New Solution Approaches for the Maximum-Reliability Stochastic Network Interdiction Problem

Eli Towle* and James Luedtke†

Department of Industrial and Systems Engineering, University of Wisconsin – Madison

Abstract

We investigate methods to solve the maximum-reliability stochastic network interdiction problem (SNIP). In this problem, a defender interdicts arcs on a directed graph to minimize an attacker's probability of undetected traversal through the network. The attacker's origin and destination are unknown to the defender and assumed to be random. SNIP can be formulated as a stochastic mixed-integer program via a deterministic equivalent formulation (DEF). As the size of this DEF makes it impractical for solving large instances, current approaches to solving SNIP rely on modifications of Benders decomposition. We present two new approaches to solve SNIP. First, we introduce a new DEF that is significantly more compact than the standard DEF. Second, we propose a new path-based formulation of SNIP. The number of constraints required to define this formulation grows exponentially with the size of the network, but the model can be solved via delayed constraint generation. We present valid inequalities for this path-based formulation which are dependent on the structure of the interdicted arc probabilities. We propose a branch-and-cut (BC) algorithm to solve this new SNIP formulation. Computational results demonstrate that directly solving the more compact SNIP formulation and this BC algorithm both provide an improvement over a state-of-the-art implementation of Benders decomposition for this problem.

Keywords. Network interdiction; stochastic programming; integer programming; valid inequalities

Acknowledgments This work was supported by National Science Foundation grants CMMI-1130266 and SES-1422768.

1 Introduction

Network interdiction problems feature a defender and an attacker with opposing goals. The defender first modifies a network using a finite budget in an attempt to diminish the attacker's ability to perform a task. These modifications could include increasing arc traversal costs, reducing arc capacities, or removing arcs altogether. The attacker then optimizes his or her objective with respect to the newly modified network. Network interdiction problems have been studied in the context of nuclear weapons interdiction [17, 20], disruption of nuclear weapons projects [6], interdiction of drug smuggling routes [24], and general critical infrastructure defense [5].

In the maximum-reliability stochastic network interdiction problem (SNIP), formulated by Pan et al. [20], an attacker seeks to maximize the probability of avoiding detection while traveling through a directed network from an origin to a destination. There is a chance the attacker will be detected when traversing each arc. A defender may expend resources to place sensors on certain arcs, thereby increasing the probability of detecting an attacker on those arcs. The origin and destination of the attacker are unknown to the defender, and are assumed to be random. The defender's goal is to minimize the expected value of the attacker's maximum-reliability path through the network over all origin–destination scenarios. This can be interpreted as minimizing the overall probability of the attacker successfully attacking a critical infrastructure target or smuggling contraband to a target location undetected.

*etowle@wisc.edu

†jim.luedtke@wisc.edu

Morton et al. [17] present a deterministic equivalent formulation (DEF) of SNIP derived from the dual of the attacker's optimization problem. Pan and Morton [21] improve the DEF by incorporating the values of uninterdicted maximum-reliability paths into the model constraints. They also derive SNIP-specific "step inequalities" to strengthen the linear programming (LP) relaxation of the Benders master problem before initiating the Benders algorithm. These step inequalities are equivalent to the mixing inequalities of Günlük and Pochet [11]. Bodur et al. [4] investigate the strength of integrality-based cuts in conjunction with Benders cuts for stochastic integer programs with continuous recourse, and use test instances of the SNIP problem to demonstrate the value of integrality-based cuts within Benders decomposition.

We propose two new approaches to solve SNIP. First, we present a more compact DEF. This formulation combines constraints of the DEF for scenarios ending at the same destination, significantly reducing the number of constraints and variables in the DEF. Second, we present a new formulation of SNIP that includes constraints for every origin–destination path. Although the formulation size grows exponentially with the number of arcs in the network, the model can be solved with delayed constrained generation. A path's reliability function is a supermodular set function representable with a strictly convex and decreasing function. We propose a branch-and-cut algorithm that exploits this structure for each path through the network to derive valid inequalities. We consider three cases for the probabilities of traversing interdicted arcs. Two of these cases use inequalities derived by Ahmed and Atamtürk [1] for the problem of maximizing a submodular utility function having similar structure.

The maximum-reliability network interdiction problem is closely related to the shortest-path variant, in which the network defender attempts to maximize the length of the attacker's shortest path from origin to destination. When the probabilities of traversing interdicted arcs in a maximum-reliability problem are all strictly positive, the problem is equivalent, via a logarithmic transformation, to a shortest-path network interdiction problem [2, 16]. This transformation, however, is not valid when there exists an arc that reduces the attacker's probability of successful traversal to zero when it is interdicted by the defender. A deterministic version of the shortest-path network interdiction problem was initially explored by Fulkerson and Harding [9], and later by Golden [10].

An alternative interdiction problem mostly unrelated to our work is the maximum-flow network interdiction problem, introduced by Wollmer [23]. In this problem, a defender changes arc capacities in order to minimize the attacker's maximum flow. Deterministic and stochastic versions of this problem have been studied by many authors, e.g., [7, 12, 13, 24].

In Section 2, we review relevant existing approaches to solving SNIP. We propose a compact DEF of SNIP in Section 3. In Section 4, we propose a path-based formulation for SNIP and describe valid inequalities for the relevant mixed-integer set. We describe our computational experiments and results in Section 5.

2 Problem statement and existing results

Let N and A denote the set of nodes and the set of arcs of a directed network. Let $D \subseteq A$ denote the set of arcs available for interdiction. In the first stage of SNIP, the defender may choose to install sensors on a subset of the interdictable arcs. The cost to install a sensor on arc $a \in D$ is $c_a > 0$. The defender is constrained by installation budget $b > 0$. The probability of the attacker's undetected traversal through arc $a \in A$ without a sensor installed is $r_a \in (0, 1]$. When a sensor is installed on arc $a \in D$, this probability reduces to $q_a \in [0, r_a)$. Let Ω be the finite set of origin–destination pairs, where $\omega = (s, t) \in \Omega$ occurs with probability $p_\omega > 0$ ($\sum_{\omega \in \Omega} p_\omega = 1$). We assume a path exists from s to t for all $(s, t) \in \Omega$. If not, the defender can discard that scenario from consideration. In the second stage of SNIP, the attacker's origin and destination are realized. The attacker traverses the network from the origin to the corresponding destination via the maximum-reliability path given the defender's set of interdicted arcs.

For $s, t \in N$, a simple s-t path P is a set of arcs, say $(i_0, i_1), (i_1, i_2), \ldots, (i_{k-1}, i_k)$, where $i_0 = s$, $i_k = t$, and the nodes $i_0, i_1, i_2, \ldots, i_{|P|} \in N$ are all distinct. Let $\mathcal{P}_{st}$ be the set of all simple paths from $s \in N$ to $t \in N$. Let $S \subseteq D$ be the set of interdicted arcs, as chosen by the defender. The function

$$h_P(S) \coloneqq \left(\prod_{a \in P} r_a \right) \left(\prod_{a \in P \cap S} \frac{q_a}{r_a} \right)$$

calculates the probability of the attacker traversing the set of arcs P undetected given the interdicted arcs S.

In the stochastic network interdiction problem (SNIP), the defender selects arc to interdict to minimize the expected value of the attacker's maximum-reliability path, subject to the budget restriction on the selected arcs. SNIP can be formulated as

$$\begin{aligned} \min_{S} \quad & \sum_{\omega=(s,t)\in\Omega} p_\omega \max\{h_P(S)\colon P \in \mathcal{P}_{st}\} \\ \text{s.t.} \quad & \sum_{a\in S} c_a \le b. \end{aligned} \tag{1}$$

2.1 Deterministic equivalent formulation

A DEF of SNIP can be obtained by using first-stage binary variables x_a to represent whether the defender installs a sensor on arc $a \in D$. That is, $x_a = 1$ if the defender elects to install a sensor on arc $a \in D$, and $x_a = 0$ if no sensor is installed on that arc. Second-stage continuous variables π_i^ω represent the maximum probability of the attacker traveling undetected from $i \in N$ to t in scenario $\omega = (s,t) \in \Omega$.

The formulation uses second-stage constraints to calculate the π variables for each scenario $\omega \in \Omega$. The π variables are calculated using an LP formulation of the dynamic programming (DP) optimality conditions for calculating a maximum-reliability path. Using the convention $q_a \coloneqq r_a$ and $x_a \coloneqq 0$ for all $a \in A \setminus D$, each π_i^ω is calculated as

$$\pi_i^\omega = \max_{a=(i,j)\in A} \left\{ \pi_j^\omega \left[r_a + (q_a - r_a)x_a \right] \right\}, \quad i \in N,\ \omega \in \Omega. \tag{2}$$

The maximum probability of the attacker reaching t undetected from node $i \in N$ is the maximum over all forward adjacent nodes $j \in N$ of π_j^t, multiplied by r_a if the arc is not interdicted, or q_a if it is interdicted.

The optimality conditions (2) imply the set of inequalities

$$\pi_i^\omega \ge \pi_j^\omega \left[r_a + (q_a - r_a)x_a \right], \quad a = (i,j) \in A,\ \omega \in \Omega. \tag{3}$$

The inequalities (3) are nonlinear. Pan and Morton [21] derive the following DEF of SNIP that uses a linear reformulation of these inequalities:

$$\min_{x,\pi} \sum_{\omega=(s,t)\in\Omega} p_\omega \pi_s^\omega \tag{4a}$$

$$\text{s.t.} \sum_{a\in D} c_a x_a \le b \tag{4b}$$

$$\pi_i^\omega - r_a \pi_j^\omega \ge 0, \qquad a = (i,j) \in A \setminus D,\ \omega \in \Omega \tag{4c}$$

$$\pi_i^\omega - r_a \pi_j^\omega \ge -(r_a - q_a) u_j^\omega x_a, \qquad a = (i,j) \in D,\ \omega \in \Omega \tag{4d}$$

$$\pi_i^\omega - q_a \pi_j^\omega \ge 0, \qquad a = (i,j) \in D,\ \omega \in \Omega \tag{4e}$$

$$\pi_t^\omega = 1, \qquad \omega = (s,t) \in \Omega \tag{4f}$$

$$x_a \in \{0,1\}, \qquad a \in D. \tag{4g}$$

The objective function (4a) minimizes the expected value of the attacker's maximum-reliability path. For each scenario $\omega = (s,t) \in \Omega$, the parameter u_j^ω represents the value of the maximum-reliability path from $j \in N$ to t when no sensors are installed, and hence is an upper bound on π_j^ω. These parameters are calculated in a model preprocessing step. The linear constraints (4c)–(4e) formulate the nonlinear DP inequalities (3). Constraints (4c) enforce $\pi_i^\omega \ge r_a \pi_j^\omega$ for all $a = (i,j) \in A \setminus D$, $\omega \in \Omega$. For $\omega \in \Omega$ and $a = (i,j) \in D$, if $x_a = 0$, then (4d) becomes $\pi_i^\omega \ge r_a \pi_j^\omega$, which dominates (4e). On the other hand, if $x_a = 1$, then (4e) implies $\pi_i^\omega - r_a \pi_j^\omega \ge -(r_a - q_a)u_j^\omega$, (4e) dominates (4d). Thus, in either case (4c)–(4e) are equivalent to (3). Since the variables π_s^ω, $\omega = (s,t) \in \Omega$, have positive coefficients in the objective, this implies the equations (2) will be satisfied in an optimal solution.

2.2 Benders decomposition

Directly solving the DEF (4) with a mixed-integer programming solver may be too time-consuming due to its large size. Benders decomposition can be used to decompose large problems like SNIP. After introducing the SNIP formulation, Pan and Morton [21] outline a Benders decomposition algorithm for the SNIP DEF (4). For a fixed vector $x \in [0,1]^D$ satisfying $\sum_{a \in D} c_a x_a \leq b$, the π_j^ω variables in (4) can be obtained by solving

$$
\begin{aligned}
E^{st}(x) \coloneqq \min_{\pi} \ & \pi_s && && (5a)\\
\text{s.t. } & \pi_i - r_a \pi_j \geq 0, && a = (i,j) \in A \setminus D && (5b)\\
& \pi_i - r_a \pi_j \geq -(r_a - q_a) u_j^\omega x_a, && a = (i,j) \in D && (5c)\\
& \pi_i - q_a \pi_j \geq 0, && a = (i,j) \in D && (5d)\\
& \pi_t = 1 && && (5e)
\end{aligned}
$$

for each scenario $\omega = (s,t) \in \Omega$. The dual of (5) is

$$
\begin{aligned}
\max_{y,z} \quad & y_t - \sum_{a=(i,j)\in D} (r_a - q_a) u_j^\omega y_{ij} x_a && && \\
\text{s.t.} \quad & \sum_{(s,j)\in A} (y_{sj} + z_{sj}) = 1 && && \\
& \sum_{(i,j)\in A} (y_{ij} + z_{ij}) - \sum_{a=(j,i)\in A} (r_a y_{ji} + q_a z_{ji}) = 0, && i \in N \setminus \{s,t\} && (6)\\
& y_t - \sum_{a=(j,t)\in A} (r_a y_{jt} + q_a z_{jt}) = 0 && && \\
& y_{ij}, z_{ij} \geq 0, && (i,j) \in A && \\
& y_t \geq 0. && &&
\end{aligned}
$$

Dual variables y_{ij} correspond to DEF constraints (5b) and (5c), z_{ij} is the dual variable for constraints (5d), and y_t is the dual variable for constraint (5e). We fix $z_{ij} \coloneqq 0$ for all $(i,j) \in A \setminus D$, as constraint (5d) only applies to arcs D.

The Benders master problem is as follows:

$$
\begin{aligned}
\min_{x,\theta} \ & \sum_{\omega \in \Omega} p_\omega \theta^\omega && && (7a)\\
\text{s.t. } & \sum_{a \in D} c_a x_a \leq b && && (7b)\\
& \theta^\omega \geq \bar{y}_t^\omega - \sum_{a=(i,j)\in D} (r_a - q_a) u_j^\omega \bar{y}_{ij}^\omega x_a, && (\bar{y}^\omega, \bar{z}^\omega) \in K^\omega,\ \omega = (s,t) \in \Omega && (7c)\\
& \theta^\omega \geq 0, && \omega \in \Omega && (7d)\\
& x_a \in \{0,1\}, && a \in D. && (7e)
\end{aligned}
$$

The Benders algorithm begins by solving the master problem with no constraints of the form (7c) to obtain a candidate solution $(\bar{x}, \bar{\theta})$. At iteration k of the Benders cutting plane algorithm, a dual subproblem (6) is solved for each scenario using the candidate solution $(\bar{x}, \bar{\theta})$ to find a dual-feasible extreme point $(\bar{y}^\omega, \bar{z}^\omega)$. If a cut in the form of constraint (7c) cuts off the candidate solution $\bar{\theta}^\omega$, this cut is added to the Benders master problem by including the point in the set K^ω. The updated master problem is solved to generate a new candidate solution $(\bar{x}, \bar{\theta})$. This process is repeated until none of the scenario cuts constructed at an iteration of the algorithm cut off the candidate solution obtained by solving the master problem at the previous iteration.

Benders decomposition can also be used in a branch-and-cut algorithm. The integrality constraint (7e) is relaxed and enforced within the branch-and-bound tree. At each integer-feasible solution $(\bar{x}, \bar{\theta})$ obtained

at a node in the master problem branch-and-bound tree, a dual subproblem (6) is solved for each scenario. Violated cuts are added to the LP formulation of that node and it is re-solved. When no violated inequalities are found for an integer-feasible solution, the upper bound may be updated and the node pruned.

Pan and Morton [21] enhance the Benders branch-and-cut algorithm by using optimal solutions to scenario subproblems from previous iterations to create step inequalities for the relaxed Benders master problem. These additional inequalities are added to the formulation as cuts to tighten the LP relaxation of the master problem. This leads to a smaller branch-and-bound tree, which in turn reduces the time spent solving mixed-integer programs.

The general-purpose Benders approach tested by Bodur et al. [4] is very effective in solving SNIP. A key implementation detail of this approach is to first solve the relaxed Benders master problem, obtained by removing the integrality restriction on x. Once no more Benders inequalities can be added to the LP relaxation of the master problem, the integrality restriction on x is restored, and the model, including the identified Benders cuts, is passed to a mixed-integer programming solver to begin the Benders branch-and-cut algorithm. The solver adds its own general-purpose integrality-based cuts to the formulation using the initial set of Benders cuts. This results in a stronger LP relaxation bound, and reduces the size of the branch-and-bound tree.

Bodur et al. [4] employed a smaller gap tolerance as a stopping criterion (10^{-3}) for their Benders branch-and-cut implementation than Pan and Morton [21] used in their experiments (10^{-2}). We implemented the Benders branch-and-cut algorithm as in [4], and tested it with a relative optimality gap tolerance of 10^{-2} to compare the results to those published in [21]. With these matching tolerances, the Benders algorithm ran 3–4 times faster than Pan and Morton's algorithm. Although the difference in hardware used in these experiments makes it impossible to directly compare the performance of these methods, we conclude that the implementation of Benders branch-and-cut described in in [4] is currently among the most efficient ways to solve instances of SNIP, and hence we compare against this approach in our numerical experiments.

3 Compact deterministic equivalent formulation

For scenarios sharing a destination node, the DP optimality conditions (2) for SNIP are identical. Hence, the DEF (4) contains redundancies by repeating the LP formulation of these optimality conditions, constraints (4c)–(4f), for each destination node. Motivated by this observation, we present a compact DEF of SNIP that groups together second-stage constraints for scenarios with a common destination. Let T be the set of unique destination nodes: $T = \{t \in N \colon (s,t) \in \Omega \text{ for some } s \in N\}$. Second-stage variables π_j^t now represent the probability of the attacker successfully traveling from node $j \in N$ to destination $t \in T$ in any scenario having destination t. We obtain the following new DEF for SNIP:

$$\min_{x,\pi} \sum_{\omega=(s,t)\in\Omega} p_\omega \pi_s^t \tag{8a}$$

$$\text{s.t.} \sum_{a\in D} c_a x_a \le b \tag{8b}$$

$$\pi_i^t - r_a \pi_j^t \ge 0, \qquad a=(i,j) \in A \setminus D,\ t \in T \tag{8c}$$

$$\pi_i^t - r_a \pi_j^t \ge -(r_a - q_a) u_j^t x_a, \qquad a=(i,j) \in D,\ t \in T \tag{8d}$$

$$\pi_i^t - q_a \pi_j^t \ge 0, \qquad a=(i,j) \in D,\ t \in T \tag{8e}$$

$$\pi_t^t = 1, \qquad t \in T \tag{8f}$$

$$x_a \in \{0,1\}, \qquad a \in D. \tag{8g}$$

Parameter u_j^t is the value of the attacker's maximum-reliability path from $j \in N$ to $t \in T$ when no sensors are installed. By definition, $u_j^t = u_j^\omega$ for all $\omega = (s,t) \in \Omega$ and $j \in N$. The objective function (8a) weights each π_s^t variable by its respective scenario probability. The summation of these terms represents the overall probability of a successful attack, which the defender seeks to minimize.

Proposition 1. *Each feasible solution of* (4) *admits a feasible solution of* (8) *of equal objective value, and vice versa.*

Proof. Let $(\bar{x}, \bar{\pi})$ be a solution to (4). For all $t \in T$, select $s_t \in N$ such that $\omega = (s_t, t) \in \Omega$. Let $\hat{\pi}_i^t \coloneqq \bar{\pi}_i^\omega$ for all $i \in N$ and $t \in T$, and let $\hat{x}_{ij} \coloneqq \bar{x}_{ij}$ for all $(i,j) \in D$. $(\hat{x}, \hat{\pi})$ is feasible to (8) with objective value $\sum_{\omega=(s,t)\in\Omega} p_\omega \hat{\pi}_s^t = \sum_{\omega=(s,t)\in\Omega} p_\omega \bar{\pi}_s^\omega$.

Now, let $(\hat{x}, \hat{\pi})$ be a solution to (8). Let $\bar{\pi}_i^\omega \coloneqq \hat{\pi}_i^t$ for all $i \in N$ and $\omega = (s,t) \in \Omega$. Let $\bar{x}_{ij} \coloneqq \hat{x}_{ij}$ for all $(i,j) \in D$. $(\bar{x}, \bar{\pi})$ is a solution to (4) with objective value $\sum_{\omega\in\Omega} p_\omega \bar{\pi}_s^\omega = \sum_{\omega\in\Omega} p_\omega \hat{\pi}_s^t$. □

A Benders algorithm can be applied to the compact formulation by introducing variables θ^t to represent the probability-weighted sum of maximum-reliability path values for scenarios ending at node $t \in T$. We experimented with a Benders decomposition on the DEF (8) using the Benders algorithm of Bodur et al. [4]. We found that the Benders algorithm performed much worse on this formulation than on the DEF (4). The reason for this poor performance is that the root relaxation obtained after the mixed-integer programming solver added its general-purpose cuts was much weaker when using a Benders reformulation of (8), as compared to the Benders reformulation (7). This is consistent with the results of [4], which indicates that the integrality-based cuts derived in the formulation (7) can be stronger than those derived in a "projected" Benders master problem that only uses θ^t variables.

4 Path-based formulation and cuts

We now derive a new mixed-integer linear programming formulation of SNIP based on the path-based formulation (1). With the introduction of binary variables x_a to denote whether network arc $a \in D$ is interdicted, we map the set function h_P to a vector function $\bar{h}_P$. In particular, for $S \subseteq D$ we let $\chi^S \in \{0,1\}^D$ be the characteristic vector of the set S: $\chi_a^S = 1$ if $a \in S$, and 0 otherwise. Then we define $\bar{h}_P : \{0,1\}^D \to \mathbb{R}$ as

$$\bar{h}_P(x) \coloneqq \left[\prod_{a\in P} r_a\right]\left[\prod_{a\in P\cap D}\left(\frac{q_a}{r_a}\right)^{x_a}\right]$$

so that $h_P(S) = \bar{h}_P(\chi^S)$ for $S \subseteq D$.

Then model (1) can be formulated as

$$\begin{aligned}
\min_{x,\pi} \quad & \sum_{\omega=(s,t)\in\Omega} p_\omega \pi_s^t \\
\text{s.t.} \quad & \sum_{a\in D} c_a x_a \le b \\
& \pi_s^t \ge \max\{\bar{h}_P(x) \colon P \in \mathcal{P}_{st}\}, \quad (s,t) \in \Omega \\
& x_a \in \{0,1\}, \qquad\qquad\qquad\quad a \in D.
\end{aligned} \tag{9}$$

We use the special structure of h_P for each $P \in \mathcal{P}_{st}$ to build a linear formulation of model (9).

Definition 1 (E.g., [22])**.** *$h \colon 2^{\mathcal{N}} \to \mathbb{R}$ is a supermodular set function over a ground set $\mathcal{N}$ if $h(S_1 \cup \{a\}) - h(S_1) \le h(S_2 \cup \{a\}) - h(S_2)$ for all $S_1 \subseteq S_2 \subseteq \mathcal{N}$ and $a \in \mathcal{N} \setminus S_2$.*

Proposition 2. *$h_P(\cdot)$ is a supermodular set function.*

Proof. Let $S_1, S_2 \subseteq D$ with $S_1 \subseteq S_2$. Let $a' \in D \setminus S_2$. The result is immediate if $a' \notin P$. Therefore, assume $a' \in P$ to obtain

$$\begin{aligned}
h_P(S_1 \cup \{a'\}) - h_P(S_1) &= \left(\prod_{a\in P} r_a\right)\left(\prod_{a\in P\cap S_1}\frac{q_a}{r_a}\right)\left(\frac{q_{a'}}{r_{a'}} - 1\right) \\
&\le \left(\prod_{a\in P} r_a\right)\left(\prod_{a\in P\cap S_2}\frac{q_a}{r_a}\right)\left(\frac{q_{a'}}{r_{a'}} - 1\right) \\
&= h_P(S_2 \cup \{a'\}) - h_P(S_2). \qquad \square
\end{aligned}$$

□

The maximum of a set of supermodular functions is not in general a supermodular function, so $\max_{P\in\mathcal{P}_{st}} h_P(S)$ is not necessarily supermodular. To exploit the supermodular structure for each individual path, we consider an equivalent formulation of (9) that contains an inequality for every scenario $(s,t)\in\Omega$ and every path from s to t:

$$\min_{\substack{x,\pi\\ \omega=(s,t)\in\Omega}} \sum p_\omega \pi_s^t \tag{10a}$$

$$\text{s.t.} \sum_{a\in D} c_a x_a \le b \tag{10b}$$

$$\pi_s^t \ge \bar{h}_P(x), \qquad P\in\mathcal{P}_{st},\ (s,t)\in\Omega \tag{10c}$$

$$x_a\in\{0,1\}, \qquad a\in D. \tag{10d}$$

Each second-stage constraint includes a supermodular function on the right-hand side.

Consider the mixed-integer feasible region of (10) for a fixed scenario $(s,t)\in\Omega$ and path $P\in\mathcal{P}_{st}$:

$$H_P \coloneqq \left\{(x,\pi)\in\{0,1\}^D\times\mathbb{R}\colon \pi\ge\bar{h}_P(x)\right\}. \tag{11}$$

We are interested in a linear formulation of H_P. Let $\rho_P^a(S)\coloneqq h_P(S\cup\{a\})-h_P(S)$ be the marginal difference function of the set function h_P with respect to arc $a\in D$. Nemhauser et al. [19] provide an exponential family of linear inequalities that can be used to define H_P. Applied to H_P, these inequalities are

$$\pi \ge h_P(S) - \sum_{a\in S}\rho_P^a(D\setminus\{a\})(1-x_a) + \sum_{a\in D\setminus S}\rho_P^a(S)x_a, \quad S\subseteq D \tag{12}$$

$$\pi \ge h_P(S) - \sum_{a\in S}\rho_P^a(S\setminus\{a\})(1-x_a) + \sum_{a\in D\setminus S}\rho_P^a(\emptyset)x_a, \quad S\subseteq D. \tag{13}$$

Only one of the sets of inequalities (12) and (13) is required to define H_P [18]. Using these inequalities, the feasible region of H_P can be formulated as

$$H_P \coloneqq \left\{(x,\pi)\in\{0,1\}^D\times\mathbb{R}\colon (12) \text{ or } (13)\right\}.$$

The number of inequalities required to define model (10) in this manner grows exponentially with the number of arcs in a path, and the number of s–t paths grows exponentially with the size of the network. Enumerating all s–t paths for scenario $(s,t)\in\Omega$ is impractical. Nevertheless, (10) lends itself to a delayed constraint generation algorithm. Instead of adding inequalities for all possible paths, we add violated valid inequalities as needed (see Section 4.4).

To demonstrate the potential of this formulation, we next show that if we could obtain the convex hull of each set H_P, this would yield a relaxation that is at least as strong as the LP relaxation of the DEF (4) (and (8)). This implies that the path-based formulation has the potential to yield a better LP relaxation if we can identify strong valid inequalities for $\text{conv}(H_P)$.

Theorem 1. *Consider the following continuous relaxation of* (10)*:*

$$\begin{aligned}
\min_{\pi}\ & \sum_{\omega=(s,t)\in\Omega} p_\omega \pi_s^\omega \\
\text{s.t.}\ & \sum_{a\in D} c_a x_a \le b \\
& (x,\pi_s^\omega)\in\text{conv}(H_P), \quad P\in\mathcal{P}_{st},\ \omega=(s,t)\in\Omega \\
& x_a\in[0,1], \qquad a\in D.
\end{aligned} \tag{14}$$

For all $(\bar{x},\bar{\pi})$ feasible to (14)*, there exists $\hat{\pi}$ such that $(\bar{x},\hat{\pi})$ is feasible to* (4) *and has objective value in* (4) *not greater than the objective value of $(\bar{x},\bar{\pi})$ in* (14)*.*

The above theorem is a consequence of Lemma 1. For a fixed $(s,t) \in \Omega$ and $x \in [0,1]^D$ satisfying $\sum_{a\in D} c_a x_a \le b$, define

$$O^{st}(x) \coloneqq \min_{\pi_s^t} \pi_s^t$$
$$\text{s.t. } (x, \pi_s^t) \in \text{conv}(H_P),\ P \in \mathcal{P}_{st}.$$

Recall that $E^{st}(x)$ is defined in (5).

Lemma 1. *Let $\omega = (s,t) \in \Omega$ and $x \in [0,1]^D$ satisfy $\sum_{a\in D} c_a x_a \le b$. Then,*

$$E^{st}(x) \le O^{st}(x).$$

Proof. The vector of all ones in $\mathbb{R}^{|N|}$ is feasible to (5). A feasible solution to (6) can be constructed as follows. Let $P = \{(i_1,i_2),(i_2,i_3),\dots,(i_{n-1},i_n)\} \in \mathcal{P}_{st}$ be a simple s–t path, where $i_1 = s$ and $i_n = t$. Let $\bar{y}_{i_1,i_2} = 1$. For $k = 2,\dots,n$, let $\bar{y}_{i_k,i_{k+1}} = r_{i_{k-1},i_k}\bar{y}_{i_{k-1},i_k}$. Let $\bar{y}_{ij} = 0$ for all $(i,j) \in A \setminus P$, and $\bar{z}_{ij} = 0$ for all $(i,j) \in A$. Then $(\bar{y},\bar{z})$ is feasible to (6). Because the feasible regions of (5) and (6) are nonempty and the objectives are bounded below by zero, both problems have an optimal solution. Let π^* be an optimal solution to (5), with corresponding optimal dual solution (y^*,z^*).

Assume first there exists an s–t path P^* such that one of the constraints (5b)–(5d) is satisfied at equality for all $a \in P^*$. Thus,

$$\pi_i^* = \max\left\{r_a\pi_j^* - (r_a - q_a)u_j^\omega x_a,\ q_a\pi_j^*\right\}$$

for all $a = (i,j) \in P^*$. Removing constraints for arcs $a \in A \setminus P^*$ from the feasible region of (5) does not change the problem's optimal solution. However, the optimal objective value of the resulting formulation, $\pi_s^* = E^{st}(x)$, is a lower bound on $\min\{\pi_s \colon (x,\pi_s) \in \text{conv}(H_{P^*})\}$, because $\text{conv}(H_{P^*})$ is a subset of the modified feasible region of (5). Therefore,

$$E^{st}(x) \le \min\{\pi_s \colon (x,\pi_s) \in \text{conv}(H_{P^*})\} \le O^{st}(x).$$

Finally, assume there is no s–t path $P^* \in \mathcal{P}_{st}$ such that one of the constraints (5b)–(5d) is satisfied at equality for all $a \in P^*$. Consider any s–t path $P \in \mathcal{P}_{st}$. By assumption, there exists an arc $(i,j) \in P$ such that constraints (5b)–(5d) are not binding. By complementary slackness, the corresponding dual optimal y_{ij}^* and z_{ij}^* are 0, and the flow of that path through the network is 0. Because there exists no sequence of arcs through the dual network of (6) that has positive flow, the dual optimal value is bounded above by 0. Thus $E^{st}(x) \le 0 \le O^{st}(x)$. □ □

Proof of Theorem 1. From Lemma 1, we have

$$\sum_{\omega=(s,t)\in\Omega} p_\omega \bar{\pi}_s^\omega \ge \sum_{\omega=(s,t)\in\Omega} p_\omega O^{st}(\bar{x}) \ge \sum_{\omega=(s,t)\in\Omega} p_\omega E^{st}(\bar{x}).$$

This implies there exists $\hat{\pi} \in \mathbb{R}^{N\times\Omega}$ such that $(\bar{x},\hat{\pi})$ is feasible to (4) with objective value $\sum_{\omega=(s,t)\in\Omega} p_\omega \hat{\pi}_s^\omega \le \sum_{\omega=(s,t)\in\Omega} p_\omega \bar{\pi}_s^\omega$. □ □

Computational experiments by Ahmed and Atamürk [1] show that the inequalities (12) and (13) may provide a poor approximation of the set $\text{conv}(H_P)$. Thus, in the following sections, we describe additional inequalities that can used to approximate $\text{conv}(H_P)$ for all $P \in \mathcal{P}_{st}$. We consider three different classes of inequalities, based on the traversal probabilities of interdicted arcs.

4.1 Inequalities for the $q > 0$ case

In this section, we assume $q_a > 0$ for all $a \in D$. For a ground set $\mathcal{N}$, Ahmed and Atamürk [1] study valid inequalities for the mixed-integer set

$$F = \left\{(x,w) \in \{0,1\}^{\mathcal{N}} \times \mathbb{R} \colon w \le f\left(\textstyle\sum_{i\in\mathcal{N}} \alpha_i x_i + \beta\right)\right\}, \tag{15}$$

where $\alpha \in \mathbb{R}^{\mathcal{N}}_+$, $\beta \in \mathbb{R}$, and f is a strictly concave, increasing, differentiable function. They derive valid inequalities for the set F and prove that they dominate the submodular inequality equivalents of (12) and (13) applied to F. These improved inequalities are shown empirically to yield significantly better relaxations than (12) and (13).

We translate the set H_P to match the structure of F. Let $\mathcal{N} \coloneqq D$, $w \coloneqq -\pi$, $f(u) \coloneqq -\exp(-u)$, $\beta \coloneqq -\sum_{a \in P} \log(r_a)$. For $a \in D$, let

$$\alpha_a \coloneqq \begin{cases} \log(r_a) - \log(q_a) & \text{if } a \in P \cap D, \\ 0 & \text{otherwise.} \end{cases}$$

Because $\log(r_a) > \log(q_a)$ for all $a \in D$, we have $\alpha \in \mathbb{R}^D_+$. With these definitions, H_P is expressed in the form of F.

We now describe how to calculate valid inequalities for H_P using the results of Ahmed and Atamtürk [1]. Without loss of generality, let $D \setminus S \coloneqq \{1, 2, \ldots, m\}$ be indexed such that $\alpha_1 \geq \alpha_2 \geq \ldots \geq \alpha_m$, and let $A_k = \sum_{a=1}^{k} \alpha_a$ for $k \in D \setminus S$, with $A_0 = 0$. Also define $\alpha(S) = \sum_{a \in S} \alpha_a$. The subadditive lifting inequality

$$\pi \geq h_P(S) - \sum_{a \in S} \phi(-\alpha_a)(1 - x_a) + \sum_{a \in D \setminus S} \rho^a_P(S) x_a \tag{16}$$

is valid for H_P for any set of interdicted arcs $S \subseteq D$, where ϕ is calculated as follows [1]. Consider the function $\zeta \colon \mathbb{R}_- \to \mathbb{R}$, calculated according to Algorithm 1. For a provided value of η, Algorithm 1 returns

Algorithm 1: Computing $\zeta(\eta)$

$k \leftarrow 0$;
while $k < m$ **and** $A_k + \eta < 0$ **do**
  $k \leftarrow k + 1$;
end
$\zeta(\eta) \leftarrow -\exp(-\alpha(S) - A_k - \beta - \eta) + \sum_{a=1}^{k} \rho^a_P(S) + \exp(-\alpha(S) - \beta)$;

a specific k. With this k, let $\phi \colon \mathbb{R}_- \to \mathbb{R}$ be defined as

$$\phi(\eta) \coloneqq \begin{cases} \zeta(\mu_k - A_{k-1}) + \rho^k_P(S) \frac{b_k(\eta)}{\alpha_k} & \text{if } \mu_k - A_k \leq \eta \leq \mu_k - A_{k-1}, \\ \zeta(\eta) & \text{otherwise,} \end{cases}$$

where $\mu_k = -\log(-\rho^k_P(S)/\alpha_k) - \alpha(S) - \beta$ and $b_k(\eta) = \mu_k - A_{k-1} - \eta$. Inequality (16) dominates the general supermodular inequality (12) [1].

We now construct Ahmed and Atamtürk's inequalities that dominate the supermodular inequalities (13). Let $S = \{1, 2, \ldots, n\}$ be indexed such that $\alpha_1 \geq \alpha_2 \geq \ldots \geq \alpha_n$. Also, let $A_k = \sum_{a=1}^{k} \alpha_a$ for $k \in S$, with $A_0 = 0$. We define the function $\xi \colon \mathbb{R}_+ \to \mathbb{R}$ to be calculated according to Algorithm 2.

Algorithm 2: Computing $\xi(\eta)$

$k \leftarrow 0$;
while $k < r$ **and** $A_k < \eta$ **do**
  $k \leftarrow k + 1$;
end
$\xi(\eta) \leftarrow -\exp(-\alpha(S) + A_k - \beta - \eta) - \sum_{a=1}^{k} \rho^a_P(S \setminus \{a\}) + \exp(-\alpha(S) - \beta)$;

With the η-dependent k calculated in Algorithm 2, let

$$\psi(\eta) \coloneqq \begin{cases} \xi(A_k - \nu_k) + \rho^k_P(S \setminus \{k\}) \frac{b_k(\eta)}{\alpha_k} & \text{if } A_{k-1} - \nu_k \leq \eta \leq A_k - \nu_k, \\ \xi(\eta) & \text{otherwise,} \end{cases}$$

where $\nu_k = \alpha(S) + \log(-\rho_P^k(S \setminus \{k\})/\alpha_k) - \beta$ and $b_k(\eta) = A_k - \nu_k - \eta$ for every $k \in \{1, 2, \ldots, r\}$. For any $S \subseteq D$, the inequality

$$\pi \geq h_P(S) - \sum_{a \in S} \rho_P^a(S \setminus \{a\})(1 - x_a) - \sum_{a \in D \setminus S} \psi(\alpha_a) x_a \tag{17}$$

is valid for H_P. For all $a \in D \setminus P$, the coefficients in inequalities (16) and (17) are 0.

We next discuss separation of the inequalities (16) and (17) for use in a delayed constraint generation framework. Given a point $(\bar{x}, \bar{\pi}) \in [0,1]^D \times \mathbb{R}$, we want to determine if there is a set S such that $(\bar{x}, \bar{\pi})$ violates either (16) or (17). For an integral vector $\bar{x}$, we construct the set $S = \{a \in D \colon \bar{x}_a = 1\}$. If $\bar{\pi} < h_P(S)$, then inequalities (16) and (17) are violated and can be added to the formulation of H_P to cut off $(\bar{x}, \bar{\pi})$.

If $\bar{x}$ is not integral, we use Ahmed and Atamtürk's scheme to heuristically construct a set S using $\bar{x}$. This set is used in inequalities (16) and (17) to cut off $(\bar{x}, \bar{\pi})$, if possible. We first consider constructing a set S to apply inequality (16). We first solve the following nonlinear program:

$$\max_{z \in [0,1]^D} \ \frac{-\bar{\pi} + 1}{-\exp(-\sum_{a \in D} \alpha_a z_a - \beta) + 1} + \sum_{a \in D} \frac{\rho_P^a(\emptyset)}{h_P(\emptyset) + 1} \bar{x}_a (1 - z_a). \tag{18}$$

Given a solution $\bar{z}$, we greedily round the fractional components in $\bar{z}$ that result in the least reduction in objective value when rounded to either 0 or 1. Given the rounded solution, say $\bar{z}'$, we set $S = \{a \in D \colon \bar{z}'_a = 1\}$. If $\pi < h_P(S) - \sum_{a \in S} \phi(-\alpha_a)(1 - \bar{x}_a) + \sum_{a \in D \setminus S} \rho_P^a(S) \bar{x}_a$, valid inequality (16) can be added to the relaxation to cut off $(\bar{x}, \bar{\pi})$.

We solve a related nonlinear program to find a set S to apply inequality (17):

$$\max_{z \in [0,1]^D} \ \frac{-\bar{\pi} + 1}{-\exp(-\sum_{a \in D} \alpha_a z_a - \beta) + 1} - \sum_{a \in D} \frac{\rho_P^a(\emptyset)}{h_P(\{a\}) + 1} (1 - \bar{x}_a) z_a. \tag{19}$$

We again greedily round the solution of (18) to obtain the characteristic vector of S. If $\pi < h_P(S) - \sum_{a \in S} \rho_P^a(S \setminus \{a\})(1 - \bar{x}_a) - \sum_{a \in D \setminus S} \psi(\alpha_a) \bar{x}_a$, inequality (17) cuts off $(\bar{x}, \bar{\pi})$.

Yu and Ahmed [25] consider deriving stronger valid inequalities by imposing a knapsack constraint on the set F given in equation (15). In particular,

$$F = \left\{(x, w) \in \{0,1\}^{\mathcal{N}} \times \mathbb{R} \colon w \leq f(\textstyle\sum_{i \in \mathcal{N}} \alpha_i x_i + \beta),\ \sum_{i \in \mathcal{N}} x_i \leq k\right\},$$

where $k > 0$. Although our test instances include this structure, we do not explore applying their results here. In particular, when interdicting arcs along an *individual* path, the knapsack constraint is only relevant if the number of arcs in the path under consideration is larger than the defender's budget.

4.2 Inequalities for $q = 0$ case

When $q_a = 0$ for all $a \in D$, $\log(q_a)$ is no longer defined, and the results of Section 4.1 do not apply. We consider a different class of inequalities for this special case of q.

If all interdicted arc probabilities are 0, $\bar{h}_P(x)$ simply reduces to

$$\bar{h}_P(x) = \left[\prod_{a \in P} r_a\right] \left[\prod_{a \in P \cap D} (1 - x_a)\right].$$

If any arc $a \in P \cap D$ is interdicted, $\bar{h}_P(x)$ equals 0. By setting $\hat{\pi} \coloneqq \pi / \prod_{a \in P} r_a$, our relevant mixed-integer linear set is

$$H_P = \left\{(x, \hat{\pi}) \in \{0,1\}^D \times \mathbb{R} \colon \hat{\pi} \geq \textstyle\prod_{a \in P \cap D} (1 - x_a)\right\}.$$

We are again interested in valid inequalities for H_P. The inequalities

$$\hat{\pi} \geq 1 - \sum_{a \in P \cap D} x_a \tag{20}$$

and $\hat{\pi} \geq 0$ are valid for H_P [8] and define its convex hull [3].

4.3 Inequalities for the mixed case

We now consider the case where $q_a > 0$ for some, but not all, $a \in D$. Let $P_+ \coloneqq \{a \in P \cap D \colon q_a > 0\}$ and $P_0 \coloneqq \{a \in P \cap D \colon q_a = 0\}$. For $S \subseteq A$ let $r(S) \coloneqq \prod_{a \in S} r_a$. We write $\bar{h}_P(x)$ as

$$\bar{h}_P(x) = r(P \setminus D)\bar{h}_{P_+}(x)\bar{h}_{P_0}(x). \tag{21}$$

We introduce two new variables, $\pi_+ \in [0, r(P_+)]$ and $\pi_0 \in [0, r(P_0)]$ to represent $\bar{h}_{P_+}(x)$ and $\bar{h}_{P_0}(x)$, respectively, and arrive at the formulation:

$$\pi \geq r(P \setminus D)\pi_+\pi_0 \tag{22}$$
$$\pi_+ \geq \bar{h}_{P_+}(x) \tag{23}$$
$$\pi_0 \geq \bar{h}_{P_0}(x). \tag{24}$$

Our approach is to use results from Sections 4.1 and 4.2 to derive valid inequalities for (23) and (24), respectively, and relax the nonconvex constraint (22) using the McCormick inequalities [15], which in this case reduce to

$$\pi \geq r(P \setminus D)\left[r(P_0)\pi_+ - r(P_+)(r(P_0) - \pi_0)\right] \tag{25}$$

and $\pi \geq 0$.

Let $\gamma^S \in \mathbb{R}^{P_+}$ and constant $\delta^S \in \mathbb{R}$ be coefficients from one inequality of the form (16) (with P_+ taking the place of P) for some $S \subseteq P_+$. An inequality of the form (17) may also be used. We relax (23) to the constraint

$$\pi_+ \geq \sum_{a \in P_+} \gamma_a^S x_a + \delta^S. \tag{26}$$

Inequality (24) is relaxed as $\pi_0 \geq 0$, and, applying (20),

$$\pi_0 \geq r(P_0)\left(1 - \sum_{a \in P_0} x_a\right). \tag{27}$$

Finally, we project variables π_+ and π_0 out of inequality (25) using Fourier-Motzkin elimination with constraints (26)–(27) and $\pi_0 \geq 0$. This results in the two inequalities

$$\pi \geq r(P \setminus D)r(P_0)\left[\sum_{a \in P_+} \gamma_a^S x_a + \delta^S - r(P_+)\sum_{a \in P_0} x_a\right] \tag{28}$$
$$\pi \geq r(P \setminus D)r(P_0)\left[\sum_{a \in P_+} \gamma_a^S x_a + \delta^S - r(P_+)\right]. \tag{29}$$

Inequality (29) is dominated by $\pi \geq 0$, because $\sum_{a \in P_+} \gamma_a^S x_a + \delta^S \leq \pi_+ \leq r(P_+)$. Therefore, inequality (28) is the only inequality we obtain from this procedure for the given inequality (16) defined by S.

We next argue that for any integer solution $(\bar{x}, \bar{\pi}) \in \{0,1\}^D \times \mathbb{R}_+$, if it is not feasible (i.e., $(\bar{x}, \bar{\pi}) \notin H_P$), then we can efficiently find an inequality of the form (28) that this solution violates. Indeed, first observe that if $\bar{x}_a = 1$ for any $a \in P_0$ then $\bar{h}_P(\bar{x}) = 0 \leq \bar{\pi}$, by assumption, and hence the solution is feasible. So, we may assume $\bar{x}_a = 0$ for all $a \in P_0$. We set $S = \{a \in P_+ : \bar{x}_a = 1\}$, which yields $\sum_{a \in P_+} \gamma_a^S \bar{x}_a + \delta^S = h_{P_+}(S) = \bar{h}_{P_+}(\bar{x})$. Inequality (28) thus yields

$$\pi \geq r(P \setminus D)r(P_0)\left[\bar{h}_{P_+}(\bar{x}) - 0\right] = \bar{h}_P(\bar{x})$$

by (21) since $\bar{h}_{P_0}(\bar{x}) = r(P_0)$.

4.4 Path-based branch-and-cut algorithm

We next describe how the inequalities derived in Sections 4.1–4.3 can be used within a branch-and-cut algorithm to solve the path-based formulation (10). Similar to the use of Benders decomposition in a branch-and-cut algorithm described in Section 2.2, the algorithm is based on solving a master problem via branch-and-cut in which the constraints (10c) are relaxed and approximated with cuts.

Let $(\bar{x}, \bar{\pi})$ be a solution obtained in the branch-and-cut algorithm with integral $\bar{x}$. Since (10c) are relaxed, we must check if this solution satisfies these constraints, and if not, finding a cut that this solution violates. We assign arc traversal probabilities σ_a for $a \in D$ according to the formula

$$\sigma_a = r_a^{1-\bar{x}_a} q_a^{\bar{x}_a}. \tag{30}$$

For each $\omega = (s,t) \in \Omega$, we find a maximum-reliability path $\bar{P} \in \mathcal{P}_{st}$. If $\bar{\pi}_s^t \geq \bar{h}_{\bar{P}}(\bar{x})$ then this solution is feasible to (10c) for this ω, since, by construction, $\bar{h}_{\bar{P}}(\bar{x}) = \max_{P \in \mathcal{P}_{st}} \bar{h}_P(\bar{x})$. Otherwise, depending on if $q_a = 0$ for any or all $a \in \bar{P}$, we derive a cut from one of Sections 4.1–4.3, using this path $\bar{P}$ and scenario ω. By construction, for the solution $\bar{x}$, this cut enforces that $\pi_s^t \geq \bar{h}_{\bar{P}}(\bar{x})$, and hence (i) cuts off the current infeasible solution $(\bar{x}, \bar{\pi})$, and (ii) implies that any solution (x, π) of the updated master problem with $x = \bar{x}$ will satisfy (10c) for this ω. This ensures that the branch-and-cut algorithm is finite (since at most finitely many such cuts are needed at finitely many integer solutions) and correct (since any infeasible solution obtained in the algorithm will be cut off).

One may also attempt to generate cuts at solutions $(\bar{x}, \bar{\pi})$ in which $\bar{x}$ is not necessarily integer, in order to improve the LP relaxation. To find a path for a scenario $\omega = (s,t)$ in this case, we again use formula (30) to define arc reliabilities, and then find the most reliable s-t path. With this path, we again apply the methods in Sections 4.1–4.3 to attempt to derive a violated cut. When $\bar{x}$ is not integral, this approach is not guaranteed to find a violated cut, even if one exists. Thus, to increase the chances a cut is found, we may also consider identifying another path by assigning arc costs as follows:

$$\sigma_a = (1 - \bar{x}_a) r_a + \bar{x}_a q_a,$$

and then finding a maximum-reliability path using these arc reliabilities. Our preliminary tests did not reveal any obvious benefit of using one particular arc-reliability calculation method.

In our implementation, before starting the branch-and-cut algorithm we solve a relaxation of the master problem (10) in which constraints (10c) are dropped, and the integrality constraints are relaxed. After solving this LP relaxation, we attempt to identify violated cuts for the relaxation solution, and if found, we add them to the master relaxation and repeat this process until no more violated cuts are found. We then begin the branch-and-cut process with all cuts found when solving the LP relaxation included in the mixed-integer programming formulation. This allows the solver to use these cuts to generate new cuts in the master problem relaxation. At any point in the branch-and-bound process where a solution $(\bar{x}, \bar{\pi})$ with $\bar{x}$ integral is obtained, we identify if there is any violated cut (as discussed above) and if so add it to the formulation via the lazy constraint callback function.

5 Computational experiments

We test the different solution methods on SNIP instances from Pan and Morton [21], which consist of a network of 783 nodes and 2586 arcs, 320 of which can be interdicted by the defender. Each instance considers the same 456 scenarios. We consider three cases for the q vector outlined by Pan and Morton [21]. In particular, we consider instances with $q \coloneqq 0.5r$, $q \coloneqq 0.1r$, and $q \coloneqq 0$. Pan and Morton also considered q_a values independently sampled from a $U(0, 0.5)$ distribution. Since the DEF for most of these instances could be solved within 30 seconds with little or no branching, we exclude these in our experiments. The cost of interdiction is $c_a = 1$ for all $a \in D$. Seven different budget levels were tested for each network and q level, and there are five test instances for each budget level and q level.

We consider the following algorithms in our tests.

- DEF: Solve DEF (4) with a MIP solver
- C-DEF: Solve compact DEF (8) with a MIP solver

		Average solve time in seconds (# unsolved)			
q	b	DEF	C-DEF	BEN	PATH
$0.5r$	30	(2)1652.1	23.9	298.5	54.1
	40	(2)2678.1	35.6	337.7	32.9
	50	(2)2242.4	31.7	355.8	42.8
	60	(2)1891.7	34.0	454.1	70.4
	70	(1)1730.0	27.4	487.4	63.9
	80	826.0	14.4	386.7	27.0
	90	460.3	6.9	385.0	26.8
$0.1r$	30	(5)	117.5	438.7	55.3
	40	(5)	466.0	812.3	301.2
	50	(5)	327.9	787.2	265.3
	60	(5)	638.6	793.7	194.1
	70	(5)	851.6	983.0	335.0
	80	(5)	(1)1257.4	(1)1106.3	(1)891.1
	90	(5)	(2)2329.4	(3)1269.4	(2)1497.5
0	30	(5)	96.1	487.0	61.7
	40	(5)	196.4	556.8	177.0
	50	(5)	332.3	663.6	250.8
	60	(5)	369.0	1141.8	374.8
	70	(5)	341.0	739.5	354.4
	80	(5)	262.9	535.1	151.5
	90	(5)	540.0	765.7	635.2

Table 1: Computational results

- BEN: Benders branch-and-cut algorithm on DEF (4); refer to (7)
- PATH: Branch-and-cut algorithm on path-based decomposition (1)

DEF and C-DEF are solved using the commercial MIP solver IBM ILOG CPLEX 12.6.3. BEN is the Benders branch-and-cut algorithm as implemented in Bodur et al. [4, Section 4]. The branch-and-cut algorithms, BEN and PATH, were implemented within the CPLEX solver using lazy constraint callbacks. Ipopt 3.12.1 was used to solve nonlinear models (18) and (19) in PATH. The computational tests were run on a 12-core machine with two 2.66GHz Intel Xeon X5650 processors and 128GB RAM, and one-hour time limit was imposed. We allowed four threads for DEF, while all other algorithms were limited to one thread. We used the CPLEX default relative gap tolerance of 10^{-4} to terminate the branch-and-bound process. The algorithms were written in Julia 0.4.5 using the Julia for Mathematical Optimization framework [14].

In our implementation of the BEN and PATH algorithms, when generating cuts at the root LP relaxation, we used the following procedure to focus the computational effort on scenarios that yield cuts. After each iteration, we make a list of scenarios for which violated cuts were found in that iteration. Then, in the next iteration, we first only attempt to identify cuts for scenarios in this list. If successful, the list is further reduced to the subset of scenarios that yielded a violated cut. If we fail to find any violated cuts in the current list, we re-initialize the list with all scenarios and attempt to identify violated cuts for each scenario.

Table 1 reports the average computational time, in seconds, over the five instances for each combination of the vector q and budget level b. The average times include only instances solved within the time limit – the number of unsolved instances are enclosed in parentheses. We find that PATH and C-DEF are consistently faster than both DEF and BEN. PATH and C-DEF have comparable performance, with C-DEF being somewhat faster on the $q = 0.5r$ instances (the easiest set) and PATH being somewhat faster on the $q = 0.1r$ instances (the hardest set).

Table 2 reports the root gap after adding LP cuts for the decomposition algorithms BEN and PATH, relative to the optimal solution value, before and after the addition of cuts from CPLEX. The root gap

		LP relaxation		LP relaxation after CPLEX cuts		
q	b	BEN	PATH	C-DEF	BEN	PATH
$0.5r$	30	10.64%	10.64%	2.38%	6.11%	6.03%
	40	11.34%	11.35%	2.80%	6.28%	6.20%
	50	11.22%	11.23%	2.40%	5.60%	5.62%
	60	10.54%	10.55%	2.07%	4.76%	4.80%
	70	8.88%	8.89%	1.83%	4.14%	4.34%
	80	6.25%	6.25%	0.97%	3.13%	3.19%
	90	3.92%	3.91%	0.43%	1.92%	2.02%
$0.1r$	30	22.47%	22.49%	6.72%	6.11%	6.67%
	40	26.22%	26.18%	9.35%	8.73%	8.21%
	50	27.54%	27.48%	9.38%	9.01%	9.28%
	60	28.16%	28.08%	9.93%	8.75%	9.35%
	70	28.92%	28.83%	9.67%	9.30%	9.70%
	80	30.88%	30.90%	10.85%	11.57%	11.96%
	90	33.07%	32.98%	12.45%	15.18%	15.46%
0	30	25.15%	25.30%	7.07%	6.72%	6.50%
	40	28.45%	28.55%	9.62%	8.07%	9.25%
	50	30.54%	30.78%	11.02%	10.03%	11.51%
	60	32.07%	32.45%	12.37%	12.48%	14.20%
	70	32.60%	33.30%	11.21%	11.82%	14.92%
	80	33.28%	33.99%	10.43%	11.44%	15.87%
	90	36.17%	36.97%	12.89%	15.45%	20.94%

Table 2: Average root gaps before and after CPLEX cuts. The LP relaxation of BEN is the same as DEF and C-DEF.

		Branch-and-bound nodes				Cut gen. time (sec.)	
q	b	DEF	C-DEF	BEN	PATH	BEN	PATH
0.5r	30	(2)1080.7	1406.6	39800.8	51512.2	266.5	18.9
	40	(2)1982.7	2470.2	21909.8	7697.8	313.6	21.2
	50	(2)1996.3	2516.4	24112	17161	324.8	21.1
	60	(2)1723.7	3270.6	35094.2	35860.2	404.5	22
	70	(1)1464	2242.4	24387.2	34456.2	456.9	22.1
	80	727	1235.4	5989.4	3509.4	378.7	21.2
	90	556	372	4930.2	7512.2	379.1	17.3
0.1r	30	(5)	3870.8	6576.2	10156.8	417.2	26.2
	40	(5)	17798.2	87973.4	108169.6	583.9	29.9
	50	(5)	8950.4	89062.4	70182	507.6	29.7
	60	(5)	22457.8	95993.8	53885.8	528.7	30.9
	70	(5)	28358.4	176635.4	83490.6	448.8	32.4
	80	(5)	(1)45951.8	(1)163500.3	(1)242734	(1)497.9	(1)33.9
	90	(5)	(2)101616.7	(3)170902.5	(2)327946.3	(3)474.8	(2)37.5
0	30	(5)	2205	5613.4	9114.8	462	27.2
	40	(5)	3821.6	13803.4	29569.6	488.2	28.1
	50	(5)	6037.8	30095.8	47806.4	511.4	28.2
	60	(5)	5144.6	127214	65936	546.7	27.9
	70	(5)	5379	45454	50358.6	477.9	28
	80	(5)	2667	18581.6	11890.6	406.7	28.1
	90	(5)	7289.4	52156.2	64416.2	424.5	29.5

Table 3: Average branch-and-bound nodes and cut generation time

is calculated as $(z^* - z^{LP})/z^*$, where z^* is the instance's optimal value and z^{LP} is the objective value obtained after running the initial cutting plane loop on the relaxed master problem. The reported value is the arithmetic mean over the five instances for each combination of interdicted traversal probabilities q and budget level b. DEF, C-DEF, and BEN all have the same LP relaxation value, so we show this gap only for BEN. The columns under the heading "LP relaxation after CPLEX cuts" refer to the gap that is obtained at the root node, after CPLEX finishes its process of adding general-purpose cuts. We exclude DEF from these results since in many instances the root node has not completed processing in the time limit. These results show that the initial LP relaxation value obtained using the cuts in the PATH method is very similar to that obtained using the BEN (or the DEF or C-DEF formulations). We also find that the general purpose cuts in CPLEX significantly improve the relaxation value for all formulations, with the greatest improvement coming in the C-DEF formulation. The gaps obtained after the addition of CPLEX cuts are quite similar between PATH and BEN, with the exception of the $q = 0$ instances, where the CPLEX cuts seem to be more effective for the BEN formulation.

Table 3 displays the average number of branch-and-bound nodes of each algorithm and the time spent generating cuts in BEN and PATH. This includes time spent generating cuts during the initial cutting plane loop as well as within the branch-and-cut process. The mean is over instances that solved within the time limit. Consistent with the results on root relaxation gaps, we find that C-DEF requires the fewest number of branch-and-bound nodes, and that PATH and BEN methods require a similar number of branch-and-bound nodes. On the other hand, significantly less time is spent generating cuts in the PATH method than in the BEN method, which explains the better performance of PATH. In particular, generating cuts in PATH requires solving a maximum-reliability path problem for each destination node, as opposed to the Benders approach which requires solving a linear program for each scenario.

6 Conclusion

We have proposed two new methods for solving maximum-reliability SNIP. The first method is to solve a more compact DEF, and the second is based on a reformulation derived from considering the reliability for each path separately. Both of these methods outperform state-of-the-art methods on a class of SNIP instances from the literature. An interesting direction for future research is to investigate whether ideas similar to the path-based formulation might be useful for solving different variants of SNIP, such as the maximum-flow network interdiction problem.

References

[1] Shabbir Ahmed and Alper Atamtürk. Maximizing a class of submodular utility functions. *Math. Program.*, 128(1-2):149–169, 2011.

[2] Ravindra K. Ahuja, Thomas L. Magnanti, and James B. Orlin. *Network flows: theory, algorithms, and applications.* Prentice Hall, Upper Saddle River, NJ, USA, 1993.

[3] Faiz A. Al-Khayyal and James E. Falk. Jointly constrained biconvex programming. *Math. of Oper. Res.*, 8(2):273–286, 1983.

[4] Merve Bodur, Sanjeeb Dash, Oktay Günlük, and James Luedtke. Strengthened Benders cuts for stochastic integer programs with continuous recourse. *INFORMS J. on Comput.*, 29(1):77–91, 2016.

[5] Gerald Brown, Matthew Carlyle, Javier Salmerón, and Kevin Wood. Defending critical infrastructure. *Interfaces*, 36(6):530–544, 2006.

[6] Gerald G. Brown, W. Matthew Carlyle, Robert C. Harney, Eric M. Skroch, and R. Kevin Wood. Interdicting a nuclear-weapons project. *Oper. Res.*, 57(4):866–877, 2009.

[7] Kelly J. Cormican, David P. Morton, and R. Kevin Wood. Stochastic network interdiction. *Oper. Res.*, 46(2):184–197, 1998.

[8] R. Fortet. Applications de l'algèbre de Boole en recherche opérationelle. *Rev. Fr. d'Inform. et de Rech. Opér.*, 4(14):17–25, 1960.

[9] Delbert R. Fulkerson and Gary C. Harding. Maximizing the minimum source-sink path subject to a budget constraint. *Math. Program.*, 13(1):116–118, 1977.

[10] Bruce Golden. A problem in network interdiction. *Nav. Res. Logist. Q.*, 25(4):711–713, 1978.

[11] Oktay Günlük and Yves Pochet. Mixing mixed-integer inequalities. *Math. Program.*, 90(3):429–457, 2001.

[12] Raymond Hemmecke, Rüdiger Schultz, and David L. Woodruff. Interdicting stochastic networks with binary interdiction effort. In David L. Woodruff, editor, *Netw. Interdiction and Stoch. Integer Program.*, pages 69–84. Springer US, 2003.

[13] Udom Janjarassuk and Jeff Linderoth. Reformulation and sampling to solve a stochastic network interdiction problem. *Networks*, 52(3):120–132, 2008.

[14] Miles Lubin and Iain Dunning. Computing in operations research using Julia. *INFORMS J. on Comput.*, 27(2):238–248, 2015.

[15] Garth P. McCormick. Computability of global solutions to factorable nonconvex programs: Part I–Convex underestimating problems. *Math. Program.*, 10(1):147–175, 1976.

[16] David P. Morton. Stochastic network interdiction. In James J. Cochran, editor, *Wiley Encyclopedia of Operations Research and Management Science*. John Wiley & Sons, Inc., 2011.

[17] David P. Morton, Feng Pan, and Kevin J. Saeger. Models for nuclear smuggling interdiction. *IIE Trans.*, 39(1):3–14, 2007.

[18] George L. Nemhauser and Laurence A. Wolsey. *Integer and combinatorial optimization.* Wiley-Interscience, New York, NY, USA, 1988.

[19] George L. Nemhauser, Laurence A. Wolsey, and Marshall L. Fisher. An analysis of approximations for maximizing submodular set functions – I. *Math. Program.*, 14(1):265–294, 1978.

[20] Feng Pan, William S. Charlton, and David P. Morton. A stochastic program for interdicting smuggled nuclear material. In David L. Woodruff, editor, *Netw. Interdiction and Stoch. Integer Program.*, pages 1–19. Springer US, 2003.

[21] Feng Pan and David P. Morton. Minimizing a stochastic maximum-reliability path. *Networks*, 52(3):111–119, 2008.

[22] Alexander Schrivjer. *Combinatorial optimization: polyhedra and efficiency.* Springer, 2003.

[23] Richard Wollmer. Removing arcs from a network. *Oper. Res.*, 12(6):934–940, 1964.

[24] R. Kevin Wood. Deterministic network interdiction. *Math. and Comput. Model.*, 17(2):1–18, 1993.

[25] Jiajin Yu and Shabbir Ahmed. Maximizing a class of submodular utility functions with constraints. *Math. Program.*, 2016.