Risk-Neutral Second Best Toll Pricing

Xuegang (Jeff) Ban* Michael Ferris† Lisa Tang‡

Abstract

We propose a risk-neutral second best toll pricing scheme to account for the possible nonuniqueness of user equilibrium solutions. The scheme is designed to optimize for the expected objective value as the UE solution varies within the solution set. We show that such a risk-neutral scheme can be formulated as a stochastic program, which complements the traditional risk-prone second best toll pricing (SBTP) approach and the risk-averse SBTP approach we developed recently. The proposed model can be solved by a simulation-based optimization algorithm that contains three major steps: characterization of the UE solution set, random sampling over the solution set, and a two-phase simulation optimization step. Numerical results illustrate that the proposed risk-neutral design scheme is less aggressive than the risk-prone scheme and less conservative than the risk-averse scheme, and may thus be more preferable from a toll designer's point of view.

1 Introduction

The Second-Best Toll Pricing (SBTP) problem has been extensively studied in the literature. It aims to determine optimal tolls for a given set of links in a transportation network to achieve certain system management objectives. For detailed reviews, one can refer to [1, 2, 3, 4] and the references therein. Many researchers have modeled SBTP as a bilevel problem or an MPEC (mathematical program with equilibrium constraints). The upper level optimizes a certain objective function from the transportation system point of view and the lower level is a user equilibrium (UE) problem to account for the choice behavior of individual motorists. These bilevel SBTP models usually assume, explicitly or implicitly, that the lower level UE problem has a unique solution. As pointed out in [5], if the UE solution is not unique, existing bilevel SBTP models usually result in a toll scheme that optimizes for the "best case" scenario - the smallest objective value is minimized as the UE solution varies. In this sense, existing SBTP models are "risk-prone." The UE solution set (in the case of non-unique solutions) actually represents uncertainty in SBTP design, which cannot be accounted for by the existing risk-prone approach. Alternatively, a "risk-averse" SBTP approach is proposed in [5] which optimizes for the "worst-case" scenario, i.e. the largest objective value over the UE solution set is minimized. Such a risk-averse model is formulated as a "min-max" problem in [5] by adopting the robust optimization concept introduced in [6]. The min-max formulation is further solved for affine UEs by a fortified-descent simplex method originally developed in [7] based on an explicit expression for the solution set of an affine UE. As illustrated in [5], if UE has multiple solutions, risk-prone and risk-averse SBTP toll schemes may produce very different tolls as well as

*Department of Civil and Environmental Engineering, Rensselaer Polytechnic Institute, banx@rpi.edu
†Computer Sciences Department, University of Wisconsin at Madison, ferris@cs.wisc.edu
‡Industrial and Systems Engineering Department, University of Wisconsin at Madison, lmtang@wisc.edu

different upper level objective values. In certain cases, risk-averse tolls are superior to traditional risk-prone tolls in terms of both system performance and robustness. This indicates the necessity to explicitly consider and address the non-unique UE solutions which are common especially for urban traffic networks. Notice that in the mathematical programming community, the issue of nonunique solutions of the lower level problem in a bilevel formulation has been recognized [8]. Most previous research focuses on the risk-prone approach (also called the "optimistic" approach in [8]). Morgan and her colleagues [9] investigated the "pessimistic" approach (similar to the risk-averse scheme proposed in this article), which they referred to as weak Stackelberg games. They mainly studied the solution existence conditions and other theoretical properties of the problem. No solution method was proposed.

It is well known however that the results obtained by solving a robust optimization model (i.e. the risk-averse approach) is generally too conservative (or too pessimistic) [10], while the the risk-prone approach is too aggressive (or too optimistic). This is because both approaches consider only extreme cases, i.e. a single point in the UE solution set. To address this issue, we propose in this article a "risk-neutral" approach that explicitly considers the entire UE solution set when designing SBTP. The approach aims to optimize for the expected objective value as the UE solution changes within the solution set. In particular, by associating certain probability distribution function to the realization of the UE solution over its solution set, we formulate the proposed risk neutral model as a stochastic program, similar to the problem studied in [11, 12]. In this setting, the UE solution set is the probability space. In [11, 12], the probability space is fixed. Our risk-neutral problem however has a changing probability space because the UE solution set varies with the toll vector. Hence, the risk-neutral model we study in this article extends the model and results in [11, 12].

By assuming certain probability distribution functions as a realization of the UE solution, the risk-neutral model can be solved by a simulation-based optimization technique for affine UEs. First, the solution set of an affine UE can be explicitly represented as a convex and compact set [5]. The realization of the UE solution can be sampled from this set based on the assumed distribution. One critical step in this process is the ability to sample over the probability space (i.e. the UE solution set). For this purpose, we extend the hit-and-run random sampling algorithm originally developed by Smith [13] from a full dimensional subset in R^n to a subset of (the translate of) a subspace in R^n. The samples are then evaluated for objective values, which are used in a two-phase simulation optimization algorithm. The algorithm first identifies promising subregions and then performs a derivative-free optimization on a quadratic approximation of the original problem. This process repeats itself until certain convergence criterions are met. We test the risk-neutral model and solution algorithm using an small example in this article to illustrate how the algorithm performs.

The proposed risk-neutral approach provides an alternative way to account for the nonuniqeness of UE solutions in SBTP design, or in a more general sense the nonuniqueness of the lower level solution in a bilevel formulation. As shown in Section 4, the risk-neutral approach is less aggressive than the risk-prone scheme and less conservative than the risk-averse scheme. Therefore it can provide more insights, and sometimes is more desirable, for toll authorities to design effective pricing schemes when UE solution is not unique. Notice that although we focus on link-based UE solutions in this article (also in [5]), the proposed risk-neutral scheme can be applied directly to path-based solutions. Path-based formulations are necessary for cases when path costs are nonadditive [14, 15, 16, 17] or path-based tolling is needed (e.g. for the purpose of controlling emissions). As path-based UE solutions are nonunique in general [18], the proposed risk-taking-based SBTP schemes are expected to play more significant roles in path-based tolling or other network design applications.

This article is organized as follows. Section 2 starts with a brief summary of the risk-prone and risk-averse SBTP approaches, and presents in detail the stochastic program for the risk-neutral

approach. A small illustrative example is also provided in this section. An algorithm based on simulation optimization is proposed in Section 3, including the characterization of the UE solution set, random sampling over the solution set, and a two-phase simulation optimization algorithm. A numerical example is provided in Section 4. We conclude the article in Section 5.

2 Risk-Neutral SBTP Model

2.1 Risk-Taking in SBTP Design

Assume a transportation network can be denoted as $G(N, A)$, where N is the set of nodes and A is the set of links. We use $a \in A$ to denote a link, and assume x_a and y_a are the total flow of and the toll imposed on link a respectively. Denote two vectors $x = (x_a)_{a\in A}$ and $y = (y_a)_{a\in A}$. Usually we impose lower and upper bounds to the toll vector. Thus we define $K_y = \{y|y_l \leq y \leq y_u\}$, where y_l and y_u are the lower and upper bounds of y respectively. Further denote $t_a(x)$ the travel time of link a, which is a function of the total link flow vector x, and $t = (t_a)_{a\in A}$. If a toll vector y is imposed, the resulting user equilibrium (UE) problem is denoted as $UE(y)$, which can be defined as to *find $x \in K$ such that the following variational inequality (VI) is satisfied*:

$$c^T(y, x)(x' - x) \geq 0, \forall x' \in K. \tag{1}$$

Here K is the constraint set of $UE(y)$ which is usually nonempty, convex, and compact. This guarantees the existence of at least one solution for any continuous c, and the solution set is compact if multiple solutions exist [19, 2.2.5]. We notice here that the risk-neutral model proposed in this article only requires solution existence conditions of an UE. The characterization of the solution set however requires monotonicity which leads to a convex UE solution set.

The *generalized* link travel time function c is defined as [5]:

$$c(y, x) = t(x) + y/\theta, \tag{2}$$

where θ is the "value of time." Denote the solution set of $UE(y)$ as $S(y)$.

We further denote $f(y, x)$ the objective function used by the toll authority to determine tolls. In practice, $f(y, x)$ may be the total system travel time or similar objectives the toll designer may have. Most existing SBTP models aim to find an optimal toll by solving the following MPEC:

$$\begin{aligned} MPECSBTP \qquad \min_{y,x} \quad & f(y, x) && (3)\\ \text{s.t.} \quad & y \in K_y && (4)\\ & x \quad \text{solves UE}(y). && (5) \end{aligned}$$

Since we use $S(y)$ to denote the solution set of $UE(y)$, we may rewrite the constraint that x solves $UE(y)$ as $x \in S(y)$. Here $S(y)$ is a set-valued map [19] of the toll vector y since $UE(y)$ may have multiple solutions. If we let

$$G = \{(y, x) \mid x \in S(y), \quad y \in K_y\}$$

be the graph of the set-valued map S, we can rewrite $MPECSBTP$ into the following single level problem:

$$RPSBTP \qquad \min_{y,x} \quad f(y,x) \tag{6}$$

$$\text{s.t.} \quad (y,x) \in G. \tag{7}$$

Here the label $RPSBTP$ stands for "risk-prone second-best toll pricing."

Notice that $RPSBTP$ is equivalent to

$$\min_{y \in K_y} \min_{x \in S(y)} \quad f(y,x). \tag{8}$$

Model (8) shows that $RPSBTP$ aims to find a toll $y \in K_y$ that optimizes the "best-case" scenario. Here the "best-case" for a given toll y refers to $\eta(y) \equiv \min_{x \in S(y)} \quad f(y,x)$, which is the smallest objective value of $f(y,x)$ as x varies within $S(y)$. This is why the $RPSBTP$ model is a "risk-prone" design approach. Clearly, when $S(y)$ contains multiple elements, the set $S(y)$ represents uncertainty for SBTP design because for a fixed toll vector y, the objective value of $f(y,x)$ may change as x varies within $S(y)$. To account for this uncertainty, a "risk-averse" SBTP approach is proposed in [5]. In particular, the risk-averse approach adopts the robust optimization concept and can be formulated as a *min-max* problem (denoted by $RASBTP$, which stands for "risk-averse second best toll pricing") as follows:

$$RASBTP \qquad \min_{y \in K_y} \quad \max_{x \in S(y)} \quad f(y,x) \tag{9}$$

If we define $\Psi(y) \equiv \max_{x \in S(y)} f(y,x)$, it is easy to see that the risk-averse model $RASBTP$ aims to design the toll so that it is optimal for the worst case scenarios. Here the "worst case" for a given toll represents the largest objective value as UE solution varies under the toll, i.e. $\Psi(y)$.

Denote y_a^* the optimal solution to $RASBTP$ and x_a^* its associated flow pattern. We will have $f(y_a^*, x_a^*) \geq f(y_a^*, x), \forall x \in S(y_a^*)$. This implies that, as the UE solution varies after the optimal risk-averse toll y_a^* is imposed, the objective value will never increase. In other words, we will always be "better off" as the UE solution changes if we implement the risk-averse toll. On the other hand, assume y_p^* is the optimal solution to $RPSBTP$ and x_p^* is its associated flow pattern. We will have $f(y_p^*, x_p^*) \leq f(y_p^*, x), \forall x \in S(y_p^*)$. This means that if the risk-prone toll is implemented, we will always be "worse off" since the objective value will never decrease as the UE solution varies. This illustrates, from the toll designer's perspective, that the risk-averse toll design approach is more robust.

2.2 A Stochastic Program for Risk-Neutral SBTP

From the above discussion, we can see that in case of the UE solution not being unique, the risk-prone SBTP approach is too optimistic by focusing on the best-case scenario. On the other hand, while being able to account for the uncertainty due to $S(y)$, it is well known that robust optimization and the min-max formulation (9) is a very conservative way to design tolls. Therefore, a design approach that is in-between, i.e. less aggressive than $RPSBTP$ and less conservative than $RASBTP$, seems more desirable.

We propose a "risk-neutral" second best toll pricing approach, which aims to minimize the expected objective value as the UE solution varies. For this purpose, we assume the *realization*

of the UE solution follows certain distribution over the solution set $S(y)$. That is, $S(y)$ is the probability space and the UE solution is assumed to be a random variable defined on $S(y)$. More specifically, we define $\tilde{x}(y)$ the (random) UE solution that follows the assumed distribution over $S(y)$ under a toll vector y. Based on this setting, the risk-neutral approach may be modeled as the following stochastic formulation:

$$RNSBTP \quad \min_{y \in K_y} \quad F(y) = E_{\tilde{x}(y) \sim S(y)}[f(y, \tilde{x}(y))]. \tag{10}$$

Here E denotes "expected value" and $\tilde{x}(y) \sim S(y)$ means that $\tilde{x}(y)$ follows certain distribution over the set $S(y)$, which is the probability space. In addition, *RNSBTP* stands for "risk-neutral second best toll pricing."

In [12, 11], a similar problem as shown below was studied extensively:

$$\min_{y \in K_y} \quad F(y) = E_{\tilde{x} \sim X}[f(y, \tilde{x})]. \tag{11}$$

Here X is the probability space, which is a fixed set. We can see that the only difference between our proposed risk-neutral SBTP model (10) and the stochastic program (11) is that the probability space is fixed in (11) while in $RNSBTP$, the probability space $S(y)$ is changing with the toll vector. In this sense, the proposed risk-neutral model $RNSBTP$ extends some of the results in [12, 11] to stochastic programs with varying probability space.

Quantifying the actual distribution of $\tilde{x}(y)$ for a given toll vector y is crucial to the risk-neutral model. In this article, we simply assume $\tilde{x}(y)$ follows a uniform distribution over $S(y)$, which implies that all solutions within $S(y)$ have the same probability to be realized in practice. This simplified assumption may not be valid in practice as the realization of the UE solutions may be determined by user choice behaviors especially the day-to-day route choice adjustment [20]. Such issues may be investigated in future research. The model presented in this article however can capture any distribution form that can be properly identified. Furthermore, the solution algorithm we present in Section 3 may also be extended to solve the *RNSBTP* model with distributions other than the uniform distribution [1].

To illustrate the risk-neutral SBTP scheme and the difference among the three toll design approaches, we present an example in this section, which was also discussed in [5]. The setting of the example is described in more detail in Section 4, where discussions of how to solve the risk-neutral SBTP model is presented. Here we notice that the problem can actually be solved analytically due to its special structure. As shown in [5], the risk-prone and the risk-averse solutions for this toll design problem are $y_p^* = 5$ and $y_a^* = 13.57$ respectively. The corresponding objective values are 125 and 167.86 respectively. This is shown in Figure 1. In this figure, the three axes represent the link flow on the three links of the small network in Figure 5. As shown in Appendix A of the article, the solution set under a given toll vector y is $S(y) = \{x = (x_1, x_2, x_3)^T \geq 0 | x_1 = (10 + y)/3, \quad x_2 + x_3 = (20 - y)/3\}$, which is a line in the three dimensional space. Appendix A also presents discussions on how the analytical risk-neutral solution $y_n^* = 11$ can be derived. The objective value for the risk-neutral approach is 155. The example illustrates that if the UE solution is not unique (e.g. the lines in Figure 1), the three toll design approaches may generate different solutions. In general, the risk-neutral solution lies in-between the risk-prone and risk-averse solutions. The objective value of the risk-neutral approach also lies in-between those of the risk-prone and risk-averse approaches, indicating that the risk-neutral approach is indeed less

[1] This requires to construct a particular distribution from the uniform distribution generated by the EHR algorithm in Section 3.3, which is a standard operation

aggressive than the risk-prone approach and less conservative than the risk-averse approach. In this sense, it provides another alternative for SBTP design. In addition, while risk-prone and risk-averse optimal tolls (i.e. y_p^* and y_a^*) have their associated optimal (predicated) UE solutions (i.e. x_p^* and x_a^* respectively), there is no a single UE solution that is"optimal" under the optimal risk-neutral toll y_n^*. Rather, the risk-neutral optimal toll is optimized over the entire solution set $S(y_n^*)$ in terms of the expected objective value. It is this capability of explicitly considering all UE solutions that makes the risk-neutral SBTP approach more appealing than the other two design approaches. In Appendix B, a larger example is provided on a grid network, which shows similar results as the small example above.

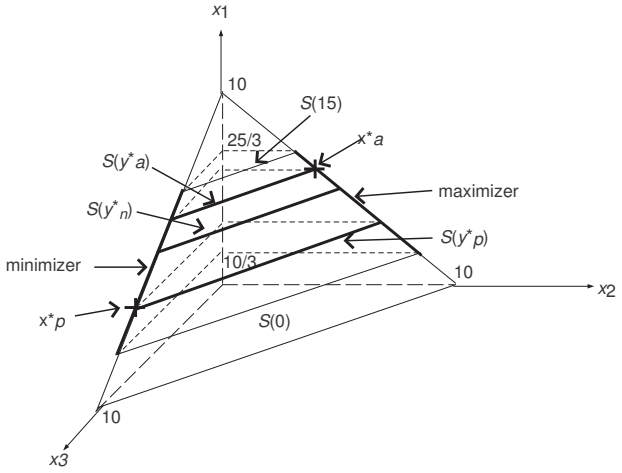

Figure 1: Solution Set of SBTP

3 Solution Algorithm

In this section, we present the solution algorithm for the risk-neutral SBTP model. The algorithm is designed based on the simulation-based optimization technique.

3.1 Outline of the Algorithm

The solution algorithm, called the Simulation Optimization based Risk-Neutral Algorithm (**SORNA**), is first given as follows:

SORNA Algorithm

Step 1. *Initialization*. Select an initial toll vector y^0 and set $k = 0$.

Step 2. *Construct solution set* $S(y^k)$. Solve the UE problem under toll y^k, $UE(y^k)$. Denote the solution as $\bar{x}^k$. Then construct the solution set of $S(y^k)$ based on $\bar{x}^k$ (See Section 3.2).

Step 3. *Random Sampling*. Perform uniform random sampling over $S(y^k)$ (See Section 3.3). Denote the samples as $x_i^k, 1 \leq i \leq M_k$. Here M_k is the number of samples which is set by the two-phase simulation optimization in Step 4 and may be different for different y^k's.

Step 4. *Two-Phase Simulation Optimization.* The first phase is a global exploration step to identify promising subregions. The second step is to solve a quadratic approximation of the original problem in each subregion by derivative-free algorithms. This step will call Step 3 to generate random samples in the solution set. An approximate solution $\hat{y}^k$ will be generated in this step (see Section 3.4).

Step 5. *Convergence Test and Move.* If $||y^k - \hat{y}^k|| \leq \varepsilon$, stop. Otherwise, set $y^{k+1} = \hat{y}^k$ and $k = k+1$, go to Step 2.

In Step 2 of the **SORNA** algorithm, an NCP (nonlinear complementarity problem)-based UE can be solved directly (for small to medium size problems) or by applying certain decomposition schemes (for large size problems). More discussions on this are provided in [21]. The method of constructing $S(y)$ is discussed in detail in Section 3.2. In Step 3, the sampling method is based on the Hit-and-Run (HR) approach originally developed in [13] for sampling over a full dimensional subset in R^n. We extend in Section 3.3 the original approach to sample over a subspace of R^n. The process of the two-phase simulation optimization in Step 4 is discussed in [12, 11]. Here we directly use the WISOPT solver developed in [11] to perform the subregion identification and optimization. In this article, we set M_k a constant for different y_k's. However, we test different M_k values to see how this may impact the performance of the SORNA algorithm. In Step 5, ε is a user-defined threshold for convergence tolerance. We use 10^{-5} in this article.

3.2 Characterizating the Solution Set of An Affine UE

To illustrate **SORNA**, we focus on affine UEs in this paper due to the ease of characterizing the solution set of an affine UE [5] which is based on characterization of solution sets of convex programs [22, 23, 24] and NCPs/VIs [19]. The method presented in this section has also been discussed in detail in [5], but is presented here for completeness. For an affine UE, the link travel time t is a linear function of total link flow x, i.e. we can define t as:

$$t(x) = \alpha x + \beta. \tag{12}$$

Here $\beta \in \mathbb{R}^{|A|}$ is a vector of link free flow travel times and $\alpha \in \mathbb{R}^{|A|} \times \mathbb{R}^{|A|}$ is a matrix representing the link interactions among different links. In other words, its entry $\alpha_{a,b}$ represents the contribution of traffic flow of link b to the travel time of link a. Therefore, we would expect all the elements of matrix α are non-negative. In particular, its diagonal entries should all be positive since as flow increases on a link, its travel time should always increase monotonically. Further, if α is a symmetric matrix, so is the resulting UE, i.e., there is no link interaction or the interactions are symmetric. Otherwise, link interactions are asymmetric and so is the resulting UE [25].

To derive an explicit expression for $S(y)$, we adopt the NCP formulation for UE as in [5]. For this purpose, denote Q the set of destinations of the network and $q \in Q$ as a given destination. Denote v_a^q the flow on link a with respect to the destination q, and $v^q = (v_a^q)_{\forall a \in A}$. This implies

$$x = \sum_{q \in Q} v^q. \tag{13}$$

Equation (12) can then be rewritten as:

$$t(\sum_{q \in Q} v^q) = \alpha(\sum_{q \in Q} v^q) + \beta, \tag{14}$$

Assume d_i^q the demand from node $i \in N$ to destination q (we conventionally set $d_q^q = 0$), π_i^q the minimum travel time from node $i \in N$ to destination q (we conventionally set $\pi_q^q = 0$), and $d^q = (d_i^q)_{\forall i \in N, i \neq q}, \pi^q = (\pi_i^q)_{\forall i \in N, i \neq q}$. Further denote Λ the node-link incidence matrix of the network and Λ_q the matrix with the row corresponding to destination q removed from Λ. As shown in [5], $UE(y)$ can be formulated as the following NCP as trying to find a pair (v, π) for $v = (v^q)_{\forall q \in Q}, \pi = (\pi^q)_{\forall q \in Q}$ such that

$$NCPUE(y) \qquad 0 \leq (\Lambda_q v^q - d^q) \perp \pi^q \geq 0, \forall q \in Q, \tag{15}$$

$$0 \leq (-\Lambda_q^T \pi^q + t(\textstyle\sum_{q \in Q} v^q) + y/\theta) \perp v^q \geq 0, \forall q \in Q. \tag{16}$$

Here "$\perp$" reads as "perpendicular", i.e. $x \perp y \leftrightarrow x^T y = 0$. The above model is denoted as *NCPUE(y)* to represent the NCP based UE model under toll vector y. Equation (15) represents the flow conservation at nodes of the network for a specific destination q, while equation (16) is for the route choice condition at nodes of the network. Detailed discussions of the model can be found in [26]. Assume $\bar{u} = (\bar{\pi}^T \ \bar{v}^T)^T$ is a known solution to $NCPUE(y)$, i.e., $\bar{u} \in S(y)$. Then the solution set $S(y)$ can be represented as follows [5]:

$$S(y) = \Big\{ x = \sum_{q \in Q} v^q | \ \exists (\pi^T \ v^T)^T \geq 0 \tag{17}$$

$$\Lambda_q v^q - d^q = 0, \forall q \in Q, \tag{18}$$

$$-\Lambda_q^T \pi^q + \alpha(\sum_{q \in Q} v^q) + \beta + y/\theta \geq 0, \forall q \in Q, \tag{19}$$

$$(\alpha + \alpha^T)(\sum_{q \in Q} v^q - \sum_{q \in Q} \bar{v}^q) = 0, \tag{20}$$

$$-\sum_{q \in Q} (d^q)^T (\pi^q - \bar{\pi}^q) + (\beta + y/\theta) \sum_{q \in Q} (v^q - \bar{v}^q) = 0 \Big\}. \tag{21}$$

The solution set $S(y)$, as represented by (17) - (21), is a nonempty polyhedron, defined on disaggregated variables (v, π). If α is a diagonal matrix (i.e., no link interaction exists), since all its entries are positive, α is positive definite. This implies that *NCPUE(y)* has a unique solution in terms of total link flow. In this case, the three SBTP design approaches are the same and will produce the same solution since the upper level objective function is defined on total link flows. However, if α is not a diagonal matrix (i.e., link interaction does exist), multiple solutions may exist when α is not positive-definite.

3.3 Random Sampling from the UE Solution Set $S(y)$

Sampling over a convex set in R^n to follow a certain distribution is a classical problem in operations research. In [13], Smith proposed the "hit-and-run" algorithm, a Monte-Carlo process to uniformly sample points from a *full dimensional* convex set of R^n. Denote such a set as $W \in R^n$. The hit-and-run (HR) algorithm for generating M uniformly distributed sample points can be summarized as follows [13]:

HR Algorithm

Step 1. Choose a starting point $w_0 \in W$ and set $m = 0$.

Step 2. Generate a random direction p uniformly distributed over a direction set $P \in R^n$.

Step 3. Find the line set $L = W \cap \{w|w = w_m + \lambda p\}$ and generate a random point w_{m+1} uniformly distributed over L.

Step 4. If $m = M$, stop. Otherwise, set $m = m + 1$ and go to Step 2.

In the above algorithm, since W is full-dimensional, the direction set P in Step 2 can be straightforwardly generated, e.g. chosen as a random point on the unit sphere $P = \{p| \parallel p \parallel = 1\}$. However, the UE solution set $S(y)$ is more complicated as it is a composite set defined on disaggregated variables v, π. We denote $u = (v^T, \pi^T)^T$, and u_v and u_π the v- and π-component of u respectively. Then we have $v^q = u_v^q$ and we can rewrite the representation of $S(y)$ in (17) - (21) as the following for a fixed vector y:

$$S(y) = \{x = \sum_{q \in Q} u_v^q | Bu = b, Cu \leq c\}. \tag{22}$$

Here B, C are matrices and b, c are vectors, all with appropriate dimensions. In particular, $Bu = b$ is used to compactly represent (18), (20), and (21), whereas $Cu \leq c$ is used to represent (17) and (19). Due to the special structure of $S(y)$, we sample over the disaggregated variable space u in this article instead of the total link flow space x. In particular, we uniformly sample u over $U(y)$ defined as:

$$U(y) = \{u|Bu = b, Cu \leq c\}. \tag{23}$$

The summation of $x = \sum_{q \in Q} u_v^q$ will then follow a uniform distribution over $S(y)$. Next we show how we can uniformly sample over $U(y)$.

First, $U(y)$ is a subset of the following set

$$U_1 = \{u|Bu = b\}, \tag{24}$$

which is a translate of the following subspace in R^n:

$$U_0 = \{u|Bu = 0\}. \tag{25}$$

Since $U(y)$ is a subset of the translate of a subspace in R^n, the original HR algorithm cannot be used directly. The reason is that the direction p generated in Step 2 of the HR algorithm is uniformly distributed over the full dimension of R^n and therefore the chance that p lies in U_1 as defined in (24) is very small. As a result, there is no guarantee that the generated samples in Step 3 of the HR algorithm are within $U(y)$.

Obviously, to sample over $U(y)$ using the HR algorithm, we need to generate the random direction p so that it is always in U_1. Notice that for $u \in U_1, u_0 \in U_0$, we always have $u + u_0 \in U_1$. In other words, if we can generate a uniformly distributed random direction in U_0, the random direction in U_1 can be readily constructed. Therefore, the question now is how to generate a uniformly distributed direction in the subspace U_0. This can be done by generating the random direction only based on the basis of the subspace U_0.

It is easy to see that $U_0 = Null(B)$, where $Null(B)$ denotes the *null space* of matrix B. Constructing the null space of a matrix is a standard operation (e.g. in Matlab, $null(B)$ produces the basis of the null space of matrix B). The random direction can then be generated randomly from the basis.

In Step 3 of the algorithm, the line set can be generated by calculating the intersecting points of line $u_m + \lambda p$ with the boundaries of $U(y)$ represented by $Cu \leq c$ [27]. In other words, the equality constraints of $U(y)$ determine the random direction of the HR algorithm, while the inequality constrains determine the random step size. We summarize the above modifications to the original HR algorithm in [13] as the following Extended Hit-and-Run (EHR) algorithm:

EHR Algorithm

Step 1. Construct matrices B, C, and vectors b, c from the representation of $U(y)$ in (17) - (21). Denote $u_0 = (v_0, \pi_0) \in U(y)$ as the disaggregated solution of the NCP-based UE formulation (15) - (16). set $m = 0$.

Step 2. Construct the basis of the null space of B. Assume its rank is k and the basis is $h_1^0, h_2^0, \ldots, h_k^0$.

Step 3. Generate k random numbers from the standard normal distribution $N(0,1)$: $\gamma_1, \gamma_2, \ldots, \gamma_k$. The random direction can then be constructed as

$$p_m = \frac{\sum_{i=1}^{k} \gamma_i h_i^0}{\sqrt{\sum_{i=1}^{k} \gamma_i^2}} + u_0. \tag{26}$$

Step 4. Compute the following two scalars [27]:

$$\lambda_{min} = \max \quad \Big\{ \frac{c_i - \sum_{j=1}^{n} C_{ij} x_m^j}{\sum_{j=1}^{n} C_{ij} p_m^j}, \forall i \text{ such that } \sum_{j=1}^{n} C_{ij} p_m^j < 0 \Big\}. \tag{27}$$

$$\lambda_{max} = \min \quad \Big\{ \frac{c_i - \sum_{j=1}^{n} C_{ij} x_m^j}{\sum_{j=1}^{n} C_{ij} p_m^j}, \forall i \text{ such that } \sum_{j=1}^{n} C_{ij} p_m^j > 0 \Big\}. \tag{28}$$

Generate a random scalar λ_m that follows the uniform distribution defined on the range $[\lambda_{min}, \lambda_{max}]$.

The new sample can then be constructed as $u_{m+1} = u_m + \lambda_m p_m$.

Step 5. If $m = M$, stop. Otherwise, set $m = m + 1$ and go to Step 3.

In Step 3 of the EHR algorithm, the random direction is generated as a linear combination of the basis of U_0. The weights follow standard normal distribution. As shown in [28], the direction generated this way does follow uniform distribution in the subspace. Therefore p_m as expressed in (26) is a uniformly distribution direction in U_1. Furthermore, as shown in [27], λ_{min} and λ_{max} represent the intersecting points of the line $u_m + \lambda p_m$ with the boundaries of $U(y)$, which are calculated using the inequality constraints of $U(y)$, i.e. $Cu \leq c$. This implies L is the line segment between $u_m + \lambda_{min} p_m$ and $u_m + \lambda_{max} p_m$, which indicates that $u_{m+1} = u_m + \lambda_m p_m$ is a uniformly distributed point along line set L as long as λ_m is uniformly distributed over $[\lambda_{min}, \lambda_{max}]$. The EHR algorithm generates random samples in the space of disaggregated variables, i.e. $U(y)$. From the samples, the total link flow x can be computed using $x = \sum_{q \in Q} u_v^q$.

3.4 Two-Phase Simulation Optimization

We use the optimization package WISOPT (WIsconsin Simulation OPTimization) that is based on a two-phase framework incorporating different optimization methodologies. See Figure 2 for a flow chart.

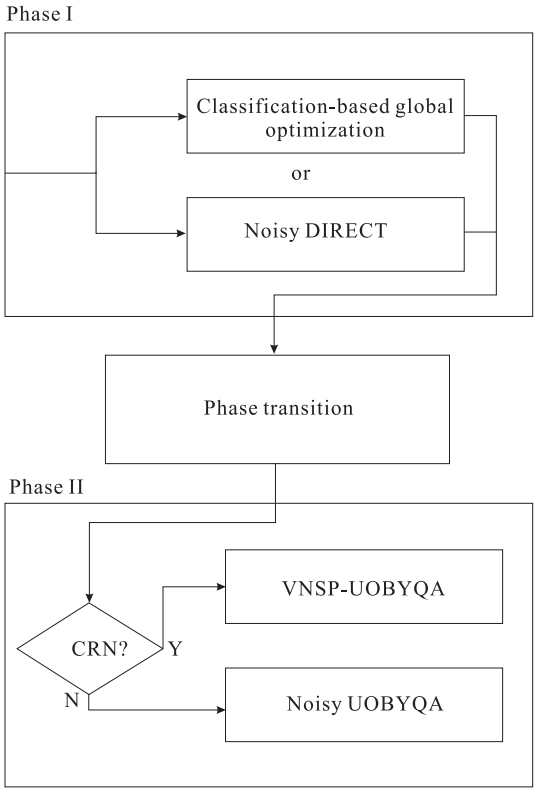

Figure 2: The two-phase WISOPT structure.

Phase I is a global exploration step over the entire domain. Algorithms in Phase I call Step 2 of the SORNA algorithm to generate (and evaluate) densely distributed samples in promising subregions and sparsely distributed samples in inferior subregions. The entire set of samples is then passed to a phase transition procedure, which implements a non-parametric statistical method to determine starting points and surrounding subregions for the multistart Phase II optimization techniques.

One of the Phase I methods employs classification tools to facilitate the global search process. By learning a surrogate from existing data the approach identifies promising subregions and generates dense samples in these regions. Another Phase I method is the Noisy DIRECT (DIviding RECTangles) algorithm, which is an extension of the DIRECT optimization algorithm.

Phase II performs local derivative-free optimization based on the UOBYQA (Unconstrained Optimization BY Quadratic Approximation) algorithm, in each of the identified subregions. If we can implement common random numbers (CRN) in the simulator, the VNSP-UOBYQA (Variable-Number Sample-Path UOBYQA) algorithm is used, while the Noisy UOBYQA algorithm is carried out in the white noise case. Both algorithms apply Bayesian techniques to guide appropriate sampling strategies while simultaneously enhancing algorithmic efficiency to obtain solutions of a

desired accuracy.

Classification-based global optimization A good surrogate model for the entire space may require a large amount of simulations and can be very expensive to compute. Since Phase I only attempts to determine promising subregions of the search space, we employ a simple indicator function:

$$I(x) = \begin{cases} 1, & \text{for } x \text{ in a promising subregion} \\ 0, & \text{otherwise,} \end{cases} \tag{29}$$

where promising subregions in the method correspond to appropriately chosen level sets of F. Approximating the indicator function I is simpler than approximating the underlying function F, especially in high dimensional cases.

We generate a boosting classifier [29] to approximate I. This classifier is used to predict whether new samples (potential evaluation points) are within promising regions or not, facilitating further evaluations of the underlying F. Figure 3 illustrates the promising local regions of a (multimodal) test function due to Griewank (the dotted circles and the level sets) and the generated samples of the algorithm (the '+'s). The method is relatively insensitive to noise, because the simplification step (29) smooths out the occurrence of noise. Therefore, we normally do not use replicated samples in training the classifiers.

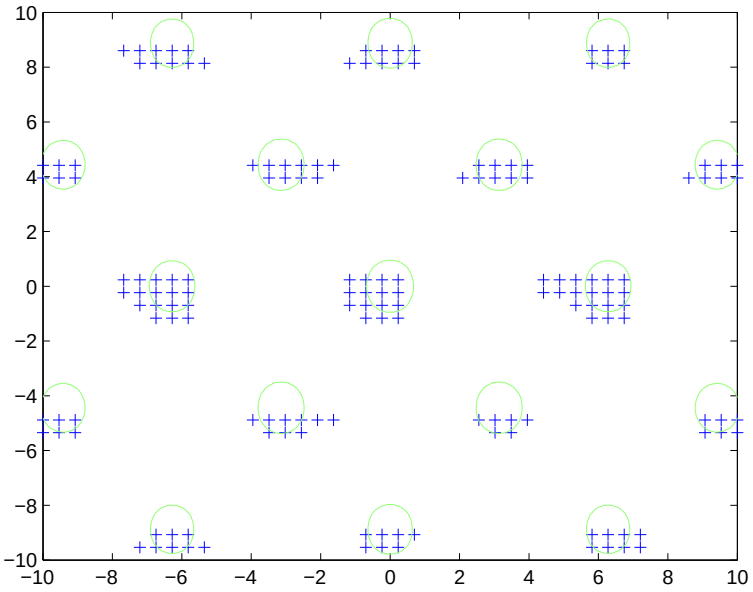

Figure 3: Predicted local subregions of the Griewank function. The function has local optimums in each subregion (circles) and the global optimum at $[0, 0]$.

The Noisy DIRECT algorithm The DIRECT optimization method [30, 31, 32, 33] is a deterministic global optimization algorithm for bound-constrained problems. The algorithm centers around a space-partitioning scheme in which promising hyperrectangles are subject to further division. Figure 4 provides an illustration of the algorithm on a test function due to Goldstein Price.

The algorithm therefore proceeds to gradually explore promising subregions. In other words, more function evaluations will be performed near a (local) optimal solution.

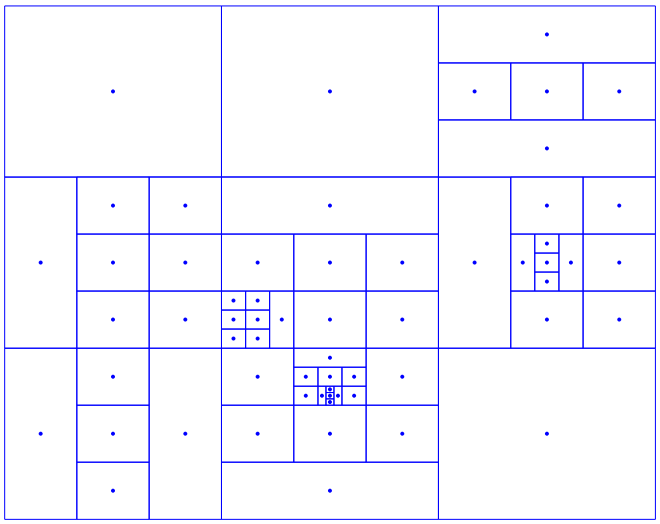

Figure 4: The DIRECT optimization algorithm on the Goldstein Price function.

When the objective function is subjected to uncertainty, some crucial operational steps of the DIRECT algorithm are affected. For example, the choice of potentially optimal hyperrectangles becomes incorrect because of the noisy function values, possibly misleading the algorithm to search in inferior regions. We modify the original DIRECT algorithm using a simple approach - multiple replications are sampled to reduce output uncertainty. Since the objective function is often computationally expensive to evaluate, we must be very cautious in using function evaluations. In our modification, we apply Bayesian techniques to derive a posterior distribution for the function output at each point, and incorporate the distribution information into the algorithm to determine an appropriate number of replications to be used.

The phase transition Using the evaluated samples in Phase I, the phase transition procedure consists of a non-parametric local quadratic regression method to determine the appropriate subregion size. Unlike regular regression methods which use the entire set of samples in the domain to construct one model, local regression makes a prediction (at a point) using a local model based on samples within a 'window size', thus the approach values the local behavior of a function more. 'Non-parametric' means the regression model is not from a single parametric family. It is presumed that the samples outside the local region have a slight relevance to the current prediction. In our procedure, we treat the resulting 'window size' as our subregion radius.

A sequence of good starting points is generated, satisfying the criteria: (a) each starting point is the center of a subregion, (b) the subregions are mutually separated. The sequence of starting points and the subregion sizes are passed to Phase II for local processing, possibly in a parallel setting.

Extended UOBYQA algorithms In Phase II, the deterministic UOBYQA algorithm is applied as the base local search method and is extended for noisy function optimization. The method is an iterative algorithm in a trust region framework [34], but it differs from a classical trust region method in that it creates a chain of local quadratic models by interpolating a set of sample points instead of using the gradient and Hessian values of the objective function (thus making it a derivative-free tool).

We developed variants of the original UOBYQA, called the VNSP-UOBYQA and the Noisy UOBYQA, that have been adapted for noisy optimization problems. The extension idea is similar to that of the Noisy DIRECT algorithm. We sample multiple replications per point to reduce variance and apply Bayesian techniques to guide appropriate sampling strategies to estimate the objective function. The two algorithms employ different mechanisms in the sampling process. The VNSP-UOBYQA determines appropriate replication numbers by whether sufficient reduction is identified in the trust-region subproblem, while the Noisy UOBYQA determines the number by whether the quadratic models can be shown to be stable or not. Generally speaking, when CRN is implemented, the noise is relatively easy to handle because it is correlated among sites.

4 Numerical Example

To illustrate the risk-neutral SBTP scheme and the **SORNA** solution algorithm, we apply the algorithm to a small test example in this section. It turns out that, as shown in Section 2, the problem can be solved analytically and exactly, which can be used to analyze the effectiveness of the solution approach.

Figure 5 depicts a hypothetical network with one origin-destination (OD) pair (from node r to node s) and three routes. A toll booth is located at the very beginning of route 2 and 3. The distance between node r and i is very small so that the travel time can be ignored (assume toll is automatically collected and therefore the delay at the toll booth can be ignored as well). Further assume the total demand $d = 10$ and the route (also link) flows are x_1, x_2, and x_3. The travel times of the links are assumed to have the following form:

$$\begin{aligned} t_1 &= 2x_1 + x_2 + x_3 \\ t_2 &= 2x_2 + 2x_3 \\ t_3 &= 2x_2 + 2x_3. \end{aligned}$$

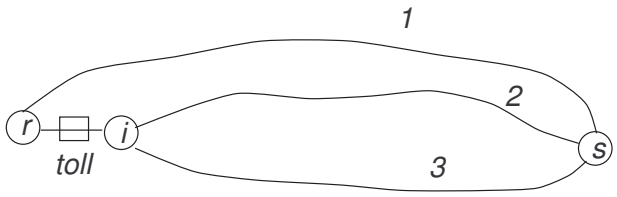

Figure 5: The Test Network

In other words, link interactions do exist among the three links. For simplicity, we assume the "value of time" $\theta = 1$. Then the link generalized travel times, with toll imposed, are:

$$
\begin{aligned}
c_1 &= \quad t_1 \\
c_2 &= \quad t_2 + y \\
c_3 &= \quad t_3 + y.
\end{aligned}
$$

Here y is the toll and $y \in K_y = \{y | 0 \leq y \leq 15\}$. Denote $c = (c_1, c_2, c_3)^T$ and $x = (x_1, x_2, x_3)^T$. To determine the "optimal" toll, we assume the objective function for the upper level as follows:

$$f(y, x) = t_1 x_1 + 3t_2 x_2 + t_3 x_3. \tag{30}$$

We first apply **SORNA** to solve the risk-neutral model *RNSBTP* for the test example. We set the number of samples for each given toll (Step 3 of **SORNA**) as $M_k = M = 300$. The algorithm converges after 8 iterations with an obtained solution 11.12 and the corresponding objective value is 154.37. The results are very close to the optimal solution (11) and the optimal objective value (155) as shown in Appendix A. The deviations are only 1.1% and 0.4% respectively. Figure 6 depicts the convergence of the SORNA algorithm.

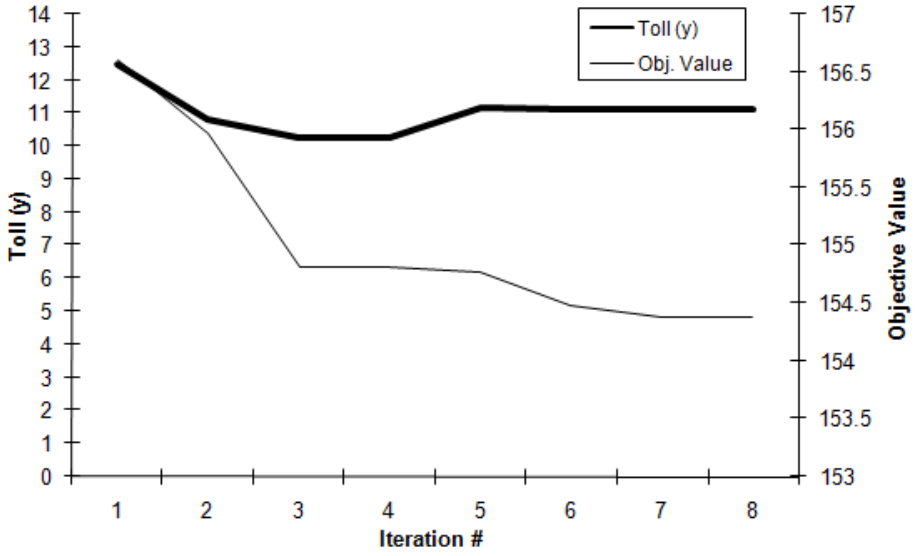

Figure 6: Convergence of the SORNA Algorithm

We now check whether the random sampling algorithm **EHR** works properly. We need to answer two questions: 1) are the generated samples within the solution set $S(y)$? and 2) do the samples follow a uniform distribution within $S(y)$? Here we use $y = 11$, i.e. the optimal toll, as a test case. First, based on (32) in Appendix A, we know that the solution set when $y = 11$, i.e. $S(11)$, can be expressed as:

$$S(11) = \{x = (x_1, x_2, x_3)^T | x_1 = 7, \quad x_2 + x_3 = 3\}. \tag{31}$$

The x_1 component of all $M = 300$ samples generated by the EHR algorithm is 7. Figure 7 further depicts the (x_2, x_3) component of the samples using plus signs. The solid line represents the "theoretical" line of $x_2 + x_3 = 3$, which (x_2, x_3) should follow. It is clear that all samples lie

on the theoretical line, indicating that the random samples are within $S(11)$ as defined in (31). To answer the second question, we notice that if x_2 follows the uniform distribution within $[0, 3]$, the generated samples will indeed follow the uniform distribution within $S(11)$. We plot in Figure 8 the histogram of the x_2 component of the 300 randomly generated samples. We particularly arrange the samples into 10 bins. The bold solid line in the figure is the "theoretical" (uniform) distribution that x_2 should follow. The thin solid line with plus signs is the histogram calculated by the generated samples. We can see that although there are significant variations, the histogram does follow and fluctuate over the theoretical line. This means that the generated samples do follow, approximately, a uniform distribution.

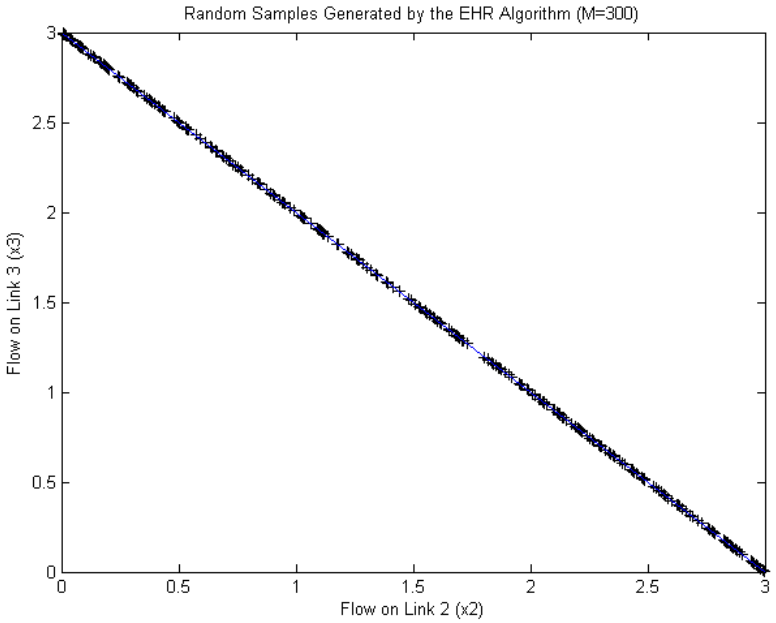

Figure 7: Random Samples Generated by the EHR Algorithm

To show how the sampled objective values compare with the *true* objective values for a given toll, we show in Figure 9 the sampled objective values vs. toll y. In this figure, the dotted line and the solid line with triangle represent respectively the average objective value of all the samples for $M = 3$ and $M = 50$. The bold solid line is the true objective value, i.e. calculated via equation (34) in Appendix A. We can see that the average objective value can approximate the true objective curve very well, in spite the fact that the average value has variations at different y's. We can also observe that as toll y becomes larger, the variation becomes smaller. This is because, as can be seen from the solution set $S(y)$ in (32), the range of the solution set becomes smaller as y increases, leading to smaller variation of the objective value.

The two-phase simulation optimization algorithm usually generate more samples in promising subregions and fewer samples in inferior subregions. This means that when a toll close to the optimal solution is evaluated, more function evaluations will be generated and evaluated. To see if this is the case for the test example, Figure 10 depicts the number of function evaluations vs. toll y. The plot confirms that as the toll is closer to the optimal solution ($y = 11$), denser samples are generated and more function evaluations are performed by the two-phase optimization algorithm.

The value of M is an important factor to the solution algorithm. Intuitively, a larger M will generate more samples for a given toll which may produce more accurate approximation to the

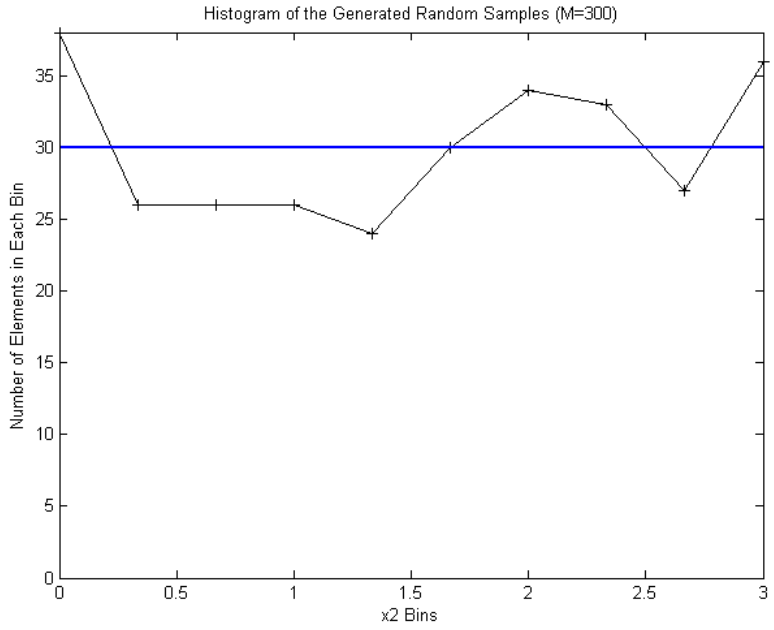

Figure 8: Histogram of Generated Random Samples

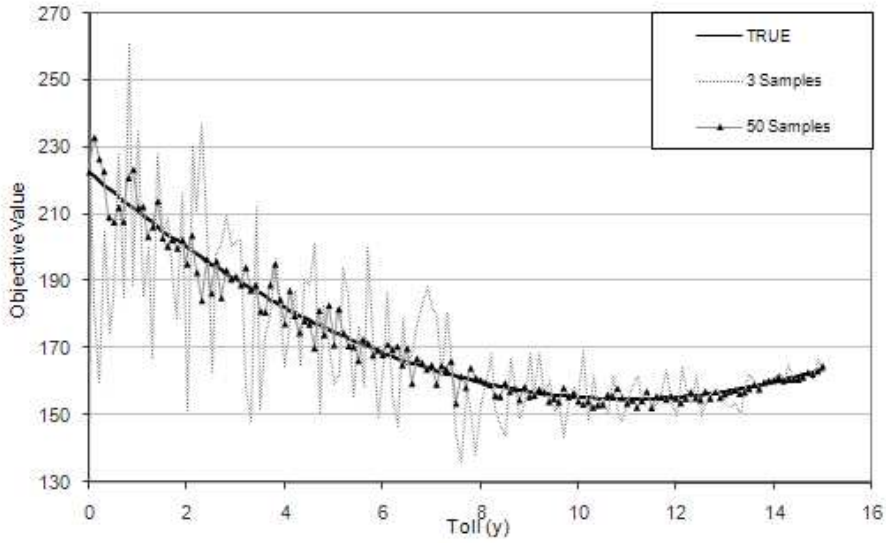

Figure 9: Sampled and True Objective Values

underlying true distribution. On the other hand, however, larger M's will also require more computational time to perform the sampling, which may not be appropriate for large scale problems. Therefore, a proper M will be most likely problem-specific, which will reflect the user's tradeoff between the solution quality and the available computational resources. To see how different M's may impact the solution quality, we show in Table 1 how the solution and objective value may change as the value of M varies. Notice that for this small example, the computational efforts is less interesting, which can be reasonably represented by the actual values of M (i.e. $M = 100$

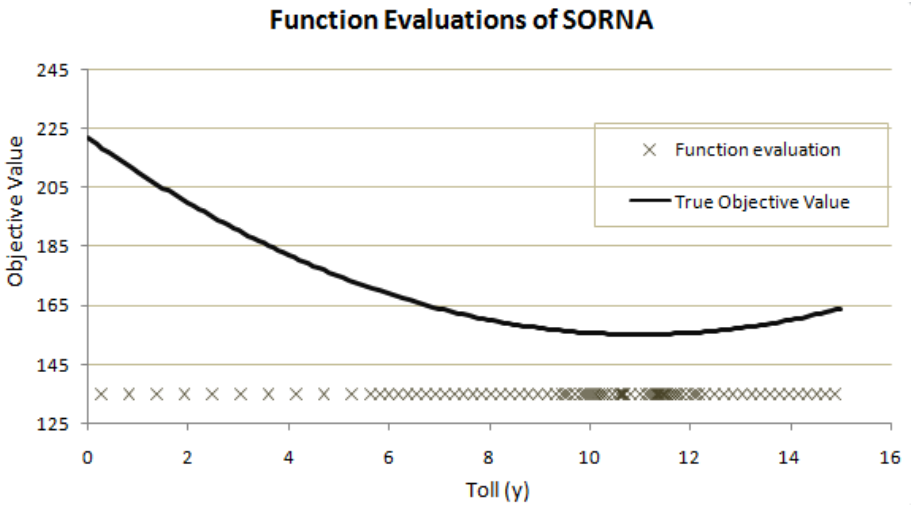

Figure 10: Function Evaluation of Two-Phase Simulation Optimization

roughly requires twice of the computational time used by $M = 50$).

		Value of M		
	30	**50**	**100**	**300**
Obtained sol	10.53	10.71	10.59	11.12
Optimal sol.	11.00	11.00	11.00	11.00
Diff. with opt. sol. (%)	4.32	2.64	3.76	1.10
Obtained obj	148.53	151.91	153.30	154.37
Optimal obj.	155.00	155.00	155.00	155.00
Diff. with opt. obj. (%)	4.18	1.99	1.09	0.40

Table 1: Impact of the Number of Samples (M) on Solution Quality

We can see from the table that as M increases, both the obtained solution (toll) and the objective value become closer to the optimal solution and the optimal objective value. In particular, the deviation for the obtained toll decreases from 4.32% to 1.1%, while the deviation for the the obtained objective value decreases from 4.18% to 0.4%. This indicates that increasing the number of samples at each iteration does produce more accurate approximation to the underlying distribution, and as a result, a better solution.

Finally, to compare the results of the three SBTP design approaches, we depict in Figure 11 the changes of the objective value as a function of the imposed toll, for the risk-prone, risk-neutral, and risk-averse approaches. The curves are based on the analytical results of the three SBTP approaches (see [5] and Appendix A of this article). The figure confirms that the risk-neutral solution and objective value are in-between those of the risk-prone and risk-averse design approaches. This indicates that the risk-neutral approach is less aggressive than the risk-prone scheme and less conservative than the risk-averse scheme.

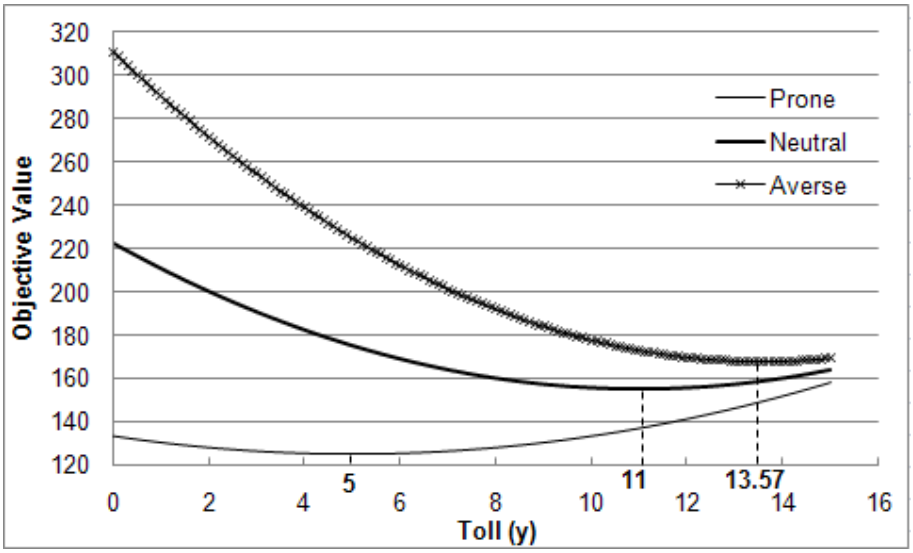

Figure 11: Comparison of Three SBTP Design Approaches

5 Conclusion

We proposed a risk-neutral scheme for the SBTP design to account for the possible nonuniqeness of the UE solution. The scheme aims to minimize the expected objective value as the UE solution varies over the solution set. The scheme provides an alternative way, from the toll designer's perspective, for addressing the uncertainty due to nonunique UE solutions. Therefore, the proposed risk-neutral scheme complements the traditionally used risk-prone SBTP scheme and the risk-averse scheme we recently developed [5].

We showed that the risk-neutral scheme can be formulated as a stochastic program, which extends recent simulation-based optimization methods studied in [11, 12]. The stochastic program can be solved iteratively via three major steps: characterization of the UE solution set, uniform sampling over the UE solution set, and a two-phase simulation optimization algorithm using the random samples. To sample uniformly over the UE solution set, we extended the Hit-and-Run (HR) sampling algorithm in [13] from a full dimensional subset in R^n to a subset of a subspace in R^n. We tested the model and solution algorithm on a small example. The results showed that 1) the extended sampling algorithm works well as it only generates samples within the UE solution set and the generated samples follow an approximate uniform distribution (Figures 7 and 8); 2) the solution algorithm can produce an approximate model that matches well the general shape and trend of the true objective function (Figure 9), and the algorithm works properly for the test example (Figure 6); 3) the number of samples generated for each iteration (i.e. M) plays an important role in the quality of the obtained optimal toll (Table 1); and 4) when the UE solution is not unique, the three design schemes could indeed produce quite different optimal toll solutions and objective values. In general, the risk-neutral toll is in-between the risk-prone and risk-averse toll; so is the risk-neutral objective value (Figure 11).

By introducing the concept of toll designer's risk-taking, one has alternative ways (i.e. the risk-averse and risk-neutral schemes) to address the uncertainty due to nonunique UE solutions. There are several questions in this direction however that still remain unresolved. Some of them are summarized as follows:

(a) The efficiency of the random sampling algorithm needs to be further improved. As shown in Figure 8, there are still significant variations in the histogram even with 300 random samples. Investigations on sampling methods that can converge faster should be conducted in future research.

(b) In this article, we assumed that the realization of the UE solutions follows a uniform distribution over the solution set. In practice, some UE solutions may be unstable [35], implying that the probability of these solutions being realized is small. As a result, constructing specific distributions accounting for the UE solution stability is an interesting topic.

(c) The solution algorithm for the risk-neutral model requires an explicit expression of the solution set of the lower level UE. Although such an expression can be readily constructed for affine UEs, extending the results to general UEs requires further research.

(d) We focus on link-based UE solutions in this article. Under situations when path costs are nonadditive or path-based tolling is needed (e.g. for the purpose of controlling emissions), Path-based formulations are necessary. As path-based UE solutions are nonunique in general even when the link travel time functions are separable and strictly monotone [18], the proposed risk-neutrl SBTP schemes are expected to play more significant roles in path-based tolling or other path-based network design applications. Research in this direction will be pursued in the future.

References

[1] S. Lawphongpanich and D. Hearn. An MPEC approach to second-best toll pricing. *Mathematical programming B*, 101:33–55, 2004.

[2] S. Lawphongpanich, D.W. Hearn, and M.J. Smith. *Mathematical And Computational Models for Congestion Charging*. Springer, 2006.

[3] H. Yang and H.J. Huang. *Mathematical and Economic Theory of Road Pricing*. Elsevier, 2005.

[4] H.M Zhang and Y.E. Ge. Modeling variable demand equilibrium under second-best road pricing. *Transportation Research B*, 38:733–749, 2004.

[5] X. Ban, S. Lu, M.C. Ferris, and H. Liu. Risk-averse second-best toll pricing. In W.H.K. Lam, S.C. Wong, and H.K. Lo, editors, *Transportation and Traffic Theory*, pages 197–218. Springer, 2009.

[6] A. Ben-Tal and A. Nemirovski. Robust optimization - methodology and applications. *Mathematical Programming, Series B*, 92:324–343, 2002.

[7] P. Tseng. Fortified-descent simplicial search method: a general approach. *SIAM Journal of Optimization*, 10(1):269–288, 1999.

[8] Z.Q Luo, J.S. Pang, and D. Ralph. *Mathematical Programs with Equilibrium Constraints*. Cambridge University Press, 1996.

[9] P. Loridan and J. Morgan. Approximate solutions for two-level optimization problems. In K. Hoffman, J. Hiriart-Urruty, C. Lemarechal, and J. Zowe, editors, *Trends in Mathematical Optimization*, pages 181–196. 1988.

[10] Y. Yin and S. Lawphongpanich. A robust approach to the continuous network design problem with demand uncertainty. In M. Bell, B. Heydecker, and R. Allsop, editors, *17th International Symposium on Transportation and Traffic Theory*, pages 79–110. 2007.

[11] G. Deng. *Simulation-Based Optimization.* PhD thesis, Mathematics and Computation in Engineering, University of Wisconsin at Madison, 2007.

[12] G. Deng and M.C.Ferris. Variable-number sample-path optimization. *Mathematical Programming, in press*, 2007.

[13] R. L. Smith. Efficient Monte Carlo procedures for generating points uniformly distributed over bounded regions. *Operations Research*, 32(6):1296–1308, 1984.

[14] H.K. Lo and A. Chen. Traffic equilibrium problem with route-specific costs: formulation and algorithms. *Transportation Research, Part B*, 34:493–513, 2000.

[15] S. Gabriel and D. Bernstein. The traffic equilibrium problem with nonadditive path costs. *Transportation Science*, 31:337–348, 1997.

[16] A. Chen, H.K. Lo, and H. Yang. A self-adaptive projection and contraction algorithm for the traffic assignment problem with path-specific costs. *European Journal of Operations Research*, 135:27–41, 2001.

[17] R.P. Agdeppa, N. Yamashita, and M. Fukushima. The traffic equilibrium problem with nonadditive costs and its monotone mixed complementarity problem formulation. *Transportation Research, Part B*, 41:862C874, 2007.

[18] M. Patriksson. *The Traffic Assignment Problem, Models and Methods.* VSP, 1994.

[19] F. Facchinei and J.S. Pang. *Finite-Dimensional Variational Inequalities and Complementarity Problems: Vol. I, II.* Springer, 2003.

[20] M.J. Smith. The existence, uniqueness and stability of traffic equilibria. *Transportation Research, Part B*, 13:295–304, 1979.

[21] X. Ban, H.X. Liu, and M.C. Ferris. A link-node based complementarity model and its solution algorithm for asymmetric user equilibria. In *Proceedings of the 85th Annual Meeting of Transportation Research Board (CD-ROM)*, 2006.

[22] O.L. Mangasarian. A simple characterization of solution sets of convex program. *Operations Research Letters*, 7(1):21–26, 1988.

[23] J.V. Burke and M.C. Ferris. Characteration of solution sets of convex programs. *Operations Research Letters*, 10:57–60, 1991.

[24] J.V. Burke and M.C. Ferris. Weak sharp minima in mathematical programming. *SIAM Journal on Control and Optimization*, 31:1340–1359, 1993.

[25] Y. Sheffi. *Urban Transportation Networks: Equilibrium Analysis with Mathematical Programming Methods.* Prentice-Hall, Inc., 1985.

[26] X. Ban, M.C. Ferris, and H.X. Liu. An MPCC formulation and numerical studies for continuous network design with asymmetric user equilibria. *Submitted for publication*, 2007.

[27] Z.B. Zabinsky. *Stochastic Adaptive Search for Global Optimization.* Kluwer Academic Publishers, 2003.

[28] D.E. Knuth. *The art of Computer Programming, Vol. 2.* Addison-Wesley, 1969.

[29] M. C. Ferris and G. Deng. Classification-based global search: An application to a simulation for breast cancer. In *Proceedings of the NSF CMMI Engineering Research and Innovation Conference*, 2008.

[30] D. E. Finkel. DIRECT optimization algorithm user guide. Technical Report Technical Report CRSC-TR03-11, Center for Research and Scientific Computation, North Carolina State University, 2003.

[31] D. E. Finkel. *Global Optimization with the DIRECT Algorithm*. PhD thesis, North Carolina State University, 2005.

[32] D. R. Jones. *The DIRECT global optimization algorithm*. Encyclopedia of Optimization, 2001.

[33] C. D. Perttunen D. R. Jones and B. E. Stuckman. Lipschitzian optimization without the Lipschitz constant. *Journal of Optimization Theory and Application*, 79(1):157–181, 1993.

[34] J. Nocedal and S. J. Wright. *Numerical Optimization*. Springer, New York, second edition, 2006.

[35] D. Watling. Stability of the stochastic equilibrium assignment problem: a dynamical systems approach. *Transportation Research, Part B*, 33:281–312, 1999.

Appendices

A Analytical Solution of the Test Example

As shown in [5], the solution set $S(y)$ of the test example in Figure 5 can be expressed as:

$$S(y) = \{x = (x_1, x_2, x_3)^T \geq 0 | x_1 = (10 + y)/3, \quad x_2 + x_3 = (20 - y)/3\}. \tag{32}$$

Clearly, for any given $y \in K_y$, $S(y)$ is a straight line (i.e., a nonempty polyhedral set) in the three dimension space $x_1 - x_2 - x_3$ (see Figure 1).

Here we want to minimize the objective function (30) over all $x \in S(y)$ that follows uniform distribution. This is equivalent to say, by (32), that x_2 follows uniform distribution in the range $[0, (20 - y)/3]$. Subsequently, the expected value of the objective for a given y will be:

$$F(y) = E_x \text{ is uniform distributed over } S(y)[f(y, x)] = \int_0^{(20-y)/3} f(y, x) dx_2. \tag{33}$$

Substituting $x_1 = (10 + y)/3$ and $x_3 = (20 - y)/3 - x_2$ into (33), we can obtain

$$F(y) = \int_0^{(20-y)/3} ((y^2 - 10y + 400)/3 + (40 - 2y)x_2/3) * 3/(20 - y) d(x_2).$$

Note that in (33), $3/(20 - y)$ is the probability density function of x_2 over its possible range $[0, (20 - y)/3]$. The above equation can be simplified as:

$$F(y) = 5(y - 11)^2/9 + 155. \tag{34}$$

Therefore, the risk neutral solution is obtained at $y_n^* = 11$, with objective value $z_n^* = 155$.

B A Larger Example

In this section, we show a larger example for which nonunique UE solutions exist such that $RPSBTP$ and $RASBTP$ produce different results. The example is based on the 3 by 3 grid network in Figure 12. The network has 9 nodes and 12 links with only one OD pair from node 1 to node 9. The total demand is assumed to be $d = 1$. For illustration purposes, we assume that the travel time for link $(1, 2)$ is $t_{1,2} = 1 + x_{1,2}$ and travel times for all other links are 2, i.e. a constant. Assuming link (2,5) passes by an environmentally sensitive region so that the toll designer would like to impose a weight 3 for its travel time; all other links have their travel time weights as 1. We install a toll booth on link $(1, 2)$ and the toll has to be within 0 and 2, i.e. $0 \leq y \leq 2$. We further assume the "value of time" $\theta = 1$. Now the question is: what are the risk-prone and risk-neutral SBTP solutions?

We first notice that there are 6 paths from the origin 1 to destination 9: denote the path $(1, 2, 3, 6, 9)$ as $P1$, and paths $(1, 2, 5, 6, 9)$, $(1, 2, 5, 8, 9)$ as $P2$ and $P3$ respectively. These three

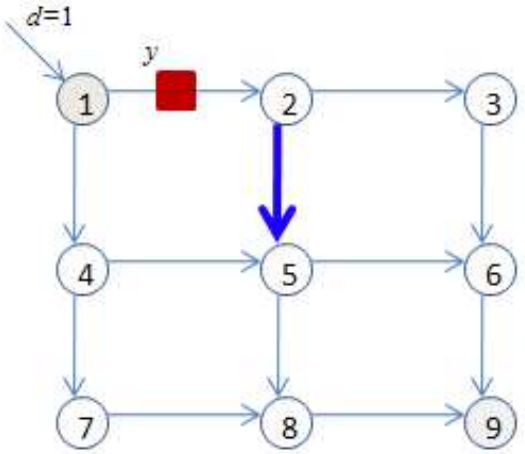

Figure 12: A Grid Network

paths traverse link $(1,2)$. Similarly, define paths $(1,4,5,6,9)$, $(1,4,5,8,9)$, and $(1,4,7,8,9)$ as $P4$, $P5$, $P6$ respectively, which all traverse link $(1,4)$. Since the travel cost on link $(1,2)$ is $1+x_{1,2}+y$, we have to consider two cases to solve the SBTP problems: $0 \leq y \leq 1$ and $1 < y \leq 2$.

If $0 \leq y \leq 1$, the equilibrium flow on link $(1,2)$ will be $x_{1,2} = 1-y$ and the flow on link $(1,4)$ is $x_{1,4} = y$. Then the risk-prone solution is obtained by assuming all flow $x_{1,2}$ will take $P1$ and flow $x_{1,4}$ could take any of the three paths $P4$ - $P6$ or any combinations of them. The resulting objective function for $RPSBTP$ is:

$$\eta(y) = y^2 - y + 8 = (y-0.5)^2 + 7.75. \tag{35}$$

The risk-averse design on the other hand assumes that $x_{1,2}$ will take either $P2$ or $P3$ or both, leading to the following equation for the $RASBTP$ objective function:

$$\Phi(y) = (y-2.5)^2 + 5.75 \tag{36}$$

To obtain the risk-neutral solution, we first notice that the upper level objective function can be expressed as: $f(y,x) = \sum_a w_a t_a x_a = \sum_a t_a x_a + 2t_{2,5}x_{2,5} = g(y,x) + 2t_{2,5}x_{2,5}$. Here $g(y,x) = \sum_a t_a x_a$ is the total system travel time, which is deterministic. To see this, notice that when $0 \leq y \leq 1$, paths $P1, P2, P3$ have the total path flow $x_{1,2} = 1-y$; they also have the same travel time: $1 + x_{1,2} + 3*2 = 8-y$. On the other hand, paths $P4, P5, P6$ have the total path flow y and the same travel time $4*2 = 8$. As a result, the total system travel time is: $g(y,x) = (1-y)*(8-y)+y*8 = y^2-y+8$. As we assume the link flow follows uniform distribution, $x_{2,5}$ will follow a uniform distribution from 0 to $1-y$. Therefore, $F(y)$ can be expressed as follows:

$$F(y) = y^2 - y + 8 + \int_0^{1-y} 2\frac{1}{1-y} w dw = y^2 - 3y + 10 = (y-1.5)^2 + 7.75 \tag{37}$$

If $1 < y \leq 2$, all OD flow will take paths $P4$ - $P6$. In this case, the three SBTP solutions coincide and the objective is a constant (8) over all $1 < y \leq 2$.

Considering the problems over the entire feasible region of y, i.e. $0 \leq y \leq 2$, we see that the three objective functions $\eta(y)$, $\Phi(y)$, and $F(y)$ are continuous (i.e. they have values 8 at $y = 1$), but not smooth at $y = 1$. This can be seen more clearly from Figure 13, which depicts how the three objective values change with toll y. The risk-prone solution for this particular example is $y = 0.5$ with the objective value 7.75; the risk averse solution is for any $1 \leq y \leq 2$ and the optimal objective value is 8; and the risk-neutral solution coincides with the risk-averse solution for any $1 \leq y \leq 2$ with the objective value 8. Figure 13 also depicts that the risk-neutral objective value is usually in-between the risk-prone and risk-averse objective values.

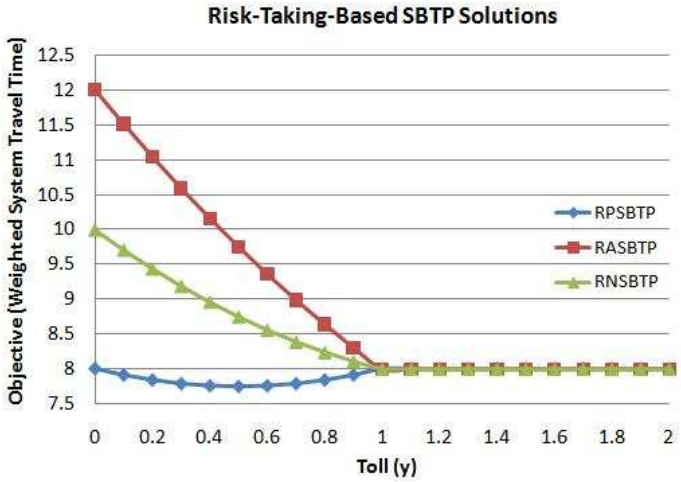

Figure 13: Comparison of Risk-Prone and Risk-Averse Solutions on the Grid Network

Urban networks have similar structures to the grid network shown in Figure 12. More importantly, due to possible link interactions that commonly exist in the urban environment (such as the interactions between left and through movements at a signalized intersection), the *separable* assumption on link travel times may not hold, implying that link travel times may not be strictly monotone with respect to link flows. Therefore, the above example, although much simplified for illustration purposes (e.g. link travel times are flow-independent except for link $(1, 2)$ and most links have an equal length), indicates that the same situation (i.e. nonunique UE solutions) may exist in some urban networks. This will result in different SBTP tolls and associated system objective values under different SBTP design approaches.