Kent Academic Repository

Full text document (pdf)

Citation for published version

Scaparra, Maria Paola and Cappanera, Paola (2008) Optimal Allocation of Protective Resources in Shortest-Path Networks. Working paper. University of Kent Canterbury, Canterbury

DOI

Link to record in KAR

https://kar.kent.ac.uk/25487/

Document Version

Publisher pdf

Copyright & reuse
Content in the Kent Academic Repository is made available for research purposes. Unless otherwise stated all content is protected by copyright and in the absence of an open licence (eg Creative Commons), permissions for further reuse of content should be sought from the publisher, author or other copyright holder.

Versions of research
The version in the Kent Academic Repository may differ from the final published version.
Users are advised to check **http://kar.kent.ac.uk** for the status of the paper. **Users should always cite the published version of record.**

Enquiries
For any further enquiries regarding the licence status of this document, please contact:
researchsupport@kent.ac.uk

If you believe this document infringes copyright then please contact the KAR admin team with the take-down information provided at **http://kar.kent.ac.uk/contact.html**

KAR Kent Academic Repository

University of Kent

ISSN 1748-7595 (Online)

Kent Business School

Working Paper Series

Optimal Allocation of Protective Resources in Shortest-Path Networks

Maria Paola Scaparra
Kent Business School

Paola Cappanera
Università degli Studi di Firenze

Working Paper No.177
December 2008

University of Kent

Optimal allocation of protective resources in shortest-path networks

Paola Cappanera* Maria Paola Scaparra†

Abstract

We develop a game theoretic approach for allocating protection resources among the components of a network so as to maximize its robustness to external disruptions. Specifically, we consider shortest-path networks where disruptions may result in traffic flow delays through the affected components or in the complete loss of some elements. We develop a multi-level program which identifies the set of components to harden so as to minimize the length of the shortest path between a supply node and a demand node after a worst-case disruption of some unprotected components. We propose an implicit enumeration algorithm to solve the multi-level problem to optimality and streamline the approach by solving the lower level interdiction problem heuristically at each node of the enumeration tree. We also propose some variable fixing rules which reduce the dimension of the lower level problems. A thorough computational investigation demonstrates that the proposed solution method is able to identify optimal protection strategies for networks of significant size. We also study the sensitivity of the proposed approach to variations of the problem parameters, such as the level of offensive and defensive resources, and the distribution of the arc lengths and delays.

Keywords. Network interdiction, multi-level programming, shortest paths, resource allocation.

Introduction

Most of today's societal, economic and industrial functions heavily rely on the correct and continued operations of life-line networks such as telecommunication, electric power,

*Dipartimento di Sistemi e Informatica, Università degli Studi di Firenze, Firenze, Italy, paola.cappanera@unifi.it

†Kent Business School, University of Kent, Canterbury, UK, M.P.Scaparra@kent.ac.uk

transportation, transit, water supply, sewage disposal, and natural gas and petroleum distribution systems. The growing complexity and interdependency of these networked systems have significantly increased their vulnerability to a dangerous mix of intentional and unintentional threats (e.g., terrorist attacks, sabotage, random failures, weather, natural disasters, and so on). As a consequence, the establishment of effective protection strategies has become an imperative need to lessen the operational impact of infrastructure losses and circumscribe the dire and far-reaching consequences of possible system disruptions.

Our focus in this paper is to develop a rigorous mathematical approach for identifying the most cost-effective way of allocating scarce protection resources among the components of a network so as to maximize its robustness to malicious attacks. Although it could be argued that protection investments against rare, extreme events are difficult to justify on economic grounds, due to the low probability of their occurrence, the need of implementing sound protection strategies is justifiable in consideration of the enormous ripple effects that these events may have, not only in terms of economical losses but also in terms of human lives. Furthermore, in light of the variety of hazards to which system facilities are exposed, there are likely to be economies of scope in hazard reduction [26].

In this paper, we consider distance-based networks where travel times, transmission or shipping costs are the main concern, as in the case of emergency transport networks, computer communication networks and some distribution systems. We develop a multi-level optimization model which identifies the set of components to harden so as to minimize the length of the shortest path from a supply node to a demand node after a worst-case disruption of some unprotected elements. We assume that if a component is disrupted or attacked, its travel cost is increased, eventually rendering it unusable, unless the component is hardened, in which case the attack has no effect.

The study of quantitative approaches for improving infrastructure security and robustness has attracted the interest of several researchers and practitioners over the years. Early studies have mainly focused on the identification of the most vulnerable and valuable system components as a tool to drive decisions on where to direct funds for hardening and reinforcing a life-line. The mathematical models developed to this end are known as interdiction problems and aim at identifying the sets of assets (e.g., nodes, links, or facilities) that, if disabled or lost, have the greatest impact on a system ability to perform its intended functions. Several interdiction models have been developed for military and homeland security applications in the last few decades. The seminal interdiction model is due to Wollmer [27] who analyzes the problem of removing a specified number of arcs

from a network in order to minimize the maximum flow between an origin and a destination node. Fulkerson and Harding [10] introduce the first shortest path interdiction model where arc lengths can be partially interdicted, subject to an interdiction budget, so as to maximize the shortest paths between supply and demand points. Golden [12] investigates a least-cost partial interdiction strategy to ensure a predetermined increase in the shortest path length. Other variations of interdiction models on shortest path networks can be found in [7], [2], [20] and [17]. For a complete survey of early interdiction problems with different underlying network properties the reader is referred to [5]. More recently, stochastic variants of network flow interdiction problems have appeared in [15], [14] and [8], whereas [19] extends the study of interdiction models to multi-commodity networks.

Although interdiction models can provide valuable information about the criticality of some system components, protection strategies which solely rely on this information, for example by prioritizing the protection of the most-critical assets, will often result in sub-optimal defense plans [3]. This kind of plans, in fact, fail to take into account how the set of critical assets varies in response to the hardening of some components and, therefore, may miss some crucial interactions and synergies within the system. As an example of this shortfall, consider the simple network depicted in Figure 1. The two numbers associated with each arc represent, respectively, the arc length under normal circumstances and the delay which occurs along the arc in case of disruption. Assume that the efficiency of the system is measured in terms of distance or travel time from the supply node 1 to the destination node 7. Figure 1.a shows the source-destination shortest path when all the arcs are fully operational (i.e., no delays occur). The length of the shortest path is 45. Shortest path interdiction models, such as the one in [17], can be used to identify the most critical arcs in this network. As an example, the two arcs which, if interdicted, results in the greatest increase of the shortest path length are arcs (2,5) and (3,6) (see the arcs with a cross in Figure 1.b). In this case, the post-interdiction shortest path in the network, 1-2-4-7, has a length of 67, an increases of almost 50%. Now, assume that resources are available to harden two network components. If the defensive resources are employed to protect the two critical links (2,5) and (3,6), an intelligent attacker aiming at inflicting maximum damage to the network, will then attack links (1,2) and (1,4). The shortest path through the network in this worst case scenario is displayed in Figure 1.c, where protected arcs are indicated with cross hatched lines. If this protection strategy is implemented, the length of the post-interdiction shortest path, 1-2-4-3-6-7, is equal to 60. Although this protection plan reduces the impact of the disruption, the shortest path length still increases by more than 30%. Now consider an alternative protection strategy

where arcs (1,2) and (3,6) are hardened instead (Figure 1.d). The worst-case disruption in response to this protection occurs if arcs (2,4) and (5,7) are subsequently attacked. Although the shortest path is the same as in the previous case, its length is now 47. This represents a length increase of less than 5% as compared to the length of the best path when no delays occur, to denote a much sounder protection strategy. Also, note that this fortification strategy renders the second interdiction ineffective. In fact, the same worst-case path length of 47 can be obtained by attacking arc (2,4) only. This interdiction generates two paths with length 47 (1-2-4-3-6-7 and 1-2-5-7). Since these two paths only share the protected arc (1,2), which cannot be interdicted, it is not possible to disrupt both of them with only one interdiction resource still available. The choice of the second interdiction, (5,7), was arbitrary. Any other unprotected arc could have been chosen. This result not only demonstrates that hardening the most critical components may be a sub-optimal choice, but also that securing arcs in an optimal way can significantly reduce the impact of a worst-case disruption and, eventually, dissipate offensive resources.

This simple example highlights the importance of explicitly including protection decisions into the mathematical models. Different mathematical theories can be employed for the analysis of optimal defensive strategies. One of the major factors driving the choice of the appropriate modeling tool is the type of threats to which infrastructure systems are exposed. In particular, reliability theoretic models are more appropriate for planning the optimal defence of systems that face *probabilistic risk* (e.g., act of nature), whereas game-theoretic models are more suitable when allocating resources to counter *strategic risk* (e.g., malicious attacks) [11]. Modeling willful attacks, in fact, pose some unique challenges, primarily because intelligent attackers can adapt their course of action to the defender's strategy. In several recent studies, this interaction between defender and antagonist has been envisaged as a game and represented mathematically as a multi-level model. As an example, [3] applies attacker-defender models to determine the value of protecting assets in real-world infrastructure systems, such as power grids and petroleum pipeline networks. Protective plans for electric power grids are also analyzed in [16], where the authors propose a two-player zero-sum game to allocate protective resources so as to minimize the expected damage under different attacking strategies. The allocation of security resources to a water supply network is investigated in [23]. The objective of this model is to allocate a security budget to maximize an attacker's marginal cost of inflicting damage through the destruction of network components. A similar objective is used in [1], where the authors assume that the defender attempts to deter attacks by making them as costly as possible. This type of objective is particularly suitable to model problems

where the amount of resources available to the attacker is not known to the defender. A variation of this model can be found in [13]. In this work, defensive and offensive strategies are analyzed to improve the reliability of series and parallel systems. The reliability of each system component is assumed to depend on the level of defensive and offensive investments. An interesting model framework, where defensive investments are allocated to counter both terrorism and natural disaster, is proposed in [28]. The authors compare a sequential formulation and a simultaneous formulation of a defender-attacker game, and discuss policy implications and important insights into the nature of equilibrium defensive strategies. Finally, [24] propose a defender-attacker model to identify the optimal way of protecting facilities in service and supply systems so as to minimize the demand weighted distance among customers and facilities after worst-case disruptions. Single-level reformulations of this problem are provided in [4] and [25], whereas [18] presents a stochastic version of the model, where the number of possible attacks is uncertain.

The contribution of this paper is to apply a game theoretic framework to identify optimal protection strategies in shortest path networks. More specifically, we build upon the shortest path interdiction model proposed in [17] and add to it an additional layer which explicitly models protection decisions. The resulting model is a three-level program, for which we propose an optimal solution approach. A thorough computational investigation demonstrates that the proposed solution method is able to find optimal solutions for networks of significant size. We also discuss the sensitivity of different variants of the proposed approach to variations of the problem parameters, such as the level of offensive and defensive investments, and the distribution of the arc costs and delays. Finally, we introduce an alternative formulation for the shortest path interdiction model given in [17] which, when embedded within the defender-attacker bilevel construct, is remarkably more robust.

The paper is organized as follows. Section 1 presents the multi-level model. Section 2 describes the solution methodology. Section 3 introduces an alternative formulation for the lower level attacker problem. Section 4 presents the computational results. Finally, Section 5 contains some concluding remarks.

1 A Multi-Level Formulation

Given a directed graph $G=(N, A)$, where N is the set of nodes and A is the set of arcs, the *Shortest Path Interdiction Problem with Fortification* (SPIF) aims at identifying the set of linkages to harden in such a way that the length increase of the shortest path from

a supply node s to a demand node t due to a worst-case interdiction of some unprotected linkages is minimized.

Note that although we make the assumption that the critical components that can be interdicted and protected are the network linkages, it is easy to prove that problems where the critical components are the nodes can be reduced to critical arc models by opportunely augmenting the underlying graph [6]. Hence, we describe the more general case of arc protection and interdiction.

Let c_k be the nominal length of arc k and d_k the delay (or penalty or cost increase) to ship flow through k if the arc is interdicted. Arc lengths and arc delays are supposed to be non-negative integer. The complete loss of an arc can be captured in the model by choosing d_k sufficiently large [17]. We assume that both offensive and protective resources are limited so that at most R components can be attacked, and at most Q components can be hardened against interdiction. We also assume that an attack on a protected component has no effect so that the attacker should always hit unprotected components to optimize the use of his resources.

The multi-level formulation of SPIF uses the following sets of decision variables:

$$z_k = \begin{cases} 1, & \text{if arc } k \text{ is hardened,} \\ 0, & \text{otherwise} \end{cases}$$

$$s_k = \begin{cases} 1, & \text{if arc } k \text{ is interdicted,} \\ 0, & \text{otherwise} \end{cases}$$

$$y_k = \begin{cases} 1, & \text{if flow is shipped through arc } k, \\ 0, & \text{otherwise} \end{cases}$$

The set of feasible strategies of the defender can be defined in terms of the protection variables z_k as $D = \{\mathbf{z} \in \{0,1\}^{|A|} | \sum_{k \in A} z_k \leq Q\}$. Similarly, the set of feasible strategies of the attacker can be defined in terms of the interdiction variables s_k as $I = \{\mathbf{s} \in \{0,1\}^{|A|} | \sum_{k \in A} s_k \leq R\}$.

SPIF can then be formulated as the following three-level problem:

$$\text{(SPIF)} \quad \min_{\mathbf{z} \in D} \max_{\mathbf{s} \in I} \min_{\mathbf{y}} \quad \sum_{k \in A} (c_k + d_k s_k) y_k \tag{1}$$

$$\text{s. t.} \quad \sum_{k \in FS(i)} y_k - \sum_{k \in BS(i)} y_k = b_i \qquad \forall i \in N \tag{2}$$

$$s_k \leq 1 - z_k \qquad \forall k \in A \tag{3}$$

$$y_k \geq 0 \qquad \forall k \in A \tag{4}$$

As in standard shortest path problems, $b_s = 1, b_t = -1$, and $b_i = 0$ for all the other nodes i in N. $BS(i)$ and $FS(i)$ denote the backward star and the forward star of node i respectively. The objective function (1) computes the minimum-cost path after the worst-case interdiction of R unprotected components. This cost includes the penalties associated with interdicted arcs. Constraints (2) are standard flow-balance constraints for shortest path problems. Constraints (3) simply state that protected arcs cannot be interdicted. Finally, constraints (4) represent the nonnegativity requirements for the flow variables y_k. Note that integer restrictions for these variables are not required since the bottom level problem is a standard shortest path problem for which an optimal integral solution always exists.

This tri-level *defender-attacker-user* model can be reduced to a bilevel model by taking the dual of the bottom minimization problem. By doing so, the bilevel *attacker-user* model is collapsed into a single level *attacker* problem which inherently incorporates the user's optimal choice (see [17] for more details). The attacker problem can hence be solved by standard, off-the-shelf optimization software.

Let π_i be the dual variable associated with the flow-balance constraint for node i. The reduced *defender-attacker* bilevel problem is as follows:

$$\min_{\mathbf{z} \in D} \max_{\mathbf{s} \in I, \pi} \quad \pi_t \tag{5}$$

$$\text{s. t.} \quad \pi_j - \pi_i - d_k s_k \leq c_k \qquad \forall k = (i,j) \in A \tag{6}$$

$$s_k \leq 1 - z_k \qquad \forall k \in A \tag{7}$$

Note that in the above formulation we have omitted the dual variable π_s that can be set to zero to offset the presence of at least one redundant constraint in the bottom level shortest path problem of SPIF.

The number of levels of the above problem can be further reduced if we explicitly enumerate all possible ways of interdicting R out of the $|A|$ linkages. To this end, let H, indexed by h, be the set of all possible interdiction patterns, and I_h the set of interdicted arcs in pattern h. We denote by s^h the interdiction vector associated with pattern h, i.e.

$s^h = (s^h_1, \ldots, s^h_{|A|})$ with $s^h_k = 1$ if $k \in I_h$, $s^h_k = 0$ otherwise. Finally, let π^h be the vector of shortest path dual variables associated with a given interdiction pattern h. We can then formulate the *defender-attacker* problem as a single level problem in the z and π variables only.

$$\min \quad v \tag{8}$$

$$\text{s.t.} \quad v \geq \pi^h_t \qquad \forall h \in H \tag{9}$$

$$\pi^h_j - \pi^h_i - d_k s^h_k (1 - z_k) \leq c_k \qquad \forall k = (i,j) \in A, \forall h \in H \tag{10}$$

$$\sum_{k \in A} z_k \leq Q \tag{11}$$

$$z_k \in \{0,1\} \qquad \forall k \in A \tag{12}$$

Note that without loss of generality we can assume that each dual variable π^h_i in the above formulation represents the post-fortification, post-interdiction shortest path length from the source node s to node i, given that interdiction pattern h occurs. Although this is not necessarily true for every h, it can be easily enforced by adding an opportunely weighted secondary objective which minimizes the variables π^h_t.

It is clear that the use of this single level model is limited to problems of very modest size, due to the rapid growth of the number of variables and constraints for even small graphs and small values of R. Nevertheless, this formulation has been introduced to prove some theoretical results in Section 2.

Finding the optimal solution to problems of realistic and practical size requires devising sophisticated approaches tailored to the bilevel structure of the *defender-attacker* problem (5)-(7). Unfortunately, bilevel optimization problems are not amenable to solution by standard mixed integer programming methodologies, and no universal algorithm exists for their solution. Furthermore, problem (5)-(7) can be classified as a particularly hard bilevel problem in that the upper level variables parametrize the constraints of the lower level problem, and integer restrictions appear at both levels (see [22] or [9] for an in-depth discussion of discrete bilevel problems).

In the following section, we describe an implicit enumeration algorithm to solve problem (5)-(7). We also propose a heuristic to solve the lower level interdiction problem and show how the information acquired when solving the problem heuristically can be used to streamline the overall implicit enumeration approach. Finally we propose some variable fixing mechanisms to further reduce the complexity of the problems solved at each node

of the enumeration tree.

2 Solution Methodology

2.1 An Implicit Enumeration Algorithm

We solve SPIF using an enumeration scheme which exploits the solutions to the lower level interdiction problems to reduce the number of defender strategies to be evaluated [24]. The algorithm is built upon the simple observation that an optimal fortification set must include at least one of the components which are interdicted in an optimal solution to the interdiction problem without fortification [4]. This observation can be used recursively at each node of the enumeration tree to narrow the set of defender's variables to branch on. The general structure of the implicit enumeration algorithm is outlined below, followed by a formal proof of the correctness of the branching strategy.

In the pseudocode, we denote by $\overline{z}$ and $\overline{s}$ the vectors of fortification and interdiction variables respectively. Let r be the root node of the search tree and S the set of search-tree nodes to be visited. Each node n has the two following sets associated with it:

C_n	the set of candidate arcs to be hardened
F_n	the set of fortifications already made.

Finally, we denote by SPI(F_n) the conditional shortest path interdiction problem with the additional restriction that the arcs in F_n cannot be interdicted. We recall that each SPI(F_n) is a standard mixed integer programming problem than can be solved by general purpose MIP solvers.

Procedure ImplicitEnumeration

Init Phase.

Set $F_r = \emptyset$. Solve SPI(F_r) for $\hat{s}$, $\hat{\pi}$ and objective value $\hat{\pi}_t$;

Set $\overline{z} = 0$, $\overline{s} = \hat{s}$, and $\overline{\pi} = \hat{\pi}$; $C_r = \{j : \hat{s_j} = 1\}$; $S = \{r\}$;

Iterative Phase.

while $S \neq \emptyset$ **do**

begin

Select $n \in S$;

Select $k \in C_n$;

Generate node n_0 with $F_{n_0} = F_n$ and $C_{n_0} = C_n \setminus \{k\}$. If $C_{n_0} \neq \emptyset$ then $S = S \cup \{n_0\}$;

Generate node n_1 with $F_{n_1} = F_n \cup \{k\}$. Solve SPI(F_{n_1}) for $\hat{s}$, $\hat{\pi}$ and objective value $\hat{\pi}_t$;

If $|F_{n_1}| = Q$ or $\hat{s} = 0$ then \\ leaf node

if $\hat{\pi}_t < \overline{\pi}_t$ then \\ update best solution

$\overline{s} = \hat{s}$; $\overline{\pi} = \hat{\pi}$; for each $j \in A$, if $j \in F_{n_1}$ then $\overline{z}_j = 1$ else $\overline{z}_j = 0$;

endif;

Else $C_{n_1} = \{j | \hat{s_j} = 1\}$, $S = S \cup \{n_1\}$;

end

Return $(\overline{z}, \overline{s}, \overline{\pi})$.

The algorithm is implemented as a binary tree. The generation of a node n_0 represents the fact that the value of the selected variable for branching, z_k, is fixed to zero. In this case, either the set of candidate fortifications is empty and the node is fathomed, or the new node n_0 is added to the set of tree nodes to be visited. The generation of a node n_1 corresponds to fixing z_k to 1, in which case we solve a conditional shortest path interdiction problem with the new set of fortifications and generate a new set of variables to branch on. A leaf node is reached when exactly Q fortification variables are fixed to 1 or when, after the protection of the components in F_{n_1}, any attack on R of the remaining linkages does not worsen the shortest path. In the latter case, we can force the MIP solver to return an empty set of interdiction (i.e., $\hat{s} = 0$) by adding a secondary objective to SPI as explained at the end of this section.

In practice, the implicit enumeration algorithm is implemented as a depth-first search which uses recursion and backtracking. At each node n, the variable z_k to branch on is selected at random. Note that, if the conditional lower level interdiction problems have multiple optimal solutions, the order in which the branching variables are selected may lead to the generation and exploration of different sets of tree nodes. Consequently, the choice of the branching variables may have an impact on the size of the search tree and, hence, on the computing time of the overall approach. The identification of more sophisticated selection strategies for the branching variables, which take into account the existence of multiple optimal solutions to SPI, is still an open problem which may deserve future investigation.

The proposed solution methodology can be easily modified to handle the problem in which each link requires a different amount of offensive resources to be interdicted and there is a limit on the total resources available for interdiction. To this end, it is sufficient to replace the attacker cardinality constraints in the lower level interdiction problem with more general resource constraints. Similarly, the methodology can be used to solve problems with a budget constraint on the fortification resources instead of the cardinality

constraint defining the set of feasible protection strategies. In this case, a node n of the enumeration tree becomes a leaf node and is fathomed if the protection cost of any facility in C_n exceeds the residual budget available at that node.

We now state the correctness of the branching rule used in the tree search approach. This is equivalent to showing that the inequality $\hat{s}z \geq 1$ is *supervalid* to the defender problem. An inequality is supervalid if it does not cut off any optimal solutions unless the incumbent solution is already optimal [17]. We have thus to prove that $\hat{s}z^\star \geq 1$, where $(z^\star, s^\star, \pi^\star)$ denote an optimal solution to the defender-attacker problem and $\hat{s}$ is a feasible solution to the interdiction problem.

Theorem 2.1. *For any interdiction strategy $\hat{s}$ generated in the search tree, the inequality $\hat{s}z \geq 1$ is supervalid for the defender-attacker problem.*

Proof. Suppose that $(\hat{s}, \hat{\pi})$ is the solution to SPI(F_n) at a generic node n of the search tree and $\hat{\pi}_t$ is its objective value. Let $\hat{z}$ be the defender strategy associated with node n (i.e., for each $j \in A$, $\hat{z}_j = 1$ if $j \in F_n$, $\hat{z}_j = 0$ otherwise). Hence, the triplet $(\hat{z}, \hat{s}, \hat{\pi})$ represents a feasible solution to the *defender-attacker* problem (5)-(7). Let $v^\star$ be the objective value of an optimal solution $(z^\star, s^\star, \pi^\star)$. Suppose also that the incumbent solution $(\overline{z}, \overline{s}, \overline{\pi})$ is not optimal, i.e. $\overline{\pi}_t > v^\star$. Finally, let $\hat{s} = s^h$ for some $h \in H$ and π^h be the shortest path in response to $z^\star$ and s^h. Observe that $\hat{s}z^\star = 0$ or $\hat{s}z^\star \geq 1$, since s and z are 0-1 vectors. By contradiction, we assume that $\hat{s}z^\star = 0$. So, we have:

$$
\begin{aligned}
v^\star &\geq \pi_t^h \quad \text{(this is true for any } h \in H\text{. See constraints (9) in problem (8)-(12))} \\
&= \pi_i^h + d_k s_k^h (1 - z_k^\star) + c_k \quad \text{($k = (i,t)$ is the arc entering t in the shortest path π^h)} \\
&= \pi_i^h + d_k s_k^h + c_k \quad \text{(by hypothesis } \hat{s} = s^h \text{ and } \hat{s}z^\star = 0\text{; hence, } s_j^h z_j^\star = 0 \ \forall j \in A) \\
&\geq \pi_i^h + d_k s_k^h + c_k - d_k s_k^h \hat{z}_k \quad \text{(because } d_k > 0 \text{ and } \hat{s}_k, \hat{z}_k \geq 0) \\
&= \pi_i^h + d_k \hat{s}_k (1 - \hat{z}_k) + c_k \quad \text{(by replacing } \hat{s}_k = s_k^h) \\
&\geq \hat{\pi}_t \quad \text{(because } \hat{\pi}_t \text{ is the shortest path in response to } \hat{z} \text{ and } \hat{s}) \\
&\geq \overline{\pi}_t \quad \text{(because } (\hat{z}, \hat{s}, \hat{\pi}) \text{ need not be the incumbent solution)} \\
&> v^\star \quad \text{(by assumption).}
\end{aligned}
$$

But this is a contradiction. Thus, if the incumbent solution is not optimal, the inequality $\hat{s}z^\star \geq 1$ must be true for every optimal solution $(z^\star, s^\star, \pi^\star)$. □

Note that our branching rule generates an enumeration tree where the number of nodes only depends on the parameters Q and R, but not on the total number of components that can be protected and interdicted (i.e., $|A|$). More specifically, the size of the enumeration tree and, hence, the number of lower level problems to be solved, is bounded above by the quantity $(R^{Q+1}-1)/(R-1)$ (see [24] for a formal proof). In practice, the actual number of attacker problems to be solved can be further reduced by observing that a given SPI(F_n) may have alternative optimal solutions which use different amount of offensive resources. By selecting one of the optimal solutions which uses the smallest number of interdictions, we can reduce the size of the set of candidate components to be hardened, C_n, and, hence, the number of nodes to be processed. To this end, we have modified the objective function of the attacker problem by adding a secondary objective as follows:

$$max \quad \pi_t - \frac{1}{R+1} \sum_{k \in A} s_k \tag{13}$$

Note that the use of the weight $1 \setminus (R+1)$ guarantees that the value of the new term added to the objective is always less than one. Since we deal with non-negative integer path lengths, this weight ensures that the maximization of the shortest path length π_t remains the primary objective. Preliminary results have shown that this simple expedient can significantly reduce the number of mixed integer problems solved in the implicit enumeration procedure.

In addition to the reduction of the search tree size, it is clear that the computing effort of the proposed approach may greatly benefit from a reduction of the computing time needed to solve each instance of the attacker problem. We now describe how to solve the interdiction problem more efficiently so as to improve the scalability of the proposed solution methodology.

2.2 Heuristic Solutions to the Interdiction Problem

Although the validity of our branching strategy relies upon finding optimal solutions to the interdiction problems, we use an heuristic approach to generate valid lower bounds and supervalid inequalities for the attacker problems. The approach consists of a greedy strategy which starts with an empty set of interdictions and iteratively adds components to it according to a greedy rule. At each iteration, the newly selected component for interdiction is the linkage on the current shortest path whose delay results in the maximum length increase of the s-t shortest path in the graph.

A scheme of the greedy process is provided below. In the pseudocode, $\mathscr{P}$ denotes the set of shortest paths processed during the heuristic execution; P^m is the shortest path after the selection of m-1 interdictions , with $1 \leq m \leq R$, and L^m is its length. Finally, P_k is the shortest path computed in the graph $G = (N, A)$ when arc k is removed; L_k is the length of path P_k.

Init Phase.

Let P be the shortest path in $G = (N, A)$ and L its length;

Set $P^1 = P$; $L^1 = L$; $\mathscr{P} = \{P\}$; $\overline{s}_j = 0$ for each $j \in A$.

Iterative Phase.

for $m = 1, ..., R$ **do**

begin

for each arc k in P^m s.t. $\overline{s}_k = 0$ **do**

begin

$P_k = ComputeShortestPath(N, A \setminus \{ k \})$;

$v_k = \min \{L^m + d_k, L_k \}$;

end

$\hat{k} = \text{argmax}_{k \in P^m}\{v_k\}$;

$c_{\hat{k}} = c_{\hat{k}} + d_{\hat{k}}$; $\overline{s}_{\hat{k}} = 1$;

if $v_{\hat{k}} = L_{\hat{k}}$ then \\ the shortest path in response to $\overline{s}$ has changed

$P^{m+1} = P_{\hat{k}}$; $L^{m+1} = L_{\hat{k}}$; $\mathscr{P} = \mathscr{P} \cup \{P^{m+1}\}$;

else $P^{m+1} = P^m$; $L^{m+1} = v_{\hat{k}}$;

end

Return $\overline{s}$, $\mathscr{P}$, P^{R+1}, L^{R+1}.

It has to be noted that in the heuristic step where the new arc $\hat{k}$ is selected, there may be several arcs whose interdiction results in the same maximum value of v_k. In this event, we break ties by selecting the arc which, when interdicted, generates an s-t shortest path different from the current one. Further ties are resolved by selecting the arc with the highest delay. The selection step also needs be modified when the heuristic is used at a generic node n of the search tree other than the root node. In this case, in fact, we must not process the arcs in P^m which have already been fortified, i.e., the arcs in the set F_n.

At termination, P^{R+1} is the shortest path in response to the interdiction plan $\overline{s}$. Its length, L^{R+1}, represents a lower bound to the optimal solution of the attacker problem. We supply the mixed integer programming solver with this bound so as to cut off some of

the branch and bound nodes at an early stage. Note that when $R = 1$, the solution found by the greedy heuristic is optimal and, therefore, there is no need to call the MIP solver.

To further reduce the number of branches needed to solve the MIP formulation of the attacker problem, we use the set of shortest paths $\mathscr{P}$ in the following supervalid inequalities which are added to the SPI(F_n) models:

$$\sum_{k \in P^m} s_k \geq 1, \quad \forall m = 1...|\mathscr{P}| \tag{14}$$

Constraints (14) simply force the interdiction of at least one of the arcs in each path generated by the heuristic.

2.3 Variable Fixing Rules

In this section, we present two different rules which can be used to eliminate some interdiction variables from the attacker problems, thus accelerating the solution time of the implicit enumeration procedure.

Bellman Fixing Rule

Let π_i^b denote the length of the shortest path from the source node s to a generic node i in the original graph $G = (N, A)$ without interdictions. Let π_i^w denote the length of the shortest path from the source node s to node i in the graph G where all the arcs are interdicted.

Proposition 2.1. *If $\pi_j^w - \pi_i^b \leq c_{ij}$, then the interdiction variable s_k associated with the arc $k = (i, j)$ can be set to zero.*

This proposition is based on the simple observation that if in the graph where all the arcs are interdicted, node j can be reached through a path which is shorter than the shortest path through node i in the graph without interdictions, then interdicting arc (i, j) is never beneficial for the attacker. Therefore, in the initialization phase of the implicit enumeration algorithm, we check the above Bellman-like condition for each arc k in the graph and exclude the interdiction variables associated with the arcs satisfying this condition from further consideration.

KSP Fixing Rule

Assume that π_t^r is the objective value of the optimal solution to the SPI problem obtained at the root node of the enumeration tree. We denote by $\mathscr{P}_r$ the set of s-t paths in $G = (N, A)$ whose nominal length is less than or equal to π_t^r. Let f_k denote the frequency of appearance of arc k in the s-t paths in $\mathscr{P}_r$.

Proposition 2.2. *If for some arc k, we have that $f_k = 0$, then the interdiction variable s_k can be set to zero.*

The rationale behind this proposition is as follows. When none of the network components is hardened, an optimal attack on R linkages can thwart all the s-t paths with length up to π_t^r. When some fortifications are made in subsequent steps of the algorithm, an optimal attack will be less disruptive and will only be able to thwart a subset of these paths. As such, offensive resources should be targeted so as to maximize the number of shortest paths in $\mathscr{P}_r$ that can be disrupted. Clearly, resources utilized to interdict arcs which do not appear in any of these paths will be dispersed. Consequently, after solving the interdiction problem at the root node, we can restrict the focus of the solution approach only to those arcs which belong to at least one of the paths in $\mathscr{P}_r$ and eliminate all the other interdiction variables. In our implementation of the modified implicit enumeration procedure, the paths in $\mathscr{P}_r$ were computed by using the shortest path ranking algorithm proposed in [21]. This algorithm proved to be very efficient for generating and ranking millions of shortest paths between pairs of nodes in networks of significant size. In our computational experiments, the computing time for generating the paths in $\mathscr{P}_r$ was negligible as compared to the computing time of the overall procedure.

3 An alternative formulation of the attacker problem

The rationale behind proposition 2.2 also suggests an alternative way of formulating the attacker problem at each node n of the enumeration tree other than the root node.

Let P_m be the m-th shortest path in $\mathscr{P}_r$ and L_m its length. An alternative formulation for SPI(F_n), denoted in the following as KSPI(F_n), is as follows:

$$\max \quad L \tag{15}$$

$$\text{s.t.} \quad L \leq L_m + \sum_{k \in P_m} d_k s_k \qquad \forall m \in \mathscr{P}_r \tag{16}$$

$$\sum_{k \in A} s_k \leq R \tag{17}$$

$$s_k \leq 1 - z_k \qquad \forall k \in F_n \tag{18}$$

$$s_k \in \{0, 1\} \qquad \forall k \in A \tag{19}$$

At optimality, the variable L in the objective function (15) represents the length of the shortest path after the most disruptive interdiction. This is enforced by constraints (16). This formulation has $|A|$ integer variables and $|\mathscr{P}_r|+|F_n|+1$ constraints. Even in this case, some of the interdiction variables can be eliminated by using propositions 2.1 and 2.2. The original formulation of the interdiction problem has $|A|$ integer variables, $|N|$ continuous variables and $|\mathrm{A}|+|F_n|+1$ constraints. Given that the number of paths in the set $\mathscr{P}_r$ is largely problem specific, it is not possible to establish a priori the dominance of one formulation over the other in terms of dimension and difficulty of solution. An advantageous feature of the alternative formulation is that the number of constraints (16) can be reduced every time we solve a new problem at a child node of the enumeration tree. Assume for example that L^* is the optimal solution to problem (15)-(19) at node n. When we solve the interdiction problem at n's child node, we can safely remove from the formulation all the constraints associated with those paths whose length is strictly greater than L^*. In fact, if the length of these paths could not be worsened with the set of interdictions available at node n, they cannot be worsened when the set of candidate interdictions is restricted at the child node by hardening an additional component. As we go down in the tree, we can use this problem reduction iteratively to obtain problems which are increasingly smaller. The use of this alternative formulation becomes therefore more attractive when solving problems with larger values of Q, since the parameter Q determines the depth of the enumeration tree.

An in-depth discussion of the tradeoffs between using formulation (15)-(19) versus the original SPI formulation within the enumeration procedure is given in Section 4.

Note that the formulation (15)-(19) can be generalized to solve generic instances of SPI by computing an upper bound of the number of shortest paths that can be interdicted with R resources and including a constraint (16) in the formulation for each of these shortest paths.

Grid	Nodes	Arcs
7-7	51	188
10-10	102	416
12-12	146	618
15-15	227	996

Table 1: Grid dimensions

4 Experimental results

4.1 Test problems and implementation issues

In our computational testing, we use directed square grid graphs with a number of rows (and columns) in $\{7, 10, 12, 15\}$. The grids are generated by following the rules described in [17] so as to obtain networks with the same topology as the ones used to test the shortest path interdiction model without fortification. Given a positive integer c which represents the maximum arc cost, arc costs are uniformly chosen in $[0, c]$; then for a given c, we consider three different delay distributions: (i) delays uniformly chosen in $[0, c/2]$; (ii) delays uniformly chosen in $[0, c]$; and delays uniformly chosen in $[0, 2c]$. In our experimental analysis we use two values for c, $c = 10$ and $c = 100$, thus summing up to 6 cost-delay combinations for each grid type. For a fixed grid dimension and cost-delay combination, we generate 3 instances with different random arc attributes. Cardinality constraints on offensive and defensive resources are considered where the maximum number Q of fortifications belongs to $\{3, 5, 7\}$ and the maximum number R of interdictions is in $\{1, 2, 3, 4, 5\}$. Each instance is solved for each combination of Q and R in the aforementioned sets, except the largest grids (15 x 15) for which the case $Q = 7$ and $R = 5$ is too computationally expensive and is therefore excluded. The total number of instances is thus equal to 1062. The characteristics of the networks in terms of nodes and arcs are given in Table 1.

We code the algorithms in C++ and run them on a PC with an Intel Core 2, 2.66 Ghz processor and 2GB of RAM. The commercial solver CPLEX, version 11 is used to solve the MIP problems at each node of the enumeration tree.

Two implementations of the implicit enumeration scheme are considered depending on the formulation used for the lower level interdiction problem at each node of the enumeration tree, namely SPI(F_n) or KSPI(F_n) described respectively in Sections 1 and 3. For each of the two formulations, three variants are investigated: (i) the basic formulations are directly solved with Cplex without any attempt at reducing the search space. These two versions are referred to as SPINot and KSPINot respectively; (ii) the implicit enu-

meration scheme is equipped with the Bellman and KSP variable fixing rules described in Section 2.3. Additionally the greedy heuristic is used at the root node. These two versions are referred to as SPIFix and KSPIFix; (*iii*) in addition to the variable fixing rules, the greedy heuristic and the inequalities (14) are used at each node of the enumeration tree. These two versions are referred to as SPIAll and KSPIAll.

4.2 Performance Comparison

In Figure 2, a performance comparison is made among the six variants of the implicit enumeration scheme described above. More specifically, for each variant we report the percentage of instances solved in a given time interval, where the time classes, represented on the x-axis of the bar chart, are as follows: less than one second, between one second and one minute, between one minute and 10 minutes, between 10 minutes and one hour, between one hour and three hours and finally, over three hours. These aggregated results highlight two important facts: 1) the use of the variable fixing rules and of the heuristic allows us to solve more instances in shorter times; 2) the use of the KSPI formulation in the lower level interdiction problem makes the enumeration approach more efficient. A direct comparison between SPIAll and KSPIAll, for example, shows that the percentage of instances solved in less than one minute with KSPIAll is as high as 85.4, whereas for the counterpart SPIAll the percentage is equal to 79.3. Additionally, the number of instances solved in over one hour is negligible when any of the KSPI-based variant is used.

To analyze in more depth the performance trend of the different approaches, we present some detailed results in Table 2. For a given grid dimension and a given cost-delay distribution, the average computational time, expressed in CPU seconds, is given for each of the six variants of the algorithm. The problem name, shown in the first column, contains four fields which represent respectively the number of rows and columns in the grid graph, the maximum cost c and the maximum delay d. Each row of the table is thus averaged over 45 instances (3 instances times 15 Q-R combinations) when the grid dimension is in $\{7, 10, 12\}$ and over 42 instances for the 15-row grids (3 instances times 14 Q-R combinations). The results are grouped according to the value of c: in the first block of rows results are shown for costs ranging in $[0, 10]$, while the second group of rows relates to instances characterized by a higher variability of the cost coefficients which range in [0,100]. In each row the most efficient approach is displayed via a bold entry. In the last row of the table average computational times over all the instances are given.

As to be expected, the overall results show that the instances become more difficult to solve as the dimension of the grid increases. Also, for a fixed grid dimension, the

difficulty grows as the delay range augments, so that solving instances with delays in the range $[0, 2c]$ consistently requires more time than solving instances where the delays vary in $[0, c/2]$. In addition, the aggregated results reported in the last row confirm that the KSPI approaches dominate the SPI implementations and that the tools used to reduce the computational burden of the basic approach are quite effective. In general, their effect is more pronounced in the SPI-based variants than in the KSPI ones.

A closer look at the two blocks of instances in Table 2 shows that the approaches may perform quite differently depending on the cost range used (and, consequently, the delay range). For instances with smaller costs (first block), the approaches based on the SPI formulation greatly benefit from the introduction of the fixing rules and the heuristic, so that the variant SPIAll is remarkably more efficient than the variant SPIFix, which, in turn, is faster than the variant SPINot. The three KSPI-based variants exhibit much smaller computational times with respect to their SPI counterparts across the full range of instances in this block and the impact of the reduction techniques is less noticeble.

On the second block of instances, characterized by wider cost ranges, the SPI and the KSPI implementations display somewhat opposite behaviours. The SPI variants, in fact, generally solve these instances more rapidly than the ones with smaller costs and delays. As an example, consider the SPIAll performance on the 10x10 and the 12x12 grids. The computing times to solve the instances in the first block can be as high as double the times to solve instances of equal size in the second block (see for example the instance 10-10-10-20 vs the instance 10-10-100-200, or the instance 12-12-10-20 vs the instance 12-12-100-200). The impact of the cost-delay range on the performance of the SPI variants seems to be negligible for the 15x15 grids. On the other side, the performance of the KSPI implementations clearly deteriorates when wider cost ranges are considered. This is particulary evident for the bigger instances in the test bed (e.g., 15-15-100-200).

Overall, the preeminence of the KSPI-based variants seems quite overwhelming, if we exclude the 15-15-100-200 instances. Possible causes which may render these instances more difficult to solve by the KSPI implementations will be further investigated in the next paragraphs.

In Table 3, we provide some information related to the fixing rules and the KSP paths generated by the approaches. The level of aggregation of the results in Table 3 is the same as in Table 2. For each grid dimension and each cost-delay combination, the table provides the following information: the average number, KSPPath, of s-t shortest paths whose cost does not exceed the upper bound to the three-level defender-attacker-user problem objective computed at the root node via the SPI formulation with no fortifications; the

average percentages of non-interdictable arcs computed by the Bellman-like conditions (%Bfix) and by the k-shortest path tools (%Kfix); and, in the last column, the average percentage of arcs fixed by both the rules described above. First of all, we observe that the KSP fixing rule is very effective in reducing the search space both for easy and critical instances and is more powerful than the Bellman fixing rule. Nevertheless, the Bellman fixing rule is always able to identify some additional variables to fix which could not be identified by using the KSP fixing rule alone, as shown by comparing the fourth and the last column of the table.

By looking at the KSPPath column in Table 3, we can observe that the average number of KSP paths for the instances in the class 15-15-100-200 is significantly higher than the KSPPath number for the other classes. This explains the greater difficulty of the KSPI formulation at solving these problems. Recall, in fact, that whereas the number of constraints in the SPI formulation of the interdiction problem strictly depends on the number of arcs, in the KSPI formulation the number of constraints is determined by the number of KSP shortest paths identified at the root node. This observation suggests the use of an hybrid approach where the interdiction problem formulation to be used in subsequent nodes of the enumeration tree (SPI or KSPI) is selected on the fly at the root node after the number KSPPath is computed: if this number is significantly higher than the number of arcs (as is the case for some of the 15-15-100-200 problems) the SPI formulation should be selected; conversely, if the number of KSP paths is in the same order of magnitude as the number of arcs, the KSPI formulation should be opted for.

It is important to note that the number of KSP paths is largely problem specific and may vary considerably even for networks of the same size and with the same cost and delay distributions. This is true, for example, for the class 15-15-100-200, where the high KSPpath number and, consequently, the high average computing times of the KSPI algorithms are mainly due to one of the three instances. In support of this observation, we provide in Table 4 the disaggregated results for two instances of type 15-15-100-200: in the first column the name of the instance is given where the last field refers to the instance index ranging in $\{0, 1, 2\}$; this is followed by the number Q of fortifications and the number R of interdictions; the next four columns give the information relative to the fixing rules in the same format as in Table 3; finally, the computational time expressed in CPU seconds is reported for the two most efficient implementations of each formulation type, i.e. SPIAll and KSPIAll. These disaggregated results reveal that KSPIAll performs better than SPIAll on the instance 15-15-100-200-0: note that in this case the number of KSPPaths is never over 3500 and there are some Q-R difficult combinations (see for

example the 5-4, 5-5, and 7-4 cases) that can be solved by KSPIAll in about one fourth of the time required by SPIAll. A completely different behavior can be observed on the instance 15-15-100-200-1: there are three cases (see the 3-4, 5-4 and 7-4 cases) where the time required by KSPIAll is remarkably greater than the one required by SPIAll; additionally there are two cases for which KSPIAll needs a huge time (see 3-5 and 5-5 cases for which the times are respectively about 4 and over 15 hours). In all these five cases the poor performance of KSPIAll is due to the high number of KSPPath which can reach the value 36805. These results seem thus to corroborate the fact that an hybrid approach may be successfully used to overcome the weaknesses of the SPI-based and the KSPI-based approaches.

Finally, note that the number of KSP paths depends on the amount of interdiction resources: in fact, the largest the value of R, the greater the number of shortest paths that can be interdicted with the R resources. This may render the use of the KSPI formulation more critical for solving problems with large values of the parameter R. Nonetheless, the range of values considered in our computational tests (R between 1 and 5) seems to be the most interesting from a practical point of view. The simultaneous loss of a much larger number of components, in fact, seems quite unrealistic or, at least, quite unlikely to occur. On the other side, the parameter Q may vary in a much broader range, given the high variability of the amount of defensive resources that may be allocated for protection purposes.

4.3 Impact of offensive and protective resources

In Figures 3 and 4 a scalability comparison with respect to the parameters Q and R is made between SPIAll and KSPIAll on medium size grids, namely the 10x10 grids, for the most critical cost-delay combinations in the test bed, i.e. respectively for costs ranging in $[0, 10]$ and delays in $[0, 20]$ and costs ranging in $[0, 100]$ and delays in $[0, 200]$. Specifically, we report in a logarithmic scale the average time computed over the three instances of the same type for each of the 15 Q-R combinations analyzed in our computational study and shown on the x-axis. The two cost-delay distributions exhibit a quite similar behavior in terms of increase in the computational time as Q and R grow.

More specifically, Figures 3 and 4 highlight that the impact of the number R of offensive resources is more decisive than the number Q of protective resources in that: 1) for a fixed value of Q (see each of the 3 blocks of 5 consecutive pairs of bars), the computational time of both the approaches increases by one order of magnitude for a unitary increase of R; 2) for a fixed value of R the time required by instances characterized by the biggest value

of Q in the test bed, namely $Q = 7$ is about one order of magnitude larger than the time required by instances with the smallest value of Q in the set considered, namely $Q = 3$; 3) when the number of interdictions is small (see R=1 and R=2) the computational times are negligible (never over 1 second) for both the approaches independently on the number of fortifications. These results indicate that, especially when the expected number of possible losses is low, the proposed method can be used to identify the optimal network components to be hardened for relatively large networks and large protection budgets.

5 Conclusions

We develop a multi-level optimization model, called the Shortest Path Interdiction Problem with Fortification (SPIF), to identify optimal protection strategies in shortest path networks. With a simple example, we show that the explicit inclusion of protection decisions into an optimization model can produce much sounder protection plans than simply using a shortest path interdiction model (such as the one proposed in [17]) to identify the critical components to be hardened: the impact of a worst-case disruption can be significantly reduced and some offensive resources can even be rendered ineffective. Clearly, the benefit of including protection decisions into the model comes at a cost: the resulting model is a three-level program and its solution requires the devise of sophisticated approaches tailored to this multi-level structure.

We propose an implicit enumeration algorithm for solving SPIF, which exploits the solution to the lower level interdiction problem to reduce the number of protection strategies to be evaluated. Given that the computation effort of the proposed method is largely determined by the number of lower level interdiction problems to be solved in the enumeration tree and by the efficiency with which they can be solved, we introduce several expedients to reduce the search tree size and speed up the computation. These include some variable fixing mechanisms, and the use of a heuristic and some supervalid inequalities at each node of the search tree. We also propose an alternative formulation for the shortest path interdiction problem which generally outperforms the formulation proposed in [17] when used within the implicit enumeration algorithm.

An extensive computational campaign demonstrates that the proposed methodology is quite effective at solving protection problems on networks with up to more than 200 nodes and almost a thousand arcs. We also show how the distribution of the arc lengths and delays can affect the efficiency of different implementation variants and suggest the use of a hybrid approach to enhance the performance of our solution method. Finally,

we study the sensitivity of the approach to variations of the number of fortifications and interdictions and we show that the effect of the number of offensive resources is more crucial than the number of protective resources.

The need of developing quantitative methods for improving the ability of logistics systems to withstand intentional or accidental disruptions has been widely recognized in the scientific community, among practitioners, supply chain managers and government agencies. Although in recent years a few researchers have begun studying mathematical models able to address protection issues in infrastructure systems, the level of sophistication of these models needs to be further improved in order to cope with the complexities of today logistics systems. Currently, we are exploring different variations of the SPIF model, which may increase its applicability to solve real-world problems. As an example, we are investigating a stochastic variant of SPIF where the number of possible losses is uncertain and unknown to the defender. In another model variation, we assume that the protection of a network component only reduces its probability of disruption, and that the extent to which the disruption probability is reduced depends on the level of investment in protective measures. Finally, we plan to develop a multi-period version of SPIF where protection resources become available in different periods and the objective is to maximize the system robustness at the end of the planning horizon while guaranteeing some minimum level of robustness in each time period.

Acknowledgment

This research was supported by EPSRC Grant EP/E048552/1. This support is gratefully acknowledged.

References

[1] M.N. Azaiez and V. M. Bier. Optimal resource allocation for security in reliability systems. *European Journal of Operational Research*, 181:773–786, 2007.

[2] M.O. Ball, B. L. Golden, and R. V. Vohra. Finding the most vital arcs in a network. *Operations Research Letters*, 8:73–76, 1989.

[3] G. Brown, M. Carlyle, J. Salmeron, and K. Wood. Defending critical infrastructure. *Interfaces*, 36(6):530–544, 2006.

[4] R. L. Church and M. P. Scaparra. Protecting critical assets: The r-interdiction median problem with fortification. *Geographycal Analysis*, 39:129–146, 2006.

[5] R. L. Church, M. P. Scaparra, and R. S. Middleton. Identifying critical infrastructure: the median and covering facility interdiction problems. *Annals of the Association of the American Geographers*, 94(3), 2004.

[6] H. W. Corley and H. Chang. Finding the most vital nodes in a flow network. *Management Science*, 21(3):362–364, 1974.

[7] H.W. Corley and D. Y. Sha. Most vital links and nodes in weighted networks. *Operations Research Letters*, 1(4):157–160, 1982.

[8] K. J. Cormican, D. P. Morton, and R. K. Wood. Stochastic network interdiction. *Operations Research*, 46(2):184–197, 1998.

[9] S. Dempe. *Foundations of Bilevel Programming*. Kluwer Academic Publishers, The Netherlands, 2002.

[10] D. R. Fulkerson and G. C. Harding. Maximizing the minimum source-sink path subject to a budget constraint. *Mathematical Programming*, 13:116–118, 1977.

[11] B. Golany, E. H. Kaplan, A. Marmur, and U. G. Rothblum. Nature plays with dice - terrorists do not: Allocating resources to counter strategic versus probabilistic risks. *European Journal of Operational Research*, 0000:00–00, 2008.

[12] B. Golden. A problem of network interdiction. *Naval Research Logistics Quarterly*, 25:711–713, 1978.

[13] K. Hausken. Strategic defense and attack for series and parallel reliability systems. *European Journal of Operational Research*, 186:856–881, 2008.

[14] H. Held, R. Hemmecke, and D. L. Woodruff. A decomposition algorithm applied to planning the interdiction of stochastic networks. *Naval Research Logistics*, 52:321–328, 2005.

[15] R. Hemmecke, R. Schultz, and D.L. Woodruff. *Network interdiction and stochastic integer programming*, chapter Interdicting stochastic networks with binary interdiction effort, pages 69–84. Kluwer, Boston, 2003.

[16] A. J. Holmgren, E. Jenelius, and J. Westin. Evaluating strategies for defending electric power networks against antagonistic attacks. *IEEE Transactions on Power Systems*, 22(1):76–84, 2007.

[17] E. Israeli and R. K. Wood. Shortest-path network interdiction. *Networks*, 40(2):97–111, 2002.

[18] F. Liberatore, M.P. Scaparra, and M. Daskin. Analysis of facility protection strategies against an uncertain number of attacks: The stochastic R-interdiction median problem with fortification. Kbs working paper 176, University of Kent, 2008.

[19] C. Lim and J. C. Smith. Algorithms for discrete and continuous multicommodity flow network interdiction problems. *IIE Transactions*, 39(1):15–26, 2007.

[20] K. Malik, A. K. Mittal, and S. K. Gupta. The k most vital arcs in the shortest path problem. *Operations Research Letters*, 8:223–227, 1989.

[21] E.Q.V. Martins and J.L.E. Santos. A new shortest path ranking algorithm. *Investigação Operational*, 20(1):47–62, 2000.

[22] J.T. Moore and J.F. Bard. The mixed integer linear bilevel programming problem. *Operations Research*, 38:911–921, 1990.

[23] J. Qiao, D. Jeong, M. Lawley, J. P. Richard, D. M. Abraham, and Y. Yih. Allocating security resources to a water supply network. *IIE Transactions*, 39:95–109, 2007.

[24] M. P. Scaparra and R. L. Church. A bilevel mixed integer program for critical infrastructure protection planning. *Computers & Operations Research*, 35:1905–1923, 2008.

[25] M. P. Scaparra and R. L. Church. An exact solution approach for the interdiction median problem with fortification. *European Journal of Operational Research*, 189:76–92, 2008.

[26] E. Sternberg and G. Lee. Meeting the challenge of facility protection for homeland security. *Journal of Homeland Security and Emergency Management*, 3(1):1–19, 2006.

[27] R. Wollmer. Removing arcs from a network. *Operations Research*, 12(6):934–940, 1964.

[28] J. Zhuang and V. M. Bier. Balancing terrorism and natural disasters - defensive strategy with endogenous attacker effort. *Operations Research*, 55(5):976–991, 2007.

ProbName	**SPIAll**	**SPIFix**	**SPINot**	**KSPIAll**	**KSPIFix**	**KSPINot**
7-7-10-5	0.44	0.59	0.84	0.26	**0.25**	0.28
7-7-10-10	3.05	6.32	8.15	**1.51**	2.22	2.28
7-7-10-20	5.90	11.19	15.22	**3.46**	4.27	4.44
10-10-10-5	32.20	48.21	63.30	**6.90**	7.01	7.74
10-10-10-10	79.36	138.23	216.10	**67.87**	87.91	86.73
10-10-10-20	392.65	903.19	1428.30	**79.43**	97.45	103.39
12-12-10-5	108.78	241.38	249.02	**37.64**	38.94	40.24
12-12-10-10	545.66	622.76	880.17	**130.54**	131.73	135.11
12-12-10-20	1008.21	1934.36	2865.31	**216.93**	220.39	221.68
15-15-10-5	69.09	98.36	124.05	**29.40**	32.48	33.63
15-15-10-10	635.08	967.96	1454.23	**72.51**	76.91	78.65
15-15-10-20	927.57	1722.24	1998.11	**225.43**	253.45	262.42
7-7-100-50	2.50	3.03	4.17	**1.46**	1.47	1.52
7-7-100-100	1.50	1.76	2.35	**1.13**	1.28	1.29
7-7-100-200	3.55	7.89	12.18	**2.59**	3.78	3.83
10-10-100-50	11.38	18.61	21.06	**9.88**	10.82	11.23
10-10-100-100	85.23	171.44	239.44	**28.62**	36.09	36.77
10-10-100-200	218.75	474.43	649.05	**46.37**	65.39	65.10
12-12-100-50	**86.57**	118.18	134.46	92.64	97.52	97.74
12-12-100-100	243.70	412.26	555.98	**78.73**	85.39	87.04
12-12-100-200	579.62	931.21	1311.34	**309.77**	315.31	352.42
15-15-100-50	**69.09**	83.82	111.98	247.36	261.88	263.61
15-15-100-100	865.63	1843.06	2760.39	**132.72**	145.89	147.95
15-15-100-200	**916.69**	1665.33	2473.77	2367.97	2831.52	2799.78
Average	287.18	517.74	732.46	**174.63**	200.39	201.87

Table 2: Time comparison results

ProbName	**Arcs**	**KSPPath**	**%Bfix**	**%Kfix**	**%BKFix**
7-7-10-5	188	87.53	26.95	72.73	21.81
7-7-10-10	188	64.47	14.01	68.72	11.21
7-7-10-20	188	94.09	7.67	67.63	4.57
10-10-10-5	416	118.47	15.95	77.92	14.25
10-10-10-10	416	390.33	9.54	72.82	7.34
10-10-10-20	416	249.13	7.37	74.44	5.66
12-12-10-5	618	675.13	11.97	80.87	10.63
12-12-10-10	618	450.73	7.71	79.36	6.31
12-12-10-20	618	438.27	4.05	76.44	3.03
15-15-10-5	996	791.43	10.07	82.96	8.96
15-15-10-10	996	223.29	7.56	85.29	6.75
15-15-10-20	996	1551.88	3.25	73.86	2.14
7-7-100-50	188	37.40	33.05	78.13	30.71
7-7-100-100	188	99.33	10.46	71.56	8.19
7-7-100-200	188	110.07	7.27	71.63	4.54
10-10-100-50	416	204.13	18.99	80.22	16.68
10-10-100-100	416	196.33	8.65	78.27	7.24
10-10-100-200	416	220.13	4.97	76.35	3.48
12-12-100-50	618	686.13	13.27	80.19	12.28
12-12-100-100	618	224.87	7.82	81.26	6.60
12-12-100-200	618	632.20	3.45	79.59	2.41
15-15-100-50	996	791.43	10.07	82.96	8.96
15-15-100-100	996	534.00	6.09	83.48	5.28
15-15-100-200	996	3266.31	3.85	79.37	2.84

Table 3: Impact of the variable fixing rules

ProbName	Q	R	KSPPath	%Bfix	%Kfix	%BKFix	SPIAll	KSPIAll
15-15-100-200-0	3	1	9	3.92	95.68	3.71	**0.06**	**0.06**
15-15-100-200-0	3	2	310	3.92	87.45	3.21	1.09	**0.91**
15-15-100-200-0	3	3	344	3.92	85.64	3.11	34.05	**6.02**
15-15-100-200-0	3	4	2146	3.92	73.69	2.71	280.81	**83.92**
15-15-100-200-0	3	5	3453	3.92	70.08	2.71	626.64	**381.52**
15-15-100-200-0	5	1	9	3.92	95.68	3.71	0.06	**0.05**
15-15-100-200-0	5	2	310	3.92	87.45	3.21	3.14	**2.00**
15-15-100-200-0	5	3	344	3.92	85.64	3.11	136.46	**16.27**
15-15-100-200-0	5	4	2146	3.92	73.69	2.71	1880.25	**413.02**
15-15-100-200-0	5	5	3453	3.92	70.08	2.71	6542.29	**2274.67**
15-15-100-200-0	7	1	9	3.92	95.68	3.71	0.08	**0.05**
15-15-100-200-0	7	2	310	3.92	87.45	3.21	7.86	**3.52**
15-15-100-200-0	7	3	344	3.92	85.64	3.11	318.44	**37.77**
15-15-100-200-0	7	4	2146	3.92	73.69	2.71	6965.67	**1546.05**
15-15-100-200-1	3	1	37	4.02	93.78	3.61	0.13	**0.09**
15-15-100-200-1	3	2	354	4.02	81.83	3.01	1.66	**1.59**
15-15-100-200-1	3	3	2008	4.02	69.48	2.71	**9.63**	44.44
15-15-100-200-1	3	4	7473	4.02	61.95	2.21	**90.99**	1169.52
15-15-100-200-1	3	5	36805	4.02	50.50	2.01	**487.15**	17645.22
15-15-100-200-1	5	1	37	4.02	93.78	3.61	0.14	**0.09**
15-15-100-200-1	5	2	354	4.02	81.83	3.01	3.50	**2.61**
15-15-100-200-1	5	3	2008	4.02	69.48	2.71	**34.36**	115.67
15-15-100-200-1	5	4	7473	4.02	61.95	2.21	**473.93**	3706.92
15-15-100-200-1	5	5	36805	4.02	50.50	2.01	**2124.87**	56625.07
15-15-100-200-1	7	1	37	4.02	93.78	3.61	0.19	**0.09**
15-15-100-200-1	7	2	354	4.02	81.83	3.01	6.39	**3.11**
15-15-100-200-1	7	3	2008	4.02	69.48	2.71	**93.63**	208.76
15-15-100-200-1	7	4	7473	4.02	61.95	2.21	**1408.74**	9000.23

Table 4: Performance comparison on two difficult instances

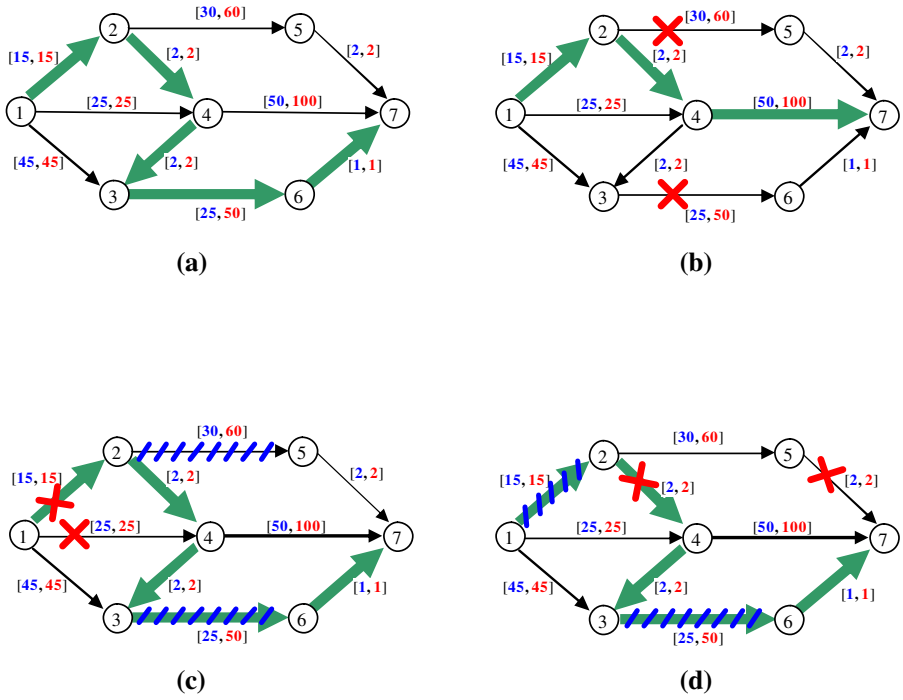

Figure 1: Interdiction and protection of a shortest path network

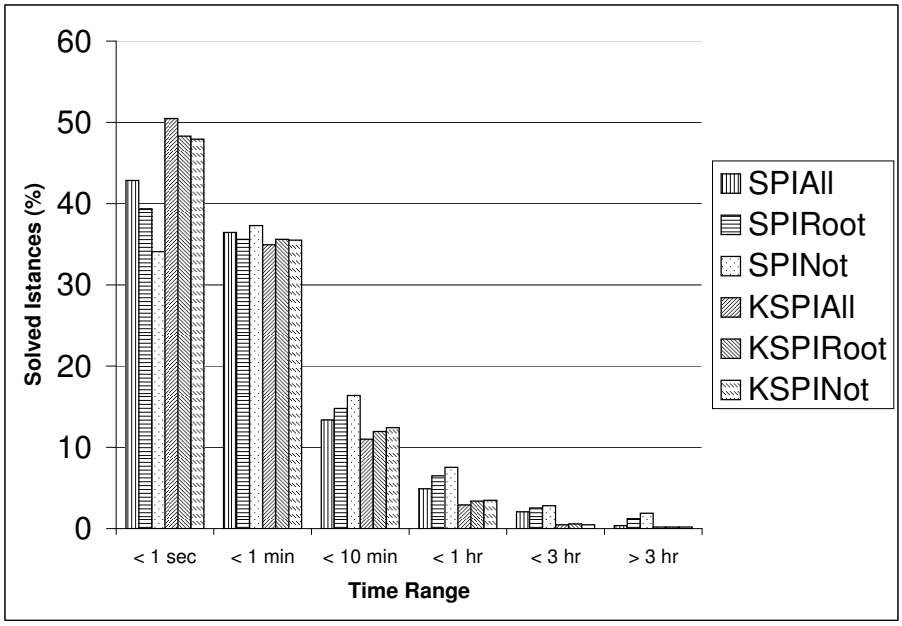

Figure 2: Frequency analysis

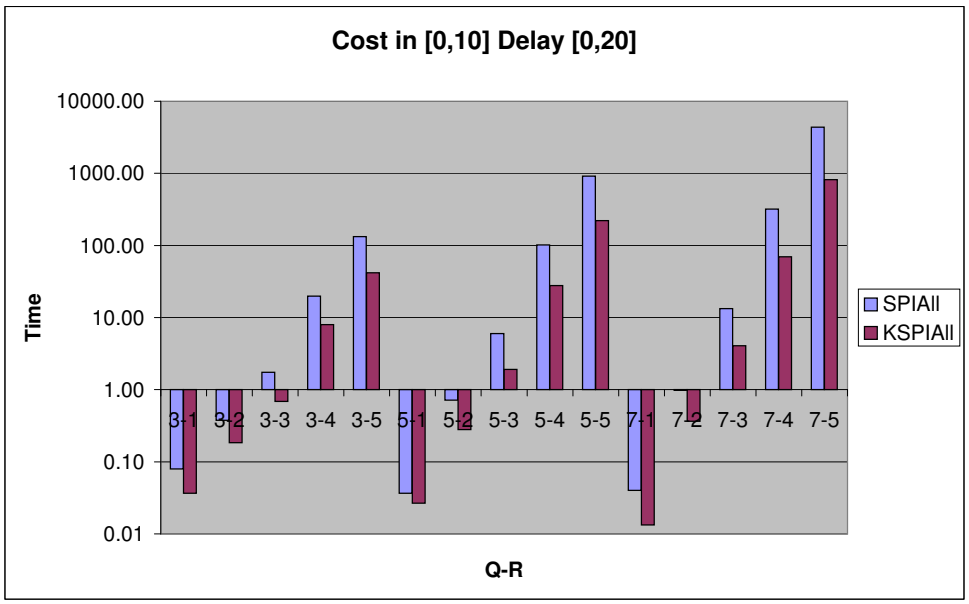

Figure 3: Q-R scalability on 10x10 grids

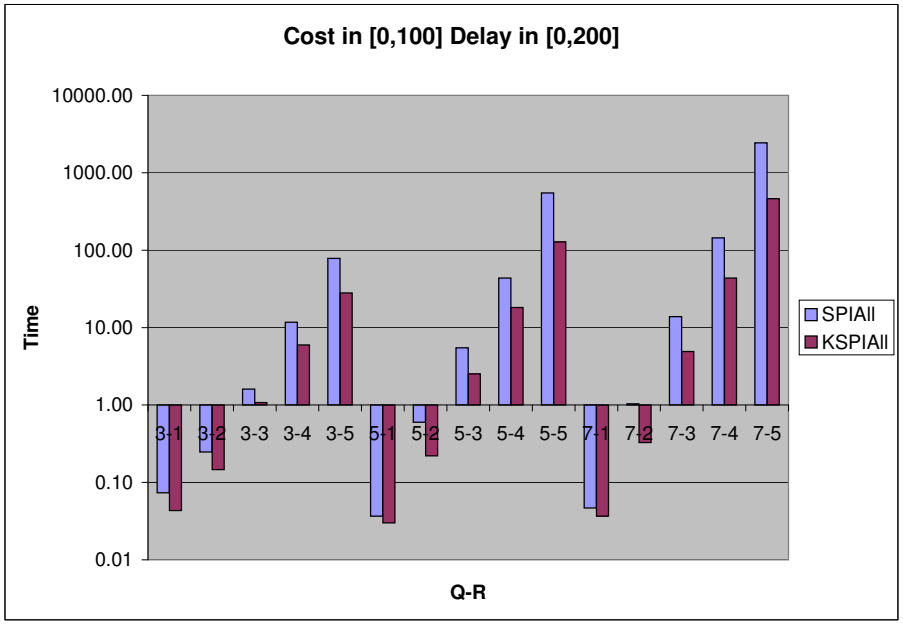

Figure 4: Q-R scalability on 10x10 grids

http://www.kent.ac.uk/kbs/research-information/index.htm