The k-Adaptive ℓ-Most-Vital-Arcs Problem

Luca G. Wrabetz[†] Jourdain Lamperski[†] Oleg A. Prokopyev[‡*]

[†] Department of Industrial Engineering, University of Pittsburgh, Pittsburgh, PA 15261, USA. Emails: {luw28,lamperski}@pitt.edu

[‡] Department of Business Administration, University of Zurich, Zurich, 8006, Switzerland. Email: oleg.prokopyev@business.uzh.ch

March 24, 2026

Abstract

In the celebrated *ℓ-most-vital-arcs problem* (MVA$_\ell$), an *interdictor* seeks to remove ℓ arcs in a given directed graph in order to maximize the length of an *evader's* shortest directed s-t path. We propose and study a generalization of the problem that we refer to as the *k-adaptive ℓ-most-vital-arcs problem* (AVA$_{k,\ell}$), which reflects the practical situation in which the interdictor does not have full information about the evader, but can prepare for uncertainty with *contingency plans*. In AVA$_{k,\ell}$, there are a finite number of possible realizations of the evader's arc costs. Before the costs are realized, the interdictor *precomputes* k interdiction policies (i.e., contingency plans). Then, after the costs are realized, the interdictor executes one of the k precomputed ℓ-arc interdiction policies. The interdictor's aim is to maximize the worst-case (i.e., shortest) shortest s-t path across all realizations of the arc costs. Since MVA$_\ell$ is NP-hard and a special case of AVA$_{k,\ell}$, it follows that AVA$_{k,\ell}$ is NP-hard as well. We show that AVA$_{k,\ell}$ is NP-hard in settings where MVA$_\ell$ is polynomial-time solvable, specifically, when either (i) the graph is *extension-parallel* or (ii) the graph is *series-parallel* and $\ell = 1$. On the flip side, we also show that there is a polynomial-time algorithm for AVA$_{k,\ell}$ when both conditions (i) and (ii) hold simultaneously.

Keywords. Shortest path interdiction, k-adaptive robust optimization, computational complexity, series-parallel graphs.

1 Introduction

In the *ℓ-most-vital-arcs problem* (MVA$_\ell$), an *interdictor* aims to remove ℓ arcs from a given directed graph in order to maximize the length of an *evader's* shortest directed s-t path, i.e., a directed path from a given node s to a given node t in the graph [2, 19]. MVA$_\ell$ is a

*Corresponding author.

fundamental and perhaps the most celebrated problem within the *network interdiction* and *critical elements detection* literature; see, e.g., the survey [20].

Formally, let $G = (N, A)$ be a directed graph, where N and A denote the sets of its nodes and arcs, respectively. Let $s, t \in N$, and assume that the evader traverses the graph along an s-t directed path. We assume that G is a *simple* directed graph, i.e, it has no self-loops and no parallel arcs. Define $c_a \in \mathbb{R}_{\geq 0}$ to be the cost for the evader to traverse arc $a \in A$. We collect the arc costs into the vector $c \in \mathbb{R}^{|A|}_{\geq 0}$.

Next, let A_ℓ denote the set of subsets of A of cardinality at most ℓ. In other words, each element of A_ℓ represents a feasible interdiction policy consisting of at most ℓ arcs. Then, MVA_ℓ is given by:

$$\max_{A' \in A_\ell} \min_{P \in \mathcal{P}(G \setminus A')} \sum_{a \in A(P)} c_a, \tag{MVA_ℓ}$$

where the graph $G \setminus A'$ is the subgraph of G induced by arcs $A \setminus A'$, i.e., $G \setminus A' := (N, A \setminus A')$, the set $\mathcal{P}(G \setminus A')$ is the set of all directed s-t paths in $G \setminus A'$, and the set $A(P)$ is the set of all arcs in the path P. We assume without loss of generality that $G \setminus A'$ contains a directed s-t path for all $A' \in A_\ell$, or, equivalently, no interdiction policy in A_ℓ induces an s-t cut.

In this note, we consider a generalization of MVA_ℓ in which the interdictor does not have full information about the evader. We model this lack of information (or uncertainty) by considering m possible scenarios (realizations) of the evader's arc costs given by $c^{(1)}, \ldots, c^{(m)} \in \mathbb{R}^{|A|}_{\geq 0}$. Alternatively, one can interpret this network interdiction setting as the problem with m distinct evaders, exactly one of whom (unknown to the interdictor) attempts to traverse the graph G from node s to node t.

Given this setup, it is natural to consider a *robust* variation of the ℓ-most-vital-arcs problem in which the interdictor aims to maximize the worst-case (i.e., shortest) shortest directed s-t path across all realizations of the evader's arc costs. This problem, which we refer to as the *robust ℓ-most-vital-arcs problem* (RVA_ℓ), is given by:

$$\max_{A' \in A_\ell} \min_{i \in [m]} Z_i(G \setminus A'), \tag{RVA_ℓ}$$

where $[m] := \{1, \ldots, m\}$, and $Z_i(G \setminus A') := \min_{P \in \mathcal{P}(G \setminus A')} \sum_{a \in A(P)} c^{(i)}_a$.

However, in some application settings, the interdictor can *adapt* to uncertainty via *contingency plans*. That is, before the evader's arc costs are realized, the interdictor *precomputes* k interdiction policies $A^{(1)}, \ldots, A^{(k)} \in A_\ell$ (i.e., contingency plans). Once the costs are revealed, the interdictor has capability to implement one of the k precomputed ℓ-arc interdiction policies. Ultimately, the interdictor's aim is to maximize the worst-case (i.e., shortest) shortest directed s-t path across the realizations of the evader's arc costs, just as in RVA_ℓ. Formally, we study the *k-adaptive ℓ-most-vital-arcs problem* ($\text{AVA}_{k,\ell}$), which is given by:

$$\max_{A^{(1)}, \ldots, A^{(k)} \in A_\ell} \min_{i \in [m]} \max_{j \in [k]} Z_i(G \setminus A^{(j)}), \tag{$\text{AVA}_{k,\ell}$}$$

and one can observe that RVA_ℓ is a special case of $\text{AVA}_{k,\ell}$ with $k = 1$.

Naturally, in the *ideal* situation for the interdictor, we have $k = m$. Then, $\text{AVA}_{k,\ell}$ simply reduces to solving m copies of MVA_ℓ, and the interdictor applies an optimal interdiction policy for each evasion cost scenario (or, equivalently, for each distinct evader). However, from the practical perspective, it is rarely feasible to prepare for every possible realization of the evader's costs. In real-world security and defense applications, the decision-makers prefer to maintain only a limited number of contingency plans. Indeed, implementing and coordinating a large set of interdiction policies may result in an extremely complex task, which can be resource-intensive and prone to errors. This trade-off motivates the study of the k-adaptive setting, where the interdictor balances the overall solution quality with the practical need for a reasonably small and manageable set of interdiction policies.

Note that $\text{AVA}_{k,\ell}$ bears a close connection to the *k-adaptive robust optimization* literature [7]. The key difference is that the underlying *nominal problem* in $\text{AVA}_{k,\ell}$, i.e., MVA_ℓ, is a bilevel optimization problem, instead of a single-level optimization problem. In particular, in MVA_ℓ the interdictor acts as the upper-level decision-maker (the leader), and given the leader's decision (interdiction), the evader (the follower) solves its own lower-level optimization problem (the shortest path problem).

The motivation behind k-adaptive robust optimization is to address the potentially overly conservative nature of robust optimization; see [1, 4, 5, 6, 9, 10, 11, 14, 18]. In the same spirit, the potential advantage of $\text{AVA}_{k,\ell}$ over RVA_ℓ is also evident. If the interdictor has the capability to precompute contingency plans and to select one after the uncertainty is realized, performance may improve significantly.

To this end, consider the example depicted in Figure 1. There are $m = 3$ evaders (or cost scenarios). Their arc traversal costs and shortest s-t paths before interdiction are depicted in the first row of the figure. Note that the worst-case (i.e., shortest) shortest s-t path length before interdiction equals $\min\{6, 5, 6\} = 5$. Suppose that the interdictor can remove one arc from the graph, i.e., $\ell = 1$. We illustrate the shortest s-t paths under the optimal robust 1-arc interdiction policy $A' = \{(s, 1)\}$ (i.e., optimal solution to RVA_1) in the second row of Figure 1. The worst-case shortest path length increases slightly to $\min\{26, 28, 6\} = 6$. Suppose that the interdictor can precompute $k = 2$ interdiction policies and choose one to implement once uncertainty (the evader) is revealed (e.g., perhaps there is surveillance installed around node s that captures which evader appears). We illustrate the shortest s-t paths under the optimal adaptive 1-arc interdiction policy $A^{(1)} = \{(s, 1)\}$, $A^{(2)} = \{(2, 4)\}$ (i.e., the optimal solution to $\text{AVA}_{2,1}$ in the third row of Figure 1). The worst-case shortest s-t path length increases significantly to $\min\{26, 28, 18\} = 18$. Thus, at least in this case, adaptive interdiction strategies provide an attractive alternative to the corresponding static robust approach.

The MVA_ℓ problem is known to be NP-hard [2]. Clearly, it follows that $\text{AVA}_{k,\ell}$ is NP-hard because MVA_ℓ is a special case of $\text{AVA}_{k,\ell}$ with $k = m = 1$. Intuitively, $\text{AVA}_{k,\ell}$ should be somewhat "harder" than MVA_ℓ, which raises a natural question: how much harder is $\text{AVA}_{k,\ell}$, if at all, in settings, where MVA_ℓ is computationally "easy"? To address this question, we explore the computational complexity of $\text{AVA}_{k,\ell}$ in two specific settings in which MVA_ℓ is known to be polynomial-time solvable.

Specifically, we restrict our attention to the case in which G is a *series-parallel* or an *extension-parallel* graph. Extension–parallel graphs form a special case (subclass) of series–parallel graphs. Series-parallel and extension-parallel graphs are considered as natural special cases in the network interdiction literature [3, 16]. MVA_ℓ remains NP-hard on series-parallel graphs [2], but there is a polynomial-time algorithm for MVA_ℓ on extension-parallel graphs [3]. The contribution of this note can be summarized as follows:

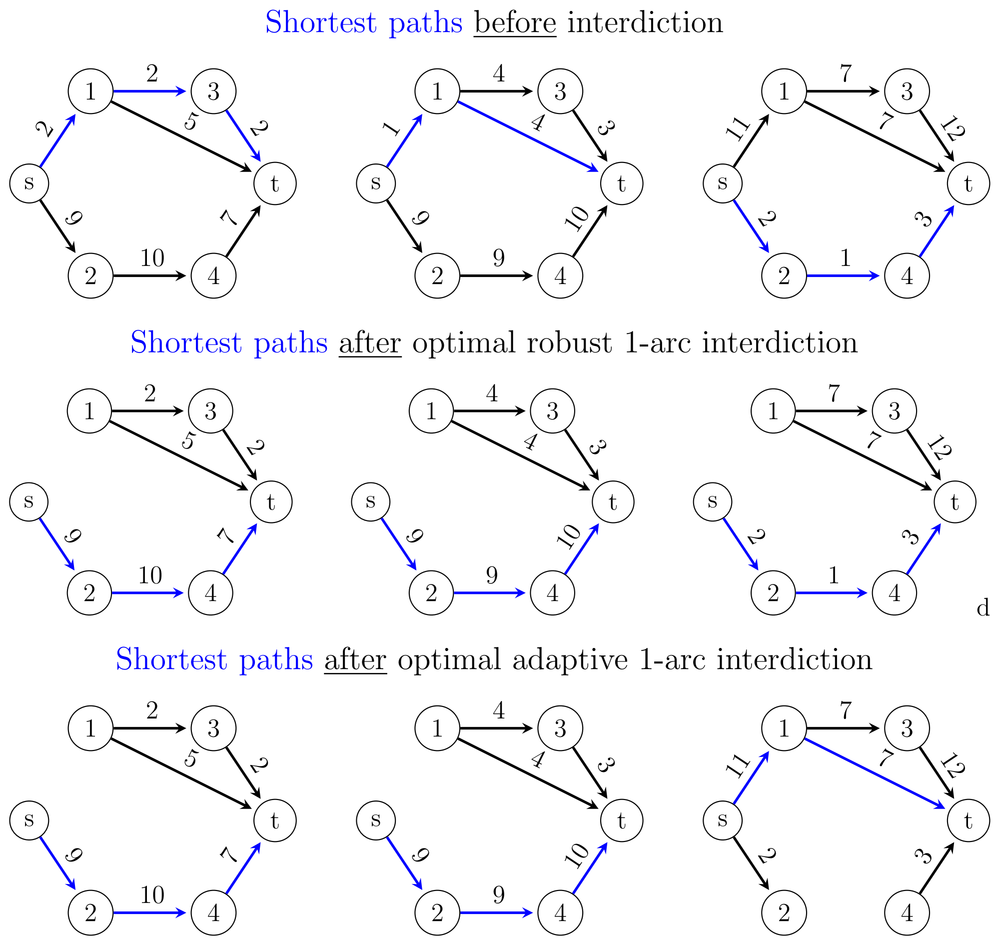

Figure 1: Illustration of an example of three evader's shortest paths (in blue) before interdiction (first row), shortest paths after optimal robust interdiction (second row), and shortest paths after optimal adaptive ($k = 2$) interdiction (last row).

- In Section 2.1, we establish that $\text{AVA}_{k,\ell}$ remains NP-hard when G is extension-parallel.
- In Section 2.2, we show that $\text{AVA}_{k,\ell}$ remains NP-hard when G is series-parallel and $\ell = 1$. Note that both MVA_1 and RVA_1 are trivially polynomial-time solvable for any graph. Indeed, there are only $|A|$ possible arcs to interdict, so we can enumerate all possible choices in polynomial time.
- On the flip side, in Section 3, we show that there is a polynomial-time greedy-like algorithm for $\text{AVA}_{k,\ell}$ when G is extension-parallel and $\ell = 1$.

Next, we review the formal definitions of these two graph structures. Every series-parallel graph $G = (N, A)$ has a unique source node $s \in N$ and sink node $t \in N$. To distinguish these nodes, we write $G = (N, A, s, t)$. We assume that the nodes s and t given in $\text{AVA}_{k,\ell}$ are the unique source and sink node, respectively, of the series-parallel graph. The one-arc directed graph (i.e., that has two nodes and one arc that is incident to both nodes) is a series-parallel graph that serves as the building block for all other series-parallel graphs. Definition 1.1

below states a recursive definition for a series-parallel graph, where the one-arc directed graph provides the "base case."

Definition 1.1. A *series-parallel graph* is either a one-arc directed graph or a directed graph that can be obtained by applying a sequence of the following two operations.

(i) *Series composition:*

- INPUT: Series-parallel graphs $G_1 = (N_1, A_1, s_1, t_1)$ and $G_2 = (N_2, A_2, t_1, t_2)$ such that $N_1 \cap N_2 = \{t_1\}$.
- OUTPUT: Series-parallel graph $G = (N_1 \cup N_2, A_1 \cup A_2, s_1, t_2)$.

(ii) *Parallel composition:*

- INPUT: Series-parallel graphs $G_1 = (N_1, A_1, s, t)$ and $G_2 = (N_2, A_2, s, t)$ such that $N_1 \cap N_2 = \{s, t\}$.
- OUTPUT: Series-parallel graph $G = (N_1 \cup N_2, A_1 \cup A_2, s, t)$.

Definition 1.2. A series-parallel graph is an *extension-parallel graph* if at least one of the input graphs in every series composition of its construction is a one-arc directed graph.

We introduce binary tree $\mathcal{T}_G = (\mathcal{H}_G, E_G)$ that captures the construction of series-parallel graph G. We refer to $\mathcal{T}_G$ as the *decomposition tree* of G. Each node $H \in \mathcal{H}_G$ in $\mathcal{T}_G$ is a series-parallel subgraph of G. Each leaf $H \in \mathcal{H}_G$ in $\mathcal{T}_G$ corresponds to a single arc within the graph G. All other (internal) nodes $H \in \mathcal{H}_G$ are the series or parallel composition of their two children.

The remainder of this note is organized as follows. In Section 2, we establish the outlined hardness results for $\text{AVA}_{k,\ell}$ when (i) graph G is extension-parallel, or (ii) graph G is series-parallel and $\ell = 1$. In Section 3, we introduce an exact, polynomial-time greedy algorithm for $\text{AVA}_{k,\ell}$ when both conditions (i) and (ii) hold simultaneously, i.e., G is extension-parallel and $\ell = 1$. Finally, we conclude the paper with some additional remarks in Section 4.

2 Computational complexity of $\text{AVA}_{k,\ell}$

In Section 2.1, we establish the hardness of $\text{AVA}_{k,\ell}$ on extension-parallel graphs via a reduction from the bin-packing problem to the decision version of $\text{AVA}_{k,\ell}$. In Section 2.2, we demonstrate the same result for $\ell = 1$ on series-parallel graphs via a reduction from the set-cover problem to the decision version of $\text{AVA}_{k,\ell}$.

2.1 Extension-parallel graphs

First, we formalize the decision version of $\text{AVA}_{k,\ell}$, namely, $\text{AVA}_{k,\ell}$-DECISION. Next, we restate the classical BIN PACKING problem, which is known to be strongly NP-complete [13]. Finally, we establish the NP-completeness of $\text{AVA}_{k,\ell}$-DECISION on extension-parallel graphs.

$\text{AVA}_{k,\ell}$-DECISION:

Instance. A graph $G = (N, A)$, nodes $s, t \in N$, positive integers $k, m, \ell, \alpha \in \mathbb{Z}_{>0}$, and the evader's traversal costs $c^{(i)} \in \mathbb{Z}^{|A|}_{\geq 0}$, $i \in [m]$.
Question. Do there exist k interdiction policies $A^{(1)}, \ldots, A^{(k)} \in A_\ell$ such that

$$\min_{i \in [m]} \max_{j \in [k]} Z_i(G \setminus A^{(j)}) \geq \alpha?$$

BIN PACKING:

Instance: A finite set of items $[m]$, a size $s_i \in \mathbb{Z}_{>0}$ for each item $i \in [m]$, a positive integer bin capacity $\beta \in \mathbb{Z}_{>0}$ and a number of bins $k \in \mathbb{Z}_{>0}$.
Question: Is there a partition of $[m]$ into k disjoint subsets $I_1, \ldots, I_k$ such that the sum of the sizes of the items in each subset I_j, $j \in [k]$, is at most β?

Theorem 2.1. $\text{AVA}_{k,\ell}$-DECISION *is NP-complete on extension-parallel graphs.*

Proof. Given an instance of BIN PACKING, we construct the following instance of $\text{AVA}_{k,\ell}$-DECISION. Define:

$$N := \{s\} \cup \{t\} \cup V^{(1)} \cup V^{(2)} \cup \ldots \cup V^{(m)},$$

where $V^{(i)} := \{v_q^{(i)} \mid q = 1, \ldots, s_i\}$ for all $i \in [m]$. Next, we connect source node s with each node in $V^{(i)}$ for all $i \in [m]$ by an arc, which are then also connected by an arc to t. In other words, we let $A := A^{(s)} \cup A^{(t)}$, where

$$A^{(s)} := \bigcup_{i \in [m]} \left\{(s, u) \mid u \in V^{(i)}\right\}, \text{ and } A^{(t)} := \bigcup_{i \in [m]} \left\{(u, t) \mid u \in V^{(i)}\right\},$$

which implies that $|N| = 2 + \sum_{i \in [m]} s_i$ and $|A| = 2 \cdot \sum_{i \in [m]} s_i$. Recall that the BIN PACKING problem is strongly NP-complete. That is, the problem remains NP-complete even if the item sizes and bin capacities are bounded by a polynomial in the number of items. Hence, the considered reduction remains polynomial as long as the item sizes s_i are bounded by a polynomial in m for all $i \in [m]$.

Furthermore, we observe that:

(i) G is an extension-parallel graph;

(ii) G contains $\sum_{i \in [m]} s_i$ arc-disjoint directed paths from s to t, and each path contains exactly 2 arcs.

Next, assume that the number of the evaders is equal to the number of items, i.e., m. Denote every path from s to t by $P_q^{(i)} := \{(s, v_q^{(i)}), (v_q^{(i)}, t)\}$, where $i \in [m]$ and $q \in [s_i]$. Then, define the cost structure for evader $i \in [m]$ as follows:

- for all $q \in [s_i]$ and for both arcs $a \in P_q^{(i)}$, set $c_a^{(i)} = 0$;
- for all $q \in [s_h]$, $h \neq i$, and for both arcs $a \in P_q^{(h)}$, set $c_a^{(i)} = 1/2$.

The outlined cost setting implies that each evader i has s_i arc-disjoint directed paths from s to t of cost 0, and each of these paths contains a node from $V^{(i)}$. The cost of each of the remaining arc-disjoint paths for the evader is equal to 1. Finally, we let $\ell = \beta$ and $\alpha = 1$.

- Let Bin Packing return "yes." That is, there exists a partition of $[m]$ into k disjoint subsets $I_1, \ldots, I_k$ such that the sum of the sizes of the items in each subset I_j, $j \in [k]$, is at most β. Then, we design the following interdiction policy for every $j \in [k]$: for all i belonging to the set I_j in the Bin Packing solution, we interdict all s_i arcs of the form $(s, v_h^{(i)})$, where $h \in [s_i]$. Since $\sum_{i \in I_j} s_i \leq \beta = \ell$, the interdictor can (adaptively) remove the necessary arcs to interdict all paths of cost 0 for all m evaders, due to our construction. Therefore, AVA$_{k,\ell}$-decision returns "yes."

- Conversely, if AVA$_{k,\ell}$-decision returns "yes," then $\sum_{a \in P} c_a^{(i)} \geq 1$, for all $i \in [m]$, where P is a shortest s-t path in G after interdiction by policy j that corresponds to evader i. By our construction, this implies that all paths $P_h^{(i)}$, where $h \in [s_i]$, are interdicted for evader i. Furthermore, recall that all paths from s to t are arc-disjoint. Hence, we can form set I_j by assigning to it all items s_i, which correspond to evaders that are interdicted using policy $A^{(j)}$. Given that $\beta = \ell$, its clear that $\sum_{i \in I_j} s_i \leq \beta$ for all $j \in [k]$, so Bin Packing returns "yes."

The proof is complete. □

Finally, Theorem 2.1 implies that:

Corollary 2.1. AVA$_{k,\ell}$ *is NP-hard on extension-parallel graphs.*

2.2 Series-parallel graphs and unit interdiction budget

First, we introduce AVA$_{k,1}$-decision, the decision version of AVA$_{k,\ell}$ with $\ell = 1$. Next, we restate the classical Set Covering problem, which is known to be NP-complete [13]. Finally, we establish the NP-completeness of AVA$_{k,1}$-decision on series-parallel graphs.

AVA$_{k,1}$-decision

Instance. A graph $G = (N, A)$, nodes $s, t \in N$, strictly positive integers k, m, α, where $k > m$, and costs $c_a^{(i)}$ for $i \in [m]$ to traverse arc a.
Question. Do there exist k arcs $a_1, \ldots, a_k \in A$ such that $\min_{i \in [m]} \max_{j \in [k]} Z_i(G \setminus \{a_j\}) \geq \alpha$?

Set Covering

Instance. A universal set of elements, $U = \{1, \ldots, m\}$, subsets $S_1, \ldots, S_p \subseteq U$, and an integer $\beta > 0$.
Question. Does there exist a valid cover $\mathcal{C} \subseteq \{S_1, \ldots, S_p\}$ such that $|\mathcal{C}| \leq \beta$?

Theorem 2.2. AVA$_{k,1}$-decision *is NP-complete on series-parallel graphs.*

Proof. For a given instance of Set Covering, we construct an instance of AVA$_{k,1}$-decision as follows. We let $N := V^{(s)} \cup V^{(u)} \cup V^{(v)}$, where $V^{(s)} := \{s_1, \ldots, s_p\}$, $V^{(u)} := \{u_1, \ldots, u_p\}$, and $V^{(v)} := \{v_1 \ldots, v_p\}$. Define $A := A^{(s)} \cup A^{(u)} \cup A^{(v)}$, where

$$A^{(s)} := \{(s_1, u_1), (s_1, v_1), \ldots, (s_p, u_p), (s_p, v_p)\},$$
$$A^{(u)} := \{(u_1, s_2), \ldots, (u_{p-1}, s_p), (u_p, t)\},$$
$$A^{(v)} := \{(v_1, s_2), \ldots, (v_{p-1}, s_p), (v_p, t)\}.$$

Note that the constructed graph is the result of a *series composition of p two-terminal series-parallel graphs*. That is, the final graph denoted by $G = G_1 \circ \ldots \circ G_p$ (where $\circ$ denotes a series composition operation), is a series-parallel graph of the form given in Figure 2.

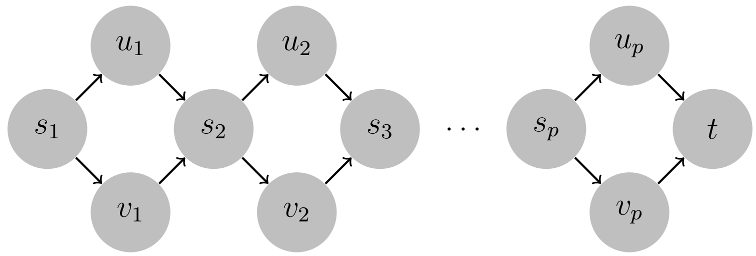

Figure 2: Construction of $\text{AVA}_{k,1}$-DECISION given SET COVERING.

We let $k = \beta$ and $\alpha = 1$ in our $\text{AVA}_{k,1}$-DECISION instance. Finally, we outline the cost structure for the instance. For all arcs in $A^{(s)} \cup A^{(u)}$, we let $c_a^{(i)} := 0$, for all $i \in [m]$. For all arcs $a := (v_q, s_{q+1}) \in A^{(v)}$, where $s_{p+1} := t$, we let

$$c_a^{(i)} := \begin{cases} 1, & i \in S_q, \\ 0, & i \notin S_q. \end{cases}$$

Some additional notation specific to the outlined construction allows us to be slightly more informal when we describe solutions to the problem. For any $q \in [p]$, we let $P_u^{(q)} := s_q, u_q, t_q$ and $P_v^{(q)} := s_q, v_q, t_q$, where $t_q := s_{q+1}$ for $q = 1, \ldots, p-1$ and $t_q := t$ for $q = p$. Further, we note that in this instance the attacker's decision simply corresponds to a selection of the subgraphs $G_1, \ldots, G_p$, or more precisely, of their paths $P_u^{(1)}, \ldots, P_u^{(p)}$. In every such subgraph, interdicting any of the arcs in $P_u^{(q)}$ removes the path from the graph, so we simply refer to the interdiction of the entire path for the rest of the proof.

We preface the final part of the proof with several observations about the constructed $\text{AVA}_{k,1}$-DECISION instance and the problem statement:

(*i*) Under this construction, the uninterdicted objective for any evader i is $Z_i(G) = 0$.

(*ii*) If an evader i is forced to traverse a single path $P_v^{(q)}$ where $i \in S_q$ in the original SET COVERING instance, then the evader's objective value is 1, as $c_i(P_v^{(q)}) = 1$ by the construction. Further, we know it is exactly 1 in this case because $\ell = 1$.

(*iii*) The $\text{AVA}_{k,1}$-DECISION instance is a "yes" instance if and only if there exists a solution that increases *every* evader's shortest path to 1.

(*iv*) In a "yes" solution to $\text{AVA}_{k,1}$-DECISION, for any evader q, there is exactly one arc a in the evasion path such that $c_a^{(q)} = 1$, which follows directly from (*i*), (*ii*), and (*iii*) above.

We are prepared to establish our result:

- Let Set Covering return “yes”. It follows from the problem definition that we have valid cover, i.e., a set $\mathcal{C} = \{S_1, \ldots, S_k\}$ such that $S_1 \cup \ldots \cup S_k = U$. Consider the following solution to $\text{AVA}_{k,1}$: for every index $q \in [p]$, if $S_q \in \mathcal{C}$, then we interdict path $P_u^{(q)}$. The $\text{AVA}_{k,1}$ instance is a “yes” instance, as observations (*ii*) and (*iii*) above are satisfied.

- Let $\text{AVA}_{k,1}$ return “yes”. We can construct a set cover solution by including a subset S_q in $\mathcal{C}$ only if $P_u^{(q)}$ is interdicted in the $\text{AVA}_{k,1}$ solution. Observation (*iv*) above, along with the construction of the graph cost structure, imply that $\mathcal{C}$ covers U, and Set Covering returns “yes”.

The proof is complete. □

Finally, Theorem 2.2 implies that:

Corollary 2.2. $\text{AVA}_{k,\ell}$ *is NP-hard on series-parallel graphs even when* $\ell = 1$.

3 Polynomial-time algorithm for extension-parallel graphs under unit interdiction budget

In this section we develop and study a greedy algorithm for solving problem $\text{AVA}_{k,\ell}$ under unit interdiction budget, i.e., $\ell = 1$. We present a description of the greedy algorithm in Algorithm 1. Note that, since $\ell = 1$, we simply refer to “policies” as “arcs” when discussing the algorithm. Indeed, each policy corresponds to interdicting exactly one arc. Also, we use $\text{MVA}_1^{(i)}$ to denote the problem MVA_1 for the i-th evasion cost scenario, i.e., the i-th evader.

The algorithm initializes an empty set of follower indices, U, and an empty set of arcs, $\hat{A}$, in Step 1. In Steps 2-7, the algorithm adds arcs to $\hat{A}$ (and evader indices to U) until it contains k arcs (and U contains k evader's indices). At every iteration, the algorithm chooses an evader i in Step 3 that currently has the shortest path in G when adaptively interdicted by the arcs available in $\hat{A}$, and adds the optimal solution to $\text{MVA}_1^{(i)}$ to the arc set.

Algorithm 1 Greedy Algorithm

1: $U \leftarrow \{\}, \hat{A} \leftarrow \{\}$
2: **for** $j = 1, \ldots, k$ **do**
3: $\quad i \leftarrow \operatorname{argmin}_{i' \in [m] \setminus U} \max_{a \in \hat{A}} Z_{i'}(G \setminus \{a\})$
4: $\quad U \leftarrow U \cup \{i\}$
5: $\quad a^{(j)} \leftarrow$ optimal solution of $\text{MVA}_1^{(i)}$
6: $\quad \hat{A} \leftarrow \hat{A} \cup \{a^{(j)}\}$
7: **end for**
8: **return** $\hat{A} = \{a^{(1)}, \ldots, a^{(k)}\}$

Relation to the greedy algorithm for the k-center problem. The k-center problem is a fundamental problem in the computer science domain that finds various applications in, for example, facility location [8] and clustering [12, 17]. Given m points in a metric space, the goal of the problem is to choose k points to be *centers* such that the maximum distance (under the metric) between a point and its closest center is minimized. Algorithm 1 can be viewed as an adaptation of the classic greedy 2-approximation algorithm for the k-center problem proposed in [15]. The algorithm can be summarized as follows. First, it chooses any point to be the first center. Then, at each iteration, it chooses the next center to be the point that is farthest from the centers chosen thus far. The algorithm terminates once k centers have been chosen. Notice that Algorithm 1 follows a similar structure, using interdiction policies (i.e., solutions to problem $\text{MVA}_1^{(i)}$) as the "points". In Step 3, we choose the evader, who is, in a sense, "worst" interdicted by the current set of policies, just like the k-center algorithm chooses the point that is farthest from the current centers.

Computational complexity. The worst-case running time of the algorithm is polynomial in the size of the input. Indeed, the bottlenecks are in Step 5 and Step 3. In Step 5, we must find an optimal solution of MVA_1 for a given evader, which is trivially solvable by enumeration. Step 3 involves solving $O(m)$ shortest path problems. Finally, note that the Algorithm can be implemented more efficiently by pre-computing and sorting all of the evader's shortest paths first, but we are only concerned with showing that the running time is polynomial, rather than further improvements, due to the scope and focus of this work.

In the remainder of this section, we show that Algorithm 1 outputs an optimal solution to Problem $\text{AVA}_{k,\ell}$ under the assumptions that (i) G is an extension-parallel graph and (ii) $\ell = 1$. Let

$$P_a(G) := \{P \in \mathcal{P}(G) : a \in A(P)\}$$

denote the set of directed s-t paths in G that contain arc $a \in A(P)$.

As a starting point, we establish the a basic property of extension-parallel graphs; see Lemma 3.1 below. The property implies that removing either the first or last arc in any s-t path of an extension-parallel graph provides at least as strong of a 1-arc interdiction policy as removing any other arc in the path.

Lemma 3.1. *Suppose that graph G is an extension-parallel graph. Let $P \in \mathcal{P}(G)$ and $a \in A(P)$. Then at least one of the following two statements holds.*

(i) $\mathcal{P}_a(G) \subseteq \mathcal{P}_{a_f}(G)$, where a_f is the first arc in path P.

(ii) $\mathcal{P}_a(G) \subseteq \mathcal{P}_{a_\ell}(G)$, where a_ℓ is the last arc in path P.

Proof. Consider the decomposition tree $\mathcal{T}_G = (\mathcal{H}_G, \mathcal{E}_G)$ of G. Let $H \in \mathcal{H}_G$ denote the smallest (say in terms of number of nodes) extension-parallel subgraph $H \in \mathcal{H}_G$ of G within $\mathcal{H}_G$ that contains the directed s-t path P. If H is a leaf of $\mathcal{T}_G$, then H is an arc in G, in which case the desired result trivially holds, so assume that H is not a leaf in $\mathcal{T}_G$. Because H is the smallest subgraph in $\mathcal{H}_G$ that contains P, it must be the case that H is a series-composition of its two children in $\mathcal{T}_G$ (otherwise one child of H would contain P, contradicting the fact that H is the smallest subgraph that contains P). It follows that

$$\mathcal{P}_a(G) = \mathcal{P}_a(H),$$

and it also follows that

$$\mathcal{P}_{a_f}(H) = \mathcal{P}(H) \quad \text{or} \quad \mathcal{P}_{a_\ell}(H) = \mathcal{P}(H)$$

given that H is extension-parallel (at least one child of H in $\mathcal{T}_G$ is a single arc). Suppose that $\mathcal{P}_{a_f}(H) = \mathcal{P}(H)$. Then,

$$\mathcal{P}_a(G) = \mathcal{P}_a(H) \subseteq \mathcal{P}(H) = \mathcal{P}_{a_f}(H) \subseteq \mathcal{P}_{a_f}(G).$$

The same argument applies when $\mathcal{P}_{a_\ell}(H) = \mathcal{P}(H)$. Thus, the proof is complete. □

Next we introduce and consider the concept of a *blocker arc*:

Definition 3.1. We say that arc $a \in A$ is a *blocker arc* if there exists an s-t path $P \in \mathcal{P}_a(G)$ such that $\mathcal{P}_{a'}(G) \subseteq \mathcal{P}_a(G)$ for all $a' \in A(P)$. We refer to path P as a *relative blocking* path for arc a.

From Lemma 3.1, if G is extension-parallel, then every s-t path $P \in \mathcal{P}(G)$ contains a blocker arc $a \in A(P)$ with P as its relative blocking path. Consequently, there is always a blocker arc that is optimal for $\text{MVA}_a^{(i)}$ for each $i \in [m]$.

We are now prepared to show that Algorithm 1 outputs an optimal solution to Problem $\text{AVA}_{k,\ell}$ when (i) G is an extension-parallel graph and (ii) $\ell = 1$. Without loss of generality, we assume that a blocker arc is always selected in Step 5.

Theorem 3.1. *Suppose that G is extension-parallel and $\ell = 1$. Further suppose that Algorithm 1 selects a blocker arc in Step 5 in all k iterations. Then Algorithm 1 is an exact algorithm for Problem* $\text{AVA}_{k,\ell}$.

Proof. First we establish some notation. Let $a^{(1*)}, \ldots, a^{(k*)}$ denote an optimal solution to $\text{AVA}_{k,\ell}$. The optimal objective value of $\text{AVA}_{k,\ell}$ is then given by

$$f^* := \min_{i \in [m]} \max_{j \in [k]} Z(G \setminus a^{(j*)}),$$

and *optimal evader clusters* are given by

$$C_j^* := \Big\{ i \in [m] : j \in \operatorname*{argmax}_{j' \in [k]} Z_i(G \setminus a^{(j'*)}) \Big\}, \quad j \in [k]. \tag{1}$$

For each $j \in [k]$, let i_j denote the evader selected in Step 3 of the j-th iteration of Algorithm 1. It follows that $a^{(j)}$ is an optimal solution for $\text{MVA}_1^{(i_j)}$. Let

$$\hat{f} := \min_{i \in [m]} \max_{j \in [k]} Z(G \setminus a^{(j)})$$

denote the objective value of the output solution $a^{(1)}, \ldots, a^{(k)}$ of Algorithm 1. Let P_j denote the relative blocker path for arc $a^{(j)}$.

Next we make note of an implication of Lemma 3.1. For $j, u \in [k]$, if $i_u \in C_j^*$ and $a^{(j)} \in A(P_u)$, then

$$Z_i(G \setminus a^{(u)}) \geq Z_i(G \setminus a^{(j*)}) \tag{2}$$

for each $i \in C_j^*$ because arc $a^{(u)}$ is a blocker arc with relative blocking path P_u (by supposition; see the theorem statement).

It is sufficient to show that $\hat{f} \geq f^*$. We consider two cases:

Case 1. Suppose that there exist $j, u \in [k]$ such that $i_u \in C_j^*$ and $a^{(j*)} \notin A(P_u)$. Then evader i_u can traverse a shortest path (for evader i_u) at optimality, so

$$f^* \leq Z_{i_u}(G) \leq \hat{f},$$

where the second inequality follows from Step 3 of Algorithm 1.

Case 2. Suppose that $a^{(j^*)} \in A(P_u)$ for any $j, u \in [k]$ such that $i_u \in C_j^*$. We consider two subcases:

Case 2.1. Suppose that each optimal evader cluster (as defined in (1)) contains exactly one of the evaders $i_1, \ldots, i_k$. So, for each $j \in [k]$, there exists some $u(j) \in [k]$ such that $i_{u(j)} \in C_j^*$. Let $i \in [m]$, and take $j \in [k]$ such that $i \in C_j^*$. Then, we have that

$$\max_{v \in [k]} Z_i(G \setminus a^{(v)}) \geq Z_i(G \setminus a^{(u(j))}) \geq Z_i(G \setminus a^{(j*)}) \geq f^*, \tag{3}$$

where the second inequality follows from (2) (which we can apply under the supposition of **Case 2**), and the last inequality follows from $i \in C_j^*$. Because inequality (3) holds for all $i \in [m]$, we conclude that $\hat{f} = \min_{i \in [m]} \max_{v \in [k]} Z_i(G \setminus a^{(v)}) \geq f^*$.

Case 2.2. Suppose that an optimal evader cluster (as defined in (1)) contains more than one evader. So, there exists $j \in [k]$ such that $i_u, i_v \in C_j^*$ for $u < v \in [k]$. From Step 3 of Algorithm 1,

$$\hat{f} \geq \min_{i \in C_j^*} Z_{i_v}(G \setminus a^{(u)}) \geq \min_{i \in C_j^*} Z_{i_v}(G \setminus a^{(j^*)}) \geq f^*,$$

where the second inequality follows from (2) (which we can apply under the supposition of **Case 2**). □

4 Conclusion

We introduce a generalization of the *robust ℓ-most-vital-arcs problem*, namely, the *k-adaptive ℓ-most-vital-arcs problem* ($\text{AVA}_{k,\ell}$), which allows for k contingency interdiction plans to be computed. The classic ℓ-most-vital-arcs problem (MVA_ℓ), which is NP-hard even on series-parallel graphs, is known to be tractable in two special cases of interest. Namely, MVA_ℓ is polynomial-time solvable when $\ell = 1$, or when the underlying graph is extension-parallel. In this note, we show that $\text{AVA}_{k,\ell}$ is difficult when either (i) the graph is extension-parallel or (ii) the graph is series-parallel and $\ell = 1$. On the other hand, we also show that there is a polynomial-time greedy algorithm for $\text{AVA}_{k,\ell}$ when both conditions (i) and (ii) hold simultaneously. As an intriguing direction for future research, it could be of interest to explore whether our greedy algorithm provides some approximation guarantees for more general graph classes.

References

[1] A. N. Arslan, M. Poss, and M. Silva, *Min-sup-min robust combinatorial optimization with few recourse solutions*, INFORMS Journal on Computing, 34 (2022), pp. 2212–2228.

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

[3] J. Boeckmann, C. Thielen, and A. Wittmann, *Complexity of the temporal shortest path interdiction problem*, in 2nd Symposium on Algorithmic Foundations of Dynamic Networks (SAND 2023), Schloss-Dagstuhl-Leibniz Zentrum für Informatik, 2023.

[4] C. Buchheim and J. Kurtz, *Min–max–min robustness: A new approach to combinatorial optimization under uncertainty based on multiple solutions*, Electronic Notes in Discrete Mathematics, 52 (2016), pp. 45–52. INOC 2015 – 7th International Network Optimization Conference.

[5] C. Buchheim and J. Kurtz, *Min–max–min robust combinatorial optimization*, Mathematical Programming, 163 (2017), pp. 1–23.

[6] C. Buchheim and J. Kurtz, *Complexity of min–max–min robustness for combinatorial optimization under discrete uncertainty*, Discrete Optimization, 28 (2018), pp. 1–15.

[7] C. Buchheim and J. Kurtz, *Robust combinatorial optimization under convex and discrete cost uncertainty*, EURO Journal on Computational Optimization, 6 (2018), pp. 211–238.

[8] M. Charikar and S. Guha, *Improved combinatorial algorithms for facility location problems*, SIAM Journal on Computing, 34 (2005), pp. 803–824.

[9] A. Chassein and M. Goerigk, *On the complexity of min-max-min robustness with two alternatives and budgeted uncertainty*, Discrete Applied Mathematics, 296 (2021), pp. 141–163.

[10] A. Chassein, M. Goerigk, J. Kurtz, and M. Poss, *Faster algorithms for min-max-min robustness for combinatorial problems with budgeted uncertainty*, European Journal of Operational Research, 279 (2019), pp. 308–319.

[11] A. Crema, *Min max min robust (relative) regret combinatorial optimization*, Mathematical Methods of Operations Research, 92 (2020), pp. 249–283.

[12] M. Ester, R. Ge, B. J. Gao, Z. Hu, and B. Ben-Moshe, *Joint cluster analysis of attribute data and relationship data: the connected k-center problem*, in Proceedings of the 2006 SIAM International Conference on Data Mining, SIAM, 2006, pp. 246–257.

[13] M. R. Garey and D. S. Johnson, *Computers and Intractability: A Guide to the Theory of NP-Completeness*, W. H. Freeman and Company, San Francisco, CA, USA, 1979.

[14] M. Goerigk, J. Kurtz, and M. Poss, *Min–max–min robustness for combinatorial problems with discrete budgeted uncertainty*, Discrete Applied Mathematics, 285 (2020), pp. 707–725.

[15] T. F. Gonzalez, *Clustering to minimize the maximum intercluster distance*, Theoretical Computer Science, 38 (1985), pp. 293–306.

[16] T. Holzmann and J. C. Smith, *Shortest path interdiction problem with arc improvement recourse: A multiobjective approach*, Naval Research Logistics (NRL), 66 (2019), pp. 230–252.

[17] M. Kleindessner, P. Awasthi, and J. Morgenstern, *Fair k-center clustering for data summarization*, in International Conference on Machine Learning, PMLR, 2019, pp. 3448–3457.

[18] J. Kurtz, *New complexity results and algorithms for min-max-min robust combinatorial optimization*, (2021).

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

[20] J. C. Smith and Y. Song, *A survey of network interdiction models and algorithms*, European Journal of Operational Research, 283 (2020), pp. 797–811.