8-2021

GP3: Gaussian process path planning for reliable shortest path in transportation networks

Hongliang GUO

Xuejie HOU

Zhiguang CAO
Singapore Management University, zgcao@smu.edu.sg

Jie ZHANG

Citation

GUO, Hongliang; HOU, Xuejie; CAO, Zhiguang; and ZHANG, Jie. GP3: Gaussian process path planning for reliable shortest path in transportation networks. (2021). *IEEE Transactions on Intelligent Transportation Systems*. 23, (8), 11575-11590.
Available at: https://ink.library.smu.edu.sg/sis_research/8126

GP3: Gaussian Process Path Planning for Reliable Shortest Path in Transportation Networks

Article *in* IEEE Transactions on Intelligent Transportation Systems · August 2022

DOI: 10.1109/TITS.2021.3105415

Hongliang Guo
Agency for Science, Technology and Research (A*STAR)

Zhiguang Cao
Singapore Management University

GP3: Gaussian Process Path Planning for Reliable Shortest Path in Transportation Networks

Hongliang Guo, Xuejie Hou, Zhiguang Cao, and Jie Zhang

Abstract—**This paper investigates the reliable shortest path (RSP) problem in Gaussian process (GP) regulated transportation networks. Specifically, the RSP problem that we are targeting at is to minimize the (weighted) linear combination of mean and standard deviation of the path's travel time. With the reasonable assumption that the travel times of the underlying transportation network follow a multi-variate Gaussian distribution, we propose a Gaussian process path planning (GP3) algorithm to calculate the a priori optimal path as the RSP solution. With a series of equivalent RSP problem transformations, we are able to reach a polynomial time complexity algorithm with guaranted solution accuracy. Extensive experimental results over various sizes of realistic transportation networks demonstrate the superior performance of GP3 over the state-of-the-art algorithms.**

Index Terms—**Reliable shortest path (RSP), mean-std minimization, Gaussian process path planning (GP3), a priori path, stochastic on time arrival (SOTA), Lagrangian relaxation.**

I. Introduction

STANDARD shortest path problem, which aims at finding a path with the minimum travel distance, time or other forms of cost from origin to destination, has long been solved optimally via efficient algorithms [1], and applied to large scale road networks [2]. However, those algorithms typically assume a deterministic and static underlying transportation network, and make use of the additive nature of the objective for label-correcting and/or dynamic programming solutions. The assumption is invalid when facing real world applications, in which travel times are essentially stochastic due to demand fluctuation and supply degradation [3], [4].

On the other hand, stochastic shortest path (SSP) explicitly targets at the problem of path planning under uncertainties and offers travellers with a wide range of objective selections depending on his/her attitude towards the risk of travel time variability. Perhaps one of the most common SSP objectives is the least expected time (LET) path, which aims at minimizing the expected travel time of the planned path. LET path is suitable for risk-neutral travellers, who do not care about the variation of the planned path. However, a large portion of the route travellers are risk averse, *e.g.* they would rather travel a longer time in expectation during navigation, in trade of a smaller variance. This kind of travellers need a reliable shortest path (RSP) which guarantees, with a high probability, that the actual arrival time will not be too late.

There already exist several RSP objectives in the path planning community, which finds the path with (1) minimal expected disutility (MED) [5], [6]; (2) maximal stochastic on-time arrival (SOTA) probability [7]–[12]; (3) minimal α-reliable travel time [13]–[15], or (4) minimal mean-variance combination [16], [17]; minimal mean-standard deviation combination (mean-std) [18]–[21]. A brief literature review over RSP-related research including the objective, travel time assumptions and the methodologies will be performed in Section II. This paper selects the mean-std minimization as GP3's RSP planning objective, and it is worth noting that in GP-regulated environments, the minimal mean-std RSP can be equivalently transformed to the α-reliable RSP as well as the SOTA RSP, which will be proved in the Appendix.

The minimal mean-std RSP problem is essentially a non-linear integer programming problem, and there are no off-the-shelf solvers guaranteed to find the *global optimal solution within polynomial time*. Current prevailing techniques either use the branch and bound (or branch and cut) algorithm to find the global optimal solution at the cost of non-polynomial computational complexity [4], or use Lagrangian multipliers to convert some of the constraints into the objective, and iteratively approximate the solution with polynomial-complexity solvers for the converted problem. However, the relaxed Lagrangian multiplier method cannot guarantee a global optimal solution, *i.e.* the duality gap is not guaranteed to converge to zero, which means that we do not know whether or not the converged solution is the global optima. Moreover, within each iteration, the duality gap shrinkage size cannot be approximated/estimated theoretically, which means that we do not know, beforehand, how many rounds of iterations is needed for the ultimate solution to converge.

This paper proposes a Gaussian process path planning (GP3) algorithm which (1) performs a series of equivalent transformations to convert the original minimal mean-std RSP problem into a (convex) quadratic integer programming (QIP) problem; (2) uses the max-flow problem solution to enumerate

Manuscript received 13 July 2020; revised 5 May 2021; accepted 13 August 2021. Date of publication 25 August 2021; date of current version 9 August 2022. This work was supported in part by the National Natural Science Foundation of China under Grant 61803104. The Associate Editor for this article was X. Ban. *(Corresponding author: Zhiguang Cao.)*

Hongliang Guo is with the Institute for Infocomm Research, A*STAR, Singapore 138632 (e-mail: guo_hongliang@i2r.a-star.edu.sg).

Xuejie Hou is with the School of Automation Engineering, University of Electronic Science and Technology of China, Chengdu 611731, China (e-mail: 201821060832@std.uestc.edu.cn).

Zhiguang Cao is with Singapore Institute of Manufacturing Technology (SIMTech), Singapore 138634 (e-mail: zhiguangcao@outlook.com).

Jie Zhang is with the School of Computer Science and Engineering, Nanyang Technological University, Singapore 639798 (e-mail: zhangj@ntu.edu.sg).

Digital Object Identifier 10.1109/TITS.2021.3105415

all the elementary paths as the candidates of the transformed QIP problem solution, (3) and then selects the 'best elementary path' as the optimal solution to the QIP problem. Theoretical proof that the 'best elementary path' is the optimal path of the QIP problem, is provided. In this way, GP3 algorithm is able to reach the global optimal solution of the minimal mean-std RSP problem with arbitrarily high accuracy within polynomial computation time.

The contributions of the paper can be summarized as follows: (1) we propose GP3, which enjoys polynomial computational complexity and in the meantime, is able to return the global optimal solution to the minimal mean-std problem; (2) we prove that in GP-regulated environments, the minimal mean-std RSP problem, SOTA RSP problem and the α-reliable shortest path problem can be equivalently transformed to each other with simple meta-parameter adjustments; (3) GP3 is tested in various realistic transportation networks with real traffic data, and the related performance is better than that of the state-of-the-art algorithms.

The remainder of the paper is organized as follows: we perform a brief literature review over related RSP research in Section II, followed by a brief introduction of Gaussian process basics in Section III. Then we present the minimal mean-std path planning problem formulation and several assumptions in Section IV, followed by the GP3 methodology introduction and its computational complexity analysis in Section V. Extensive experiments over various transportation networks are executed and the performance comparisons are presented in Section VI, and the paper ends with conclusions and future work directions in Section VII.

II. Literature Review

Reliable shortest path (RSP) has been widely recognized as an important research topic to deal with travel time variability in the path planning community, and this section provides a brief review over various RSP objectives and the corresponding solution approaches. Generally speaking, RSP planning algorithms assume either independent travel time distributions [20], [22]–[25] across different road segments, or spatially correlated travel times [15], [18], [21], [26], [27] within the underlying transportation networks. The spatially correlated travel time assumption is more complex but more realistic, *e.g*. traffic accidents on a certain link will probably lead to high travel times over its downstream and upstream links.

This paper presumes that the travel time over the entire transportation network is correlated, and can be expressed by a multi-variate Gaussian distribution, *i.e*. GP, which is the same assumption used in [4], [16]. Empirical studies based on real-world traffic data show that the use of multi-variate normal distribution, *i.e*. GP, reflects observed path travel time distributions very well [28]. It is also reported in [9] that GP approximation achieves 98.3% and 94.9% of accuracy at the 10^{th} and 90^{th} percentiles of the sampled real world travel time data. Moreover, GP is both flexible to capture the spatial correlations of travel time over the underlying transportation network, and convenient to derive posterior distributions when given only a subset of samples.

The essence/objective of an RSP is to find a path with one of the following four objectives, namely (1) minimal expected disutility (MED); (2) minimal high percentile travel time (α-reliable shortest path); (3) maximal stochastic on time arrival probability (SOTA); and (4) minimal linear combination of the path's expected travel time and standard deviation (mean-std) or variation (mean-variation). While this paper provides a GP path planning algorithm with the minimal mean-std objective, a brief review of the literature over the first three categories is also provided.

The MED RSP introduces a (usually non-linear) disutility function over the path's arrival time, representing the user's risk aversion attitude, and seeks for a path with the minimal expected disutility score. The exponential disutility function is first proposed and solved in [29] for stationary and independent travel time distributions, and the authors expand the application scenario to time-dependent and correlated travel time in [6].

The α-reliable RSP, as defined in [14], aims at finding a path with the minimal travel time with high probability guarantee, *i.e.* find a path with the minimal scalar value T, of which the probability that the path's travel time is smaller than T is higher than a predefined threshold value (α). When $\alpha = 0.5$, the α-reliable shortest path is equivalent to the LET path. The authors in [14] propose to use genetic algorithms to find the α-reliable shortest path for the independent travel time use case, and the case of α-reliable RSP planning for spatially correlated travel time is solved in [18].

The SOTA criterion for RSP, first proposed by Frank in [30], aims at finding a path with the maximal probability of arriving at a destination before a user-specified deadline. Fan *et al.* [23] study the SOTA RSP problem from the perspective of Bellman's principle of optimality, and the approach uses the dynamic programming procedure to iteratively solve the SOTA RSP problem, and outputs a routing policy instead of a priori path so as to maximize the on-time arrival probability. Since then, various extensions with the backbone of using dynamic programming and Bellman's principle of optimality have been proposed for SOTA RSP solutions [31], [32].

The mean-variance or mean-std RSP incorporates the path's travel time variance or standard deviation directly into the formulation, either as a constraint [33] or as an additional term in the objective function [16], [26], [34], [35]. The authors in [16] propose a polynomial time algorithm to solve the mean-variance RSP problem, in which the optimal path is proved to be within a set of pre-calculated set of paths, named as component paths. The algorithms for mean-std RSP can be roughly categorized into three groups: (1) formulate the RSP problem into a mixed integer non-linear programming (MINLP) problem and use off-the-shelf optimization solvers for the solution [35]–[37][1]; (2) transform the problem into a series of easily solvable problems for iterative solutions [20]; (3) apply Lagrangian relaxation and duality theory for approximate solution with duality gap analysis [15], [21], [26], [27].

[1] Note that some MINLP approaches also make use of (partial) Lagrangian relaxation for approximate solutions.

The mainstream approaches for the polynomial complexity RSP solutions are through Lagrangian relaxation, *e.g.* [21], [26], [38], [39]. Xing and Zhou propose a sample-based Lagrangian relaxation method for mean-std RSP planning, in which they approximate the overall path travel time as a normal distribution, and use the sampled travel time data to gauge the distribution parameters [26]. The proposed method is applicable to the transportation network with or without travel time correlations. Zeng *et al.* use Cholesky decomposition to decompose the covariance matrix and rely on partial Lagrangian method to solve the partially unified RSP planning problem [15]. The master problem can be decomposed to several standard shortest path problems, and Dijkstra's algorithm is used to solve the master problem. The dual variable is iteratively updated through subgradient method. During the iteration process, the original problem's lower bound and upper bound are updated and the convergence of the duality gap is provided. Similar works are proposed in [21], [27], respectively, with different techniques for the covariance matrix decomposition.

Lagrangian relaxation method is fast and can be proved for convergence. However, there are two main drawbacks. Firstly, for each iteration, the shrinkage size of the duality gap cannot be estimated beforehand, which means that we do not know how much improvement we can get after one iteration. Secondly, the Lagrangian relaxation method cannot guarantee a final zero duality gap, which means that even with infinite amount of time, the algorithm cannot guarantee to converge to the ultimate optimal solution.

This paper proposes a polynomial computational complexity algorithm, named as GP3, which guarantees that (1) after each iteration, the gap (difference between upper bound and lower bound) will shrink by at least a half, and (2) the gap converges to zero in the limit. It is worth noting that, in GP-regulated environments, the mean-std RSP problem has a one-to-one mapping relationship with the α-reliable RSP problem, and the SOTA RSP problem can be solved by iteratively finding an updated α-reliable RSP.

III. Gaussian Process Introduction

Since this paper presumes that the travel times of the underlying transportation networks follow a Gaussian process (GP), this section briefly introduces the basics of GP, and more detailed contents about GP can be referred in [40]. A Gaussian process is a stochastic process (a collection of random variables indexed by time or space), such that every finite collection of those random variables has a multivariate normal distribution, *i.e.* every finite linear combination of them is normally distributed. Assume $X \in \mathcal{R}^n$ is a GP, *i.e.* $X \sim \mathrm{GP}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$, the probability density function (pdf) of X can be expressed as:

$$p(\boldsymbol{x}) = \frac{1}{\sqrt{|2\pi\boldsymbol{\Sigma}|}}\exp(-\frac{1}{2}(\boldsymbol{x}-\boldsymbol{\mu})^\top\boldsymbol{\Sigma}(\boldsymbol{x}-\boldsymbol{\mu})), \tag{1}$$

where $\boldsymbol{\mu}$ and $\boldsymbol{\Sigma}$ are the mean and covariance of $\boldsymbol{X}$. One of the GP characteristics is that the conditional distribution a GP is still a GP. Formally, it is stated as follows. When X is partitioned into two subsets: $X = \begin{bmatrix} X_1 \\ X_2 \end{bmatrix}$, with $X_1 \in \mathcal{R}^p$ and $X_2 \in \mathcal{R}^{(n-p)}$. Correspondingly, $\boldsymbol{\mu}$ and $\boldsymbol{\Sigma}$ are partitioned as $\boldsymbol{\mu} = \begin{bmatrix} \mu_1 \\ \mu_2 \end{bmatrix}$ and $\boldsymbol{\Sigma} = \begin{bmatrix} \boldsymbol{\Sigma}_{11} & \boldsymbol{\Sigma}_{12} \\ \boldsymbol{\Sigma}_{21} & \boldsymbol{\Sigma}_{22} \end{bmatrix}$ respectively. Then the distribution of X_1 conditional on $X_2 = \boldsymbol{x_2}$ is multivariate normal $X_{1|X_2=x_2} \sim \mathcal{N}(\boldsymbol{\mu}_{1|2}, \boldsymbol{\Sigma}_{1|2})$ where

$$\boldsymbol{\mu}_{1|2} = \boldsymbol{\mu}_1 + \boldsymbol{\Sigma}_{12}\boldsymbol{\Sigma}_{22}^{-1}(\boldsymbol{x_2} - \boldsymbol{u_2}), \tag{2}$$

and the conditional co-variance matrix

$$\boldsymbol{\Sigma}_{1|2} = \boldsymbol{\Sigma}_{11} - \boldsymbol{\Sigma}_{12}\boldsymbol{\Sigma}_{22}^{-1}\boldsymbol{\Sigma}_{21}. \tag{3}$$

This characteristic of GP gives us the capability and flexibility of describing the whole transportation network's travel time distribution with only a subset of samples.

IV. Problem Formulation

This section introduces the notations used throughout the paper, followed by the minimal mean-std RSP problem statement. Then, we lay down two reasonably assumptions used in the paper, which 'naturally'[2] avoid the cycle-containing RSP solutions.

A. Notations

Let $\mathcal{G}(\mathcal{N}, \mathcal{A})$ represent a directed and connected transportation network, where $\mathcal{N}$ ($|\mathcal{N}| = n$) refers to the set of nodes and $\mathcal{A}$ ($|\mathcal{A}| = m$) refers to the set of links. $\mathbf{c} \in \mathcal{R}^m$ is a random variable (RV) vector, representing the joint travel time distribution over the entire transportation network $\mathcal{G}$. This paper presumes that $\mathbf{c} \sim \mathcal{N}(\mu, \Sigma)$, where $\mu \in \mathcal{R}^m$ is the mean of $\mathbf{c}$, and $\Sigma \in \mathcal{S}_{++}^{m\times m}$ refers to the positive definite covariance matrix capturing the travel time's spatial correlation relationships. Here, we wish to articulate that GP3 targets at a stationary travel-time environment, where the mean and co-variance matrix of travel-time distribution are independent of the link flows and can be estimated statistically with real travel-time samples.

Other related variable notations are listed as follows: (1) $i, j \in \mathcal{N}$ refers to the node index, with r indicating the origin node and s indicating the destination node; (2) $ij, kl \in \mathcal{A}$ refers to link index; (3) $x_{ij} \in \{0, 1\}$ is a binary decision variable, when $x_{ij} = 1$, it means that link ij is selected as a part of the RSP; (4) σ_{ij} is the standard deviation of link ij's travel time, $\mathrm{Cov}(ij, kl)$ refers to the covariance of travel times between link ij and link kl (both σ_{ij}^2 and $\mathrm{Cov}(ij, kl)$ are elements of the co-variance matrix Σ); and (5) $\zeta \geq 0$ is the reliability coefficient representing the user's risk aversion attitude.

B. Problem Statement

The output of our algorithm is an a prior path connecting the origin node r to the destination node s, with the minimal

[2] Here, the word 'naturally' means that the optimization algorithm will automatically search for the path without cycles (no need for the explicit cycle-elimination constraint).

mean-std objective. The minimal mean-std RSP problem can be mathematically formulated as:

(P):

$$\min_{\boldsymbol{x}} \boldsymbol{\mu}^\top \boldsymbol{x} + \zeta\sqrt{\boldsymbol{x}^\top \Sigma \boldsymbol{x}}$$

$$\text{s.t.} \sum_{j:ij\in\mathcal{A}} x_{ij} - \sum_{k:ki\in\mathcal{A}} x_{ki} = \begin{cases} 1, & i = r \\ -1, & i = s \\ 0, & i \in \mathcal{N} - \{r, s\} \end{cases}$$

$$x_{ij} \in \{0, 1\}.$$

Note that in (P), the constraints are flow balancing constraints with Boolean decision variables (x_{ij}). For representation simplicity, the flow balancing constraints in (P) is usually succinctly stated as $\boldsymbol{A}\boldsymbol{x} = \boldsymbol{b}$, where $\boldsymbol{A} \in \mathcal{R}^{n\times m}$ is the node-incidence Matrix [41], with its element $a_{ij} \in \{1, -1, 0\}$. The node-incidence matrix ($\boldsymbol{A}$) represents the topological structure of the underlying transportation network. Its element a_{ij} represents the relationship between link j and node i, thus $1 \le j \le m$, and $1 \le i \le n$. When $a_{ij} = 1$, it means that link j starts at node i; when $a_{ij} = -1$, it means that link j ends at node i, and when $a_{ij} = 0$, it means that there is no relationship between link j and node i. $\boldsymbol{b} \in \{1, 0, -1\}^n$, is a column vector representing the origin-destination (OD) information, with $b_i = 1$ representing the origin node, and $b_i = -1$ representing the destination node, and $b_i = 0$ representing the usual node. Note that with the node-incidence matrix representation, the variable notations are slightly overloaded. Normally, a link should be labeled as link ij, with i indicating the link's starting node, and j indicating the link's ending node. However, when we are representing the underlying transportation network with node-incidence matrix ($\boldsymbol{A}$), node is labeled as i, and link is labeled as j, which means that when its element $a_{ij} = 1$, link j start at node i. With the help of the node-incidence matrix ($\boldsymbol{A}$) and the OD vector ($\boldsymbol{b}$), (P) can be succinctly represented as:

$$\begin{aligned} \underset{\boldsymbol{x}}{\text{minimize}} \ & \boldsymbol{\mu}^\top \boldsymbol{x} + \zeta\sqrt{\boldsymbol{x}^\top \Sigma \boldsymbol{x}} \\ \text{subject to} \ & \boldsymbol{A}\boldsymbol{x} = \boldsymbol{b} \\ & x_{ij} \in \{0, 1\}. \end{aligned} \tag{4}$$

C. Assumptions

This subsection lays down several reasonable assumptions which naturally avoid the 'cycling' problem, *i.e.* the optimal path contains cycles/loops. For a minimal mean-std RSP problem as formulated in (P), normally, it is not guaranteed that the yielded solution does not contains cycles/loops.

Assumption 1 (Travel-Time Assumption): The mean travel times of all the links are strictly positive, i.e. $\boldsymbol{\mu} \succ \boldsymbol{0}$.

Assumption 2 (Cycle Co-Variance Assumption): Removing a cycle in a path results in a path whose total variance is strictly less than the original path.

It is straightforward to justify that Assumption 1 is reasonable, as any link that the traveller travels on should consume time, and thus the mean-travel time for any link should be greater than zero. For the class of applications that we are targeting at, Assumption 2 is also reasonable, as travel on additional links can only add uncertainties, and hence increase the variance. There are, of course, instances in which the co-variance structure makes Assumption 2 invalid, which makes the RSP planning problem much more difficult. We wish to note that we rule out these cases in this paper, but will deliver explicit discussions about cycle-elimination techniques towards the end of Section V.

Theorem 1: The optimal solution to (P) cannot be a cycle-containing path.

Without loss of generality, we only prove the case with one cycle in the path.

Proof: (1) Let $\boldsymbol{x}^c$ represent a one-cycle-containing path, which connects the origin node r to the destination node s. Denote $\mathcal{M}$ as the set of links that is contained in $\boldsymbol{x}^c$, *i.e.* $\forall ij \in \mathcal{M}$, we have $\boldsymbol{x}^c_{ij} = 1$; $\forall ij \notin \mathcal{M}$, we have $\boldsymbol{x}^c_{ij} = 0$. (2) Let $\boldsymbol{x}^o$ be the (only) cycle contained in $\boldsymbol{x}^c$. Denote $\mathcal{M}^c$ as a subset of $\mathcal{M}$, which forms the cycle, *i.e.* the set of links $\{ij|ij \in \mathcal{M}^c\}$ forms a cycling path whose source node is the same as its destination node. (3) Denote $\mathcal{M}\backslash\mathcal{M}^c$ as the set of links that belong to $\mathcal{M}$ but do not belong to $\mathcal{M}^c$, and the corresponding path is represented $\boldsymbol{x}^c - \boldsymbol{x}^o$. Now we begin a two-stage proving process.

Stage 1: prove that $\boldsymbol{x}^c - \boldsymbol{x}^o$ is a valid path for (P), *i.e.* a path connecting r to s. Since $\boldsymbol{x}^c$ is a path connecting r to s, we have

$$\sum_{j:ij\in\mathcal{A}} x^c_{ij} - \sum_{k:ki\in\mathcal{A}} x^c_{ki} = \begin{cases} 1, & i = r \\ -1, & i = s \\ 0, & i \in \mathcal{N} - \{r, s\} \end{cases} . \tag{5}$$

Moreover, since $\boldsymbol{x}^o$ is a cycle, which means that $\forall i \in \mathcal{N}$,

$$\sum_{j:ij\in\mathcal{A}} x^o_{ij} - \sum_{k:ki\in\mathcal{A}} x^o_{ki} = 0. \tag{6}$$

Subtracting Eq. (5) by Eq. (6), we obtain that the $\boldsymbol{x}^c - \boldsymbol{x}^o$ is a still valid path connecting r to s.

Stage 2: prove that the mean-std RSP metric for $\boldsymbol{x}^c - \boldsymbol{x}^o$ is smaller than that of $\boldsymbol{x}^c$. The proving process is as follows:

$$\begin{aligned} & \boldsymbol{\mu}^\top(\boldsymbol{x}^c - \boldsymbol{x}^o) + \zeta\sqrt{(\boldsymbol{x}^c - \boldsymbol{x}^o)^\top \boldsymbol{\Sigma}(\boldsymbol{x}^c - \boldsymbol{x}^o)} \\ & \quad - \left(\boldsymbol{\mu}^\top \boldsymbol{x}^c + \zeta\sqrt{(\boldsymbol{x}^c)^\top \boldsymbol{\Sigma}\boldsymbol{x}^c}\right) \\ & = -\boldsymbol{\mu}^\top \boldsymbol{x}^o + \zeta\left(\sqrt{(\boldsymbol{x}^c - \boldsymbol{x}^o)^\top \boldsymbol{\Sigma}(\boldsymbol{x}^c - \boldsymbol{x}^o)} - \sqrt{(\boldsymbol{x}^c)^\top \boldsymbol{\Sigma}\boldsymbol{x}^c}\right) \\ & < \zeta\left(\sqrt{(\boldsymbol{x}^c - \boldsymbol{x}^o)^\top \boldsymbol{\Sigma}(\boldsymbol{x}^c - \boldsymbol{x}^o)} - \sqrt{(\boldsymbol{x}^c)^\top \boldsymbol{\Sigma}\boldsymbol{x}^c}\right) \\ & = \zeta\frac{\left((\boldsymbol{x}^c - \boldsymbol{x}^o)^\top \boldsymbol{\Sigma}(\boldsymbol{x}^c - \boldsymbol{x}^o) - (\boldsymbol{x}^c)^\top \boldsymbol{\Sigma}\boldsymbol{x}^c\right)}{\left(\sqrt{(\boldsymbol{x}^c - \boldsymbol{x}^o)^\top \boldsymbol{\Sigma}(\boldsymbol{x}^c - \boldsymbol{x}^o)} + \sqrt{(\boldsymbol{x}^c)^\top \boldsymbol{\Sigma}\boldsymbol{x}^c}\right)} \\ & < 0. \end{aligned}$$

During the derivation process, there are two inequalities. The first inequality is the result of applying Assumption 1, and the second inequality is because of Assumption 2, which presumes that the cycle-containing path's variance is larger than that of the corresponding non-cycle containing path. With proofs from Stage 1 and Stage 2, we can conclude that $\boldsymbol{x}^c - \boldsymbol{x}^o$ is a valid path from r to s, and has a strictly lower mean-variance RSP metric than that of $\boldsymbol{x}^c$. □

V. GP3: Gaussian Process Path Planning

In the last section, we have laid down the minimal mean-std RSP problem formulation, two realistic assumptions guaranteeing the natural avoidance of cycle-containing solutions, and the minimal mean-std RSP problem's equivalent relationship with the α-reliable shortest path, and the SOTA RSP. This section presents the GP3 algorithm with its computational complexity analysis.

The first three subsections introduce the GP3 algorithm. In GP3, we first transform the minimal mean-std RSP problem into an indefinite quadratic integer programming (QIP) problem, and then continue to transform the indefinite QIP problem into an equivalent convex QIP problem. Finally, we propose a polynomial complexity algorithm to solve the equivalent convex QIP problem. The GP3 algorithm's flow process and the corresponding computational complexity analysis are presented, with discussions about cycle-elimination techniques towards the end of this section.

A. Transform the Minimal Mean-Std RSP Problem to Indefinite Quadratic Integer Programming Problem

The minimal mean-std RSP problem as stated in (P) is a non-linear integer programming problem, and there is no off-the-shelf solver to efficiently find the global optimal solution.[3] The key non-linear part is the term $\sqrt{\boldsymbol{x}^\top \boldsymbol{\Sigma} \boldsymbol{x}}$, in the following, we will, step by step, convert (P) to an equivalent but easy-to-solve problem.

First, we simplify the objective function, and put the complex representation forms, *i.e.* $\sqrt{\boldsymbol{x}^\top \boldsymbol{\Sigma} \boldsymbol{x}}$ into the constraints. We will get:

$$\begin{aligned}
&\underset{\boldsymbol{x},t}{\text{minimize}}\ t\\
&\text{subject to } \boldsymbol{\mu}^\top \boldsymbol{x} + \zeta\sqrt{\boldsymbol{x}^\top \Sigma \boldsymbol{x}} \le t\\
&\qquad\qquad \boldsymbol{A}\boldsymbol{x} = \boldsymbol{b}\\
&\qquad\qquad x_{ij} \in \{0, 1\}.
\end{aligned} \tag{7}$$

The first constraint in Eq. (7) can be further transformed as follows:

$$\begin{aligned}
&\boldsymbol{\mu}^\top \boldsymbol{x} + \zeta\sqrt{\boldsymbol{x}^\top \Sigma \boldsymbol{x}} \le t\\
&\quad \Rightarrow \zeta\sqrt{\boldsymbol{x}^\top \Sigma \boldsymbol{x}} \le t - \boldsymbol{\mu}^\top \boldsymbol{x}\\
&\quad \Rightarrow \zeta^2 \boldsymbol{x}^\top \Sigma \boldsymbol{x} \le (t - \boldsymbol{\mu}^\top \boldsymbol{x})^2\\
&\quad \Rightarrow \zeta^2 \boldsymbol{x}^\top \Sigma \boldsymbol{x} \le (t - \boldsymbol{\mu}^\top \boldsymbol{x})^\top (t - \boldsymbol{\mu}^\top \boldsymbol{x})\\
&\quad \Rightarrow \zeta^2 \boldsymbol{x}^\top \Sigma \boldsymbol{x} \le \boldsymbol{x}^\top \boldsymbol{\mu}\boldsymbol{\mu}^\top \boldsymbol{x} - 2t\boldsymbol{\mu}^\top \boldsymbol{x} + t^2\\
&\quad \Rightarrow \boldsymbol{x}^\top (\zeta^2 \Sigma - \boldsymbol{\mu}\boldsymbol{\mu}^\top)\boldsymbol{x} + 2t\boldsymbol{\mu}^\top \boldsymbol{x} \le t^2
\end{aligned} \tag{8}$$

The last row of Eq. (8) is a quadratic function over $\boldsymbol{x}$, and will be used to replace the first constraint in Eq. (7). However, in order to get an equivalent replacement between the constraints, an additional constraint needs to be inserted, which is

$$\boldsymbol{\mu}^\top \boldsymbol{x} \le t. \tag{9}$$

Because during the constraint conversion process (from the second line to the third line) in Eq. (8)), we have assumed that $\boldsymbol{\mu}^\top \boldsymbol{x} \le t$. In this case, the first constraint in Eq. (7) is equivalently transformed to the two constraints as expressed in Eq. (8) and Eq. (9), and the original minimal mean-std RSP problem in Eq. (7) is equivalently transformed into the following form:

$$\begin{aligned}
&\underset{\boldsymbol{x},t}{\text{minimize}}\ t\\
&\text{subject to } \boldsymbol{x}^\top (\zeta^2 \Sigma - \boldsymbol{\mu}\boldsymbol{\mu}^\top)\boldsymbol{x} + 2t\boldsymbol{\mu}^\top \boldsymbol{x} \le t^2\\
&\qquad\qquad \boldsymbol{\mu}^\top \boldsymbol{x} \le t\\
&\qquad\qquad \boldsymbol{A}\boldsymbol{x} = \boldsymbol{b}\\
&\qquad\qquad x_{ij} \in \{0, 1\}.
\end{aligned} \tag{10}$$

Now, we use the fact that $\forall t \ge 0$, we have $\arg\min_t t^2 = \arg\min_t t$, then the objective of Eq. (10) can be squared without changing the problem's optimal solution. After that, the first constraint in Eq. (10) can be converted into the objective, and the converted problem is:

$$\begin{aligned}
&\underset{\boldsymbol{x}}{\text{minimize}}\ \boldsymbol{x}^\top (\zeta^2 \Sigma - \boldsymbol{\mu}\boldsymbol{\mu}^\top)\boldsymbol{x} + 2t\boldsymbol{\mu}^\top \boldsymbol{x}\\
&\text{subject to } \boldsymbol{\mu}^\top \boldsymbol{x} \le t\\
&\qquad\qquad \boldsymbol{A}\boldsymbol{x} = \boldsymbol{b}\\
&\qquad\qquad x_{ij} \in \{0, 1\}.
\end{aligned} \tag{11}$$

Note that in Eq. (11), t is not a decision variable, but an input parameter. For any given $t \ge 0$, we solve the current problem as stated in Eq. (11), then if the optimal objective value is less than t^2, it means that the problem in Eq. (10) is feasible, which means that we find a feasible solution to the original minimal mean-std RSP problem in Eq. (7). On the other hand, if for a given $t \ge 0$, the solution to Eq. (11) cannot yield an objective value that is less than t^2, it means that there is no feasible solution to Eq. (10), hence, no feasible solution to Eq. (7). In this case, it means that the value t is too small, and we need to increase it to get a feasible solution, so as to get a feasible RSP as the output.

The problem as defined in Eq. (11) is an indefinite QIP problem, in the next subsection, we will convert the indefinite QIP problem into a convex QIP problem.

B. Transform the Indefinite QIP to Convex QIP

In the last subsection, we have successfully transformed the original problem (Eq. (7)) into the problem defined in Eq. (11). The optimal solution to Eq. (11) will judge whether the originally proposed value (t) in (Eq. (7)) is feasible or not.

However, the problem as defined in Eq. (11) is a large scale $0-1$ indefinite quadratic integer programming problem, and there is also no efficient off-the-shelf solvers for the polynomial-time optimal solution. Canonical non-linear integer programming software is usually inefficient, fortunately, the flow balancing constraints impart a very special structure, which makes it possible for us to transform the problem and hence reach an efficient polynomial time solver.

There are two sources of difficulties in Eq. (11), first, the quadratic term in the objective function is neither positive

[3] Here, the word 'efficiently' refers to an efficient algorithm which return the optimal solution with polynomial computational complexity.

definite, nor negative definite, which means that it is an indefinite quadratic matrix. Secondly, the decision variables x_{ij} belong to $\{0, 1\}$, whose non-continuity poses great difficulties in finding efficient solvers. In the following, we will first convert the quadratic part in the objective function of Eq. (11) to a positive definite matrix, and then use the max-flow method to find the optimal solution.

For representation simplicity, we make the following definitions: (1) $\boldsymbol{e} \in \mathcal{R}^m$, with all the elements equal to 1, which means that $\boldsymbol{e} = [1, 1, \ldots, 1]^\top$; (2) $\boldsymbol{H} = \zeta^2\boldsymbol{\Sigma} - \boldsymbol{\mu}\boldsymbol{\mu}^\top$; (3) $\lambda_{\min}$: the minimum of the eigenvalues of matrix $\boldsymbol{H}$, note that it is possible that $\lambda_{\min} < 0$, which means that $\boldsymbol{H}$ is an indefinite matrix.

Since $\boldsymbol{x} \in \{0, 1\}^m$, it is easy to verify that $\boldsymbol{x}^\top\boldsymbol{x} = \boldsymbol{e}^\top\boldsymbol{x}$. Then, the objective function in Eq. (11) can be converted as follows:

$$\begin{aligned}
&\boldsymbol{x}^\top(\zeta^2\Sigma - \boldsymbol{\mu}\boldsymbol{\mu}^\top)\boldsymbol{x} + 2t\boldsymbol{\mu}^\top\boldsymbol{x}\\
&\quad = \boldsymbol{x}^\top\boldsymbol{H}\boldsymbol{x} + 2t\boldsymbol{\mu}^\top\boldsymbol{x}\\
&\quad = \boldsymbol{x}^\top\boldsymbol{H}\boldsymbol{x} + 2t\boldsymbol{\mu}^\top\boldsymbol{x} - (\lambda_{\min} - 1)\boldsymbol{x}^\top\boldsymbol{x} + (\lambda_{\min} - 1)\boldsymbol{e}^\top\boldsymbol{x}\\
&\quad = \boldsymbol{x}^\top(\boldsymbol{H} - (\lambda_{\min} - 1)\boldsymbol{I})\boldsymbol{x} + (2t\boldsymbol{\mu}^\top + (\lambda_{\min} - 1)\boldsymbol{e}^\top)\boldsymbol{x},
\end{aligned}$$

where $\boldsymbol{I}$ is an $m \times m$ identity matrix. Now, the problem defined in Eq. (11) is converted to:

$$\begin{aligned}
\min_{\boldsymbol{x}}\ & \boldsymbol{x}^\top(\boldsymbol{H} - (\lambda_{\min} - 1)\boldsymbol{I})\boldsymbol{x} + (2t\boldsymbol{\mu}^\top + (\lambda_{\min} - 1)\boldsymbol{e}^\top)\boldsymbol{x}\\
\text{s.t. } & \boldsymbol{\mu}^\top\boldsymbol{x} \le t\\
& \boldsymbol{A}\boldsymbol{x} = \boldsymbol{b}\\
& x_{ij} \in \{0, 1\}. \qquad (12)
\end{aligned}$$

Since $\lambda_{\min}$ is the minimal eigenvalue of matrix $\boldsymbol{H}$, thus it is straightforward to verify that the quadratic term in the objective function in Eq. (12) is a positive definite matrix with the minimal eigenvalue at 1. Hence the problem defined in Eq. (12) is a convex quadratic integer programming problem.

So far, we have converted the indefinite QIP problem into an equivalent convex quadratic integer programming problem, but the $\{0, 1\}$ constraint still makes the problem difficult to solve. In the next subsection, we will propose an efficient solver which finds the optimal solution to Eq. (12) within polynomial time. Before that, we would like to simplify the representation form of Eq. (12).

Defining $\boldsymbol{Q} = \boldsymbol{H} - (\lambda_{\min} - 1)\boldsymbol{I}$, and $\boldsymbol{l} = 2t\boldsymbol{\mu} + (\lambda_{\min} - 1)\boldsymbol{e}$, the problem defined in Eq. (12) can be succinctly represented as:

$$\begin{aligned}
\underset{\boldsymbol{x}}{\text{minimize}}\ & \boldsymbol{x}^\top\boldsymbol{Q}\boldsymbol{x} + \boldsymbol{l}^\top\boldsymbol{x}\\
\text{subject to } & \boldsymbol{\mu}^\top\boldsymbol{x} \le t\\
& \boldsymbol{A}\boldsymbol{x} = \boldsymbol{b}\\
& x_{ij} \in \{0, 1\}. \qquad (13)
\end{aligned}$$

C. An Efficient Algorithm for the Convex QIP Problem

The canonical methods to solve the convex QIP problem in Eq. (13) is branch and bound [42], which has an exponential computational complexity. It prohibits the application to path planning in large scale transportation networks. In this subsection, we will first lay down the polynomial-complexity algorithm to solve Eq. (13), then prove its correctness.

We first solve the relaxed problem of Eq. (13), which is to relax the Boolean constraints to be convex constraints, as stated in the following:

$$\begin{aligned}
\underset{\boldsymbol{x}}{\text{minimize}}\ & \boldsymbol{x}^\top\boldsymbol{Q}\boldsymbol{x} + \boldsymbol{l}^\top\boldsymbol{x}\\
\text{subject to } & \boldsymbol{\mu}^\top\boldsymbol{x} \le t\\
& \boldsymbol{A}\boldsymbol{x} = \boldsymbol{b}\\
& 0 \le x_{ij} \le 1. \qquad (14)
\end{aligned}$$

Eq. (14) is a standard quadratic programming (QP) problem, which can be solved efficiently with off-the-shelf solvers. We define the optimal solution to Eq. (14) as $\boldsymbol{x}_t$, which is dependent on the input value t. Since all the elements of $\boldsymbol{x}_t$ are between 0 and 1, we can formulate a max-flow path planning problem [43] as follows: find a set of elementary paths $\boldsymbol{x}_{p_i}$ $(1 \le i \le K)$ connecting origin r to destination s, such that,

$$\begin{aligned}
\sum_{i=1}^{K}\theta_i\boldsymbol{x}_{p_i} &= \boldsymbol{x}_t\\
\sum_{i=1}^{K}\theta_i &= 1\\
\theta_i &\ge 0. \qquad (15)
\end{aligned}$$

Note that Eq. (15) is a max-flow problem, which decomposes the previous problem's (Eq. (13)) solution into several elementary paths, from which the optimal solution to Eq. (13) is chosen, and the optimality will be proved. The max-flow problem is readily solved in the path planning community, and there exists several polynomial-complexity algorithms for the solution. The number of elementary paths, *i.e.* K, depends on $\boldsymbol{x}_t$, and in practice, it is usually more or less comparable with the number of links of the shortest path in the transportation network.

Now, we will prove that out of the elementary paths, the one with the minimal objective value and satisfying the constraints in Eq. (13) will be the optimal solution to the problem in Eq. (13).

Theorem 2: The optimal solution to Eq. (13) is one of the elementary paths which holds the minimal objective value and satisfies the inequality constraints.

Before the formal proving process, we sketch the proof skeleton here. We first prove that $\nabla f(\boldsymbol{x}_t)^\top(\boldsymbol{x}_{p_i} - \boldsymbol{x}_t) = 0$, where $f(\boldsymbol{x})$ is the objective function in Eq. (13), then we establish a contradiction that if there is another path which is not in the elementary paths, and has an objective value lower than that of all the elementary paths, we will violate the optimality condition, which makes $\boldsymbol{x}_t$ not the optimal solution to the problem in Eq. (14).

Proof: Step 1: prove that $\forall p_i$, we have $\nabla f(\boldsymbol{x}_t)^\top(\boldsymbol{x}_{p_i} - \boldsymbol{x}_t) = 0$, where $f(\boldsymbol{x})$ is the objective function.

Since $\boldsymbol{x}_t$ is the optimal solution to Eq. (14), hence it satisfies all the constraints, *e.g.* $\boldsymbol{\mu}^\top\boldsymbol{x_t} \le t$. Recall the original problem as defined in Eq. (7), we can make a further conclusion that $\boldsymbol{\mu}^\top\boldsymbol{x_t}$ is strictly less than t, *i.e.* $\boldsymbol{\mu}^\top\boldsymbol{x_t} < t$.

Because if $\boldsymbol{\mu}^\top \boldsymbol{x_t} = t$, the first inequality constraint in Eq. (7) would not be satisfied, and the problem is infeasible.

On the other hand, $\boldsymbol{x}_{p_i}$ is the elementary path connecting r to s, which means that it satisfies the flow balancing constraints. Now, we define $\boldsymbol{d} = \boldsymbol{x}_{p_i} - \boldsymbol{x}_t$, which is a feasible direction starting from the point $\boldsymbol{x}_t$.

Consider the term $\boldsymbol{x}_t + \lambda(\boldsymbol{x}_{p_i} - \boldsymbol{x}_t)$, where λ is arbitrarily small, *i.e.* $-\epsilon \leq \lambda \leq \epsilon$, with ϵ being a very small positive number, we can get that $\forall\lambda$ such that $-\epsilon \leq \lambda \leq \epsilon$, the resulting point is a feasible solution to Eq. (14). Hence, $\nabla f(\boldsymbol{x}_t)^\top (\boldsymbol{x}_{p_i} - \boldsymbol{x}_t) = 0$. Otherwise, suppose that we could flip the sign of λ, and always get a negative gradient direction, which will decrease the objective value, and in this case, the original solution would not be the minimum point anymore.

Second step: we establish a contradiction that if there is another path who has a lower objective value than all the elementary paths' objective value, the path's objective value with be also lower than $\boldsymbol{x_t}$'s objective value.

Suppose there is another path, $\boldsymbol{x}_q$, which has a smaller objective value than that of all the $\boldsymbol{x}_{p_i}$, $\forall 1 \leq i \leq K$, we have:

$$\nabla f(\boldsymbol{x}_{p_i})^\top (\boldsymbol{x}_q - \boldsymbol{x}_{p_i}) < 0. \quad (16)$$

Recall that $\boldsymbol{x}_t = \sum_{i=1}^K \theta_i \boldsymbol{x}_{p_i}$, since the objective function $(f(x))$ is a quadratic function, which makes its derivative function a linear one, and we have

$$\nabla f(\boldsymbol{x}_t) = \sum_{i=1}^K \theta_i \nabla f(\boldsymbol{x}_{p_i}). \quad (17)$$

Deriving the relationship between $\boldsymbol{x}_q$ and $\boldsymbol{x}_t$, we have

$$\begin{aligned}
&\nabla f(\boldsymbol{x}_t)^\top (\boldsymbol{x}_q - \boldsymbol{x}_t)\\
&= \sum_{i=1}^K \theta_i \nabla f(\boldsymbol{x}_{p_i})^\top (\boldsymbol{x}_q - \boldsymbol{x}_t)\\
&= \sum_{i=1}^K \theta_i \nabla f(\boldsymbol{x}_{p_i})^\top (\boldsymbol{x}_q - \boldsymbol{x}_t) - \sum_{i=1}^K \theta_i \nabla f(\boldsymbol{x}_{p_i})^\top (\boldsymbol{x}_{p_i} - \boldsymbol{x}_t)\\
&= \sum_{i=1}^K \theta_i \nabla f(\boldsymbol{x}_{p_i})^\top (\boldsymbol{x}_q - \boldsymbol{x}_{p_i})\\
&< 0.
\end{aligned}$$

From the second line to the third line in the derivation, we make use of the fact that $\nabla f(\boldsymbol{x}_t)^\top (\boldsymbol{x_{p_i}} - \boldsymbol{x_t}) = 0$ and the property of Eq. (17). From the fourth line to the last line in the derivation, we make use of Eq. (16). However, the conclusion is that $\boldsymbol{x}_q$ has a lower objective value than that of $\boldsymbol{x}_t$, which violates the fact that $\boldsymbol{x}_t$ is the optimal solution. Hence, there does not exist such a path $\boldsymbol{x}_p$. □

To conclude this subsection, we propose an efficient algorithm to solve the convex QIP problem in Eq. (13). We first relax the convex QIP problem to a standard QP problem, and use the off-the-shelf solver to calculate the optimal solution. Then, we decompose the optimal solution to several elementary paths, and select the path, which has the smallest objective value and satisfies the inequality constraint as the final optimal solution to Eq. (13). The theoretical proof is also provided.

D. Algorithm Flow Process for GP3

In the previous three subsections, we have introduced the GP3 algorithm in detail, and this subsection will present the GP3's algorithm flow process. Before the algorithm flow process introduction, we first estimate the lower bound and upper bound of the value t in Eq. (7). Since the objective of Eq. (7) is to minimize t subject to a feasible path connecting r with s, we can evaluate any path's t value as its upper bound.

We calculate the standard fastest path in expectation, denoted as $\boldsymbol{x}_f$, thus we have the upper bound of t as $\boldsymbol{\mu}^\top \boldsymbol{x}_f + \zeta\sqrt{\boldsymbol{x}_f^\top \boldsymbol{\Sigma} \boldsymbol{x}_f}$. Deriving the lower bound of t is a little difficult, one of the lazy and simple solution is to let the lower bound of t be zero. However, there are much better estimates for t's lower bound. Let us denote the minimal variance path as $\boldsymbol{x}_{\text{var}}$, which is a path connecting r to s, while minimizes the objective function $\boldsymbol{x}^\top \boldsymbol{\Sigma} \boldsymbol{x}$. It is worth noting that the minimal variance path can be calculated in polynomial time after considering the GP3 algorithm. Then the lower bound of t can be estimated as $\boldsymbol{\mu}^\top \boldsymbol{x}_f + \zeta\sqrt{\boldsymbol{x}_{\text{var}}^\top \boldsymbol{\Sigma} \boldsymbol{x}_{\text{var}}}$. Now we have:

$$\boldsymbol{\mu}^\top \boldsymbol{x}_f + \zeta\sqrt{\boldsymbol{x}_{\text{var}}^\top \boldsymbol{\Sigma} \boldsymbol{x}_{\text{var}}} \leq t \leq \boldsymbol{\mu}^\top \boldsymbol{x}_f + \zeta\sqrt{\boldsymbol{x}_f^\top \boldsymbol{\Sigma} \boldsymbol{x}_f}. \quad (18)$$

Before introducing the flow process of GP3, we introduce the following definition.

Definition 1 (Tolerance η): A tolerance value η is a threshold value representing the tolerance of the maximal absolute value difference between the optimal value of an optimization problem and the current-solution-yielding value.

In the reliable shortest path planning problem, for example, we have the absolute best RSP, which yields an objective value of f^*, and we have a proposed path, which yields a value of f. If $|f - f^*| < \eta$, it means that our proposed solution is within the tolerance of the problem. The algorithm flow process of GP3 is depicted in **Algorithm 1**. When supplied with required parameters, *i.e.* $\boldsymbol{\mu}$, $\boldsymbol{\Sigma}$ and ζ, GP3 will output the a prior optimal path.

Tolerance η is a very useful parameter to gauge the accuracy of the RSP planning algorithm, and will be used in the algorithm's computational complexity analysis. However, in practice, the ratio of η and the upper bound's value is often used, as it reflects the percentage of deviation between the optimal solution and the current best solution. Thus, we introduce the following definition:

Definition 2 (Relative Tolerance η_r): A relative tolerance value η_r is a threshold value defining the maximal allowed value of the ratio of the current upper and lower bound's difference to the current upper bound.

With the definition of η_r, the algorithm will terminate when

$$\frac{\text{UB} - \text{LB}}{\text{UB}} \leq \eta_r, \quad (19)$$

where UB and LB are the values of the upper bound and lower bound, respectively.

E. Computational Complexity Analysis

This section analyzes the computational complexity of GP3. The computational cost of an operation can often be expressed

Algorithm 1 The Algorithm Flow Process of GP3

```
Input: GP parameters for the underlying transportation
       network ($\boldsymbol{\mu}$ and $\boldsymbol{\Sigma}$), reliability coefficient $\zeta$,
       relative tolerance $\eta_r$, current node $r$, destination
       node $s$.
Output: minimal mean-std RSP
Calculate the upper bound and lower bound of $t$
according to Eq. (18), and denote the values as $t_{\max}$ and
$t_{\min}$, respectively.
calculate the minimum eigenvalue of matrix
$\boldsymbol{H} = \zeta^2 \boldsymbol{\Sigma} - \boldsymbol{\mu}\boldsymbol{\mu}^\top$;
Convert to the problem defined in Eq. (13);
while $(t_{max} - t_{min})/t_{max} > \eta_r$ do
    propose $t = (t_{\max} + t_{\min})/2$; Call QP solver to solve
    the relaxed problem defined in Eq. (14);
    Call the max-flow algorithm to find the set of
    elementary paths satisfying Eq. (15);
    Evaluate all the elementary paths and find the one
    with the minimal objective value of Eq. (13) while
    satisfying the inequality constraint;
    Use the selected elementary path to calculate the
    objective value in Eq. (7), record the path as $\boldsymbol{x}_e$, and
    denote the objective value as val;
    if $val \leq t$ then
        $t_{\max} = \text{val}$;
    else
        $t_{\min} = t$;
Return the path denoted as $\boldsymbol{x}_e$;
Final.
```

through the number of floating-point operations (flops). A flop is defined as an addition, subtraction, multiplication or division of two floating-point numbers. To evaluate the complexity of an algorithm, we count the total number of flops; express it as a function (usually a polynomial) of the dimensions of the matrices and vectors involved, and simplify the expression by ignoring all terms except the leading terms.

Examining Algorithm 1, we can see that the overhead (before the 'while' loop) of GP3 involves calculating the shortest path, and minimal variance path, and computing the eigenvalues of matrix $\boldsymbol{H}$. The related computational complexity can be represented by $\mathcal{O}(m^2)$ for the shortest path and minimal variance path computation. The computational complexity for computing the eigenvalues of matrix $\boldsymbol{H}$ is also $\mathcal{O}(m^2)$. Thus, the overhead computational complexity is $3\mathcal{O}(m^2)$.

Each time in the loop of Algorithm 1, we need to solve the relaxed QP problem, whose computational complexity is $\mathcal{O}(m^3)$, and solve the max-flow problem, whose computational complexity is $\mathcal{O}(m^2 n)$. Evaluating the elementary path's performance is pure matrix multiplication, which also holds a $\mathcal{O}(m^2 n)$ complexity. Thus, inside the loop, the scale[4] of computational complexity is $\mathcal{O}(m^3 + m^2 n)$. Considering the fact that $m > n$ in realistic networks, we represent the computational complexity inside the loop as $\mathcal{O}(m^3)$.

Now, the key part is to estimate the number of loops in the GP3 algorithm. For a given transportation network, with fixed parameters, the algorithm will stop when $t_{\max} - t_{\min} \leq \eta$, since each time, we are able to cut the value of $t_{\max} - t_{\min}$ by half, the total number of loops that GP3 needs can then be calculated as $\lceil \log_2((t_{\max} - t_{\min})/\eta) \rceil$. Here $\lceil x \rceil$ refers to find the minimal integer value that is larger than or equal to the x.

Thus, GP3's computational complexity is $\lceil \log_2((t_{\max} - t_{\min})/\eta) \rceil \mathcal{O}(m^3)$ after only accepting the leading terms, where m is the number of links in the transportation network.

F. Discussions About Cycle-Elimination Techniques for GP3

In Section IV-C, we presume that the 'natural' solution of GP3 does not contain cycles. However, in real world applications, one cannot justify the absolute in-existence of a cycle from GP3's output. Now, the question is how to formulate the problem in a way of completely avoiding cycles in the output path. Theoretically, DFJ method [44] for cycle elimination can be employed in GP3's problem formulation, specifically, we add the following constraints into Eq. (7):

$$\forall i \in \mathcal{N} \sum_{j:ij \in \mathcal{E}} x_{ij} \leq 1. \tag{20}$$

The rationale of Eq. (20) is that the solution to GP3 cannot contain two links, which come out of the same node. We wish to note that adding the set of constraints of Eq. (20) makes the problem much more complex than the original one, as there are n additional inequality constraints in the newly formed problem. In the experiment section, we stick with the original problem formulation, and the aim of this subsection is to provide a way of eliminating cycles in path planning.

VI. Experimental Results and Analysis

In this section, we compare the performance of GP3 with three state-of-the-art algorithms over various transportation networks. For state of the arts, we select the subgradient (SG) method proposed in [27], the subgradient projection (SP) algorithm proposed in [21] and the Lagrangian relaxation (LR) method proposed in [15]. Those three algorithms are representative in the minimal mean-std RSP planning research domain.

We first test GP3 and the state-of-the-art algorithms in a self-constructed simple network, and demonstrate the evolution process of GP3's parameters iteration by iteration, then, we perform a case study in the Sioux Falls network which is a canonical transportation test network. Later, we apply GP3 to three real networks, namely, Anaheim, Chicago Sketch, and Barcelona, with real mean travel time but self-generated covariance matrix, and this section ends with GP3's application to a large scale network with real traffic data.

Table I summarizes the four algorithms' parameter configuration. The algorithms are implemented in MATLAB(2018b) as well as Python 3.6 (Python 3.6 is used for testing the algorithms' deployment to Chengdu Network).[5] All the tests,

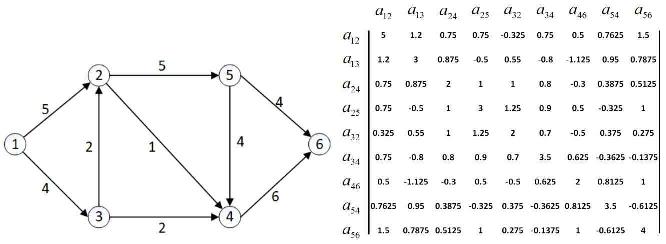
[4] Here, the scale of computational complexity means that it may incur several times of computation with complexity of $\mathcal{O}(m^2 n)$.

[5] Both Matlab code and Python code are available at https://github.com/scott0793-debug/GP3.

TABLE I

ALGORITHM PARAMETERS

Parameter	Description	Algorithm	Value
η_r	relative tolerance/duality gap	GP3, SG, SP, LR	0.05
MaxIt	maximum iteration number	SG, SP, LR	200
δ	step size factor	SG, SP, LR	0.1



	a_{12}	a_{13}	a_{24}	a_{25}	a_{32}	a_{34}	a_{46}	a_{54}	a_{56}
a_{12}	5	1.2	0.75	0.75	-0.325	0.75	0.5	0.7625	1.5
a_{13}	1.2	3	0.875	-0.5	0.55	-0.8	-1.125	0.95	0.7875
a_{24}	0.75	0.875	2	1	1	0.8	-0.3	0.3875	0.5125
a_{25}	0.75	-0.5	1	3	1.25	0.9	0.5	-0.325	1
a_{32}	0.325	0.55	1	1.25	2	0.7	-0.5	0.375	0.275
a_{34}	0.75	-0.8	0.8	0.9	0.7	3.5	0.625	-0.3625	-0.1375
a_{46}	0.5	-1.125	-0.3	0.5	-0.5	0.625	2	0.8125	1
a_{54}	0.7625	0.95	0.3875	-0.325	0.375	-0.3625	0.8125	3.5	-0.6125
a_{56}	1.5	0.7875	0.5125	1	0.275	-0.1375	1	-0.6125	4

(a) Simple network topology (b) Link travel time covariance

Fig. 1. The simple network (adopted from [27]) topology and corresponding link travel time covariance matrix.

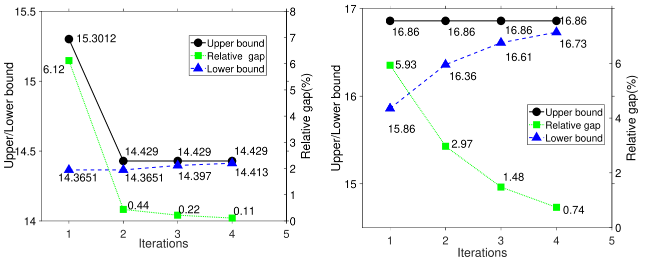
except for the experiments in Section VI-D, are executed on a 2.80 GHz, Intel(R) Xeon(R) CPU E5-2680 v2 computer with the 64-bit version of the Windows 10 operating system and 32GB RAM. The experiments in Section VI-D are conducted on a Linux-based operating system with an i9-9700K @ 3.6GHz (8 cores) processor and 32 GB of DDR4 3000Mhz RAM, which is equipped with the NVIDIA GeForce RTX 2070, Super GPU with 8 GB of VRAM. The reason that we change hardware for the experiments in Chengdu is that it takes quite a long time (several minutes) for GP3 to yield the optimal solution with the original hardware setup. With the help of GPU, we are able to accelerate the computation time to be in the range of 40 seconds. We use cuOSQP [45] to solve the QP problem within GP3, and believe that if GPU is upgraded to NVIDIA GeForce RTX 3090, GP3's required computation time can be further trimmed down to as low as 10 seconds (as the authors claim in [45] that the acceleration of the GPU-assisted QP solver is 15 to 270 times faster than that of the CPU solver.).

Additionally, it is worth noting that, over the entire experimental section, the units for all the time-related values, *i.e.* performance, upper bound, lower bound, algorithm running time, are seconds.

A. Simple Network for Illustration

In order to demonstrate the evolution process of GP3 and its characteristics within each iteration, we present the results of a simple yet illustrative network borrowed from [27]. Fig. 1 shows the topology of the simple network as well as the corresponding link travel time covariance matrix. The numbers on the links of the network indicate the mean values of the travel time. For the experiments, the origin node $r = 1$, and the destination node $s = 6$.

Table II shows the performance (upper bound, lower bound and the relative gap) evolution process of the four algorithms iteration by iteration. We can see that GP3 converges with the minimal number of iterations (the number of required iterations is 2), while SG, which is the second fastest convergence algorithm, requires 4 iterations to converge. Moreover, with simple calculation, we can see that the absolute gap of GP3 (UB − LB) will shrink by at least a half after each iteration. Fig. 2 shows GP3's evolution process of the upper bound, lower bound and the relative gap ((UB−LB)/UB) for different reliability coefficients ($\zeta = 1$, and $\zeta = 2$, respectively). Note that for better visualization purpose, we plot GP3's related performance for 4 iterations, but we can see from the figure that after 2 iterations, GP3's relative gap has already decreased to below 0.05, which is one of the algorithm's termination conditions.



(a) $\zeta = 1$ (b) $\zeta = 2$

Fig. 2. Performance evolution process of GP3 over the simple network shown in Fig. 1a, for different reliability coefficients.

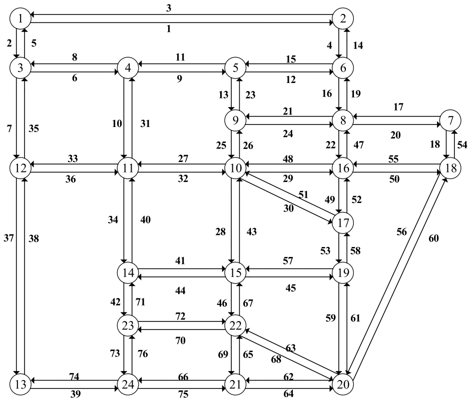


Fig. 3. Sioux falls network.

B. Canonical Transportation Network (Sioux Falls Network)

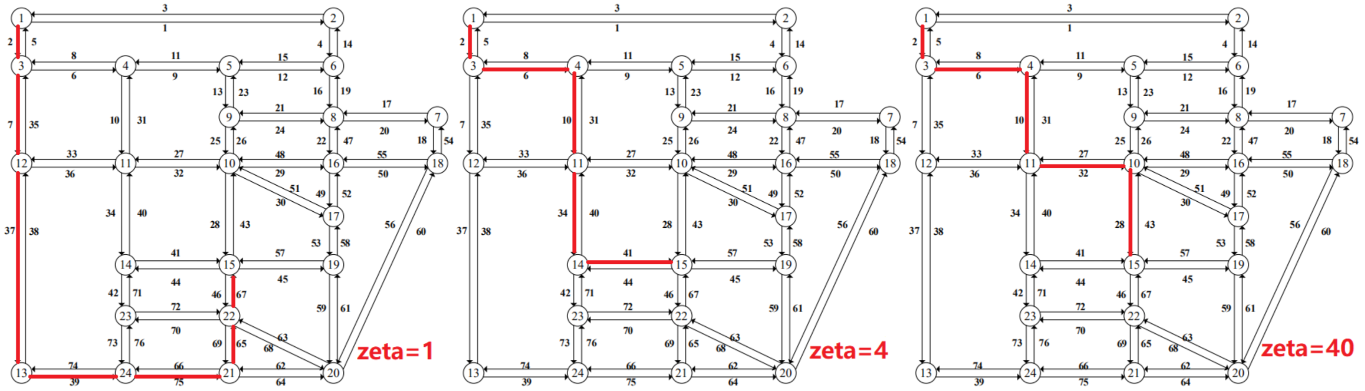
This subsection tests the performance of the four algorithms (GP3, SG, SP, LR) over the Sioux Falls network, which is visualized in Fig. 3. Sioux Falls network is a canonical network for transportation test problems. It has 24 nodes and 76 links, with publicly available mean travel time, *i.e.* μ_{ij}. We use the same method as reported in [20] to randomly generate the covariance matrix. Specifically, the standard deviation σ_{ij} of link ij's travel time is generated as $\sigma_{ij} = \text{Uniform}(0, \kappa)\mu_{ij}$, where $\kappa > 0$ is the maximum value of the coefficient of variation, and is set to be 0.2 in this subsection. Note that different values of κ will be tested in the next subsection, where we apply GP3 to three real networks, namely, Anaheim, Chicago sketch, and Barcelona.

GP3, SG, SP and LR are tested to generate the reliable shortest path from the origin node $r = 1$ to the destination

TABLE II

PERFORMANCE EVOLUTION PROCESS OF GP3, SG, SP AND LR FOR THE SIMPLE NETWORK (RELIABILITY COEFFICIENT $\zeta = 1$)

iteration	GP3			SG			SP			LR		
	UB	LB	relative gap	UB	LB	relative gap	UB	LB	relative gap	UB	LB	relative gap
1	15.30	14.37	0.06	15.30	10.80	0.29	15.30	13.00	0.15	15.30	12.27	0.20
2	14.43	14.37	0.004	15.30	12.65	0.17	15.30	13.22	0.14	15.30	12.59	0.18
3	-	-	-	15.30	13.54	0.12	15.30	13.42	0.12	15.30	12.87	0.16
4	-	-	-	14.43	13.87	0.04	15.30	13.61	0.11	15.30	13.11	0.14
5	-	-	-	-	-	-	15.30	13.77	0.10	15.30	13.33	0.13
6	-	-	-	-	-	-	14.43	13.79	0.04	15.30	13.52	0.12
7	-	-	-	-	-	-	-	-	-	15.30	13.69	0.11
8	-	-	-	-	-	-	-	-	-	14.43	13.71	0.05
9	-	-	-	-	-	-	-	-	-	14.43	13.78	0.04



(a) $\zeta = 1$ (b) $\zeta = 4$ (c) $\zeta = 40$

Fig. 4. Different resulting RSPs for different ζ values from node 1 to node 15.

TABLE III

PERFORMANCE OF GP3, SG, SP AND LR FOR THE SIOUX FALLS NETWORK ($r = 1$, $s = 15$), NOTE THAT ‘GAP’ REFERS TO THE ‘RELATIVE GAP’, WHICH IS CALCULATED AS (UB-LB)/UB

ζ	GP3			SG			SP			LR		
	gap	iteration	time	gap	iteration	time	gap	iteration	time	gap	iteration	time
1	0.0045	1	0.0329	0.0072	1	0.0046	0.0112	2	0.0065	0.0031	5	0.0089
4	0.0072	2	0.0546	0.0028	4	0.0063	0.0171	2	0.0042	0.0080	23	0.0277
40	0.0075	5	0.2681	0.0065	7	0.0092	0.0499	6	0.0065	0.0071	36	0.0399

node $s = 15$ for illustration purpose. Fig. 4 shows different resulting RSPs for different reliability coefficients from the GP3 algorithm, and Table III shows the relative gap, the number of needed iterations and total computation time for different algorithms. From Table. III, we can see that GP3's needs the least number of iterations to converge, and all the algorithms' relative gaps are very small. Admittedly, on average, GP3's total computation time is higher than those of the three algorithms, this is because GP3's core computation time is allocated to solving a QP problem, while all of the other three algorithms are allocating core time for Dijkstra's algorithm, and Dijkstra's algorithm is much faster than a QP solver. However, GP3 guarantees a polynomial computational complexity as proved in the last section.

We further test GP3 for the RSP planning with the origin node $r = 1$, and the destination node s covering all the remaining nodes in the Sioux Falls network, *i.e.* $s = 2, 3, \ldots, 24$, and the reliability coefficient ζ spans from 1 to 45. We set the maximum ζ at 45, because experimentally, we find that when $\zeta \geq 45$, the resulting RSP remains at the minimal variance path. Table IV shows the resulting performance of the GP3 algorithm, and for $s = 15$, we use three different markers to indicate the three different physical RSPs as illustrated in Fig. 4, *i.e.* the numbers with the † symbol (the first three) corresponds to the path visualized in Fig. 4a, the numbers with the $\star$ symbol corresponds to the path visualized in Fig. 4b, and the numbers with the ‡ symbol (the last two) corresponds to the path visualized in Fig. 4c.

C. Real Network Comparison: Anaheim, Chicago Sketch and Barcelona

In the previous two subsections, we have compared the performance of GP3 with the other three state-of-the-art algorithms (SG, SP and LR) over two simple yet illustrative self-constructed networks and the canonical transportation test network (Sioux Falls network). This subsection will evaluate GP3's performance over three realistic networks, namely Anaheim, Chicago Sketch and Barcelona. The network details (the number of nodes and links) are presented in Table V.

TABLE IV

PERFORMANCE OF GP3 FOR DIFFERENT DESTINATIONS AND DIFFERENT ζ VALUES FOR THE SIOUX FALLS NETWORK (r = 1)

s	the value of ζ											
	1	2	3	4	10	15	20	25	30	35	40	45
2	361.58	364.78	367.98	371.18	390.38	406.38	422.38	438.38	454.38	470.38	486.38	502.38
3	242.39	249.38	256.37	263.36	305.32	340.28	375.24	410.20	445.16	480.12	515.08	550.04
4	481.79	490.82	499.85	508.88	563.07	608.23	653.38	698.54	743.70	788.85	834.01	879.17
5	605.04	616.85	628.67	640.49	711.40	770.49	829.58	888.67	947.76	1006.85	1065.94	1125.03
6	668.95	675.13	681.31	687.49	724.56	755.45	786.34	817.23	848.13	879.02	909.91	940.80
7	985.23	998.18	1011.14	1024.09	1101.83	1166.61	1231.39	1296.16	1360.94	1425.72	1490.50	1555.28
8	791.76	801.84	811.92	822.00	882.48	932.87	983.27	1033.67	1084.07	1134.46	1184.86	1235.26
9	856.20	870.14	884.09	898.03	981.67	1051.38	1121.08	1190.79	1260.50	1330.20	1395.18	1457.79
10	1038.78	1054.30	1069.82	1085.34	1178.46	1256.06	1333.67	1391.07	1442.61	1494.15	1545.70	1597.24
11	839.50	849.18	858.86	868.54	926.63	975.03	1023.44	1071.84	1120.25	1168.65	1217.06	1265.47
12	477.59	486.91	496.23	505.55	561.45	608.04	654.63	701.21	747.80	794.39	840.97	887.56
13	663.64	675.10	686.56	698.02	766.77	824.07	881.37	938.67	995.97	1053.27	1110.57	1167.87
14	1101.76	1114.14	1126.52	1138.89	1213.15	1275.03	1336.91	1398.79	1460.67	1522.55	1584.43	1646.32
15	**1389.6**†	**1406.7**†	**1423.7**†	**1438.6***	**1518.9***	**1585.9***	**1652.8***	**1719.8***	**1786.8***	**1853.7***	**1919.7**‡	**1973.6**‡
16	1092.71	1104.08	1115.45	1126.81	1195.01	1251.85	1308.68	1365.52	1422.35	1479.19	1536.02	1592.86
17	1210.68	1224.32	1237.96	1251.60	1333.46	1401.67	1469.88	1538.09	1606.30	1674.51	1742.73	1810.94
18	1106.08	1120.71	1135.35	1149.98	1237.79	1310.97	1384.15	1457.32	1530.50	1603.67	1676.85	1750.02
19	1342.62	1358.19	1373.76	1389.32	1482.73	1560.57	1638.42	1716.26	1794.10	1871.94	1949.78	2027.62
20	1352.24	1367.79	1383.33	1398.88	1492.15	1569.88	1647.61	1725.34	1803.07	1880.80	1958.52	2036.25
21	1087.27	1101.95	1116.63	1131.31	1219.37	1292.76	1366.15	1439.54	1512.92	1586.31	1659.70	1733.09
22	1209.94	1226.32	1242.70	1259.08	1357.36	1439.27	1521.17	1603.08	1684.98	1766.88	1848.79	1930.69
23	1030.03	1044.92	1059.81	1074.70	1164.06	1238.52	1312.99	1387.45	1461.91	1536.38	1610.84	1685.30
24	908.44	921.00	933.55	946.10	1021.42	1084.18	1146.95	1209.71	1272.47	1335.24	1398.00	1460.77

TABLE V

TEST NETWORK SUMMARY

Test network	Number of links(m)	Number of nodes(n)
Anaheim	914	416
Chicago sketch	2950	933
Barcelona	2522	1020

TABLE VI

EXPERIMENT PARAMETERS

Parameter	Testing values	Parameter description
κ	{0.15, 0.25, 0.5}	maximum of coefficient deviation
ζ	{1.0, 10, 30}	reliability coefficient

The mean travel time data comes from transportation test networks [46]. However, the minimal mean-std RSP planning problem requires the standard deviation of the link travel times as inputs so as to capture the path's travel time variability. We follow the works in [27] and [37] to generate the standard deviations of each link, the same as what we have done for the Sioux Falls network. We sample 50 travel time data for each link out of the corresponding distribution, and then use the sampled travel time data to estimate the covariance matrix $\mathbf{\Sigma}$. Table VI presents the parameter values adopted in the experiments.[6]

Table VII presents comparative results of GP3, SG, SP and LR across different ζ values, different κ values and different networks. In the table, firstly, we can see that across all the three networks and for different ζ and κ values, GP3 required only 1 full iteration to reach the RSP solution, with relative gap less than 5%. The underlying reason is that the LET path's variance is not far from that of the minimal variance path, which makes the initial gap, *i.e.* $t_{\max} - t_{\min}$, very small, and 1 full iteration of GP3 computation is able to return a 'qualified' RSP, *i.e.* the RSP with a relative gap less than η_r.

Secondly, for the same network, the required computation time does not change much when we change the ζ values or the κ values, but when the network size increases, *i.e.* the number of links increases, the average computation time increases. This is because the core computation load of GP3 is spent on computing the relaxed convex QP problem, which inherently depends on the number of the links in the network, *i.e.* the computational complexity of the QP solver is $\mathcal{O}(m^3)$, where m is the number of links in the network. Finally, when GP3 is compared with SG, SP, LR, on average, it returns the RSP with a smaller relative gap, but admittedly requires more computation time. We wish to point out that GP3's average computation time is more or less on the same scale of SG's, SP's and LR's computation time. As it can be seen in Table VII, the maximal average computation time of GP3 is less than 7.04 seconds, and according to the survey work in [47], most users would bear 30 seconds for the planner to give him/her a good reliable shortest path. Hence, we deem the GP3 algorithm as a time-wise acceptable navigation algorithm for a priori path computation.

Additionally, we wish to note that due to some unknown but recurring reasons, SP's average number of iterations increases significantly for the Barcelona network when $\zeta = \{10, 30\}$, *i.e.*, from around 2 iterations to more than 60 iterations, and the corresponding computation time increases significantly as well. We repeat the experiment for 5 independent times, and SP's related performance (number of iterations and computation time) stays more or less the same.

[6] Note that we select three different ζ values (1, 10, 30) for the evaluation, as too large reliability coefficients *i.e.* $\zeta \geq 30$ will stick the RSP with the minimal variance path, and too small reliability coefficients will yield LET RSP solutions.

TABLE VII

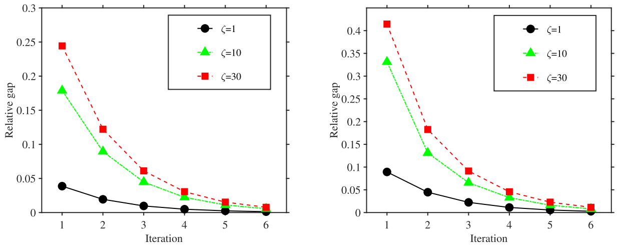
PERFORMANCE COMPARISON OF GP3, SG, SP AND LR FOR DIFFERENT ζ AND κ VALUES OVER THE THREE NETWORKS

	ζ	κ	Average relative gap				Average number of iterations				Average computation time(s)			
			GP3	SG	SP	LR	GP3	SG	SP	LR	GP3	SG	SP	LR
Anaheim	1	0.15	0.005	0.052	0.004	0.004	1.00	138.64	2.00	2.00	2.18	1.23	0.02	0.03
		0.25	0.007	0.065	0.005	0.004	1.00	149.02	2.00	2.00	2.08	0.96	0.02	0.03
		0.5	0.014	0.053	0.009	0.008	1.00	147.23	2.00	2.00	2.00	1.22	0.02	0.03
	10	0.15	0.012	0.048	0.033	0.040	1.00	132.09	2.19	31.45	1.98	0.93	0.03	0.36
		0.25	0.007	0.052	0.032	0.036	1.00	169.97	2.13	21.32	2.19	0.76	0.03	0.26
		0.5	0.009	0.052	0.030	0.052	1.00	173.39	2.81	101.19	2.38	1.21	0.04	1.30
	30	0.15	0.011	0.053	0.034	0.059	1.00	188.07	3.37	138.13	1.87	0.92	0.03	1.46
		0.25	0.026	0.053	0.033	0.066	1.00	188.06	3.35	141.15	1.98	1.15	0.03	1.51
		0.5	0.017	0.054	0.034	0.063	1.01	184.07	5.53	135.53	1.97	0.91	0.16	1.49
Chicago Sketch	1	0.15	0.005	0.048	0.010	0.071	1.00	146.81	2.00	102.98	6.99	7.62	0.19	7.48
		0.25	0.008	0.054	0.011	0.245	1.00	169.40	2.00	157.50	7.04	6.50	0.18	11.23
		0.5	0.001	0.058	0.010	0.053	1.00	188.74	2.00	91.10	6.63	6.90	0.18	6.45
	10	0.15	0.008	0.047	0.010	0.055	1.00	96.52	2.00	163.88	6.66	6.92	0.18	6.82
		0.25	0.008	0.053	0.022	0.351	1.00	166.23	2.16	188.91	6.60	4.89	0.19	13.36
		0.5	0.014	0.067	0.019	0.314	1.00	185.29	2.09	184.16	6.62	5.00	0.18	12.95
	30	0.15	0.002	0.048	0.011	0.048	1.00	106.47	2.00	93.08	6.51	7.67	0.19	6.77
		0.25	0.003	0.054	0.011	0.040	1.00	110.45	2.00	47.54	5.64	6.81	0.16	3.29
		0.5	0.015	0.044	0.016	0.051	1.00	58.86	2.02	197.76	6.69	5.53	0.17	13.77
Barcelona	1	0.15	0.009	0.053	0.026	0.012	1.00	145.65	2.00	168.32	4.30	5.36	0.12	9.06
		0.25	0.007	0.052	0.039	0.017	1.00	176.76	2.00	152.48	4.16	5.01	0.12	8.19
		0.5	0.010	0.065	0.034	0.098	1.00	189.61	2.00	150.50	3.86	4.91	0.12	8.06
	10	0.15	0.013	0.063	0.019	0.114	1.00	165.44	63.57	165.63	4.02	5.53	21.81	8.84
		0.25	0.006	0.069	0.019	0.043	1.00	173.21	64.08	157.56	3.94	5.40	21.95	8.43
		0.5	0.004	0.056	0.039	0.145	1.00	178.45	62.87	151.84	3.85	5.25	21.47	8.13
	30	0.15	0.008	0.034	0.046	0.064	1.00	142.29	56.06	173.09	4.27	5.63	18.98	9.21
		0.25	0.043	0.063	0.058	0.109	1.00	116.42	61.62	155.09	3.91	4.99	21.19	8.29
		0.5	0.056	0.057	0.039	0.112	1.00	146.27	69.63	169.90	4.20	5.79	24.13	9.14

D. Large Scale Network Test With Real Traffic Data: Chengdu

The experiments in the previous subsection evaluate the algorithms with real mean travel time and real transportation network topology, however, the covariance matrix is generated according to the method proposed in [27] and [37]. In this subsection, we will test the performance of GP3 and other state-of-the-art algorithms in a large scale network (Chengdu Network) with real traffic data. The real world network consists of 1902 nodes and 5943 links in Chengdu city, China. Travel speed samples of the links are measured through loop detectors, which, together with the link length, are provided in [48].

Since the travel time distributions are different in different times slots of the day, *e.g.* peak hour, non-peak hour, and across different types of the day, *e.g.* weekday, weekend, we separate the real traffic data into four different groups, namely, weekday peak hour,[7] weekday non-peak hour, weekend peak hour, weekend non-peak hour. In the following, we will evaluate (1) the convergence of GP3, and the number of required iterations of GP3 VS. the shortest distance between the origin node and the destination node, (2) how changing the reliability coefficient ζ affects the path selection, and (3) how GP3 performs against different types of time slots with real data, *i.e.* weekend, weekday, peak hour, non-peak hour. Note that, when we evaluate (1), we randomly sample the OD pairs in Chengdu, and also pick the time slots randomly out of the four groups. When we evaluate (2), we focus on the weekday peak hour time slot, as we find that when the traffic demand is high, *i.e.* weekday peak hour, there is a large variety of reliable paths for user's selection.



(a) OD distance $2km - 5km$ (b) OD distance $\geq 5km$

Fig. 5. Average relative gap evolution process for different ζ values in Chengdu network.

To verify the convergence of the GP3 algorithm, 100 random origin-destination (OD) pairs are extracted from the network for testing. The extracted OD pairs must satisfy that the shortest distance between the origin node and the destination node is larger than $2km$, as too small an OD distance cannot offer enough number of diversified paths for selection. Fig. 5 shows that the average relative gap between the upper and lower bounds decreases as the number of iterations increases. It is found that $4 - 5$ iterations is sufficient for the algorithm to converge to a relatively small gap (5%) under different reliability coefficients.

Table VIII shows the relationship between the required number of iterations of GP3 VS. the distance between the origin-destination pair. We find that, in general, when we increase the distance of the OD pair, the required number of

[7] peak hours are defined as the following two time slots: (1) between 7:30am to 9:30am, and (2) between 5:00pm to 7:00pm.

TABLE VIII
PERFORMANCE WITH DIFFERENT OD PAIRS

OD distance	$2km - 5km$	$5km - 10km$	$\geq 10km$
Initial relative gap	0.1577	0.3362	0.5721
Average iteration	1.76	4.34	5.81

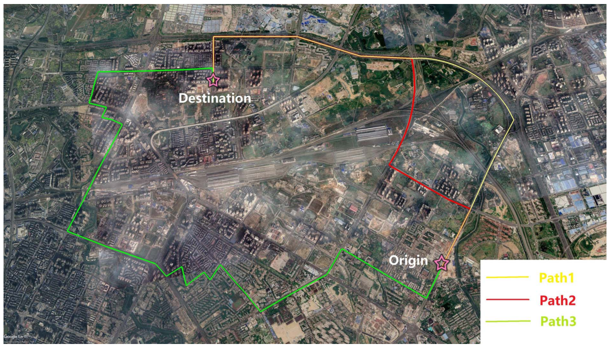


Fig. 6. Three different RSPs for different ζ values (the same OD pair) in Chengdu network (Best viewed in color).

TABLE IX
PROPERTIES OF THE THREE DIFFERENT RSPS

Path	Path1	Path2	Path3
Expected travel times(s)	757.68	817.74	1396.9
Variance of travel times(s^2)	51191	22443	12938
Path length(m)	8560.6	8319.6	12285
Corresponding ζ ranges	<0.5	0.5-16	>16

iterations of GP3 increases. Moreover, another quite indicative index is the initial relative gap, *i.e.* $(t_{\max} - t_{\min})/t_{\max}$ in the algorithm. When the initial relative gap is small, the required number of iterations is small.

Fig. 6 shows three different RSPs for the same OD pair with different reliability coefficients. When we increase the reliability coefficients (ζ) from 0 to ∞, the path gradually shifts from the LET path to the minimal variance path.

Table IX shows the three different RSPs' mean travel time, variance and the total length, and the corresponding ζ ranges.[8]

We further separate the transportation data into four categories, namely, workday peak hour data, workday non-peak data, holiday peak hour data, and holiday non-peak data, and test the performance of the algorithms (GP3, SG, SP, LR) for all the four categories. Table X shows the comparative performance of the four algorithms for different categories and different reliability coefficients. We are keen on the algorithm's (1) convergence; (2) convergence rate/speed; and (3) the quality of the final solution. For (1), we show the final relative gap; for (2) we show the average computation time, and for (3), we introduce a new concept: optimality index, which is defined as:

$$\text{Opt} = \frac{f(\boldsymbol{x}_{\text{alg}}) - f(\boldsymbol{x}_{\text{LET}})}{f(\boldsymbol{x}_{\text{LET}})}, \tag{21}$$

where $\boldsymbol{x}_{\text{alg}}$ refers to the path given by the evaluation algorithm (*i.e.* GP3, SG, SP, LR), and $\boldsymbol{x}_{\text{LET}}$ is the LET path. $f(\boldsymbol{x})$ is the evaluating path's RSP metric, which is calculated as: $f(\boldsymbol{x}) = \boldsymbol{\mu}^\top \boldsymbol{x} + \zeta\sqrt{\boldsymbol{x}^\top \boldsymbol{\Sigma} \boldsymbol{x}}$. A negative Opt value indicates that the referred RSP solution is better than that of the LET path. We wish to note that the average relative gap in Table X indicates the theoretical performance (*i.e.* the worst case performance index) of the corresponding algorithm, while the optimality index value shows the actual performance of the algorithm when compared with the LET path. Both are important indicators of the underlying RSP planning algorithm.

From Table X, we can see that all of the four RSP planning algorithms are behaving better than the LET path, *i.e.* exhibiting a negative Opt value, and the average relative gaps are near 0.05, which is the predefined threshold value. Additionally, we have the following observations and related explanations. (1) When we increase ζ from 1 to 30, the average computation time increases for all the four algorithms, but GP3's computation time stays at a stable range. The underlying reason is that when we increase ζ values, the initial gap between $t_{\max}$ and $t_{\min}$ increases significantly, and thus all the algorithms need more iterations to converge. But GP3 is able to shrink the size of the gap by at least a half within each iteration, thus the number of iterations, and hence the computation time does not increase significantly.

(2) When $\zeta = 1$, except for the period of workday peak hour, all the other three periods will have the LET path as the RSP solution, which makes all the algorithms' optimality index stay at 0. This is because with small ζ values, the optimal RSP favors greatly the LET path. But during workday peak hour, the travel time variance is so large across the transportation networks that sometimes, we can find an RSP which is not the LET path.

(3) GP3 is not always the best algorithm across all the scenarios, in terms of the optimality index value. For example, with 200 iterations, both SG and SP perform better than GP3 in the period of holiday peak hour and $\zeta = 10$. We perform the paired significant test between GP3 and SG, and find that for most of the periods of analysis, the differences in optimality index between GP3 and SG are statistically *insignificant*. It means that both GP3 and SG are behaving equally well in this use case. This is because throughout the experiments, η_r is set to be 0.05 for all the algorithms, which means that when the relative gap is less than 0.05, the corresponding algorithm terminates and returns the current RSP. However, GP3 has a theoretical guarantee that the relative gap between GP3's output and the optimal path is less than 5%. This kind of theoretical guarantee is not available for SG, SP and LR. If we need GP3 to perform always the best out of the four algorithms, we may just decrease η_r values, and GP3 will have an improved optimality index value; while SG, SP and LR would not be able to improve much because experimentally, they cannot even reach a $\eta_r = 0.05$ relative gap threshold within 200 iterations.

E. Discussions

We have presented the performance of GP3 against that of the state-of-the-art algorithms for the mean-std shortest path problem. As shown in the four previous subsections, GP3 is able to converge to the optimal solution with arbitrarily

[8] Note that the phase 'corresponding ζ ranges' means that when ζ falls into the specified range, the corresponding path is the resulting optimal RSP.

TABLE X

PERFORMANCE COMPARISON OF GP3, SG, SP AND LR FOR DIFFERENT TIME SLOTS AND ζ VALUES OVER CHENGDU NETWORK

Period of analysis	ζ value	Average relative gap				Average computation time (s)				Average optimality index (Opt)			
		GP3	SG	SP	LR	GP3	SG	SP	LR	GP3	SG	SP	LR
Workday (Off-peak hour)	1	0.042	0.053	0.050	0.051	41.12	27.42	12.15	13.22	0	0	0	0
	10	0.041	0.050	0.051	0.029	45.53	30.15	20.73	17.78	-0.416	**-0.433**	-0.308	-0.311
	30	0.042	0.058	0.054	0.042	44.62	37.68	24.68	17.91	-0.697	**-0.734**	-0.479	-0.441
Workday (Peak hour)	1	0.043	0.051	0.047	0.052	43.74	28.76	11.95	19.61	**-0.160**	-0.130	-0.130	0
	10	0.043	0.057	0.052	0.047	43.33	31.65	20.74	16.42	-0.343	-0.301	**-0.411**	**-0.411**
	30	0.026	0.082	0.053	0.052	44.93	37.82	22.91	21.12	**-0.414**	-0.324	-0.258	0
Holiday (Off-peak hour)	1	0.003	0.050	0.032	0.049	40.27	15.28	11.45	18.69	0	0	0	0
	10	0.004	0.054	0.053	0.050	43.16	27.06	16.06	19.68	**-0.445**	-0.421	-0.341	-0.169
	30	0.048	0.057	0.054	0.054	42.79	31.56	17.56	21.25	-0.079	**-0.096**	-0.059	0
Holiday (Peak hour)	1	0.045	0.078	0.021	0.044	41.24	22.65	10.85	11.82	0	0	0	0
	10	0.030	0.057	0.052	0.038	43.90	31.73	17.19	17.88	-0.727	**-0.757**	-0.738	-0.683
	30	0.027	0.071	0.052	0.052	44.01	33.65	19.82	21.48	-0.542	**-0.575**	-0.439	-0.184

high accuracy, and the computation time is guaranteed to be polynomial with respect to the size (number of nodes: m) of the network. During the experiments, we do observe that, when deployed to the realistic network (Anaheim, Chicago Sketch and Barcelona) and Chengdu, GP3's computation time is longer than that of SG, SP and LR. The underlying reason is that GP3 calls up the QP solver to solve the core problem which has a computational complexity of $\mathcal{O}(m^3)$, while SG, SP and LR iteratively use Dijkstra's algorithm as the underlying solver, whose computational complexity is $\mathcal{O}(m+n\log n)$.

On the one hand, this leaves room for GP3's improvement in the future. On the other hand, we observe that in real use cases, when ζ is large, *i.e.* 30 in the experiments, GP3's computation time is able to stay at the same level as that of the state of the arts. It means that if we are targeting at a mean-std shortest path problem with larger ζ values, GP3 achieves the best solution with more or less the same computation time as the state of the arts. While if we target at the mean-std shortest path problem with small ζ values, LET path can already deliver a good enough solution as shown in Table X. In this way, we can come up with a simple blending strategy of LET and GP3 for the mean-std shortest path problem, with short computation time and high accuracy.

VII. CONCLUSION AND FUTURE WORKS

This paper proposes GP3, a polynomial computational complexity algorithm for accurate RSP solutions in GP-regulated transportation networks. Through a series of problem transformations, the original mean-std problem is finally translated to a convex QIP problem, and solved by an efficient algorithm. Computational complexity analysis shows that GP3 is a polynomial complexity algorithm and it guarantees to return the RSP solution with arbitrarily high accuracy. We performed extensive experiments over various sizes of transportation network, from self-constructed simple networks, to canonical transportation test networks (Sioux Falls network), to realistic networks (Anaheim, Chicago, Barcelona and Chengdu), performance comparison indicates that (1) GP3 is able to return an RSP solution with guaranteed relative gap within several iterations; (2) admittedly, GP3's required computation time is higher than that of selected state-of-the-art algorithms, but they are, more or less, at the same scale.

In the future, we plan to make use of GP's posterior inference ability to estimate/predict the 'unknown' links' travel time distribution. Here, 'unknown' links refer to the links without or with only a few real measurement samples. In this way, we are able to perform RSP planning with only a subset of samples from the underlying transportation network. We are also planning to embed GP3 into a real navigation system, and speed up GP3's core computation (the QIP solver) so that it meets users' real time path planning requirement. Another future work direction is to relax GP3's Gaussian process assumption of the underlying transportation networks, and extend its application to transportation networks with non-Gaussian distribution assumptions, *e.g.* Levy distribution, Weibull distribution, log-normal distribution.

APPENDIX

In the appendix, we deliver the proofs of the equivalence relationship among the minimal mean-std RSP, α-reliable path, and the SOTA path, by introducing the following two theorems and the respective proofs.

Theorem 3: In GP-regulated transportation networks, the 'risk-averse' α-reliable RSP ($\alpha \geq 0.5$) is equivalent to a mean-std RSP, with $\zeta = \Psi^{-1}(\alpha)$, where $\Psi(x)$ is the cumulative distribution function (CDF) of the standard normal distribution (a normal distribution with mean $\mu = 0$ and standard deviation $\sigma = 1$).

Proof: The α-reliable RSP problem can be formulated as:

$$\begin{aligned} &\underset{\boldsymbol{x}, T_0}{\text{minimize}}\ T_0 \\ &\text{subject to}\ \text{Prob}(\boldsymbol{c}^\top \boldsymbol{x} \leq T_0) \geq \alpha \\ &\qquad\qquad \boldsymbol{A}\boldsymbol{x} = \boldsymbol{b} \\ &\qquad\qquad x_{ij} \in \{0, 1\}. \end{aligned} \tag{22}$$

Define $y = \boldsymbol{c}^\top \boldsymbol{x}$. Note that y is a linear transformation of $\boldsymbol{x}$. From GP properties, we know that $y \sim \mathcal{N}(\boldsymbol{\mu}^\top \boldsymbol{x}, \boldsymbol{x}^\top \Sigma \boldsymbol{x})$. Then, we have: $\text{Prob}(y \leq T_0) = \text{Prob}\big((y - \boldsymbol{\mu}^\top \boldsymbol{x})/\sqrt{\boldsymbol{x}^\top \boldsymbol{\Sigma} \boldsymbol{x}} \leq (T_0 - \boldsymbol{\mu}^\top \boldsymbol{x})/\sqrt{\boldsymbol{x}^\top \boldsymbol{\Sigma} \boldsymbol{x}}\big) = \Psi((T_0 - \boldsymbol{\mu}^\top \boldsymbol{x})/\sqrt{\boldsymbol{x}^\top \boldsymbol{\Sigma} \boldsymbol{x}})$.

The transformation process of the first constraint is performed as follows: $\Psi\big((T_0 - \boldsymbol{\mu}^\top \boldsymbol{x})/\sqrt{\boldsymbol{x}^\top \boldsymbol{\Sigma} \boldsymbol{x}}\big) \geq \alpha \Rightarrow (T_0 - \boldsymbol{\mu}^\top \boldsymbol{x})/\sqrt{\boldsymbol{x}^\top \boldsymbol{\Sigma} \boldsymbol{x}} \geq \Psi^{-1}(\alpha) \Rightarrow T_0 \geq \boldsymbol{\mu}^\top \boldsymbol{x} + \Psi^{-1}(\alpha)\sqrt{\boldsymbol{x}^\top \boldsymbol{\Sigma} \boldsymbol{x}}$. Unifying the transformed first constraint with the objective, the optimization problem is equivalent to the mean-std RSP problem with $\zeta = \Psi^{-1}(\alpha)$. □

Theorem 4: In GP-regulated transportation networks, the 'risk averse' SOTA RSP[9] *with deadline T is equivalent to a mean-std RSP, with a specific ζ, which returns the optimal path with the objective value equal to T, and the bisection algorithm can be used to identify ζ.*

Proof: The SOTA problem is formulated as follows:

$$\begin{aligned} &\underset{\boldsymbol{x},\alpha}{\text{maximize}}\ \alpha \\ &\text{subject to } \text{Prob}(\boldsymbol{c}^{\top}\boldsymbol{x} \le T_0) \ge \alpha \\ &\qquad\qquad \boldsymbol{A}\boldsymbol{x} = \boldsymbol{b} \\ &\qquad\qquad x_{ij} \in \{0, 1\}. \end{aligned} \tag{23}$$

According to Theorem 5 of [25], the SOTA RSP problem is equivalent to the α-reliable RSP problem with a specific α which makes the α-reliable RSP problem output the objective value at T_0. In the meanwhile, according to Theorem 3 in this paper, in GP-regulated environments, the α-reliable RSP problem is equivalent to the SOTA RSP problem with $\zeta = \Psi^{-1}(\alpha)$. Thus, we can conclude that the SOTA RSP problem with a predefined deadline (T_0) as the input is equivalent to the mean-std RSP problem with a specific ζ which makes its equivalent α-reliable RSP output an objective value of T_0. Moreover, when we increase ζ, it means that we are increasing α in the α-reliable RSP problem. As a result, T_0 will be increased. Therefore, the three involved parameters (ζ, α and T_0) are positively related to each other. We can use the bisection method [49] to identify ζ for a specific input of T_0 for the SOTA RSP problem. □

ACKNOWLEDGMENT

The authors would like to thank the reviewers for their thoughtful comments and efforts toward improving the manuscripts. Additionally, they would like to thank Chen Gao and Wenda Sheng for re-implementing GP3, SG, SP, and LR in Python, enabling all the algorithms to run on GPU, and making the related codes publicly available.

REFERENCES

[1] P. E. Hart, N. J. Nilsson, and B. Raphael, "A formal basis for the heuristic determination of minimum cost paths," *IEEE Trans. Syst. Sci. Cybern.*, vol. SSC-4, no. 2, pp. 100–107, Jul. 1968.

[2] W. Zeng and R. L. Church, "Finding shortest paths on real road networks: The case for A," *Int. J. Geograph. Inf. Sci.*, vol. 23, no. 4, pp. 531–543, Apr. 2009.

[3] J. Zhang, M. Liu, and B. Zhou, "Analytical model for travel time-based BPR function with demand fluctuation and capacity degradation," *Math. Problems Eng.*, vol. 2019, pp. 1–13, Nov. 2019.

[4] Y. Pan, L. Sun, and M. Ge, "Finding reliable shortest path in stochastic time-dependent network," *Procedia-Social Behav. Sci.*, vol. 96, pp. 451–460, Nov. 2013.

[5] H. Lee, S. Choi, H. Jung, B. B. Park, and S. H. Son, "A route guidance system considering travel time unreliability," *J. Intell. Transp. Syst.*, vol. 23, no. 3, pp. 282–299, May 2019.

[6] H. Huang and S. Gao, "Optimal paths in dynamic networks with dependent random link travel times," *Transp. Res. B, Methodol.*, vol. 46, no. 5, pp. 579–598, 2012.

[7] G. Sabran, S. Samaranayake, and A. Bayen, "Precomputation techniques for the stochastic on-time arrival problem," in *Proc. Meeting Algorithm Eng. Exp. (ALENEX)*. Philadelphia, PA, USA: SIAM, 2014, pp. 138–146.

[8] Z. Cao, H. Guo, J. Zhang, D. Niyato, and U. Fastenrath, "Finding the shortest path in stochastic vehicle routing: A cardinality minimization approach," *IEEE Trans. Intell. Transp. Syst.*, vol. 17, no. 6, pp. 1688–1702, Jun. 2015.

[9] B. Y. Chen, C. Shi, J. Zhang, W. H. K. Lam, Q. Li, and S. Xiang, "Most reliable path-finding algorithm for maximizing on-time arrival probability," *Transportmetrica B, Transp. Dyn.*, vol. 5, no. 3, pp. 248–264, Jul. 2017.

[10] Y. Liu, S. Blandin, and S. Samaranayake, "Stochastic on-time arrival problem in transit networks," *Transp. Res. B, Methodol.*, vol. 119, pp. 122–138, Jan. 2019.

[11] J. Zhou, X. Lai, and J. Y. J. Chow, "Multi-armed bandit on-time arrival algorithms for sequential reliable route selection under uncertainty," *Transp. Res. Record, J. Transp. Res. Board*, vol. 2673, no. 10, pp. 673–682, Oct. 2019.

[12] Z. Cao *et al.*, "Using reinforcement learning to minimize the probability of delay occurrence in transportation," *IEEE Trans. Veh. Technol.*, vol. 69, no. 3, pp. 2424–2436, Mar. 2020.

[13] P. Chen, R. Tong, G. Lu, and Y. Wang, "The α-reliable path problem in stochastic road networks with link correlations: A moment-matching-based path finding algorithm," *Expert Syst. Appl.*, vol. 110, pp. 20–32, Nov. 2018.

[14] A. Chen and Z. Ji, "Path finding under uncertainty," *J. Adv. Transp.*, vol. 39, no. 1, pp. 19–37, 2005.

[15] W. Zeng, T. Miwa, Y. Wakita, and T. Morikawa, "Application of Lagrangian relaxation approach to α-reliable path finding in stochastic networks with correlated link travel times," *Transp. Res. C, Emerg. Technol.*, vol. 56, pp. 309–334, Jul. 2015.

[16] S. Sen, R. Pillai, S. Joshi, and A. K. Rathi, "A mean-variance model for route guidance in advanced traveler information systems," *Transp. Sci.*, vol. 35, no. 1, pp. 37–49, Feb. 2001.

[17] H. Hu and R. Sotirov, "On solving the quadratic shortest path problem," *Informs J. Comput.*, vol. 32, no. 2, pp. 219–233, Nov. 2019.

[18] B. Y. Chen, W. H. K. Lam, and Q. Li, "Efficient solution algorithm for finding spatially dependent reliable shortest path in road networks," *J. Adv. Transp.*, vol. 50, no. 7, pp. 1413–1431, Nov. 2016.

[19] L. Shen, H. Shao, T. Wu, W. H. K. Lam, and E. C. Zhu, "An energy-efficient reliable path finding algorithm for stochastic road networks with electric vehicles," *Transp. Res. C, Emerg. Technol.*, vol. 102, pp. 450–473, May 2019.

[20] A. Khani and S. D. Boyles, "An exact algorithm for the mean–standard deviation shortest path problem," *Transp. Res. B, Methodol.*, vol. 81, pp. 252–266, Nov. 2015.

[21] Y. Zhang, Z.-J. M. Shen, and S. Song, "Lagrangian relaxation for the reliable shortest path problem with correlated link travel times," *Transp. Res. B, Methodol.*, vol. 104, pp. 501–521, Oct. 2017.

[22] A. Flajolet, S. Blandin, and P. Jaillet, "Robust adaptive routing under uncertainty," *Oper. Res.*, vol. 66, no. 1, pp. 210–229, Feb. 2018.

[23] Y. Fan, R. Kalaba, and J. Moore, "Arriving on time," *J. Optim. Theory Appl.*, vol. 127, no. 3, pp. 497–513, 2005.

[24] S. Lim, C. Sommer, E. Nikolova, and D. Rus, "Practical route planning under delay uncertainty: Stochastic shortest path queries," in *Robotics: Science and Systems*. 2012, pp. 249–256.

[25] Y. Gao, "Shortest path problem with uncertain arc lengths," *Comput. Math. Appl.*, vol. 62, no. 6, pp. 2591–2600, 2011.

[26] T. Xing and X. Zhou, "Finding the most reliable path with and without link travel time correlation: A Lagrangian substitution based approach," *Transp. Res. B, Methodol.*, vol. 45, no. 10, pp. 1660–1679, Dec. 2011.

[27] Y. Zhang and A. Khani, "An algorithm for reliable shortest path problem with travel time correlations," *Transp. Res. B, Methodol.*, vol. 121, pp. 92–113, Mar. 2019.

[28] H. A. Rakha, I. EL-Shawarby, M. Arafeh, and F. Dion, "Estimating path travel-time reliability," in *Proc. IEEE Intell. Transp. Syst. Conf.*, Sep. 2006, pp. 236–241.

[29] A. Eiger, P. B. Mirchandani, and H. Soroush, "Path preferences and optimal paths in probabilistic networks," *Transp. Sci.*, vol. 19, no. 1, pp. 75–84, Feb. 1985.

[30] H. Frank, "Shortest paths in probabilistic graphs," *Oper. Res.*, vol. 17, no. 4, pp. 583–599, Aug. 1969.

[9] The 'risk averse' SOTA RSP refers to the SOTA problem with a deadline T larger than the LET path's expected travel time.

[31] Y. Nie and Y. Fan, "Arriving-on-time problem: Discrete algorithm that ensures convergence," *Transp. Res. Rec., J. Transp. Res. Board*, vol. 1964, no. 1, pp. 193–200, Jan. 2006.

[32] Y. M. Nie and X. Wu, "Shortest path problem considering on-time arrival probability," *Transp. Res. B, Methodol.*, vol. 43, no. 6, pp. 597–613, 2009.

[33] R. A. Sivakumar and R. Batta, "The variance-constrained shortest path problem," *Transp. Sci.*, vol. 28, no. 4, pp. 309–316, Nov. 1994.

[34] H. Hu and R. Sotirov, "Special cases of the quadratic shortest path problem," *J. Combinat. Optim.*, vol. 35, no. 3, pp. 754–777, Apr. 2018.

[35] Q. Tu, L. Cheng, T. Yuan, Y. Cheng, and M. Li, "The constrained reliable shortest path problem for electric vehicles in the urban transportation network," *J. Cleaner Prod.*, vol. 261, Jul. 2020, Art. no. 121130.

[36] L. Yang and X. Zhou, "Optimizing on-time arrival probability and percentile travel time for elementary path finding in time-dependent transportation networks: Linear mixed integer programming reformulations," *Transp. Res. B, Methodol.*, vol. 96, pp. 68–91, Feb. 2017.

[37] M. Shahabi, A. Unnikrishnan, and S. D. Boyles, "Robust optimization strategy for the shortest path problem under uncertain link travel cost distribution," *Comput.-Aided Civil Infrastruct. Eng.*, vol. 30, no. 6, pp. 433–448, Jun. 2015.

[38] Z. Cao, H. Guo, J. Zhang, D. Niyato, and U. Fastenrath, "Improving the efficiency of stochastic vehicle routing: A partial Lagrange multiplier method," *IEEE Trans. Veh. Technol.*, vol. 65, no. 6, pp. 3993–4005, Jun. 2016.

[39] Y. Pan, "Lagrangian relaxation for the multiple constrained robust shortest path problem," *Math. Problems Eng.*, vol. 2019, pp. 1–13, Jun. 2019.

[40] C. E. Rasmussen and C. K. I. Williams, *Gaussian Processes for Machine Learning* (Adaptive Computation and Machine Learning). Cambridge, MA, USA: MIT Press, 2005.

[41] H. Coxeter, *Regular Polytopes*. New York, NY, USA: Dover, 1973.

[42] S. Boyd and L. Vandenberghe, *Convex Optimization*. Cambridge, U.K.: Cambridge Univ. Press, 2004.

[43] R. Aharoni, E. Berger, A. Georgakopoulos, A. Perlstein, and P. Sprüssel, "The max-flow min-cut theorem for countable networks," *J. Combinat. Theory, B*, vol. 101, no. 1, pp. 1–17, 2009.

[44] G. Dantzig, R. Fulkerson, and S. Johnson, "Solution of a large-scale traveling-salesman problem," *J. Oper. Res. Soc. Amer.*, vol. 2, no. 4, pp. 393–410, 1954.

[45] M. Schubiger, G. Banjac, and J. Lygeros, "GPU acceleration of ADMM for large-scale quadratic programming," *J. Parallel Distrib. Comput.*, vol. 144, pp. 55–67, Oct. 2020.

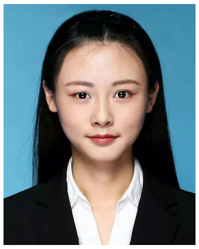
[46] *Transportation Networks for Research Core Team*. Accessed: Aug. 14, 2021. [Online]. Available: https://github.com/bstabler/TransportationNetworks

[47] R. K. Kamalanathsharma, H. A. Rakha, and I. H. Zohdy, "Survey on in-vehicle technology use: Results and findings," *Int. J. Transp. Sci. Technol.*, vol. 4, no. 2, pp. 135–149, Jun. 2015.

[48] F. Guo, D. Zhang, Y. Dong, and Z. Guo, "Urban link travel speed dataset from a megacity road network," *Sci. Data*, vol. 6, no. 1, pp. 1–8, May 2019.

[49] E. Süli and D. F. Mayers, *An Introduction to Numerical Analysis*. Cambridge, U.K.: Cambridge Univ. Press, 2003.

Hongliang Guo received the B.E. degree in dynamic engineering and the M.E. degree in dynamic control from Beijing Institute of Technology, China, in 2001 and 2005, respectively, and the Ph.D. degree in electrical and computer engineering from Stevens Institute of Technology, USA. From 2016 to 2020, he served as an Associate Professor for the University of Electronic Science and Technology of China. In 2021, he joins the Institute for Infocomm Research (I2R), A*STAR as a Scientist. His research interests include planning and learning under uncertainties.



Xuejie Hou received the bachelor's degree in automation engineering from the University of Electronic Science and Technology of China in 2018, where she is currently pursuing the master's degree majoring in pattern recognition. Her research interests include path planning in uncertainties, off-policy reinforcement learning, and bandit learning.

Zhiguang Cao received the B.S. degree in automation from Guangdong University of Technology, China, in 2009, the M.Sc. degree from the School of Electrical and Electronic Engineering, Nanyang Technological University in 2012, and the Ph.D. degree from Interdisciplinary Graduate School & School of Computer Science and Engineering, Nanyang Technological University in 2016. He was a Research Assistant Professor with the Department of Industrial Systems Engineering and Management, National University of Singapore (NUS) and a Research Fellow with the BMW@NTU Future Mobility Lab, Nanyang Technological University (NTU). He is currently a Scientist with the Singapore Institute of Manufacturing Technology (SIMTech), Agency for Science Technology and Research (A*STAR). His research interests focus on AI and optimization for intelligent systems.

Jie Zhang received the Ph.D. degree from Cheriton School of Computer Science, University of Waterloo, Canada, in 2009. He is currently an Associate Professor with the School of Computer Engineering, Nanyang Technological University, Singapore. His research has been focused on the design of effective and robust intelligent software agents, through the modeling, such as trustworthiness and preferences and simulation of different agents in a wide range of environments, using AI techniques and multi-agent technologies.