Solving Stochastic Transportation Network Protection Problems Using the Progressive Hedging-based Method

Yueyue Fan*

Assistant Professor, Department of Civil and Environmental Engineering and Institute of Transportation Studies, University of California, Davis, CA 95616. Fax: (530)752-7872, Email address: yyfan@ucdavis.edu

Changzheng Liu

PhD Candidate, Department of Civil and Environmental Engineering and Institute of Transportation Studies, University of California, Davis, CA 95616

* Corresponding author

Abstract

This research focuses on pre-disaster transportation network protection against uncertain future disasters. Given limited resources, the goal of the central planner is to choose the best set of network components to protect while allowing the network users to follow their own best-perceived routes in any resultant network configuration. This problem is formulated as a two-stage stochastic programming problem with equilibrium constraints, where the objective is to minimize the total expected physical and social losses caused by potential disasters. Developing efficient solution methods for such a problem can be challenging. In this work, we will demonstrate the applicability of progressive hedging-based method for solving large scale stochastic network optimization problems with equilibrium constraints. In the proposed solution procedure, we solve each modified scenario sub-problem as a mathematical program with complementary constraints and then gradually aggregate scenario-dependent solutions to the final optimal solution.

Key words: Stochastic programming, networks, disaster mitigation, equilibrium constraints, decomposition method

1. Introduction

Transportation systems are critical for maintaining the basic functions of a modern society. A spatially distributed transportation system can be represented as a network consisting of a set of nodes and links connecting these nodes. In this paper, we propose a method for finding the most efficient mitigation strategy for protecting a transportation network against potential hazards, which may be caused by natural disasters and human attacks. A more specific question is: given limited resources, which subset of network components should be protected so that the potential damage to the whole system is minimized. We name this problem as a stochastic network protection problem.

Two classes of stochastic network optimization problems may shed light on this work. These are the stochastic network design problems and network interdiction problems, both involving decisions on changing network topologies, uncertain future events, limited resource restriction, and standard network flow conservation constraints. A stochastic network design problem studied in the field of transportation engineering and planning is usually posed as how to expand the network capacity in order to best serve uncertain future travel demands (Magnanti and Wong, 1984; Yang and Bell, 1998; Patil and Ukkusuri, 2007). Our problem is similar in spirit to a stochastic network design problem, except that the randomness appears in different parameters. A network interdiction problem is often posed as how to interdict a network in order to block the flow on the network, which is highly relevant to military and defense applications (Woodruff, 2003). Network interdiction adopts an attacker's perspective, whereas network protection takes the planners perspective. In most network interdiction problems, flow (e.g., information and military goods) is considered to be controllable by a central commander. In our problem, network users (individual drivers) are assumed to follow their own best perceived routes, resulting in Nash equilibrium, also called as user equilibrium (UE) traffic condition (Dafermos and Nagurney, 1984). Equilibrium conditions introduce nonconvexity in a network optimization problem, thus making the problem much more difficult to solve.

In a stochastic network protection problem, one may face a wide range of possible future hazard scenarios. A common method used in practice is to separately examine all possible (or most likely) scenarios and recommend a set of scenario-specific solutions. There are two main reasons why we do not prefer such a scenario-specific method. First of all, scenario-specific policies have little relevance to policy makers. Since the future event is unknown at the time of decision making, one would not know exactly which policy to implement. Moreover, even if a representative scenario can be identified, a protection policy that works best for this particular scenario may not be even feasible for other possible scenarios, and the penalty for encountering an infeasible solution can be extremely high. Therefore, we need to find a solution that is feasible, implementable, and meanwhile achieves the best performance in an expected sense. Stochastic programming is a powerful tool for such modeling needs. It was first introduced by Dantzig (1955) to handle mathematical programming with uncertainty, and was further developed by subsequent researchers both in theory and computational aspects (Wets, 1966; Van Slyke and Wets, 1969; Wets, 1974).

The two-stage stochastic programming modeling framework suits network protection problems well, in which the protection decisions (such as whether to protect a link or not) may be considered as the first-stage decisions since these are planning decisions and must be made before a random disaster event occurs, while the network flow variables are considered as the second stage recourse variables because flow can be redistributed depending on the actual realization of the random disaster. The deterministic equivalent of this problem is a large size mixed integer nonlinear programming problem with equilibrium constraints. The large scale of the problem, the presence of integer variables, and the nonconvexity impose tremendous computational challenges. The emphasis of this work is on the development of an efficient solution method that can overcome these numerical difficulties.

Research on numerical implementation for stochastic network optimization problems without equilibrium constraints is available in stochastic programming and transportation literature (e.g., Mulvey and Vladimirou, 1991; Woodruff, 2003; Held et al, 2005; Patil and Ukkusuri, 2007). When equilibrium constraints are present, computational experiences reported in the literature are limited to deterministic problems. Among the various solution methods proposed for deterministic network design problems (Bell and Iida, 1997), we are particularly interested in methods that exploit recent advances in mathematical programming with equilibrium constraints (MPEC) and mathematical programming with complementarity constraints (MPCC) problems (Marcotte and Zhu, 1996; Patriksson and Rockafellar, 2002; Ban et al, 2006). Numerical implementation of stochastic network optimization problems with equilibrium constraints has been lacking in the field of transportation engineering.

The remainder of this paper is organized as follows. In Section 2, we introduce the notations, modeling assumptions, and the mathematical formulation of the stochastic network protection problem. In Section 3, we focus on the numerical implementation using a progressive hedging based algorithm. Results and observations gained from numerical experiments are reported in Section 4. Section 5 concludes this study with key findings, remaining limitations, and future extensions.

2. Stochastic Network Protection Problems

In a stochastic network protection problem, the future disaster and its damaging effects on the network are uncertain. Here we assume that the hazard estimation, including a set of possible future hazard scenarios and their associated probabilities, is given. With this assumption, the network protection problem is stated as: given budget constraints and hazard estimation, which set of links should be protected so that the total physical and network performance loss due to future disaster hazards is minimized? In the framework of two-stage stochastic programming, the first stage of our protection problem is to make protection decisions before the disaster happens, while the second stage is to evaluate the total loss due to a realized disaster including repair cost and increased travel delay in the network. The second stage cost (recourse cost) is a random variable dependent on the first-stage protection decision and the particular realization of link damages. Our objective is to minimize the expected loss caused by the disaster through pre-disaster protection subject to the limited budget.

Denote a transportation network as $G(N, A)$, where N is the set of nodes of size n and A is the set of network links of size m. Denote $\overline{A}$ ($\overline{A} \subset A$) as the set of candidate links that are subject to modification decisions. The size of $\overline{A}$ is $\overline{m}$. A link can be labeled by its link index as link a, or by its starting and ending node as link ij. The decision variable u_a represents the protection action on link a ($a \in \overline{A}$), which could be a continuous variable if the decision is on the amount of protection resource to be allocated to link a, an integer variable if the decision is on the level of protection efforts, or a binary variable if the decision is simply whether or not to protect link a. In the numerical implementation presented in the later sections, we focus on the discrete cases. Consider the flow on the same link but destined to different nodes as distinguished commodities. Let x_a^k (or x_{ij}^k) be the flow of commodity k on link a (from node i to node j). For each commodity $k \in \{1...K\}$, $x^k \in R_+^m$ is the link flow vector containing elements x_a^k ($\forall a \in A$), and $q^k \in R^n$ is the vector of travel demands destinated at node k. Denote f_a as the total flow on link a, i.e., $f_a = \sum_{k=1}^{K} x_a^k, \forall a \in A$, and f as the vector of elements f_a ($\forall a \in A$). Let ξ_a represent the random hazard event on link a, and ξ be the vector of elements ξ_a ($\forall a \in A$). We may consider two possible outcomes of ξ_a (i.e., ξ_a=1 states that link a, if not protected, will be damaged in a disaster; and 0 otherwise). Apparently, the post-disaster condition of link a depends on both the protection decision u_a and the actual realization of ξ_a. We introduce function $h_a(u_a, \xi_a)$ to represent the post-disaster capacity of link a. At this point, let us only consider a deterministic value of function h_a for given (u_a, ξ_a).

A scenario sub-problem:

If the future disaster scenario is known, that is if the exact value of ξ is known, the network protection problem can be formulated as

$$\min_u Q(u, f), \tag{1}$$

s.t. $$u \in F, \tag{2}$$

$$f_a = \sum_{k=1}^{K} x_a^k \le h_a(u_a, \xi_a), \forall a \in A\,, \tag{3}$$

$$x_{ij}^k (t_{ij} + \lambda_j^k - \lambda_i^k) = 0\text{, and } x_{ij}^k \ge 0, (t_{ij} + \lambda_j^k - \lambda_i^k) \ge 0\,,\ \forall (i,j) \in A, k = 1...K \tag{4}$$

$$Wx^k = q^k - d^k,\ \ \forall k = 1...K\,, \tag{5}$$

Expression (1) states that the objective is to find the optimal protection strategy that will minimize the total losses (i.e. repair cost, delay, and other penalty costs), quantified by function Q, caused by the given disaster scenario, where u is an $\overline{m}$ by 1 vector of elements u_a ($a \in \overline{A}$), and f is an m by 1 vector of elements f_a ($a \in A$). A suitable choice of function Q can be

$$Q(u, f) = <\rho, (c - h(u, \xi))> + \gamma < f, t(f)> + M \sum_{k=1}^{K} \left\| d^k \right\|, \tag{1a}$$

which is the sum of total repair cost, the monetary value of the total travel delay on the network, and the penalty cost for unsatisfied travel demand. Explanation of the penalty term will be provided shortly. Parameter ρ represents the repair costs, c is the original link capacity, the parameter γ converts travel time to monetary value, and the operator $<a, b>$ represents the inner product of vectors a and b. The link time function $t(f)$ is assumed to be in the standard form of BPR function $t_a^0[1 + \alpha(\frac{x_a}{c'_a})^\beta]$ to capture congestion effects, where t_a^0 and x_a are free flow travel time and flow for link a respectively, α and β are parameters, and c'_a is the "practical capacity" of link a. Constraint (2) specifies that the protection strategy must belong to the feasible set F, which may depend on budgetary and technological restrictions. Constraint (3) restricts the total flow on each link not to exceed the link capacity. Constraints (4-5) define the user equilibrium traffic flow. The quantity t_{ij} is the travel time on link *ij* which is a function of the flow on that link. The quantity λ_i^k is the minimum time from node *i* to destination *k*. The complementary condition (4) indicates that if a positive amount of flow travels on link *ij* toward destination *k* (i.e., $x_{ij}^k > 0$), then link *ij* must be on the shortest path from *i* to *k* (i.e., $t_{ij} + \lambda_j^k = \lambda_i^k$). Constraint (5) defines the conservation of network flow, in which *W* represents the node-link adjacency matrix. Ideally, we wish to assign all travel demand to the network, i.e., we want constraints $Wx^k = q^k, \ \forall k = 1...K$ to be satisfied. However, the network may lose some capacity or even be disconnected in a post-disaster situation, thus may not be able to accommodate all travel demand. In order to guarantee feasible network flow solution, we introduce vector d^k to capture the amount of unsatisfied travel demand of each commodity *k*, and penalize these unsatisfied demands in function *Q*. Note that d^k is dependent on protection decision *u* and the disaster scenario ξ. The parameter *M* in function *Q* should be set as a very large number, so that only the trips that cannot be accommodated by the network are captured by d^k and penalized. Readers may note that for a given disaster scenario, constraints (3-5) are equivalent to the lower-level user equilibrium traffic assignment formulation in the well-known bi-level program of network design problems (Yang and Bell, 1998).

The numerical difficulties of solving the problem defined by (1-5) come from the complementary constraint (4), which makes the program nonconvex and Mangasarian Fromovitz Constraint Qualification not satisfied (Luo et al, 1996). Solving a mathematical program with complementary constraints (MPCC) directly by off-shelf solvers is difficult. In this work, we adopt a relaxation approach to convert a MPCC to a series of nonlinear programs. Details of this approach are given in Section 3.

A stochastic programming formulation:

Now, let us consider the real-world situation where decisions must be made without an exact foresight of the future. In this case, it is common to consider a range of possible scenarios and their associated probabilities or importance weights. Let S be the set of possible scenarios for ξ, and s ($s \in S$) denote an individual scenario.

Solving the scenario sub-problems defined in (1-5) for all s ($s \in S$) will give us different s-dependent protection solutions, denoted as u^s for each s. Note that u^s is a vector containing elements of u_a^s. However, these solutions can not be directly implemented, because at the time when the protection solution is implemented, one does not know yet which scenario is going to happen. In order to consolidate the s-dependent solutions to an *implementable* solution, we must impose the following condition:

$$u^s = u^{s'}, \ \forall s \in S, s' \in S, s \neq s', \tag{6}$$

or equivalently

$$u^s - z = 0, \ \forall s \in S \tag{7}$$

where z is a vector of free variables. This condition is called a *nonanticipativity* constraint defined by Rockafellar and Wets (1991), which states that a reasonable policy should not require different actions relative to different scenarios if the scenarios are not distinguishable at the time when the actions are taken.

If we agree upon setting the performance measure in a "weighted average" sense, the overall stochastic problem can be formulated as:

$$\min_{u^s} \sum_{s\in S} p^s Q^s(u^s, f^s) \tag{8}$$

s.t. $$u^s \in F^s,\ \forall s \in S, \tag{9}$$

$$f_a^s = \sum_{k=1}^{K} x_a^{k,s} \le h_a(u_a^s, \xi_a^s),\ \forall a \in A, s \in S, \tag{10}$$

$$x_{ij}^{k,s}(t_{ij}^s + \lambda_j^{k,s} - \lambda_i^{k,s}) = 0, \text{ and } x_{ij}^{k,s} \ge 0, (t_{ij} + \lambda_j^{k,s} - \lambda_i^{k,s}) \ge 0,$$

$$\forall (i,j) \in A, k = 1...K,\ \forall s \in S, \tag{11}$$

$$Wx^{k,s} = q^k - d^{k,s},\ \forall k = 1...K,\ \ \forall s \in S, \tag{12}$$

$$u^s - z = 0,\ \forall s \in S, \tag{13}$$

where a quantity with a superscript s indicates that the quantity is scenario dependent. Constraints (9-12) guarantee that the solution to the first- and second-stage variables is feasible in all scenarios. Condition (13) guarantees that the first-stage protection decision is also implementable. A suitable choice of the s-dependent function Q can be

$$Q^s(u, f^s) = <\rho, (c - h(u, \xi^s))> + \gamma < f^s, t(f^s)> + M\sum_{k=1}^{K} \left\| d^{k,s} \right\|, \tag{8a}$$

where all the parameters keep the same definitions as in expression (1a).

However, a stochastic programming problem is significantly larger in size than the corresponding deterministic version of the problem. For example, for an L-scenario problem, the number of constraints in the stochastic problem would be L folds of that in a deterministic problem, plus the additional nonanticipativity constraints that can not be separated across scenarios. An MPCC problem is already difficult. Solving a stochastic version of MPCC problem imposes even more challenges. In the following sections, we describe our approach to solve stochastic network problems with equilibrium constraints.

3. Numerical Methods

The key computational difficulties faced in this work stem from the large problem size and the complementary conditions. Various decomposition methods have been proposed to handle large scale stochastic programming problems (Ruszczynski, 1997), of which cutting plane based procedures, such as the L-shape decomposition method (Van Slyke and Wets, 1969) and its variants, are the most popular and well studied. In our previous study, we have successfully implemented a modified L-shape method to solve large scale stochastic network design problems without equilibrium constraints (Liu and Fan, 2007). However, the variants of L-shape method strongly rely on convexity assumption and therefore are not suitable for the problem in question.

Through numerical experiments, we found that the progressive hedging (PH) method of Rockafellar and Wets (1991) is suitable for solving our problem. The PH method decomposes a stochastic problem across scenarios and partitions the problem into manageable sub-problems. Let us denote G_s as the feasible solution set defined by constraints (9-12) in each scenario s. Define

$$L_r(U, X, z, W) = \sum_{s \in S} p_s (Q_s(u^s, x^s) + (w^s)' \cdot (u^s - z) + \frac{1}{2} r \left\| u^s - z \right\|^2) \tag{14}$$

as the augmented Lagrangian, where W is the vector of dual variables for the constraints in (13) and $r > 0$ is a penalty parameter associated with violation of the nonanticipativity constraints. Therefore, the augmented Lagrangian integrates the nonanticipativity constraints with the original objective function. The stochastic network protection problem becomes

$$\text{minimize } L_r(U, X, z, W) \text{ over all } (u, x) \in G_s. \tag{15}$$

Due to the nonseparable penalty term $\frac{1}{2}r\left\|u^{s}-z\right\|^{2}$ in (14), the problem can not be decomposed directly. The PH method achieves decomposition by alternatingly fixing the scenario solutions (u,x) and the implementable solution z in (15). The detailed procedures are described below.

<u>The progressive hedging algorithm [PH]</u>

Step 1.

Set the iteration index ν to 0. Solve for each scenario sub-problem defined in (1-5) and obtain $(u^{s},x^{s})\ \forall s\in S$. Initialize $z^{\nu}=\sum_{s\in S}p_{s}u^{s}$. If $(u^{s})^{\nu}=z^{\nu},\forall s\in S$, then the optimal solution is found, otherwise continue with step 2.

Step 2.

Repeat step 2 until the termination criterion

$$\varepsilon=[\| z^{\upsilon}-z^{\upsilon-1}\|^{2}+\sum_{s\in S}p_{s}\|(u^{s})^{\upsilon}-z^{\upsilon}\|^{2}]^{1/2}\approx 0 \tag{16}$$

is reached.

Solve for each scenario

$$((u^{s})^{\nu+1},(x^{s})^{\nu+1})\in\arg\min_{(u^{s},x^{s})\in G_{s}}\left\{Q_{s}(u^{s},x^{s})+((w^{s})^{\nu})'\cdot u^{s}+\frac{r^{\nu}}{2}\left\|u^{s}-z^{\nu}\right\|^{2}\right\},\ s\in S\ .$$

Obtain a new implementable solution

$$z^{\nu+1}=\sum_{s\in S}p_{s}(u^{s})^{\nu+1}\ .$$

Update the dual variable estimates

$$(w^{s})^{\nu+1}=(w^{s})^{\nu}+r^{\nu}((u^{s})^{\nu+1}-z^{\nu+1}),s\in S\ .$$

Increase the iteration index ν by 1. One may also adjust the penalty parameter r^{ν} as the iteration proceeds. We will have more discussion on the choices of parameter r in the next section.

Solving MPCC problem via relaxation

The computational intensive part of the PH-based solution procedure is in solving many MPCC scenario subproblems. Thus it is crucial to select an effective algorithm for solving MPCC problems. In this work, we adopt an approach of reformulating MPCC into mixed integer nonlinear program (MINLP) through relaxation of complementary constraints. Two relaxation schemes are considered: regularization and penalization. The regularization scheme relaxes the right hand side constant of the complementary constraints from zero to a positive number, in which case constraint (4) becomes

$$x_{ij}^{k}(t_{ij}+\lambda_{j}^{k}-\lambda_{i}^{k})\le\mu\,. \tag{17}$$

In the penalization scheme, the complementary constraints are added to the objective as penalty terms:

$$\min_{u} Q(u,f)+\frac{1}{\mu}\sum_{(i,j),k} x_{ij}^{k}(t_{ij}+\lambda_{j}^{k}-\lambda_{i}^{k})\,. \tag{18}$$

A series of MINLPs are generated when the relaxation parameter μ, initialized as a relatively large positive number, is gradually reduced to close to zero. These resultant MINLPs are solved directly by commercial solvers.

Ralph and Wright (2004) showed under certain conditions, the above relaxation schemes can solve MPCC to attain a local optimum. Ban et al (2006) also reported successful experience of applying relaxation approaches to solving deterministic network design problems. One can control solution accuracy by adjusting the “track” of parameter μ, i.e., the initial and final values of μ and the reducing factor. For example, a possible choice of the track of μ can be $(10, 1, \ldots, 10^{-6})$. A coarse track of μ results in speed-up but leads to a less accurate solution. Since the PH method does not require scenario subproblems to be solved accurately, we shall keep a balance between solution accuracy and speed so that a good approximate solution to the subproblem can be generated rather quickly.

Thus far, we have shown that in order to solve a large scale stochastic network optimization problem with equilibrium constraints, we can rely on the PH algorithm to decompose the large scale problem to subproblems of manageable sizes, and use relaxation approaches to convert MPCC subproblems to a series of mixed integer nonlinear programs which can be directly solved by commercial solvers. In summary, the major advantages of this PH-based solution procedure applied to our work are:

(1) Each sub-problem is a typical network design problem, for which many specialized solution algorithms are available.

(2) The core of PH algorithm is an augmented Lagrangian, which is not limited to problems of convexity.

(3) The PH algorithm only requires that the sub-problems be solved approximately. In our work, solving each MPCC subproblem for the exact solution is highly time consuming, but a good approximation can be found much more easily through relaxing the complementarity constraints.

(4) The PH algorithm decomposes a problem to multiple independent subproblems of similar size and complexity, thus lands itself to easy parallelization.

4. Numerical Examples

Rigorous convergence proof of the PH algorithm in convex and continuous problems is given by Rockafellar and Wets (1991). For nonconvex and continuous problems, the best theoretic knowledge about the convergence of the PH algorithm is that if all scenario subproblems are solved to local optimal solutions in each iteration, and if the sequences of the decision and the dual variables do converge, they converge to the optimal solutions. These convergence theorems are valid unconditional to any particular choice of the penalty parameter *r*. However, it was also pointed out by the authors (Rockafellar and Wets, 1991) that parameter *r* plays an important role in convergence in practice. Mulvey and Vladimirou (1991, 1992) and Lokketangen and Woodruff (1996) reported some important factors that may influence the setting of penalty parameter *r* in convex problems, which provide valuable numerical results for our research. Since the problem here is nonconvex and discrete, for which no previous numerical implementation of the PH algorithm is available, we designed the following numerical experiments to explore the applicability of the PH algorithm to an extended range of problem types.

Three well known networks in transportation network literature are used in our numerical experiments, including the Braess network (Hagstrom and Abram , 2001), the Sioux Fall city network (Leblanc, 1975), and the network used by Harker and Friesz (1984). The numerical results obtained from the three networks are consistent. Due to limited space, we only provide detailed data description and numerical results for the Sioux Fall network.

The Sioux Fall city road network has 24 nodes and 76 links, as shown in Figure 1. It is assumed that six road segments (twelve links), labeled as A to F in Figure 1, are subjected to potential hazards. These twelve links are the candidate links for receiving protection action. However, due to insufficient resources, only four links can receive immediate protection. The question is: which set of four links should be protected so that the total expected loss (quantified by the sum of repair costs for the damaged links and the total network delay) caused by future hazards is minimized? In this case study, we require that the protection decision be binary (u_a= 1 if link a is to be protected, u_a= 0 otherwise), and that two links of opposite directions on the same road segment should receive the same protection action. Including more options for u_a does not change the nature of the problem.

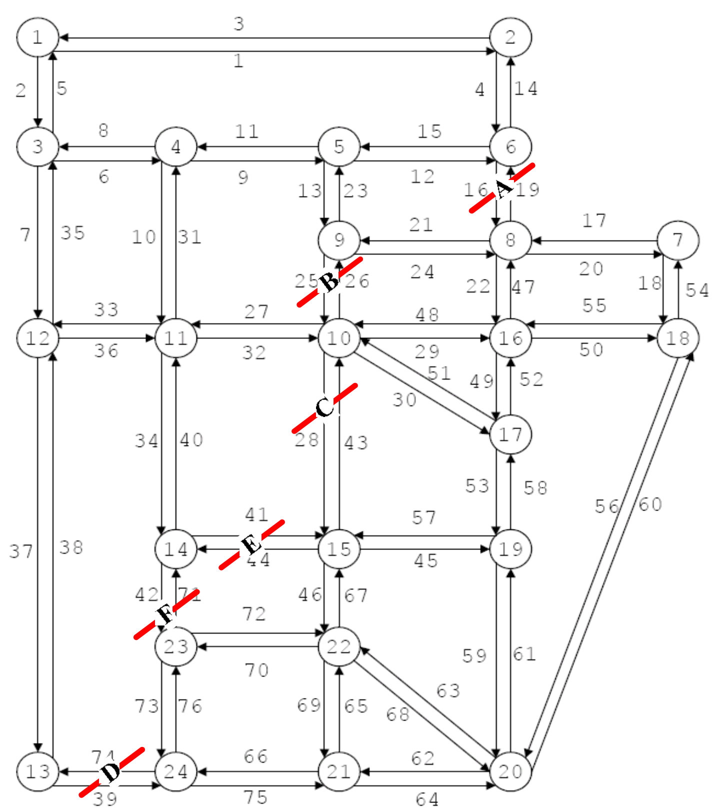

FIGURE 1 Sioux Fall city network

There are totally $2^6 = 64$ possible future scenarios for the random vector $\boldsymbol{\xi}$. We use the independent probabilities given in Table 1 to generate the probabilities associated with those 64 scenarios. Note that assumption of independent probabilities is only for the convenience of generating test data. Probabilities of damage scenarios generated with consideration of correlation between individual link damage states can be used in the same manner as an input to the model.

TABLE 1 Independent probability of link damage for generating the set of damage scenarios

Segment	A	B	C	D	E	F
Probability of damage	0.1	0.1	0.4	0.5	0.8	0.7

Specific forms are assigned to functions $h(u,\xi)$ and $Q^s(u,f)$. Let

$h(u,\xi) = (e - \Xi(\xi,u))c$,

where c represents pre-protection link capacities, e is a vector of all elements being 1, and

$$\Xi(\xi,u)_a = \begin{cases} \xi_a(\xi_a - u_a), & \forall a \in \overline{A} \\ 0, & \forall a \in A \setminus \overline{A} \end{cases}. \tag{19}$$

The choice of $\Xi(\xi,u)_a$ is based on the assumption that a link, once protected, will remain intact under any disaster scenario. A more realistic assumption may be that once a link is protected, its chance of being damaged is reduced but not zero. However, this more realistic assumption will lead to a stochastic programming problem with decision-dependent uncertain scenarios. At this point, we do not know of a good method for solving such a problem other than probably relying on Monte Carlo sampling (Viswanath et al, 2007).

Each s-dependent function Q^s is defined in Equation (8a), where the parameters are set as ρ=1.5, γ=1, $\alpha = 0.15$, $c' = 0.9c$, and $\beta = 1$, $M = 10^6$. With this dataset, travel demand can be accommodated by the network in all post-disaster scenarios. Therefore, the value of the penalty term in function Q^s is zero for all *s*. The optimal solution obtained from the PH algorithm is to protect links $13 \leftrightarrow 24$ and $14 \leftrightarrow 15$, which leads to an optimal objective value of 45.55. Through enumeration, we found that the worst protection strategy is to protect links $6 \leftrightarrow 8$ and $9 \leftrightarrow 10$, which leads to an objective value of 54.98. The gain of following an optimal protections strategy can be as high as 20% in this case study. In case of unsatisfied travel demand, i.e., in case $d^{k,s}$ is positive, the value of the objective function E{Q} should be used with caution, because *M* is set high for computational purpose, and does not reflect the actual value of trips forgone. Some discussion on quantifying the value of trips forgone due to lost network capacity can be found in (Cho, Fan, and Moore, 2003).

Effects of penalty parameter r on convergence

The PH algorithm was implemented with different values of *r* for solving the 64-scenario stochastic programming problem. In Figure 2, the sequences of convergence resulted from different *r* values are plotted. The *x* axis corresponds to the number of iterations, and the *y* axis corresponds to the value of the error term ε defined in Equation (16). When *r* is set between 0.2-0.7, the algorithm was able to converge to the optimal solution, even though the convergence speeds varied. When *r* is set to be 1, the sequence started to oscillate after a few iterations. This oscillation continues if the value of *r* remains the same. Usually reducing the *r* value can break the oscillations. As shown in Figure 2, reducing *r* from 1 to 0.5 terminated the oscillation and led to convergence.

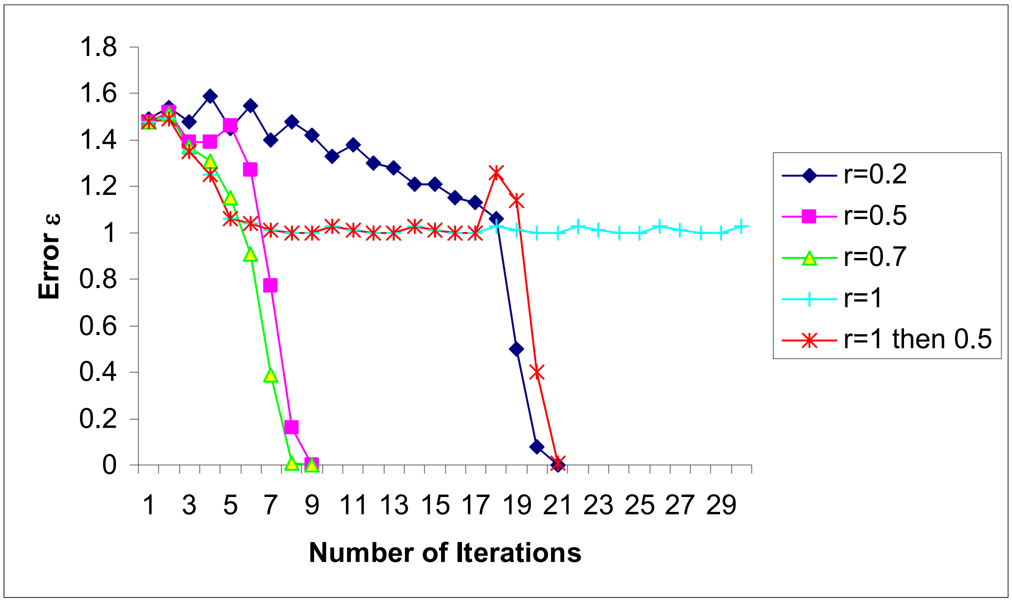

FIGURE 2 Sequences of convergence resulted from different values of the penalty Parameter *r*

One can discern from Figure 2 that among all feasible values of *r*, some perform better than others. How should one select an appropriate *r* that could lead to fast convergence to an optimal solution? Previous research has demonstrated in convex cases that the setting of *r* is strongly influenced by the sensitivity of the objective function with regard to changes in the first-stage decision variables. In order to investigate this issue in the context of discrete and nonconvex problems, we implemented the PH algorithm using different *r* values for problems with different repair costs (i.e., different ρ values). To speed up the computation, only the ten most likely scenarios are included. The convergence performances of the PH algorithm in different settings are summarized in Table 2. A cell with symbol * indicates convergence to an optimal solution of the model. For example, in problems where ρ =1.5, the PH algorithm with r = 0.5 solved the problem optimally within 10 iterations, but when r = 50 the algorithm only converged to a suboptimal solution. The scale of ρ has a direct impact on the setting of r. As ρ increased from 1.5 to 1500, the effective range of r value also changed from around the neighborhood of 0.5 to the neighborhood of 500. In addition to the repair costs, other parameters that affect the sensitivity of the objective value with respect to a change in the first stage variables (u_a), such as β, also matter to the setting of r. For example, as β increased from 1 to 4, the good range of r value changed from the neighborhood of 1 to the neighborhood of 15.

TABLE 2 Penalty parameter *r* scales to the objective value: *r* vs. ρ ($\beta = 1$)

	ρ =1.5	ρ =1.5x10	ρ =1.5x100	ρ =1.5x1000
r = 0.15	*23 iterations	>40 iterations	>40 iterations	>40 iterations
r = 0.5	***10 iterations**	*34 iterations	>40 iterations	>40 iterations
r = 5	oscillation after 2 iterations	***6 iterations**	*31 iterations	>40 iterations
r = 50	Suboptimal, converged in 4 iterations	*7 iterations	***5 iterations**	*31 iterations
r = 500	Suboptimal, converged in 4 iterations	Suboptimal, converged in 4 iterations	Suboptimal, converged in 4 iterations	***5 iterations**

Based on our numerical experiments, we draw the following general rules for choosing an appropriate range for parameter r. Most of these rules are consistent with those drawn for convex problems in Mulvey J.M. and H. Vladimirou (1991) and Lokketangen and Woodruff (1996):

1. A small r usually results in gradual convergence to the optimal solution; while a big r generally produces faster initial convergence, but may arrive at a suboptimal solution. Thus an intermediate r is preferred for the best overall performance of the PH algorithm.
2. When r is not carefully chosen, oscillated results may appear. This phenomenon is unique in discrete problems.
3. The choice of r depends on the sensitivity of the objective to changes in the first stage variables.

In addition to the above three remarks, Mulvey and Vladimirou (1991) also reported that in convex problems the scale of r is dependent on the structure of the problem. If the nonanticipativity constraints are highly restrictive, a bigger value of r should be used.

Effects of number of scenarios

Numerical tests with different number of scenarios (10, 20, and 64 scenarios) are carried out to study the effects of number of scenarios on the performance of the PH algorithm. It was found that the number of iterations required by the PH algorithm to converge to an optimal solution is not conditional to the number of scenarios. For example, in all testing problems, an optimal solution was reached in about eight or nine iterations (r was set to be 0.7 in all cases). The approximate computing time is about 10 min for a 10-scenario problem, about half hour for a 20-scenario problem, and about 2 hours for a 64-scenario problem[1]. The increasing computing time associated with larger number of scenarios is due to the increasing number of MPEC sub-problems that need to be solved in each iteration of the PH procedure. In fact, more than 99% of the computing time was devoted to repeatedly solving scenario subproblems. Significant amount of computing time can

[1] Intel Xeon 3060 CPU @2.40 and 2.39GHZ, 2GB RAM.

be saved via parallel computation of the subproblems. Because the subproblems in the PH procedure are of similar size and complexity, and they are independent of each other, it is quite straight forward to implement parallel processors in the PH procedure. This makes the PH algorithm particularly favorable for problems with large number of scenarios. More discussion on parallel computation implemented for the PH procedure can be found in (Mulvey and Vladimirou, 1992).

Effects of initial solution on the convergence

In the PH procedure description provided in Section 3, we stated that an initial solution to the first-stage decision variables can be found by simply aggregating the scenario-dependent solutions. This is an easy choice, but may not be the most efficient. One may first solve the problem without the equilibrium constraints (i.e., to solve a stochastic problem with the system optimal (SO) traffic condition), and then use the SO solution as the initial solution to the corresponding problem with equilibrium conditions. The cost of this choice is the additional computing efforts spent on solving a stochastic programming problem that is large in size but convex. The benefit is the reduced number of iterations in the PH procedure. In most problems where the number of scenario subproblems is not trivially small, such a tradeoff would be worthwhile. This is because the computational complexity introduced by equilibrium constraints is tremendous. In our case, the time required for solving a MPCC subproblem is about 10 times of that for the corresponding SO subproblem. Considering that each iteration of the PH procedure involves many MPCC problems, the time saved from reducing one iteration can be significant. In addition, since the proper settings of r do not differ by much between problems with or without equilibrium constraints, starting with an SO problem can help tune r value in a much less expensive manner.

<u>Notes on integrating MPEC solvers with the PH algorithm</u>

As mentioned earlier, the effectiveness of the solution method for MPCC subproblems is critical to the performance of the entire solution procedure. We adopt the NLPEC (nonlinear program with equilibrium constraint) solver by Ferris *et al* (2002), which can automatically reformulate an MPCC to an MINLP, and call GAMS solvers to solve the MINLP. Users can specify the choices of the MINLP solver, reformulation type, and relaxation setting through the NLPEC option files.

For solving scenario subproblems, we observe that the regularization scheme (also called multiplication in NLPEC) provides higher solution accuracy and the penalty scheme has faster convergence. One can adjust the settings of the relaxation parameter μ to control the balance between solution accuracy and speed. For example, one setting for the sequence of μ is $(10,1,0.1,\ldots,10^{-6})$ with a reducing factor 0.1, whereas a more approximate setting can be (10,1,0.1). Obviously the latter setting achieves a faster solution speed but a less accurate solution. In the numerical experiments we have conducted, the PH algorithm performed well in terms of both solution accuracy and speed, when the regularization reformulation type is adopted with the track of μ set as (10, 1, 0.1) and the resultant MINLPs are solved using SBB solver[2].

5. Discussions

Stochastic network optimization problems with equilibrium constraints are important in many areas of science and engineering particularly in transportation and logistics. However, due to their computational complexity, numerical implementation for such

[2] a GAMS solver for mixed integer nonlinear program problems, http://www.gams.com/solvers/solvers.htm

problems has been lacking. In this paper, we have demonstrated that such problems can be successfully solved by integrating progressive hedging based decomposition method and an efficient solution method for MPCC subproblems. Previous research has demonstrated the applicability of the PH algorithm to problems of convex and in most cases continuous nature. This work extends the PH method to a broader range of applications including discrete and nonconvex problems as well.

Several issues need further investigation. From a computational viewpoint, the subproblems to be solved in consecutive iterations share similar structures and properties. One may make use of solutions to the problems in previous iterations so that the remaining subproblems can be solved with a warm-start. From a modeling viewpoint, if the planning horizon is relatively short, say within hours or days, then the dynamics of traffic should be explicitly modeled instead of using the averaged static flow. Also, it would be more realistic to use a complete probability distribution to describe the post-disaster link condition, instead of the binary assumption we made to represent $h(u,\xi)$. Such an extension leads to a stochastic programming problem with decision-dependent random events. Mathematical analysis for this class of problems is very sparse, and is only limited to convex problems of special structures (Jonsbraten et al, 1997). Finding a suitable solution method for such problems with equilibrium constraints is an ongoing endeavor.

In addition to transportation systems, other utility networks, financial networks, and social networks may face a similar need of modifying network topology under future uncertainties. We believe the computational methodology presented herein can be valuable to a broader audience whose interests lie in stochastic network problems in general.

ACKNOWLEDGMENTS

The authors would like to thank National Science Foundation and California Department of Transportation for supporting this research through the Pacific Earthquake Engineering (PEER) research center. Helpful discussions with Prof. Roger Wets and Prof. David Woodruff at University of California, Davis, and Dr. Jeff Ban at University of California, Berkeley, are gratefully acknowledged.

References

1. Magnanti, T.L., and R.T. Wong. Network design and transportation planning: models and algorithms. *Transportation Science* Vol. 18, 1984, pp.1-55.
2. Yang, H., and M.G.H Bell. Models and algorithms for road network design: a review and some new developments. *Transport Reviews* Vol. 18, 1998, pp. 257-278.
3. Patil, G. and S. Ukkusuri, Stochastic system optimal network design problem. *Proceedings of Transportation Research Board Annual Meeting*, 2007.
4. Woodruff, D., *Network Interdiction and Stochastic Programming*, Kluwer Academic Publishers. 2003.
5. Dafermos, S. and A. Nagurney, On some traffic equilibrium theory paradoxes, *Transportation Research B*, Vol. 18, 1984, pp. 101-110.
6. Dantzig, G.B. Linear programming under uncertainty. *Management Science*. Vol. 1, 1955, pp. 197-206.

7. Wets, R.J.B. Programming under uncertainty: The equivalent convex program. *SIAM Journal of Applied Mathematics,* Vol.14, No. 1, 1966, pp. 89-105.

8. Van Slyke, R.M., and R.J.B. Wets. L-shaped programs with applications to optimal control and stochastic linear programming. *SIAM Journal of Applied Mathematics* Vol. 17, 1969, pp. 638-663.

9. Wets, R.J.B. Stochastic programs with fixed recourse: the equivalent deterministic program, *SIAM Review*, Vol. 16, 1974, pp. 309-339.

10. Mulvey, J.M. and Vladimirou H., Applying the progressive hedging algorithm to stochastic generalized networks, *Annals of Operations Research*, Vol. 31, 1991, pp. 399-424.

11. Bell, M. and Iida Y., *Transportation Network Analysis*, John Willy & Sons, 1997.

12. Marcotte, P. and D.L. Zhu, Exact and inexact penalty methods for the generalized bilevel problem, *Mathematical Programming*, Vol. 74, 1996, pp.141-157.

13. Patriksson M. and R.T. Rockafellar, A mathematical model and descent algorithm for bilevel traffic management, *Transportation Science*, Vol. 36, 2002, pp.271-291.

14. Ban, J., H. Liu, Ferris, M.C., and B. Ran, A general MPCC model and its solution algorithm for continuous network design problem, *Mathematical and Computer Modeling*, Vol. 43, 2006, pp.493-505.

15. Luo, Z.Q., J.S. Pang, D. Ralph, *Mathematical Programs with Equilibrium Constraints*, Cambridge University Press, Cambridge, UK,1996

16. Rockafellar, R.T. and R. J.-B. Wets, Scenarios and policy aggregation in optimization under uncertainty, *Mathematics of Operations Research*, Vol. 16, 1991, pp. 119-147.

17. Ruszczynski, A., Decomposition methods in stochastic programming, *Mathematical Programming*, Vol. 79, 1997, pp. 333-353.

18. Liu, C.Z. and Y.Y. Fan, A two-stage stochastic programming model for transportation network protection, *Proceedings of Transportation Research Board Annual Meeting*, 2007.

19. Ralph, D. and S.J. Wright, Some properties of regularization and penalization schemes for MPECs, *Optimization Methods and Software*, Vol. 19, 2004, pp. 527-556.

20. Lokketangen A. and D. Woodruff, Progressive Hedging and Tabu Search Applied to Mixed Integer (0,1) Multi-stage Stochastic Programming, *Journal of Heuristics*, Vol. 2, 1996, pp. 111-128.

21. Mulvey J.M. and H. Vladimirou, Stochastic network programming for financial planning problems, *Management Science*, Vol. 38, No.11, 1992, pp. 1642-1664.

22. Hagstrom J.N. and R.A. Abram, Characterizing Braess's Paradox for Traffic Networks, *2001 IEEE Intelligent Transportation Systems Conference Proceedings*, August 2001, Oakland, California.

23. Leblanc L.J., E.K. Morlok, W.P. Pierskalla, An efficient approach to solving the road network equilibrium traffic assignment problem, *Transportation Research*, Vol. 9, 1975, pp. 309-318. Data available at http://www.bgu.ac.il/~bargera/tntp/

24. Harker, T.L. and T.L. Friesz, Bounding the solution of the continuous equilibrium network design problem, *Proceedings of the Ninth International Symposium on Transportation and Traffic Theory*, 1984, Netherlands.

25. Viswanath, K., S. Peeta and F.S. Salman, Investing in the links of stochastic network to minimize expected shortest path length, working paper.

26. Cho, S., Y. Fan, and J. Moore, Modeling Transportation Network Flows as a Simultaneous Function of Travel Demand, Network Damage, and Network Level of Service, the Proceedings of the ASCE Technical Council on Lifeline Earthquake Engineering (TCLEE) Sixth U.S. Conference and Workshop on Lifeline Earthquake Engineering, August 10-13, 2003, Long Beach, CA, pp. 868-877.

27. Ferris, M.C. et al, Mathematical Programs with Equilibrium Constraints: Automatic Reformulation and Solution via Constrained Optimization (2002), Oxford University Computing Laboratory, http://citeseer.ist.psu.edu/633625.html

28. Jonsbraten, T. W., Roger J-B Wets, and David L. Woodruff, A Class of Stochastic Programs with Decision Dependent Random Elements, Technical Report, University of California Davis, August 1997.