Military Operations Research Society

Implementing Defender-Attacker-Operator (DAO) Optimization, with a Military Force Deployment Example

Author(s): Gerald G. Brown and Emily M. Craparo

Source: *Military Operations Research*, 2023, Vol. 28, No. 3 (2023), pp. 5-30

Published by: Military Operations Research Society

Stable URL: https://www.jstor.org/stable/10.2307/27254913

JSTOR is a not-for-profit service that helps scholars, researchers, and students discover, use, and build upon a wide range of content in a trusted digital archive. We use information technology and tools to increase productivity and facilitate new forms of scholarship. For more information about JSTOR, please contact support@jstor.org.

Your use of the JSTOR archive indicates your acceptance of the Terms & Conditions of Use, available at https://about.jstor.org/terms

Military Operations Research Society is collaborating with JSTOR to digitize, preserve and extend access to *Military Operations Research*

This content downloaded from 205.155.65.226 on Tue, 14 Nov 2023 16:34:43 +00:00
All use subject to https://about.jstor.org/terms

RESEARCH ARTICLE

Implementing Defender-Attacker-Operator (DAO) Optimization, with a Military Force Deployment Example

Gerald G. Brown[1] and Emily M. Craparo[2]

[1]Naval Postgraduate School, ggbrown@nps.edu
[2]Naval Postgraduate School, emcrapar@nps.edu

ABSTRACT

We show how to formulate and solve a trilevel defender-attacker-operator (DAO) optimization model for an operational system. Here, integer linear programs may represent limits on courses of action for the system defender, attacker, and operator. The defender acts first and defensive preparations influence following courses of action by the attacker and then the operator. The attacker has full knowledge of prior defender actions and of the following operator's abilities. With this knowledge, the attacker seeks to maximize damage regardless of what the operator can do subsequently to respond. Finally, the operator moves to minimize the effects of attacks on the system. We first show how to solve an attacker-operator (AO) model, then how to use this as a submodel to solve a DAO model. A numerical example illustrates planning defense and operation of a U.S. military force deployment under risk of interdiction.

"Master, go on, and I will follow thee"

—Shakespeare: *As You Like It*

INTRODUCTION

Optimization models describe how a wide variety of systems work and guide the operations of many. Recently, critical infrastructure systems have received increased attention because it has become evident that they are vulnerable to natural disasters and to attacks planned by intelligent adversaries.

To assess vulnerability of a system, we hold that one must understand how that system can be operated. What are the limits on operator resources and courses of action and how can the operator respond to damages to the system?

Optimization is frequently used to model system operation and is ubiquitous in government and industry. In particular, integer linear optimization models offer the ability to capture the details of discrete and continuous decisions and their consequence.

We consider a trilevel defender-attacker-operator (DAO) optimization, which models the actions of three "players."

https://doi.org/10.5711/108259832835
© Copyright 2023, Military Operations Research Society
APPLICATIONS AREAS:
Computational Advances in OR, Linear/Integer Programming, Homeland Security and Defense

OR METHODS:
Infrastructure Analyses

This content downloaded from
205.155.65.226 on Tue, 14 Nov 2023 16:34:43 +00:00
All use subject to https://about.jstor.org/terms

Given an underlying system (e.g., an infrastructure system), the defender first has the opportunity to harden, defend, or otherwise render some components of the system less vulnerable or invulnerable using finite available resources. Then, having seen the defensive preparations, the attacker deploys its own finite resources to attack a limited number of still-vulnerable components. Finally, the operator utilizes the resulting system in the best way possible, given the prior actions of the defender and the attacker. Of note, the three players share the same objective, which evaluates the operator's ability to utilize the system. The defender and the operator wish to influence this objective in the same direction (maximizing or minimizing) while the attacker wishes the opposite. For this reason, DAO models have sometimes been referred to as defender-attacker-defender (DAD) problems in the literature.

DAD models are a natural extension of bilevel defender-attacker interdiction models. Smith and Song (2020) provide a survey of such models and the algorithms used to solve them.

Some of the earliest DAD formulations for infrastructure systems were described by Salmeron et al. (2009) and Brown et al. (2006). Alderson et al. (2011) describe the applicability of DAD models to critical infrastructure systems and provide a method for solving these problems when the operator's model is continuous and convex.

DAD models have since been used in a wide variety of applications, many network-based. For instance, Nicholas and Alderson (2015) use DAD methodology in the context of wireless mesh network design to find jamming-resistant network topologies. Simchi-Levi et al. (2019) use a trilevel model to address the problem of prepositioning medical supplies to protect against bioattacks. By utilizing the affinely adjustable robust counterpart to their problem, they are able to solve supply chain problems with millions of nodes while ensuring optimality under certain conditions.

Often, application-specific considerations drive modifications to the classic DAO framework. For example, Xiang and Wang (2019) consider a DAD model where the attacker's available resources are uncertain. In the context of defending electric power systems, they demonstrate a column-and-constraint generation algorithm. Alassad et al. (2020) use a shared cognition approach to model deceptive tactics at the defender level, resulting in a mixed integer nonlinear master problem which they then linearize and solve with column-and-constraint generation. Likewise, Davarikia et al. (2020) consider defender posturing instead of only component hardening. Conversely, Dahan et al. (2020) consider attacks that are not immediately apparent to the defender, but that can be detected upon inspection. Assuming an intelligent attacker, they formulate and solve a network inspection problem to determine the detection strategy that minimizes the number of required detectors while achieving a target expected detection rate.

Focusing on the case where the objective function is additively separable (i.e., where variables from among the various stages interact only additively in the objective function), Bolusani and Ralphs (2022) discuss a generalization of the Benders approach to an arbitrary number of stages via projection and dual bounding functions. Baggio et al. (2021) consider a trilevel problem to model cascade failures or viral attacks on nodes in a network. Notably, their third stage allows nodes to be "protected," limiting the spread of attacks conducted in the second stage.

In the context of power grid operations, Yuan et al. (2014) describe a column-and-constraint generation method for solving the DAD problem. Yuan and Zeng (2020) then consider an extension of the usual DAD model by allowing line switching in the inner (operator) level. To solve the resulting problem, which has integer decision variables at every level, they develop an exact algorithm based on nested column-and-constraint generation. Thompson and Tran (2020) use a DAD model to study vulnerabilities in the U.S. air transportation network. They identify improved intermodal linkages between air and ground networks as a possible means to alleviate disruptions to network operations.

Due to the computational challenges of DAD problems, many works have focused on simplifying heuristics. Wu and Conejo (2017), for instance, formulate a trilevel DAD model to provide insight on how best to protect electrical transmission assets from attack. They use strong duality to merge the two inner layers, resulting in a bilevel model that can be solved with Benders

This content downloaded from
205.155.65.226 on Tue, 14 Nov 2023 16:34:43 +00:00
All use subject to https://about.jstor.org/terms

decomposition. San Martin (2007) formulates a trilevel DAD model aimed at optimally defending an infrastructure system from attack, then develops four decomposition-based algorithms to solve it. Wu et al. (2021) consider the defense of an urban water supply system by formulating a DAO model, which they solve using hydraulic analysis and neighborhood search. Alguacil et al. (2014) formulate DAD model to tackle the problem of electric power grid defense. They solve it by first converting the problem to an equivalent bilevel problem, then solve this bilevel problem using implicit enumeration. Avraamidou and Pistikopoulos (2020) propose an algorithm for solving trilevel optimization problems with integer and continuous variables at each level, where the objective function contains quadratic terms of a particular form. Their algorithm formulates the inner problems as multiparametric programming problems and substitutes the resulting solutions into the higher-level problems. Lin and Bie (2018) study power system resilience using a DAD model and solve it using column constraint generation. Ding et al. (2018) use uncertainty sets to model the attacker, then solve the resulting two-stage robust optimization model using column-and-constraint generation.

Lazzaro (2016) develops an implicit enumeration algorithm for solving DAD problems on network models and develops a parametric programming extension of the DAD for cases where the defense budget is uncertain and a family of "nested" defenses is required. Ghorbani-Renani et al. (2021) consider both vulnerability reduction and recoverability enhancement in their DAD model and use a clustering technique to efficiently identify candidate components for protection. Fakhry et al. (2021) explore three heuristic approaches to solve trilevel optimization problems and illustrate their approaches on an electric power grid application.

Lozano and Smith (2017) consider DAD problems where only binary variables are present in the first and second stages, and where the third stage may take any form. Their approach utilizes a sample of third-stage solutions to reduce computational complexity and is particularly effective when a diverse set of solutions to the third-stage problem can be efficiently created (e.g., via a heuristic).

The remainder of this paper explicitly describes a general solution procedure for DAO linear integer optimization problems that can involve integer decision variables at all levels (defender, attacker, and operator) while using an off-the-shelf mathematical programming language and solver. We provide detailed guidance and practical advice for solving difficult problems and addressing practical issues. We illustrate with a realistic military logistics model.

DEFENDER-ATTACKER-OPERATOR (DAO) MODEL FORMULATION

We now describe an exact method for solving the DAO problem. We represent the system operation with the operator model (O):

$$z_{\mathrm{O}}^{*} = \min_{\mathbf{y} \in Y} f(\mathbf{y}). \tag{1}$$

The operator actions (decision variables) $\mathbf{y}$ must be feasible, which we represent by requiring that they belong to the non-empty domain Y that expresses constraints on actions and interactions among actions. The objective f measures operating costs and constraints describing Y can be expressed as a linear function of the decision variables, and, importantly, integer linear optimization admits discrete operator decisions, as well as continuous controls.

To introduce the attacker, we state an attacker-operator (AO) model:

$$z_{\mathrm{AO}}^{*} = \max_{\mathbf{x} \in X} \min_{\mathbf{y} \in Y(\mathbf{x})} f(\mathbf{x}, \mathbf{y}). \tag{2}$$

Attacker actions $\mathbf{x}$ must satisfy a non-empty constraint domain X that can include discrete decisions and may have influence on the operator's domain, as represented by $\mathbf{y} \in Y(\mathbf{x})$. The attacker shares the same objective as the operator, but with opposite intent to influence it. Again, to simplify exposition, we assume that there must be a feasible attacker course of action, and that the

This content downloaded from
205.155.65.226 on Tue, 14 Nov 2023 16:34:43 +00:00
All use subject to https://about.jstor.org/terms

attacker may increase the costs for the operator, but cannot render the operator model infeasible. Accordingly, we change the notation just a bit:

$$z^*_{\mathrm{AO}} = \max_{\mathbf{x}\in X} \min_{\mathbf{y}\in Y} f(\mathbf{x}, \mathbf{y}). \tag{3}$$

Even lacking the ability to render the operator infeasible, the attacker can make operator actions so expensive that they would be avoided if at all possible.

Finally, we introduce a defender, forming the full defender-attacker-operator (DAO) model:

$$z^*_{\mathrm{DAO}} = \min_{\mathbf{w}\in W} \max_{\mathbf{x}\in X} \min_{\mathbf{y}\in Y(\mathbf{w})} f(\mathbf{w}, \mathbf{x}, \mathbf{y}). \tag{4}$$

Defender actions $\mathbf{w}$ must satisfy a non-empty domain W that can include discrete decisions and, because the defender seeks to make things better for the operator, we allow defense actions that influence the domain of feasible operator actions, i.e., $\mathbf{y} \in Y(\mathbf{w})$.

Solving an Attacker-Operator (AO) Model with Benders Decomposition

Alderson et al. (2011) provide an outline of techniques available to solve this problem. We now amplify these solution techniques and include several key details missing from that work, including additional suggestions for convergence tolerances and anticycling techniques.

If we fix operator decisions to $\hat{\mathbf{y}}$ (such a caret hat subsequently denotes a fixed variable throughout), we can find the best attacks by solving pure attacker model (A):

$$z^*_{\mathrm{A\hat{O}}} = z^*_{\mathrm{A}} = \max_{\mathbf{x}\in X} f(\mathbf{x}, \hat{\mathbf{y}}). \tag{5}$$

Similarly, if we fix attacker decisions $\hat{\mathbf{x}}$, we can solve a pure operator model (O):

$$z^*_{\mathrm{\hat{A}O}} = z^*_{\mathrm{O}} = \min_{\mathbf{y}\in Y} f(\hat{\mathbf{x}}, \mathbf{y}). \tag{6}$$

When the cost-maximizing attacker (A) and cost-minimizing operator (O) models are each *integer* linear programs, unless they are unimodular network flow models, we cannot employ duality to convert either model to its dual, and simply nest the resulting optimization monolith (see, e.g., Owen, 1969). Rather, we employ exhaustive enumeration or solve indirectly by decomposition.

This suggests solving by alternating play, fixing attacker actions $\hat{\mathbf{x}}$ and solving operator model O, then observing the operator's response $\hat{\mathbf{y}}$ and re-optimizing attacker model A, and so forth. Such an approach forms the basis of Benders decomposition (e.g., Geoffrion 1972). Benders decomposition alternates between an operator's subproblem and an attacker's master problem, where for each iteration a "cut" is added to prevent repeating attacker solutions. Table 1 shows an attacker master problem (A) arising after L operator responses to prior trial attacks. One can think of this as the attacker probing the operator, observing how the operator responds, and learning.

Unfortunately, the number of attacks can be exponentially large, so we need a way to decide when we have discovered a good enough set of attacks. We know that z^*_{A} for any subset of all attacks is a relaxation of (A) if all attacks were known, and if all attacks were known, we would have the solution z^*_{AO}. Thus, z^*_{A} is an upper bound on the achievable operating costs resulting from attacks and operator responses. Symmetrically, we know that for a fixed set of attacks, the operator response is a restriction, so keeping track of the best of the restricted operator responses gives us a

Table 1. Attacker master problem (A).

$z^*_{\mathrm{A}} = \max\limits_{z,\mathbf{x}\in X} z$	
s.t.	$z \le f(\hat{\mathbf{w}}, \mathbf{x}, \hat{\mathbf{y}}^l) \quad \forall l = 1, \ldots, L.$

Note: Given some fixed defense $\hat{\mathbf{w}}$ and a fixed set of L prior operator responses, this seeks attacks at least as effective as those already attempted.

This content downloaded from
205.155.65.226 on Tue, 14 Nov 2023 16:34:43 +00:00
All use subject to https://about.jstor.org/terms

Table 2. Attacker-operator (AO) optimization with Benders decomposition solving attacker master (A) and operator subproblems (O) *exactly* optimally.

```
AO 1: $z_{\mathrm{AO}}^{\mathrm{LB}} = -\infty,\ z_{\mathrm{AO}}^{\mathrm{UB}} = +\infty$, define relative tolerance $\varepsilon_{\mathrm{AO}} > 0, \hat{\mathbf{x}} = 0$
AO 2: Fix $\hat{\mathbf{x}}$ and solve Operator (sub-)problem (O) for $\hat{\mathbf{y}}$ with objective value $z_{\mathrm{O}}^{*}$
    If $z_{\mathrm{AO}}^{\mathrm{LB}} < z_{\mathrm{O}}^{*}$, then $z_{\mathrm{AO}}^{\mathrm{LB}} = z_{\mathrm{O}}^{*}$ and $\mathbf{x}_{\mathrm{AO}}^{*} = \hat{\mathbf{x}}$
    If $z_{\mathrm{AO}}^{\mathrm{UB}} - z_{\mathrm{AO}}^{\mathrm{LB}} \le \varepsilon_{\mathrm{AO}}\left(|z_{\mathrm{AO}}^{\mathrm{LB}}| + \delta\right)$, go to step AO 5
AO 3: Fix $\hat{\mathbf{y}}$ and solve Attacker (master) problem (A) in Table 1 for $\hat{\mathbf{x}}$
      with objective value $z_{\mathrm{A}}^{*}$
AO 4: $z_{\mathrm{AO}}^{\mathrm{UB}} = z_{\mathrm{A}}^{*}$
    If $z_{\mathrm{AO}}^{\mathrm{UB}} - z_{\mathrm{AO}}^{\mathrm{LB}} > \varepsilon_{\mathrm{AO}}\left(|z_{\mathrm{AO}}^{\mathrm{LB}}| + \delta\right)$, go to step AO 2
    If desired, the optimal operator solution can be recovered by fixing $\hat{\mathbf{x}} = \mathbf{x}_{\mathrm{AO}}^{*}$ and solving operator
    (sub-)problem (O) for $\mathbf{y}_{\mathrm{AO}}^{*}$
AO 5: $\mathbf{x}_{\mathrm{AO}}^{*}$ is $\varepsilon_{\mathrm{AO}}$-optimal attacker solution, and $\mathbf{y}_{\mathrm{AO}}^{*}$ is the responding optimal operator solution.
```

Note: The relative decomposition gaps employ a small constant $\delta \approx 10^{-10}$ in the fashion of many optimization packages.

lower bound on achievable operating costs z_{AO}^{*}. Table 2 organizes a sequence of alternating model solutions that improves these bounds and ultimately discovers an optimal attack $\mathbf{x}_{\mathrm{AO}}^{*}$ and responding operator plan $\mathbf{y}_{\mathrm{AO}}^{*}$. We call this the "best worst-case solution." Neither opponent can do any better.

We can also invert the decomposition, and express an operator master problem (OM), but we do not pursue that here.

As a practical matter, the "solve" in this abstract decomposition algorithm must often be relaxed to "approximately solve"; exact solutions to integer linear optimization models may take too long. In this case, these integer linear programs are only approximately solved with some

Table 3. Nested inner attacker-operator (AO) optimization with Benders decomposition solving *K*th attacker master (A) and operator subproblems (O) *approximately*. Initially, $\mathbf{x}_{\mathrm{AO}}^{*}$ is set to an admissible value, e.g., zero.

```
AO 1: Outer (Defender) iteration $K$ is an input and constant here.
    $z_{\mathrm{AO}}^{\mathrm{LB}} = -\infty, z_{\mathrm{AO}}^{\mathrm{UB}} = +\infty$, define $\varepsilon_{\mathrm{A}} > 0, \varepsilon_{\mathrm{O}} > 0, \varepsilon_{\mathrm{AO}} > 0$, set $\hat{\mathbf{x}}^{K} = \mathbf{x}_{\mathrm{AO}}^{*}$,
    set inner iteration counter $L$=0, iteration limit $L_{\mathrm{AO}}^{\max}$
AO 2: $L = L + 1$
    Fix $\hat{\mathbf{x}}^{K}$ and solve Operator (sub-)problem (O) for $\hat{\mathbf{y}}^{K}$ with objective value $z_{\mathrm{O}}$ and integrality gap
    $z_{\mathrm{O}} - z_{\mathrm{O}}^{\mathrm{LB}} \le \varepsilon_{\mathrm{O}}\left(|z_{\mathrm{O}}^{\mathrm{LB}}| + \delta\right)$
    If $z_{\mathrm{AO}}^{\mathrm{LB}} < z_{\mathrm{O}}^{\mathrm{LB}}$, then $z_{\mathrm{AO}}^{\mathrm{LB}} = z_{\mathrm{O}}^{\mathrm{LB}}$ and $\mathbf{x}_{\mathrm{AO}}^{*} = \hat{\mathbf{x}}^{K}$
    If $z_{\mathrm{AO}}^{\mathrm{UB}} - z_{\mathrm{AO}}^{\mathrm{LB}} \le \varepsilon_{\mathrm{AO}}\left(|z_{\mathrm{AO}}^{\mathrm{LB}}| + \delta\right)$ or $L \ge L_{\mathrm{AO}}^{\max}$,
      go to step AO 5
AO 3: Fix $\hat{\mathbf{y}}^{K}$ and solve Attacker (master) problem (A) for $\hat{\mathbf{x}}^{K}$ with objective value $z_{\mathrm{A}}$ and integrality gap
    $z_{\mathrm{A}}^{\mathrm{UB}} - z_{\mathrm{A}} \le \varepsilon_{\mathrm{A}}(|z_{\mathrm{A}}| + \delta)$
    If A_infeasible, $z_{\mathrm{AO}}^{\mathrm{UB}} = z_{\mathrm{AO}}^{\mathrm{LB}}$
      go to step AO 5
AO 4: If $z_{\mathrm{AO}}^{\mathrm{UB}} > z_{\mathrm{A}}^{\mathrm{UB}}$, then $z_{\mathrm{AO}}^{\mathrm{UB}} = z_{\mathrm{A}}^{\mathrm{UB}}$
    If $z_{\mathrm{AO}}^{\mathrm{UB}} - z_{\mathrm{AO}}^{\mathrm{LB}} > \varepsilon_{\mathrm{AO}}\left(|z_{\mathrm{AO}}^{\mathrm{LB}}| + \delta\right)$ and $L < L_{\mathrm{AO}}^{\max}$,
      go to step AO 2
AO 5: $relgap_{\mathrm{AO}} = \left(z_{\mathrm{AO}}^{\mathrm{UB}} - z_{\mathrm{AO}}^{\mathrm{LB}}\right) / \left(|z_{\mathrm{AO}}^{\mathrm{LB}}| + \delta\right)$
    $\mathbf{x}_{\mathrm{AO}}^{*}$ is a "$relgap_{\mathrm{AO}}$-optimal" attacker solution
```

Note: The method terminates when the desired decomposition gap $\varepsilon_{\mathrm{AO}}$ is achieved, or after $L_{\mathrm{AO}}^{\max}$ iterations, or if A_*infeasible*.

This content downloaded from
205.155.65.226 on Tue, 14 Nov 2023 16:34:43 +00:00
All use subject to https://about.jstor.org/terms

permitted integrality gap (i.e., the difference between the objective value of the best integer solution found and a bound on the best possible objective value). We denote these "intervals of uncertainty" for the operator subproblem as $\left[z_{\mathrm{AO}}^{\mathrm{LB}}, z_{\mathrm{O}}\right]$ and for the attacker master problem as $\left[z_{\mathrm{A}}, z_{\mathrm{A}}^{\mathrm{UB}}\right]$, interpreted above as $\left[z_{\mathrm{AO}}^{\mathrm{LB}} \leq z_{\mathrm{AO}}^{*} \leq z_{\mathrm{A}}^{\mathrm{UB}}\right]$. Note that *the bounds on objective values, rather than the objective values actually attained, must be used in the decomposition*. Table 3 summarizes the modified decomposition, anticipating that this will need to be solved repeatedly for a DAO problem and introducing the current outer (defender) iteration K accordingly.

The decomposition solution of (AO) illustrated in Table 3 assumes that it will be invoked initially with $K = 1$ and the set $\hat{X}^{1}$ empty. As we will see, an outer optimization depends on a record of each (AO) solution $\hat{\mathbf{x}}^{K}$ and $\hat{\mathbf{y}}^{K}$.

Even though the successive attacker master problems (A) are restrictions, the solution values achieved with an interval of uncertainty may not exhibit monotonic objective improvements. Thus, the bound improvement in step AO 4 is by test, rather than unconditional (as it is in step AO 4 in Table 2).

When solving attacker problems approximately, it is possible to revisit previously obtained solutions with no improvement in bounds, *cycling*. To prevent cycling and ensure that a new solution is obtained by each iteration, one can introduce a solution elimination constraint (SEC), described in the next section. With these constraints comes the possibility that either the defender's or the attacker's problem may be infeasible. Fortunately, infeasibility implies that all possible solutions have been enumerated, and that the best solution considered so far is, in fact, optimal.

One can store the operator solution $\hat{\mathbf{y}}$ associated with each new incumbent $\mathbf{x}_{\mathbf{AO}}^{*}$ in step AO 2 during every iteration. However, the operator solution is typically much (usually very much) larger than the attacker one, so it is usually faster to just store the best attacker solution and defer recovering the associated operator response $\mathbf{y}_{\mathbf{AO}}^{*}$ until the end of the decomposition by solving the operator problem (O) one last time with $\hat{\mathbf{x}}^{K} = \mathbf{x}_{\mathbf{AO}}^{*}$. This also permits recovering an operator response with a smaller interval of uncertainty (i.e., with better resolution and fidelity), given this particular operator response likely will attract most attention and analysis.

Solving a Trilevel Defender-Attacker-Operator (DAO) Optimization with Nested Benders Decomposition

It is tempting to use a solution from (AO) to reveal preferred attacks and responding operator actions, then use this to develop defense courses of action. Unfortunately, common-sense rules of thumb and intuition about what to defend do not work in general (Alderson et al., 2013) and the only reliable alternative in our experience is the modeling that follows.

Recall the form above of z_{DAO}^{*} in (4), where defensive actions are governed by constraints in the domain $\mathbf{w} \in W$ and subsequent operator actions may be influenced by these defensive actions, as seen in the domain of operator actions $\mathbf{y} \in Y(\mathbf{w})$. Although we could let defender decisions restrict the resulting domain of admissible attacker courses of action, we advise against this. Similarly, we could let the attacker actions influence the resulting domain of operator ones. Again, we advise against this. Our motive is to avoid the possibility of domain restrictions inducing infeasibility for either the attacker or the operator models that can represent the functions of complex systems. This would complicate our solution methods and analysis, where an infeasible solution signals exhaustive enumeration and thus optimality. We only allow defender preparations to influence the objective function for the attacker. As with the (AO) model, attacker actions may only influence the objective function for the operator. That is the reason for the domains expressed above as they are in the terse definition of z_{DAO}^{*}.

Alderson et al. (2013) show a simple, but complete, concrete DAD formulation. We depart from this DAD nomenclature by distinguishing between the initial defender and subsequent operator as distinct entities by our adoption of a DAO formulation. The operator model there is a multicommodity flow problem, attacks increase operator flow costs, and defenses that influence attacker costs appear in a relatively short list of defensive options, one of which must be selected. There is a complete set of operator flows for each defense option, and each defense option can influence the capacity of flows for the

This content downloaded from
205.155.65.226 on Tue, 14 Nov 2023 16:34:43 +00:00
All use subject to https://about.jstor.org/terms

Table 4. Defender-operator master problem (D) for (DAO) after K attacks.

$$z^*\left(\hat{X}^K\right) = \min_{\substack{z,\mathbf{w}\in W \\ \mathbf{y}^k\in Y(\mathbf{w})}} z$$

$$\text{s.t.} \quad z \geq f\left(\mathbf{w}, \hat{\mathbf{x}}^k, \mathbf{y}^k\right) \qquad \forall k = 1, \ldots, K$$

Note: Given a set of K prior attacks, this seeks a good set of defenses, while observing how the operator would have responded to each attack with this improved set of defenses. Our sole interest is in the optimal defense solution $\mathbf{w}^*_{\text{DAO}}$.

operator. For simplicity, clarity, and to ease model verification, there is one "do nothing" defense option that has no influence at all on costs or capacities for either attacker or operator.

Viewed this way, our (AO) solution procedure in Table 3 solves $z^*_{\text{AO}} = \max_{\mathbf{x}\in X} \min_{\mathbf{y}\in Y(\hat{\mathbf{w}})} f(\hat{\mathbf{w}}; \mathbf{x}; \mathbf{y})$.

The inner attacker-operator optimization seeks attacks leading to the worst minimum operator cost.

We will again form a Benders decomposition of the (DAO) problem by solving a defender-operator model (D) for every attack plan observed in K attacks.

Given a set of K attacks, the defender-operator master problem seeks a defense that minimizes the maximum cost of the operator response to any attack.

Table 5. Defender-attacker-operator (DAO) optimization with nested Benders decomposition.

```
DAO 1: $z^{\text{LB}}_{\text{DAO}} = -\infty$, $z^{\text{UB}}_{\text{DAO}} = +\infty$,
        define relative optimality tolerances $\varepsilon_{\text{AO}} > 0$, $\varepsilon_{\text{DAO}} \geq \varepsilon_{\text{AO}} > 0$,
        fix initial defense, e.g., $\widehat{\mathbf{w}} = \mathbf{0}$, and initial attacks $\mathbf{x}^*_{\text{AO}} = \mathbf{0}$
        Set outer iteration counter $K = 0$ with limit $K^{\max}_{\text{DAO}}$
DAO 2: $K = K + 1$
DAO 3: Solve DAO$(\widehat{\mathbf{w}}, \mathbf{x}, \mathbf{y})$ problem (AO) as shown in Table 3, giving $z^{\text{UB}}_{\text{AO}}$;
        $\widehat{\mathbf{w}}$ does not appear in (AO), but has influenced its objective function;
        Add candidate $\widehat{\mathbf{x}}^K$ to attacker solution set: $\widehat{X}^K = \widehat{X}^{K-1} \cup \widehat{\mathbf{x}}^K$
        If A_infeasible, $\mathbf{x}^*_{\text{AO}} = \mathbf{x}^*_{\text{DAO}}$
DAO 4: If $z^{\text{UB}}_{\text{DAO}} > z^{\text{UB}}_{\text{AO}}$, then $z^{\text{UB}}_{\text{DAO}} = z^{\text{UB}}_{\text{AO}}$,
        $\mathbf{w}^*_{\text{DAO}} = \widehat{\mathbf{w}}$, $\mathbf{x}^*_{\text{DAO}} = \mathbf{x}^*_{\text{AO}}$
DAO 5: If A_infeasible or $z^{\text{UB}}_{\text{DAO}} - z^{\text{LB}}_{\text{DAO}} \leq \varepsilon_{\text{DAO}}\left(|z^{\text{LB}}_{\text{DAO}}| + \delta\right)$ or $K \geq K^{\max}_{\text{DAO}}$,
        go to step DAO 7
DAO 6: Given attack plans $\widehat{\mathbf{x}}^K, k = 1, \ldots, K$, free defense variables $\mathbf{w}$,
        solve (D) as shown in Table 4 with optimality tolerance $\varepsilon_{\text{DAO}}$ giving $z^{\text{LB}}_{\text{D}}$
        If D_infeasible go to step DAO 7
        If $z^{\text{LB}}_{\text{DAO}} < z^{\text{LB}}_{\text{D}}$, $z^{\text{LB}}_{\text{DAO}} = z^{\text{LB}}_{\text{D}}$
        If $z^{\text{UB}}_{\text{DAO}} - z^{\text{LB}}_{\text{DAO}} > \varepsilon_{\text{DAO}}\left(|z^{\text{LB}}_{\text{DAO}}| + \delta\right)$ and $K < K^{\max}_{\text{DAO}}$,
        $\widehat{\mathbf{w}} = \mathbf{w}^*$, go to step DAO 2
DAO 7: $relgap_{\text{DAO}} = \left(z^{\text{UB}}_{\text{DAO}} - z^{\text{LB}}_{\text{DAO}}\right) / \left(|z^{\text{LB}}_{\text{DAO}}| + \delta\right)$.
        Defense plan $\mathbf{w}^*_{\text{DAO}}$ and attack plan $\mathbf{x}^*_{\text{DAO}}$ are "$relgap_{\text{DAO}}$-optimal".
        Recover a corresponding operator response by fixing these defense and attack plans and solving
        the Operator problem (O).
        END.
```

Note: This algorithm depends in its iterations of step DAO 2 on solution of a Benders decomposition of (AO) problem in Table 3 and solution of a master problem (D) in step DAO 5 shown in Table 4.

This content downloaded from
205.155.65.226 on Tue, 14 Nov 2023 16:34:43 +00:00
All use subject to https://about.jstor.org/terms

Table 5 displays a complete nested Benders decomposition procedure, incorporating the defender-operator master problem (D) from Table 4. The outer decomposition step DAO 4 uses the upper bound of AO $z_{\text{AO}}^{\text{UB}}$ to improve the global upper bound, and step DAO 6 considers the lower bound from (D) z_{D}^{LB} to improve the global lower bound.

Dealing with cycles. Even though successive defender iterations are restrictions, cycles (repeated solutions) are possible because we are not solving these integer linear programs exactly, but instead within some stated relative integer convergence tolerance. If not an outright cycle, an intermediate stall may yield no improvement for some iterations. To deal with this, it proves useful to be able to preclude reappearance of a binary attacker action $\hat{\mathbf{x}}$. Define the solution elimination constraint (SEC) function as follows:

$$add\_SEC(\hat{\mathbf{x}}): \sum_{j|\hat{x}_j=0} x_j + \sum_{j|\hat{x}_j=1} (1 - x_j) \geq 1. \tag{7}$$

These usually inflict so little overhead they can be added with every new solution. However, one can apply more careful controls, discarding these SECs when a solution with a better objective value is found and resuming addition of them if either the objective function is not improved, or more specifically when a binary solution reappears. To detect this latter case, one can avoid relatively expensive tests to compare each new binary solution to its predecessor candidates for repetition by computing for each solution a numeric hash score: If hash scores differ between binary solutions, they are different. If the hash scores match, a detailed comparison resolves things.

By construction, an infeasibility unambiguously signals that our SECs have ruled out all alternate solutions, so the one we have (i.e., $\mathbf{x}_{\text{AO}}^{*}$) is optimal.

AN EXAMPLE PROBLEM

We demonstrate our solution procedure on a time-phased force deployment data (TPFDD) plan (e.g., Joint Chiefs of Staff, 2017a, 2017b, 2018, 2019) where military forces (personnel, materiel, ammunition, and equipment) are mobilized and conveyed through a logistics network to one or more destinations (see Figure 1). The "operator" manages the multimodal transport system to try to arrange destination arrivals in desired time windows. Early arrivals may be vulnerable, and late arrivals impede planned military operations. An intelligent adversary "attacker" has full knowledge of the logistics system and its operation. The attacker can anticipate deployment plans and act to interfere with a limited number of components of the deployment network, delaying or otherwise interfering with operations. The "defender" has ability to defend a limited number of network components from attack. The problem is to find the best worst-case deployment, defending what we can and deploying as best as possible anticipating optimal attacks evading our defenses.

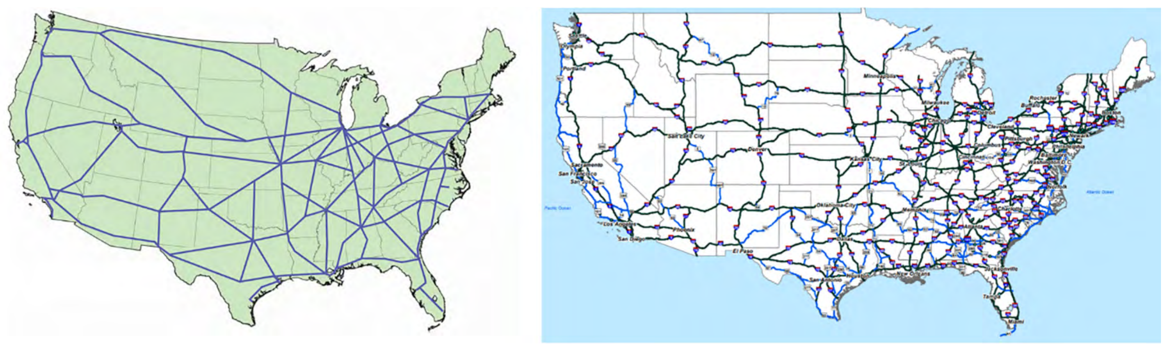

Figure 1. The U.S. strategic rail system (STRACNET) (left, GSO, 2012; U.S. Army, 2018) extends 58,000 km and the U.S. strategic highway system STRAHNET (right, U.S. Army, 2013) 101,000 km. These networks connect about 120 military installations with each other and with airports and seaports.

This content downloaded from
205.155.65.226 on Tue, 14 Nov 2023 16:34:43 +00:00
All use subject to https://about.jstor.org/terms

DAO algorithm index use (denoted by superscripts in the formulation below) [$\sim$cardinality]:

$\ell = 1,2,\ldots,L$	Inner decomposition (attacker-operator) iteration [$\sim$20]
$k = 1,2,\ldots,K$	Outer decomposition (defender) iteration [$\sim$20]

Operator model index use (denoted by subscripts) [$\sim$cardinality]:

$n \in N$	Geographic nodes (an ordinal set, alias nx, $n1$, $n2$) [$\sim$1,000]
$m \in M$	Mode of transport (e.g., ROAD, RAIL, AIR, SHIP, alias $m1$, $m2$) [4]
$p \in P \subseteq N$	Ports of embarkation [$\sim$20]
$p \in POE_{nx} \subseteq P$	Ports of embarkation for shipments originating at node nx [$\sim$2]
$d \in D \subseteq N$	Destination nodes [$\sim$2]
$u \in U$	Units deploying (e.g., Brigade Combat Teams [$\sim$50])
$l \in L$	Lines (e.g., PAX, EQUIP, AMMO, AIRCRAFT [$\sim$4]
$nx_{u,l}$	Origin node location nx of unit u, line l (each unit may have each of its lines pre-positioned at advantageous locations)
d_u	Unit u has destination node d

Operator model index tuples:

$(u,l) \in UL$	Unit u uses line l.
$(l,m) \in LM$	Line l can use transport mode m.
$(nx,m) \in LOAD$	Node nx can load mode m.
$(n1,n2,m) \in EDGE$	$(\lvert n1\rvert<\lvert n2\rvert)$ Undirected edge.
$(n1,n2,m) \in ARCS$	Directed arc from node $n1$ to $n2$ via mode m. $\lvert n1\rvert < \lvert n2\rvert \Rightarrow (n1,n2,m) \in EDGE$, else $(n2,n1,m) \in EDGE$
$(n,m1,m2) \in CHNG$	Unit u line l node n change from mode $m1$ to $m2$

Given data [units]:

$supply_{u,l}$	Supply of unit u line l [STONS]
$load\_rate_{l,nx,m}$, $add\_load\_rate_{l,nx,m}$ $(nx,m) \in LOAD \wedge (l,m) \in LM$	Rate that line l at node nx can be loaded on mode m if not attacked or defended, and additional rate if attacked and not defended [time/STON]
$load\_time_{u,l,nx,m}$, $load\_add\_time_{u,l,nx,m}$, $(u,l) \in UL, (nx,m) \in LOAD, (l,m) \in LM$	Using load rates, time required to load $supply_{u,l}$ for unit u line l at node nx on mode m if not attacked or defended, and additional time if attacked and not defended [time]
$edge\_time_{n1,n2,m}$, $edge\_add\_time_{n1,n2,m}$, $(n1,n2,m) \in EDGE$	Time between nodes $n1$ and $n2$ via mode m if not attacked or defended, and additional time if attacked and not defended [time]
$chng\_rate_{l,n,m1,m2}$, $chng\_add\_rate_{l,n,m1,m2}$, $(n,m1,m2) \in CHNG, (l,m1) \in LM, (l,m2) \in LM$	Rate that line l at node n can be changed from mode $m1$ to mode $m2$ if not attacked or defended, and additional rate if attacked and not defended [time/STON]
$chng\_time_{u,l,n,m1,m2}$, $chng\_add\_time_{u,l,n,m1,m2}$, $(u,l) \in UL(n,m1,m2) \in CHNG,$ $(l,m1) \in LM, (l,m2) \in LM$	Using change rates, time required to change $supply_{u,l}$ for unit u line l at node n from mode $m1$ to mode $m2$ if not attacked or defended, and additional time if attacked and not defended [time]

$Wbudget$	Budget limiting number of defenses [defenses]
$Xbudget$	Budget limiting number of attacks [attacks]

This content downloaded from
205.155.65.226 on Tue, 14 Nov 2023 16:34:43 +00:00
All use subject to https://about.jstor.org/terms

Operator decision variables [units] (there can be K copies of each of these):

$YL^k_{u,l,nx,m} \geq 0$	Load unit *u* line *l* at node *nx* for transport on mode *m* [fraction]
$YF^k_{u,l,n1,n2,m} \geq 0$	Flow of unit *u* line *l* from node *n1* to node *n2* via mode *m* [fraction]
$YC^k_{u,l,n,m1,m2} \geq 0$	Change flow of unit *u* line *l* at node *n* from transport mode *m1* to *m2* [fraction]

Attacker decision variables [units] (there can be K copies of each of these):

$XL^k_{nx,m} \in \{0,1\}$	Attack load node *nx*, transport mode *m* [binary]
$XE^k_{n1,n2,m} \in \{0,1\}$	Attack flows between node *n1* and *n2* via mode *m* [binary]
$XC^k_{n,m1,m2} \in \{0,1\}$	Attack flows at node *n* changing between transport mode *m1* and *m2* [binary]

Defender decision variables [units]:

$WL_{nx,m} \in \{0,1\}$	Defend load node *nx*, transport mode *m* [binary]
$WE_{n1,n2,m} \in \{0,1\}$	Defend flows between nodes *n1* and *n2* via mode *m* [binary]
$WC_{n,m1,m2} \in \{0,1\}$	Defend mode changes at node *n* between transport mode *m1* and *m2* [binary]

Overall formulation objective:

$$
\begin{aligned}
f(\mathbf{w},\mathbf{x},\mathbf{y}) = {} & \\
& \sum_{\substack{(u,l) \in UL,\\ (nx_{u,l},m) \in LOAD\\ |(l,m) \in LM}} \left(load\_time_{u,l,nx,m} YL^k_{u,l,nx,m} + load\_add\_time_{u,l,nx,m}(1 - WL_{nx,m}) * XL^k_{nx,m} YL^k_{u,l,nx,m} \right) \\
& + \sum_{\substack{(u,l) \in UL,\\ (n1,n2,m) \in EDGE\\ |(l,m) \in LM}} \Bigg(edge\_time_{n1,n2,m} \left[YF^k_{u,l,n1,n2,m} + YF^k_{u,l,n2,n1,m} \right] \\
& + edge\_add\_time_{n1,n2,m}(1 - WE_{n1,n2,m}) XE^k_{n1,n2,m} \left[YF^k_{u,l,n1,n2,m} + YF^k_{u,l,n2,n1,m} \right] \Bigg) \\
& + \sum_{\substack{(u,l) \in UL,\\ (n,m1,m2) \in CHNG\\ |(l,m1) \in LM \wedge (l,m2) \in LM}} \Big(chng\_time_{u,l,n,m1,m2} YC^k_{u,l,n,m1,m2} + chng\_add\_time_{u,l,n,m1,m2} \\
& \qquad \big(1 - W_{n,m1,m2)}\big) XC^k_{n,m1,m2} YC^k_{u,l,n,m1,m2} \Big)
\end{aligned}
\tag{OBJ}
$$

Discussion

For brevity, defender, attacker, and operator variables appear in the objective function as respective vectors **w**, **x**, and **y**. In use to solve an attacker-operator (AO) problem, **w** defense variables are always fixed, and alternating problems use either attack variables **x** or operator variables **y**, with those of the opponent fixed. Successive inner decomposition solutions of an (AO) problem are indexed by (superscript) $\ell \in L$, outer decompositions solutions of (DAO) are indexed by (superscript) $k \in K$, and alternate between solving for each of k attack variables **x** with defense variables **w** and k operator variables **y** fixed, or the reverse of this. Each outer iteration k of (DAO) requires some set of inner iterations L to solve (AO). In an outer iteration k, we reckon

This content downloaded from
205.155.65.226 on Tue, 14 Nov 2023 16:34:43 +00:00
All use subject to https://about.jstor.org/terms

some new proposed defense **w** and simultaneously adjust all K operator responses to the K prior attacks. Thus, the full detail of operator responses needs to be recorded.

Operator model:

$$\min_{(\mathbf{y}^k)\in Y} f\left(\hat{\mathbf{w}},\hat{\mathbf{x}},\mathbf{y}^k\right) \tag{O1}$$

$$\sum_{\substack{(nx,m)\,\in\, LOAD,\\ |(l,m)\,\in\, L}} YL^k_{u,l,nx,m} = 1 \qquad \forall (u,l)\in UL,\, nx_{u,l} \tag{O2}$$

$$\begin{aligned} & YL^k_{u,l,n,m} | n = nx_{u,l} \wedge (n,m)\in LOAD \wedge (l,m)\in LM + \sum_{\substack{(n,m1,m2)\,\in\, CHNG\\ |(l,m1)\,\in\, LM}} YC^k_{u,l,n,m1,m2} \\ & = YF^k_{u,l,n,n2,m2} \,|(n,n2,m2)\in ARC \qquad \forall (u,l)\in UL,\, n\in N, \\ & \qquad m2\in M|\{l,m2\}\in LM \end{aligned} \tag{O3}$$

$$\begin{aligned} & \sum_{(n1,n,m1)\in ARCS} YF^k_{u,l,n1,n,m1} \\ & = \sum_{\substack{(n,m1,m2)\,\in\, CHNG\\ |\{l,m2\}\,\in\, LM}} YC^k_{u,l,n,m1,m2} \qquad \forall (u,l)\in UL,\, n\in N,\, m1\in M \\ & \qquad |(l,m1)\in LM \wedge n\neq d_u \end{aligned} \tag{O4}$$

$$\sum_{\substack{(l,m)\,\in\, LM,\\ p\,\in\, POE_{nx_{u,l}}}} YF^k_{u,l,p,d,m} = 1 \qquad \forall (u,l)\in UL,\, d_u\in N \tag{O5}$$

Discussion

The objective (O1) expresses operating costs for fixed defense and binary attack variables by choice of operating variables. Each constraint (O2) establishes flow at a load node for a unit and line. Each constraint (O3) balances load and mode change inputs to a node with flows out of that node for a unit and line. Each constraint (O4) balances flows into a node with mode change flows out for a unit and line. Each constraint (O5) ensures that a unit-line exits the model via an admissible port-of-embarkation.

Attacker constraints:

$$\max_{z,\hat{\mathbf{w}},\mathbf{x}\in X,\hat{\mathbf{y}}} z \le f\left(\hat{\mathbf{w}},\mathbf{x},\hat{\mathbf{y}}^\ell\right) \qquad \forall \ell = 1,2,\ldots,L \tag{A1}$$

$$\begin{aligned} & \sum_{(nx,m)\in LOAD} XL^k_{nx,m} \\ & + \sum_{(n1,n2,m)\in EDGE} XE^k_{n1,n2,m} \\ & + \sum_{(n,m1,m2)\in CHNG} XC^k_{n,m1,m2} \le Xbudget \end{aligned} \tag{A2}$$

Discussion

Each objective cut (A1) has fixed defender variables and a record of an operator action observed so far in this AO subproblem (see Table 3). Constraint (A2) limits the number of system components that can be attacked.

It can be useful to condense cuts (A1) as follows:

This content downloaded from
205.155.65.226 on Tue, 14 Nov 2023 16:34:43 +00:00
All use subject to https://about.jstor.org/terms

$$load\_coef^{\ell}_{nx,m} = \sum_{\substack{(u,l) \in UL \\ |(l,m) \in LM}} load\_add\_time_{u,l,nx,m} \widehat{YL}^{k}_{u,l,nx,m}$$

$$load\_cnst^{\ell}_{nx,m} = \sum_{\substack{(u,l) \in UL \\ |(l,m) \in LM}} load\_time_{u,l,nx,m} \widehat{YL}^{k}_{u,l,nx,m}$$

$$\forall \ell \le L, \quad (nx,m) \in LOAD \qquad \text{(A1L)}$$

$$edge\_coef^{\ell}_{n1,n2,m} = \sum_{\substack{(u,l) \in UL \\ |(l,m) \in LM}} edge\_add\_time_{u,l,n1,n2,m} \widehat{YE}^{k}_{u,l,n1,n2,m}$$

$$edge\_cnst^{\ell}_{n1,n2,m} = \sum_{\substack{(u,l) \in UL \\ |(l,m) \in LM}} edge\_time_{u,l,n1,n2,m} \widehat{YE}^{k}_{u,l,n1,n2,m}$$

$$\forall \ell \le L, \quad (n1,n2,m) \in EDGE \qquad \text{(A1E)}$$

$$chng\_coef^{\ell}_{n,m1,m2} = \sum_{\substack{(u,l) \in UL \\ |(l,m1) \in LM \\ \wedge(l,m2) \in LM}} chng\_add\_time_{u,l,n,m1,m2} \widehat{YC}^{k}_{u,l,n,m1,m2}$$

$$chng\_cnst^{\ell}_{n,m1,m2} = \sum_{\substack{(u,l) \in UL \\ |(l,m1) \in LM \\ \wedge(l,m2) \in LM}} chng\_time_{u,l,n,m1,m2} \widehat{YC}^{k}_{u,l,n,m1,m2}$$

$$\forall \ell \le L, \quad (n,m1,m2) \in CHNG \qquad \text{(A1C)}$$

$$\begin{aligned} z \le & \sum_{(nx.m)} load\_coef^{\ell}_{nx,m}\left(1 - \widehat{WL}_{nx,m}\right) XL^{K}_{nx,m} + load\_cnst^{l} \\ & + \sum_{(n1,n2,m) \in EDGE} edge\_coef^{\ell}_{n1,n2,m}\left(1 - \widehat{WE}_{n1,n2,m}\right) XE^{K}_{n1,n2,m} + edge\_cnst^{\ell} \\ & + \sum_{(n,m1,m2) \in EDGE} chng\_coef^{\ell}_{n,m1,m2}\left(1 - \widehat{WC}_{n,m1,m2}\right) XC^{K}_{n,m1,m2} + chng\_cnst^{\ell} \end{aligned}$$

$$\forall \ell \le L \qquad \text{(A1F)}$$

The factoring (A1L), (A1E), and (A1C) of terms appearing in (A1F) emphasizes the operator's use of each component the attacker might target.

Defender constraints:

$$\min_{z, \mathbf{w} \in W, \mathbf{y}^k \in Y(\mathbf{w})} z \ge f\left(\mathbf{w}, \hat{\mathbf{x}}^k, \mathbf{y}^k\right) \qquad \forall k = 1,2,\ldots,K \qquad \text{(D1)}$$

$$\sum_{(nx,m) \in LOAD} WL_{nx,m} + \sum_{(n1,n2,m) \in EDGE} WE_{n1,n2,m} + \sum_{(n,m1,m2) \in CHNG} WC_{n,m1,m2} \le Wbudget \qquad \text{(D2)}$$

Additional auxiliary linearizing decision variables:

This content downloaded from
205.155.65.226 on Tue, 14 Nov 2023 16:34:43 +00:00
All use subject to https://about.jstor.org/terms

$$WYL^k_{u,l,nx,m} \geq 0 \qquad \forall (u,l) \in UL,\ (nx,m) \in LOAD\ |(l,m) \in LM$$

$$WYE^k_{u,l,n1,n2,m} \geq 0 \qquad \forall (u,l) \in UL,\ (n1,n2,m) \in EDGE\ |(l,m) \in LM$$

$$WYC^k_{u,l,n,m1,m2} \geq 0 \qquad \forall (u,l) \in UL,\ (n,m1,m2) \in CHNG\ |\{l,m1\} \in LM \wedge \{l,m2\} \in LM$$

Linearizing constraints using these auxiliary variables:

$$WYL^k_{u,l,nx,m} \leq WL_{nx,m};\quad WYL^k_{u,l,nx,m} \leq YL^k_{u,l,nx,m};\text{ and } WYL^k_{u,l,nx,m} \geq WL_{nx,m} + YL^k_{u,l,nx,m} - 1 \qquad \forall (u,l) \in UL,\ (nx,m) \in LOAD\ |(l,m) \in LM \tag{D3-D5}$$

$$WYE^k_{n1,n2,m} \leq WE_{n1,n2,m};\quad WYE^k_{n1,n2,m} \leq YF^k_{u,l,n1,n2,m} + YF^k_{u,l,n2,n1,m};\text{ and } WYE^k_{n1,n2,m} \geq WE_{n1,n2,m}\left(YF^k_{u,l,n1,n2,m} + YF^k_{u,l,n2,n1,m}\right) - 1 \qquad \forall (u,l) \in UL,\ (nx,m) \in LOAD\ |(l,m) \in LM \tag{D6-D8}$$

$$WYC^k_{u,l,n,m1,m2} \leq WC_{n,m1,m2};\quad WYC^k_{u,l,n,m1,m2} \leq YC^k_{u,l,n,m1,m2};\text{ and } WYC^k_{u,l,n,m1,m2} \geq WC_{n,m1,m2} + YC^k_{u,l,n,m1,m2} - 1 \qquad \forall (u,l) \in UL,\ (n,m1,m2) \in CHNG\ |(l,m1) \in LM \wedge (l,m2) \in LM \tag{D9-D11}$$

$$\sum_{\substack{(nx,m) \in LOAD \\ |\hat{WL}^k_{nx,m} = 0}} WL_{nx,m} + \sum_{\substack{(nx,m) \in LOAD \\ |\widehat{WL}^k_{nx,m} = 1}} (1 - WL_{nx,m}) + \sum_{\substack{(n1,n2,m) \in EDGE \\ |\widehat{WE}^k_{n1,n2,m} = 0}} WE_{n1,n2,m} + \sum_{\substack{(n1,n2,m) \in EDGE \\ |\widehat{WE}^k_{n1,n2,m} = 1}} (1 - WE_{n1,n2,m}) + \sum_{\substack{(n,m1,m2) \in CHNG \\ |\widehat{WC}^k_{n,m1,m2} = 0}} WC_{n,m1,m2} + \sum_{\substack{(n,m1,m2) \in CHNG \\ |\widehat{WC}^k_{n,m1,m2} = 1}} (1 - WC_{n,m1,m2}) \geq 1 \qquad \forall k < K \tag{D12}$$

Discussion

Each defender objective cut in (D1) expresses for a prior attack the cost of a defense and the best the defender can do to minimize damage from that attack. The object is to

This content downloaded from
205.155.65.226 on Tue, 14 Nov 2023 16:34:43 +00:00
All use subject to https://about.jstor.org/terms

minimize the cost of the worst-case attack (see Table 4). Constraints (D3)–(D11) are used to linearize the nonlinear products of binary defense variables $\mathbf{w}$ and nonnegative operator variables $\mathbf{y}$ (see overall objective). This admits linear integer programming, i.e., a linear mixed integer programming (MIP) model. We also solve without these auxiliary features as a quadratically constrained model, i.e., a mixed integer quadratically constrained program (MIQCP) model. Optional solution elimination constraints (D12) can be used to ensure each successive defense solution differs from its predecessors in at least one binary defense detail.

Optimization Formulations

Operator (O) consists of (O1)–(O5) with $\hat{\mathbf{w}}$ and $\hat{\mathbf{x}}$ fixed. Similar to the attacker model, during solutions of (AO) a single copy of (O1)–(O5) is used, but for (D) solutions, all K copies of these are used.

Attacker uses (A1)–(A2), or optionally (A1) can be replaced by the more condensed (A1L), (A1E), (A1C), and (AIF). Each model instance includes a single copy of (A1)–(A2) for (AO) solutions, and K copies of these features accumulated from prior outer iterations $k = 1,2,\ldots,K$ for (D) solutions of (DAO).

Defender employs (D1)–(D11) and (O2)–(O5). Each model instance includes K copies of constraints (O2)–(O5) accumulated from prior outer iterations $k = 1,2,\ldots,K$. Solution elimination constraints (D12) are optional.

Each solution of (AO), the inner decomposition, consists of iterations $\ell = 1,2,\ldots,L$. Each solution of (D) consist of outer iterations $k = 1,2,\ldots,K$.

NUMERICAL EXAMPLES

Our numerical experiments use the Generalized Algebraic Modeling System (GAMS) 31.1.0 (2022), Gurobi 9.1.2rc0 (2022), and CPLEX 12.10.0.0 (IBM 2022). The entire GAMS script has 2,200 lines, exclusive of data tables; of this, about 500 lines are about models, and the rest stethoscope views of the actions of our three players. Total nested decomposition solution time depends on the convergence tolerances and decomposition iteration limits; moderation is a virtue. These results are from a Lenovo P17 portable workstation with 128GB random access memory and a Passmark (2022) rating of 4,552.

We have been loaned an unclassified continental United States transportation dataset used by military deployment planners (see acknowledgements). This consists of 1,229 nodes and 2,077 undirected edges. There are four transport modes: road (with 1,625 edges), rail (288), air (147), and ship (17). The road and rail networks are drawn from the 101K km of the U.S. Strategic Highway Network STRAHNET and the 58K km of the Strategic Rail Network STRACNET (e.g., U.S. Army, 2013, 2018). The network connects 18 load nodes near military installations, 27 airports and 17 seaports. Our planning focus is on domestic U.S. transport, so our vulnerability and defensive interests are limited up to but not beyond United States points of embarkation.

We aggregate shipments by similarity of logistic transport into four lines: PAX (personnel), EQUIP, AMMO, and AIRCRAFT. These lines can use the transport modes in Table 6.

Each line must be prepared for shipment at its source node, and this takes significant time. There are 44 nodes where a transfer between modes can take place. These transitions also take time and are of particular interest. We assume that once PAX are boarded on AIR, we plan no additional mode change for them. Once any line is loaded on a SHIP, no mode change is planned to follow that.

Armageddon Reduction

As part of our data verification, we use a preliminary reduction to ensure connectivity of unit-line sources and destinations, and to identify and remove unusable components. Given the

This content downloaded from
205.155.65.226 on Tue, 14 Nov 2023 16:34:43 +00:00
All use subject to https://about.jstor.org/terms

Table 6. Transport modes for each line. All lines can move by road.

Mode line	PAX	EQUIP	AMMO	AIRCRAFT
ROAD	X	X	X	X
RAIL		X	X	X
AIR	X			X
SHIP		X	X	X

number of times we anticipate having to solve this operator problem, any reduction can be beneficial. We call this the Armageddon Reduction. From each unit-line source, we find a shortest path to its destination. Every component is vulnerable to attack, we so attack (and thus render longer duration for) every component in these paths. We repeat until no additional unattacked component is attacked, then delete all unattacked components.

Five examples follow. Table 7 gives basic problem entity numbers and problem dimensions. Although we would prefer to represent each historic decision variable in a decomposition by merely its value, each is represented via a mathematical programming language by at least a tuple of variable type, lower bound, current value, upper bound, and reduced cost.

Table 8 displays default convergence tolerance and iteration limit controls. Table 9 gives some idea of the convergence trajectory of the nested decompositions on these problems. The instances here have 14 attacks, but only six defenses—these are tough planning problems. Convergence rate is influenced by the relative numbers of attacks and defenses, but surprisingly not by much.

The results in Table 9 are representative of those achieved in hundreds of scenarios. We normally do not require a relative DAO convergence gap of less than 10%, and one can see this is routinely achieved in only a couple outer DAO iterations. All scenarios solve in essentially the same time whether linearized or not with GORUBI MIP for (D) or MIQCP for (DQ). CPLEX solves the

Table 7. TPFD model statistics.

	Instance				
Entities	ONE	EIGHT	TWENTY	THIRTY	FORTY
Units	1	8	20	30	40
Unit origins	1	5	9	9	18
Lines	4	25	62	95	126
Modes	4	4	4	4	4
Nodes before reduction	1,229	1,229	1,229	1,229	1,229
Edges before reduction	2,077	2,077	2,077	2,077	2,077
Nodes after reduction	566	824	954	964	1,044
Edges after reduction	629	966	1,209	1,229	1,389
$(\mathbf{w}, \mathbf{x})$ Binary variables	2,394	3,604	4,360	4,418	4,920
O constraints x000	5	41	117	181	262
O variables x000 ⊗	7	66	200	312	450
A constraints	3+L	3+L	3+L	3+L	3+L
A variables x000 ⊗	1	2	2	2	2
D constraints x000	19+K	171+K	508+K	789+K	1,157+K
D variables x000 ⊗	13	112	336	521	770
DQ constraints x000	5+K	61+K	117+K	181+K	262+K
DQ variables x000 ⊗	8	123	200	314	462

Notes: All units are destined for Europe. Each of these instances has models that can be initially reduced in size, depending on the origin(s) and destination(s) of units and their lines. Note the number of attacker constraints grows (+L) with inner iterations, but the number of attacker variables does not. Note that the number of constraints for defender master problem models (D) or (DQ) grows (+K) with each outer iteration, and that these problems require copies of each of prior K outer iteration solutions (⊗) for attacker and operator. D has linearizing constraints (D3)–(D11), DQ does not. The largest model FORTY involves moving about 160K personnel and 430K STONS of materiel. FORTY uses every unit-origin in our network. Larger instances scale as would be expected from these results.

This content downloaded from
205.155.65.226 on Tue, 14 Nov 2023 16:34:43 +00:00
All use subject to https://about.jstor.org/terms

Table 8. Convergence tolerances and solution controls. The outer decomposition iteration seeks a 10% relative convergence gap.

δ	10^{-10}
ε_{A}	0.0001
ε_{O}	0.000
ε_{AO}	0.050
ε_{DAO}	0.100
L_{AO}^{max}	20
K_{DAO}^{max}	20

linearized scenarios with essentially the same performance as GORUBI, but CPLEX fails to solve any of these MIQCP scenarios, falsely declaring lack of convexity.

Table 10 shows computation times for problem generation and optimization, and between the Armageddon Reduction and the three basic optimization models. Note that optimization takes much less computation than the overhead required to generate these problems and recover their solutions. This has been our experience with general-purpose mathematical programming languages, among which GAMS is a fast competitor. This shows how much performance can be gained by replacing a general-purpose mathematical modeling language by a custom model generation in a computationally efficient programming language. Doing so is tedious and model-specific, but in our experience reduces generation and solution extraction overhead to negligible times. These sequences of decomposition master problems only differ in the constraints added at each iteration, but our (GAMS) program regenerates each entire master problem from scratch. Directly executable models can be restricted with additional constraints (i.e., cuts) without this significant overhead. So, the results here give a worst-case assessment of achievable model response time.

For scenario FORTY, six optimal defenses protect military installation unit-origin load nodes for roads, while the 14 attacks are at 10 other unit-origin load nodes for roads and four rail edges. The optimization advises defending the most important "break-out" unit-origin load nodes, and the attacker goes after the remaining undefended load nodes and rail edges that have a lot less redundancy than roads. We are surprised that mode change nodes escape defense, or attack (this may be due to our estimates of attack delays). We interpret each component transit time and attack delay time exactly as the network authors defined these—an attack does not interdict flow, but instead delays it and may divert operator flow to alternate, unattacked, but longer routes. Average resulting transit times are shown in Tables 11 and 12.

If we increase to 20 defenses the attacker could degrade the operator by 5%. Adding more defenses gradually improves things for the operator, but it takes several hundred defenses to

Table 9. DAO nested decomposition convergence trajectory for linearized instances.

Instance	DAO iterations	AO L-iterations	relgap_% DAO	Cumulative seconds
ONE	K01	4	60	11
	K02	4	7	13
EIGHT	K01	3	21	55
	K02	3	6	70
TWENTY	K01	2	16	135
	K02	2	4	178
THIRTY	K01	2	18	204
	K02	2	6	266
FORTY	K01	2	12	255
	K02	2	8	350

Notes: There are 14 attacks, but only six defenses—these are tough planning scenarios. Using the convergence tolerances from Table 8, the FORTY instance requires two major DAO iterations and almost seven minutes. These results are from a Lenovo P17 portable workstation with 128GB random access memory and a Passmark (2022) rating of 4,552. At most one GB random access memory was used.

This content downloaded from
205.155.65.226 on Tue, 14 Nov 2023 16:34:43 +00:00
All use subject to https://about.jstor.org/terms

Table 10. (DAO) computation times.

Instance	Model	Solves	Gen secs	Gen avg	Opt secs	Opt avg	Total secs
ONE	O reduce	14	7	1	1	0	9
	D	2	1	1	1	0	2
	A	8	1	0	0	0	1
	O	8	2	0	0	0	2
	Total	32	11	0	1	0	12
EIGHT	O reduce	12	38	3	3	0	40
	D	2	10	5	7	3	17
	A	6	1	0	0	0	1
	O	6	8	1	1	0	8
	Total	26	57	2	11	0	67
TWENTY	O reduce	11	87	8	7	1	94
	D	2	33	16	19	9	51
	A	4	2	1	0	0	2
	O	4	16	1	0	0	19
	Total	21	138	7	28	1	166
THIRTY	O reduce	11	126	11	11	1	127
	D	2	48	24	28	14	76
	A	4	3	1	0	0	3
	O	4	23	6	3	1	25
	Total	21	200	10	42	2	242
FORTY	O reduce	9	145	16	13	1	158
	D	2	72	36	48	24	120
	A	4	4	1	0	0	4
	O	4	34	8	4	1	38
	Total	19	256	13	66	3	321

Notes: The number of models solved appears with generation and optimization seconds required. The last defender (D) model in FORTY (Table 9) has about 2,052 K constraints and 1,536 K decision variables of which about 2 K are binary. Note the dominance of model generation time, generally around 80% of total time.

render the operator immune from the worst-case attack. Experimentation reveals that the first priority should be to defend break-out unit-origin load nodes and that the rail network has less redundancy than the roads: the attacker will be attracted in that order.

Figure 2 illustrates how optimization can discover simplicity hidden in clutter. In this excursion, we utilize the network from FORTY and allow unlimited defenses. The initial map (Figure 2(a)) shows the locations of 40 deploying units and their 126 lines, the candidate ports of embarkation and the rail and road networks (air and sea networks are not shown). Figure 2(b) shows the networks after (Armageddon) removal of components that won't be used by this deployment. Anticipating 14 attacks, the optimization chooses to defend 12 roads coming from unit-origin load nodes and two rail edges. (It is coincidental that the number of defenses selected equals the number of attacks.) Seeing these defenses, the attacker shifts attention in Figure 2(c) to six undefended roads and one rail edge coming from other load nodes, as well as seven rail edges. This leaves us with a strategically defended deployment network (Figure 2(d)) that the operator will be able to use after a worst-case attack with only about an 8% degradation in objective value.

Table 11. Average transit days for unit lines in scenario FORTY. Planners use this to synchronize arrivals of lines.

Lines	Load	Flow	Change	Total
PAX	1	1	0	2
AMMO	1	13	0	14
EQUIP	1	13	0	15
AIRCRAFT	4	3	0	7

This content downloaded from
205.155.65.226 on Tue, 14 Nov 2023 16:34:43 +00:00
All use subject to https://about.jstor.org/terms

Table 12. For scenario FORTY, the range of transit times in days with no attacks, and with every vulnerable component attacked.

Lines	No attack	Armageddon
PAX	1–4	2–14
AMMO	10–21	12–29
EQUIP	10–22	15–46
AIRCRAFT	4–5	16–27

Recall that we have $\varepsilon_{\mathrm{DAO}} = 0.1$; with $\varepsilon_{\mathrm{DAO}} = 0$ and unlimited defenses, we expect no degradation in the operator's objective value. For this extreme case we wouldn't use decomposition, but would solve the operator problem and defend every component used.

The formulations here accommodate only the most primitive actions by attacker and defender. This particular system exhibits an objective terrain that is quite smooth and pushing on with more whack-a-mole DAO iterations does not lead to much improvement. Realistically, one would want to embellish each of these actors with much more detailed descriptions of feasible courses of action. In particular, the decisions to attack and defend are not indexed by time here. Index these actions by time, and the most attractive modification to the operator model is a mechanism for

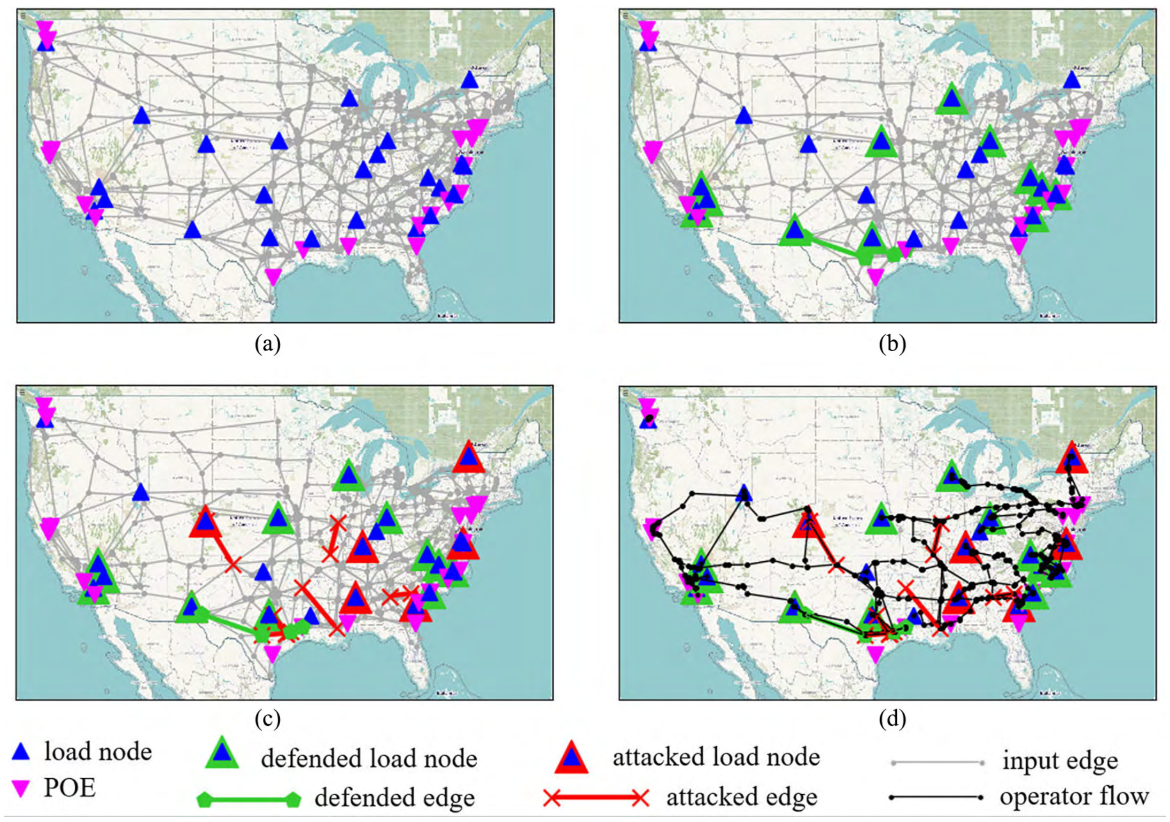

Figure 2. Unit origins and points of embarkation (POEs), defenses, attacks, and flows for FORTY.
Notes: The TPFD network in (a) consists of 1,625 road, 288 rail edges, and (not shown) 154 air and 34 ship edges. The 40 units with 126 lines are located in 25 CONUS military installations (upright blue triangles in (a)). There are 11 sea ports of embarkation (SPOEs) and six air ports of embarkation (APOEs) (inverted purple triangles in (a)); these serve as flow destinations. Knowing this, we can drop 247 road, 113 rail, 140 air, and 20 ship edges, reducing to the network shown in (b). Allowing an unlimited number of defenses, *only 14 are selected*, 12 protecting roads from military unit-origin load nodes (green triangles in (b)) and two defending rail edges (green edges in (b)). The 14 attacks then target six undefended roads and one rail edge departing military unit-origin load nodes and seven other rail edges (red in (c)). Subsequent operator flows (black in (d)) use 416 road edges, 93 rail edges, and (not shown) 11 ship and six air edges.

This content downloaded from
205.155.65.226 on Tue, 14 Nov 2023 16:34:43 +00:00
All use subject to https://about.jstor.org/terms

Table 13. Attacker master problem (AMAO).

$$z^*_{\text{AMAO}} = \max_{z,\mathbf{x}\in X} z \qquad \text{(AMAO1)}$$

$$\text{s.t.} \quad z \geq f\left(\mathbf{x}, \mathbf{y}^{\ell}_X, y^{\ell}_0\right) \quad \forall \ell = 1, \ldots, L \qquad \text{(AMAO2)}$$

Note: Operator penalties of the ℓ−th defender solution have been accumulated to match the components vulnerable to the attacker, and a fixed penalty sum for operating components not subject to attack. (For our specific example instance of (AMAO2), see (A1F).)

post-attack reconstitution (see, e.g., Alderson et al., 2017). These and many other realistic model features are accommodated by the methods presented here.

The analyses here use the attack delays we received with the network, verbatim. Given these performance estimates, the U.S. Strategic Highway and Rail networks appear remarkably resilient, with ample means at large scale to avoid attacked components when necessary. In contrast, see Alderson et al. (2017) for an analysis of the vulnerability of bridges and roads of the San Francisco Bay area, with reconstitution times extending from months to years. These two cases contrast attacks spread among many targets that are mere delaying irritants with destruction of a few key components. Both these extreme cases are amenable to the analysis presented here.

A useful specialization restricted to solving bilevel attacker-operator (AO). When we solve (AO), the preceding procedures can be specialized to good effect. Specialization (A1F) applies if the attacker's actions can be restricted to attack each vulnerable system component at most once; then we can accumulate the total penalty the operator expends in operation of each vulnerable component, and substitute that in the (A1) cut for each iteration, along with a term for total operator penalties not associated with vulnerable components. Call such projected penalty accumulation for solution $\mathbf{y}^*$ of (6) $\mathbf{y}^* \succ \{\mathbf{y}^*_X, y^*_0\}$, with $\mathbf{y}^*_X$ corresponding one-to-one with the attacks in $\mathbf{x} \in X$, and y^*_0 the penalties associated with operating all nonvulnerable components. Table 13 shows the Benders decomposition master problem (AMAO).

The motive for doing this is that the number of vulnerable operator activities is usually far fewer than of all the operator activities. In the numerical example FORTY, this condenses over two hundred thousand operator decisions during each iteration to just 2,460 penalty values to convey to the attacker master problem (AMAO).

A useful specialization for linear program operator models. The operator model example here is a linear program. During its development, we expected at any time that binary variables would need to be introduced, and in some excursions they have been.

But suppose the operator model is known to be a linear program.

We can replace the inner decomposition to solve (AO) by using the maximizing dual of the minimizing operator model. We introduce dual variables (unrestricted in sign) for operator (equality) constraints (O2)–(O5) and index these dual constraints (DYL), (DYF), and (DYC) to trace their provenance.

$$f(\hat{\mathbf{w}}, \mathbf{x}, \mathbf{y}) = g(\hat{\mathbf{w}}, \mathbf{x}, \mathbf{o}) = \sum_{\substack{(u,l)\in UL,\\ nx_{u,l}}} O2_{u,l,nx} + \sum_{\substack{(u,l)\in UL,\\ d_u \in N}} O5_{u,l,d_u} \qquad \text{(DUAL OBJ)}$$

$$\begin{aligned} &+O2_{u,l,nx} + O3_{u,l,nx,m} \\ &\leq load\_time_{u,l,nx,m} \\ &+load\_add\_time_{u,l,nx,m}\left(1 - \widehat{WL}_{nx,m}\right) XL^k_{nx,m} \qquad && \forall (u,l) \in UL, \\ & && (nx,m) \in LOAD \\ & && |(l,m) \in LM \qquad \text{(DYL)} \end{aligned}$$

This content downloaded from
205.155.65.226 on Tue, 14 Nov 2023 16:34:43 +00:00
All use subject to https://about.jstor.org/terms

$$\begin{aligned}
&-O3_{u,l,n1,m} + O4_{u,l,n2,m}\big|_{n2\neq d_u} \\
&+O5_{u,l,n2}\big|_{n1\in POE_{nx_{u,l}}\wedge n2=d_u} \\
&\quad \le \Big[+ edge\_time_{n1,n2,m} \\
&\qquad + edge\_add\_time_{n1,n2,m}\left(1-\widehat{WE}_{n1,n2,m}\right)XE^{k}_{n1,n2,m}\Big]_{n1<n2} \\
&\qquad \Big[+ edge\_time_{n2,n1,m} \\
&\qquad + edge\_add\_time_{n2,n1,m}\left(1-\widehat{WE}_{n2,n1,m}\right)XE^{k}_{n2,n1,m}\Big]_{n2<n1} \\
&\qquad\qquad\qquad \forall (u,l)\in UL,\ (n1,n2,m)\in ARC\ |(l,m)\in LM \qquad \text{(DYF)}
\end{aligned}$$

$$\begin{aligned}
&O3_{u,l,n,m2} - O4_{u,l,n,m1}\big|_{n\neq d_u} \\
&\quad \le chng\_time_{u,l,n,m1,m2} \\
&\quad +chng\_add\_time_{u,l,n,m1,m2}\left(1-\widehat{WC}_{n,m1,m2}\right)XC^{k}_{n,m1,m2} \\
&\qquad\qquad\qquad \forall (u,l)\in UL,\ (n,m1,m2)\in CHNG\ |(l,m1)\in LM \wedge (l,m2)\in LM \qquad \text{(DYC)}
\end{aligned}$$

Corresponding with this is a new master problem with objective $\min_{\mathbf{x}\in X} g(\mathbf{w},\hat{\mathbf{x}},\hat{\mathbf{o}})$. By now, in this paper it should be clear how these constraints would be expressed and handled and what solution elimination constraints for **w** look like.

Table 14 shows results of solving (AO) directly as a dual integer linear program and extracting the operator solution from the duals of constraints (DYL), (DYF), and (DYC).

We advise caution. Solving this operator-dual-integer linear program condensation of the inner AO optimization problems is not always faster than the inner AO decomposition in Table 3. Comparing times in Tables 9 and 14 show this trend. We have encountered two reasons for this. First, the solver may take longer for the dual. Second, to our surprise, GAMS requires a lot of time to convert the constraint marginals to recover a corresponding primal operator solution. We are familiar enough with an interface to GAMS to know this is a GAMS problem, not the solver.

And if you end up needing a discrete decision for the operator model, you will regret this effort.

Some alternatives for difficult problems. When solving either (AO) or (DAO), some "tricks of the trade" can be helpful.

Table 14. Solution with an operator-dual integer linear program solving (AO) directly using ε_A. These times are comparable to those in Table 10.

Instance	DAO iterations	relgap_% DAO	Cum. seconds
ONE	K01	59	10
	K02	5	19
EIGHT	K01	21	54
	K02	3	74
TWENTY	K01	15	132
	K02	2	176
THIRTY	K01	16	208
	K02	3	274
FORTY	K01	10	217
	K02	4	312

This content downloaded from
205.155.65.226 on Tue, 14 Nov 2023 16:34:43 +00:00
All use subject to https://about.jstor.org/terms

Table 15. Sequential operator defender-attacker heuristic.

Scenario	(ODA)	(ODAO)	(DAO)
ONE	112	4*	4, 8, 12
EIGHT	32	3*	3, 4, 9
TWENTY	24	12	5, 6*, 9
THIRTY	24	22	6, 7*, 12
FORTY	20	19	9, 13*, 13

Notes: An operate-defend-attack (ODA) heuristic achieves the percentage increase shown in delay when an operator plan initially reveals the desired utilization of each system component, then the defender sequentially protects the six ones that would be most delayed by an attack, and then the attacker sequentially targets the 14 most delayable remaining undefended components. ODAO shows how the operator would optimally respond after those ODA defenses and attacks. DAO shows the lower bound, solution value and upper bound when all actors are modeled to sequentially behave optimally. For the two smallest cases, the ODAO heuristic beats the decomposition and these solutions (*) can be substituted for those from the optimization. However, without the bounds from optimization, we wouldn't know when to trust this heuristic. As problems get larger and more nuanced, the primitive greedy heuristic doesn't do as well.

If one is fortunate enough to have defender and/or attacker models as simple as those presented here, these defender and attacker optimizations are easy to replace with sequential thumb rule heuristics. Table 15 shows what happens when the defender can see in a preferred operator plan which system components are most vulnerable to delay and sequentially allocate available defenses in decreasing order (ODA); then, the attacker targets remaining undefended components in decreasing order of delay. If the operator plan is revised after these fixed, sequential allocations (ODAO), delays can be reduced. In the smaller, simpler cases, this primitive greedy heuristic beats the formal optimization (unless the decomposition is run to a smaller decomposition gap). However, we wouldn't know whether to trust the heuristic without the reassuring lower bound from the DAO decomposition.

We again advise that the defender, attacker, and operator models be expressed so they each have an initial feasible solution. If the influence of defender decisions on the attacker model, and those of the attacker decisions on the operator model, are expressed in terms of penalties that do not interfere with feasibility, then infeasibility uniquely signifies that cuts and/or SECs have isolated an optimal solution.

For solving (AO), step AO 1 of Table 3 suggests an initial attacker solution $\hat{\mathbf{x}}^1 = 0$. Similarly, for solving (DAO), step DAO 1 in Table 5 suggests initial defender solution $\hat{\mathbf{w}} = \mathbf{0}$. These null initial solutions ease verification of models. Surprisingly, even when planners can suggest admissible (i.e., feasible) initial solutions (for instance, from prior solutions or subject matter expertise, or from a simple heuristic such as ODAO), although these can provide much improved initial solutions, this does not necessarily contribute significantly to the progress of the decomposition. An interesting characteristic of decomposition is that it seems to need to learn what not to do.

It is also easy to conduct intermediate solves, archiving resulting solutions and bounds, and retrieving and restoring these later for resumed solves. Resumed solves, lacking the history of Benders cuts, may introduce some cycling and/or stalling to reestablish support from cuts, but generally work well in practice. This permits periodic examination of solution progress.

Another useful decomposition technique is periodic editing of the Benders cuts (constraints (A1) and (D1)) to eliminate dominated ones. Although this could involve solving an ancillary optimization, there is a simple, effective heuristic: over some epoch of iterations, keep track of how often each cut is taut. Relax those that have been frequently inactive, perhaps even using exponential decay to degrade the recency of frequencies. If such a relaxed cut later proves essential, the decomposition will generate it again.

As a practical matter, if the defender-operator model (see Table 4) grows too large as outer iteration count K increases, an easy quick heuristic is simply to limit the domain of iterations (i.e., retained attacks and cuts) to a most-recent set, perhaps even only the Kth iteration. This relaxation frequently renders a good defense and a lower bound on how much better a defense might be (see Table 15).

Sometimes the operator model exhibits some well-known special structure, such as a multicommodity flow model. In such cases, using ancillary methods to identify things like vulnerable cut-

This content downloaded from
205.155.65.226 on Tue, 14 Nov 2023 16:34:43 +00:00
All use subject to https://about.jstor.org/terms

set choke points can be useful to guide better attacks earlier in the solution. However, it is advisable to seek features that are structural in the operator model and its data, rather than those that may be easy for the operator to accommodate (Alderson et al., 2013).

We have set up models based on the example data here with explicit time periods. For a planning horizon of 50 daily time periods, this expands the number of operator decision variables by about that factor. Undaunted, we employed servers with a lot more power than our Lenovo portable workstation and concluded that the insight gained for this paper was not worth the investment in computation and planning delay.

Simulation is an effective alternative, which can easily accommodate the scale of such a model, but which cannot to our knowledge provide in general a reliable objective function bound for any player here. Our limited experiments with simulation of DAO- and merely AO-style models have rendered stark results (e.g., see Alderson et al., 2013). This is not surprising, given that for the DAO models presented in this paper, the opponents present a sample space (employing casual, but unambiguous notation) proportionate to:

$$\binom{|defendables|}{|\boldsymbol{w}|}\binom{|vulnerables|}{|\boldsymbol{x}|}. \tag{8}$$

(For the 1,236 entities that can be defended and are vulnerable, with six defenses and 14 attacks, this presents merely 10^{48} alternatives.)

A time- and otherwise-aggregated DAO optimization model such as the one here can be used as a guide and correction model for a more detailed one. This is worth the effort if, e.g., a simulation-guided restriction of a DAO optimization can render objective bounds for either, or both opponents. We advise keeping as much formal optimization foundation as possible in lieu of abandoning such for sole use of simulation and/or heuristics, a path that has not worked well for us.

There is an advantage for opponents who can keep some actions secret from the adversary. Brown et al. (2005, 2011) show how to assess the value of such secrecy. When DAO is used to model vulnerability to actions mixing intelligent actors and Mother Nature, this is useful.

Our numerical example here offers few embellishing details about the actions of the opponents whose activities are not heavily intermingled. This is for simplicity of exposition. In practice, for example, defender activities can include stocking anticipatory spare replacement components, adding redundancy and increasing operator resources. Defensive measures can influence restoration time after an attack. When an integer linear program is admitted in all three modeling decision levels, one is limited only by one's imagination.

There are a number of suggestions for generalization to include other than linear integer optimization models in Alderson et al. (2014). These authors also explain how to use (DAO) (which they call DAD) to formally define and assess the crucial property of system *resilience*.

CONCLUSION

Although it can be very enlightening to apply DAO with these systems to see how they can be defended, then attacked, and then operated, in the end with real systems there are real defenders, attackers and operators. The attackers might be us, or some adversary, with the opposing role(s) also reversed.

For advising policy for dealing with real systems, actions take permission and resources, and policymakers will need to be briefed and comfortable enough with our advice to commit to action.

We have shown that insights from such analysis defy simple thumb rules. We have struggled to develop ways to illustrate the what and the why of these nuanced solutions (e.g., Alderson, et al. 2011, 2013, 2014, and Brown, et al. 2005, 2011). We have much more to learn about how to interpret and convey the insights offered by these models. The main purpose of this piece is to inform clearly how DAO optimization can be conducted with widely available modeling tools, and understood. For examples of how to present results, please see our references.

This content downloaded from
205.155.65.226 on Tue, 14 Nov 2023 16:34:43 +00:00
All use subject to https://about.jstor.org/terms

ACKNOWLEDGEMENTS

Although the results reported here are from a single portable workstation, getting to that sole computational device has involved a lot of experimental computing on a lot more computers. Our Professor Thomas Halwachs set up software and expanded hardware suites we could easily access during the COVID-19 months, providing support around-the-clock throughout. We came to call this our DAO-cloud. Dr. Michael Martinez and Samuel Billingham of MITRE Corporation have been working independently in parallel with us on their own DAO modeling of force deployment. They generously loaned us a carefully crafted transportation network database, and have shared their insights with us as they have developed their optimization tools. Along the way, we have been able to cue each other on potential improvements. We at Naval Postgraduate School have been gratified to be able to apply DAO to a number of military and homeland defense situations, but it is even more satisfying to see these methods applied by others in our national defense.

REFERENCES

Alassad, M., Davarikia, H., and Chan, Y. 2020. An Epistemic Utility-Theoretic Model in Fortifying Oil-and-Gas Production Networks, *Applied Sciences*, Vol 10, No 11, 3870.

Alderson, D. L., Brown, G. G., and Carlyle, W. M. 2014. Assessing and Improving Operational Resilience of Critical Infrastructures and Other Systems, *INFORMS Tutorials in Operations Research*, 180–215.

Alderson, D. L., Brown, G. G., Carlyle, W. M., and Cox, L. A. 2013. Sometimes There Is No 'Most-Vital' Arc: Assessing and Improving the Operational Resilience of Systems, *Military Operations Research*, Vol 18, No 1, 21–37.

Alderson, D. L., Brown, G. G., Carlyle, W. M., and Wood, R. K. 2017. Assessing and Improving the Operational Resilience of a Large Highway Infrastructure System to Worst-Case Losses, *Transportation Science*, Vol 52, No 1, 1012–1034.

Alderson, D. L., Brown, G. G., Carlyle, W. M., and Wood, R. K. 2011. Solving Defender-Attacker-Defender Models for Infrastructure Defense. In *Operations Research, Computing, and Homeland Defense*, R. K. Wood and R. F. Dell, eds. INFORMS, 28–49.

Alguacil, N., Delgadillo, A., and Arroyo, J. 2014. A Trilevel Programming Approach for Electric Grid Defense Planning, *Computers & Operations Research*, Vol 41, 282–290.

Avraamidou, S., and Pistikopoulos, E. N. 2020. A Global Optimization Algorithm for the Solution of Tri-Level Mixed-Integer Quadratic Programming Problems. In *Optimization of Complex Systems: Theory, Models, Algorithms and Applications. (World Congress on Global Optimization 2019). Advances in Intelligent Systems and Computing*, Vol 991. Springer, 579–588. https://doi.org/10.1007/978-3-030-21803-4_58

Baggio, A., Carvalho, M., Lodi, A., and Tramontani A. 2021. Multilevel Approaches for the Critical Node Problem, *Operations Research*, Vol 69, No 2, 486–508.

Bolusani, S., and Ralphs, T. K. 2022. A Framework for Generalized Benders' Decomposition and Its Application to Multilevel Optimization, *Mathematical Programming*, Vol 196, 389–426.

Brown, G. G., Carlyle, M., Salmeron, J., and Wood, K. 2006. Defending Critical Infrastructure, *Interfaces*, Vol 36, 530–544.

Brown, G. G., Carlyle, W. M., Diehl, D., Kline, J. E., and Wood, R. K. 2005, A Two-Sided Optimization for Theater Ballistic Missile Defense, *Operations Research*, Vol 53, No 5, 745–763.

Brown, G. G., Kline, J. E., Thomas, A., Washburn, A. R., and Wood, R. K. 2011. A Game-Theoretic Model for Defense of an Oceanic Bastion Against Submarines, *Military Operations Research*, Vol 16, No 4, 25–40.

Dahan, M., Sela, L., and Amin, S. 2020. Network Inspection for Detecting Strategic Attacks. Preprint, https://doi.org/https://doi.org/10.48550/arXiv.1705.00349.

Davarikia, H., Barati, M., Mustafa Al-Assad, M., and Chan, Y. 2020. A Novel Approach in Strategic Planning of Power Networks against Physical Attacks, *Electric Power Systems Research*, Vol 180, 106140.

This content downloaded from
205.155.65.226 on Tue, 14 Nov 2023 16:34:43 +00:00
All use subject to https://about.jstor.org/terms

Ding, T., Yao, L., and Li, F. 2018. A Multi-Uncertainty-Set Based Two-Stage Robust Optimization to Defender-Attacker-Defender Model for Power System Protection, *Reliability Engineering & System Safety*, Vol 169, 179–186.

Fakhry, R., Hassini, E., Ezzeldin, M., and El-Dakhakhni, W. 2021. Tri-level Mixed-Binary Linear Programming: Solution Approaches and Application in Defending Critical Infrastructure, *European Journal of Operational Research*, Vol 298, No 3, 1114–1131.

GAMS. 2012. GAMS.com.

Geoffrion, A. M. 1972. Generalized Benders Decomposition, *Journal of Optimization Theory and Applications*, Vol 10, 237–260.

Ghorbani-Renani, N., González, A. D., and Barker, K. 2021. A Decomposition Approach for Solving Tri-level Defender-Attacker-Defender Problems, *Computers & Industrial Engineering*, Vol 153, 107085.

GSO. 2012. Global Security, https://www.globalsecurity.org/military/facility/stracnet.htm.

Gurobi. 2022. https://www.gurobi.com/.

IBM. 2022. https://www.ibm.com/analytics/cplex-optimizer.

Joint Chiefs of Staff. 2017a. Joint Publication 4-01 The Defense Transportation System, July 18.

Joint Chiefs of Staff. 2017b. *Joint Planning 5-0*, June 16.

Joint Chiefs of Staff. 2018. *Deployment and Redeployment Operations*, January 10.

Joint Chiefs of Staff. 2019. *Joint Publication 4-0 Joint Logistics*, May 8.

Lazzaro, G. 2016. *Tri-Level Optimization Algorithms for Solving Defender-Attacker-Defender Network Models*. Naval Postgraduate School dissertation.

Lin, Y., and Bie, Z. 2018. Tri-level Optimal Hardening Plan for a Resilient Distribution System Considering Reconfiguration and DG Islanding, *Applied Energy*, Vol 210, 1266–1279.

Lozano, L., and Smith, J. C. 2017. A Backward Sampling Framework for Interdiction Problems with Fortification, *INFORMS Journal on Computing*, Vol 29, No 1, 123–139.

Nicholas, P., and Alderson, D. L. 2015. *Designing Interference-Robust Wireless Mesh Networks Using a Defender-Attacker-Defender Model*. Naval Postgraduate School Technical Report, February 2015.

Owen, G. 1969. Minimization of Fatalities in a Nuclear Attack Model, *Operations Research*, Vol 17, No 3, 489–505.

Passmark. 2022. https://www.cpubenchmark.net/cpu_list.php.

Salmeron, J., Wood, K., and Baldick, R, 2009. Worst-Case Interdiction Analysis of Large-Scale Electric Power Grids, *IEEE Transactions on Power Systems* Vol 24, 96–104

San Martin, P. A. 2007. *Tri-Level Optimization Models to Defend Critical Infrastructure*. Master's thesis, Naval Postgraduate School.

Simchi-Levi, D., Trichakis, N., and Yun Zhang, P. 2019. Designing Response Supply Chain Against Bioattacks, *Operations Research*, Vol 67, No 5, 1246–1268.

Smith, J. C., and Song, Y. 2020. A Survey of Network Interdiction Models and Algorithms, *European Journal of Operational Research*, Vol 283, 797–811.

Thompson, K. H., and Tran, H. T. 2020. Operational Perspectives Into the Resilience of the U.S. Air Transportation Network Against Intelligent Attacks, *IEEE Transactions on Intelligent Transportation Systems*, Vol 21, No 4, 1503–1513.

U.S. Army. 2013. Highways for National Defense, *Surface Deployment and Distribution Command*. https://www.sddc.army.mil.

U.S. Army. 2018, Strategic Rail Corridor Network (STRACNET), *Surface Deployment and Distribution Command*. https://www.sddc.army.mil.

Wu, X., and Conejo, A. J. 2017. An Efficient Tri-Level Optimization Model for Electric Grid Defense Planning, *IEEE Transactions on Power Systems*, Vol 32, No 4, 2984–2994.

Wu, Y., Chen, Z., Gong, H., Feng, W., Chen, Y., and Tang, W. 2021. Defender–Attacker–Operator: Tri-level Game-Theoretic Interdiction Analysis of Urban Water Distribution Networks, *Reliability Engineering & System Safety*, Vol 214, 107703.

This content downloaded from
205.155.65.226 on Tue, 14 Nov 2023 16:34:43 +00:00
All use subject to https://about.jstor.org/terms

Xiang, Y., and Wang, L. 2019. An Improved Defender–Attacker–Defender Model for Transmission Line Defense Considering Offensive Resource Uncertainties, *IEEE Transactions on Smart Grid*, Vol 10, No 3, 2534–2546.

Yuan, W., and Zeng, B. 2020. Cost-Effective Power Grid Protection through Defender–Attacker–Defender Model with Corrective Network Topology Control, *Energy Systems*, Vol 11, 811–837.

Yuan, W., Zhao, L., and Zeng, B. 2014. Optimal Power Grid Protection through a Defender–Attacker–Defender Model, *Reliability Engineering & System Safety*, Vol 121, 83–89.

This content downloaded from
205.155.65.226 on Tue, 14 Nov 2023 16:34:43 +00:00
All use subject to https://about.jstor.org/terms

This content downloaded from
205.155.65.226 on Tue, 14 Nov 2023 16:34:43 +00:00
All use subject to https://about.jstor.org/terms