Stochastic Route Planning for Electric Vehicles

Payas Rajan
Department of Computer Science, University of California, Riverside, CA, USA

Chinya V. Ravishankar
Department of Computer Science, University of California, Riverside, CA, USA

Abstract

Electric Vehicle routing is often modeled as a generalization of the energy-constrained shortest path problem, taking travel times and energy consumptions on road network edges to be deterministic. In practice, however, energy consumption and travel times are stochastic distributions, typically estimated from real-world data. Consequently, real-world routing algorithms can make only probabilistic feasibility guarantees. Current stochastic route planning methods either fail to ensure that routes are energy-feasible, or if they do, have not been shown to scale well to large graphs. Our work bridges this gap by finding routes to maximize on-time arrival probability and the set of non-dominated routes under two criteria for stochastic route feasibility: $\mathbb{E}$-feasibility and p-feasibility. Our $\mathbb{E}$-feasibility criterion ensures energy-feasibility in expectation, using expected energy values along network edges. Our p-feasibility criterion accounts for the actual distribution along edges, and keeps the stranding probability along the route below a user-specified threshold p. We generalize the charging function propagation algorithm to accept stochastic edge weights to find routes that maximize the probability of on-time arrival, while maintaining $\mathbb{E}$- or p-feasibility. We also extend multi-criteria Contraction Hierarchies to accept stochastic edge weights and offer heuristics to speed up queries. Our experiments on a real-world road network instance of the Los Angeles area show that our methods answer stochastic queries in reasonable time, that the two criteria produce similar routes for longer deadlines, but that $\mathbb{E}$-feasibility queries can be much faster than p-feasibility queries.

2012 ACM Subject Classification Mathematics of computing → Graph algorithms; Theory of computation → Shortest paths; Theory of computation → Stochastic control and optimization

Keywords and phrases Stochastic Routing, Electric Vehicles, Route Planning Algorithms

Digital Object Identifier 10.4230/LIPIcs.SEA.2022.15

Acknowledgements The authors would like to thank the Mapbox Community team for access to the Mapbox Traffic Data used in our experiments.

1 Introduction

Routing methods for Electric Vehicles (EVs) cannot just minimize travel time, but must also address driver *range anxiety*. EVs have limited battery capacity, charging times are long, and the charging infrastructure remains relatively sparse, so a major concern is *stranding*, which occurs when the battery's State of Charge (SoC) reaches zero en route. A route for an EV is hence considered *feasible* only if the SoC along the route never reaches zero.

Merely trying to minimize travel time greatly increases the risk of stranding, since energy consumption is typically quadratic in vehicle speed. Standard formulations such as [9, 14, 30] model EV routing as a generalization of the NP-hard Constrained Shortest Path problem [26, 58], and seek to minimize travel time while maintaining a non-zero SoC along the route. Some recent work [7, 36] even tries to exercise direct control over travel time and route feasibility, by pre-determining and assigning optimal EV travel speeds for each road segment.

20th International Symposium on Experimental Algorithms (SEA 2022).
Editors: Christian Schulz and Bora Uçar; Article No. 15; pp. 15:1–15:17
Leibniz International Proceedings in Informatics
LIPICS Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

Most existing problem formulations also assume that travel times and energy consumption values on road network edges are deterministic. In practice, both travel time and energy consumption are stochastic, and difficult to estimate reliably [2, 17, 46]. In such a context, even routing algorithms offering strong feasibility guarantees are of limited value. Approaches that pre-determine and assign vehicle speeds for each edge are not practical, since speed is not always a variable under driver control, but rather a *result* of prevalent traffic conditions.

Consequently, travel times and route feasibility may only be defined *probabilistically*. Stochastic routing algorithms [12, 20, 40, 39, 38, 42, 43, 44, 27, 59], model travel times along network edges as random variables with given probability distributions, and allow richer query semantics, such as finding paths to maximize the probability of arrival before a deadline [15], or finding the latest departure time and path to guarantee a certain probability of arrival before a deadline [35]. Despite recent improvements [47], stochastic routing is typically several orders of magnitude slower than deterministic routing, since obtaining travel time distributions along a path requires very expensive convolutions of its edge distributions.

Limited work exists on stochastic route planning for EVs. Chen et al. [13] assume lognormal travel-time and Gaussian energy-consumption distributions, and uses bicriteria search to find the Pareto-optimal routes optimizing energy consumption and travel time reliability. Jafari et al. [25] allow arbitrary distributions of travel times on edges and charging stations, and uses multicriteria search to minimize the cost of charging and travel time, subject to a minimum reliability threshold, on small synthetic graphs with randomly generated edge weights and charging station placements. Shen et al. [51] allow correlated travel time distributions between edges, and use bicriteria search on travel times and energy consumptions. However, they assume deterministic energy consumptions, and run experiments on a network of only a few hundred vertices.

1.1 Our Contributions

We study EV routing when both travel times and energy consumptions are stochastic. The travel time on each edge $e \in E$ of a road network $G = \langle V, E \rangle$ is always a random variable $\boldsymbol{T}_e$ with a known distribution (estimated from data, say). The energy consumption along e is a function ε_e of EV speed and distance. We introduce two probabilistic definitions of route feasibility: We say that a route is $\mathbb{E}$-feasible if the SoC of the EV is always maintained above zero *in expectation*, and p-feasible if the probability of route feasibility is *at least* p. We show how to enhance stochastic routing queries for travel times with these feasibility criteria to find non-dominated feasible routes and probabilistic budget feasible routes. Our work addresses the four types of stochastic routing queries in the cells of the following table:

	$\mathbb{E}$-Feasibility	p-Feasibility
Non-Dominated Routes	✓	✓
Probabilistic Budget Routes	✓	✓

We address these queries by generalizing the Charging Function Propagation algorithm of [8, 9] to accommodate stochastic edge weights. We evaluate our methods experimentally using a realistic road network instance with travel time distributions derived from traffic speeds observed over four and a half months in the Los Angeles area, and real-world elevations and charging station locations. Further, we apply an uncertain variant of Contraction Hierarchies [21] to speed up our queries and present results. Our results indicate that in general, $\mathbb{E}$-feasible routing queries can be computed much faster than p-feasible queries, and produce similar routes for longer routes with higher time budgets.

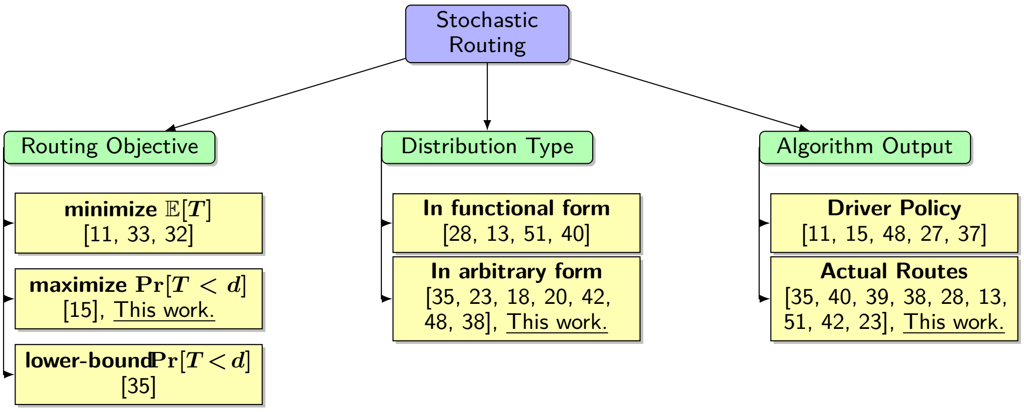

Figure 1 Stochastic route planning, classified by routing objective, edge distribution, and result. Our work finds energy-feasible routes that maximize probability of arrival before deadline.

2 Related Work

EV routing has been typically modeled as energy-aware routing, with objective functions ranging from minimizing the total energy consumption [14, 49], to minimizing travel time while maintaining route *feasibility* [3, 9, 36, 54], to multicriteria search on both travel time and energy consumption [22]. In contrast, most prior work on routing Internal Combustion-based vehicles merely minimizes the total travel time [4, 53].

More attention is now being paid to real-world issues. Examples include allowing battery-swapping stations [56], partial recharges at stations [9, 30, 8, 57] and maintaining battery buffer to relieve range anxiety [45, 24, 18]. Many challenges remain, however. The energy consumption models are imperfect, and factors such as battery wear, driver aggressiveness [31], or traffic conditions are hard to model, but can have a significant impact. Data also suggests that drivers may prefer familiar paths to shortest paths [60, 29].

2.1 Stochastic Route Planning

Stochastic route planning goes back to [20], which attempted an exact solution for the shortest path problem in stochastic graphs, using Monte Carlo simulations to derive path weights. It is now known that driver behavior changes if travel time is stochastic [19, 52], so problem variants have been explored. Existing work can be categorized in three ways: by objective function, the forms assumed for edge probability distributions, and by targeted outcome. For conciseness, we discuss our categorization here, and show references in Figure 1.

By routing objective. Routing objectives can be quite varied, such as minimizing expected time [11, 33, 32], maximizing the on-time-arrival probability [15], maintaining on-time-arrival probability above a given threshold [35]. Some works [51, 13, 25] apply stochastic routing algorithms to EVs, while others [1] route multiple EVs collaboratively, on-line.

By distribution. The edge distributions assumed can have a functional form, or be arbitrary without a closed form. This choice also affects the edge weight representations used. For functional forms, storing the distributional parameters suffices, but arbitrary distributions require more space-intensive representations such as histograms. Further, with functional forms, a small number of observations can suffice to capture real-world behaviour, but histograms require much more data. Edge weight representations have been shown to significantly affect the runtime performance of stochastic shortest path queries [44, 47].

Output. Adaptive methods [41, 35] output policies for drivers to make routing decisions on-line, as they reach vertices or edges during the drive. In contrast, a-priori approaches produce routes before travel begins. [35] showed that adaptive approaches can produce strictly better solutions than the a-priori approaches, but are much more computationally expensive. Recently, performance improvements to policy-based approaches, such as the Stochastic On-Time Arrival problem have also been proposed [48, 37, 27].

3 Problem Setup

A road network is a directed graph $G = \langle V, E\rangle$ where V is the set of vertices and $E : V \times V$ is the set of edges. An s-t path $P = [s = v_1, v_2 \cdots , v_n = t]$ is a sequence of adjacent vertices in the road network G. A set $C \subseteq V$ is marked as *charging stations*.

▶ **Definition 1** (State of Charge). *The State of Charge (SoC) of an EV is the charge status of the EV's battery, lying between 0 and the battery capacity M. We denote the SoC on arrival at a vertex v by ${}_v\beta$ and the SoC at departure from v by β_v. We have $\beta_v \geq {}_v\beta$ if the EV charges its batteries at node v, and $\beta_v = {}_v\beta$ otherwise.*

Each $c \in C$ has a monotonically increasing, piecewise-linear charging function Φ_c such that $\Phi_c({}_c\beta, t_c) \mapsto \beta_c$ where t_c is charging time. We require ${}_c\beta \geq 0$, and $\beta_c \leq M$ [45].

▶ **Definition 2** (Leg and Prefix). *A subpath $L = [c_1, \dots, v, \dots, c_2]$ is a* leg *of path P iff c_1, c_2 are successive charging stations along P. Each $\lambda_v = [c_1, \dots, v]$, $v \neq c_2$ is a* prefix *of L.*

3.1 Travel Times and Energy Depletion

The travel time along each edge e is a random variable $\boldsymbol{T}_e$ with a known probability distribution. For problem tractability, we assume that the EV travels on e at a uniform speed drawn from the distribution $\boldsymbol{T}_e$. This is reasonable, since variable travel time on an edge can be easily modeled by splitting an edge into several smaller edges.

Let $e_1, e_2, \dots, e_{n-1}$ be the edges along path P, and let e_k have travel time distribution T_k. The aggregate travel time distribution for the path P is $\boldsymbol{T}_P = \boldsymbol{T}_1 * \boldsymbol{T}_2 * \cdots * \boldsymbol{T}_{n-1}$, where $*$ denotes linear convolution. Let $\boldsymbol{T}_\emptyset$ be the Dirac "delta" distribution defined so that $\boldsymbol{T}_\emptyset(0) = 1$ and $\boldsymbol{T}_\emptyset(x) = 0$ at $x \neq 0$. Now, $\boldsymbol{T}_\emptyset$ functions as a convolution identity, so $\boldsymbol{T}_\emptyset * \boldsymbol{T}_P = \boldsymbol{T}_P$.

We assign to each edge e a function $\varepsilon_e : \mathbb{R}^+ \to \mathbb{R}$, which maps a travel time to the battery energy depleted by travel along e. The total energy depletion is the sum of the work done along e by the EV against air resistance, rolling resistance, and against gravity. The wind resistance grows quadratically with speed. If t is the travel time along edge e, these three terms cause $\varepsilon_e(t)$ to assume the form

$$\varepsilon_e(t) = \frac{a_e}{t^2 - b_e} + \frac{c_e}{t} + d_e. \tag{1}$$

where a_e, b_e, c_e, d_e are fixed coefficients for each edge e. We can derive the edge *energy depletion distribution* $\boldsymbol{D}_e$ from the travel time distribution $\boldsymbol{T}_e$ using Equation 1, thereby associating probabilities with energy depletions. A path may have negative energy depletion; EVs have regenerative brakes, and can accumulate charge, say, when going down a slope.

We can also aggregate energy depletion distributions using convolutions. If $e_1, e_2, \dots, e_{n-1}$ are the edges along a path P, and edge e_i has the depletion distribution $\boldsymbol{D}_i$, the aggregate energy depletion distribution for P is $\boldsymbol{D}_P = \boldsymbol{D}_1 * \boldsymbol{D}_2 * \cdots * \boldsymbol{D}_{n-1}$. By analogy with $\boldsymbol{T}_\emptyset$, we define $\boldsymbol{D}_\emptyset$ to be the Dirac "delta" function corresponding to energy depletion, so that $\boldsymbol{D}_\emptyset * \boldsymbol{D}_P = \boldsymbol{D}_P$. Sometimes, as with expected-feasibility queries, it suffices to add expectations directly, since $\mathbb{E}[\boldsymbol{D}_1 * \boldsymbol{D}_2] = \mathbb{E}[\boldsymbol{D}_1] + \mathbb{E}[\boldsymbol{D}_2]$.

Table 1 Symbols used in this paper.

Sym	Meaning	Sym	Meaning
$\boldsymbol{T}_P$	Travel time distribution on path P	$\boldsymbol{T}_\emptyset$	Convolution identity for $\boldsymbol{T}$
$\boldsymbol{D}_P$	Energy depletion distribution on path P	$\boldsymbol{D}_\emptyset$	Convolution identity for $\boldsymbol{D}$
δ_λ	Depletion function for leg prefix λ	$\delta_\emptyset$	Depletion function for null path
${}_u\beta$	SoC on arrival at vertex u	β_u	SoC at departure from vertex u
Φ_c	Charging function at charging station c	ε_e	Energy depletion function on edge e

3.2 $\mathbb{E}$-Feasible Routing

In this class of queries, we assume that the travel times are stochastic, but define feasibility in terms of the expectations for the energy depletion distributions. Say that an EV starts from vertex s with State of Charge (SoC) $\beta_s \in [0, M]$ and wishes to travel to vertex t along the s-t path P. Let leg $L = [c, \dots, c']$ of P lie between charging stations c and c' along P.

▶ **Definition 3** ($\mathbb{E}$-Feasible Path). *Leg L is* expected-feasible *(or* $\mathbb{E}$-feasible*) iff $\mathbb{E}[\boldsymbol{D}_\lambda] \leq \beta_c$, where $\boldsymbol{D}_\lambda$ is the depletion distribution for all prefixes λ of L, and β_c is the EV's SoC when it departs c. A path $P = [L_1, L_2, \dots, L_n]$ is $\mathbb{E}$-feasible iff each of its legs L_i is $\mathbb{E}$-feasible.*

We consider two $\mathbb{E}$-feasible queries:

▶ **Query 4** (Non-Dominated $\mathbb{E}$-feasible Paths). *Find the set of $\mathbb{E}$-feasible s-t paths such that their travel time distributions are not dominated by any other path.*

▶ **Query 5** (Probabilistic Budget $\mathbb{E}$-feasible Path). *Find an $\mathbb{E}$-feasible s-t path that maximizes the probability of reaching t before a given deadline d.*

3.3 p-Feasible Routing

▶ **Definition 6** (p-Feasible Path). *A path P with legs $L_1, L_2, \dots, L_n$ is* p-feasible *iff the probability of the EV not being stranded along P is at least p, the non-stranding probability.*

The non-stranding probability of P is given by the product of non-stranding probabilities of P's legs. For P to be p-feasible, each of its legs must have a non-stranding probability of at least p. We consider two p-feasible queries:

▶ **Query 7** (Non-Dominated p-Feasible Paths). *Find the set of s-t paths whose travel time distributions are not dominated by any other path, and which ensure that probability not being stranded is at least p.*

▶ **Query 8** (Probabilistic Budget p-Feasible Paths). *Find an s-t path which maximizes the probability of reaching t before a given deadline d, while keeping the probability of not being stranded is at least p.*

4 Charging Function Propagation for $\mathbb{E}$-Feasible Routing

The CFP algorithm of [9] uses only deterministic edge weights, but we show how to extend it to answer expected-feasible stochastic shortest path queries. As in [9], we ensure that the SoC on departing a charging station suffices to complete the ensuing leg. Our Dijkstra's search labels maintain the set of all possible tradeoffs between charging time and the resulting SoC.

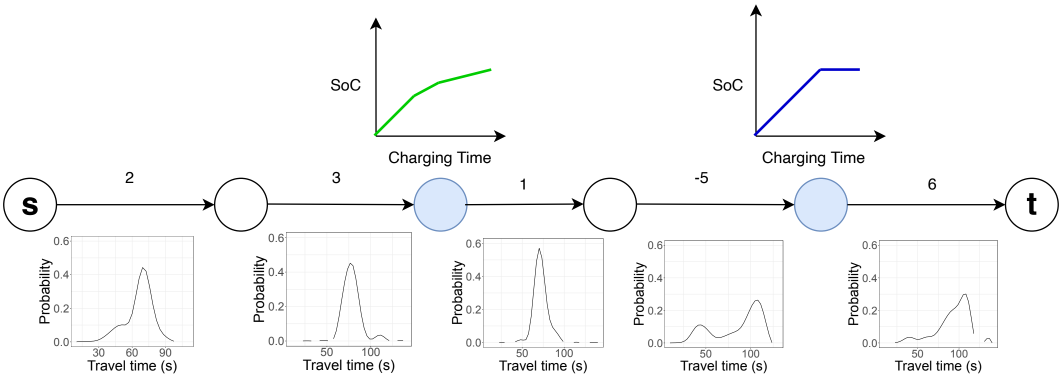

Figure 2 $\mathbb{E}$-feasible queries. Edges have two weights: a travel time distribution (below), and an expected energy depletion (above). Shaded nodes are charging stations, with piecewise-linear charging functions. The CFP search propagates travel time distributions using convolutions.

However, complexities arise since we use stochastic travel times. The deterministic case can simply use a min-priority queue ordered by travel times, but distributions can be ordered in different ways. For simplicity, we will use *usual stochastic ordering* [50] to order the travel time distributions in the priority queue, under which two random variables X and Y obey $X \preceq Y$ iff $\Pr[X > x] \leq \Pr[Y > x], \forall x \in (-\infty, \infty)$. Other stochastic orderings, such as the hazard rate or likelihood ratio ordering, may result in interesting tradeoffs for the EV, but are beyond the scope of this paper. Also, deterministic travel times can be simply added along a path, but travel time distributions must be convolved to aggregate travel time distributions.

For expected-feasible routes, we will use stochastic travel times, but expected values for energy depletion. That is, let $e_1, e_2, \ldots, e_n$ be the edges comprising a path P, and let edge e_i have travel time and energy depletion distributions $\boldsymbol{T}_i$ and $\boldsymbol{D}_i$. For expected-feasible routing, the aggregate travel time distribution $\boldsymbol{T}_P = \boldsymbol{T}_1 * \boldsymbol{T}_2 * \cdots * \boldsymbol{T}_n$, and the aggregate energy depletion value is $\mathbb{E}[\boldsymbol{D}_P] = \mathbb{E}[\boldsymbol{D}_1] + \mathbb{E}[\boldsymbol{D}_2] + \ldots + \mathbb{E}[\boldsymbol{D}_n]$.

4.1 The Depletion Function Along Route Legs

Even if the energy depletion over leg $L = [c_1, \ldots, v, \ldots, c_2]$ is deterministic with value E_L, departing c_1 with an SoC of $\beta_{c_1} = E_L$ may not suffice to complete L. For instance, L may go up a hill, climbing which requires more energy than β_{c_1}. Similarly, ${}_{c_2}\beta$, the arrival SoC at c_2, may not equal $\beta_{c_1} + E_L$ when $E_L < 0$, since the SoC can never exceed M.

Consider a prefix $\lambda = [c, \ldots, v]$ of some leg that starts with charging station c. Let S_λ be the minimum starting SoC required to traverse λ, E_λ be the maximum ending SoC possible at v, and let $\mathrm{C}_\lambda = \mathbb{E}[\boldsymbol{D}_\lambda]$. The *depletion function* δ_λ (similar to *SoC profiles* in [8, 9]) for prefix λ maps the SoC at the start of λ to the SoC at the end of λ, and is defined as

$$\delta_\lambda(\beta_c) = {}_v\beta = \begin{cases} -\infty, & \text{if } \beta_c < \mathrm{S}_\lambda, \\ \mathrm{E}_\lambda, & \text{if } \beta_c - \mathrm{C}_\lambda > \mathrm{E}_\lambda, \\ \beta_c - \mathrm{C}_\lambda, & \text{otherwise.} \end{cases} \tag{2}$$

The depletion function for a null path comprising a single vertex s is the *identity* depletion function $\delta_\emptyset : \beta_s \mapsto \beta_s$. Let $P_1 = [v_i, v_{i+1}, \ldots, v_j]$ and $P_2 = [v_{j+1}, v_{j+2}, \ldots, v_k]$ be contiguous segments, and $P = P_1P_2 = [v_i, \ldots, v_k]$ be their concatenation. In this case, we have $\mathrm{S}_P = \max\{\mathrm{S}_{P_1}, \mathrm{C}_{P_1} + \mathrm{S}_{P_2}\}$, $\mathrm{E}_P = \min\{\mathrm{E}_{P_2}, \mathrm{E}_{P_1} - \mathrm{C}_{P_2}\}$ and $\mathrm{C}_P = \mathrm{C}_{P_1} + \mathrm{C}_{P_2}$.

4.2 Dijkstra Search for $\mathbb{E}$-feasible Routes

We find expected-feasible paths via Dijkstra search using two types of priority queues: the global queue Q_G holds the travel time distributions from s to all other vertices in the road network G, and per-vertex queues $L_u(v)$ and $L_s(v)$. $L_u(v)$ and $L_s(v)$ hold the unsettled and settled search labels at vertex v respectively. All priority queues are ordered by $\boldsymbol{T}_{[s\ldots v]}$ using the usual stochastic ordering $\preceq$ defined above. Each label in $L_s(v)$ corresponds to an s-v path already known to be feasible, and gives the required charging time at the last charging station. Consequently, as in [9], we maintain the invariant that the minimum element in $L_u(v)$ is not dominated by any label in $L_s(v)$.

The EV leaves s having acquired an SoC of β_s at s, so we treat s as a charging station, by default. One of our major challenges in the search will be to determine at which stations to charge, and for how long. Our search hence remembers the last charging station c along the route in the search labels, since dropping to an SoC below a permissible threshold signals the need to include a charging time at c, and update route times accordingly.

4.2.1 The Search Algorithm

When the search reaches vertex v, the label at v is a four-tuple $\langle \boldsymbol{T}_{[s\ldots v]}, {}_c\beta, c, \delta_{[c\ldots v]}\rangle$, where $\boldsymbol{T}_{[s\ldots v]}$ is the travel time distribution for the subpath $[s\ldots v]$, c is the last charging station en route from s to v, ${}_c\beta$ is the arrival SoC at c, and $\delta_{[c\ldots v]}$ is the depletion function of the subpath $[c\ldots v]$. We note that the charging times at some charging stations may be zero.

A label is extracted from $L_u(v)$ on each search iteration, where v is the minimum-travel time vertex in Q_G. It is then settled, and added to $L_s(v)$. A label in $L_s(v)$ represents a path from s to v that we know to be feasible, along with the exact charging time at the last charging station. A label in $L_u(v)$ represents a potentially feasible path that we haven't checked for feasibility. If an unsettled label in $L_u(v)$ is dominated by a label in $L_s(v)$, we can discontinue search along that path and discard that label, because we already know a better feasible path. The search proceeds as follows:

1. *At s*: Mark s as a charging station. Add the label $\langle \boldsymbol{T}_\emptyset, {}_s\beta, s, \delta_\emptyset\rangle$ to $L_u(s)$.
2. *At a non-charging vertex v*: Let $\ell = \langle \boldsymbol{T}_{[s\ldots v]}, {}_c\beta, c, \delta_{[c\ldots v]}\rangle$ be the label extracted from $L_u(v)$. Since ℓ indicates that c is the last charging station encountered, add label $\langle \boldsymbol{T}_{[s\ldots v]}, \delta_{[c\ldots v]}({}_c\beta), c, \delta_{[c\ldots v]}\rangle$ to $L_u(v)$ and update the travel times for v in Q_G.
3. *At a charging vertex v*: Let label $\ell = \langle \boldsymbol{T}_{[s\ldots v]}, {}_c\beta, c, \delta_{[c\ldots v]}\rangle$ be the minimum element extracted from $L_u(v)$. Let t_c be the charging time at the last charging station c, so that $\beta_c = \Phi_c({}_c\beta, t_c)$ is the SoC when the EV departs c.
The CFP algorithm of [8, 9] uses only deterministic travel times, but our travel times are distributions. As [8] shows, however, the charging times corresponding to the breakpoints of the charging function $\Phi_c(\cdot)$ capture the information required to make the required tradeoffs between charging times and travel times. To see how we approach the problem, let τ represent some value for the travel time from s to v, and compute
$$b_\ell(t_c,\tau) := \begin{cases} \delta_{[c\ldots v]}(\beta_c) & \text{if } t_c > 0 \text{ and } \boldsymbol{T}_{[s\ldots v]}(\tau) > 0 \\ -\infty & \text{otherwise} \end{cases}$$
Since the charging function $\Phi_c(\cdot)$ is assumed to be piecewise linear, its breakpoints induce breakpoints for b_ℓ. For a given value of τ we need to create one label per breakpoint of b_ℓ [8]. For a fixed τ and each breakpoint $B = (t_B, \mathrm{SoC}_B)$ of b_ℓ, we add to $L_u(v)$ the label $\langle t_B, \mathrm{SoC}_B, v, \boldsymbol{T}_\emptyset\rangle$, and update the travel times to v in Q_G.

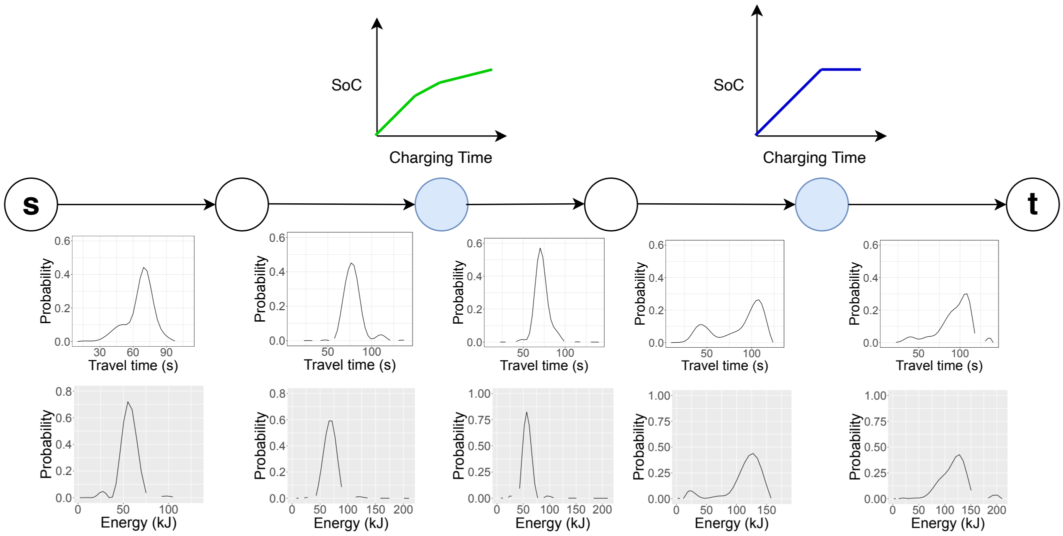

Figure 3 p-Feasible queries. Travel time and energy depletion are both distributions., propagated by the CFP search using convolutions. While the non-dominated search stops only when Q_G becomes empty, the probabilistic budget route search can stop when $\boldsymbol{T}_P(t)$ drops to 0.

In principle, τ can take an infinite number of values. We handle this difficulty by discretizing the domain of $\boldsymbol{T}_e$. We use histograms to represent $\boldsymbol{T}_e$ in our implementation, and generate only one set of breakpoints per histogram bin.

4. *At the destination* t: End search, backtrack using parent pointers to extract an s-t path.

A label ℓ is said to dominate another label ℓ' if $b_\ell(t, \tau) \geq b_{\ell'}(t, \tau)$ for all $t > 0$ and all $\tau > 0$.

If we end the search only when Q_G is empty, not simply when t is reached, we obtain the $\mathbb{E}$-feasible non-dominated paths. For $\mathbb{E}$-feasible probabilistic budget paths, we end the search when $\boldsymbol{T}_P(d) = 0$, i.e. the probability of reaching t within the time budget d drops to 0.

5 Charging Function Propagation for p-Feasible Routing

For p-feasible routing, we must consider the actual depletion distribution $\boldsymbol{D}_P$ for a path P, not simply $\mathbb{E}[\boldsymbol{D}_P]$, which sufficed for expected-feasible paths. As with expected-feasible paths, we must also deal with the travel time distribution $\boldsymbol{T}_P$. If path P has the edges $e_1, e_2, \ldots, e_n$, then $\boldsymbol{T}_P = \boldsymbol{T}_1 * \boldsymbol{T}_2 * \cdots * \boldsymbol{T}_n$ and $\boldsymbol{D}_P = \boldsymbol{D}_1 * \boldsymbol{D}_2 * \cdots * \boldsymbol{D}_n$. We can use p to place a bound on the maximum energy depletion we can accommodate over a path P. Let

$$\mathrm{c}_P(p) = \arg\max{}_x \{\boldsymbol{D}_P(x) \leq p\},$$

so that $\mathrm{c}_P(p)$ is the highest energy depletion that could occur along P with a probability of no more than p, that is, to ensure a non-stranding probability of p.

We define the *stochastic depletion function* analogously to Equation 2. Let $\text{S}_P(p)$ be the minimum starting SoC at s required to traverse P with non-stranding probability p. Similarly, let $\text{E}_P(p)$ be the maximum SoC possible on arriving at vertex t with at least probability p, and $\text{C}_P = \boldsymbol{D}_P$. The *stochastic depletion function* for P is

$$\sigma_P(\beta_s, p) = {}_t\beta = \begin{cases} -\infty, & \text{if } \beta_s < \text{S}_P(p), \\ \text{E}_P(p), & \text{if } \beta_s - \text{C}_P(p) > \text{E}_P(p), \\ \beta_s - \text{C}_P(p), & \text{otherwise.} \end{cases} \tag{3}$$

Let $\sigma_\emptyset$ be the *identity* stochastic depletion profile for a null path, so that $\sigma_\emptyset(\beta_s, p) = \beta_s$. If $P_1 = [v_i, v_{i+1}, \ldots, v_j]$ and $P_2 = [v_{j+1}, v_{j+2}, \ldots, v_k]$, the depletion profile of the concatenated path $P = P_1P_2 = [v_i \ldots v_k]$ is given by $\text{S}_P(p) = \max\{\text{S}_{P_1}(p), \text{C}_{P_1}(p) + \text{S}_{P_2}(p)\}$, $\text{E}_P(p) = \min\{\text{E}_{P_2}(p), \text{E}_{P_1}(p) - \text{C}_{P_2}(p)\}$ and $\text{C}_P = \text{C}_{P_1} * \text{C}_{P_2}$.

5.1 Dijkstra Search for p-feasible Routes

The label at vertex v is a four-tuple $\langle \boldsymbol{T}_{[s\ldots v]}, {}_c\beta, c, \sigma_{[c\ldots v]} \rangle$, where $\boldsymbol{T}_{[s\ldots v]}$ is the travel time distribution for the subpath $[s \ldots v]$, c is the last charging station enroute from s to v, ${}_c\beta$ is the arrival SoC at c, and $\sigma_{[c\ldots v]}$ is the stochastic depletion function of the subpath $[c \ldots v]$.

As for $\mathbb{E}$-feasible routes, we maintain a global priority queue Q_G storing the travel time distributions from s, and queues $L_u(v)$ and $L_s(v)$ to store the unsettled and settled labels at vertex v respectively. All queues use the usual stochastic ordering. On each search iteration, a label is extracted from $L_u(v)$, where v is the minimum-travel time vertex in Q_G, settled, and added to $L_s(v)$. Each label in $L_s(v)$ represents a feasible path from s to v, including the charging time at the last charging station. Each label in $L_u(v)$ represents a potentially feasible path whose feasibility is yet unverified. If a label $\ell \in L_u(v)$ is dominated by $\ell' \in L_s(v)$, we can prune the search along that path and discard ℓ, because a faster feasible path is already known. p-feasible queries have four parameters: the source vertex s, the destination vertex t, the β_s, and the given p. The search proceeds as follows:

1. *At s*: Mark s as a charging station. Add the label $\langle \boldsymbol{T}_\emptyset, {}_s\beta, s, \sigma_\emptyset \rangle$ to $L_u(s)$.
2. *At a non-charging vertex v*: Let $\ell = \langle \boldsymbol{T}_{[s\ldots v]}, {}_c\beta, c, \sigma_{[c\ldots v]} \rangle$ be the label extracted from $L_u(v)$. Since c is the last charging station encountered on the route represented by ℓ, add label $\langle \boldsymbol{T}_{[s\ldots v]}, \sigma_{[c\ldots v]}({}_c\beta, p), c, \sigma_{[c\ldots v]} \rangle$ to $L_u(v)$ and update the travel times for v in Q_G.
3. *At a charging vertex v*: Let $\ell = \langle \boldsymbol{T}_{[s\ldots v]}, {}_c\beta, c, \sigma_{[c\ldots v]} \rangle$ be the label extracted from $L_u(v)$. Let t_c be the charging time at the last charging station c, so $\beta_c = \Phi_c({}_c\beta, t_c)$. As with $\mathbb{E}$-feasible routes, the charging times corresponding to breakpoints of $\Phi_c(\cdot)$ suffice to make the required trade-off between charging and travel times. Let τ represent some value for travel time from s to v, and compute

$$b'_\ell(t_c, \tau, p) := \begin{cases} \sigma_{[c\ldots v]}(\beta_c, p) & \text{if } t_c > 0 \text{ and } \boldsymbol{T}_{[s\ldots v]}(\tau) > 0 \\ -\infty & \text{otherwise} \end{cases}$$

   Since $\Phi_c(\cdot)$ is piecewise linear, its breakpoints induce breakpoints for b_ℓ. Moreover, p is already known at query time, so for a given value of τ, we only need to create one label per breakpoint of b'_ℓ [8]. For a fixed τ and each breakpoint $B = (t_B, \text{SoC}_B)$ of b'_ℓ, we add to $L_u(v)$ the label $\langle t_B, \text{SoC}_B, v, \boldsymbol{T}_\emptyset \rangle$, and update the travel times to v in Q_G.

   As with $\mathbb{E}$-feasible routes, τ can take an infinite number of values, but we use histograms to represent $\boldsymbol{T}_e$, and we need to generate only one set of breakpoints per histogram bin. Lastly, we verify p-feasibility of path $[s \ldots v]$, by maintaining the running product of the non-stranding probabilities of all legs over this path. If this product falls below p, the path $[s \ldots v]$ is no longer p-feasible. The search is pruned and labels for v are dropped.

4. *At the destination t*: End search, backtrack using parent pointers to extract an s-t path. For a given p, a label ℓ dominates another label ℓ' if $b'_\ell(t,\tau,p) \geq b'_\ell(t,\tau,p)$ for all $t > 0$ and $\tau > 0$. If the search terminates only when Q_G is empty, the resulting s-t paths are the p-feasible Non-Dominated Paths. For Probabilistic Budget queries, we end the search only when it reaches far enough for the probability of reaching t within the time budget d is 0.

6 Stochastic Contraction Hierarchies

For deterministic queries, Contraction Hierarchies (CHs) [21] are widely used for speed up. Graph vertices are ranked, and *contracted* in ranked order. If u-v-w is a shortest path from u to w, vertex v is contracted by adding an edge u-w, and removing v from the graph. Such shortcuts significantly speed up the query-time Dijkstra search. The vertex ranks and the edge-weight hierarchy significantly affect preprocessing and query times [6]. A multicriteria CH variant is used in [55] for constrained shortest paths with positive weights. The CHArge algorithm [9, 8] combines a partial multicriteria CH with A* search. It contracts most graph vertices, creating a partial multicriteria CH but keeps an uncontracted *core* with charging stations. A* search using potential functions is used in the core to find routes at query time.

CHs have also been applied recently to stochastic route planning [42, 47]. However, we are interested in finding *feasible* routes that satisfy the energy bounds on EVs. Our queries are stochastic, and in fact doubly so. Travel time is always stochastic, and energy depletion is also stochastic for p-feasible queries. The stochastic dominance criterion is known to be too restrictive in practice [61], so it is hard to find dominating paths for most shortest paths in the network. Since the added shortcuts in CHs must not violate correctness, we can only avoid adding a shortcut covering a shortest path P only if we can find another witness path that dominates P [21, 55].

We solve this problem by relaxing our definition of dominance as follows. For distributions $\boldsymbol{T}_P$ and $\boldsymbol{D}_P$, we use the restricted-dominance criterion of [10], which checks if the CDF of one distribution is greater than that of the other within a fixed interval I, which we set to two standard deviations on each side of $\mathbb{E}[\boldsymbol{T}_P]$ or $\mathbb{E}[\boldsymbol{D}_P]$. For search labels, we use a definition of ϵ-dominance similar to that of [5, 45]. We say that a label ℓ_1 dominates another label ℓ_2 if all breakpoints of b_{ℓ_1} or b'_{ℓ_1} have SoC_B values within ϵ of b_{ℓ_2} or b'_{ℓ_2}. We set $\epsilon = 2\%$ of battery capacity in our experiments.

7 Experiments

Our algorithms were implemented in Rust 1.60.0-nightly with full optimizations and run on an Intel core i5-8600K processor with 3.6GHz base clock, 192KB of L1, 1.5 MB of L2, and 9 MB of L3 cache and equipped with 64GB of dual-channel 3200MHz DDR4 RAM.

7.1 Preparing a realistic routing instance

We extracted traffic speeds from Mapbox Traffic Data[1] for Tile 0230123,[2] between 15th July and 30th November, 2019. Tile 0230123 covers Los Angeles county between Long Beach and Oxnard, and yielded a graph with 559,271 vertices and 1,058,450 edges. The dataset contained speed updates for an edge subset at 5-minute intervals, which we aggregated

[1] https://www.mapbox.com/traffic-data
[2] https://labs.mapbox.com/what-the-tile

into weekday and weekend speed histograms. We discarded the weekend histograms due to sparsity, and used only the weekday speeds for our experiments. We added latitudes and longitudes for each vertex from the OSM dataset taken from GeoFabrik,[3] contracted the degree-2 vertices, and extracted the largest connected component. This step resulted in the final routing graph of 244,728 vertices and 453,942 edges.

We added elevation data from the NASADEM dataset [34] at 30M resolution to each vertex, using bilinear interpolation to estimate elevations at vertex locations. Lastly, we obtained charging stations from the Alternative Fuels Data Center,[4] marking the vertex closest to each charging station as the charging vertex. The charging function Φ_c on each vertex c was linear, and either (1) a *slow*, charging to 100% in 100 minutes, or (2) *fast*, charging up to 80% in 30 minutes, and up to 100% in 60 minutes. We randomly assigned the slow charging function to 70% of charging stations, the fast charging function to the rest.

Energy consumption parameters for ε_e on all edges e were derived using the vertex elevations and the values a_e, b_e, c_e, d_e used for Nissan Leaf 2013 in [16]. To force the search to require charging en route for feasibility, we assumed that the EV had a 12 kWh battery.

Choice of edge weight representation. Histograms capture arbitrary $\boldsymbol{T}_e$ and $\boldsymbol{D}_e$ distributions, but take more space. Functions may be less faithful to real-world distributions, but are compact and may lead to faster queries in some cases [47]. We used histograms to represent the travel time and energy consumption distributions on edges since our dataset had enough data for most edges. This allows us to represent arbitrary distributions while keeping the implementation simple.

Applying Contraction Hierarchies. Building a full CH by contracting all vertices of the graph can be prohibitively expensive due to the high cost of contracting the highest ranked vertices [9]. So, we build a only partial CH by contracting 97% of the vertices, keeping an uncontracted *core* containing all the charging stations on the network. Queries are run in three stages—from s to a vertex in the core restricted to using only (upward) edges from lower to higher ranked vertices, backward search from t to a vertex in the core using downward edges, and a simple bidirectional search within the core of the network.

7.2 Results

Using stochastic edge weights raises many challenges that do not arise for deterministic weights. Two obvious issues are maintaining route feasibility, and aggregating edge distributions $\boldsymbol{T}_e$ and $\boldsymbol{D}_e$ into path distributions $\boldsymbol{T}_P$ or $\boldsymbol{D}_P$, which requires expensive convolutions. Several other issues also arise, two of which we will discuss.

Number of histogram bins. The time and energy value ranges in the path distributions $\boldsymbol{T}_P, \boldsymbol{D}_P$ increases linearly with the number of edges in P, so more histogram bins are needed to maintain accuracy. As in the deterministic case, the Dijkstra search labels track the travel time-charging time tradeoff. The labels represent histograms, so the label sizes increase with the number bins used for $\boldsymbol{T}_P$ and $\boldsymbol{D}_P$. Labels become progressively larger for longer routes, raising the cost of all operations on the distributions, (convolution, dominance checks, etc.).

[3] `https://download.geofabrik.de/north-america/us/california/socal.html`
[4] `https://afdc.energy.gov/fuels/electricity_locations.html`

At charging stations, moreover, we must create a set of breakpoints per bin of the energy depletion histogram. More breakpoints are created for charging stations further along the route, increasing costs and making label dominance checks labels more difficult.

We also note that the CH shortcuts represent longer routes, whose histograms have more bins than the original graph edges. Shortcut edges are hence more expensive to handle than original graph edges, decreasing the utility of shortcuts in speeding up route planning queries.

Ensuring stochastic feasibility. Standard probabilistic budget routes use a single criterion, such as travel time [42, 47]. In contrast, our queries must handle search with two criteria to maintain feasibility. Further, the number of breakpoints in the charging functions along a route determines the number of labels generated.

For deterministic edge weights, path costs are just sums of edge costs, so routing takes just microseconds even on continent-sized road networks [4]. Routing with stochastic edge weights is far slower, since the convolutions needed to get path costs are very expensive. Prior work [42, 47] deals only with stochasticity in time, ignoring energy feasibility, but we consider both aspects. Our methods take tens of seconds, which is comparable to these prior methods. In preliminary experiments, our use of stochastic, multicriteria CH yielded a 2–2.4 factor speedup over queries not using CH. In deterministic settings, similar methods have been reported to achieve speedups of two to three orders of magnitude [21]. This lower gain can be attributed to the weaker "hierarchy" with stochastic edge weights, causing far more shortcuts to be added to the original graph. This forces the Dijkstra search to scan many more edges on settling each vertex.

Table 2 Single-criterion probabilistic budget routing queries [47] vs. our $\mathbb{E}$-feasible and p-feasible queries on the Tile 0230123 graph. Query times (seconds) are averages over 100 random vertex pairs. The EV is a Nissan Leaf 2013 with 12 kWh battery and 50% starting SoC.

d	Feasibility Ignored [47]	$\mathbb{E}$-feasible Routes	p-feasible Routes		
			$p = 0.8$	$p = 0.85$	$p = 0.9$
5 min.	6.192	10.662	12.993	11.99	10.31
15 min.	19.999	24.711	38.71	38.9	36.2
25 min.	45.384	38.123	75.05	73.8	71.34

Table 2 quantifies the overhead of maintaining feasibility of routes in stochastic settings, and compares the query times for single-criteria probabilistic budget routes (time only, feasibility ignored) with those of our two-criteria feasible probabilistic budget routes. Single-criteria routing is fastest, followed by $\mathbb{E}$-feasible routing, and p-feasible routing. The anomaly for $d = 25$ minutes can be understood as follows. Multicriteria search must explore a larger set of routes from the source than single-criteria queries, because it needs to return the pareto frontier of routes, rather than a single route. The $\mathbb{E}$-feasible and p-feasible queries must also carry and update per-vertex labels, and maintain more information in each label to capture the travel time-charging time tradeoffs. However, we use the restricted dominance criterion for $\mathbb{E}$-feasible and p-feasible routes but not for the single-criteria routes, making the cost per convolution is slightly lower for the two feasible-path queries. This suffices to make $\mathbb{E}$-feasible routing slightly faster for longer routes than even single-criterion queries.

Table 3 compares $\mathbb{E}$-feasible and p-feasible queries, for longer deadlines. $\mathbb{E}$-feasible queries are generally faster than p-feasible queries because they must convolve only $\boldsymbol{T}_P$, but p-feasible queries convolve both $\boldsymbol{T}_P$ and $\boldsymbol{D}_P$. p-feasible queries with higher p thresholds tend to run slightly faster, as they can prune the search quicker than searches run with lower p.

Table 3 $\mathbb{E}$-feasible and p-feasible query performance on the Tile 0230123 graph, with real-world charging station and elevation data. Query times (seconds) are over 500 random vertex pairs. EV used is a Nissan Leaf 2013 fitted with a 12 kWh battery and 50% starting SoC.

Query Type	**Feasibility Threshold**	**Time Budget (d)**			
		10 min.	**20 min.**	**30 min.**	**40 min.**
$\mathbb{E}$-feasible	—	18.01	34.975	48.662	72.198
p-feasible	$p = 0.8$	33.1	52.895	81.94	91.04
	$p = 0.85$	27.1	49.58	80.35	98.312
	$p = 0.9$	26.43	47.901	78.419	96.51

Table 4 Average Jaccard Index for 500 random $\mathbb{E}$-feasible and p-feasible routes, with $p = 0.85$. The index is 0 when the routes are edge-disjoint, and 1 when they are identical.

Queries Compared	d	**Avg. Jaccard Index**
$\mathbb{E}$-feasible and p-feasible, for $p = 0.85$	10 min.	0.73
	20 min.	0.74
	30 min.	0.87
	40 min.	0.94

Table 4 shows how similar the $\mathbb{E}$-feasible and p-feasible routes are, using the average Jaccard Similarity between the set edges of a route chosen by each of them. The Jaccard similarity for two routes P_1 and P_2 is the number of edges common to both divided by the number of edges in their union. That is,

$$J(P_1, P_2) = \frac{|\{e \in P_1\} \cap \{e \in P_2\}|}{|\{e \in P_1\} \cup \{e \in P_2\}|}$$

The Jaccard index clearly increases with the time budget, so the $\mathbb{E}$-feasible and p-feasible routes are more similar when the routes are longer. This is because longer routes require more convolutions, making $\boldsymbol{D}_P$ closer to the Gaussian, which is more concentrated near its mean. In such cases, the pruning of edges forced by the feasibility criterion brings the set of edges of $\mathbb{E}$-feasible routes closer to the set of edges for p-feasible routing. For shorter routes, however, the difference between the two types of queries is higher. Hence, if stronger feasibility guarantees are desired for shorter routes, p-feasible queries may be better.

8 Conclusion and Future Work

EV routing methods usually model the problem as a generalized constrained shortest-path problem, with deterministic travel times and energy consumptions. This is unrealistic since these are really stochastic parameters. Current stochastic route planning methods either fail to ensure that routes are energy-feasible, or when they do, have not been shown to scale well to large graphs. In this work, we address this shortcoming by making travel time and energy consumption stochastic, and requiring paths to be energy-feasible. We defined two energy-feasibility criteria, namely, $\mathbb{E}$-feasibility and p-feasibility. We showed how to generalize the standard Charging Function Propagation algorithm of [8, 9] to accept stochastic edge weights, while allowing recharging stations. We also applied a multicriteria variant of stochastic Contraction Hierarchies to speed up our queries, using the restricted stochastic dominance criterion of [10] and the ϵ-dominance among labels. We demonstrated that our techniques were feasible in the real world by running experiments on a realistic routing instance, using real-world travel speeds in the Los Angeles area collected over four and a half months. The similarity between $\mathbb{E}$-feasible and p-feasible routes indicates the potential applicability of a tiered-hierarchy style approach [47] to help speed up stochastic feasible routing queries even further, and could be an interesting avenue for further work.

References

1. Niklas Akerblöm, Yuxin Chen, and Morteza Haghir Chehreghani. An online learning framework for Energy-Efficient navigation of electric vehicles. In *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence (IJCAI-20)*, pages 2051–2057, 2020.
2. Yazan Al-Wreikat, Clara Serrano, and José Ricardo Sodré. Driving behaviour and trip condition effects on the energy consumption of an electric vehicle under real-world driving. *Appl. Energy*, 297:117096, September 2021.
3. Andreas Artmeier, Julian Haselmayr, Martin Leucker, and Martin Sachenbacher. The shortest path problem revisited: Optimal routing for electric vehicles. In *KI 2010: Advances in Artificial Intelligence*, Lecture Notes in Computer Science, pages 309–316. Springer, Berlin, Heidelberg, September 2010.
4. Hannah Bast, Daniel Delling, Andrew Goldberg, Matthias Müller-Hannemann, Thomas Pajor, Peter Sanders, Dorothea Wagner, and Renato F Werneck. Route planning in transportation networks. In *Algorithm Engineering*, Lecture Notes in Computer Science, pages 19–80. Springer, Cham, 2016.
5. Lucas S Batista, Felipe Campelo, Frederico G Guimarães, and Jaime A Ramírez. A comparison of dominance criteria in many-objective optimization problems. In *2011 IEEE Congress of Evolutionary Computation (CEC)*, pages 2359–2366, June 2011.
6. Reinhard Bauer, Tobias Columbus, Ignaz Rutter, and Dorothea Wagner. Search-space size in contraction hierarchies. *Theor. Comput. Sci.*, 645:112–127, September 2016.
7. Moritz Baum. *Engineering Route Planning Algorithms for Battery Electric Vehicles*. PhD thesis, Karlsruhe Institute of Technology, 2018.
8. Moritz Baum, Julian Dibbelt, Andreas Gemsa, Dorothea Wagner, and Tobias Zündorf. Shortest feasible paths with charging stops for battery electric vehicles. In *Proceedings of the 23rd SIGSPATIAL International Conference on Advances in Geographic Information Systems*, page 44. ACM, November 2015.
9. Moritz Baum, Julian Dibbelt, Andreas Gemsa, Dorothea Wagner, and Tobias Zündorf. Shortest feasible paths with charging stops for battery electric vehicles. *Transportation Science*, 53(6):1627–1655, November 2019.
10. Roger L Berger. A nonparametric, intersection-union test for stochastic order. Technical report, North Carolina State University. Dept. of Statistics, 1986.
11. Dimitri P Bertsekas and John N Tsitsiklis. An analysis of stochastic shortest path problems. *Math. Oper. Res.*, 16(3):580–595, August 1991.
12. Anthony Chen and Zhaowang Ji. Path finding under uncertainty. *J. Adv. Transp.*, 39(1):19–37, September 2005.
13. Xiao-Wei Chen, Bi Yu Chen, William H K Lam, Mei Lam Tam, and Wei Ma. A bi-objective reliable path-finding algorithm for battery electric vehicle routing. *Expert Syst. Appl.*, 182:115228, November 2021.
14. Jochen Eisner, Stefan Funke, and Sabine Storandt. Optimal route planning for electric vehicles in large networks. In *AAAI*, 2011.
15. Y Y Fan, R E Kalaba, and J E Moore. Arriving on time. *J. Optim. Theory Appl.*, 127(3):497–513, December 2005.

16 Chiara Fiori, Kyoungho Ahn, and Hesham A Rakha. Power-based electric vehicle energy consumption model: Model development and validation. *Appl. Energy*, 168:257–268, April 2016.

17 Chiara Fiori, Vittorio Marzano, Vincenzo Punzo, and Marcello Montanino. Energy consumption modeling in presence of uncertainty. *IEEE Trans. Intell. Transp. Syst.*, pages 1–12, 2020.

18 Matthew William Fontana. *Optimal routes for electric vehicles facing uncertainty, congestion, and energy constraints*. PhD thesis, Massachusetts Institute of Technology, 2013.

19 Mogens Fosgerau. The valuation of travel time variability. Technical report, International Transport Forum, 2016.

20 H Frank. Shortest paths in probabilistic graphs. *Oper. Res.*, 17(4):583–599, 1969.

21 Robert Geisberger, Peter Sanders, Dominik Schultes, and Daniel Delling. Contraction hierarchies: Faster and simpler hierarchical routing in road networks. In *Experimental Algorithms*, pages 319–333. Springer, Berlin, Heidelberg, May 2008.

22 Michael T Goodrich and Paweł Pszona. Two-phase bicriterion search for finding fast and efficient electric vehicle routes. In *Proceedings of the 22nd ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems*, pages 193–202. ACM, November 2014.

23 Ming Hua and Jian Pei. Probabilistic path queries in road networks: traffic uncertainty aware path selection. In *Proceedings of the 13th International Conference on Extending Database Technology*, EDBT '10, pages 347–358, New York, NY, USA, March 2010. Association for Computing Machinery.

24 Gerhard Huber, Klaus Bogenberger, and Hans van Lint. Optimization of charging strategies for battery electric vehicles under uncertainty. *IEEE Trans. Intell. Transp. Syst.*, pages 1–17, 2020.

25 Ehsan Jafari and Stephen D Boyles. Multicriteria stochastic shortest path problem for electric vehicles. *Networks Spat. Econ.*, 17(3):1043–1070, September 2017.

26 H C Joksch. The shortest route problem with constraints. *J. Math. Anal. Appl.*, 14(2):191–197, May 1966.

27 Moritz Kobitzsch, Samitha Samaranayake, and Dennis Schieferdecker. Pruning techniques for the stochastic on-time arrival problem- an experimental study. *arXiv*, July 2014. `arXiv:1407.8295`.

28 Sejoon Lim, Christian Sommer, Evdokia Nikolova, and Daniela Rus. Practical route planning under delay uncertainty: Stochastic shortest path queries. In *Robotics: Science and Systems*, volume 8, pages 249–256. books.google.com, 2013.

29 Antonio Lima, Rade Stanojevic, Dina Papagiannaki, Pablo Rodriguez, and Marta C González. Understanding individual routing behaviour. *J. R. Soc. Interface*, 13(116), March 2016.

30 Sören Merting, Christian Schwan, and Martin Strehler. Routing of electric vehicles: Constrained shortest path problems with resource recovering nodes. In *OASIcs-OpenAccess Series in Informatics*, volume 48. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik GmbH, Wadern/Saarbruecken, Germany, 2015.

31 Sebastiano Milardo, Punit Rathore, Marco Amorim, Umberto Fugiglando, Paolo Santi, and Carlo Ratti. Understanding drivers' stress and interactions with vehicle systems through naturalistic data analysis. *IEEE Trans. Intell. Transp. Syst.*, pages 1–12, 2021.

32 Elise D Miller-Hooks and Hani S Mahmassani. Least expected time paths in stochastic, Time-Varying transportation networks. *Transportation Science*, 34(2):198–215, May 2000.

33 Elise Deborah Miller-Hooks. *Optimal routing in time-varying, stochastic networks: Algorithms and implementations*. PhD thesis, The University of Texas at Austin, 1997.

34 J P L Nasa. NASADEM merged DEM global 1 arc second V001 [dataset], 2020. Accessed: 2021-6-9. `doi:10.5067/MEaSUREs/NASADEM/NASADEM_HGT.001`.

35 Yu (marco) Nie and Xing Wu. Shortest path problem considering on-time arrival probability. *Trans. Res. Part B: Methodol.*, 43(6):597–613, July 2009.

36 Patrick Niklaus. A unified framework for electric vehicle routing. Master's thesis, Karlsruhe Institute of Technology, 2017.

37 Mehrdad Niknami and Samitha Samaranayake. Tractable pathfinding for the stochastic On-Time arrival problem. In *Experimental Algorithms*, pages 231–245. Springer International Publishing, 2016.

38 Evdokia Nikolova. Approximation algorithms for reliable stochastic combinatorial optimization. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques*, pages 338–351. Springer Berlin Heidelberg, 2010.

39 Evdokia Nikolova, Matthew Brand, and David R Karger. Optimal route planning under uncertainty. *ICAPS*, 2006.

40 Evdokia Nikolova, Jonathan A Kelner, Matthew Brand, and Michael Mitzenmacher. Stochastic shortest paths via quasi-convex maximization. In *Algorithms – ESA 2006*, pages 552–563. Springer Berlin Heidelberg, 2006.

41 Axel Parmentier and Frédéric Meunier. Stochastic shortest paths and risk measures. *arXiv*, August 2014. `arXiv:1408.0272`.

42 Simon Aagaard Pedersen, Bin Yang, and Christian S Jensen. Fast stochastic routing under time-varying uncertainty. *VLDB J.*, October 2019.

43 Simon Aagaard Pedersen, Bin Yang, and Christian S Jensen. Anytime stochastic routing with hybrid learning. *Proceedings VLDB Endowment*, 13(9):1555–1567, May 2020.

44 Simon Aagaard Pedersen, Bin Yang, and Christian S Jensen. A hybrid learning approach to stochastic routing. In *2020 IEEE 36th International Conference on Data Engineering (ICDE)*, pages 1910–1913, April 2020.

45 Payas Rajan, Moritz Baum, Michael Wegner, Tobias Zündorf, Christian J West, Dennis Schieferdecker, and Daniel Delling. Robustness generalizations of the shortest feasible path problem for electric vehicles. In Matthias And Federico, Müller-Hannemann, editor, *Proceedings of 21st Symposium on Algorithmic Approaches for Transportation Modelling, Optimization, and Systems (ATMOS 2021)*, Open Access Series in Informatics (OASIcs), pages 11:1–11:18, Dagstuhl, Germany, September 2021. Schloss Dagstuhl - Leibniz-Zentrum für Informatik.

46 Payas Rajan and Chinya V Ravishankar. The phase abstraction for estimating energy consumption and travel times for electric vehicle route planning. In *Proceedings of the 27th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems*, SIGSPATIAL '19, pages 556–559, New York, NY, USA, 2019. ACM.

47 Payas Rajan and Chinya V Ravishankar. Tiering in contraction and edge hierarchies for stochastic route planning. In *Proceedings of the 29th International Conference on Advances in Geographic Information Systems*, SIGSPATIAL '21, pages 616–625, New York, NY, USA, November 2021. Association for Computing Machinery.

48 Guillaume Sabran, Samitha Samaranayake, and Alexandre Bayen. Precomputation techniques for the stochastic on-time arrival problem. In *2014 Proceedings of the Meeting on Algorithm Engineering and Experiments (ALENEX)*, Proceedings, pages 138–146. Society for Industrial and Applied Mathematics, December 2013.

49 M Sachenbacher, M Leucker, A Artmeier, and J Haselmayr. Efficient Energy-Optimal routing for electric vehicles. *AAAI*, 2011.

50 Moshe Shaked and J George Shanthikumar. *Stochastic Orders*. Springer New York, October 2006.

51 Liang Shen, Hu Shao, Ting Wu, William H K Lam, and Emily C Zhu. An energy-efficient reliable path finding algorithm for stochastic road networks with electric vehicles. *Transp. Res. Part C: Emerg. Technol.*, 102:450–473, May 2019.

52 Kenneth A Small, Clifford Winston, and Jia Yan. Uncovering the distribution of motorists' preferences for travel time and reliability. *Econometrica*, 73(4):1367–1382, July 2005.

53 Christian Sommer. Shortest-path queries in static networks. *ACM Computing Surveys (CSUR)*, 46(4):45, April 2014.

54 Sabine Storandt. Quick and energy-efficient routes: Computing constrained shortest paths for electric vehicles. In *Proceedings of the 5th ACM SIGSPATIAL International Workshop on Computational Transportation Science*, IWCTS '12, pages 20–25, New York, NY, USA, 2012. ACM.

55 Sabine Storandt. Route planning for bicycles – exact constrained shortest paths made practical via contraction hierarchy. In *Proceedings of the Twenty-Second International Conference on Automated Planning and Scheduling*, 2012.

56 Sabine Storandt and Stefan Funke. Cruising with a Battery-Powered vehicle and not getting stranded. In *AAAI*, volume 3, page 46, 2012.

57 Martin Strehler, Sören Merting, and Christian Schwan. Energy-efficient shortest routes for electric and hybrid vehicles. *Trans. Res. Part B: Methodol.*, 103(Supplement C):111–135, September 2017.

58 Christoph Witzgall and Alan J Goldman. Most profitable routing before maintenance. In *OPERATIONS RESEARCH*, page B82. INST OPERATIONS RESEARCH MANAGEMENT SCIENCES 901 ELKRIDGE LANDING RD, STE 400, LINTHICUM HTS, MD, USA, 1965.

59 Bin Yang, Chenjuan Guo, Christian S Jensen, Manohar Kaul, and Shuo Shang. Stochastic skyline route planning under time-varying uncertainty. In *2014 IEEE 30th International Conference on Data Engineering*, pages 136–147, March 2014.

60 Shanjiang Zhu and David Levinson. Do people use the shortest path? an empirical test of wardrop's first principle. *PLoS One*, 10(8):e0134322, August 2015.

61 Weiwei Zhuang, Yadong Li, and Guoxin Qiu. Statistical inference for a relaxation index of stochastic dominance under density ratio model. *J. Appl. Stat.*, pages 1–19, August 2021.