Appears in *The Next Wave in Computing, Optimization and Decision Technologies*, 2005, B. Golden, S. Raghavan and E. Wasil editors, Springer, New York, pp. 3-17.

ON THE COMPLEXITY OF DELAYING AN ADVERSARY'S PROJECT

Gerald G. Brown,[1] W. Matthew Carlyle, [2] Johannes Royset,[3] and R. Kevin Wood[4]

Naval Postgraduate School
Monterey, CA 93943

[1] gbrown@nps.edu

[2] mcarlyle@nps.edu

[3] joroyset@nps.edu

[4] kwood@nps.edu

Abstract A "project manager" wishes to complete a project (e.g., a weapons-development program) as quickly as possible. Using a limited interdiction budget, an "interdictor" wishes to delay the project's overall completion time by interdicting and thereby delaying some of the project's component tasks. We explore a variety of PERT-based interdiction models for such problems and show that the resulting problem complexities run the gamut: polynomially solvable, weakly NP-complete, strongly NP-complete or NP-hard. We suggest methods for solving the problems that are easier than worst-case complexity implies.

Keywords: Interdiction, PERT, NP-complete

1. Introduction

Brown et al. (2004) (see also Reed 1994 and Skroch 2004) model the completion of an adversarial nation's nuclear-weapons program using general techniques of PERT. (See PERT 1958 and Malcolm et al. 1959 for the original descriptions of PERT, and see Moder et al. 1983 for a comprehensive review.) Brown et al. (2004) ask the question: How do we most effectively employ limited interdiction resources, e.g., military strikes or embargoes on key materials, to delay the project's component tasks, and thereby delay its overall completion time? They answer the question by describing an interdiction model that maximizes minimum project-completion time. This model is a Stackelberg game (von Stack-

elberg 1952), formulated as a bilevel integer-linear program (Moore and Bard 1990).

Brown et al. (2004) consider a highly general model for project networks. Specifically, they (*i*) allow the interdictor to employ various interdiction resources, (*ii*) allow the project manager to "crash" the project to speed project completion by applying various, constrained, task-expediting resources, and (*iii*) allow the project manager to employ alternative technologies to complete the project. The authors successfully test an algorithm that solves a realistic example of the resulting interdiction problem, but we shall see that the most general problem is NP-hard. Thus, other large, general problems could be extremely difficult to solve.

This paper therefore asks: How hard is the "project interdiction problem" when full modeling generality is unnecessary? Can we assure analysts that their version of the problem is not too difficult if modeling a single interdiction resource suffices; or crashing is impossible; or only a single technology, or modest number of technologies, need be modeled?

We show that these less general problems are, in fact, easier to solve, and go on to describe special solution techniques for them. All of these techniques are simpler than the decomposition algorithm described by Brown et al. (2004), which requires that an alternating sequence of two integer-linear programs be solved. Thus, simpler, more accessible and more efficient solution methods may be employed for these problem restrictions.

Before beginning mathematical developments, we note that we have chosen the activity-on-arc (AOA) model of a project network rather than the interchangeable activity-on-node (AON) model. The AON model is the more common of the two nowadays; however, the mathematics in this paper prove easier to describe using the AOA model, so we adopt that model from the outset.

The next section provides basic definitions for our project interdiction problems. Section 3 describes the most general model, which includes multiple technologies and project crashing. Subsequent sections discuss restricted model variants and solution techniques for them.

2. Basic Definitions

Let $G = (N, A)$ denote a *directed acyclic graph* with *node set* N and *arc set* $A \subset N \times N$. Since G is acyclic, there exists a *topological ordering*, or labeling, $1, 2, \ldots, |N|$ of the nodes $i, j \in N$ such that $i < j$ for each arc $k = (i, j) \in A$. For graphs of interest in this paper, the first node a in any such ordering is unique, as is the last node b. The *forward star* of

node i, $FS(i) \subseteq A$, is the set of all arcs of the form $k = (i, j)$; the *reverse star* of node i, $RS(i) \subseteq A$, is the set of all arcs of the form $k = (j, i)$.

G represents the *activity-on-arc* diagram used in a PERT model of a project, controlled by a project manager (e.g., Elmaghraby 1977). Each arc $k \in A$ corresponds to a *task* which must be completed in order to finish the project. For each node $i \in N$, all tasks $k \in RS(i)$ must be completed before any task $k \in FS(i)$ can begin. Every node $i \in N$ represents a *milestone event* that occurs when all predecessor tasks, i.e., all $k \in RS(i)$ are complete. A milestone event might be something important like "completion of the weapon delivery system," or might simply correspond to the completion of a group of simpler tasks along the course of the project. The latter situation may occur frequently in AOA representations of projects which often have many dummy nodes (and arcs). Node b is the *project-completion event* and, because event i may also be viewed as the start of follow-on tasks $k \in FS(i)$, node a is the *project-start event*.

Each activity k has associated with it a nominal *task completion time* $t_k \geq 0$ and a variable e_k, $0 \leq e_k \leq \bar{e}_k \leq t_k$, which denotes the reduction in the activity's completion time achieved by applying *expediting resources*. No matter how much expediting resource is a applied, however, task k cannot be completed any faster than the crashed duration, $t_k - \bar{e}_k$. For simplicity in writing models, but without loss of generality, we assume that only a single expediting resource exists (e.g., money); the unit cost of expediting task k is m_k; and a total *expediting budget* of m_0 monetary units is available to the project manager. We assume that the project manager schedules tasks in order to minimize the project completion time. It is well known that the shortest completion time, for fixed expediting decisions, corresponds to a longest a-b path in G.

An interdictor who wishes to disrupt the project possesses a set of *interdiction resources* with which to effect this disruption. Interdiction of arc k consumes $c_{rk} \in Z^+$ units of each interdiction resource type $r \in R$, and results in adding a *delay*, $d_k \in Z^+$, to the completion time of task k. The total *interdiction budget* for resource r is $c_{r0} \in Z^+$.

If we assume that no expediting will occur, the project-interdiction model looks much like the shortest-path interdiction model of Israeli and Wood (2002). There, the interdictor attacks a road network using limited interdiction resources, and the "network user," analogous to the project manager, moves along a post-interdiction shortest path in the network. In that model, an interdiction plan is evaluated by solving a shortest-path in a general network. Our simplest model can evaluate an interdiction plan by solving a longest-path problem in an acyclic network. However, this evaluation will require the solution of a more gen-

eral linear or integer-linear program if the project manager can crash his project or can employ multiple technologies, as described below. Thus, project interdiction is truly a "system-interdiction problem" (Israeli and Wood 2002), not a network-interdiction problem.

Crowston and Thompson (1967) describe an extension of project management models in which the project manager can complete a project using alternative technologies. Brown et al. (2004) use this extension to model different means of uranium enrichment. Crowston and Thompson create graphical constructs to represent alternative technologies in their AON model, but they boil down to this in the mathematical model: Using binary variables to represent whether or not a particular technology is used, certain precedence relationships will be enforced and certain others will be relaxed.

Brown et al. (2004) also include in their model several different types of precedence relationships between tasks (Elmaghraby 1977). We do not specify details, but all models in this paper can be easily adjusted for these more general precedence relationships. A fixed "lag time" may also be interjected between any pair of tasks, if required.

3. Project Interdiction Model

Here we define the general project interdiction model, **MAXMIN0**. We assume the unit of time is "(one) week" and that each interdiction resource r is measured in "r-dollars:"

MAXMIN0

Indices and Index Sets

$i, j \in N$	generic milestone events
$a, b \in N$	project start event and project completion event, respectively
$k \in A$	tasks and precedence relationships ($k = (i, j) \in A$)

Data [units]

t_k	task duration [weeks]
m_k	per-unit expediting cost of task k [dollars/week]
m_0	total expediting budget [dollars]
$\bar{e}_k$	maximum expediting of task k [weeks]
d_k	interdiction delay of task k [weeks]
c_{rk}	interdiction cost for task k, resource r [r-dollars/week]
c_{r0}	total amount of interdiction resource r available [r-dollars]
M	a sufficiently large constant, e.g., $M = \sum_{k \in A}(t_k + d_k)$ [weeks] (used to relax precedence constraints)

Decision Variables [units]

s_i	completion time of event i [weeks]
e_k	amount that task k is expedited [weeks]
w_i	1 if technology at node i is used, else 0
x_k	1 if task k is interdicted, else 0

Formulation (**MAXMIN0**)

$$z_0^* \equiv \max_{\mathbf{x}\in\mathbf{X}} \min_{\mathbf{s},\mathbf{e},\mathbf{w}} s_b \tag{1}$$

$$\text{s.t. } s_j - s_i + e_k + M(1-w_i) \geq t_k + d_k x_k \;\forall\; k=(i,j) \tag{2}$$

$$e_k \leq \bar{e}_k \;\; \forall\, k \in A \tag{3}$$

$$\sum_{k\in A} m_k e_k \leq m_0 \tag{4}$$

$$\mathbf{w} \in \mathbf{W} \tag{5}$$

$$w_i = 1 \;\; \forall\, i \in N - N_T \tag{6}$$

$$s_a = 0 \tag{7}$$

$$s_i \geq 0 \;\; \forall\, i \in N \tag{8}$$

$$e_k \geq 0 \;\; \forall\, k \in A \tag{9}$$

$$w_i \in \{0,1\} \;\; \forall\, i \in N. \tag{10}$$

where

$$\mathbf{X} \equiv \left\{ \mathbf{x} \in \{0,1\}^{|A|} \;\middle|\; \sum_{k\in A} c_{rk} x_k \leq c_{r0} \;\; \forall\, r \in R \right\}, \tag{11}$$

and where the set $\mathbf{W} \subset \{0,1\}^{|N|}$ represents all feasible combinations of alternative technologies.

For a fixed interdiction plan $\mathbf{x} = \hat{\mathbf{x}}$, the inner minimization in **MAXMIN0** is the project manager's problem: Compute the earliest project-completion time through the objective (1), subject to standard precedence constraints (2). Assuming all $w_i = 1$ so that all terms $M(1-w_i) = 0$, these constraints state that if activity $k = (i,j)$ exists between events i and j, then event j can occur no sooner than $s_i + t_k - e_k + d_k\hat{x}_k$. For $i \in N_T$, the term $M(1-w_i)$ simply relaxes all constraints for $k \in FS(i)$ when the alternative technology associated with i is not used, i.e., if $w_i = 0$. Constraint (4) reflects the project manager's limited budget for expediting tasks.

The interdictor controls the vector $\mathbf{x}$, and will use his limited interdiction resources (constraints 11) to maximize the project manager's minimum time to project completion. This is represented by the outer maximization in **MAXMIN0**.

The formulation **MAXMIN0** clarifies the opposing forces in our "Stackelberg interdiction game." The key features of this game are: (*i*) A "leader," i.e., the interdictor, first takes his actions, (*ii*) the "follower," i.e., the project manager, sees these actions and responds optimally, and (*iii*) the game finishes. Randomized strategies, as in two-person zero-sum games, are irrelevant here because the leader has complete information regarding the follower's behavior, and the follower will not act until after obtaining complete information of the leader's actions.

If we view **MAXMIN0** as the interdictor's optimization problem

$$z_0^* \equiv \max_{\mathbf{x}\in\mathbf{X}} z(\mathbf{x}), \tag{12}$$

where $z(\mathbf{x})$ defines the value of the resulting minimization problem for any value of $\mathbf{x}$, it is easy to see that the problem may be unusually difficult: Just to evaluate a potential interdiction plan $\hat{\mathbf{x}}$, i.e., just to compute $z(\hat{\mathbf{x}})$, requires the solution of an integer-linear program (ILP). If that ILP corresponds to an NP-hard problem, then **MAXMIN0** is NP-hard. In the following, we consider some special cases that are not quite that difficult.

4. One Technology, No Expediting

Suppose that a fixed set of technologies will be used, so $N_T = \emptyset$, and $w_i = 1$ for all $i \in N$. Further, assume that expediting is impossible, i.e., $e_k = 0$ for all $k \in A$. Then, **MAXMIN0** simplifies to:

MAXMIN1

$$z_1^* \equiv \max_{\mathbf{x}\in\mathbf{X}} \min_{\mathbf{s}} s_b \tag{13}$$

$$\text{s.t. } s_j - s_i \geq t_k + d_k x_k \quad \forall\, k = (i,j) \in A \tag{14}$$

$$s_a = 0 \tag{15}$$

$$s_i \geq 0 \quad \forall\, i \in N \tag{16}$$

For the time being, we will also assume that only a single interdiction resource (e.g., dollars) need be modeled, so that $\mathbf{X}$ is replaced by

$$\mathbf{X}_1 \equiv \left\{ \mathbf{x} \in \{0,1\}^{|A|} \,\middle|\, \sum_{k\in A} c_k x_k \leq c_0 \right\} \tag{17}$$

For fixed $\mathbf{x} = \hat{\mathbf{x}}$, the inner minimization of **MAXMIN1** is a linear program (LP) with a corresponding dual. In fact, the inner minimization in **MAXMIN1** is the well-known "earliest project completion time"

problem with the longest-path problem as its dual (e.g., Ahuja et al. 1993 pp. 732-737). Hence, fixing $\mathbf{x}$ temporarily, manipulating **MAXMIN1** slightly, taking the dual of the inner minimization, and releasing $\mathbf{x}$ leads to the following useful model:

$$z_1^* = \max_{\mathbf{x}\in\mathbf{X}_1} \max_{\mathbf{y}} \sum_k (t_k + x_k d_k) y_k \tag{18}$$

$$\text{s.t.} \sum_{k\in FS(i)} y_k - \sum_{k\in RS(i)} y_k = \begin{cases} 1 & \text{if } i = a \\ 0 & \text{if } i \in N - \{a,b\} \\ -1 & \text{if } i = b \end{cases} \tag{19}$$

$$y_k \geq 0 \;\; \forall\, k \in A \tag{20}$$

A max-max problem is a "simple" maximization, but the nonlinear, nonconcave objective function (18) is problematic. This model linearizes easily, however: Replace each arc k with a pair of arcs, k and k', with fixed lengths $t_k + d_k$ and t_k, respectively, and let x_k control which arc is part of the project manager's model:

MAXMAX1

$$z_1^* = \max_{\mathbf{x},\mathbf{y},\mathbf{y}'} \sum_{k\in A} (t_k + d_k) y_k + \sum_{k\in A} t_k y'_k \tag{21}$$

$$\text{s.t.} \sum_{k\in FS(i)} (y_k + y'_k) - \sum_{k\in RS(i)} (y_k + y'_k) = \begin{cases} 1 & \text{if } i = a \\ 0 & \text{if } i \in N - \{a,b\} \\ -1 & \text{if } i = b \end{cases} \tag{22}$$

$$y_k - x_k \leq 0 \;\; \forall\, k \in A \tag{23}$$

$$y_k, y'_k \geq 0 \;\; \forall\, k \in A \tag{24}$$

$$\mathbf{x} \in \mathbf{X}_1 \tag{25}$$

THEOREM 1 **MAXMAX1** *is solvable in* $O(c_0|A|)$ *time, i.e., in pseudo-polynomial time.*

Proof: **MAXMAX1** represents a singly-constrained longest-path problem in which traversal of arc k consumes c_k units of interdiction resource and traversal of arc k' consumes none. Thus, **MAXMAX1** may be solved through the following dynamic-programming recursion in $O(c_0|A|)$ time:

$$f(1,c) = 0 \text{ for } c = 0,\ldots,c_0 \tag{26}$$

$$f(i,c) = -\infty \;\forall\, i \in N,\; c < 0 \tag{27}$$

$$f(j,c) \quad = \quad \max \left\{ \begin{array}{l} f(j,c-1), \\ \max\limits_{k=(i,j)\in RS(j)} (f(i,c-c_k)+t_k+d_k), \\ \max\limits_{k=(i,j)\in RS(j)} (f(i,c)+t_k) \end{array} \right\}. \quad (28)$$

∎

Our next task is to show that **MAXMIN1** is weakly NP-complete. Later in the paper we will require the formality of "decision problems" to show NP-completeness, but here the reader should have no difficulty in seeing the equivalence of certain optimization problems and how that equivalence implies NP-completeness.

THEOREM 2 **MAXMIN1** *is weakly NP-complete.*
Proof: Define the *binary knapsack problem* (**BKP**) as

$$\max_{\mathbf{x}} \sum_{k\in A} d_k x_k \quad (29)$$

$$\text{s.t.} \sum_{k\in A} c_k x_k \quad \leq \quad c_0 \quad (30)$$

$$x_k \quad \in \quad \{0,1\} \quad \forall\, k \in A. \quad (31)$$

BKP is known to be NP-complete (e.g., Garey and Johnson 1979, p. 247) and can be modeled as an instance of **MAXMAX1** as follows:

1 Let $t_k = 0$ for all $k \in A$.

2 Let each item k in the knapsack correspond to an arc k with length $t_k + d_k$ and with "traversal cost" c_k.

3 Place all arcs in series.

4 In parallel with each arc k place an arc k' with length $t_k = 0$ and no traversal cost.

This transformation shows that **MAXMAX1** is NP-hard. But Theorem describes a pseudo-polynomial solution procedure for **MAXMAX1**, so it must, in fact, be weakly NP-complete. Since **MAXMAX1** is equivalent to **MAXMIN1**, the result follows. ∎

If the interdictor is only limited by a specific number of interdictions, **MAXMIN1** becomes even easier:

COROLLARY 3 **MAXMIN1** *is solvable in* $O(|N||A|)$ *time when* $c_k = 1$ *for all* $k \in A$.

Proof: In this case, any value of $c_0 > |N| - 1$ is equivalent to $c_0 = N - 1$ since no a-b path in G can have more than $|N| - 1$ arcs. The complexity result in Theorem 2, plus equivalence of models, then yields the result. ∎

Being able to solve these problems by dynamic programming means that fairly large problems can be solved quite effectively. However, dynamic programming can, in fact, bog down and we suggest using the constrained-shortest-path algorithm of Carlyle and Wood (2003), which converts directly to longest paths in directed acyclic paths. These authors show orders of magnitude speedups over previously known methods, including standard dynamic-programming formulations. (See Handler and Zang 1980 for a basic reference on this topic.)

5. One Technology With Expediting

Suppose the project manager can expedite certain tasks, but still, only a single set of technologies exists. **MAXMIN0** simplifies to:

MAXMIN2

$$z_2^* \equiv \max_{\mathbf{x}\in\mathbf{X}} \min_{\mathbf{s},\mathbf{e}} s_b \tag{32}$$

$$\text{s.t. } s_j - s_i + e_k \geq t_k + d_k x_k \ \ \forall\ k = (i,j) \in A \tag{33}$$

$$e_k \leq \bar{e}_k \ \forall\ k \in A \tag{34}$$

$$\sum_{k\in A} m_k e_k \leq m_0 \tag{35}$$

$$s_a = 0 \tag{36}$$

$$s_i \geq 0 \ \ \forall\ i \in N \tag{37}$$

Similar to **MAXMIN1**, for fixed $\mathbf{x}$, the inner minimization in **MAXMIN2** is an LP and we may thus take its dual. Doing that and manipulating the resulting model a bit leads to the following ILP:

MAXMAX2

$$\begin{aligned} z_2^* = \max_{\mathbf{x},\mathbf{y},\mathbf{y}',\pi_0} & \sum_{k\in A}(t_k + d_k)y_k \ + \ \sum_{k\in A} t_k y'_k \\ & -(m_0 - \sum_{k\in A} m_k \bar{e}_k)\pi_0 \\ & \text{s.t.} \sum_{k\in FS(i)} (y_k + y'_k) \end{aligned} \tag{38}$$

$$-\sum_{k\in RS(i)}(y_k+y'_k) = \begin{cases} 1 & \text{if } i=a \\ 0 & \text{if } i\in N-\{a,b\} \\ -1 & \text{if } i=b \end{cases} \quad (39)$$

$$y_k - x_k \leq 0 \quad \forall\, k\in A \quad (40)$$

$$y_k + y'_k - m_k\pi_0 \geq 0 \quad \forall\, k\in A \quad (41)$$

$$y_k, y'_k \geq 0 \quad \forall\, k\in A \quad (42)$$

$$\pi_0 \geq 0 \quad (43)$$

$$\mathbf{x} \in \mathbf{X} \quad (44)$$

We will next prove that **MAXMIN2** is NP-complete, or rather, that its associated decision problem, **MAXMIN2d**, is NP-complete. We need the formality of decision problems now, and the definition of **MAXMIN2d** is:

DEFINITION 4 **MAXMIN2d**. *Given: Data for* **MAXMIN2** *and threshold* $\underline{z}$. *Question: Does there exist an interdiction plan* $\mathbf{x}^*$ *such that the optimally expedited project (optimal for the project manager) has length at least* $\underline{z}$*?* ∎

And, we will use a transformation from **SETCOVERd** in the proof:

DEFINITION 5 **SETCOVERd**. *Given:* $N_2 \equiv \{m+1, m+2, \ldots, m+n\}$, *the "ground set" to be covered; subsets* $N_i \subseteq N_2$, *for* $i \in N_1 \equiv \{1,\ldots,m\}$, *and threshold* $\bar{n} \in Z^+$. *Question: Does there exist a set* $N'_1 \subset N_1$, *with* $|N'_1| \leq \bar{n}$, *such that* $\cup_{i\in N'_1} N_i = N_2$*?* ∎

For our purposes, it is easier to use **SETCOVERd** defined through the bipartite graph $G' \equiv (N_1, N_2, A')$, where $A' \equiv \{(i,j)|i \in N_1, j \in N_i \text{ for some } i\}$:
Does there exist a set $N'_1 \subset N_1$ *with* $|N'_1| \leq \bar{n}$ *such that* $\cup_{i\in N'_1} N_i = N_2$*?*

THEOREM 6 **MAXMIN2** *is strongly NP-complete.*
Proof: Since the decision version of **MAXMIN1** is NP-complete and it is a special case of **MAXMIN2d**, **MAXMIN2d** must be NP-hard. Because we can formulate an ILP to represent the optimization problem, **MAXMIN2d** must, in fact, be NP-complete. The only open questions is whether **MAXMIN2d** is NP-complete in the strong or weak sense. We will show that a standard set-covering problem, **SETCOVERd**, well-known to be strongly NP-complete, can be transformed into an instance of **MAXMIN2d**. The transformation will obviously not require an exponential increase in the size of this instance's data, so it will follow that **MAXMIN2d** is strongly NP-complete.

We are given an instance of **SETCOVERd**, defined as in Definition 5 through the bipartite graph $G' = (N_1, N_2, A)$ and the threshold parameter $\bar{n}$. Next, we form a corresponding instance of **MAXMIN2d**: Create

the directed, acyclic project network $G \equiv (N, A)$ from G' by adding two nodes, a and b, and two sets of arcs so that $N \equiv N_1 \cup N_2 \cup \{a, b\}$ and $A \equiv A' \cup A_1 \cup A_2$ where $A_1 \equiv \{(a, i) | i \in N_1\}$ and $A_2 \equiv \{(j, b) | j \in N_2\}$. Let $t_k = 1$ for all $k \in A$; let $d_k = 1$ for all arcs $k \in A_1$, and $d_k = 0$, otherwise; assume each arc $k \in A_2$ can be expedited by e_k, $0 \leq e_k \leq 1$; let the unit cost of expediting be 1; and assume a total of $|N_2| - 1$ units of expediting resource are available. The number $\bar{n} \in Z^+$ carries over directly from above.

So, we have created a directed acyclic network with three echelons of arcs, but only those in the first echelon may be interdicted (with any effect), and only those in the last may be expedited. The instance of **MAXMIN2d** is defined as: Does there exist a set of $\bar{n}$ or fewer interdictions of arcs in A_1 such that the longest path in G, with optimal expediting has length strictly greater than 3? The answer to this problem is "yes," if and only if the answer is "yes" to the original set-covering problem.

To see this, suppose that every collection of $\bar{n}$ subsets N_i leaves at least one element of N_2 uncovered. The corresponding interdiction plan interdicts arc (s, i) for each subset N_i. Because at least one node $j \in N_2$ is left uncovered in the set-covering problem, at least one arc (j, t) is not on an interdicted path. This means there is at least one path of length 3 in the network. Furthermore, the $N_2 - 1$ units of expediting resource suffice to reduce the length of all arcs in A_2 that are on interdicted paths to 0, and, hence, every interdicted path's length is dropped from 4 to 3. So, if the answer to **SETCOVERd** is "no," the answer to the corresponding instance of **MAXMIN2d** must be "no."

On the other hand, suppose that the answer to **SETCOVERd** problem is "yes." Interdict arcs corresponding to the cover as above. Then, the interdicted but unexpedited length of each path is 4, and the $|N_2|-1$ units of expediting resource only suffice to reduce those path lengths to $3+1/|N_2|$. So, the answer to the corresponding instance of **MAXMIN2d** is "yes." ∎.

Note that Theorem 6 holds also in the special case of $c_k = d_k = 1$, $t_k \in \{0, 1\}$ for all $k \in A$. Since **MAXMAX2** is a (linear) ILP, it can be solved by a standard LP-based branch-and-bound algorithm. In addition, **MAXMAX2** motivates a solution approach for the *general* problem **MAXMIN0** as described in the following.

6. Alternative Technologies

The discussion at the end of Section 2 implies that adding alternative technologies into the mix, i.e., going from **MAXMIN2** to the com-

pletely general model **MAXMIN0**, may move us from the realm of NP-complete problems into NP-hard problems that may not be in NP. This will be the case if, for fixed $\mathbf{x} = \hat{\mathbf{x}}$, the solution of **MAXMIN0** requires the solution of an NP-complete ILP. That is, just checking whether the interdictor's objective $z(\hat{\mathbf{x}})$ exceeds a specified threshold for a candidate solution $\hat{\mathbf{x}}$ requires the solution of an NP-complete problem, rather than the application of some polynomial-time procedure.

However, if no expediting is allowed we would like to know the resulting complexity of evaluating $z(\mathbf{x})$. That is, faced with a fixed set of task lengths $\hat{t}_k = t_k + d_k\hat{x}_k$, the project manager would like to solve the **DCPM**, the "decision CPM problem," (Crowston and Thompson 1967), which selects a set of alternative technologies by choosing $\mathbf{w} \in \mathbf{W}$ to minimize project completion time. We state **DCPMd**, the decision version of **DCPM**, in terms of deleting technologies (and represent the remaining technologies after deleting $\mathbf{w}$ by $\mathbf{1} - \mathbf{w}$) to help show its NP-completeness:

DEFINITION 7 **DCPMd**. *Given: A project network $G = (N, A)$ with arc lengths $\hat{t}_k \in Z^+$; constraints $\mathbf{w} \in \mathbf{W}$ indicating feasible sets of alternative technologies; and threshold $\underline{z} \in Z^+$. Question: Does there exist a set of technologies represented by $\mathbf{w}'$, with $\mathbf{1} - \mathbf{w}' \in \mathbf{W}$, such that the longest path in $G' = G - N'$ is no longer than $\underline{z}$, given $N' \equiv \{i \in N | w'_i = 1\}$?* ∎

We will show that **DCPMd** is strongly NP-complete through a transformation of **VERTEXCOVERd** (Garey and Johnson 1979, pp. 79, 190). We note that De et al. (1997) prove the NP-completeness of the "discrete time-cost tradeoff problem for project networks" (i.e., optimal project crashing with discrete expediting quantities), and that proof can be applied to **DCPMd**. However, our proof is substantially shorter than that of De et al., and we believe its inclusion is warranted for that reason, as well as for the sake of completeness.

DEFINITION 8 **VERTEXCOVERd**. *Given: An undirected graph $G = (N, A)$ and threshold $\bar{n}$. Question: Does there exist a set of nodes, (a "vertex cover," or "node cover"), $N' \subset N$, with $|N'| \leq \bar{n}$, such that every edge $k \in A$ is incident to at least one node in N'?* ∎

Note that N' is a node cover if $G' = G - N'$ consists of a set of completely disconnected nodes.

THEOREM 9 **DCPMd** *is strongly NP-complete.*
Proof: We are given an instance of **VERTEXCOVERd** with $G = (N, A)$ and will show how to construct an instance of **DCPMd** with

project network $G'' = (N'', A'')$ such that N', with $\bar{n} \equiv |N'|$, is a node cover for G if and only if the longest path in $G'' - N'$ has length $\bar{n}$ (where N' has been translated into G'' appropriately). $G'' - N'$ is the solution to an instance of **DCPMd** where $\mathbf{w} \in \mathbf{W}$ simply requires $\sum_{i \in N''} w_i = |N'| - \bar{n}$, $w_i \in \{0, 1\}$ for all $i \in N''$ and $w_a = w_b = 1$.

1 Convert G into a directed acyclic graph $G' = (N, A)$ by orienting arcs appropriately, and place the nodes N in topological ordering $N = \{1, 2, \ldots, n\}$

2 Create N'' by adding to N a set of "parallel" nodes $\{1', 2', \ldots, n'\}$ plus an extra node denoted $(n+1)'$. Node $1'$ will be the project start node, and node $(n+1)'$ will be the project completion node.

3 Define $t_k = 1$ for all $k \in A$.

4 Create A'' by adding to A the following arcs, all with $t_k = 0$;

   (a) $(i', (i+1)')$ for $i = 1, \ldots, n$,
   (b) (i', i) for $i = 1, \ldots, n$, and
   (c) $(i, (i+1)')$ for $i = 1, \ldots, n$.

This construction creates a directed acyclic graph that may be interpreted as a project network. And, a small example should convince the reader that N' is a node cover for G if and only if $G'' - N'$ has a longest path length of 0: (i) If N' is a cover, then $G'' - N'$ contains none of the original edges from A and all paths must have length 0 (and such paths do exist), and (ii) if $G'' - N'$ has a path of length greater than 0, then at least one edge $k = (i, j)$ remains in $G - N'$ so that N' is not a cover.

The fact that we transform a strongly NP-complete problem into **DCPMd**, and do not substantially change the size of the data required to describe the problem implies that **DCPMd** is strongly NP-complete. ∎

So, **MAXMIN0**, even without expediting, is NP-hard and may not be a member of NP. But, when the number of alternative technologies is limited to a few (e.g., Spears 2001), **MAXMIN0** can be solved by solving the ILP **MINMAX2** just a few times. Specifically, enumerate all possible combinations of technologies, i.e., for each feasible vector $\hat{\mathbf{w}} \in \mathbf{W}$, solve the resulting instances of **MAXMAX2**, and choose the best interdiction plan from among those solutions. The instances of **MAXMAX2** would be polynomially solvable, pseudo-polynomially solvable or would be ILPs with exponential worst-case complexity. However, the most difficult of these solution techniques, solving a few ILPs, is likely to be easier than devising an effective, and completely general algorithm for **MAXMIN0**.

7. Conclusions

This paper has investigated the computational complexity of variants of an interdiction model that uses limited resources to delay tasks of an adversary's project in order to delay the project's overall completion time. We show that the most general "project-interdiction problem," and certain variants, are NP-hard. However, we also show that potentially useful variants may be strongly NP-complete, weakly NP-complete, or even solvable in polynomial time.

Furthermore, in practice, the NP-hard problems may not be as difficult as they appear to be at first glance. Their complexity derives from binary variables that model alternative technologies; however, in the real world, the number of such options will often be quite small. For example, if the project's manager must use one of, say, three mutually exclusive technologies, then only three instances of a simpler project-interdiction problem need be solved. Each of these would be an integer-linear program, a dynamic program, or a simple network-optimization problem.

Acknowledgments

Kevin Wood thanks the Naval Postgraduate School and the University of Auckland for their research support. Gerald Brown and Kevin Wood thank the Office of Naval Research and the Air Force Office of Scientific Research for their research support. Johannes Royset expresses his thanks for financial support from the National Research Council's Associateship program.

References

Ahuja, R.K., Magnanti, T.L. and Orlin, J.B. (1993). *Network Flows: Theory, Algorithms, and Applications*: Upper Saddle River, NJ: Prentice-Hall.

Brown, G.G, Carlyle, W.M., Harney, R. C., Skroch, E. M. and Wood, R. K., (2004). "Interdicting a Nuclear Weapons Project," draft, May 9.

Carlyle, W.M. and Wood, R.K., (2003). "Lagrangian Relaxation and Enumeration for Solving Constrained Shortest Paths," *Proceedings of the 38th Annual ORSNZ Conference*, University of Waikato, Hamilton, New Zealand, 21-22 November, pp. 3–12.

Crowston, W. and Thompson, G.L. (1967). "Decision CPM: A Method for Simultaneous Planning, Scheduling, and Control of Projects," *Operations Research*, **15**, pp. 407–426.

De, P., Dunne, E.J., Ghosh, J.B., Wells, C.E. (1997). "Complexity of the Discrete Time-Cost Tradeoff Problem for Project Networks," *Operations Research*, **45**, pp. 302–306.

Israeli, E. and Wood, R.K. (2002). "Shortest-path Network Interdiction," *Networks*, **40**, pp. 97–111.

Elmaghraby, S.E. (1977). *Activity Networks.* New York: Wiley.

Garey, M.R. and Johnson, D.S. (1979). *Computers and Intractability. A Guide to the Theory of NP-Completeness*. New York: W. H. Freeman and Co.

Handler, G.Y. and Zang, I. (1980). A Dual Algorithm for the Constrained Shortest Path Problem, *Networks*, **10**, pp. 293–310.

Hindelang, T.J. and Muth, J.F. (1979) "A Dynamic Programming Algorithm for Decision CPM Networks" *Operations Research*, **27**, pp. 225–241.

Malcolm, D.G., Roseboom, J.H., Clark, C.E., and Fazar, W. (1959) "Application of a Technique for Research and Development Program Evaluation," *Operations Research*, **7**, pp. 646–669.

Moder, J.J., Phillips, C.R., and Davis, E.W. (1983). *Project Management with CPM, PERT and Precedence Diagramming*, 3rd ed. New York: Van Nostrand Reinhold Company Inc.

Moore, J.T. and Bard, J.F. (1990). "The Mixed Integer Linear Bilevel Programming Problem," *Operations Research*, **38**, pp. 911-921.

PERT. (1958). "Program Evaluation Research Task, Phase 1 Summary Report," Special Projects Office, Bureau of Ordinance, 7, Department of the Navy, Washington, D.C., pp. 646-669.

Reed, B. K. (1994). "Models for Proliferation Interdiction Response Analysis," Masters Thesis, Naval Postgraduate School, Monterey, California, September.

Skroch, E. (2004). "How to Optimally Interdict a Belligerent Project to Develop a Nuclear Weapon," Masters Thesis, Naval Postgraduate School, Monterey, California, September.

Spears, D. (ed.). (2001). "Technology R&D for Arms Control. Arms Control and Nonproliferation Technologies", US Department of Energy, National Nuclear Security Administration, Defense Nuclear Nonproliferation Programs, Washington, D.C.

von Stackelberg, H. (1952). *The Theory of the Market Economy*, (trans. from German). London: William Hodge & Co.

Wood, R.K. (1993). "Deterministic Network Interdiction," *Mathematical and Computer Modelling*, **17**, pp. 1–18.