Query-Decision Regression Between Shortest Path and Minimum Steiner Tree

Guangmo Tong(✉), Peng Zhao, and Mina Samizadeh

University of Delaware, Newark, USA
{amotong,pzhao,minasmz}@udel.edu

Abstract. Considering a graph with unknown weights, can we find the shortest path for a pair of nodes if we know the minimal Steiner trees associated with some subset of nodes? That is, with respect to a fixed latent decision-making system (e.g., a weighted graph), we seek to solve one optimization problem (e.g., the shortest path problem) by leveraging information associated with another optimization problem (e.g., the minimal Steiner tree problem). In this paper, we study such a prototype problem called *query-decision regression with task shifts*, focusing on the shortest path problem and the minimum Steiner tree problem. We provide theoretical insights regarding the design of realizable hypothesis spaces for building scoring models, and present two principled learning frameworks. Our experimental studies show that such problems can be solved to a decent extent with statistical significance.

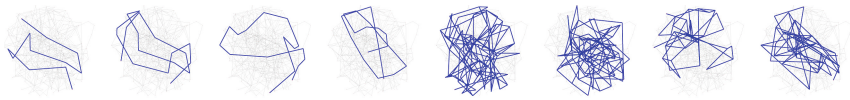
Keywords: Statistical Learning · Data-driven Optimization · Combinatorial Optimization

1 Introduction

In its most general sense, a decision-making problem seeks to find the best decision for an input query in terms of an objective function that quantifies the decision qualities [6]. Traditionally, the objective function is given a prior, and we thus focus primarily on its optimization hardness. However, real-world systems are often subject to uncertainties, making the latent objective function not completely known to us [11]; this creates room for data-driven approaches to play a key role in building decision-making pipelines [17].

Query-decision Regression with Task Shifts (QRTS). When facing an unknown objective function, one can adopt the learn-and-optimize framework where we first learn the unknown objective function from data and then solve the target optimization problem based on the learned function [3], which can be dated back to Bengio's work twenty years ago [2]. Nevertheless, the learn-and-optimize framework suffers from the fact that the learning process is often driven by the *average* accuracy while good optimization effects demand *worst-case* guarantees [7]. Query-decision regression (QR), as an alternative decision-making diagram, seeks to infer good decisions by learning directly from successful

D.-N. Yang et al. (Eds.): PAKDD 2024, LNAI 14646, pp. 314–326, 2024.
https://doi.org/10.1007/978-981-97-2253-2_25



(a) (b) (c) (d) (e) (f) (g) (h)

Fig. 1. Associated with a fixed weighted graph, (a)–(d) show the shortest paths of four pairs of nodes, and (e)–(h) show the minimal Steiner trees for four node subsets. Can we leverage the information in (a)–(d) to compute the solutions in (e)–(h), or vice versa?

optimization results. The feasibility of such a diagram has been proved by a few existing works [4]. Such a success points out an interesting way of generalizing QR called query-decision regression with task shifts (QRTS): assuming that there are two query-decision tasks associated with a latent system, can we solve one task (i.e., the target task) by using optimization results associated with the other task (i.e., the source task) – Fig. 1? Proving the feasibility of such problems is theoretically appealing, as it suggests that one can translate the optimal solutions between different optimization problems.

Contribution. This paper presents the first study on QRTS over stochastic graphs for two specific problems, the shortest path problem and the minimum Steiner tree problem. Taking QRTS as a statistical learning problem, we provide theoretical analysis regarding the creation of realizable hypothesis spaces for designing score functions, seeking to integrate the latent decision objective into the learning pipeline for better optimization effects. Based on the proposed hypothesis space, we design two principled methods QRTS-P and QRTS-D, where P stands for **p**oint estimation and D stands for **d**istribution learning. In particular, QRTS-P is designed based on the principle of point estimation that implicitly searches for the best mean graph, while QRTS-D leverages distribution learning to compute the pattern of the edge weights that can best fit the samples. We present empirical studies using graphs of classic families. As one of the main contributions of this paper, our results confirm that QRTS can be solved to a satisfactory extent, which may be the first piece of evidence showing that one can successfully translate knowledge between different optimization problems sharing the same underlying system. The appendix and supplementary materials[1] include technical proofs, more discussions on experiments, source code, and data.

2 Preliminaries

We consider a countable family $\mathcal{G}$ of weighted directed graphs sharing the same graph structure $G = (V, E)$. For each weighted graph $g \in \mathcal{G}$, let $g() : E \to \mathbb{R}^+$ be its weight function. Without loss of generality, we assume that G has no multiple edge when the edge directions are omitted; therefore, each graph $g \in \mathcal{G}$ can also

[1] https://github.com/cdslabamotong/QRTS.

be taken as an undirected graph, without causing confusion regarding the edge weights. Associated with $\mathcal{G}$, there is an *unknown* distribution $\mathcal{D}_{\mathcal{G}}$ over $\mathcal{G}$. We consider optimization problems in the following form.

Definition 1. (Query-decision Optimization). *Let $\mathcal{X} \subseteq 2^V$ be a query space and $\mathcal{Y} \subseteq 2^E$ be a decision space. In addition, let $f(x, y, g) \in \mathbb{R}^+$ be the decision value associated with a query $x \in \mathcal{X}$, a decision $y \in \mathcal{Y}$, and a graph $g \in \mathcal{G}$ (either directed or undirected). For a given query $x \in \mathcal{X}$, we seek to find the decision that can minimize the expected decision value:*

$$\arg\min_{y \in \mathcal{Y}} F_{f,\mathcal{D}_{\mathcal{G}}}(x, y) \text{ where } F_{f,\mathcal{D}_{\mathcal{G}}}(x, y) \coloneqq \mathbb{E}_{g \sim \mathcal{D}_{\mathcal{G}}}[f(x, y, g)]. \tag{1}$$

Such a task is specified by a three-tuple $(f, \mathcal{X}, \mathcal{Y})$. It reduces to the deterministic case with $|\mathcal{G}| = 1$ (e.g., Fig. 1).

When the distribution $\mathcal{D}_{\mathcal{G}}$ is known to us, the above problems fall into stochastic combinatorial optimization [18]. In addressing the case when $\mathcal{D}_{\mathcal{G}}$ is unknown, query-decision regression emphasizes the scenario when there is no proper data to learn $\mathcal{D}_{\mathcal{G}}$, and it is motivated by the aspiration to *learn directly from successful optimization results*. Since the optimization problem in question (i.e., Eq. (1)) can be computationally hard under common complexity assumptions (e.g., NP≠P), we assume that an approximate solution is observed. Accordingly, we will utilize samples in the form of

$$D_{f,\alpha} = \Big\{(x_j, y_j) \Big| F_{f,\mathcal{D}_{\mathcal{G}}}(x_j, y_j) \leq \alpha \cdot \min_{y \in \mathcal{Y}} F_{f,\mathcal{D}_{\mathcal{G}}}(x_j, y)\Big\}, \tag{2}$$

where the quality of the observed decision is controlled by a nominal ratio $\alpha \geq 1$. With such, we formulate query-decision regression with/without task shifts as statistical learning problems.

Definition 2. (Query-decision Regression (QR)). *Associated with a query-decision optimization problem $(f, \mathcal{X}, \mathcal{Y})$ and a distribution $\mathcal{D}_{\mathcal{X}}$ over $\mathcal{X}$, given a collection $D_{f,\alpha} = \{(x_j, y_j)\}$ of query-decision pairs with x_j being iid from $\mathcal{D}_{\mathcal{X}}$, we aim to learn a decision-making model $M : \mathcal{X} \to \mathcal{Y}$ that can predict high-quality decisions for future queries.*

Definition 3. (Query-decision Regression with Task Shifts (QRTS)). *Consider a source task $(f_S, \mathcal{X}_S, \mathcal{Y}_S)$ and a target task $(f_T, \mathcal{X}_T, \mathcal{Y}_T)$ sharing the same $\mathcal{G}$ and $\mathcal{D}_{\mathcal{G}}$. Suppose that we are provided with query-decision samples $D_{f_S,\alpha}$ associated with the source task. We aim to learn a decision-making model $M : \mathcal{X}_T \to \mathcal{Y}_T$ for the target task, with the goal of maximizing the optimization effect:*

$$\mathcal{L}(M, \mathcal{D}_{\mathcal{X}_T}) \coloneqq \mathbb{E}_{x \sim \mathcal{D}_{\mathcal{X}_T}}\Big[\frac{\min_{y \in \mathcal{Y}_T} F_{f_T,\mathcal{D}_{\mathcal{G}}}(x, y)}{F_{f_T,\mathcal{D}_{\mathcal{G}}}(x, M(x))}\Big], \tag{3}$$

where $\mathcal{D}_{\mathcal{X}_T}$ is the query distribution of the target task.

*Remark 1. (**Technical Challenge**).* In principle, QR falls into the setting of supervised learning, in the sense that it attempts to learn a mapping using labeled data. Therefore, standard supervised learning methods can solve such problems with statistical significance more or less, although they may not be the

optimal methods for specific tasks [21]. QRTS reduces to QR when the source task is identical to the target one, but it is arguably more challenging: one can no longer apply standard supervised learning methods because the query-decision mapping we seek to infer is associated with the target task but the samples are of the source task. To the best of our knowledge, no existing method can be directly applied to solve problems like the one in Fig. 1.

3 A Warm-Up Method: QRTS-P

In this section, we present a simple and intuitive method called QRTS-P for solving QRTS. To illustrate the idea, we notice that the latent objective function $F_{f,\mathcal{D}_\mathcal{G}}$ can be expressed as a function of the mean weights of the edges, which is due to the linearity of expectation.

Example 1. Suppose that the considered query-decision optimization problem is the stochastic shortest path problem. For each node pair $x = (u, v)$, let $\mathcal{Y}_x$ be the set of all paths from u to v. The latent objective function can thus be expressed as

$$F_{f,\mathcal{D}_\mathcal{G}}(x, y) = \begin{cases} \sum_{e\in y} \mathbb{E}_{g\sim\mathcal{D}_\mathcal{G}}[g(e)] & \text{if } y \in \mathcal{Y}_x \\ +\infty & \text{otherwise.} \end{cases} \tag{4}$$

Similarly, for the stochastic minimal Steiner tree problem [22], which finds the min-weight subgraph that connects a given set of nodes, we may define $\mathcal{Y}_x$ as the set of valid Steiner trees of a node set x, and with such, the latent objective has the identical form as Eq. (4).

In abstract, let $\mathcal{Y}_{f,x} \subseteq \mathcal{Y}_f$ be the set of the feasible solutions associated with a query x, and $\mathbb{1}_S \in \{0, 1\}$ be the set indicator function, i.e., $\mathbb{1}_S(x) = 1 \iff x \in S$. The query-decision optimization now has the generic form of

$$\arg\min_{y\in\mathcal{Y}_{f,x}} \sum\nolimits_{e\in y} \mathbb{E}_{g\sim\mathcal{D}_\mathcal{G}}[g(e)] = \sum\nolimits_{e\in E} \mathbb{E}_{g\sim\mathcal{D}_\mathcal{G}}[g(e)]\mathbb{1}_y(e) \tag{5}$$

Such an abstraction suggests that it would be sufficient for solving the target task if one can find the mean graph induced by $\mathcal{D}_\mathcal{G}$, which essentially asks for good estimations of $\{\mathbb{E}_{g\sim\mathcal{D}_\mathcal{G}}[g(e)]|e \in E\}$ – leading to a point estimation problem [13]. In what follows, we will see how samples $D_{f_S,\alpha}$ associated with the source task can be helpful for such a purpose.

For each $e \in E$, let $w_e \in \mathbb{R}^+$ be the sought-after estimation of $\mathbb{E}_{g\sim\mathcal{D}_\mathcal{G}}[g(e)]$. Since each sample (x_j, y_j) in $D_{f_S,\alpha}$ is an α-approximation, in light of Eq. (5), a desired set $\{w_e\} := \{w_e|e \in \mathbb{E}\}$ should satisfy the linear constraint

$$\alpha \cdot \min_{y\in\mathcal{Y}_{f_S,x_j}} \sum_{e\in E} w_e\mathbb{1}_y \geq \sum_{e\in E} w_e\mathbb{1}_{y_j}, \tag{6}$$

which means that the sample decision y_j is also an α-approximation in the mean graph induced by $\{w_e\}$. Applying the standard large-margin training to Eq.

(6) [19], a robust estimation can be inferred by solving the following quadratic program

$$\min_{w_e,\eta_j} \sum_{e\in E} w_e^2 + C\cdot\sum_j \eta_j \text{ s.t. } \alpha\cdot \underbrace{\min_{y\in\mathcal{Y}_{f_S,x_j}} \sum_{e\in E} w_e \mathbb{1}_y(e)}_{\textbf{source inference}} - \sum_{e\in E} w_e \mathbb{1}_{y_j}(e) \geq -\eta_j, \forall j. \quad (7)$$

where C is a hyperparamter. Optimization problems in the above form have been widely discussed for training structured prediction models, and they can be solved efficiently as long as the source inference problem in Eq. (7) can be effectively solved for a given $\{w_e\}$ [15]. We adopt the cutting plane algorithm in our experiments and defer the details to the Appendix. With the learned weights $\{w_e\}$, the inference for a query $x^* \in \mathcal{X}_T$ of the target problem can be computed through

$$\textbf{target inference}: \quad \min_{y\in\mathcal{Y}_{f_T,x^*}} \sum\nolimits_{e\in E} w_e \mathbb{1}_y(e) \quad (8)$$

We denote such an approach as QRTS-P. The source and target inferences will be discussed later in Remark 2, as they are special cases of later problems.

4 A Probabilistic Perspective: QRTS-D

In this section, we present a more involved method called QRTS-D for solving QRTS. It turns out that QRTS-D subtly subsumes QRTS-P as a special case.

4.1 Overall Framework

QRTS-D follows the standard scoring model, where we assign each decision a score and make a prediction by selecting the decision with *the lowest score*:

$$\begin{aligned} \textbf{score function}: &\quad h : \mathcal{X}_T \times \mathcal{Y}_T \to \mathbb{R} \\ \textbf{inference}: &\quad \arg\min_{y\in\mathcal{Y}_T} h(x,y). \end{aligned} \quad (9)$$

Such a framework is expected to solve QRTS well, provided that for each pair $(x,y) \in \mathcal{X}_T \times \mathcal{Y}_T$, a low score $h(x,y)$ can imply a small objective value $F_{f_T,\mathcal{D}_\mathcal{G}}(x,y)$. Implementing such an idea hinges on three integral parts: **a)** a hypothesis space H of h; **b)** training methods to search for the best score function within H based on the empirical evidence $D_{f_S,\alpha}$; **c)** algorithms for solving the inference problem. With such a framework, we will first discuss insights for designing a desired hypothesis space and then present training methods.

4.2 Hypothesis Design

In designing a desired score function h, the key observation is that the true objective function $F_{f_T,\mathcal{D}_\mathcal{G}}$ of the target task is a perfect score function, in that the inference over $F_{f_T,\mathcal{D}_\mathcal{G}}(x,y)$ recovers the exact optimal solution. While $\mathcal{D}_\mathcal{G}$ is unknown to us, the technique of importance sampling offers a means of deriving

a parameterized approximation [20]. In particular, for any empirical distribution $\mathcal{D}_{\mathcal{G}}^{em}$ over $\mathcal{G}$, we have

$$F_{f_T,\mathcal{D}_{\mathcal{G}}}(x,y) = \int_{g\in\mathcal{G}} \frac{\mathcal{D}_{\mathcal{G}}[g]}{\mathcal{D}_{\mathcal{G}}^{em}[g]} f_T(x,y,g) d\,\mathcal{D}_{\mathcal{G}}^{em}, \tag{10}$$

which immediately implies the function approximation guarantee between $F_{f_T,\mathcal{D}_{\mathcal{G}}}$ and an affine combination of f_T.

Theorem 1. *Let $\|\cdot\|$ denote the function distance with respect to the Lebesgue measure associated with any distribution $\mathcal{D}$ over $\mathcal{X}_T \times \mathcal{Y}_T$. For each $\epsilon \geq 0$, $\lim_{K\to\infty} \Pr_{g_i\sim\mathcal{D}_{\mathcal{G}}^{em}} \left[\inf_{w_i\in\mathbb{R}} \left\| F_{f_T,\mathcal{D}_{\mathcal{G}}}(x,y) - \sum_{i=1}^{K} w_i f_T(x,y,g_i) \right\| \leq \epsilon \right] = 1$.*

Theorem 1 justifies the following hypothesis space for the score function of which the complexity is controlled by its dimension $K \in \mathbb{Z}$.

$$H_{K,\mathcal{D}_{\mathcal{G}}^{em}} \coloneqq \left\{ h_{\boldsymbol{w},\{g_i\}}(x,y) \coloneqq \sum_{i=1}^{K} w_i f_T(x,y,g_i) \middle| g_i \sim \mathcal{D}_{\mathcal{G}}^{em}, \boldsymbol{w} = (w_1,...,w_K) \in \mathbb{R}^K \right\}.$$

These score functions are very reminiscent of the principled kernel machines [8], with the distinction that our kernel function, namely f_T, is inherited from the latent optimization problem rather than standard kernels [16]. For such a score function $h_{\boldsymbol{w},\{g_i\}}(x,y)$, the inference process is further specialized as

$$\textbf{target inference: } M_{\boldsymbol{w},\{g_i\}}(x) \coloneqq \min_{y\in\mathcal{Y}_T} \sum\nolimits_i w_i f_T(x,y,g_i). \tag{11}$$

With the construction of $H_{K,\mathcal{D}_{\mathcal{G}}^{em}}$, a realizable space can be achieved provided that the dimension K is sufficiently large, which allows us to characterize the generalization loss Eq. (3).

Theorem 2. *Suppose that a β-approximation is adopted to solve the target inference problem Eq. (11). Let $D_\infty \in \mathbb{R}$ be the ∞-order Rényi divergence between $\mathcal{D}_{\mathcal{G}}$ and $\mathcal{D}_{\mathcal{G}}^{em}$. For each $\epsilon \geq 0$ and $\delta > 0$, there exists*

$$C = O\Big(\frac{\ln|\mathcal{X}_T| + \ln|\mathcal{Y}_T|}{\epsilon^2} \cdot \ln\frac{1}{\delta} \cdot \exp(D_\infty) \Big)$$

such that when $K \geq C$, with probability at least $1-\delta$ over the selection of $\{g_i\}$, we have $\sup_{\boldsymbol{w}\in\mathbb{R}^K} \mathcal{L}(M_{\boldsymbol{w},\{g_i\}}, \mathcal{D}_{\mathcal{X}_T}) \geq \beta \cdot \frac{1-\epsilon}{1+\epsilon}$.

Theorem 2 suggests that a high dimension (i.e., K) may be needed when a) the spaces are large and/or b) the deviation between $\mathcal{D}_{\mathcal{G}}$ and $\mathcal{D}_{\mathcal{G}}^{em}$ is high, which is intuitive. The proof of Theorem 2 leverages point-wise concentration to acquire the desired guarantees, while Theorem 1 is proved through concentrations in function spaces.

4.3 QRTS-D

With the design of $H_{K,\mathcal{D}_{\mathcal{G}}^{em}}$, we now present methods for computing a concrete score function $h_{\boldsymbol{w},\{g_i\}}$, which is to decide a collection $\{g_i\}$ of subgraphs as well

as the associated weights $\boldsymbol{w}$. In light of Eq. (10), $\boldsymbol{w}_i$ can be viewed as the importance of graph g_i. We will not restrict ourselves to a specific choice of $\mathcal{D}_{\mathcal{G}}^{em}$ and thus assume that a nominal parametric family $\mathcal{D}_{\mathcal{G},\theta}^{em}$ is adopted, with an extra subscription θ added to denote the parameter set. Assuming that the hyperparameter K is given, the framework of QRTS-D loops over three phases: **a) graph sampling**, to sample $\{g_1, ..., g_K\}$ iid from $\mathcal{D}_{\mathcal{G},\theta}^{em}$; **b) importance learning**, to compute $\boldsymbol{w}$ for $\{g_i\}$; **c) distribution tuning**, to update $\mathcal{D}_{\mathcal{G},\theta}^{em}$. The first phase is trivial, and we will therefore focus on the other two phases.

Importance Learning. In computing the weights $\boldsymbol{w}$ for a given $\{g_i\}$, we have reached a key point to attack the challenges mentioned in Remark 1: the function approximation guarantee in Theorem 1 holds not only for the target task but also for the source task. That is, $\sum_{i=1}^{K} w_i f_T(x, y, g_i)$ is a desired score function (for solving the target task) if and only if $\sum_{i=1}^{K} w_i f_S(x, y, g_i)$ can well approximate the true objective function $F_{f_S, \mathcal{D}_{\mathcal{G}}}(x, y)$ of the source task. Therefore, since the samples in $D_{f_S,\alpha} = \{(x_j, y_j)\}$ are α-approximations to the source task, the ideal weights $\boldsymbol{w}$ should satisfy

$$\alpha \min_{y \in \mathcal{Y}_S} \sum\nolimits_{i=1}^{K} w_i f_S(x_j, y, g_i) \geq \sum\nolimits_{i=1}^{K} w_i f_S(x_j, y_j, g_i).$$

In this way, we have been able to leverage the samples from the source task to decide the best $\boldsymbol{w}$ associated with $\{g_i\}$. This owes to the fact that our design $H_{K,\mathcal{D}_{\mathcal{G}}^{em}}$ allows us to separate $\mathcal{D}_{\mathcal{G}}$ from the task-dependent kernels (i.e., f_S and f_T), which is otherwise not possible if we parametrized the score function (i.e., Eq. 9) using naive methods (e.g., neural networks). Following the same logic behind the translation from Eq. (6) to Eq. (7), the above constraints lead to a similar optimization program:

$$\min_{\boldsymbol{w}, \eta_i} \|\boldsymbol{w}\|^2 + C \cdot \sum_i \eta_i \ \textbf{s.t.} \underbrace{\min_{y \in \mathcal{Y}_S} \alpha \sum_{i=1}^{K} w_i f_S(x_j, y, g_i)}_{\textbf{source inference}} - \sum_{i=1}^{K} w_i f_S(x_j, y_j, g_i) \geq -\eta_j, \forall j. \tag{12}$$

The above program shares the same type with Eq. (7), and we again defer the optimization details to the appendix.

Distribution Tuning. With the importance vector $\boldsymbol{w}$ learned based on the subgraphs $\{g_i\}$ sampled from the current θ, we seek to fine-tune $\mathcal{D}_{\mathcal{G},\theta}^{em}$ to make it aligned more with the latent distribution $\mathcal{D}_{\mathcal{G}}$, which is desired as suggested by the proof of Theorem 1. Inspired by Eq. (10), the true likelihood associated with g_i is approximated by $w_i^* := w_i \, \mathcal{D}_{\mathcal{G},\theta}^{em}[g_i]$. Consequently, one possible way to reshape $\mathcal{D}_{\mathcal{G},\theta}^{em}$ is to find the θ^* that can minimize the discrepancy between $\mathcal{D}_{\mathcal{G},\theta^*}^{em}$ and $\mathcal{D}_{\boldsymbol{w}^*}$, i.e., $\theta^* = \arg\min_\theta D(\mathcal{D}_{\mathcal{G},\theta}^{em} \,||\, \mathcal{D}_{\boldsymbol{w}^*})_{|\{g_i\}}$, where $\mathcal{D}_{\boldsymbol{w}^*}$ is the discrete distribution over $\{g_i\}$ defined by normalizing $(w_1^*, ..., w_K^*)$, and the distance measure $D(||)$ can be selected at the convenience of the choice of the $\mathcal{D}_{\mathcal{G},\theta}^{em}$ – for example, cosine similarity or cross-entropy. For such problems, standard methods can be directly applied when $\mathcal{D}_{\mathcal{G},\theta}^{em}$ is parameterized by common distribution families; first- and second-order methods can be readily used if $\mathcal{D}_{\mathcal{G},\theta}^{em}$ has a complex form such as neural networks.

Algorithm 1. QRTS-D

Input: $D_{f_S,\alpha} = \{x_y, y_i\}, C, K, T, \alpha, \mathcal{D}^{em}_{\mathcal{G},\theta}$;
Output: $\boldsymbol{w} = (w_1, ..., w_K)$ and $\{g_1, ..., g_K\}$
Initialize θ, $t = 0$;
repeat
  $\{g_1, ..., g_K\}$ iid from $\mathcal{D}^{em}_{\mathcal{G},\theta}$;
  Compute $\boldsymbol{w}$ via Eq. (12) based on $D_{f_S,\alpha}$ and C;
  Update θ via $\theta^* = \arg\min_\theta D(\mathcal{D}^{em}_{\mathcal{G},\theta} \,||\, \mathcal{D}_{\boldsymbol{w}^*})_{|\{g_i\}}$;
  $t = t + 1$
until $t = T$
Return $\{g_1, ..., g_K\}$ and $\boldsymbol{w}$

The QRTS-D method is conceptually simple, as summarized in Alg. 1. Similar to QRTS-P, using such a method requires algorithms for solving the source and target inferences in Eqs. (11) and (12). In what follows, we discuss such issues as well as the possibility of enhancing QRTS-D using QRTS-P

*Remark 2 (**Source and Target Inferences**).* For QRTS-P, the source (resp., target) inference problem is nothing but to solve the source (resp., target) query-decision optimization task in its deterministic case. For QRTS-D, the inference problems are to solve the source and target tasks over a weighted combination of deterministic graphs. For the shortest path problem, such inference problems can be solved in polynomial time; for the minimum Steiner tree problem, such inference problems admit 2-approximation [22].

*Remark 3 (**QRTS-P vs QRTS-D**).* As one may have noticed, QRTS-P is a natural special case of the importance learning phase of QRTS-D, in the sense that each w_e in QRTS-P corresponds to the importance of the subgraph with one edge (i.e., e). In other words, QRTS-P can be viewed as the QRTS-D where the support of $\mathcal{D}^{em}_{\mathcal{G},\theta}$ is the span of single-edge subgraphs with unit weights. Notably, the dimension of QRTS-P is fixed and thus limited by the number of edges, while the dimension K of QRTS-D can be made arbitrarily large. For this reason, QRTS-P may be preferred if the sample size is small, while QRTS-D can better handle large sample sets, which is evidenced by our experimental studies.

*Remark 4 (**QRTS-PD**).* In QRTS-D, the initialization of $\mathcal{D}^{em}_{\mathcal{G},\theta}$ is an open issue, and this creates the possibility of integrating QRTS-P into QRTS-D by initializing $\mathcal{D}^{em}_{\mathcal{G},\theta}$ using the weights $\{w_e\}$ learned from QRTS-P. This leads to another approach called QRTS-PD consisting of three steps: **a)** run QRTS-P to acquire the estimations $\{w_e\}$; **b)** stabilize θ based on $\{w_e\}$ through maximum likelihood estimation, i.e., $\min_\theta -\sum_{e\in E} \log \sum_{g\in\mathcal{G}} \mathcal{D}^{em}_{\mathcal{G},\theta}[g|g(e) = w_e]$; **c)** run QRTS-D. From such a perspective, QRTS-PD can be taken as a continuation of QRTS-P to further improve the generalization performance by building models that are more expressive.

5 Empirical Studies

In this section, we present empirical studies demonstrating that QRTS can be solved with statistical significance using the presented methods.

5.1 Experimental Settings

Source and Target Tasks. We specifically focus on two query-decision optimization tasks: shortest path and minimum Steiner tree [9]. Depending on the selection of the source and target tasks, we have two possible settings: *Path-to-Tree* and *Tree-to-Path*. The source and target inferences can be approximated effectively, as discussed in Remark 2. These algorithms are also used to generate samples of query-decision pairs (i.e., Eq. (2)).

Table 1. Results for Path-to-Tree on Kro, Col and BA. Each cell shows the mean ratio together with the standard deviation (std). The top three results in each column are highlighted.

Train Size		Kro			Col			BA		
		60	240	2400	60	240	2400	60	240	2400
QRTS-P		**4.0**$_{(0.3)}$	3.4$_{(0.5)}$	2.4$_{(0.3)}$	291$_{(78)}$	**150**$_{(29)}$	128$_{(7.4)}$	181$_{(28)}$	144$_{(41)}$	63$_{(35)}$
QRTS **-PD$^-$**	60	4.4$_{(0.3)}$	3.4$_{(0.8)}$	2.3$_{(0.4)}$	**250**$_{(82)}$	367$_{(56)}$	123$_{(2.3)}$	209$_{(22)}$	195$_{(22)}$	40$_{(13)}$
	240	4.6$_{(0.7)}$	3.7$_{(0.1)}$	2.7$_{(0.2)}$	330$_{(65)}$	227$_{(14)}$	117$_{(6.1)}$	**129**$_{(22)}$	149$_{(19)}$	35$_{(12)}$
	2400	4.3$_{(0.9)}$	**3.2**$_{(0.5)}$	2.4$_{(0.1)}$	**214**$_{(68)}$	183$_{(69)}$	**88**$_{(12)}$	**139**$_{(38)}$	**131**$_{(44)}$	**27**$_{(7.2)}$
QRTS **-PD-1**	60	**3.9**$_{(0.7)}$	3.4$_{(0.8)}$	**2.3**$_{(0.4)}$	361$_{(55)}$	225$_{(11)}$	115$_{(5.1)}$	177$_{(31)}$	130$_{(34)}$	43$_{(14)}$
	240	4.2$_{(0.4)}$	3.2$_{(0.5)}$	2.4$_{(0.2)}$	350$_{(88)}$	245$_{(31)}$	129$_{(16)}$	166$_{(34)}$	132$_{(49)}$	34$_{(13)}$
	2400	4.3$_{(0.5)}$	**3.1**$_{(0.3)}$	**2.3**$_{(0.4)}$	**261**$_{(93)}$	186$_{(24)}$	**106**$_{(6.2)}$	**160**$_{(29)}$	**119**$_{(3)}$	34$_{(7.3)}$
QRTS **-PD-3**	60	4.3$_{(1.1)}$	3.2$_{(0.6)}$	2.6$_{(0.5)}$	431$_{(27)}$	**167**$_{(99)}$	**110**$_{(15)}$	391$_{(46)}$	144$_{(13)}$	60$_{(13)}$
	240	4.3$_{(0.9)}$	3.3$_{(0.6)}$	2.3$_{(0.4)}$	317$_{(87)}$	183$_{(35)}$	126$_{(11)}$	202$_{(38)}$	138$_{(17)}$	**30**$_{(17)}$
	2400	**4.0**$_{(0.4)}$	**3.2**$_{(0.4)}$	**2.2**$_{(0.2)}$	324$_{(38)}$	**107**$_{(12)}$	113$_{(6.1)}$	184$_{(49)}$	**120**$_{(2.4)}$	**23**$_{(2.7)}$
Unit & Rand		5.2$_{(0.2)}$ & 10.4$_{(0.3)}$			990$_{(68)}$ & 2231$_{(45)}$			349$_{(4.7)}$ & 749$_{(13)}$		

Graph, True Distribution, and Samples. We adopt a collection of graphs of classic types: a Kronecker graph (**Kro**) [14], a road network of Colorado (**Col**) [5], a Barabasi-Albert graph (**BA**) [1], and two Watts-Strogatz graphs with different densities (**WS-dense** and **WS-sparse**) [23]. The statistics of these graphs can be found in the appendix. To have a diverse graph pattern, we generate the ground truth distribution $\mathcal{D}_{\mathcal{G}}$ by assigning each edge a Weibull distribution [12] with parameters randomly selected from $\{1, ..., 20\}$. For each graph and each problem instance, we generate a pool of $10,000$ query-decision pairs.

QRTS Methods. We use QRTS-PD-1 (resp., QRTS-PD-3) to denote the QRTS-PD method when one (resp., three) iterations over the three phases are

used. Based on QRTS-PD-1, we implement QRTS-PD$^-$ that foregoes the distribution tuning phase. For these methods, the model dimensions K are selected from $\{60, 240, 2400\}$. We utilize the one-slack cutting plane algorithm [10] for the large-margin training (i.e., Eqs. (7) and (12)). The empirical distribution $\mathcal{D}_{\mathcal{G}}^{em}$ is parameterized by assigning each edge an exponential distribution.

Baselines. We set up two baselines **Unit** and **Rand**. Unit computes the predictions based on the graph with unit-weight edges. Rand computes the decision based on the graph with random weights. Unit essentially leverages only the graph structure to compute predictions. We note that none of the methods in existing papers can be directly applied to QRTS (Remark 1).

Training and Testing. In each run, the training size is selected from $\{60, 240, 2400\}$, and the testing size is 1000, where samples are randomly selected from the sample pool. Given a testing set $\{x_i, y_i\}$ of the target task, the performance is measured by the ratio $\sum_i F_{f_T,\mathcal{D}_{\mathcal{G}}}(x_i, y_i^*) / \sum_i F_{f_T,\mathcal{D}_{\mathcal{G}}}(x_i, y_i)$, where y_i^* is the predicted decision associated with x_i; a lower ratio implies better performance. We report the average ratios and the standard deviations over five runs for each method.

Table 2. Results on Tree-to-Path. Each cell shows the mean ratio together with the standard deviation (std). Small stds (< 0.1) are denoted as 0.0. The top three results in each column are highlighted.

Train Size		Kro			Col			BA		
		60	240	2400	60	240	2400	60	240	2400
QRTS-P		**1.46** (0.0)	**1.37** (0.0)	1.60 (0.0)	9.8 (0.2)	6.6 (0.4)	**6.1** (0.1)	1.6 (0.1)	1.4 (0.2)	1.4 (0.1)
QRTS -PD$^-$	60	**1.44** (0.1)	1.41 (0.1)	1.39 (0.0)	11 (5.8)	8.6 (1.5)	7.1 (0.6)	1.9 (0.4)	1.4 (0.2)	1.5 (0.1)
	240	**1.42** (0.1)	1.46 (0.0)	1.39 (0.0)	9.9 (4.2)	6.8 (3.6)	6.5 (0.4)	1.7 (0.3)	1.5 (0.2)	1.5 (0.1)
	2400	1.48 (0.1)	1.45 (0.0)	1.38 (0.0)	**8.9** (3.1)	**6.0** (1.3)	6.2 (0.9)	**1.5** (0.2)	**1.3** (0.1)	**1.3** (0.1)
QRTS -PD-1	60	1.55 (0.0)	1.42 (0.1)	1.37 (0.0)	6.6 (0.6)	6.3 (2.4)	6.8 (0.6)	1.6 (0.3)	1.5 (0.0)	1.4 (0.1)
	240	1.56 (0.1)	1.44 (0.1)	1.36 (0.0)	11 (3.2)	7.6 (1.4)	6.6 (0.0)	1.6 (0.3)	1.5 (0.3)	1.5 (0.1)
	2400	1.52 (0.1)	1.39 (0.1)	1.37 (0.0)	14 (2.4)	7.7 (3.9)	7.2 (0.3)	1.7 (0.1)	1.5 (0.3)	1.7 (0.1)
QRTS -PD-3	60	1.53 (0.1)	**1.41** (0.1)	**1.34** (0.1)	13 (4.2)	**5.9** (1.5)	**5.5** (0.9)	1.7 (0.2)	1.5 (0.1)	1.4 (0.1)
	240	1.50 (0.1)	1.42 (0.0)	**1.36** (0.0)	**9.4** (1.4)	**6.6** (0.2)	**5.9** (0.1)	**1.6** (0.1)	**1.4** (0.2)	**1.2** (0.1)
	2400	1.47 (0.1)	**1.41** (0.1)	**1.32** (0.0)	**7.8** (0.6)	8.5 (1.4)	7.7 (2.0)	**1.3** (0.1)	**1.3** (0.1)	**1.3** (0.2)
Unit & Rand		1.57 (0.1) & 1.78 (0.1)			9.2 (0.1) & 19 (0.1)			1.57(0.0) & 2.2(0.3)		

5.2 Analysis

The results on Kro, Col, and BA are given in Tables 1 and 2. The results on WS-sparse and WS-dense can be found in the Appendix. The main observations are listed below, and the minor observations are given in the appendix.

O1: The Proposed Methods Behave Reasonably with Promising Performance. First, we observe that the proposed methods perform significantly better when more samples are given, which suggests that they are able to

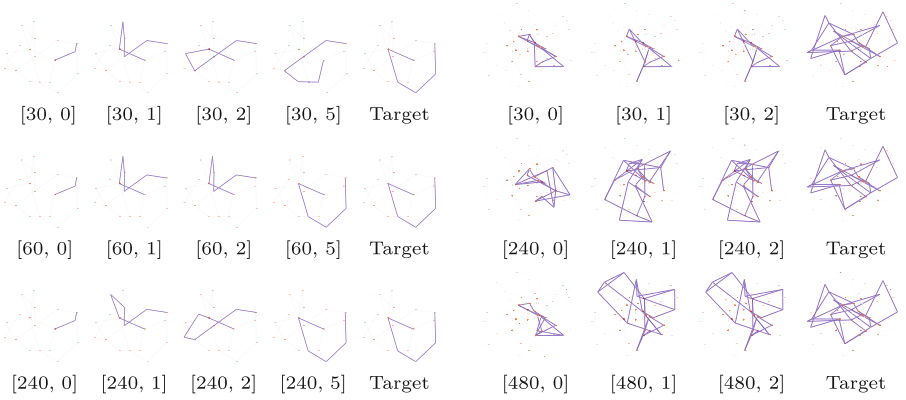


Fig. 2. The left (resp., right) shows the visualizations of the solution to one testing query for the Tree-to-Path (resp., Path-to-Tree) problem under QRTS-P on Kro. The figure labeled by $[a, b]$ shows the result with training size a after b iterations in the cutting plane algorithm. Each row shows the results under one training size, where the last figure shows the optimal solution.

infer meaningful information from the samples toward solving the target task. On the other hand, all the proposed methods are clearly better than Rand, implying that the model efficacy is non-trivial. In addition, they easily outperform Unit by an evident margin in most cases. For example, for Path-to-Tree on BA in Table 1, the best ratio achieved by QRTS-PD-3 is 23, while Unit and Random cannot produce a ratio smaller than 300.

O2: QRTS-P Offers an Effective Initialization for QRTS-D. With very few exceptions, QRTS-PD performs much better than QRTS-P under the same sample size, which confirms that QRTS-P can indeed be improved by using importance learning through re-sampling, which echos Remark 4. This is especially true when the sample size is large; for example, for Path-to-Tree on BA with 2400 samples, methods based on QRTS-PD with a dimension of 2400 are at least twice better than QRTS-P in terms of the performance ratio, demonstrating that QRTS-PD of a high dimension can better consume large datasets.

O3: Distribution Tuning is Helpful After Multiple Iterations. Since QRTS-D can be used without the distribution tuning phase, we are wondering if the distribution turning phase is necessary. By comparing QRTS-PD-1 with QRTS-PD$^-$, we see that the distribution tuning phase can be useful in many cases, but its efficacy is not very significant. However, combined with the results of QRTS-PD-3, we observe that the distribution turning phase can better reinforce the optimization effect when multiple iterations are used. Finally, by comparing QRTS-PD-1 and QRTS-PD-3, we find that training more iterations is useful mostly when the model dimension is large, which is especially the case for Tree-to-Path (Table 2).

O4: The Learning Process is Smooth. Fig. 2 visualizes the learning process of QRTS-P for two example testing queries. One can see that QRTS-P tends to select solutions with fewer edges under the initial random weights $\boldsymbol{w}$, and it gradually finds better solutions (possibly with more edges) when better weights are learned. We have such observations for most of the samples, which suggest that the proposed method works the way it is supposed to. More visualizations can be found in the appendix.

6 Future Directions

Although we observed that the approximation ratio becomes better with the increase in training size and training iteration, it is not always the case that the solution converges to the optimal one – for example, Fig. 2-Right and the visualizations in the Appendix. This is reasonable because two solutions may have similar costs but with very different edge sets. Depending on the needs of the agents, other metrics can be adopted, which may require new designs of the hypothesis space and training methods. Another important future direction is to enhance the proposed method through (deep) representation learning.

Acknowledgement. This project is supported in part by National Science Foundation under Award Career IIS-2144285.

References

1. Barabási, A.L., Albert, R.: Emergence of scaling in random networks. Science **286**(5439), 509–512 (1999)
2. Bengio, Y.: Using a financial training criterion rather than a prediction criterion. Int. J. Neural Syst. **8**(04), 433–443 (1997)
3. Bertsimas, D., Kallus, N.: From predictive to prescriptive analytics. Manage. Sci. **66**(3), 1025–1044 (2020)
4. Chen, C., Seff, A., Kornhauser, A., Xiao, J.: Deepdriving: learning affordance for direct perception in autonomous driving. In: Proceedings of the IEEE International Conference on Computer Vision, pp. 2722–2730 (2015)
5. Demetrescu, C., Goldberg, A.V., Johnson, D.S.: Implementation challenge for shortest paths. Encyclopedia Algorithms **15**, 54 (2008)
6. Edwards, W.: The theory of decision making. Psychol. Bull. **51**(4), 380 (1954)
7. Ford, B., Nguyen, T., Tambe, M., Sintov, N., Fave, F.D.: Beware the soothsayer: from attack prediction accuracy to predictive reliability in security games. In: Khouzani, M.H.R., Panaousis, E., Theodorakopoulos, G. (eds.) GameSec 2015. LNCS, vol. 9406, pp. 35–56. Springer, Cham (2015). https://doi.org/10.1007/978-3-319-25594-1_3
8. Hofmann, T., Schölkopf, B., Smola, A.J.: Kernel methods in machine learning (2008)
9. Hwang, F.K., Richards, D.S.: Steiner tree problems. Networks **22**(1), 55–89 (1992)
10. Joachims, T., Finley, T., Yu, C.N.J.: Cutting-plane training of structural svms. Mach. Learn. **77**(1), 27–59 (2009)

11. Kochenderfer, M.J.: Decision making under uncertainty: theory and application. MIT press (2015)
12. Lai, C., Murthy, D., Xie, M.: Weibull distributions. Wiley Interdisciplinary Reviews: Computational Statistics **3**(3), 282–287 (2011)
13. Lehmann, E.L., Casella, G.: Theory of point estimation. Springer Science & Business Media (2006)
14. Leskovec, J., Chakrabarti, D., Kleinberg, J., Faloutsos, C., Ghahramani, Z.: Kronecker graphs: an approach to modeling networks. J. Mach. Learn. Res. **11**(2) (2010)
15. Lucchi, A., Li, Y., Fua, P.: Learning for structured prediction using approximate subgradient descent with working sets. In: Proceedings of the CVPR (2013)
16. Murphy, K.P.: Machine learning: a probabilistic perspective. MIT press (2012)
17. Provost, F., Fawcett, T.: Data science and its relationship to big data and data-driven decision making. Big Data **1**(1), 51–59 (2013)
18. Rajkumar, A., Agarwal, S.: Online decision-making in general combinatorial spaces. Advances in Neural Information Processing Systems 27 (2014)
19. Suthaharan, S., Suthaharan, S.: Support vector machine. Machine learning models and algorithms for big data classification: thinking with examples for effective learning, pp. 207–235 (2016)
20. Tokdar, S.T., Kass, R.E.: Importance sampling: a review. Wiley Interdisciplinary Reviews: Computational Statistics **2**(1), 54–60 (2010)
21. Tong, G.: Usco-solver: solving undetermined stochastic combinatorial optimization problems. NeurIPS **34**, 1646–1659 (2021)
22. Vazirani, V.V.: Approximation algorithms, vol. 1. Springer (2001)
23. Watts, D.J., Strogatz, S.H.: Collective dynamics of 'small-world' networks. Nature **393**(6684), 440–442 (1998)