consider a route $R = \langle e_1, e_2\rangle$. The travel time and GHG emissions costs of e_1 and e_2 are represented as time-dependent histograms. Next, if a trip from e_1 to e_2 starts at time t, an **arrival time period** P indicates that entering e_2 at any time point $t_p \in P$ leads to a single travel time histogram associated with a single GHG emissions histogram for e_2. An **arrival confidence** associated with P that indicates the confidence of entering e_2 during P. To illustrate, consider the running example in Fig. 6 and a traversal that starts from e_1 at 8:58 a.m. Table 6 illustrates the arrival time periods for entering e_4 and the corresponding arrival confidence.

Algorithm 5 computes the GHG emissions histogram(s) for a route R at a given starting time ts. The algorithm starts with the first sub-route in R and calls Algorithm 6 iteratively to compute the GHG emissions histogram of R, from the first sub-route to the last sub-route in R. The intermediate result set G_R contains **travel cost estimation tuples** of the form (H_G, H_T, c). A tuple indicates that traversing the first i sub-routes $\langle sr_1, \ldots, sr_i\rangle$ in R starting at ts, the GHG emissions and travel time cost distributions are H_G and H_T, with the confidence of getting H_G and H_T is c. As shown in Figs. 8–9, when traversing a route at a given starting time, the potential travel time may overlap with multiple time intervals with different GHG emissions and travel costs histograms. Therefore, multiple histograms are needed to reprensent all the possible GHG emissions and travel cost distributions with

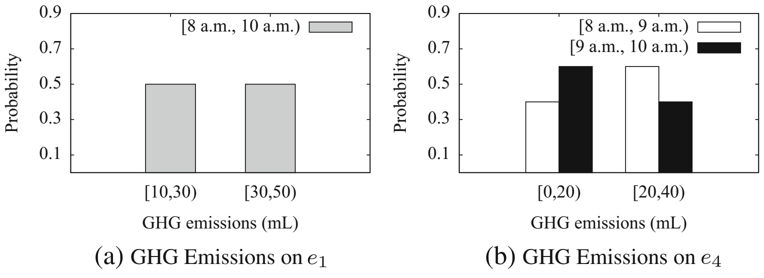


Fig. 9 GHG Emissions Distribution

Table 6 Time-Dependent Histogram Aggregation Example

Arrival time period	Arrival confidence	GHG emissions of e_4	Travel time of e_4
$[8:58a.m., 9:00a.m.)$	0.5	$\langle([0, 20), 0.4),$ $([20, 40), 0.6)\rangle$	$\langle([1, 3), 0.4),$ $([3, 5), 0.6)\rangle$
$[9:00a.m., 9:02a.m.)$	0.5	$\langle([0, 20), 0.6),$ $([20, 40), 0.4)\rangle$	$\langle([1, 3), 0.4),$ $([3, 5), 0.6)\rangle$

corresponding confidences. Set G_R is initialized to contain a single travel cost estimation tuple of the GHG emissions and travel time cost histogram of the first sub-route sr_1 of R (Line 1). Taking the example in Figs. 8–9, if starting at 8:58 a.m., G_R is initialized to $(\langle([10, 30), 0.5), ([30, 50), 0.5)\rangle, \langle([0, 2), 0.5), ([2, 4), 0.6)\rangle, 1)$. For each, Algorithm 5 (Lines 2–6) considers a sub-route sr_i, aggregating its GHG emissions histograms and travel costs histograms with all the tuples in the current G_R (Lines 4–5). Using the confidence of the GHG emissions histograms in G_R as their weights and applying Eq. 2, we merge them into a single histogram H_R as the final result (Line 7).

Algorithm 5 HistogramAggregateForRoute

```
  Input:
    A route $R$ of consecutive sub-routes $\langle sr_1, \ldots, sr_n\rangle$;
    Starting time of the trajectory $ts$;
  Output:
    GHG emissions histograms for $R$;
  1. $G_R \leftarrow \{(H_{G.sr_1}, H_{T.sr_1}, 1)\}$;
  2. for $i \leftarrow 2$ to $n$ do
  3.     $G_{tmp} \leftarrow \emptyset$;
  4.     for all $g \in G_R$ do
  5.         $G_{tmp} \leftarrow G_{tmp} \cup$ TimeDependentHistogramAggregate($sr_i, g.H_G, g.H_T, g.c$);
  6.     $G_R \leftarrow G_{tmp}$;
  7. $H_R \leftarrow Merge(G_R)$;
  8. Return $H_R$;
```

Algorithm 6 aggregates the GHG emissions and travel time cost histograms of the $n+1$-st sub-route sr with those of the first n sub-routes in a route R assuming the traversal of R starts at time ts. The algorithm first determines the time intervals in which the GHG emissions time-dependent histograms of sr that overlap with the range of H_T(line 1). For the i-th time interval with its corresponding histograms, we aggregate the the GHG emissions histogram $H_{G.i}$ with H_G and travel time histogram $H_{T.i}$ with H_T using the aggregation method shown in Algorithm 3. The aggregated results represent the corresponding GHG emissions and travel time cost histograms if traversing sr in the i-th travel time interval, and the confidence of getting such histograms is the product of c_i and c, where c_i is the confidence of traversing sr in the i-th time interval (Lines 3–7). Table 7 shows the intermediate results of the aggregation. In the example in Figs. 8–9, consider route $R = \langle e_1, e_4\rangle$. If the traserval starts from e_1 at 8:58 a.m, a candidate set G_{sr} of travel cost estimation tuples is returned. Thus, the final result is $\langle([10, 30), 0.1125), ([30, 50), 0.3675), ([50, 70), 0.3875), ([70, 90), 0.1375)\rangle$.

Table 7 Time-Dependent Histogram Aggregation Result

Arrival Time Period	Arrival Confidence	GHG emissions of e_4	Aggregated GHG emissions
$[8:58a.m., 9:00a.m.)$	0.5	$\langle([0,20),0.4),$ $([20,40),0.6)\rangle$	$\langle([10,30),0.1),$ $([30,50),0.35),$ $([50,70),0.4),$ $([70,90),0.15)\rangle$
$[9:00a.m., 9:02a.m.)$	0.5	$\langle([0,20),0.6),$ $([20,40),0.4)\rangle$	$\langle([10,30),0.125),$ $([30,50),0.375),$ $([50,70),0.375),$ $([70,90),0.125)\rangle$

Algorithm 6 TimeDependentHistogramAggregate

```
Input:
    Sub-route $sr$;
    GHG emissions histogram $H_G$;
    Travel time histogram $H_T$;
    Confidence $c$;
Output:
    Candidate set $G_{sr}$;
1. $H_T$ overlaps with $n$ intervals in the GHG emission time-dependent histograms of $sr$;
2. $G_{sr} \leftarrow \emptyset$;
3. for $i \leftarrow 1$ to $n$ do
4. Compute the confidence $c_i$ of starting in the $i$-th interval $intvl_i$.
5. Aggregate $H_G$ and the GHG emissions histogram of $sr$ in $intvl_i$, denoted as $H_{G.i}$;
6. Aggregate $H_T$ and the travel time histogram of $sr$ in $intvl_i$, denoted as $H_{T.i}$;
7. $G_{sr} \leftarrow G_{sr} \cup (H_{G.i}, H_{T.i}, c \cdot c_i)$;
8. Return $G_{sr}$;
```

6 Empirical study

We conduct a range of experiments to gain insight into the accuracy, efficiency, and storage properties of the proposed methods.

6.1 Experimental settings

We use a large GPS tracking data set containing more than 200 million GPS records collected at 1 Hz from 150 vehicles in Denmark from January 2007 to December 2008. A total of 802K traversal records are generated from the data set.

We use the road network of Denmark from OpenStreetMap.[1] To get the best map-matching, we extract edges from OpenStreetMap data with the finest granularity, with 414K vertices and 1,628K edges. We apply an existing map-matching tool [19] to match the GPS records to the road network, from which we get a trajectory set *TR* of 62K trajectories. We

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

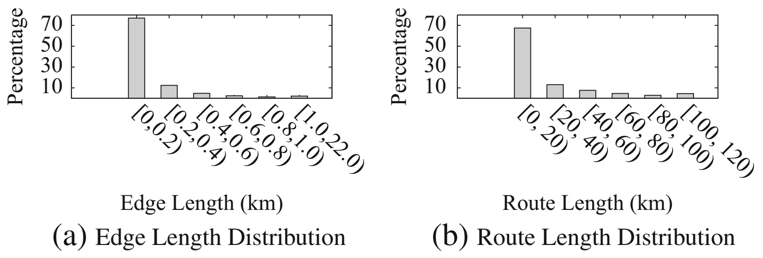


Fig. 10 Distributions of Edge and Route Length

merge adjacent edges into longer ones when they can be viewed as conceptually a single edge.

Figure 10 shows the distribution of the lengths of the edges in the road network and the routes derived from our GPS data.

VT-micro [1], a robust model that can compute vehicular environmental impact from GPS data [6], is applied to compute GHG emissions of of the road network edges from the trajectories. The time interval of interest *TI* is set to a day: [0 a.m., 12 p.m.).

For each edge, we build time dependent eco-weight histograms. Our analysis indicates that 75.2 % of the major roads in urban region, i.e., Aalborg in North Jutland, have good coverage of traversal records, and the remaining 24.8 % of the roads are not covered by sufficient traversal records. For an edge with sufficient traversal records, we use the paper's proposal to enable time-dependent eco-weight histograms. For an edge without sufficient traversal records, a GHG emissions value *EH* is derived based on the length and the speed limit of the edge, which can be obtained from OpenStreetMap. Thus, the edge is associated with a single histogram with only one bucket, indicating an *EH* with probability 1.

For each edge, we consider a baseline approach that uses isomorphic histograms with variable bucket width for different time intervals to represent the uncertain time-dependent weight of each edge. Thus, every edge is assigned an uncertain time-dependent weight in this baseline approach and no compression techniques are employed.

The values used for the finest temporal granularity α, the merge threshold T_{merge}, the reduction threshold T_{red}, the dependency threshold T_{dep}, and the aggregation threshold T_{agg} for point-wise aggregation are shown in Table 8, where default values are shown in bold.

The algorithms for histogram merging and bucket reduction are implemented in Python, which is well suited for scientific and numeric computations. A machine with a 64-core AMD Opteron(tm) 2.24GHZ CPU, 528GB main memory, and 413GB external memory is used.

Table 8 Parameter Settings

Parameters	Values	Note
N_{bucket}	20, 40	bucket number of initial histograms
T_{merge}	0.9, 0.91, 0.92, 0.93, 0.94,	
	0.95, 0.96, 0.97, 0.98, 0.99	histogram merge threshold
T_{red}	35, 40, 45, 50, 55, 60	bucket reduction threshold
T_{dep}	0.15 0.2, 0.25	edge dependence threshold
T_{agg}	1, 2, 3, 4, 5	point-wise aggregation threshold
α	15 mins, 30 mins, 1 hour	finest time granularity

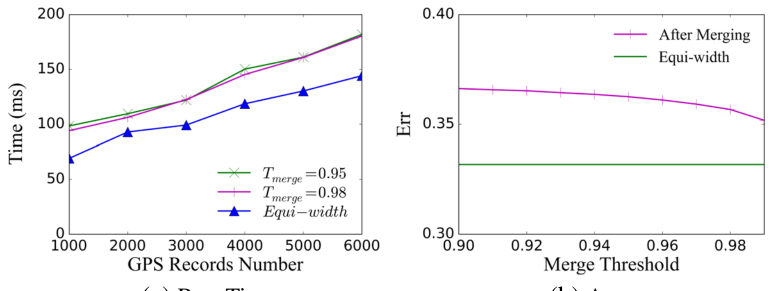


Fig. 11 Histogram Merging Study

6.2 Running time efficiency

Figure 11a shows the time needed to build equi-width histograms and to merge the histograms for an edge. The time increases as the number of GPS records associated with an edge increases. Thus, if an edge is covered by more GPS records, it takes longer to build the initial histograms. We show the running time when setting T_{merge} to 0.95 and 0.98 in Fig. 11a, and our experiments suggest that smaller T_{merge} leads to less histogram merge time. Figure 12a depicts the effect of varying the number of buckets N_{bucket} on the running time of histogram merging. The figure suggests that a smaller N_{bucket} renders histogram merging more efficient. Figure 13a shows the running time of bucket reduction using 6,000 traversal records, which is the largest number of records that an edge has in our dataset.

We further study the efficiency of aggregating histograms. We use a set of 3,849 routes that each is traversed by at least 1,000 trajectories. On average, a route covers 4.5 edges. If the **NMI** of two adjacent edges exceeds T_{dep}, we treat them as a virtual edge. Figure 14a shows the performance of our bucket-wise histogram aggregation method (*BA*) for routes in comparison to the baseline method, as well as results when virtual edge (*BA* + *VE*) and extended virtual edges (*BA* + *EVE*) are considered, where T_{dep} value is set to 0.2. Similarly, Figs. 15a and 16a show the performance of the point-wise and median aggregation methods in comparison to the baseline. Recall that the dependence threshold T_{dep} is configurable. To illustrate the impact of T_{dep}, Fig. 17a shows the performance of bucket-wise aggregation when varying dependence threshold T_{dep} and also considering the result of not considering T_{dep}, where we assume the GHG emission distributions are completely independent between any adjacent edges. The results suggest that with a smaller T_{dep}, the histogram aggregation

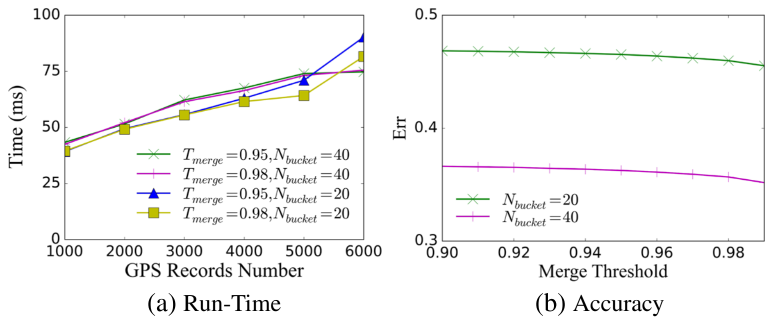
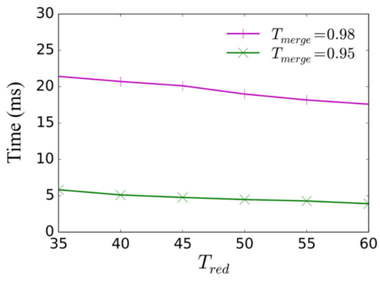
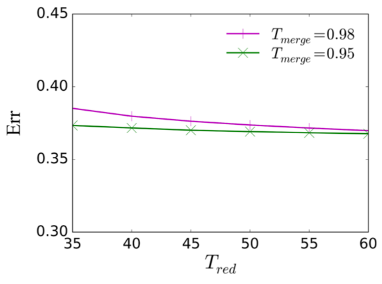


Fig. 12 Varying the Number Buckets of Initial Histograms



(a) Run-Time



(b) Accuracy

Fig. 13 Bucket Reduction Study

run-time is lower, as more virtual edges are generated when building the ERN. The average time to generate GHG emissions histograms for a single edge is 45 milliseconds. It is acceptable as it is considered as one-off preprocessing. Moreover, the generation of a virtual edge is done in 600 milliseconds.

6.3 Estimation accuracy

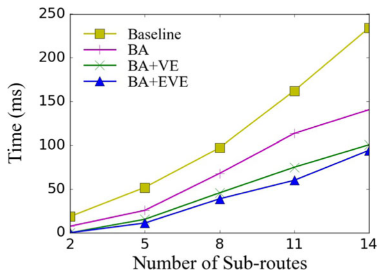
To study the approximation accuracy of our histograms, we measure the distance between the original data distribution and the derived histogram representations, including initial equi-width histograms, histograms after merging, and histograms after bucket reduction.

Let $dd = \{(v_1, p_1), \ldots, (v_n, p_n)\}$ be the original data distribution in a period, where v_i and p_i indicate a value and its probability. The accuracy of a histogram in the period is defined by Eq. 9.

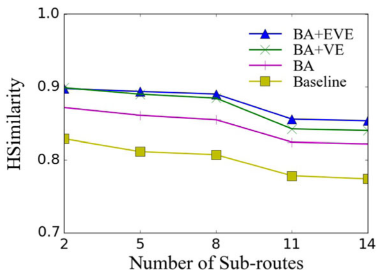
$$Err(H, dd) = \frac{1}{n} \sum_{i=1}^{n} \frac{\left| p_i - \frac{H.p_k}{|H.b_k|} \right|}{max(p_i, \epsilon)}, \tag{9}$$

where the k-th bucket in H contains v_i, i.e., $v_i \in H.b_k$, and $H.p_k$ is the total probability of the k-th bucket, and $|H.b_k|$ is the width of the k-th bucket in H. We use a constant $\epsilon = 0.1$ to avoid fluctuations caused by small probabilities, where ϵ is the smallest probability we consider in the our ERN. This accuracy metric captures the relative accumulative deviation from the original distribution.

Figure 11b shows average *Err* values of the initial equi-width histograms and the histograms after merging for varying merge thresholds. The initial equi-width histograms



(a) Aggregation Run-Time



(b) Aggregation Accuracy

Fig. 14 Bucket-Wise Histogram Aggregation Study

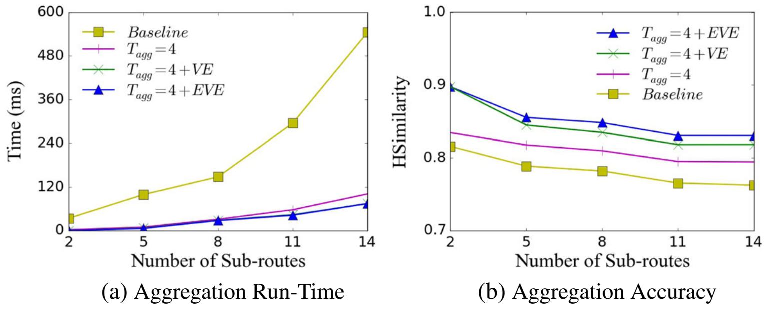


Fig. 15 Point-wise Histogram Aggregation Study

achieve the least accuracy loss, and the merged histograms achieve improved accuracy as the merge threshold increases. We also evaluate the accuracy of bucket reduction. We choose two sets of merged histograms with merge thresholds 0.95 and 0.98, respectively. Moreover, Fig. 12b shows the impact of varying the number of buckets N_{bucket} on the accuracy of histogram merging. The figure suggests that we can achieve a lower histogram merge loss by assigning more buckets to the initial histograms at the cost of run time. Figure 13b shows that the accuracy increases as the reduction threshold increases.

Next, we consider the 3,849 trajectories from Section 6.3 to evaluate the accuracy of histogram aggregation. We split the trajectories into two sets: training trajectories (from the first 18 months, 43K trajectories) and testing trajectories (from the last 6 months, 19K trajectories). For each route, we aggregate the histograms from the ERN to estimate its GHG emissions histogram. We then generate ground-truth GHG emissions histograms for the route in each time interval without any data compression using the testing trajectories. The accuracy of histogram aggregation is defined as the histogram similarity (*HSimilarity*) between the estimated histogram and the ground-truth histogram, using Eq. 1. Figure 14b shows the accuracy of bucket-wise histogram aggregation (*BA*) with varying the number of edges in a route, as well as the accuracy when virtual edges ($BA + VE$) and extended virtual edges ($BA + EVE$) are taken into account, and it also includes the baseline method. Similarly, Figs. 15b and 16b show the accuracy of point-wise and median aggregation and the baseline method.

Again, Fig. 17b uses bucket-wise histogram aggregation as an example when virtual edges and extended virtual edges are considered as an example, and it shows the impact of the dependence threshold T_{dep} on aggregation accuracy and it also covers the case where T_{dep} was not considered. The figure indicates that a smaller T_{dep} yields more virtual edges

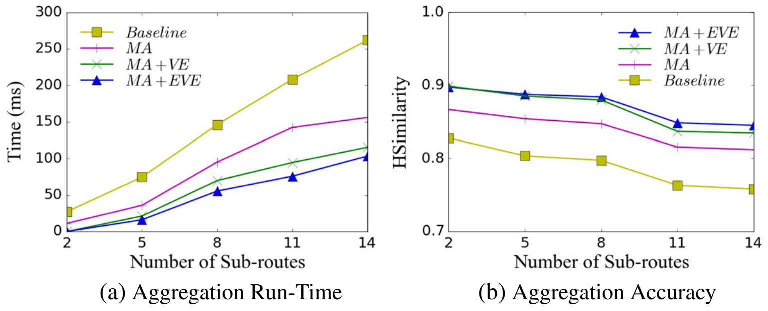


Fig. 16 Median Histogram Aggregation Study

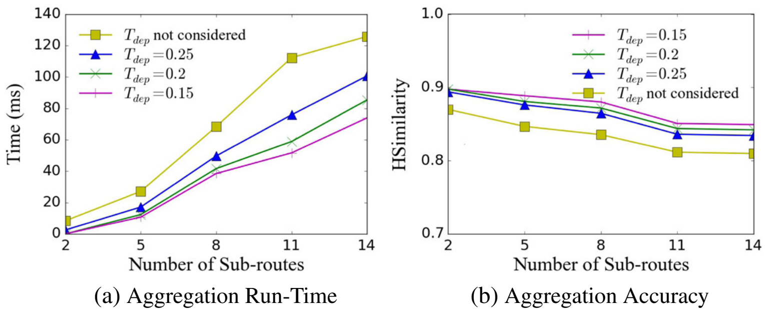


Fig. 17 T_{dep} Study (bucket-wise histogram aggregation method)

and extended virtual edges, which better capture the traffic patterns, and thus better GHG emissions estimation accuracy is achieved. Figure 17a and b show that we can get estimation accuracy improvements by modeling adjacent edges with high dependency as virtual edges and extended virtual edges.

The effect of α is shown in Fig. 18. As the the merge threshold and reduction threshold increase, the more accurate the resulting histograms are. Figure 18a shows that the accuracy improves with finner time granularities, because histograms can capture distinct distributions during short intervals. Figure 18b shows that the accuracy decreases with finner time granularities, because the finer time granularities, the more histograms an edge has and thus the less buckets of a histogram has. This adversely affect the accuracy of the resulting histograms.

6.4 Storage consumption

We next evaluate the storage savings that can be obtained by using histogram merging and bucket reduction. A bucket in a histogram requires two integer values (i.e., 8 bytes) to indicate the lower and upper bound of the bucket and a double (i.e., 8 bytes) for the bucket probability.

We introduce the *memory compression ratio MCR* to measure the storage reduction. In particular, the *MCR* for histogram merging is computed as $MCR_m = \frac{M_{init}-M_{merge}}{M_{init}}$,where M_{init} and M_{merge} represent the storage required to represent the initial time dependent histograms and the merged histograms. The *MCR* for bucket reduction is computed as $MCR_r = \frac{M_{merge}-M_{redu}}{M_{merge}}$,where M_{redu} represents the storage required to represent the histograms after bucket reduction based on merged histograms.

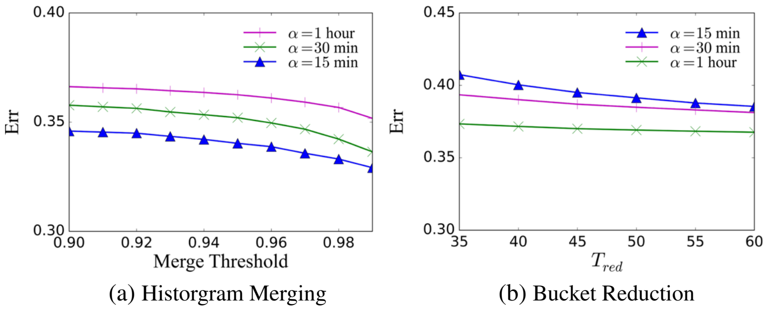
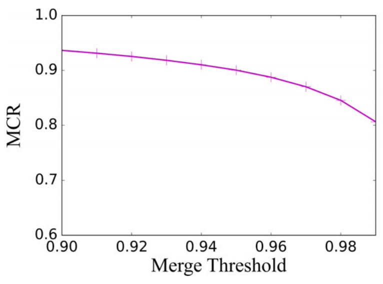
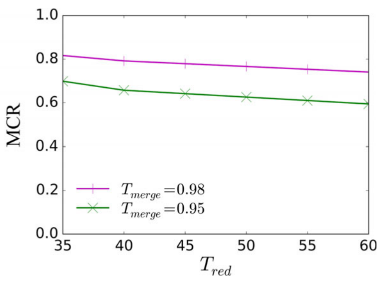


Fig. 18 Impact of α



(a) Histogram Merging



(b) Bucket Reduction

Fig. 19 *MCR* Study

Figure 19a shows that when the merge threshold is set to 0.9, the storage required by the initial histograms can be reduced by 93 %. When the merge threshold is 0.98, the reduction is 84 %. Recall that the accuracy when using this merge threshold is quite close to that of the initial histograms (see Fig. 11b). We fix two sets of merged histograms (with merge thresholds 0.95 and 0.98) and observe the *MCR* with varying reduction thresholds. Figure 19b shows that bucket reduction can further reduce the required storage: smaller thresholds achieve better *MCR*.

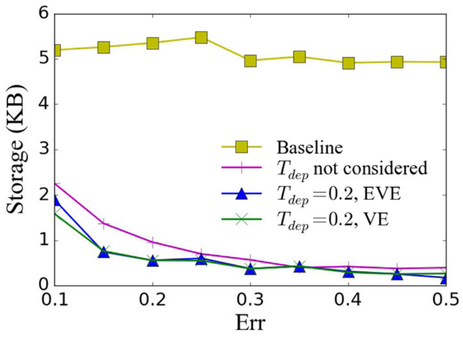
Figure 20 reports the average storage required for a single edge in order to achieve different accuracies for our different methods (T_{dep} not considered, *VE* used and *EVE* used) and the baseline method. The figure shows that to achieve a higher accuracy (i.e., a smaller *Err* value), more storage is required.

Figure 21 shows the average number of histograms generated with different settings (α and T_{merge}). With smaller α, we have much more histograms in the beginning before histogram merging. For instance, we have 96 and 24 histograms when α =15 minutes and α =1 hour, respectively. Thus, after histogram merging, a smaller α results in more histograms. As T_{merge} increases, fewer histograms with similarity higher than T_{merge} can be merged, and thus more histograms remain.

6.5 Eco-routes versus shortest routes

Finally, we conduct experiments to compare eco-routes and shortest routes. Figure 22 shows an example where the eco-route (in blue) and the shortest route (in black) from a location *A* to a location *B* in Aalborg, Denmark. As can be seen, the two routes differ significantly.

Fig. 20 Storage Usage vs. *Err*



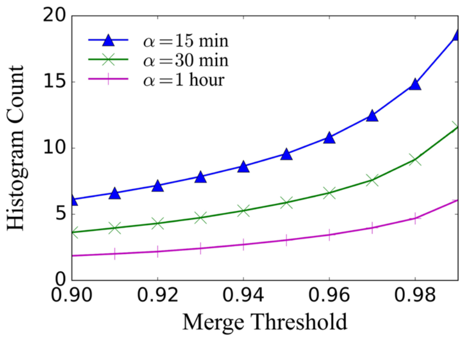


Fig. 21 Avg. Numbers

Next, we compute the shortest routes and eco-routes for 1,000 randomly chosen source-destination pairs. We group the source-destination pairs according to the Euclidean distances between the sources and the destinations. Figure 23 shows the average percentage of the edges that are shared between eco-routes and shortest routes for different groups

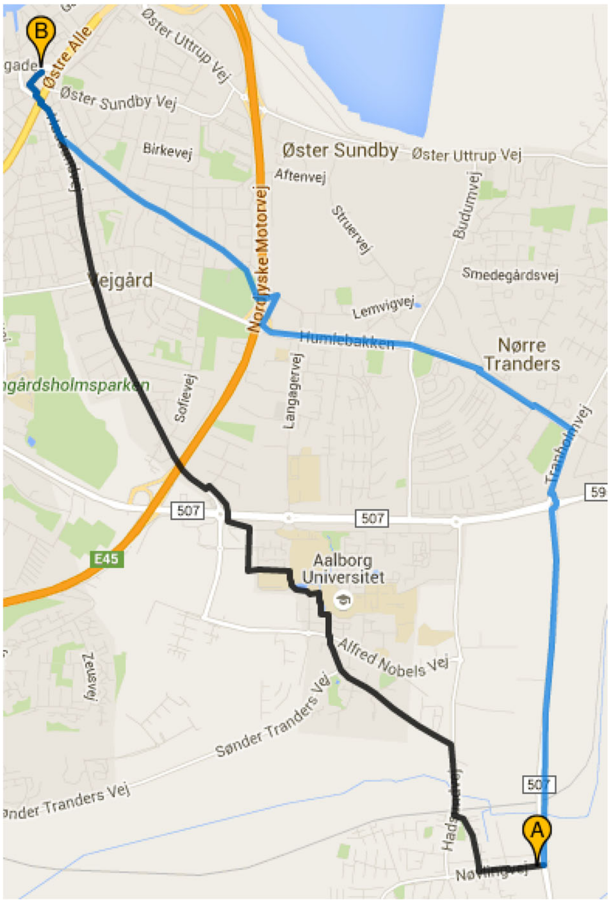


Fig. 22 Eco Route vs. Shortest Route

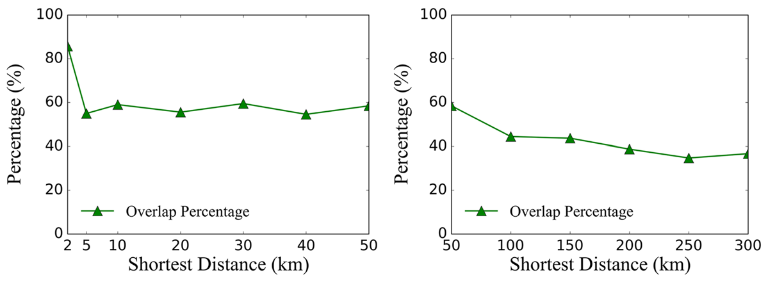


Fig. 23 Eco-Routes and Shortest Routes Overlapping Percentage

of source-destination pairs. The figure indicates that eco-routes can be significantly different from shortest routes; and the further the source and the destination are apart, the more different the eco-routes are from the shortest routes. This is partly due to the fact that highways, which are used often longer routes, may not necessarily be the shortest paths. However, when traveling for long distances, driving on highways may produce less GHG emissions compared to driving on roads in urban areas. This is because frequent brakes and accelerations, due to traffic lights in urban areas, produce more emissions.

6.6 Summary

Based on our experimental results, the recommended parameter settings for T_{merge}, T_{red}, and T_{dep} are 0.95, 50, and 0.2, respectively. With these settings, each edge requires on average 3.98 histograms and about 0.61 KB storage space. Assuming there is sufficient GPS data for all edges in the Denmark road network, to achieve the accuracy of the initial histograms without any compression, the storage usage of the compact histograms is $2.82GB$. By using the default settings, the storage usage is $0.50GB$ for the ERN for the Denmark road network. This suggests that our compact representation of time-dependent histograms reduces the storage substantially while maintaining good accuracy.

To sum up, better estimation accuracy can be achieved by using higher settings for T_{merge} and T_{red} and a lower setting for T_{dep}. This comes at the cost of more preprocessing time and higher storage consumption, and our recommended settings are chosen by taking the tradeoffs among all the different thresholds into consideration.

7 Conclusions

We present techniques capable of using a large trajectory data set for assigning eco-weights to the edges in a road network. The resulting Eco-Road Network, which assigns *compact* and *time-dependent*, *uncertain eco-weights* to edges, provides the key foundation for enabling eco-routing. We also study the correlations between the GHG emissions from adjacent edges, and we propose time-dependent GHG emissions distributions estimation techniques for routes. The accuracy and compactness of the proposed techniques are evaluated based on a 2-year GPS vehicle tracking data set. The experimental study shows that our method is able to save up to 20 % storage space in comparison with the baseline method while providing the same accuracy.

In future work, it is of interest to explore advanced routing algorithms that can fully utilize the time-dependent, uncertain eco-weights, e.g., to compute probabilistic eco-routes.

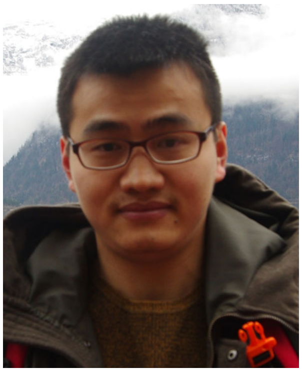
Additionally, using an inverted approach that assigns a time dependent histogram based eco-weight to a group of edges that have similar travel cost distributions may result in further storage space reductions.

Acknowledgments This work was supported by the Reduction project that is funded by the European Commission as FP7-ICT-2011-7 STREP project number 288254, by the DiCyPS project, and by a grant from the Obel Family Foundation.

References

1. Ahn K, Rakha H, Trani A, Van Aerde M (2002) Estimating Vehicle Fuel Consumption and Emissions based on Instantaneous Speed and Acceleration Levels. J Transp Eng 128(2):182–190
2. Andersen O, Jensen CS, Torp K, Yang B (2013) Ecotour: Reducing the environmental footprint of vehicles using eco-routes. In: Proc MDM, 338–340
3. Chen A, Ji Z (2005) Path finding under uncertainty. J Adv Transp 39(1):19–37
4. Ding B, Yu JX, Qin L (2008) Finding time-dependent shortest paths over large graphs. In: Proc EDBT, 205–216
5. Disser Y, Müller-Hannemann M., Schnee M (2008) Multi-criteria shortest paths in time-dependent train networks. In: Proc WEA, 347–361
6. Guo C, Ma Y, Yang B, Jensen CS, Kaul M (2012) Ecomark: evaluating models of vehicular environmental impact. In: Proc ACM SIGSPATIAL, 269–278
7. Guo C, Yang B, Andersen O, Jensen CS, Torp K (2015) Ecomark 2.0: empowering eco-routing with vehicular environmental models and actual vehicle fuel consumption data. GeoInformatica 19(3):567–599
8. Guo C, Yang B, Andersen O, Jensen CS, Torp K (2015) Ecosky: Reducing vehicular environmental impact through eco-routing. In: Proc ICDE, 1412–1415
9. Hua M, Pei J (2010) Probabilistic path queries in road networks: traffic uncertainty aware path selection. In: Proc EDBT, 347–358
10. Cormode G, Garofalakis MN (2009) Histograms and Wavelets on Probabilistic Data. In: Proc ICDE, 293–304
11. Ioannidis Y (2003) The history of histograms (abridged). In: Proc VLDB, 19–30
12. Kanoulas E, Du Y, Xia T, Zhang D (2012) Finding fastest paths on a road network with speed patterns. In: Proc ICDE, 10–21
13. Demiryurek U, Pan B, Banaei-Kashani F, Shahabi C (2009) Towards modeling the traffic data on road networks. In: Proc IWCTS, 13–18
14. Ali RY, Gunturi V, Shekhar S (2015) Spatial big data for eco-routing services: computational challenges and accomplishment. In: SIGSPATIAL Special, 6(2): 19–25
15. Lim S, Sommer C, Nikolova E, Rus D (2012) Practical route planning under delay uncertainty: Stochastic shortest path queries. In: Proc Robotics: Science and Systems, 249–264
16. Ma Y, Yang B, Jensen CS (2014) Enabling time-dependent uncertain eco-weights for road networks. In: Proc. GeoRich@SIGMOD, 1–6
17. Miller-Hooks E, Mahmassani HS (1998) Optimal routing of hazardous materials in stochastic, time-varying transportation networks. Transportation Research Record: Journal of the Transportation Research Board 1645(1):143–151
18. Nikolova E, Brand M, Karger DR (2006) Optimal route planning under uncertainty. In: Proc ICAPS, 131–141
19. Pereira FC, Costa H, Pereira NM (2009) An off-line map-matching algorithm for incomplete map databases. Eur Transp Res Rev 1(3):107–124
20. Wijeratne AB, Turnquist MA, Mirchandani PB (1993) Multiobjective routing of hazardous materials in stochastic networks. Eur J Oper Res 65(1):33–43
21. Yang B, Guo C, Jensen CS (2013) Travel cost inference from sparse, spatio-temporally correlated time series using markov models. PVLDB 6(9):769–780
22. Dai J, Yang B, Guo C, Jensen CS, Hu J (2016) Path Cost Distribution Estimation Using Trajectory Data. PVLDB 10(3)
23. Dai J, Yang B, Guo C, Ding Z (2015) Personalized route recommendation using big trajectory data. In: Proc ICDE, 543–554

24. Kaul M, Yang B, Jensen CS (2013) Building Accurate 3D Spatial Networks to Enable Next Generation Intelligent Transportation Systems. In: Proc MDM, 137–146
25. Yang B, Guo C, Jensen CS, Kaul M, Shang S (2014) Stochastic skyline route planning under time-varying uncertainty. In: Proc ICDE, 136–147
26. Yang B, Guo C, Ma Y, Jensen CS (2015) Toward personalized, context-aware routing. VLDB J 24(2):297–318
27. Yang B, Kaul M, Jensen CS (2014) Using incomplete information for complete weight annotation of road networks. IEEE Trans Knowl Data Eng 26(5):1267–1279
28. Yuan J, Zheng Y, Zhang C, Xie W, Xie X, Sun G, Huang Y (2010) T-drive: driving directions based on taxi trajectories. In: Proc SIGSPATIAL, 99–108
29. Ahn K, Rakha H, Trani A, Van Aerde M (2002) Estimating vehicle fuel consumption and emissions based on instantaneous speed and acceleration levels. J Transp Eng 128(2):182–190



Jilin Hu is a PhD candidate at Aalborg University, Denmark. He was a Research Assistant at Aalborg University in 2015 for half year. He received his M.E. degree from Shanghai Jiao Tong University, China, in 2014 and B.E. degree from China Jiliang University, China, in 2011. His research interests include spatial-temporal data management, uncertain data, data analytics. He was an invited reviewer for journal of ACM TSAS.

Bin Yang is an Associate Professor at Aalborg University, Denmark. He was at Aarhus University, Denmark, during 2011–2014 and at Max-Planck-Institut für Informatik, Germany, during 2010–2011. He received his B.E. and M.E. degrees from Northwestern Polytechnical University, China, in 2004 and in 2007, respectively, and the Ph.D. degree in computer science from Fudan University, China in 2010. His research interests include data management and data analytics. He has served on program committees of several database conferences and as an invited reviewer for several database journals, including ICDE, TKDE, and The VLDB Journal.

Christian S. Jensen is Obel Professor of Computer Science at Aalborg University, Denmark. He was recently at Aarhus University for three years and at Google Inc. for one year. His research concerns data management and data-intensive systems, and its focus is on temporal and spatio-temporal data management. Christian is an ACM and an IEEE Fellow, and he is a member of the Academia Europaea, the Royal Danish Academy of Sciences and Letters, and the Danish Academy of Technical Sciences. He has received several national and international awards for his research. He is editor-in-chief of ACM Transactions on Database Systems.

Yu Ma is a Software Engineer at Uber, Inc. He received the B.E. degree from Beijing University of Posts and Telecommunications, China in 2007, the M.Sc degree from Peking University, China in 2010, and the Ph.D. degree from Aarhus University, Denmark in 2015. His research interests include database management systems and large scale spatial-temporal data analytics.