ARGONNE NATIONAL LABORATORY
9700 South Cass Avenue
Argonne, IL 60439

Optimal Response to Epidemics and Cyber Attacks in Networks

Noam Goldberg, Sven Leyffer, and Ilya Safro

Mathematics and Computer Science Division

Preprint ANL/MCS-1992-0112

January 18, 2012

This work was supported by the Office of Advanced Scientific Computing Research, Office of Science, U.S. Department of Energy, under Contract DE-AC02-06CH11357.

Contents

Optimal Response to Epidemics and Cyber Attacks in Networks

Noam Goldberg * Sven Leyffer† Ilya Safro‡

January 18, 2012

Abstract

This paper introduces novel formulations for optimally responding to epidemics and cyber attacks in networks. In our models, at a given time period, network nodes (e.g., users or computing resources) are associated with probabilities of being infected, and each network edge is associated with some probability of propagating the infection. A decision maker would like to maximize the network's utility; keeping as many nodes open as possible, while satisfying given bounds on the probabilities of nodes being infected in the next time period. The model's relation to previous deterministic optimization models and to both probabilistic and deterministic asymptotic models is explored. Initially, we formulate a nonlinear integer program with high order multilinear terms. We then propose a quadratic approximation that provides a lower bound and feasible solution to the original problem and can be easily linearized and solved by standard integer programming solvers. We also devise a novel application and extension of cover inequalities for our formulation, to speed the solution using standard solvers.

Keywords: Nonlinear integer programming, cutting planes, network optimization, probability bounds, cybersecurity, epidemiology

AMS-MSC2010: 90C10, 90C35, 90B10, 90B18, 90C90, 92D30.

Contents

1 Introduction and Background

Mathematical models of biological epidemics and their extensions to cybersecurity have been extensively investigated during past few decades; see, for example, [18, 17, 8]. In contrast to previous work that analyzes the asymptotic growth of infections, we consider a decision problem that arises

*Mathematics and Computer Science Division, Argonne National Laboratory, Argonne, IL 60439, noamgold@mcs.anl.gov.

†Mathematics and Computer Science Division, Argonne National Laboratory, Argonne, IL 60439, leyffer@mcs.anl.gov.

‡Mathematics and Computer Science Division, Argonne National Laboratory, Argonne, IL 60439, safro@mcs.anl.gov.

at a given time period in a network where a certain infection may spread. Each node is associated with the probability of being infected at a given time period, and some of the nodes may be shut down in order to maintain desired bounds on the infection probabilities of the nodes. Similar decision problems involving a response to spread of an infection or fire in a network have been introduced by [1] and [14]; as far as we are aware, however, previous work on optimal response to network infections has considered only deterministic models.

Probabilistic network optimization models have been extensively studied and applied in the literature of reliable network design [6, 23]. In network reliability one is usually interested in designing a network that can withstand the possibility that some of the terminal nodes become disconnected, with a certain probability. In contrast in our application: We would like to remove rather than to install nodes. The probability of a node becoming infected depends on the probability of contracting the virus from a neighbor. A bound on the probability applies to each network node, rather than to the entire network or to pair of terminals. Finally, the objective is to maximize rather than minimize an increasing function of the open nodes. In the following subsections we first expand on mathematical models of epidemics and their extensions to cybersecurity. Next, we elaborate on previous work on the decision problems of determining which nodes to treat in the network in order to mitigate an infection.

1.1 Models for the Spread of Computer Viruses Motivated by Epidemiology

Traditional epidemiological models use a few parameters to estimate the growth of the infected population as a whole and, in particular, to predict whether an epidemic dies out at the limit or whether it ensues. A key input parameter of the epidemic in these models is the virus *birth rate*, β, the rate at which healthy individuals become infected with a virus when coming in contact with infected individuals. Similarly, a virus *death rate*, δ, is the rate at which infected individuals are cured. Some recent models address more details such as an underlying structure for the propagation of the infection. Typically, the structure can be described by a *contact graph*: a graph $G = (V, E)$ with n nodes corresponding to individuals, or groups, of the general population and edges corresponding to possible contacts among individuals. The Kephart-White model [18] applies epidemiology-based modeling to computer viruses. Let η denote the size of the infected population and k denote the average degree of the contact graph. The model is captured by the ordinary differential equation

$$\frac{d\eta}{dt} = \beta k \eta \left(1 - \frac{\eta}{n}\right) - \delta \eta.$$

The steady state solution of this model may provide an approximate solution of the growth of an epidemic in networks where the contact among individuals is sufficiently homogeneous (which is probably not true in the case of computer networks). However, this model does not model specific individuals, nor does it suggest which individuals should be treated with limited resources.

Let $N(i)$ be the set of neighbors of node i. Let $h_{i,t}$ be the probability that a node i is not infected from one of its neighbors at time period t, and let $\pi_{i,t}$ be the probability that a node i is infected at

time t. Chakrabarti et al. [8] propose a probabilistic model in which

$$h_{i,t} = \prod_{j \in N(i)} (1 - \beta \pi_{j,t-1}), \tag{1}$$

where β is now the probability of a node to contract the virus from an infected neighbor. This model assumes independence of the probabilities $\pi_{j,t-1}$, an assumption that is justifiable in particular if the time steps are small, in which case the probability of multiple events occuring may be negligible. Further, the assumption of probability independence significantly simplifies their analysis. Chakrabarti et al. [8] analyze an *epidemic threshold* for the dynamical system associated with (1); the magnitude of this threshold compared with β/δ determines whether an infection ensues or becomes endemic. They determine a threshold value that is inversely proportional to the largest eigenvalue of the adjacency matrix of the network. Note that this result is derived under a restrictive assumption that the probability of infection spreading along every link equals the same probability β.

1.2 Deterministic Network Response Optimization Models

A well studied deterministic network response model is known as the firefighter problem; see [2] and [14] and references therein. The problem is to iteratively decide which nodes to defend (or immunize) while a fire (or infection) is spreading in the network. At each time step the network nodes are partitioned into three groups: infected, recovered, and susceptible (those that may be infected or catch on fire). The fire spreads out deterministically from each node to its neighbors. To prevent the fire from spreading, at each step only a limited number of nodes can be defended. Once a node is defended (or recovered), it cannot be infected at succeeding iterations. The objective is to contain the fire while minimizing the total number of nodes lost. The problem is known to be $\mathcal{NP}$-hard even for tree networks with a maximum node degree of 3 [11]. An alternative formulation of the problem, also known to be $\mathcal{NP}$-hard, has the objective of minimizing the cost of saving all nodes from the fire [2].

The model of the firefighter problem is inherently iterative. The model is restrictive in assuming that all unprotected neighbors of an infected node become infected in the next time period. It also assumes that the fire may not spread beyond the neighbors of an infected node within a single time period. In practice there is a decision problem to be solved at a particular moment in time. Further, the time periods in which action can be taken to respond to the epidemic may not necessarily correspond to iterations over which the epidemic may spread.

In contrast to the firefighter's problem, Altunay et al. [1] suggest an optimization problem for determining the optimal response to cyber attacks. This approach is more closely related to our problem formulation. The network is modeled by an undirected graph $G = (V, E)$ with vertex set V, consisting of n nodes and edge set E, with $|E| = m$. The set of nodes may correspond to sites, servers, or individual users, and the edges may correspond to communication links or connections. Further suppose $V_c \subseteq V$ is a set of sites known to be compromised or infected. Accordingly, $V_u = V \setminus V_c$ is the set of uncompromised or susceptible sites. The network operator would like to maximize the utility of the network resources that remain open while shutting down some of the nodes to maintain an acceptable level of threat.

Let $x \in \{0,1\}$ be a vector of decision variables; $x_i = 1$ means that node i remains open, and $x_i = 0$ corresponds to shutting it down. A network utility function measures the usability of the network as a function of the node configuration $x \in \{0,1\}^n$, for some $0 \leq W \in \mathbb{R}^{n \times n}$:

$$u(x) = \sum_{\{i,j\} \in E} W_{ij} x_i x_j.$$

Altunay et al. [1] formulate an optimization problem for determining $x \in \{0,1\}^n$ such that each node i has a level of threat (determined endogenously and) given by a decision variable $t_i \in [0, T_i]$, for some input threat upper bound

$$0 < T_i \begin{cases} \leq 1 & i \in V_u \\ = 1 & i \in V_c. \end{cases}$$

For $i \in V$, let $T_i^0 \in [0,1)$ be the level of threat of node i if it is isolated (i.e., if none of its neighbors are open), and let p_{ij} be a rate of propagation of an infection from node j to node i. For $i \in V$, let $N(i) = \{j \in V \mid \{i,j\} \in E\}$. A nonlinear model based on [1] for network response is

$$\begin{aligned}
\underset{x,t}{\text{maximize}} \quad & u(x) && && \text{(2a)} \\
\text{subject to} \quad & t_i = T_i^0 x_i + \sum_{j \in N(i)} p_{ij} t_j x_i x_j && i \in V_u && \text{(2b)} \\
& 0 \leq t_i \leq T_i && i \in V_u && \text{(2c)} \\
& t_i = T_i = 1 && i \in V_c && \text{(2d)} \\
& x \in \{0,1\}^n. && && \text{(2e)}
\end{aligned}$$

The formulation (2) relaxes a set of additional constraints that are present in the site model of Altunay et al. [1] in order to discourage shutting down uncompromised sites. Instead we may add a term $-C \sum_{i \in V_c} x_i$ in (2a); here, $C \geq 0$ is a penalty for compromised sites that remain open. For small values of C the parameter serves as a tiebreaker among different optimal solutions, favoring those that shut down more compromised sites.

2 Preliminary Analysis and Models

We now continue to investigate the model (2). Initially, we use a computational point of view. We then reinterpret the formulation within a probabilistic context. Specifically, we reinterpret the deterministic model as a conservative approximation of a formulation with probabilistic constraints, under appropriate assumptions. We then introduce a model that better approximates the probabilistic formulation, and that we are able to solve more efficiently in experiments, than we are able to solve the original problem.

2.1 Analysis of the Deterministic Model

Altunay et al. [1] formulate their problem as a mixed-integer nonlinear program (MINLP) similar to (2) and show that it can be solved as a mixed integer program (MIP) through a particular linearization scheme. This solution approach is well justified for problems that are recognized to be

hard (otherwise, more efficient and scalable solution methods may exist). We now show that their network response problem (more precisely, its relaxation (2)), is $\mathcal{NP}$-hard.

Proposition 1. *The problem* (2) *with* $W_{ij} = 1$ *for every* $\{i, j\} \in E$ *is* $\mathcal{NP}$*-hard.*

Proof. The proof follows by reduction of the maximum independent set problem. Let $\bar{G} = (\bar{V}, \bar{E})$ denote an input graph of the independent set problem where $\bar{V} = \{1, \ldots, \ell\}$. Let $V = \{1, \ldots, 2\ell\}$ and
$E = \bar{E} \cup \{\{i, \ell + i\} \mid i = 1, \ldots, \ell\}$. Let $T_i^0 = T_i > 0$ for all $i \in V$, let

$$p_{ij} = \begin{cases} 0 & \{i, j\} \in E \setminus \bar{E}, \\ \epsilon > 0 & \text{otherwise,} \end{cases}$$

and let $W_{ij} = 1$ for all $\{i, j\} \in E$. Evidently, since $T_i^0 = T_i > 0$ and $p_{ij} > 0$ for all $\{i, j\} \in \bar{E}$, every solution (x, t) that is feasible for (2) satisfies $x_i x_j = 0$ for all $\{i, j\} \in \bar{E}$ and thus corresponds to an independent set in the graph $\bar{G}$. Conversely, given an independent set in $S \subseteq \bar{V}$, for any $x \in \{0, 1\}^n$ such that $x_k = 0$ for all $k \in \bar{V} \setminus S$, then $x_i x_j = 0$ for all $\{i, j\} \in \bar{E}$, and $(x, x \circ T^0)$ (where $\circ$ denotes the Hadamard product) is feasible for (2). For a solution (x^*, t^*) that is optimal to (2), it follows that

$$\sum_{\{i,j\} \in E} W_{ij} x_i^* x_j^* = \sum_{\substack{\{i,j\} \in E: \\ i \in V \setminus \bar{V}, j \in \bar{V}}} x_i^* x_j^* = \sum_{i \in \bar{V}} x_i^* x_{i+\ell}^* = \sum_{i \in \bar{V}} x_i^*. \tag{3}$$

The last equality followed from optimality; $p_{i,i+\ell} = 0$, for all $i \in \bar{V}$, implies the feasibility of $x \in \{0, 1\}^n$ defined by

$$x_j = \begin{cases} x_j^* & j \in \bar{V}, \\ 1 & j \in V \setminus \bar{V}, \end{cases}$$

and if $x_i^* = 1$ and $x_{i+\ell}^* = 0$, together with $W_{i,i+\ell} = 1 > 0$ it implies $u(x) > u(x^*)$, a contradiction. By (3) and the correspondence of feasible solutions of (2) and independent sets, it follows that the optimal solution of (2) in G yields a maximum independent set in $\bar{G}$. □

2.2 Probabilistic Network Response and Reinterpreting the Deterministic Model

We now introduce an alternative formulation for network response within a probabilistic framework. The motivation is related to the application described in [1], though our modeling assumptions with respect to the spread of the infection are similar to those of the asymptotic analysis appearing in [8]. As in [8], we assume that the probabilities of nodes being infected in the previous time period are independent of one another. Although it may be more realistic to assume that the probabilities that neighboring nodes are infected are dependent on one another, assuming independence provides a significant simplification in terms of both computation and modeling (i.e., relieving one from having to specify the joint probabilities). Let p_{ij} be the probability of node i being infected by node j. Let $\pi_{i,t}$ be the probability of node i being infected at time t. The probability of node i not being infected at time t is then

$$h_{i,t} = \prod_{j \in N(i)} (1 - p_{ij} \pi_{j,t-1}).$$

The probability of a node i being infected equals the probability of at least one of its neighbors being infected times the probability that the neighbor infects i. In the following only a single time period is considered, and accordingly we will drop the subscript t. We also note that one may experiment with the effectiveness of our models over several time periods, for example, by repeatedly applying them within a discrete event simulation.

We would like to choose a network configuration $x \in \{0,1\}^n$ that maximizes $u(x)$ while bounding the infection probability of each node $i \in V$ by the given T_i. For convenience, for each $\{i,j\} \in E$, let $q_{ij} = p_{ij}\pi_j$. The resulting formulation is

$$\begin{aligned} \underset{x}{\text{maximize}} \quad & u(x) && && \text{(4a)} \\ \text{subject to} \quad & x_i - \prod_{j \in N(i)} (1 - q_{ij}x_j) \le T_i && i \in V, && \text{(4b)} \\ & x \in \{0,1\}^n. && && \text{(4c)} \end{aligned}$$

The constraint (4b) for $i \in V$ ensures that either the node is shut down, that is $x_i = 0$, or if $x_i = 1$, then the probability of i being infected is at most T_i.

Figure 1 shows a plot of the optimal objective values $u(x^*)$ of formulations (2), and (4) as T is varied in the interval $[0,1]$ (we let T be the common value of T_i for all $i \in V$). In this experiment the input data for the components of π (which is a part of the input of (4)) was generated uniformly at random from $[0, 0.8]$, except for the component i corresponding to the single compromised node, for which $\pi_i = 1$. The input data of (2) and (4) are different; in particular, the level of threat t_i is a variable for each $i \in V$ in the first, while the corresponding π_i is an input parameter in the latter. Nevertheless, Figure 1 shows that the two models are roughly similar in capturing tradeoff of the "level of threat" and utility. This observation can be further explained by the fact that with $t = \pi$ the right-hand side of (2b) provides an upper bound for the left-hand side of (6b). We elaborate on this fact in Section 2.3 where we reinterpret (2) as a conservative formulation in a probabilistic context. The fact that the curve of (4) dominates that of (2) may be related to this observation about (2), together with the fact that the left-hand side of the constraints (2b) includes an additional positive coefficient $T_i^0 = 0.1$, as in the experiments of Altunay et al. [1].

Proposition 2. *The problem* (4)*, with* $W_{ij} = 1$ *for all* $\{i,j\} \in E$*, is* $\mathcal{NP}$*-hard.*

Proof. The proof follows by a similar reduction of the maximum independent set problem in the input graph $\bar{G} = (\bar{V}, \bar{E})$, as in the proof of Proposition 1; specifically, let G and W be defined as in that proof. Assume some $\epsilon, \delta \in (0,1)$ with $0 < \epsilon < \delta < 1$, let $T_i = \epsilon$ for $i \in \bar{V}$, and let

$$p_{ij} = \begin{cases} \delta & \{i,j\} \in \bar{E} \\ 0 & \{i,j\} \in E \setminus \bar{E} \end{cases}.$$

It follows that every solution x that is feasible for (4) has $x_i x_j = 0$, for every $\{i,j\} \in E$; otherwise, by the fact that the left-hand side of (4b) is increasing in x,

$$x_i - \prod_{k \in N(i)} (1 - \delta x_k) \ge 1 - (1 - \delta) = \delta > \epsilon = T_i,$$

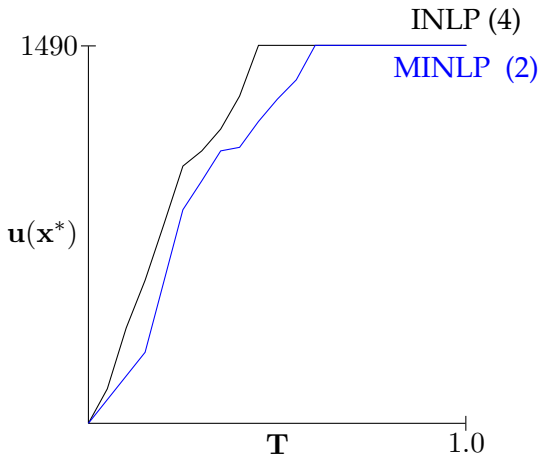


Figure 1: Plots of $u(x^*)$ vs. $T(= T_i$ for all $i \in V)$ for the MINLPs (2) and (4) with an aggregated Open Science Grid example network, OSG-1 appearing in [1].

a contradiction to the feasibility of x. Conversely, for an independent set $S \subset \bar{V}$, if $x_i = 0$ for every $i \in \bar{V} \setminus S$, then x is feasible for (4). Further, the fact that $q_{ij} = 0$ for $\{i, j\} \in E \setminus \bar{E}$ implies that (3) holds for all x^* that is optimal to (4). Therefore, an optimal solution of (4) yields a solution that is optimal to the maximum independent set problem. □

2.3 Approximating the Infection Probability Constraints and INLP

The formulation (4) is a nonconvex integer nonlinear program (INLP) and a linearized formulation of (4) may require a large number of variables. We consider an approximation of (4). In particular, the approximation we consider provides a feasible solution for (4) and thus a lower bound on its optimal solution. To this end, instead of the exact form of the constraints of (4), we bound the probability of the union of events of a node being infected from each of its neighbors. A known bound in terms of the joint probability of pairs of events is Hunter's bound [16, 25]: for a sample space Ω, given a finite set of events $\{A_i \subseteq \Omega \mid i \in N\}$,

$$P(\cup_{i \in N} A_i) \leq \sum_{i \in N} P(A_i) - \sum_{\{i,j\} \in \mathcal{T}} P(A_i \cap A_j),$$

where $\mathcal{T}$ is a maximum weighted spanning tree of the complete graph with vertex set N, having $P(A_i \cap A_j)$ as the weight of each edge $\{i, j\}$. Other upper bounds exist, most requiring the computation of third- or higher-order terms; see for example [7].

Let $\mathcal{T}_i$ be the maximum spanning tree of the complete graph with nodes $N(i)$, with each edge $\{j, k\}$ weighted by the joint probability of i being infected by both j and k. To simplify the notation, for each $\{j, k\} \in \mathcal{T}_i$, let $r_{ijk} = q_{ij} q_{ik}$. For $i \in V$, let M_i be a suitably large constant; for example, it suffices to set $M_i = \sum_{j \in N(i)} q_{ij}$. Then, for each $i \in V$, we replace the constraint of (4) by a more

conservative (tighter) constraint

$$\sum_{j\in N(i)} q_{ij}x_j - \sum_{\{j,k\}\in\mathcal{T}_i} r_{ijk}x_jx_k \le T_i - M_i(x_i-1),$$
$$\Leftrightarrow g_i(x) \equiv M_ix_i + \sum_{j\in N(i)} q_{ij}x_j - \sum_{\{j,k\}\in\mathcal{T}_i} r_{ijk}x_jx_k \le M_i + T_i.$$

The resulting formulation is

$$\underset{x}{\text{maximize}} \quad u(x) \tag{6a}$$
$$\text{subject to} \quad g_i(x) \le M_i + T_i \qquad \text{for } i \in V \tag{6b}$$
$$x \in \{0,1\}^n. \tag{6c}$$

Note that Hunter's bound applies also in the more general case that the probabilities π_{ij} are not independent for all $j \in N(i)$. In this case, one may set the coefficient r_{ijk} with the joint probability that i is infected from both j and k (and which may not equal $q_{ij}q_{ik}$).

In the following we empirically compare (4) with two formulations: the first-order (union bound) formulation, with all bilinear terms of the constraints of (4) having a zero coefficient, and formulation (6), To evaluate each, we consider the relative error, defined as

$$\frac{u(x^{(4)}) - u(x^{(6)})}{u(x^{(4)})},$$

where $x^{(\cdot)}$ is an optimal solution of the corresponding formulation $(\cdot)$. The computational experiments are run using the state-of-the art CPLEX solver, version 12.3, to compute (the linearized version of) (6) with $r = 0$ and (6) with $r_{ijk} = q_{ij}q_{ik}$ for all $i \in V$, and $\{j,k\} \in \mathcal{T}_i$. To compute (4), we use the open source state-of-the-art MINLP solver BARON, version 9.3.1.

The results of the experiments are shown in Table 1. Each row displays an average of 10 runs, each with a randomly generated subset of $V_c \subseteq V$ of the indicated cardinality having $\pi_j = 1$ for all $j \in V_c$. For each $i \in V \setminus V_c$, π_i is generated uniformly at random from $(0, 0.8)$.

Table 1 shows that the relative error of the formulation (6) with $g_i(x)$ having bilinear terms is substantially smaller than the relative error of the formulation involving only linear terms in $g_i(x)$. Further, the error does not significantly grow as $|V_c|$ increases in contrast to the formulation involving only linear terms. The running times of the exact (assuming the independence of infection probabilities) nonlinear formulation (4) are about twice as much as that of the quadratic approximation (6). However, the full product formulation (4) becomes increasingly difficult to solve: the running times of (6) with $|V| = 60$ and $|E| = 328$ are on the order of a few minutes, while the full product form (4) could not be solved to optimality within two hours. Thus, we can use the quadratic formulation (6) as a suitable surrogate for the nonconvex multilinear formulation (4), which is much harder to solve. Moreover, the quality of the quadratic approximation significantly improves on the quality of the linear approximation.

Although the formulation (6) bilinear rather than general multilinear terms, and is simpler compared with the nonlinear formulation (4), the next proposition establishes that it remains $\mathcal{NP}$-hard.

Table 1: Average error over ten runs with the aggregated Open Science Grid example network, OSG-1 appearing in [1]; each run corresponds to a random subset V_c of the indicated cardinality, and which contains all nodes i with $\pi_i = 1$. The results of the linear (union bound) and quadratic (Hunter bound) conservative approximations are shown in the second and third set of columns, respectively.

$\lvert\mathbf{V_c}\rvert$	(6) **with** $r = \mathbf{0}$				**Runs of** (6)			
	Avg Err	Max Err	Avg Sec	Max Sec	Avg Err	Max Err	Avg Sec	Max Sec
1	0.10	0.15	0.21	0.31	0.07	0.13	0.37	0.43
2	0.08	0.14	0.21	0.27	0.02	0.07	0.45	0.59
3	0.06	0.14	0.20	0.28	0.01	0.06	0.51	0.62
4	0.09	0.16	0.22	0.29	0.02	0.06	0.52	0.81
5	0.12	0.16	0.28	0.49	0.01	0.05	0.58	0.69
6	0.12	0.17	0.26	0.35	0.03	0.07	0.77	2.14
7	0.13	0.20	0.26	0.41	0.02	0.06	0.86	2.38
8	0.14	0.17	0.28	0.50	0.01	0.09	0.65	0.97
9	0.15	0.19	0.38	0.64	0.01	0.05	0.85	1.43
10	0.15	0.21	0.33	0.50	0.02	0.10	0.92	1.75

Proposition 3. *The problem* (6) *with* $W_{ij} = 1$ *for every* $\{i, j\} \in E$ *is* $\mathcal{NP}$*-hard.*

Proof. The proof follows by a similar reduction of a maximum independent set problem in the input graph $\bar{G}$ as in the proof of Proposition 2. Let $G = (V, E)$, W, q, and T be as defined as in that proof, but choosing $\delta, \epsilon > 0$ so that $\delta = 1/n > \epsilon$. It follows that for all $x \in \{0, 1\}^n$ with $x_j = 0$, for some $j \in V$,

$$g(x + e_j) - g(x) = q_{ij} \left(1 - \sum_{\substack{k \in N(i): \\ \{j,k\} \in \mathcal{T}_i}} q_{ik} x_k \right) \geq \frac{1}{n} - N(i) \left(\frac{1}{n^2} \right) \geq \frac{1}{n} - \frac{n-1}{n^2} > 0.$$

Now, assume x' is feasible for (4). Then, $x'_i x'_j = 1$ for some $\{i, j\} \in \bar{E}$ implies that

$$g_i(x') \geq M_i + q_{ij} = M_i + \delta > \epsilon + M_i = T_i + M_i,$$

a contradiction. Therefore $x'_i x'_j = 0$ for all $\{i, j\} \in \bar{E}$. So, applying the converse argument as in the proof of Proposition 2, it follows that a solution x' is feasible for (6) if an only if $\left\{ i \in \bar{V} \mid x'_i = 1 \right\}$ is an independent set. Then, it follows from (3) that each solution x^* that is optimal for (6) yields a maximum independent set in $\bar{G}$. □

3 Computational Techniques

In this section we consider computational techniques to speed the computation of the formulations (2) and (6). We first consider valid inequalities for each problem that tighten the continuous relaxation upper bound. Next, we devise a heuristic for determining a feasible solution, that is, a global lower bound.

3.1 Valid Inequalities

A cover inequality for $S \subseteq V$ is a linear inequality of the form

$$\sum_{j \in S} x_i \leq |S| - 1. \tag{7}$$

Inequalities of this form were first developed as valid inequalities for linear knapsack problems; see [4, 5]. In particular, (7) must hold if S is the support of $x \in \{0,1\}^n$ for which a corresponding knapsack constraint is violated. Such inequalities and their extensions have been used for general MIPs, and have also been specialized for specific applications; see for example [19] for a recent application to a network optimization problem. Further, cover inequalities have recently been extended for conic-quadratic nonlinear formulations [3]. However, we note that these recent extensions, as well as standard applications to knapsack constraints, all require that the right-hand side of the associated constraint is monotone in x.

In the case of formulation (6), note that for every $i \in V$, $g_i(x)$ has negative quadratic coefficients. For every $i \in V$ and $j \in N(i)$, although

$$\frac{\partial g_i}{\partial x_j} = q_{ij} \left(1 - \sum_{\substack{k \in N(i): \\ \{k,j\} \in \mathcal{T}_i}} q_{ik} x_k \right)$$

tends to be positive, it may be negative for if $\sum_{\substack{k \in N(i): \\ \{k,j\} \in \mathcal{T}_i}} q_{ik} x_k > 1$ (as $q_{ij} \leq 1$ for all $\{i,j\} \in E$). Hence, in general $g_i(x)$ is not monotone in x and consequently it is not straightforward to apply standard cover inequalities to formulation (6).

We note, however, that the negative coefficients of the quadratic terms tend to be small in our stochastic application, and since $\mathcal{T}_i$ is a tree, the degree of $j \in \mathcal{T}_i$, and corresponding number of quadratic terms of g_i, also tend to be small with respect to $|N(i)|$. Therefore, in the following lemma we define a weaker condition than monotonicity that is useful for determining valid cover inequalities.

Lemma 4. *Suppose $x, y \in \{0,1\}^n$ satisfy $\mathcal{S}(x) \equiv \{ i \in V \mid x_i = 1 \} \subset \mathcal{S}(y)$. Then*

$$g_i(y) \geq g_i(x) + \sum_{j \in \mathcal{S}(y) \setminus \mathcal{S}(x)} [g_i(x + e_j) - g_i(x) - p_{ij} \pi_j].$$

Proof. Letting x and y be defined as in the claim of the lemma, we have that

$$g_i(y) = g_i(x) - \sum_{\substack{\{j,k\}\in\mathcal{T}_i:\\ j\in\mathcal{S}(y)\setminus\mathcal{S}(x),k\in\mathcal{S}(x)}} q_{ij}q_{ik} - \sum_{\substack{\{j,k\}\in\mathcal{T}_i:\\ j\in\mathcal{S}(y)\setminus\mathcal{S}(x),k\in\mathcal{S}(y)\setminus\mathcal{S}(x)}} q_{ij}q_{ik} + \sum_{j\in\mathcal{S}(y)\setminus\mathcal{S}(x)} q_{ij}. \tag{8}$$

Note that since $\mathcal{T}_i$ is a tree, there are at most $|\mathcal{S}(y)\setminus\mathcal{S}(x)|-1$ quadratic terms of the form $q_{ij}q_{ik}$ in the right-hand side of (8); each satisfies $q_{ij}q_{ik}\le\min\{q_{ij},q_{ik}\}$. Hence,

$$\sum_{\substack{\{j,k\}\in\mathcal{T}_i:\\ j\in\mathcal{S}(y)\setminus\mathcal{S}(x),k\in\mathcal{S}(y)\setminus\mathcal{S}(x)}} q_{ij}q_{ik} \le \sum_{j\in\mathcal{S}(y)\setminus\mathcal{S}(x)} q_{ij},$$

implying that

$$g_i(y) - g_i(x) \ge -\sum_{\substack{\{j,k\}\in\mathcal{T}_i:\\ j\in\mathcal{S}(y)\setminus\mathcal{S}(x),k\in\mathcal{S}(x)}} q_{ij}q_{ik} = \sum_{j\in\mathcal{S}(y)\setminus\mathcal{S}(x)} [g_i(x+e_j) - g_i(x) - q_{ij}]. \qquad \square$$

Lemma 4 identifies a lower bound on $g_i(y)$ for $y\in\{0,1\}^n$ in terms of a given $x\le y$. In the following proposition, we use this bound to determine a sufficient condition for a given set $S\subseteq V$ to define a valid inequality (7). Further, this sufficient condition is testable; it is straightforward to determine whether the condition holds in polynomial (in fact linear) time.

Proposition 5. *Suppose $x'\in\{0,1\}^n$ with $S=\mathcal{S}(x')$ and $i\in S$ so that* (6b) *is violated. If*

$$g_i(x'+e_j) - g_i(x') - q_{ij} > -(g_i(x') - T_i - M_i)/(n-|S|) \qquad \text{for all } j\in V\setminus S, \tag{9}$$

then (7) *is a valid inequality for* (6).

Proof. Assume the hypothesized conditions of the proposition hold, and that $y\in\{0,1\}^n$ is an optimal solution of (2) satisfying $\bar{S}=\mathcal{S}(y)\supseteq S$. By Lemma 4 and (9) it follows that

$$\begin{aligned} g_i(y) &\ge g_i(x') + \sum_{j\in\bar{S}\setminus S} [g_i(x'+e_j) - g_i(x') - q_{ij}] > g_i(x') - (|\bar{S}| - |S|)(g_i(x') - T_i - M_i)/(n-|S|) \\ &= g_i(x')\frac{n-|\bar{S}|}{n-|S|} + (T_i+M_i)\frac{|\bar{S}|-|S|}{n-|S|} > T_i + M_i. \end{aligned}$$ $\square$

For knapsack constraints, extended covers provide stronger inequalities that can also be generated and separated efficiently in practice [4, 13, 12]. We now consider extended cover inequalities for (6): we say that $X_{S,i}\subseteq V$ is an extension of a covering set $S\subseteq V$, for some $i\in V$, if

$$q_{ij} - \sum_{k\in X_{S,i}\cup S} r_{ijk} > \max_{\ell\in S} q_{i\ell}, \qquad \text{for all } j\in X_{S,i}\,. \tag{10}$$

We refer to $E_{S,i}\equiv S\cup X_{S,i}$ as an *extended covering set*. For such a set the following proposition suggests valid inequalities that are clearly stronger than (7).

Proposition 6. *Suppose $x'\in\{0,1\}^n$ with $S=\mathcal{S}(x')$ and some $i\in S$ so that* (6b) *is violated and* (9) *is satisfied. If $X_{S,i}\subseteq N(i)\setminus S$ satisfies* (10)*, then*

$$(E_{S,i}\setminus S+1)x_i + \sum_{j\in E_{S,i}\setminus\{i\}} x_j \le |E_{S,i}| - 1 \tag{11}$$

is valid for (6).

Proof. Suppose $\widehat{x} \in \{0,1\}^n$ that is feasible for (6). First, considering the case that $\widehat{x}_i = 0$, then

$$\sum_{j \in E_{S,i} \setminus \{i\}} \widehat{x}_j \leq |E_{S,i}| - 1 = |E_{S,i}| - 1 - (|E_{S,i} \setminus S| + 1)\widehat{x}_i.$$

Otherwise, consider the case that $\widehat{x}_i = 1$. Assume for the sake of deriving a contradiction that

$$\sum_{j \in E_{S,i} \setminus \{i\}} \widehat{x}_j = \sum_{j \in S \setminus \{i\}} \widehat{x}_j + \sum_{j \in E_{S,i} \setminus S} \widehat{x}_j \geq |E_{S,i}| - (|E_{S,i} \setminus S| + 1)\widehat{x}_i = |S| - 1.$$

Then, by the definition of $E_{S,i}$, and (10), it follows that

$$\begin{aligned}
&\sum_{j \in S \setminus \{i\}} q_{ij}\widehat{x}_j - \sum_{\substack{\{j,k\} \in \mathcal{T}_i: \\ j,k \in S}} r_{ijk}\widehat{x}_j\widehat{x}_k + \\
&\sum_{j \in X_{S,i}} q_{ij}\widehat{x}_j - \sum_{\substack{\{j,k\} \in \mathcal{T}_i \\ j \in X_{S,i} \text{ or } k \in X_{S,i}}} r_{ijk}\widehat{x}_j\widehat{x}_k \geq \sum_{j \in S \setminus \{i\}} q_{ij}\widehat{x}_j - \sum_{\substack{\{j,k\} \in \mathcal{T}_i: \\ j,k \in S}} r_{ijk}\widehat{x}_j\widehat{x}_k + \max_{j \in S}\{q_{ij}\} \sum_{k \in E_{S,i} \setminus S} \widehat{x}_k \\
&\qquad \geq \sum_{j \in S \setminus \{i\}} q_{ij}x'_j - \sum_{\{j,k\} \in \mathcal{T}_i} r_{ijk}x'_jx'_k > T_i,
\end{aligned}$$

which contradicts the feasibility of $\widehat{x}$. □

Quadratic and higher-order INLPs can be linearized through standard techniques involving the introduction of additional auxiliary variables. For each $\{i,j\} \in E$, let $u_{ij} \in [0,1]$ be a real variable used to linearize the corresponding bilinear term x_ix_j using the following additional constraints:

$$u_{ij} \leq x_i \qquad u_{ij} \leq x_j \qquad x_i + x_j - 1 \leq u_{ij} \qquad u_{ij} \geq 0. \tag{12}$$

Let $\bar{E} = \bigcup_{i \in V} \mathcal{T}_i \cup E$. We refer to the *linearization* of formulation (6) as (6) with each quadratic term x_ix_j for $\{i,j\} \in \bar{E}$ replaced by u_{ij}, and with the additional constraints (12). Also, we refer to the linear relaxation of (6) as the linearization of (6) with the integrality constraints (6c) replaced by $x \in [0,1]^n$. This is the approach that we follow in Section 4 in order to solve formulation (6) using a standard MIP solver. Further, for the linear relaxation it may be useful to derive cover inequalities in terms of the linearization variables.

Proposition 7. *Suppose that for some $S \subseteq V$,* (7) *is valid for* (6)*, and let $\mathcal{M} \subseteq \big\{\{i,j\} \in \bar{E} \mid i,j \in S\big\}$. If $\{i \in S \mid \forall k : \{i,k\} \notin \mathcal{M}\} = \emptyset$, then*

$$\sum_{\{i,j\} \in \mathcal{M}} u_{ij} \leq |\mathcal{M}| - \min_{k \in S} |\{j \in S \mid \{j,k\} \in \mathcal{M}\}| \tag{13}$$

is a valid inequality for the linearization of (6)*.*

Particular cases of (13) in the literature (see [15] and references therein) are matching-cover inequalities with $\mathcal{M}$ corresponding to maximum matchings (and with $\min_{i \in V} |\{j \in S \mid \{i,j\} \in \mathcal{M}\}| = 1$); these are considered in the context of quadratic knapsack

semi-definite relaxations [15]. Also, cycle inequalities are obtained by choosing a subset of variables u_{ij} corresponding to a Hamiltonian cycle of the subgraph of $(V, \bar{E})$ induced by S, and replacing the second term in the right-hand side of (13) with 2 [10, 15].

Note that in the case that $\mathcal{M}$ is a (perfect) matching (i.e., $\mathcal{M}$ contains a single edge $\{i, j\}$ for each $i \in S$), then for all (x, u) that are optimal to the linear relaxation of (6),

$$\sum_{\{i,j\}\in\mathcal{M}} u_{ij} = \sum_{\{i,j\}\in\mathcal{M}} \min\{x_i, x_j\} \leq |\mathcal{M}| - 1.$$

Now rewriting (7):

$$\frac{1}{2}\sum_{i\in S} x_i \leq (|S| - 1)/2 = |\mathcal{M}| - 1/2,$$

the right-hand side may suggest that it is weaker, however, as $(x_i + x_j)/2 \geq \min\{x_i, x_j\}$ for each $\{i, j\} \in \mathcal{M}$, it follows that the latter, and (7), are not necessarily weaker than the former inequality, and (13), respectively. Therefore, in Section 4 we generate the inequalities (11), the extended inequalities corresponding to (7), in addition to extended inequalities based on (13).

The following corollary suggests the extended inequalities in terms of the linearization variables; the validity of these inequalities for the linearization of (6) follows from Propositions 7 and 6.

Corollary 8. *Suppose $x' \in \{0, 1\}^n$, and for some $i \in S = \mathcal{S}(x')$,* (6b) *is violated and* (9) *is satisfied. Let $\mathcal{M} \subseteq \big\{\{j, k\} \in \bar{E} \mid i, j \in S\big\}$, and $U = \{j \in S \setminus \{i\} \mid \forall k : \{j, k\} \notin \mathcal{M}\}$. If $X_{S,i}$ satisfies* (10) *then*

$$(|X_{S,i}| + 1)\, x_i + \sum_{j\in U\cup X_{S,i}} x_j + \sum_{\{j,k\}\in\mathcal{M}} u_{jk} \leq |\mathcal{M}| + |U| + |X_{S,i}|\,. \tag{14}$$

is valid for the linearization of (6)*. Further, if $U = \emptyset$ and $\mathcal{M}$ is a cycle then*

$$(|X_{S,i}| + 2)\, x_i + \sum_{j\in X_{S,i}} x_j + \sum_{\{j,k\}\in\mathcal{M}} u_{jk} \leq |\mathcal{M}| + |X_{S,i}|\,. \tag{15}$$

is valid for the linearization of (6)*.*

To apply Proposition 6 and Corollary 8, we first need to determine a subset S that violates (6b) for some $i \in V$. Note that, for a given a candidate $S \subseteq V$, it is straightforward to verify condition (9). Then, using S that satisfies (9) we may extended it to a set $E_{S,i} \supseteq S$.

We now motivate a greedy heuristic for determining $S \subseteq V$. If one presupposes that this condition is satisfied, then the separation problem of determining S, for a given row corresponding to some $i \in V$, and relaxation solution x^*, can be simplified and formulated as a quadratic knapsack problem:

$$\underset{z}{\text{maximize}} \quad \sum_{j\in N(i)} g_i(z) \tag{16a}$$

$$\text{subject to} \quad \sum_{j\in N(i)} (1 - x^*_j) z_j \leq 1 - \epsilon \tag{16b}$$

$$z \in \{0, 1\}^n, \tag{16c}$$

where $\epsilon \in (0, \min_{j\in N(i)} x^*_j)$. We note that the function $g_i : \{0,1\}^n \to \mathbb{R}$, the objective (16a), is a submodular quadratic function of binary variables. This observation motivates a greedy heuristic for the separation problem. For the special case with monotone $g_i(x)$ (for example, if $q_{ij} = p \leq \frac{1}{|N(i)|}$ for all $j \in N(i)$, as in the proof of Proposition 3), and equal coefficients $(1 - x^*_j)$ for all $j \in N(i)$, a greedy algorithm is known to have an $(e-1)/e$ approximation factor guarantee (this result applies more generally to a monotone submodular objective function, not necessarily quadratic) [22]. For a nonmonotone submodular objective function, and with general knapsack constraints, there are more computationally intensive extensions that yield constant factor approximations; see, respectively, [24] and [20]. However, due to the computational advantage and ease of implementation, here we only apply the simpler greedy algorithm.

Let $x(S) \in \{0,1\}^{|V|}$ be the characteristic vector of a set $S \subseteq V$. For a given function $\phi : 2^V \times V \to \mathbb{R}$, and $i \in V$, we consider Algorithm 1 as a greedy procedure for computing an extended covering set. Such a set can then used to generate valid inequalities for the linearization of (6). Algorithm 1 is run for $i \in V$ whose corresponding constraint (6b) is tight; in Section 4 we invoke this algorithm for $i \in V$ corresponding to the largest predetermined number of dual multiplier values. We

Algorithm 1 An algorithm for computing an extended covering set $E_{S,i} \subseteq V$.

1: **Input:** an optimal solution x^* of the relaxation of (6), $i \in V$
2: $Q_1 \leftarrow \{i\}$
3: **for** $k = 1, \ldots, n$ **do**
4: **if** (6b) is violated with x replaced by $x(Q_k)$, and (9) holds **then**
5: break
6: **end if**
7: Pick some $j^* \in \operatorname{argmax}_{i\in N(i)} \phi(Q_k, j)$
8: $Q_{k+1} \leftarrow Q_k \cup \{j^*\}$
9: **end for**
10: $S \leftarrow Q_k$
11: $X_{S,i} = \emptyset$
12: **while** $\Phi \equiv \operatorname{argmax}_{j\in N(i)\setminus(X_{S,i}\cup S)} \{g_i(x(S \cup \{j\}) \mid (10)\} \neq \emptyset$ **do**
13: Pick some $j^* \in \Phi$
14: $X_{S,i} \leftarrow X_{S,i} \cup \{j^*\}$
15: **end while**
16: **Output:** $E_{S,i} = S \cup X_{S,i}$

consider two different variants of Algorithm 1, each corresponding to a different definition of the function ϕ in line 7 of the algorithm. In particular, we consider either

$$\phi(\cdot, j) = x^*_j, \qquad \text{or} \tag{17a}$$

$$\phi(S, j) = \frac{g_i(x(S \cup \{j\})) - g_i(x(S))}{1 - x^*_i}. \tag{17b}$$

In the context of the knapsack problem, extended cover inequalities are effectively generated in practice by greedily constructing S to consist of $j \in V$ for which x^*_j is largest (where x^* is an

optimal solution of the relaxation); see for example [13]. This approach is similar in to our choice of ϕ defined by (17a) in Algorithm 1. Given the cover S that is computed by Algorithm 1 in steps 2 – 10, it is then iteratively extended; starting with $E_{S,i} = S$, at each iteration we insert into $E_{S,i}$ an element $j^* \in N(i)$ that maximizes $g_i(x(E_{S,i} \cup \{j\}))$ over all $j \in N(i) \setminus E_{S,i}$ satisfying (10). Note that upon termination of Algorithm 1, assuming that the continuous relaxation solution x^* is noninteger, we are guaranteed to get a valid cover inequality; however, the resulting cover inequality does not necessarily cut off x^*. Thus, upon termination we may need to verify that the resulting cover inequalities are violated before appending them to (6).

3.2 Feasible Solution Heuristic

We now develop a specialized heuristic to be invoked as a part of the branch-and-bound algorithm in order to determine a feasible solution (i.e., a global lower bound). Algorithm 2 applies a knapsack heuristic for iteratively rounding the linear relaxation solution, and repeatedly resolving and rounding. For a constraint (6b), for each $i \in V$, the cost coefficient for each $j \in N(i)$ is approximated (in fact bounded from above) using the linear coefficient q_{ij}. For with a tight constraint: setting $x_j = 1$ for $j \in N(i)$ in a decreasing (approximate) objective-to-cost ratio.

Unless x^* is integer, at each iteration of the main loop we shut down at least one node; specifically, we fix $x_\ell = 0$ for some $\ell \in V \setminus (Z \cup O)$ in either step 16, or step 23 of the algorithm. Further, at either step of the algorithm, element ℓ is inserted into Z. Hence, the algorithm is guaranteed to terminate in at most $|V \setminus (Z \cup O)| \leq n$ iterations; further, in the case that $O = \emptyset$ (that is if none of the nodes are required to remain open), as $x = \mathbf{0}$ is feasible, it implies that a feasible solution is guaranteed to be found.

4 Computational Results

In this section we show computational results for the linearization of (6) using the methods described in Section 3. We experiment with networks that are generated randomly by using the Erdös-Rényi random graph model, with different graph density values: $0.1, 0.2$, and 0.25. Examples of the graphs generated with 50 nodes are shown in Figure 2. For each network we generated 10 different instances with different random input parameter values for W, π, p, and T. A set of compromised nodes $V_c \subseteq V$ is chosen at random so that $|V_c| = \lceil 0.1\,|V| \rceil$, for which $\pi_i = 1$ for each $i \in V_c$. Otherwise, for $i \in V \setminus V_c$, π_i is generated uniformly at random from $(0, 0.8)$. For each $\{i, j\} \in E$, W_{ij} is generated uniformly at random from $(0, 1)$. For each $\{i, j\} \in E$, the probability of the infection propagating from node j to i is given by

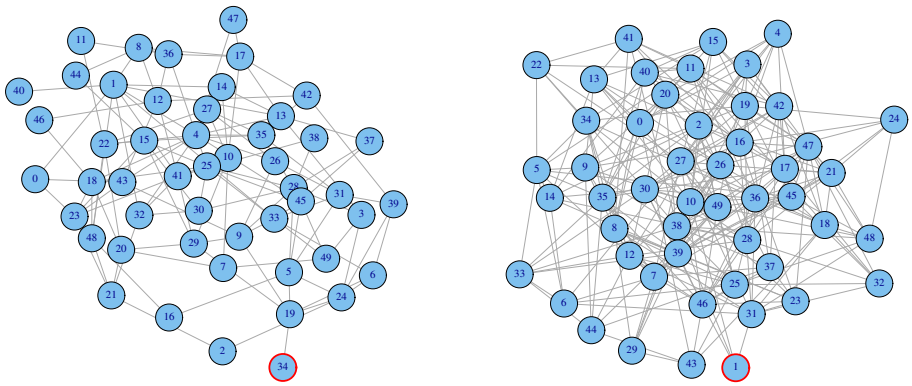
$$p_{ij} = \frac{W_{ij}}{\sum_{k \in N(j):(k,j) \in E} W_{kj}},$$

as in the experiments of Altunay et al. [1].

We run the linearized formulation of (6) using the open source CBC [21] solver (trunk version, revision #1759). Also, our cut generation routines and heuristics use the open source network algorithm LEMON [9] library (trunk version, revision #952).

Algorithm 2 LP-rounding based heuristic for determining a feasible solution.

Input: $G=(V,E), W, T, p, q$, and $Z, O \subseteq V$
while true **do**
  Solve the relaxation of (6), with $x_i = 0$ fixed for $i \in Z$ and $x_i = 1$ fixed for $i \in O$, obtaining primal vector x^* and dual multiplier vector λ^*.
  if $x^* \in \{0,1\}^n$ **then**
    break.
  end if
  $x \leftarrow \lceil x^* \rceil$
  Sort $i_1, \ldots, i_n$ so that $\lambda_{i_1} \geq \cdots \geq \lambda_{i_n}$ and let $p = \max\{k = 1, \ldots, n \mid \lambda_{i_k} > 0\}$.
  for $\ell = i_1, \ldots, i_p$ **do**
    Let $N_F(\ell) = N(\ell) \setminus (Z \cup O)$
    Sort $j_1, \ldots, j_{|N_F(\ell)|} \in N_F(\ell)$ in decreasing order of $\frac{\sum_{i \in N(j_k)} W_{ij_k} x_i x_{j_k}}{q_{\ell j_k}}$ for $k = 1, \ldots, |N_F(\ell)|$.
    $\Delta \leftarrow \sum_{j \in N(\ell) \cap O} q_{\ell j}$
    for $k = 1, \ldots, |N_F(\ell)|$ **do**
      $\Delta \leftarrow \Delta + \frac{\sum_{i \in N(j_k)} W_{ij_k} x_i x_{j_k}}{q_{\ell j_k}}$
      if (6b) is violated **then**
        $x_{j_k} \leftarrow 0$, $Z \leftarrow Z \cup \{k\}$
      else
        break
      end if
    end for
    if $\ell \notin O$ and $\Delta > \sum_{j \in N_F(\ell)} W_{\ell j}$ **then**
      $x_i \leftarrow 1$ for all $i \in N_F(\ell)$, $Z \leftarrow Z \setminus N_F(\ell)$
      $x_\ell \leftarrow 0, Z \leftarrow Z \cup \{\ell\}$
    end if
  end for
end while
Output: x



(a) An example network with 0.1 density.

(b) An example network with 0.2 density.

Figure 2: Randomly generated networks with 50 nodes that are used in the computational study of Section 4.

Although the CBC solver and our implementation support running branch-and-bound in parallel using multithreading, in the following we show experimental results only for serial runs. The experiments are run on a machine with Intel Xeon 2.66 GHz CPUs with a cache size of 6144 KB, and a 64-bit Ubuntu 10.04 Linux operating system.

In the following tables when "-" is indicated for all features, this implies the default configuration of the CBC solver[1]. In all cases the heuristic, cuts, and cover methods indicated in the tables are run in addition to the CBC defaults. In addition, when applying our cuts we set CBC to emphasize generating knapsack cover inequalities. Otherwise, the generation of our cuts at the root node seemed to be causing CBC to disable the generation of standard knapsack cover inequalities later, which, in some cases, seemed to degrade the overall performance. We set the time limit in all runs to two hours. "LIMIT" is used to indicate that the data is not displayed due to the time limit being reached. When the time limit is reached only for some of the runs, then the node and time averages that are displayed pertain to those runs in which the time limit has not been reached.

4.1 Testing the Cover and Cut Methods

Tables 2–4 show our experimental results for different methods for computing the cover: running Algorithm 1 with ϕ defined either by (17a) or (17b), as indicated. We initialized CBC to invoke our cover and cut generation routine once in every 50 nodes, with automatic adjustment of the CBC solver based on the effectiveness of the cuts that are generated. The algorithm is invoked for $i \in V$ whose constraint (6b) is among the 60 with largest dual multiplier values at the root node, and among the two largest dual multiplier values at other branch-and-bound nodes. Also, we compare the performance with the extended cover inequalities in the original variables (11), vs. generating also the inequalities using the linearization variables (14), and (15); cuts of the form (14)

[1]Due to implementation issues with our cut and heuristic routines, along with the libraries being used, in all cases we had CBC preprocessing switched off. Switching the preprocessing off did not have any significant impact on the running times of CBC with our test instances.

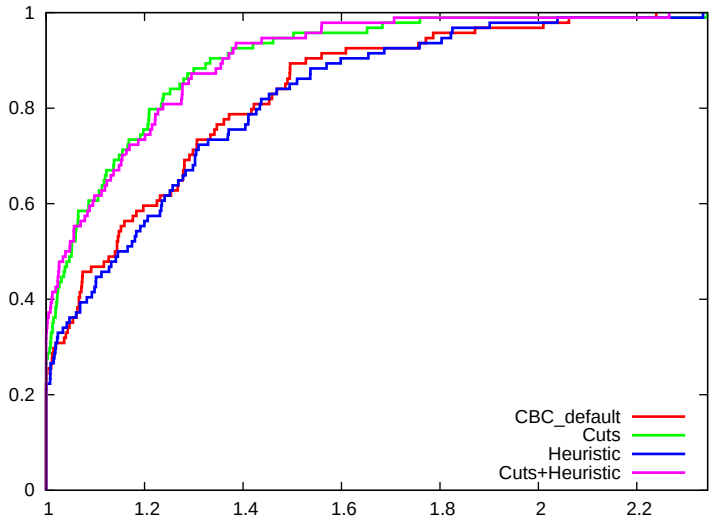


Figure 3: Performance profile showing the proportion of runs within a given factor of the best solution. The cuts are generated using the method that performed best in the experiments of Section 4.1.

are generated with $\mathcal{M}$ corresponding to maximum weighted matchings (with edges weighted by the relaxation solution values, u_{jk} for each $\{j,k\} \in \mathcal{T}_i$). The cuts of the form (15) are generated only in the case that $|S| = 4$ and $|\mathcal{M}| = 3$; evidently, in this case, these cuts are stronger than (14), and they are also trivial to generate given $|S|$.

Tables 2 – 4 show that in nearly all cases it was beneficial in terms of overall CPU time and branch-and-bound nodes to compute covers using Algorithm 1 with ϕ defined by (17b), rather than by (17a). In most cases the valid inequalities (11) combined with (14) and (15), whose covering set is computed using (17b), outperformed the state-of-the-art open source CBC solver with its default settings[1]. In both Table 2 and 4 this was also evident in fewer runs that exceeded the time limit (of two hours).

4.2 Testing the Heuristic Method

Tables 5–7 show the experimental results testing the effectiveness of the heuristic Algorithm 2. The heuristic is run with the default CBC configuration or together with the cuts that seemed to perform best in the experiments of the previous subsection: (11), (14), and (15) using Algorithm 1 with ϕ defined by (17b). Figure 3 also shows the results of these experiments as a performance profile in order to provide further insight. In addition to the data displayed in the table, the performance profile also shows the results of the runs that use the cuts without having the heuristic enabled. The heuristic for finding a feasible solution, although effective in quickly finding a feasible solution, did not seem to improve the overall running time of the CBC solver. The reason may be that CBC heuristics were sufficiently effective in finding feasible solutions and that, in fact, the tightening of the upper bounds, rather than the global lower bound, dominated the computation running times.

Table 2: Computational results with random graphs of 0.10 density. Here - indicates CBC defaults w/o preprocessing; $(\cdot)$ indicates the use of the corresponding feature and equation number, as defined in this paper. Each row displays the statistics of 10 runs, each corresponding to the same graph but with different values for the input parameters W, π, P, and T.

Nodes	**Edges**	**Cover**	**Cuts**	**Nodes**		**CPU sec**		**# over**
				Avg	Max	Avg	Max	Limit
50	123	-	-	83.4	178	8.2	11.7	0
		(17a)	(11)	84.3	230	8.7	12.3	0
		(17a)	(11),(14),(15)	88.6	178	9.0	13.3	0
		(17b)	(11)	78.4	177	7.9	11.0	0
		(17b)	(11),(14),(15)	100.1	256	8.5	15.5	0
60	185	-	-	516.7	924	40.1	63.7	0
		(17a)	(11)	453.1	959	39.6	75.6	0
		(17a)	(11),(14),(15)	527.1	998	40.3	66.2	0
		(17b)	(11)	427.2	745	37.0	53.4	0
		(17b)	(11),(14),(15)	386.0	982	34.7	65.5	0
70	249	-	-	2736.5	6118	181.9	381.8	0
		(17a)	(11)	2535.6	4758	171.6	302.3	0
		(17a)	(11),(14),(15)	2686.4	6176	185.2	390.7	0
		(17b)	(11)	2481.9	4900	171.9	347.6	0
		(17b)	(11),(14),(15)	2441.2	4693	168.9	303.9	0
80	317	-	-	16258.4	51079	1371.0	4070.1	0
		(17a)	(11)	18728.9	61948	1622.4	4939.2	0
		(17a)	(11),(14),(15)	14741.1	39759	1286.5	3143.0	0
		(17b)	(11)	18094.9	74449	1563.8	5918.8	0
		(17b)	(11),(14),(15)	16169.0	40388	1327.3	3399.1	0
90	390	-	-	36739.8	LIMIT	3739.7	LIMIT	1
		(17a)	(11)	33396.2	LIMIT	3468.5	LIMIT	2
		(17a)	(11),(14),(15)	30160.9	LIMIT	3225.9	LIMIT	2
		(17b)	(11)	32638.2	LIMIT	3352.2	LIMIT	1
		(17b)	(11),(14),(15)	33329.0	LIMIT	3250.5	LIMIT	1

Table 3: Computational results with random graphs of 0.20 density. Here - indicates CBC defaults w/o preprocessing; $(\cdot)$ indicates the use of the corresponding feature and equation number, as defined in this paper. Each row displays the statistics of 10 runs, each corresponding to the same graph but with different values for the input parameters W, π, P and T.

Nodes	**Edges**	**Cover**	**Cuts**	**Nodes**		**CPU sec**		**# over**
				Avg	Max	Avg	Max	Limit
		-	-	2053.0	4242	124.4	235.2	0
		(17a)	(11)	2012.4	4242	126.4	238.1	0
50	256	(17a)	(11),(14),(15)	2043.3	4422	127.1	261.7	0
		(17b)	(11)	1812.3	4795	115.9	271.7	0
		(17b)	(11),(14),(15)	1845.2	3234	114.9	181.2	0
		-	-	7522.4	16386	524.5	1016.4	0
		(17a)	(11)	7809.2	13950	577.2	1007.1	0
60	328	(17a)	(11),(14),(15)	7325.1	16193	556.7	1008.6	0
		(17b)	(11)	7813.4	14136	553.6	857.0	0
		(17b)	(11),(14),(15)	6996.0	13608	507.8	888.4	0
		-	-	39569.6	LIMIT	4238.0	LIMIT	2
		(17a)	(11)	40539.6	LIMIT	4372.9	LIMIT	2
70	471	(17a)	(11),(14),(15)	38864.5	LIMIT	4404.7	LIMIT	2
		(17b)	(11)	41642.4	LIMIT	4518.5	LIMIT	1
		(17b)	(11),(14),(15)	40894.3	LIMIT	4408.0	LIMIT	1

Table 4: Computational results with random graphs of 0.25 density. Here, - indicates CBC defaults w/o preprocessing; $(\cdot)$ indicates the use of the corresponding feature and equation number as defined in this paper. Each row displays the statistics of 10 runs, each corresponding to the same graph but with different values for the input parameters W, π, P and T.

Nodes	**Edges**	**Cover**	**Cuts**	**Nodes**		**CPU sec**		**# over**
				Avg	Max	Avg	Max	limit
		-	-	4521.1	15150	274.8	733.0	0
		(17a)	(11)	3681.4	10674	245.1	670.8	0
50	256	(17a)	(11),(14),(15)	3693.7	8257	249.2	487.5	0
		(17b)	(11)	3846.0	11320	253.0	630.9	0
		(17b)	(11),(14),(15)	3684.4	9130	259.0	583.7	0
		-	-	25152.4	51683	2095.7	4199.8	0
		(17a)	(11)	22696.5	48611	2047.5	4360.3	0
60	328	(17a)	(11),(14),(15)	21724.3	45772	1918.5	4027.6	0
		(17b)	(11)	22971.5	48582	1988.7	3589.9	0
		(17b)	(11),(14),(15)	19728.4	35397	1658.8	2953.7	0

Table 5: Computational results with random graphs of 0.10 density. Here, - indicates CBC defaults without preprocessing; + indicates the use of the corresponding feature and equation number, as defined in this paper. Each row displays the statistics of 10 runs, each corresponding to the same graph but with different values for the input parameters W, π, P and T.

Nodes	**Edges**	**Heuristic**	**Cuts**	**Nodes**		**CPU sec**		# **over**
				Avg	Max	Avg	Max	limit
50	123	-	-	83.4	178	8.2	11.7	0
		+	-	89.6	236	8.3	12.2	0
		+	+	96.9	224	8.2	13.7	0
60	185	-	-	516.7	924	40.3	63.9	0
		+	-	501.6	856	39.9	64.1	0
		+	+	390.3	978	34.0	69.1	0
70	249	-	-	2736.5	6118	183.6	383.2	0
		+	-	2756.3	6118	182.7	384.1	0
		+	+	2390.1	4693	165.6	391.4	0
80	317	-	-	16258.4	51079	1371.0	4070.1	0
		+	-	15890.6	48255	1375.7	3880.2	0
		+	+	17378.7	48131	1387.3	3888.8	0
90	390	-	-	36739.8	LIMIT	3739.7	LIMIT	1
		+	-	34343.3	LIMIT	3599.6	LIMIT	1
		+	+	35766.9	LIMIT	3499.5	LIMIT	1

Table 6: Computational results with random graphs of 0.20 density. Here, - indicates CBC defaults without preprocessing; + indicates the use of the corresponding feature and equation number, as defined in this paper. Each row displays the statistics of 10 runs, each corresponding to the same graph but with different values for the input parameters W, π, P and T.

Nodes	**Edges**	**Heuristic**	**Cuts**	**Nodes**		**CPU sec**		# **over**
				Avg	Max	Avg	Max	limit
50	256	-	-	2053.0	4242	124.4	235.2	0
		+	-	2234.5	4878	128.5	258.6	0
		+	+	1895.6	4023	118.2	233.7	0
60	328	-	-	7522.4	16386	524.5	1016.4	0
		+	-	6997.7	16386	513.7	1051.0	0
		+	+	6879.8	14248	502.1	910.4	0
70	471	-	-		LIMIT	4238.0	LIMIT	2
		+	-	40858.4	LIMIT	4480.8	LIMIT	2
		+	+		LIMIT	4439.8	LIMIT	1

Table 7: Computational results with random graphs of 0.25 density. Here, - indicates CBC defaults without preprocessing; + indicates the use of the corresponding feature and equation number as defined in this paper. Each row displays the statistics of 10 runs, each corresponding to the same graph but with different values for the input parameters W, π, P and T.

Nodes	**Edges**	**Heuristic**	**Cuts**	**Nodes**		**CPU sec**		# **over**
				Avg	Max	Avg	Max	limit
50	256	-	-	4521.1	15150	274.8	733.0	0
		+	-	4257.5	12444	275.2	712.8	0
		+	+	3626.2	9088	254.5	577.7	0
60	328	-	-	25152.4	51683	2095.7	4199.8	0
		+	-	26585.8	59357	2266.4	4982.1	0
		+	+	20742.0	35490	1733.2	3095.6	0

5 Conclusion

We have suggested a reformulation of a network response optimization problem considered in [1] within a probabilistic framework. This framework, motivated by a setting considered in [8], assumes independence of the infection probabilities in the previous time period. To solve the problem, we considered a quadratic formulation that bounds the infection probabilities using Hunter's bound and thus computes a more conservative solution. The advantage of this approach is twofold. First, the quadratic formulation can be easily linearized and solved by standard MIP solvers. Second, the application of the quadratic probability bound, and thus the resulting formulation, need not require probability independence. In computational experiments, as the problem instances get larger, we find the quadratic formulation is much easier to solve compared with using a state-of-the-art MINLP solver to solve the original problem.

To improve on the standard solver solution time, we suggest a novel application of cover inequalities as cutting planes for the quadratic formulation and its linearized counterpart. We are also able to derive stronger extended cover inequalities for our problem. Our inequalities are novel compared with well studied variants of knapsack cover inequalities especially considering the fact that the latter are usually derived for (knapsack or other) constraints whose left-hand side is monotone in the vector of integer variables, whereas in our case the left-hand side of the constraints is nonmonotone in the vector of binary decision variables. We prove the validity of our inequalities upon verifying a simple, testable condition. This condition is verified on-the-fly before appending the inequality as a cutting plane within branch-and-bound. In computational experiments we show significant improvement in branch-and-bound nodes and running times when applying our cuts.

References

[1] M. Altunay, S. Leyffer, J.T. Linderoth, and Z. Xie. Optimal security response to attacks on open science grids. *Computer Networks*, 55(1):61–73, 2011.

[2] E. Anshelevich, D. Chakrabarty, A. Hate, and C. Swamy. Approximation algorithms for the firefighter problem: Computing cuts over time. *Algorithmica, to appear*, 2010.

[3] A. Atamtürk, L.F. Muller, and D. Pisinger. Separation and extension of cover inequalities for second-order conic knapsack constraints with gubs. Technical Report 6.2011, DTU Management Engineering, 2011.

[4] E. Balas. Facets of the knapsack polytope. *Mathematical Programming*, 8(1):146–164, 1975.

[5] E. Balas and E. Zemel. Facets of the knapsack polytope from minimial covers. *SIAM Journal on Applied Mathematics*, 34(1):119–148, 1978.

[6] F. Boesch, A. Satyanarayana, and C. Suffel. On residual connectedness network reliability. In F. Roberts, F. Hwang, and C. Monma, editors, *Reliability of Computer and Communication Networks*. DIMACS: Series in Discrete Mathematics and Theoretical Computer Science, 1991.

[7] J. Bukszár and A. Prékopa. Probability bounds with cherry trees. *Mathematics of Operations Research*, 26:174–192, 2001.

[8] D. Chakabarti, Y. Wang, C. Wang, J. Leskovec, and C. Faloutsos. Epidemic thresholds in real networks. *ACM Transactions on Information and System Securitys*, 10, 2008.

[9] B. Dezsö, A. Jüttner, and P. Kovács. LEMON an open source C++ graph template library. In *Workshop on Generative Technologies (WGT)*, 2010.

[10] C.E. Ferreira, A. Martin, C.D. Souza, R. Weismantel, and L. Wolsey. Formulations and valid inequalities for the node capacitated graph partitioning problem. *Mathematical Programming*, 74(3):247–267, 1996.

[11] S. Finbow, A.D. King, G. MacGillivray, and R. Rizzi. The firefighter problem for graphs of maximum degree three. *Discrete Mathematics*, 307(16):2094–2105, 2007.

[12] V. Gabrel and M. Minoux. A scheme for exact separation of extended cover inequalities and applicaiton to multidimensional knapsack problems. *Operations Research Letters*, 30:252–264, 2002.

[13] Z. Gu, G.L. Nemhauser, and M.W.P. Savelsbergh. Lifted cover inequalities for 0-1 integer programs: Computation. *INFORMS Journal on Computing*, 10(10):427–437, 1998.

[14] B. Hartnell. Firefighter! an application of domination. Presented at the 25th Manitoba Conference on Combinatorial Mathematics and Computing, 1995.

[15] C. Helmberg, F. Rendl, and R. Weismantel. A semidefinite approach to the quadratic knapsack problem. *Journal of Combinatorial Optimization*, 4(2):197–215, 2000.

[16] D. Hunter. An upper bound for the probability of a union. *Journal of Applied Probability*, 13:597–603, 1976.

[17] J.O. Kephart, G.B. Sorkin, W.C. Arnold, D.M. Chess, G. Tesauro, and S.R. White. Biologically inspired defenses against computer viruses. In *IJCAI*, pages 985–996, 1995.

[18] J.O. Kephart and S.R. White. Measuring and modeling computer virus prevalence. In *Proceedings of the 1993 IEEE Symposium on Security and Privacy*, pages 2–15, Washington, DC, USA, 1993.

[19] O. Klopfenstein and D. Nace. Cover inequalities for robust knapsack sets – application to the robust bandwidth packing problem. *Networks*, online before print, 2011.

[20] J. Lee, V.S. Mirrokni, V. Nagarajan, and M. Sviridenko. Maximizing nonmontone submodular functions under matroid and knapsack constraints. *SIAM Journal on Discrete Math*, 23:2053–2078, 2010.

[21] R. Lougee-Heimer, F. Barahona, B.L. Dietrich, J. Fasano, J. J. Forrest, R. Harder, L. Ladanyi, T. Pfender, T. Ralphs, M. Saltzman, and K. Scheinberg. The COIN-OR initiative: accelerating operations research progress through open-source software. *ORMS Today*, 28(5), 2001.

[22] G.L. Nemhauser, L.A. Wolsey, and M.L. Fisher. An analysis of approximations for maximizing submodular set functions. *Mathematical Programming*, 14:265–294, 1978.

[23] A. Prékopa, E. Boros, and K-W Lih. The use of binomial moments for bounding network reliability. In F. Roberts, F. Hwang, and C. Monma, editors, *Reliability of Computer and Communication Networks*. DIMACS: Series in Discrete Mathematics and Theoretical Computer Science, 1991.

[24] M. Sviridenko. A note on maximizing a submodular set function subject to a knapsack constraint. *Operations Research Letters*, 32:41–43, 2004.

[25] P. Veneziani. Optimality conditions for Hunter's bounds. *Discrete Mathematics*, 308:6009–6014, 2008.