Highway Systems: How Good are They, Really?

Theodoros Chondrogiannis
Department of Computer and Information Science
University of Konstanz, Germany
theodoros.chondrogiannis@uni.kn

Michael Grossniklaus
Department of Computer and Information Science
University of Konstanz, Germany
michael.grossniklaus@uni.kn

ABSTRACT

Highways play a crucial role in transportation services as they facilitate long-distance traveling and allow driving at an almost constant speed, thus resulting in lower fuel consumption and emissions. Many existing highway systems were designed before practical computational tools had been developed. Furthermore, most existing approaches to evaluating highways focus on analyzing mobility data rather than studying the design of the highway system. To address this gap in existing research, in this paper, we study the problem of evaluating the efficacy of the design of real-world highway systems. To this end, we propose two novel measures for the efficacy of highway systems, along with algorithms to compute them. In addition, we present a first-cut heuristic algorithm that aims at computing a highway system that optimizes our proposed measures. In our experiments, we demonstrate the potential of our methods in measuring the efficacy of real-world highway systems. We also evaluate the performance of our heuristic algorithm in computing a rough design of an efficient highway system.

CCS CONCEPTS

• **Information systems** → **Geographic information systems**; • **Theory of computation** → *Routing and network design problems.*

KEYWORDS

highway systems, transportation network analysis, GIS

ACM Reference Format:
Theodoros Chondrogiannis and Michael Grossniklaus. 2023. Highway Systems: How Good are They, Really?. In *Symposium on Spatial and Temporal Data (SSTD '23), August 23–25, 2023, Calgary, AB, Canada.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3609956.3609963

1 INTRODUCTION

In transportation networks, highways are multi-lane roads with higher speed limits and capacity than most other roads. While highways take up only a small part of the entire road network and do not lead to most destinations directly, they usually make up for the detour by allowing faster driving, thus resulting in shorter travel times. In addition, highways allow driving at almost constant speeds that result in lower fuel consumption and carbon dioxide emissions. However, over time, highways may fail to serve this purpose due to changes in traffic volume and mobility patterns. Rapid changes in urban and rural areas, along with population shifts, dictate the constant improvement of road networks [20]. In this spirit, measuring the efficacy of highway systems is crucial for improving existing and designing new ones.

Many previous works that study the design of highway systems have focused on the *highway alignment* problem [33]. More specifically, they aim to come up with a feasible design for a highway or a highway system. To this end, they solve a complex optimization problem considering multiple criteria, e.g., construction cost, landscape, etc. Regarding the evaluation of highway systems, existing works focus on evaluating traffic [22], robustness [31], environmental impact [12], and others. However, while there have been some attempts [23, 35], the utilization of network analysis to aid the design of highway systems has yet to be studied in depth. Furthermore, network analysis measures, such as centrality [9], have yet to be explored for evaluating highway systems.

In this paper, we study the evaluation of highway systems design from the perspective of algorithmic network analysis. In particular, we first introduce two novel measures, i.e., the *network highwayness* and the *highway utilization gain*. The network highwayness is based on a novel variant of the edge betweenness we introduce and quantifies how much a given highway system acts as a backbone for the underlying road network. The highway utilization gain measures how much time, on average, drivers save by choosing paths that use the highway system. In addition, we introduce the problem of computing an optimal highway system. Our purpose for computing such an optimal system is twofold. First, we aim to compute a rough design of a highway system that can be used as a basis for building highways on a road network. Second, we aim to provide a tool to test how efficient a highway system can be on a given road network. Such a hypothetical system can be used as a reference to evaluate the efficacy of existing highway systems.

In short, our paper makes the following contributions.

- We identify desirable properties of real-world highway systems that one must consider when improving existing or designing new ones (Section 3).
- We introduce two novel measures, i.e., the *network highwayness* and the *highway utilization gain*, that enable us to evaluate different aspects of highway systems (Section 4).
- We define the *optimal highway system* along with a heuristic algorithm that aims at computing a rough design of an efficient highway system (Section 5).

To demonstrate the value of our contributions, we conduct a comprehensive experimental evaluation and report its results and findings in Section 6. In Section 2, we discuss the related work. Finally, we offer some concluding remarks and promising directions for future work in Section 7.

SSTD '23, August 23–25, 2023, Calgary, AB, Canada

ACM ISBN 979-8-4007-0899-2/23/08.
https://doi.org/10.1145/3609956.3609963

2 RELATED WORK

The work of Trietsch [35] is one of the first to deal with designing a highway system as a network analysis problem. In this work, Trietsch introduces the *network design problem with extra nodes* that aims at creating an extension of a given road network that minimizes flow costs, thus functioning as a highway system. Similar to our work, the author proposes the construction of a Euclidean Steiner Tree [14] to connect predetermined highway entry points. In contrast to our work, Triesch does explore the selection of the entry point locations. Instead, the author focuses on aligning the computed Steiner Tree to obtain a realistic design.

Most existing works regarding the design of highway systems have studied the *highway alignment problem*, i.e., the realization of a feasible design of a highway system with a construction cost that is as low as possible. Song et al. [33] provide a comprehensive survey of these works. According to Song et al., Howard et al. [15] have been the first to study the highway alignment problem. Jong et al. [16–18] propose a heuristic-based framework that considers a terrain model as input along with a construction model and employs a genetic algorithm to compute highway alignments. Kang et al. [19] present an intelligent optimization tool that computes multiple potential highway alignments using a genetic algorithm utilizing data from multiple sources to provide accurate feasibility analysis and cost estimation. More recent works focus on solving the highway alignment problem by defining elaborate cost and objective functions. Sameer et al. [27] propose a framework similar to the one by Kang et al. that considers a multi-criteria objective function to choose the best highway alignment among alternatives. In contrast to our approach, these works do not employ network analysis to evaluate the effect of proposed highway systems on the road network. Instead, they focus much more on coming up with a feasible design that satisfies a set of socio-economic objectives.

The concept of highways has also been utilized for shortest path query processing on road networks. Many preprocessing-based methods for shortest path queries define highway-like structures utilized to process queries efficiently [32]. Highway Hierarchies (HH) [28] and Dynamic Highway-Node Routing [30] precompute hierarchical structure by contracting overlay networks induced by edges that lie on many shortest paths. The *transit network* [8] and the *path skeleton* [34] are similar structures used to compute shortest paths between regions and a similar function to highways. Pruned Highway Labeling (PHL) [2] defines a *highway decomposition* as a set of shortest paths P, such that for any shortest path in the network, there is at least one sub-path in the highway decomposition. Abraham et al. [1] introduced the *highway dimension* to enable the theoretical analysis of shortest path query processing methods. In short, if a network has a low highway dimension, it is spatially coherent [29], i.e., it contains many paths that function as highways. Despite relying on the existence of highway-like roads, the structures mentioned above focus on query processing and cannot be used to evaluate a highway system.

Last but not least, some methods that analyze trajectories to detect roads used often by drivers. The structures used by some of these methods [7, 21, 26, 37] are similar to highways. However, in contrast to our approach, these methods depend on the availability of trajectory data and can only be applied to road networks.

3 PRELIMINARIES & PROBLEM STATEMENT

A road network $G = (N, E)$ is represented by a directed *graph* with N nodes and $E \subseteq N \times N$ edges. Each edge $e = (n, n') \in E$ is assigned a direction $n \rightarrow n'$, and a weight $w(e)$ that captures the cost of the transition from n to n'. A *path* $p(s \rightarrow t)$ from a source node s to a target node t is a connected and cycle-free sequence of edges $\langle (s, n_1), (n_1, n_2), \ldots, (n_m, t) \rangle$. The length/cost $\ell(p)$ of a path p is equal to the sum of the weights of all of its edges and the size $|p|$ of a path p is equal to the number of its edges. A *shortest path* $p_{sp}(s \rightarrow t)$ between two nodes s and t is a path that has the lowest cost among all paths from s to t. The cost of $p_{sp}(s \rightarrow t)$ is also termed the *(network) distance* between s and t, i.e., $d(s, t) = \ell(p_{sp}(s \rightarrow t))$. We also denote by $P_{sp}(s \rightarrow t)$ the set of all shortest paths from s to t, by $\sigma_{s \rightarrow t}$ the number of shortest paths $|P_{sp}(s \rightarrow t)|$, and by $\sigma_{s \rightarrow t}(e)$ the number of paths in $P_{sp}(s \rightarrow t)$ that contain edge e.

3.1 Real-world Road Networks

Real road networks are usually treated as edge-labeled multicriteria road networks. Each edge $e \in E$ of a multicriteria road network is assigned more than one weight. Naturally, the shortest path for one cost may not be the shortest path for another. In order to distinguish between shortest paths and different costs, we denote the shortest path between two nodes $s, t \in N$ computed for an edge weight w_i by $p^i_{sp}(s \rightarrow t)$, the cost of the shortest path w.r.t. w_i by $\ell_i(p^i_{sp}(s \rightarrow t))$, and the set of all shortest paths from s to t for w_i by $P^i_{sp}(s \rightarrow t)$.

On edge-labeled road networks, each $e \in E$ is assigned a label λ_e that indicates the type of road e represents, e.g., motorway, primary, residential, etc. Given an edge-labeled road network $G = (N, E)$, the *highway (road) network* is the subgraph $G_H = (N_H, E_H)$ of G formed by all edges $e \in E_H$ that λ_e indicates that the edge represents a highway segment, i.e., λ_e = `'highway'`. The *plain (road) network* of G is the subgraph $G_\Pi = (N_\Pi, E_\Pi)$ that contains all the nodes and edges of G excluding all edges $e \in E_H$, and all nodes $n \in N_H$ where $\forall (n, n') \in E : \lambda_{(n,n')}$ = `'highway'` and $\forall (n', n) \in E : \lambda_{(n',n)}$ = `'highway'`.

3.2 Properties of Real-world Highways

For real-world highway systems we identify the following empirically observed properties.

Property 1. *Traveling via highways can reduce the total duration of a trip, even if the distance is longer.*

Highways are typically roads that have the highest speed limit among all types of roads, have multiple lanes, and do not have any traffic lights. As such, it is usually faster to travel using a highway than to follow a shorter route in terms of distance, which might go through roads with low speed limit and/or traffic congestion [24].

Property 2. *The longer the distance between two locations in a road network, the higher the probability that the fastest path connecting these locations goes through a highway.*

This property is based on the fact that road networks have an inherent hierarchical structure, i.e., small streets enable traveling within neighborhoods, major arteries allow traveling between neighborhoods in a city, and highways allow traveling between cities. In fact, some existing approaches take advantage of this

property to generate shortcuts and optimize shortest path query processing [32].

Property 3. *Edges representing highway segments are fewer than edges representing roads of all other types in a road network.*

Since they must abide by certain specifications, highways are more expensive to build than other roads. At the same time, though, highways accommodate larger traffic volumes than the average road. As such, given a limited budget, building a few highways is usually enough to accommodate the needs of cities or states w.r.t. traffic [35]. The data on the number of highway edges compared to the total number of edges on a road network reported in Table 1 support our claim, i.e., highway edges constitute approximately $0.5-1.5\%$ of the total edges of each road network. Furthermore, the length of a highway network is much smaller than the total length of the road network it belongs to.

Property 4. *All nodes in a road network must be connected by a path that crosses a highway and a path that does not.*

Highways are usually *controlled-access* roads, either with tolls or requiring a fixed-term subscription, e.g., vignette. As a result, many countries regulate the design of highways such that there is always a route that does not access a highway between any two locations [25].

3.3 Problem Statement

Based on the properties discussed above, we observe that a highway system functions as a *backbone* for a road network. An ideal highway system is a sub-network that a) contains large segments of as many shortest paths as possible, b) reduces travel times, primarily facilitating long-distance traveling, and c) is as small as possible. However, in practice, these properties only sometimes apply. As a result of historical and political developments and constraints, e.g., population shifts and local community pressure, the design of highways may not be optimized for efficiency. Motivated by the potential discrepancy between how highway systems should function and how they are designed in practice, we attempt to answer the following research questions in this paper:

(1) *How can we measure the efficacy of the design of existing highway systems?*
(2) *How can we facilitate the improvement of existing and the design of new highway systems?*

To this end, we first introduce two measures that quantify the properties of highway systems. Then, we propose a method that aims at optimizing these measures to support the design process of highway systems.

4 HIGHWAY EVALUATION MEASURES

Based on the properties discussed above, an ideal highway system is a sub-network that a) contains large segments of as many fastest paths as possible and b) is as small as possible. One way to determine such a sub-network is to examine all possible subsets of paths in a given network and identify the subset that optimizes for both criteria. This approach is prohibitively expensive. Instead, we aim to assign a score to each edge e of the network, indicating whether e represents a highway segment. We refer to such a score as the *edge highwayness* of e, i.e., $hw(e)$. The higher the score of an edge is, the higher the likelihood that this edge is part of a path that functions as a highway for the network at hand.

4.1 Network Highwaynness

Following from Property 2, the further apart two given locations s and t are, the higher the chance that the shortest path $p_{sp}(s \to t)$ uses a highway. Also, despite the highway system being relatively small in comparison to the full network (cf. Property 3), since highways aim at reducing travel time (cf. Property 1), we expect the shortest paths that use highways to be more than the shortest paths that do not. Consequently, we expect that many shortest paths use at least some part of a highway and that edges representing highway segments lie on many shortest paths. This trait is formally captured by the *Edge Betweenness* (EB) [3], which is given by

$$eb(e) = \sum_{\forall s,t \in N, \forall e' \in E} \frac{\sigma_{s \to t}(e')}{\sigma_{s \to t}}.$$

Intuitively, long paths have a higher impact on the EB score. If an edge lies on a long shortest path, it also lies on multiple shortest paths that are sub-paths of the longer one. To address this issue, Borgatti and Everett [4] introduced a measure that weights all shortest paths inversely proportional to their length, termed *Length-scaled Betweenness* (LSEB), which is given by

$$lseb(e) = \sum_{\forall s,t \in N, \forall e' \in E} \frac{1}{d(s,t)} \cdot \frac{\sigma_{s \to t}(e')}{\sigma_{s \to t}}.$$

The *scaling factor* $1/d(s,t)$ ensures that long shortest paths contribute less to the betweenness of an edge.

The classical approach to compute both EB and LSEB is *Brandes' algorithm* [5, 6]. For each node of the network, the algorithm executes a single-source shortest path exploration (SSSP) to all other nodes similarly to Dijkstra's algorithm [10]. However, in contrast to Dijkstra's algorithm, the exploration does not maintain just the predecessor of each expanded node n, but a stack S that contains all the predecessors in the traversal order that yield the same distance to n. After each traversal, the algorithm updates each node's shortest path count $\sigma(n)$ by going through the elements of S and exploring paths backward. The computed values σ for each node are then accumulated to compute the final result.

4.1.1 Generalized Scaled Edge Betweenness. Despite being intuitive as measures, both EB and LSEB consider only a single edge weight. Towards defining a better measure, we generalize LSEB. Given two edge weights w_i and w_j, we define the *generalized scaled edge betweenness* (GSEB) as

$$gseb^{w_i}_{w_j}(e) = \sum_{\forall s,t \in N} \sum_{\forall p \in P^{w_{i,j}}_{sp}(s \to t) : e' \in p} \frac{1}{\sum_{\forall e'' \in p} w_j(e'')} \cdot \frac{\sigma_{s \to t}(e')}{\sigma_{s \to t}}.$$

For the computation of GSEB, the edge weight used to compute the shortest paths does not have to be the same as the one used in the scaling factor. A different weight or even a linear combination of multiple weights can be used instead. However, if the edge weight used to compute the shortest paths is also used as a scaling factor, i.e., $\forall e \in E : w_i(e) = w_j(e)$, then the generalized scaled edge betweenness is equivalent to the length-scaled edge betweenness.

Algorithm. To compute GSEB, we need to extend Brandes' algorithm [6] to support different edge weights for the computation of the shortest paths and the scaling factor. Consider a network $G = (N, E)$, the edges of which are assigned two weights w_i and w_j. Without loss of generality, we assume that the shortest paths $p_{sp}(s \rightarrow t)$ are computed in order to minimize w_i, i.e., $p_{sp}^{w_i}(s \rightarrow t)$. The first step towards the computation of the GSEB is to build the shortest paths directed acyclic graph (DAG) from every node $s \in N$ with regard to cost w_i. Then, we traverse the DAG for each node s from the leaves to the root. On the way to the root, we accumulate the costs $1/\sum w_j$. In contrast to Brandes' algorithm, the DAG is constructed in a way such that the two costs of the paths are minimized in sequence. Given two nodes $s, t \in N$, among all shortest paths $p_{sp}^{w_i}(s \rightarrow t)$, we consider only those with minimum ℓ_j. Furthermore, the accumulation uses $\ell_j(p_{sp})$ as scaling factor instead of $\ell_i(p_{sp})$.

Algorithm 1 gives the pseudocode of our algorithm to compute GSEB, which is based on Brandes' algorithm [6]. Initially, the betweenness B of each edge of the graph is initialized to 0 (Lines 1–2). Then, for each node s of the input network, the algorithm performs two operations: i) the *SSSP exploration* that constructs the shortest path tree and stores the order in which nodes/paths need to be examined in a stack (Lines 4–27), and ii) the *Accumulation* that involves the examination of nodes in the stack and the computation of the final GSEB scores (Lines 28–34).

In Lines 4–8, the algorithm initializes the following lists that share a 1:1-association with the nodes of the graph: *settled* stores Boolean values that indicate which nodes of the graph have been settled, i.e., their distance from s is known during the SSSP exploration, and is initially set to false for all nodes; $C_A[n]$ stores for each node n its tentative distance from s during the exploration, which corresponds to edge weight w_a used to compute the shortest paths and is initially set to $+\infty$ for all nodes; $C_B[n]$ stores for each node n the minimum length from s w.r.t. edge weight w_b which corresponds to the cost used as the scaling factor and is initially set to $+\infty$ for all nodes; $cnt[n]$ stores for each node n the number of shortest paths from s computed during the exploration and is initially set to 0 for all nodes; $pred[n]$ stores the predecessor of each node in the shortest path DAG and is initially set to $\emptyset$ for all nodes. A priority queue Q that is used to determine the traversal order and a stack S that stores the order in which nodes are settled during the SSSP exploration are initialized in Line 9.

In Lines 10 to 27, a modified version of Dijkstra's algorithm [10] from node s to all other nodes $n \in N \setminus \{s\}$ is executed. For each node, our routine maintains the distance $C_A[n]$, the shortest path count $cnt[n]$, and the cumulative cost $C_B[n]$. More specifically, in Line 10 both costs C_A and C_B of s are set to 0 and s is added to Q. At each iteration (Lines 12–27), the front node u of Q is retrieved in Line 13. Then, in Lines 14–17 node u is checked whether it is settled or not, and if not, it is marked as settled and is added to S. The examination of outgoing edges (u, v) begins in Line 18. If the newly found path to v yields the lowest C_A cost, or has equal C_A cost and the lowest C_B cost among all already found paths to v (Line 19), then in Lines 20–23 the algorithm sets the costs $C_A[v]$ and $C_B[v]$, the size $size[v]$, the list of predecessors $pred[v]$ in the expansion, and the shortest path count $cnt[v]$, and adds v in Q in Line 24. If the newly found path yields the same costs C_A and C_B as a previously found shortest path (Line 25) then u is added to $pred[v]$ in Line 26 and $cnt[v]$ are updated in Line 27.

After the completion of the SSSP exploration, the algorithm proceeds with the accumulation of scores. First, in Lines 28–29, the list of accumulated scores (dependencies) $\delta[n]$ is initialized to 0 for all nodes. Then, the algorithm iterates over the elements of S. The predecessor u of each popped node v is retrieved in Line 31. Subsequently, the accumulated score $\delta[u]$ is updated in Line 33, and the bicriterial betweenness $B[(u, v)]$ is updated in Line 34. Finally, the list B containing the bicriterial edge betweenness for each network edge is returned in Line 35.

Algorithm 1: Generalized Scaled Edge Betweenness

Input: Bicriteria Network $G(N, E)$
Output: List B of generalized scaled edge betweenness

1 **foreach** $e \in E$ **do**
2 $B[e] \leftarrow 0$;
3 **foreach** $s \in N$ **do**
4 **foreach** $u \in N$ **do** ▷ SSSP Exploration
5 $settled[u] \leftarrow$ false;
6 $C_A[u] \leftarrow \infty, C_B[u] \leftarrow \infty$;
7 $pred[u] \leftarrow \emptyset$;
8 $cnt[u] \leftarrow 0$;
9 $S \leftarrow$ empty stack, $Q \leftarrow$ empty priority queue;
10 $C_A[s] \leftarrow 0, C_B[s] \leftarrow 0$;
11 **enqueue** $\langle s, C_A[s] \rangle$ on Q;
12 **while** Q *is not empty* **do**
13 **dequeue** $\langle u, C_A[u] \rangle$ from Q;
14 **if** $settled[u]$ **then**
15 **continue**;
16 $settled[u] \leftarrow$ true;
17 **push** u onto S;
18 **for** $e = (u, v) \in E$ **do**
19 **if** $C_A[u] + w(e) < C_A[v]$ **or** $(C_A[u] + w_A(e) = C_A[v]$ **and** $C_B[u] + w_B(e) < C_B[v])$ **then**
20 $C_A[v] \leftarrow C_A[u] + w_A(e)$;
21 $C_B[v] \leftarrow C_B[u] + w_B(e)$;
22 $pred[v] \leftarrow \{u\}$;
23 $cnt[v] \leftarrow cnt[u]$;
24 **enqueue** $\langle v, C_A[v] \rangle$ on Q;
25 **else if** $C_A[u] + w_A(e) = C_A[v]$ ***and*** $C_B[u] + w_B(e) = C_B[v]$ **then**
26 $pred[v] \leftarrow pred[v] \cup \{u\}$;
27 $cnt[v] \leftarrow cnt[v] + cnt[u]$;
28 **foreach** $u \in N$ **do** ▷ Accumulation
29 $\delta[u] \leftarrow 0$;
30 **while** S *not empty* **do**
31 **pop** v from S;
32 **for** $u \in pred[v]$ **do**
33 $\delta[u] \leftarrow \delta[u] + \frac{cnt[u]}{cnt[v]} \cdot (\frac{1}{C_B[v]} + \delta[v])$;
34 $B[(u, v)] \leftarrow B[(u, v)] + \frac{cnt[u]}{cnt[v]} \cdot (\frac{1}{C_B[v]} + \delta[v])$;
35 return B;

(a) Original highway system (b) Top-$|E_H|$ of G by GSEB on G (c) Top-$|E_H|$ of G by GSEB on G_{ext}

Figure 1: Original network and backbones determined using GSEB on the road network of Berlin.

Correctness. The correctness of Algorithm 1 follows from the proof of correctness of the betweenness centrality [5, Theorem 6] and the length-scaled betweenness [4]. More specifically, Brandes has proven that given the directed acyclic graph (DAG) from a source node s, the δ dependencies can be propagated up along the edges of the DAG. This allows the correct computation of the LSEB as well [6]. Given a source node s, Algorithm 1 constructs a DAG that is a subgraph of the DAG constructed by Brandes' algorithm and maintains the same properties. In Lines 18–27, Algorithm 1 ensures that all shortest paths $P_{sp}(s \to n)$ minimize both costs, i.e., $\forall p, p' \in P_{s \to n}$ we have that $\ell_1(p) = \ell_1(p')$ and $\ell_2(p) = \ell_2(p')$. As such, the scaling factor can be computed using any of the minimized costs, as all shortest paths between any two nodes s and t have all costs equal. Therefore, the propagation of δ in Line 33 guarantees the correct computation of GSEB.

Complexity. The SSSP exploration of Algorithm 1 (Lines 3–27) performs a run of Dijkstra's algorithm for each node of the network. Each run requires $O(|E| + |N| \cdot \log |N|)$ time. The accumulation of the GSEB values (Lines 28–34) is done by a single scan over the nodes in the stack, thus requiring $O(|N|)$. Hence, Algorithm 1 has a time complexity of $O(|N| \cdot |E| + |N|^2 \cdot \log |N|)$.

4.1.2 From GSEB to Network Highwaynness. Let $G = (N, E)$ be a road network and $G_H = (N_H, E_H)$ be the highway system of G. We first compute the GSEB for all edges of G. Then, we define the set of the *backbone edges* of G as the subset $E_b \subseteq E$ of the $k = |E_H|$ edges with the highest score. Since GSEB is defined based on the properties of real-world highways, the backbone edges are expected to have those properties as well. To compare the set of highway edges E_H with the set of backbone edges E_b in terms of GSEB, we define the *highwayness* of G_H as:

$$hw(G_H) = \frac{\sum_{e \in E_H} gseb(e)}{\sum_{e' \in E_b} gseb(e')} \tag{1}$$

Following from Properties 1– 2, we use the travel time as the edge weight to compute shortest paths, while we use the distance to compute the scaling factor. The highwayness indicates how much the highway system G_H functions as a backbone for the road network G. Under this premise, the design of a given highway system G_H can be considered good if the cumulative GSEB of E_H is equal or comparable with that of E_b. Note that instead of comparing how many edges in E_H are also in E_b, we focus solely on GSEB. In practice, an edge $e \in E_H$ does not absolutely have to be a backbone edge. It is sufficient if the GSEB of all E_H edges is high.

4.1.3 Effective Area. The highwayness of a highway system G_H of a road network G indicates how effective is G_H in facilitating traveling within G. However, highways are not necessarily built to facilitate traveling within a given road network but also through that network. A highway that crosses a specific city may not only be helpful for trips within that city but also trips between neighboring cities as well. This fact affects the computation of highwayness as the backbone edges E_b are determined based on the GSEB values computed over G excluding its surroundings.

Consider the examples on the road network $G = (N, E)$ of Berlin shown in Figure 1. Specifically, Figure 1a illustrates the highway system $G_H = (N_H, E_H)$, Figure 1b illustrates the backbone as determined by the Top-$|E_H|$ of G in GSEB order computed over G, and Figure 1c illustrates the backbone as determined by the Top-$|E_H|$ of G in GSEB order computed over G_{ext}. The highway system G_H shares more edges with the backbone computed using the extended network than the backbone computed using the original network. As such, G_H facilitates traveling not only between locations within Berlin but also between locations outside Berlin.

To address this issue, we must consider the *effective area* of a given highway system, i.e., the area within which a given highway system facilitates trips. We first define the extended road network $G_{ext} = (N_{ext}, E_{ext})$ of a road network $G = (N, E)$, for which we have $N \subseteq N_{ext}$ and $E \subseteq E_{ext}$, and we compute the GSEB of all edges in E_{ext}. Then, we define the backbone edges E_b as the top-$|E_H|$ edges of E is terms of GSEB. In other words, while the GSEB is computed over all edges E_{ext} of the extended network G_{ext}, the backbone edges E_b are picked only out of the edges of the road network G, i.e., $\nexists e \in E_b : e \in E_{ext} \wedge e \notin E$. Note that the exact computation of the effective area is out of the scope of our work and is a direction we plan to explore in the future. However, to obtain some insight into the effective area, in our experiments in Section 6.1, we also present results on the highwayness computed using extended networks.

4.2 Highway Utilization Gain

Despite the fact that network highwaynness can offer valuable insights, it also comes with a shortcoming. By definition, highwaynness indicates whether many trips between locations in a road network benefit from utilizing highways. However, the highwaynness needs to provide the means to quantify this benefit. In other words, we can use the highwaynness to determine whether a random driver can reduce their travel time by following a route that uses a highway, but we have no way to measure how much time this driver is going to save. In addition, highwaynness does not allow us to determine whether a given highway system has the best possible design. Even if we have a highway system G_H for which $hw(G_H) = 1$, that still does not mean that G_H is designed optimally. If one were to design a highway system G'_H that reduces the average travel time for drivers more than G_H, then G'_H would be a better-designed highway system. The network highwaynness cannot capture this difference.

To address these shortcomings, we introduce the *Highway Utilization Gain* (HUG). This new measure considers all possible trips between nodes of a road network and shows how much time drivers can save by using routes that cross highways. In other words, HUG measures the average benefit of using routes that cross highways in terms of travel time reduction. Formally, for a given road network G, HUG is given by:

$$hug(G) = \frac{1}{|N_\Pi|^2} \cdot \sum_{\forall s,t \in N_\Pi} d_G(s,t) - d_{G_\Pi}(s,t)$$

where $d_G(s,t)$ is the network distance computed over the road network G, and $d_{G_\Pi}(s,t)$ is the network distance computed over the plain network G_Π of G, i.e., the subgraph of G that does not contain any highway edges.

Additionally, since HUG captures the benefit only in terms of absolute improvement, we also introduce the *relative highway utilization gain* (RHUG) as

$$rhug(G) = \frac{1}{|N_\Pi|^2} \cdot \sum_{\forall s,t \in N_\Pi} \frac{d_{G_\Pi}(s,t) - d_G(s,t)}{d_{G_\Pi}(s,t)}$$

which captures the relative improvement in travel time per trip.

Computation. The computation of HUG and RHUG is straightforward. Since we have to examine all pairs of nodes in G_Π, the computation is essentially equivalent to the all pairs distances computation. However, since we only need the sum of the cost differences for computing HUG or the sum of the ratios for computing RHUG, there is no need to maintain the entire distance matrices. We simply iterate over all nodes N_Π, compute all distances over G_Π and G, collect either the difference of the travel times (for HUG) or the improvement ratio (for RHUG), and return the average. Algorithm 2 outlines the computation of HUG, and can also compute RHUG by modifying the assignment in Line 6.

Lastly, note that the computation of HUG or RHUG relies on the validity of Property 4, which implies that the plain network G_Π of a road network G is strongly connected. In cases Property 4 does not apply to a given road network, one has to compute HUG or RHUG over the largest connected component(s) of G_Π.

4.3 Extensions for Weighted Trips

In their current form, our measures consider all $s - t$ trips in a road network to be equally important. However, when designing a highway system, urban planners may want to optimize trips between specific areas, e.g., one highly populated area and one with many businesses. To support such scenarios, we extend our measures to enable the consideration of the importance of different trips. Assume that the importance of every $s - t$ trip is stored in a matrix A and is given by $A_{s,t}$. Regarding the network highwaynness, we need to extend the definition of GSEB. Hence, we have

$$gseb^{w_i}_{w_j}(e) = \sum_{\forall s,t \in N} \sum_{\forall p \in P^{w_{i,j}}_{sp} (s \to t) : e' \in p} \frac{A_{s,t}}{\sum_{\forall e'' \in p} w_j(e'')} \cdot \frac{\sigma_{s \to t}(e')}{\sigma_{s \to t}}.$$

This variant of GSEB can be computed by Algorithm 1 by modifying Line 33. Subsequently, for the highway utilization gain we have

$$hug(G) = \sum_{\forall s,t \in N} A_{s,t} \cdot (d(s,t) - d_{\neg hw}(s,t)).$$

The extension of RHUG is similar. Both extensions can be computed by Algorithm 2 by modifying Line 7. As we were unable to find data that indicate the importance of different trips, we have not included these extensions in our experimental evaluation.

5 GAIN-OPTIMAL HIGHWAY SYSTEM

Consider a given road network $G = (N, E)$ with no highways. Designing a highway system G_H for G can be split into two parts. The first part is determining the set of nodes $N_H \subseteq N$ that serve as entry points to the highway system. This step is crucial as it indirectly determines the areas that are served by the highways. The second part is the determination of the connections between the entry points. Given a set of entry points, the most efficient way to connect them is to connect every pair of entry points with an edge e creating a clique. Then set the weight $w(e)$ to the geodetic distance of the entry points. This is impractical. When determining the connections between the entry points, one must also consider that the budget for building a highway system is limited (cf. Property 1). Therefore, the task involves finding the best way to connect the entry points while staying within a given budget.

Based on the above, and following the work of Trietsch [35], we define the *optimal highway system* as follows:

Definition 5.1 (Optimal Highway System). *Given a road network $G = (N, E)$ without any highway edges, a budget b, and an objective function $f(G)$, and a function $c(e)$ that gives the cost to construct an edge $e = (n_i, n_j) \notin E$, the optimal highway system $G_{opt} = (N_{opt}, E_{opt})$ is a network with*

(a) $N_{opt} \subseteq N$,
(b) $E_{opt} \subseteq N \times N : E_{opt} \cap E = \emptyset$, and
(c) $\sum_{\forall e \in E_{opt}} c(e) \leq b$,

such that the road network $G' = (N, E \cup E_{opt})$ maximizes $f(G')$.

Following the definition of our evaluation measures in the previous section, the objective function f can be either the highwaynness, the HUG, the RHUG or some linear combination of these measures.

Algorithm 2: Highway Utilization Gain

Input: Road network $G = (N, E)$
Output: Highway utilization gain h

$G_\Pi \leftarrow$ plain network of G;
$h \leftarrow 0$;
foreach $s \in G_\Pi.nodes$ **do**
  $D_s \leftarrow$ `dijkstra_one_to_many` (G, s, N_Π);
  $D'_s \leftarrow$ `dijkstra_one_to_many` (G_Π, s, N_Π);
  for $i \in [0, |N_\Pi|]$ **do**
    $h \leftarrow h + D'_s[i] - D_s[i]$;
$h \leftarrow h/|N_\Pi|^2$;
return h;

Heuristic Algorithm. Let us assume that the cost to construct a new edge is given by the geodetic distance between its endpoints, and the objective function is given by HUG, i.e., $f(G) = hug(G_H)$. To compute the optimal highway system, one must examine all subsets of nodes and compute connections between nodes in each subset to maximize the objective function f. As such an approach is prohibitively expensive, in what follows, we investigate the heuristic computation of the optimal highway system.

Our *Greedy Steiner Highway Computation* (GSHC) algorithm takes as input a plain network $G_\Pi = (N_\Pi, E_\Pi)$, a budget b, i.e., the maximum allowed construction cost, the maximum number of highway entry points p_{max}, the minimum allowed distance g_{min} between entry points to ensure entry points are distributed throughout the network, and a factor $\alpha \in [0, 1]$. The first step of GSHC is to rank nodes based on how promising they are as entry points. To this end, we modify Algorithm 1 to return the generalized scaled betweenness *gsb* of each node. Then, we store all nodes of the plain network in a list L_N ordered by *gsb*.

Subsequently, we compute the potential connections between all nodes $n_i, n_j \in L_N$, and we filter out all connections where the geodetic distance between n_i and n_j violates a user-defined threshold g_{min}. We store all remaining potential connections in a list L_E along with the sum of the betweenness of the endpoints of every connection, i.e., $gsb_{sum} = gsb(n_i) + gsb(n_j)$ and the gain it yields for the trip from n_i, to n_j, i.e., $l_{gain} = d(n_i, n_j) - d_{geodetic}(n_i, n_j)$. We then order the list by $\alpha \cdot b_{sum} + (1 - \alpha) \cdot l_{gain}$.

The next step involves the greedy selection of the endpoints. For each connection in L_E, we add its endpoint in a set N_{ep} as long as the distance of each endpoint to its nearest neighbor (some node already added) in N_{ep} exceeds g_{min}. Once the number of entry points reaches p_{max}, we stop iterating over L_E. Finally, we compute the Euclidean Steiner Tree [13] $G_H = (N_H, E_H)$ to connect the collected nodes N_{ep}, similar to previous works [11, 35]. If the cumulative construction cost of the edges in E_H exceeds b, we remove the latest added node to N_{ep} and repeat the process until the cumulative cost of E_H is equal to or lower than the budget.

Algorithm 3 outlines the entire process for computing a Euclidean Steiner Tree as the highway system of a road network. Note that our algorithm is intended to determine potential locations of entry points and provide only a rough design of a highway system. For the design to be realistic, it is necessary to solve the highway alignment problem [19, 36], which is out of the scope of our paper.

Algorithm 3: Greedy Steiner Highway Computation

Input: Plain network $G_\Pi(N_\Pi, E_\Pi)$, budget b, factor $\alpha \in [0, 1]$, entry-point limit p_{max}, entry-point min gap g_{min},
Output: Road network $G = (N, E)$

$M^N_{gsb} \leftarrow$ `generalized_scaled_betweenness` (G_Π);
$L_N \leftarrow$ list of nodes in M^N_{gsb} ordered by GSB;
$L_E \leftarrow \emptyset$;
foreach $n_i \in L_N$ **do**
  foreach $n_j \in L_N \setminus \{n_i\}$ **do**
    $d_{geo} \leftarrow$ `geodetic_distance`(n_i, n_j);
    if $d_{geo} > g_{min}$ **then**
      $gsb_{sum} \leftarrow M^N_{gsb}(n_i) + M^N_{gsb}(n_j)$;
      $l_{gain} \leftarrow d(n_i, n_j) - d_{geo}$;
      $L_E \leftarrow L_E \cup (n_i, n_j, gsb_{sum}, l_{gain})$;
order L_E by $\alpha \cdot b_{sum} + (1 - \alpha) \cdot l_{gain}$;
$N_{ep} \leftarrow \emptyset$;
foreach $(n_i, n_j, \_, \_) \in L_e$ **do**
  foreach $n \in \{n_i, n_j\}$ **do**
    if `distance_to_nearest_neighbor` $(n, N_{ep}) > g_{min}$ **then**
      $N_{ep} \leftarrow N_{ep} \cup \{n\}$;
      if $|N_{ep}| == p_{max}$ **then**
        break
$(N_H, E_H) \leftarrow$ `euclidean_steiner_tree` (N_{ep});
while $\sum_{\forall e \in E_H} cost(e) > b$ **do**
  $N_{ep} \leftarrow N_{ep} \setminus \{$latest added endpoint$\}$;
  $(N_H, E_H) \leftarrow$ `euclidean_steiner_tree` (N_{ep});
return $(N_\Pi \cup N_H, E_\Pi \cup E_H)$;

6 EXPERIMENTAL EVALUATION

We present the results of our experimental evaluation in two parts. In the first part, we demonstrate how to use the measures presented in Section 4 to evaluate highways on real road networks. In the second part, we examine the efficacy of our GSHC algorithm in determining shortcuts and computing a rough design of a highway system that optimizes HUG. We obtained all the road networks from OpenStreetMap[1] (OSM) using OSMnx 1.3.0[2]. Table 1 reports the number of nodes, the number of edges, and the number of edges that are labeled as highway segments by OSM, distinguishing those that are reported as links (ramps) for all road networks.

We implemented our algorithms to compute GSEB, HUG, and RHUG in C++ using the Boost Graph Library and GCC 12.2.0. We implemented the GSHC algorithm in Python, but for its most computationally intensive components, we used implementations in C and C++. In particular, for the computation of HUG and RHUG we used graph-tool 2.55[3], while for the computation of the Euclidean Steiner Tree we used GeoSteiner 5.2[4]. We ran all experiments on a Max Book Pro with a 10-core Apple M1 Pro processor and 32 GiB LPDDR5 memory running Mac OS X 13.3.

[1] https://www.openstreetmap.org/
[2] https://osmnx.readthedocs.io/en/stable/
[3] https://graph-tool.skewed.de
[4] http://www.geosteiner.com

Table 1: Road network data sets.

road net.	nodes	edges	highway edges	
			with links	w/o links
Seattle	19,084	50,349	421	144
Singapore	23,313	44,875	1,571	431
Berlin	27,675	72,473	448	163
Indianapolis	35,326	90,530	964	293
Rome	42,839	89,294	756	274
NYC	55,201	139,710	2,768	1,108
Thuringia	94,089	229,070	1,356	388
Connecticut	120,467	304,214	4,229	1,758

Table 2: Evaluation measures for all road networks.

road net.	Highwaynness		Gain	
	w/ links	w/o links	HUG	RHUG
Seattle	0.671	0.850	199.1	15.5
Singapore	0.203	0.193	93.1	7.8
Berlin	0.551	0.851	77.6	4.9
Indianapolis	0.517	0.726	80.2	5.8
Rome	0.585	0.776	193.2	10.6
NYC	0.706	0.813	454.3	22.1
Thuringia	0.55	0.765	752.	13.1
Connecticut	0.785	0.914	1292.8	24.7

6.1 Highway Evaluation Measures

We begin by demonstrating how the measures we presented in Section 4 can be used to evaluate real-world highway systems. For both highwaynness and HUG, we used the travel time as the edge weight to compute shortest paths. Regarding network highwayness, we consider both the case where link edges (ramps) are part of the highway system and where they are not. By examining the results in Table 2, it is clear that the link edges have a negative effect on the highwaynness. That is to be expected as link edges do not speed up traveling but merely work as access points for the edges that do. We also observe that networks designed mainly with cars in mind, e.g., road networks in the USA, demonstrate high highwaynness.

To further understand how highwaynness can be used as an evaluation measure, we report the highwaynness after extending the plain networks of all road networks in Figure 2. More specifically, we obtain the polygon of the road network from OSM, enlarge it, and obtain the new extended road network. Then, we compute the highwaynness considering only the initial highway system. This process allows us to get insight into the effective area of each highway system. We observe that the effective area is similar to the original road network for Seattle. We observe a similar effect in Indianapolis, while, for Berlin and Thuringia, the highwaynness increases with the size of the extended road network. This result indicates that the highway systems in Berlin and Thuringia have been designed to serve much larger regions.

Regarding the highway utilization gain, reported in Table 2, we observe that the largest road networks demonstrate the highest HUG. This is expected as such networks have trips of much larger duration. Regarding RHUG, although we observe a similar tendency, there are some outliers. In particular, we observe that RHUG is relatively low for Indianapolis and Berlin, two networks that have performed well in terms of highwaynness. As such, we can conclude that a highway system that demonstrates high highwaynness may not also yield large HUG and RHUG. Hence, both measures are necessary for the analysis of highway systems.

Another way to evaluate a highway system in terms of highway utilization gain is to measure HUG or RHUG in relation to the length of the highway system. Such an analysis can provide insight into the obtained gain in relation to the financial cost. To this end, Figure 3 reports HUG over the length of the highway system, including link edges (left), and over the percentage of the road network that consists of highways (right). When comparing HUG to the actual length, we observe that the larger the road and highway systems,

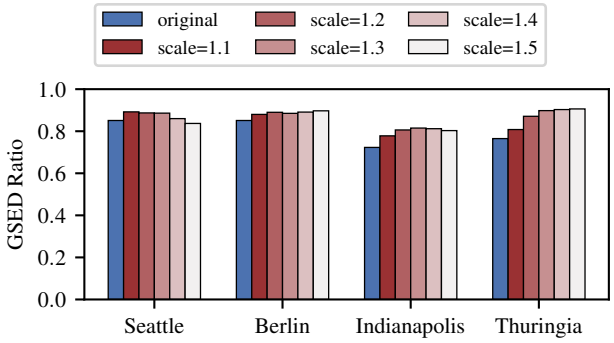

Figure 2: Highwaynness of highway systems w/o links on extended road networks.

the larger the HUG. However, when we plot HUG to the percentage of the road network that consists of highways, we observe that statewide networks, i.e., Thuringia and Connecticut, perform better.

Finally, Table 3 reports the runtime of our algorithms for computing our measures. First, regarding the computation of highwaynness, we observe significant overhead over Brandes' algorithm [6] for the edge betweenness. This cost is due to considering the additional weight and the additional check to remove non-valid paths based on both costs. Regarding HUG and RHUG, we observe that the computation time is significantly higher. This is especially due to the two separate all-pairs shortest path computations, one over the road network and one over the plain network.

Discussion. To sum up, in this section, we have demonstrated how our measures can be used in practice to evaluate existing highway systems. Our measures enable the analysis of various aspects of highway systems. Also, based on our results, both measures are useful as they provide different insights into the efficacy of highway systems. Nevertheless, our current analysis is meant to demonstrate our measures' potency and should not be considered an accurate evaluation of real-world highway systems. More data is necessary to conduct an accurate evaluation, especially traffic data. For such an evaluation, our measures need to be extended for time-dependent road networks, which is a direction we plan to explore in the future.

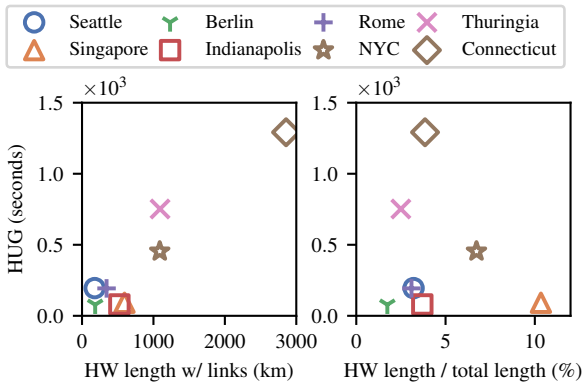

Figure 3: Highway utilization gain over highway length.

6.2 Highway System Computation

In this section, we evaluate the efficacy of our GSHC algorithm. As the parameter with the most significant effect on the algorithm is α, we vary only α in our experiments and set g_{min} to 1 km (geodetic distance). Also, to avoid large computation times, in Lines 4-5 of Algorithm 3, instead of examining all nodes, GHSC considers only to top-10% of the nodes.

To enable a fair comparison with real-world highways, we compare the highways constructed by GSHC with the Euclidean Steiner Tree T_{orig} constructed over the original highway entry points. We then set the budget b to the length T_{orig} and the p_{max} to the number of the original entry points divided by four (the average number of nodes per highway junction). As a result, GSHC computes a highway system with approximately the same number of junctions as real-world highways and, at most, the same length. With this comparison, we aim to show that our GSHC algorithm selects better locations to serve as entry points to highways, resulting in a better highway system than existing ones, thus demonstrating its potential to aid the design of new highways.

Figure 4 reports all the results of our experiments. More specifically, Figures 4a and b report the highways' HUG and RHUG computed using GSHC. In most cases, GSHC computes highways that yield higher HUG and RHUG than the Euclidean Steiner Tree constructed over the original entry points. Regarding parameter α we observe that lower values, i.e., giving slightly more importance to the local gain rather than the betweenness of nodes, yield slightly better results. In Figure 4c, we report the average length of the Steiner Trees. Except for Seattle, the Euclidean Steiner Tree constructed over the nodes selected by GSHC is much smaller than the one constructed over the original nodes. Seattle is the only road network for which GSHC terminated because of the budget constraint; in all other road networks, GSHC terminates by reaching p_{max}. Furthermore, by Figure 4c alongside Figures 4a and b, we observe that GSHC achieves higher HUG and RHUG while constructing a much smaller tree. Regarding the highwaynness reported in Figure 4d, we observe that in the vast majority of cases, the Steiner Tree-based highway system created over the nodes selected by GHSC demonstrates higher highwaynness than the one created over the original entry points. Also, regarding α, similar to HUG and RHUG, we observe slightly better results for $\alpha \leq 0.5$ values.

Table 3: Evaluation measures computation time (seconds).

road net.	Brandes	Highwaynness	HUG/RHUG
Seattle	27	49	134
Singapore	36	64	178
Berlin	58	108	280
Indianapolis	97	185	453
Rome	138	250	657
NYC	263	476	820
Thuringia	871	1,493	3,396
Connecticut	1,821	2,916	5,931

Table 4: GSHC runtime varying α (seconds).

road net.	0.1	0.3	0.5	0.7	0.9
Seattle	462	349	414	415	412
Singapore	1,733	1,650	1,593	1,337	1,094
Berlin	975	1,009	1,010	1,018	1,007
Indianapolis	1,183	1,175	1,208	1,162	1,154

Finally, in Table 3, we report the runtime of our GHSC algorithm varying α. Our results do not indicate that α significantly affects the performance. More specifically, for Seattle and Berlin, the runtime seems to remain unaffected by α; in Singapore, we observe a consistent reduction in runtime with an increasing α; in Indianapolis, we observe the highest runtime for $\alpha = 0.5$.

7 CONCLUSIONS

In this paper, we investigated the evaluation of highway systems from a network analysis perspective and introduced two novel measures. The network highwaynness, a measure based on a novel variant of the edge betweenness, provides insight on whether a given highway system works as a backbone on the road network it has been created for. The highway utilization gain quantifies the benefit of a highway system in terms of travel time per trip. Additionally, we presented a first-cut heuristic algorithm to compute a rough design of an efficient highway system. Our experiments on real road networks demonstrate the practical value of our measures and investigate the performance of our heuristic algorithm. In the future, apart from the directions stated in the paper, we plan to extend our measures to time-dependent road networks. Furthermore, we plan to integrate our solutions into a highway systems analysis framework and a highway alignment solution to support further transportation network analysis tasks.

ACKNOWLEDGMENTS

This work is supported by Grant No. CH 2464/1-1 of the Deutsche Forschungsgemeinschaft (DFG)

REFERENCES

[1] Ittai Abraham, Daniel Delling, Amos Fiat, Andrew V. Goldberg, and Renato F. Werneck. 2016. Highway Dimension and Provably Efficient Shortest Path Algorithms. *J. ACM* 63, 5, Article 41 (2016), 41:1-41:26 pages.

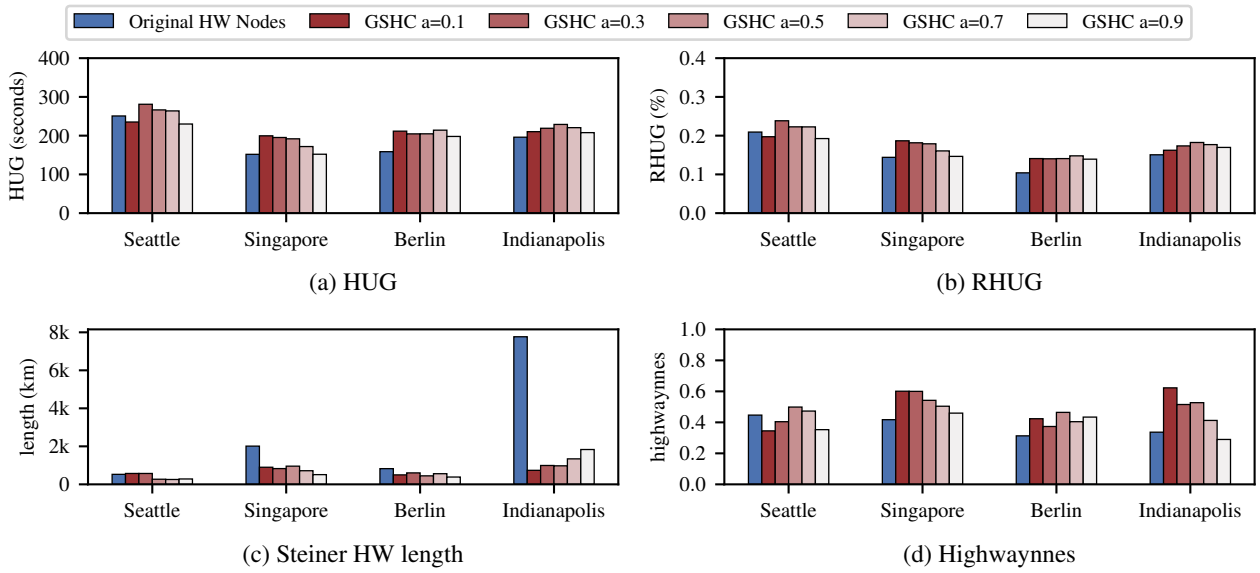

Figure 4: Highway utilization gain for highway systems created as Euclidean Steiner Trees.

[2] Takuya Akiba, Yoichi Iwata, Kenichi Kawarabayashi, and Yuki Kawata. 2014. Fast Shortest-path Distance Queries on Road Networks by Pruned Highway Labeling. In *SIAM ALENEX*. 147–154.

[3] Jac M. Anthonisse. 1971. The Rush in a Directed Graph. *Stichting Mathematisch Centrum. Mathematische Besliskunde* (1971).

[4] Stephen P Borgatti and Martin G Everett. 2006. A graph-theoretic perspective on centrality. *Social networks* 28, 4 (2006), 466–484.

[5] Ulrik Brandes. 2001. A Faster Algorithm for Betweenness Centrality. *The Journal of Mathematical Sociology* 25, 2 (2001), 163–177.

[6] Ulrik Brandes. 2008. On Variants of Shortest-Path Betweenness Centrality and their Generic Computation. *Social Networks* 30, 2 (2008), 136–145.

[7] Zaiben Chen, Heng Tao Shen, and Xiaofang Zhou. 2011. Discovering popular routes from trajectories. In *IEEE ICDE*. IEEE, 900–911.

[8] Theodoros Chondrogiannis and Johann Gamper. 2016. ParDiSP: A Partition-based Framework for Distance and Shortest Path Queries on Road Networks. In *IEEE MDM*. 242–251.

[9] Kousik Das, Sovan Samanta, and Madhumangal Pal. 2018. Study on centrality measures in social networks: a survey. *Social network analysis and mining* 8 (2018), 1–11.

[10] Edsger W. Dijkstra. 1959. A Note on Two Problems in Connexion with Graphs. *Numer. Math.* 1, 1 (1959), 269–271.

[11] Ding-Zhu Du, Frank K Hwang, and Guoliang Xue. 1999. Interconnecting highways. *SIAM Journal on Discrete Mathematics* 12, 2 (1999), 252–261.

[12] Enrico Ferrero, Stefano Alessandrini, and Alessia Balanzino. 2016. Impact of the electric vehicles on the air pollution from a highway. *Applied energy* 169 (2016), 450–459.

[13] Edgar N Gilbert. 1967. Minimum cost communication networks. *Bell System Technical Journal* 46, 9 (1967), 2209–2227.

[14] Edgar N Gilbert and Henry O Pollak. 1968. Steiner minimal trees. *SIAM J. Appl. Math.* 16, 1 (1968), 1–29.

[15] Bernard E Howard, Zacarias Bramnick, and Jocelyn FB Shaw. 1968. Optimum curvature principle in highway routing. *Journal of the Highway Division* 94, 1 (1968), 61–82.

[16] Jyh-Cherng Jong, Manoj K Jha, and Paul Schonfeld. 2000. Preliminary highway design with genetic algorithms and geographic information systems. *Computer-Aided Civil and Infrastructure Engineering* 15, 4 (2000), 261–271.

[17] Jyh-Cherng Jong and Paul Schonfeld. 1999. Cost functions for optimizing highway alignments. *Transportation Research Record* 1659, 1 (1999), 58–67.

[18] Jyh-Cherng Jong and Paul Schonfeld. 2003. An evolutionary model for simultaneously optimizing three-dimensional highway alignments. *Transportation Research Part B: Methodological* 37, 2 (2003), 107–128.

[19] Min-Wook Kang, Manoj K Jha, and Paul Schonfeld. 2012. Applicability of highway alignment optimization models. *Transportation Research Part C: Emerging Technologies* 21, 1 (2012), 257–286.

[20] Yimin Lin and Kyriakos Mouratidis. 2015. Best upgrade plans for single and multiple source-destination pairs. *GeoInformatica* 19 (2015), 365–404.

[21] Wuman Luo, Haoyu Tan, Lei Chen, and Lionel M Ni. 2013. Finding time period-based most frequent path in big trajectory data. In *ACM SIGMOD*. 713–724.

[22] Yongchang Ma, Mashrur Chowdhury, Adel Sadek, and Mansoureh Jeihani. 2009. Real-time highway traffic condition assessment framework using vehicle–infrastructure integration (VII) with artificial intelligence (AI). *IEEE Transactions on Intelligent Transportation Systems* 10, 4 (2009), 615–627.

[23] Neville A Parker. 1977. Rural highway route corridor selection. *Transportation Planning and Technology* 3, 4 (1977), 247–256.

[24] Ciaran S Phibbs and Harold S Luft. 1995. Correlation of travel time on roads versus straight line distance. *Medical Care Research and Review* 52, 4 (1995), 532–542.

[25] Ioannis Politis, Michalis Kyriakoglou, Georgios Georgiadis, and Panagiotis Papaioannou. 2020. Evidence from highway drivers in Greece showing toll avoidance and utility of alternative routes. *Transportation Research Record* 2674, 9 (2020), 948–958.

[26] Dimitris Sacharidis, Kostas Patroumpas, Manolis Terrovitis, Verena Kantere, Michalis Potamias, Kyriakos Mouratidis, and Timos Sellis. 2008. On-line discovery of hot motion paths. In *EDBT*. 392–403.

[27] Yasmeen Mohammed Sameer, Adil Nuhair Abed, and Khamis Naba Sayl. 2023. Geomatics-based approach for highway route selection. *Applied Geomatics* 15, 1 (2023), 161–176.

[28] Peter Sanders and Dominik Schultes. 2005. Highway Hierarchies Hasten Exact Shortest Path Queries. In *ESA*. 568–579.

[29] Jagan Sankaranarayanan, Hanan Samet, and Houman Alborzi. 2009. Path oracles for spatial networks. *Proceedings of the VLDB Endowment* 2, 1 (2009), 1210–1221.

[30] Dominik Schultes and Peter Sanders. 2007. Dynamic Highway-Node Routing. In *WEA*. 66–79.

[31] Darren M Scott, David C Novak, Lisa Aultman-Hall, and Feng Guo. 2006. Network robustness index: A new method for identifying critical links and evaluating the performance of transportation networks. *Journal of Transport Geography* 14, 3 (2006), 215–227.

[32] Christian Sommer. 2014. Shortest-Path Queries in Static Networks. *Comput. Surveys* 46, 4 (2014), 1–31.

[33] Taoran Song, Paul Schonfeld, and Hao Pu. 2023. A Review of Alignment Optimization Research for Roads, Railways and Rail Transit Lines. *IEEE Transactions on Intelligent Transportation Systems* (2023).

[34] Sabine Storandt. 2018. Region-Aware Route Planning. In *W2GIS Symp.* 101–117.

[35] Dan Trietsch. 1987. Comprehensive design of highway networks. *Transportation Science* 21, 1 (1987), 26–35.

[36] Dan Trietsch. 1987. A family of methods for preliminary highway alignment. *Transportation Science* 21, 1 (1987), 17–25.

[37] Ling-Yin Wei, Yu Zheng, and Wen-Chih Peng. 2012. Constructing popular routes from uncertain trajectories. In *ACM SIGKDD*. 195–203.