IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Initial submission date April 22, 2021, revised submission June 23, 2021, accepted July 8, 2021, date of publication July 14, 2021, date of current version July 22, 2021.

Digital Object Identifier 10.1109/ACCESS.2021.3097039

Attack Strategies on Networks With a Budget Constraint

MEHDI MRAD[1], UMAR S. SURYAHATMAJA[1], ASMA BEN YAGHLANE[2], AND M. NACEUR AZAIEZ[2]

[1]Department of Industrial Engineering, College of Engineering, King Saud University, Riyadh 11421, Saudi Arabia
[2]Business Analytics and DEcision Making Laboratory (BADEM), Tunis Business School, Université de Tunis, Tunis 2059, Tunisia

Corresponding authors: Mehdi Mrad (mmrad@ksu.edu.sa) and Umar S. Suryahatmaja (usuryahatmaja@ksu.edu.sa)

This work was supported by the Deanship of Scientific Research at King Saud University through the Research Group Project under Grant RG-1438-079.

ABSTRACT In this paper, we develop solutions to the problem of identifying the *k most vital cut-sets* of a network under limited attacking resources in a network interdiction context. We treat three different versions of the problem, each having a different objective: namely, the highest probability of a successful attack, the least cost attack, and the least expected cost attack combining both success probability and attacking cost. We introduce a budget constraint to reflect the resource limitation of the attacker. We consider both the case of disjoint and non-disjoint cut-sets to target. We develop a simulation-optimization approach accounting for various techniques including the shortest path problem and the min-cut problem. The solution method is found to be very efficient even for large-scale networks. We implement the suggested algorithms for illustration purposes.

INDEX TERMS Attack, budget constraint, cut-set, interdiction networks.

I. INTRODUCTION

Many real-world systems can be modeled as networks. A large class of studies on networks consider finding optimal routes between two or more points. Another class deals with protecting network infrastructure and operations from the impacts of potential threats. These threats could be unintentional (e.g., natural disasters) or intentional (e.g., war, vandalism, and terrorism). Various terminologies are used almost interchangeably in addressing problems of intentional attacks on networks, including, interdiction networks, adversarial networks, and network survivability. The most common terminology however is interdiction networks, which basically deals with the operational side of a network. In this context, at least an attacker attempts to disrupt the functionality of the network while a defender tries to preserve its operation (see for instance Gharbi *et al.* [1]).

The world has suffered from many attacks on networks that resulted in huge damages and/or casualties. For illustration purposes, we set few examples. Saudi Arabia has frequently witnessed attacks on its oil distribution network. The South West London attack on its underground transportation network of September 2017 through an improvised explosive device resulted in 29 injuries. The French Institute of International Relations reported that the power sector has become a prime target for cyber-criminals in the last decade surging by 380% between 2014 and 2015. The US Department of Energy reported 150 successful attacks between 2010 and 2014 that targeted systems holding information regarding electricity grids. The February 2020 Distributed denial of service (DDoS) attacks on Amazon Web Services (AWS) is reported to be the largest in history blocking almost half the BT traffic on its entire UK network during a normal working day.

This article extends interdiction network models by Mrad *et al.* [2] and Yaghlane *et al.* [3], [4] by incorporating a budget constraint. In fact, the above studies identify optimal attack strategies without accounting for resource restrictions. This is certainly unrealistic and represents some limitation of the introduced models. The contribution of this article is mainly to overcome this drawback. In particular, the article illustrates the significant change on the attack strategies because of reduced resources. Moreover, it identifies multiple cut-sets to target instead of a single cut as considered in Yaghlane *et al.* [3], [4]. The problem is however highly challenging from a technical viewpoint. It deals with attacks

The associate editor coordinating the review of this manuscript and approving it for publication was Cristian Zambelli.

that disable network links in a way to partition the network and prevent the operations of receiving or sending flow. We assume that the attacker has perfect information related to the outcome of the attack on a single link. This information is instantly obtained so that the attacker may select the next move. That is, the attacker strategy is dynamic and accounts for the results of previous trials when identifying the next link to target.

The paper is organized as follows. Section II reviews the literature. Section III explains and formulates the problem at hand. Section IV proposes solution methodologies with illustrations. Section V explores the large scale of networks through an optimization-simulation approach. Finally, section VI provides conclusions and discussions of future work.

II. LITERATURE REVIEW

Interdiction network represents an area of intensive investigation in the last two decades, particularly with the existence of intelligent threats in all corners of the world (wars, terrorism, vandalism, sabotage, etc.). The problem mainly deals with attacks on networks that may disrupt network operations. Studies concern various types of networks, including telecommunication networks (e.g., Cox [5], Bellmore and Ratliff [6] and Frank *et al.* [7]), transportation networks (e.g., Alderson *et al.* [8], Kanturska *et al.* [9], and Bier and Hausken [10]), Electrical networks (e.g., Fang *et al.* [11]), computer networks (e.g., Obert *et al.* [12], and Hu *et al.* [13]), supply chains (e.g., Zhang *et al.* [14]), and counter-drug interdiction (e.g., Magliocca *et al.* [15] and Zhang *et al.* [16]), to name a few.

An interdiction problem may naturally be considered as a game (e.g., Kanturska *et al.* [9], Yaghlane *et al.* [4], Reilly *et al.* [17], and Bricha and Nourelfath [18]), where the leader attempts to disrupt the network operation while the follower seeks to make necessary moves/adjustments to preserve the network functionality even under some degraded mode. Common actions by the follower to respond to some nodes/links failure include rerouting. At a strategic level, followers may consider increasing capacities of nodes/links to allow for such rerouting under some limited protection resources, hiding operated paths and using deception plans to mislead the adversary. Attackers try to determine the best links/nodes to target to disturb/disable the network. The problem of each antagonist is often modeled as an optimization problem (both in the deterministic and stochastic frameworks). Yaghlane and Azaiez [19] investigate Nash equilibrium in various settings of the network survivability problem considering both zero and nonzero sum games. Xiao *et al.* [20] consider cumulative prospect theory (CPT) and derive the Nash equilibria in order to find out the attacker and defender's interaction when each of their actions is made subjectively. Bricha and Nourelfath [18] suggest a game-theoretical model of defense/attack strategies in networks in the context of incapacitated fixed-charge location problem. The model considers a non-cooperative two-period game. Casorrán *et al.* [21] consider multi attackers and one defender in general for security Stackelberg games. Li *et al.* [22] studies Stackelberg games with two attackers and one defender that fight over the hub nodes by considering cost-sensitive parameters.

Various optimization techniques are used in modeling network interdiction problems. Among these, one may cite the shortest path problem (e.g., Israeli and Wood [23], Wei *et al.* [24], Yaghlane *et al.* [4], etc.), the stochastic shortest path problem (e.g., Zhang *et al.* [16]), the min-cut problem (e.g., Yaghlane *et al.* [4]), Benders decomposition (e.g., Wei *et al.* [24]), bilevel knapsack (e.g., Caprara *et al.* [25]), to name a few. Smith and Song [26] provide a recent review on interdiction network mathematical formulation and solution approaches.

Yaghlane *et al.* [4] adapt system survivability as developed in Ben Yaghlane and Azaiez [19] to networks and distinguish the cases of perfect and absence of information. Under perfect information, the attacker knows with certainty the paths to be operated and plans attack strategies accordingly. In the case of absence of information, the attacker attempts to prevent the destinations from receiving flow by attempting to fully disabling a well-selected cut-set of the network. In a similar setting, Gharbi *et al.* [1] consider the least-cost attack problem to totally disable a network. They identify the optimal cut-set to target. Optimality is taken with respect to minimizing the expected attack cost. They approach the problem through a chance-constrained integer program where they consider a confidence level for a successful attack. They opt for a branch-and-bound spirit using various operations research tools to generate bounds. Yaghlane *et al.* [3] identify an exact method for the same problem treated by Gharbi *et al.* [1].

One of the important classes of interdiction network deals with the k-Most Vital Arcs Problem. The focus is to identify the most critical nodes/arcs susceptible of making the maximum disturbance on the network performance once disabled. In this context, Walteros *et al.* [27] discuss a framework to detect the set of critical nodes that maximize the possibility of disconnecting the network. Karakose and McGarvey [28] propose a path-based formulation and multi-commodity flow-based formulations to identify the optimal K-node to be attacked on a directed flow network so that to maximize the network disruption.

In a similar spirit, the problem of identifying the most critical network hubs is also investigated. Yahyaei *et al.* [29] design a single allocation hub network reliable to massive disruption using a bi-objective quadratic model. Lei [30] determines the critical air transportation hub facilities to be protected to reduce the transit time using an integer linear programming formulation. Ramamoorthy *et al.* [31] identify n critical hubs from a set of candidate hubs to reduce the routing cost using Bender decomposition. Quadros *et al.* [32] propose a branch and cut technique to fortify n hubs recognizing that m hubs will be attacked in a way to avoid an increase in the total distribution

cost. Mrad *et al.* [2] investigate the most critical cut-sets (instead of hubs) of a network that an attacker may determine to disable the network. They develop algorithms both in the disjoint and non-disjoint cases of cut-sets to identify optimal attack strategies. They investigate the large scale of networks through a simulation-optimization approach. Their contribution naturally extends the min-cut problem of Yaghlane *et al.* [4] to the most *critical k cut-sets* to target. While the simulation-optimization approach used in Mrad *et al.* [2] as well as the current paper relies on the expected performance of attacks in a risk-neutral framework, some other studies incorporate the risk aversion attitudes of the antagonists particularly to avoid disastrous consequences of attacks. For instance, Song and Shen [33] use chance-constrained programming to formulate their risk-averse shortest interdiction path model. Atamtürk *et al.* [34] suggest a convex quadratic mixed-integer program to model the risk-aversion problem of the leader using a mean-risk approach.

Interdiction network problems under limited attack-defense resource considerations has not been largely investigated. Powell [35] and Zenklusen [36] are among the first papers that explicitly account for resource availability. Powell [35] suggests a framework for analyzing defender scarce resource allocations in front of a strategic antagonist such as a terrorist. The paper considers Nash equilibria for various settings of the problem. Zenklusen [36] investigates a flow-max network problem with attacks on arcs and nodes. The concern was to remove enough links and nodes with the minimum attack budget in a way to make demand dissatisfaction of flow at the destination node. The review paper by Smith and Song [26] discusses additional resource-constrained interdiction problem concepts and approaches. Ravishankar *et al.* [37] suggest a game theoretic model to approach an attacker-defender game model by identifying the appropriate attack/defense strategies in a cyber security framework under limited resource availability. They introduce reinforcement learning technique to compute the expected payoff using linguistic fuzzy variables.

III. THE PROBLEM

In this section, we expose the network interdiction problem of interest to the current study. Actually, we extend previous contributions of Yaghlane *et al.* [3], [4] and Mrad *et al.* [2]. We will suggest a solution method that adapts to each of the three problems under consideration, as adequate, accounting for attacker resource availability as well as multiple cut-sets to target. The details are specified below.

A. THE PROBLEM SITUATION

We consider a network subject to intentional attacks. We assume one attacker trying to disable the network operation and one defender aiming to protect it. The attacks on the selected links/nodes of the network are considered as successful if they fully prevent the network of sending any flow from the source to the destination. This occurs if a full cut-set is disabled. As proven in Yaghlane *et al.* [4], the same problem can easily extend to account for multiple sources and destinations. In addition, the authors show that attacks on nodes can equivalently be reformulated to be considered as attacks on links. Therefore, without loss of generality, we will assume for this paper that the network accounts for a single source and a single destination. Further, attacks occur only on the links of the network.

Yaghlane *et al.* [4] determine the optimal cut-set to attack in a way to maximize the probability of disabling the network. In a similar context, Yaghlane *et al.* [3] solve the least expected cost of an attack on a full cut-set using an exact approach. A confidence level of a minimum guaranteed probability of success is modeled as a chance constraint in the problem formulation.

In order to determine a single cut-set to attack, one can use a linear programming (LP) formulation suggested by Yaghlane *et al.* [4] to generate the min-cut-set as follows:

$$\min \sum_{(i,j)\in E} -\ln(1-p_{ij})y_{ij} \tag{1}$$

s.t.

$$\forall (i,j) \in E : x_j \leq x_i + y_{ij} \tag{2}$$

$$x_n \geq x_1 + 1 \tag{3}$$

$$\forall i \in V : x_i \in \{0,1\} \tag{4}$$

$$\forall (i,j) \in E : y_{ij} \in \{0,1\} \tag{5}$$

where, V is the set of nodes, E is the set of arcs between nodes i and j, and p_{ij} is the survival probability of each $arc(i,j)$. Node 1 is the source and node n is the sink or the destination of the network. A set of arcs is a cut-set if it fragments the set of nodes V into two disjoint subsets called S_1 and S_2, where the source belongs to S_1 and the destination belongs to S_2. We let x_i be the binary variable that takes the value 1 if i belongs to S_2 and 0 otherwise. Also, we let y_{ij} be 1 if $arc(i,j)$ is selected for attack and 0 otherwise.

In general, attacking a single cut-set will provide a low probability of a successful attack given the necessity of successfully deactivating all corresponding arcs. Therefore, targeting several cut-sets seems to be more plausible to end up disabling the network. Somewhat in analogy with the identification of the k critical network hubs (discussed above), Mrad *et al.* [2] determine using a simulation-optimization approach the critical k *cut-sets* to target so that to maximize the probability of a successful attack. The main shortcoming of models by Mrad *et al.* [2] and Yaghlane *et al.* [4] resides in ignoring the resource availability of the attacker, which is often unrealistic. Moreover, the models by Yaghlane *et al.* [3], [4] content by determining a single cut-set to target (usually with very low probability of success). This is limiting, as both attackers and defenders are interested in identifying the most vital cut-sets to target/reinforce.

The current paper suggests considering three versions. Each version will modify one of the models by Yaghlane *et al.* [3], [4] and Mrad *et al.* [2] by incorporating a budget constraint to reflect the limited attacking resources that could be available to the attacker. Furthermore,

the suggested models extend Yaghlane *et al.* [3], [4] by identifying the most critical *k cut-sets* of the network. The treatment will account for both situations of generating disjoint and non-disjoint cut-sets.

B. THE PROBLEM STATEMENT

The attacker will identify up to *k cut-sets* to target. Attacks are made sequentially over the various links of a given selected cut-set. A cut-set is disabled if all corresponding links fail. The attacker has full knowledge of the outcome of an attack on an attempted link before deciding on the next trial to carry out. Each link having a known probability to survive an attack may be tried at most once. The model treats each of the two cases of attacks on disjoint versus non-disjoint cut-sets. The attack will stop if one of the following situations occurs:

1) A full cut-set is disconnected, in which case the network is disabled, and the attack succeeds.
2) The budget is not sufficient to carry out additional attacks, in which case the attack fails.
3) The network is guaranteed to send flow to the destination (i.e., a path set is identified to safely operate independently of the outcome of any potential future attacks), in which case the attack fails.
4) All selected *k cut-sets* are unsuccessfully tried, in which case the attack fails.

The attacker seeks to identify the best *k cut-sets* to target in the appropriate order so that to satisfy one of the three objectives below.

(a) The success probability of an attack is maximized
(b) The overall attack cost is minimized
(c) The expected cost of the overall attack (combining survival probabilities and attack cost) is minimized

Objective (a) has been treated in Mrad *et al.* [2] without accounting for a budget constraint. Objective (b) has been treated in Yaghlane *et al.* [4] for the special case of $k = 1$ and without accounting for a budget constraint. Algorithm 1 is used to solve the min-cut problem while minimizing the cost of the attack under a budget constraint. Finally, Objective (c) has been treated in Yaghlane *et al.* [3] and Algorithm 2 is proposed to solve the least expected cost cut problem.

IV. SOLUTION METHODOLOGY

Algorithms 3 and 4 solve the disjoint and the non-disjoint cases, respectively. Three subroutines are used to take care of each of the three objectives stated above. The next three subsections will elaborate on each version with additional details on the technical treatment.

A. MIN-CUT WITH SUCCESS PROBABILITY (MCSP)

The first version attempts to identify an optimal attack strategy that identifies the most critical *k cut-sets* so that to maximize the success probability of the attacker. Cost coefficients related to each link of the network are introduced in this formulation based on the survival probability of the specific link. To generate a cut-set using this method, a budget constraint (9) is augmented to the displayed mathematical model on sub-section III-A. This constraint will limit the selected cut-set based on the available resources.

$$\min \sum_{(i,j)\in E} -\ln(1-p_{ij})y_{ij} \tag{6}$$

s.t.

$$\forall (i,j) \in E : x_j \leq x_i + y_{ij} \tag{7}$$

$$x_n \geq x_1 + 1 \tag{8}$$

$$\sum_{(i,j)\in E} cost_{ij} y_{ij} \leq budget \tag{9}$$

$$\forall i \in V : x_i \in \{0,1\} \tag{10}$$

$$\forall (i,j) \in E : y_{ij} \in \{0,1\} \tag{11}$$

In this formulation, $cost_{ij}$ is the required cost to attack $arc(i,j)$ and *budget* is the total available resources of the attacker. A subroutine based on this linear programming formulation will be repeatedly called in Algorithms 3 and 4 when solving *MCSP*.

Links of a selected cut-set under this method are supposed to be sorted according to the descending rule of their survival probabilities as explained in Mrad *et al.* [2]. In fact, for disabling a cut-set, all its (vulnerable and robust) links must fail. Therefore, when starting by vulnerable ones, this may unnecessarily consume the budget before failing the attack when trying robust ones. However, given the weight of the cost of a given link introduced in this study, the sorting order must combine both survival probability and cost in accordance with equation (18) below as explained in Yaghlane *et al.* [3].

B. MIN-CUT WITH COST (MCC)

Instead of selecting a cut-set based on the lowest survival probability, one may consider minimizing the total cost of attacking a cut-set by modifying the objective function of the mathematical formulation of sub-section III-A by equation (12) as follows:

$$ILP = \min \sum_{(i,j)\in E} cost_{ij} y_{ij} \tag{12}$$

s.t.

$$\forall (i,j) \in E : x_j \leq x_i + y_{ij} \tag{13}$$

$$x_n \geq x_1 + 1 \tag{14}$$

$$\forall i \in V : x_i \in \{0,1\} \tag{15}$$

$$\forall (i,j) \in E : y_{ij} \in \{0,1\} \tag{16}$$

A subroutine based on this linear programming formulation (Algorithm 1) will be repeatedly called in Algorithms 3 and 4 when solving the *MCC*. If the optimal solution of the above LP turns out to exceed the remaining budget at any call of Algorithms 3 and 4, then the algorithm stops returning the cut-sets generated by that step. Clearly, when the budget is not enough to warrant the first step, the attack problem is infeasible. Algorithm 1 will be used as the subroutine for the *MCC* version of the problem.

Algorithm 1 Budget Constraint for Min-Cut With Cost

currentBudget = remaining budget of the attack
set $A = \emptyset$ (cut-set with the minimum cost)
if ILP (12) - (16) $\leq$ *currentBudget* **then**
  set A = the solution of the Integer Program (12) - (16)
end if

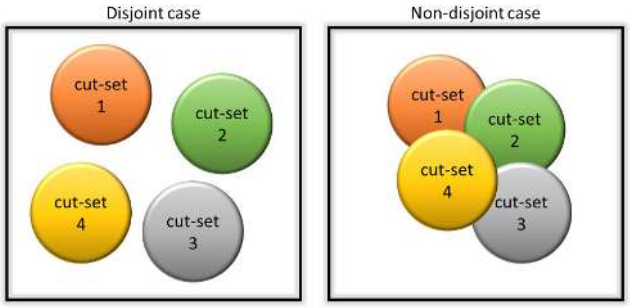


FIGURE 1. Disjoint vs non-disjoint case.

C. MIN-CUT WITH THE LEAST EXPECTED COST (MCLEC)

The third method to generate budget-constrained k *cut-sets* deals with the least expected cost as treated in Yaghlane *et al.* [3]. The authors propose an exact solution under a budget constraint accounting for a threshold of success probability of the attack on one targeted cut-set. The objective function is nonlinear, and the authors develop a constraint generation algorithm to determine the cut with the least expected cost as described in Algorithm 2. This algorithm will generate different cut-sets sequentially by solving the ILP provided in the subsection IV-A augmented with the following constraint:

$$\sum_{(i,j)\in S} y_{ij} \leq |S| - 1 \tag{17}$$

where S is the generated cut-set and $|S|$ is the cardinality of S. The new constraint will ensure that the generated new cut-set is different from the previous cut-set.

In each iteration, Algorithm 2 is also considering the budget availability when solving the ILPs. The cut-set selection from the cut-sets pool is based on the minimum expected cost calculated by using the following equation.

$$E(S) = cost_{i_1 j_1} + \sum_{k=2}^{|S|} \Big(\prod_{l=1}^{k-1} 1 - p_{i_l j_l}\Big) cost_{i_k j_k} \tag{18}$$

where S is sorted using the non-descending rule of $cost_{ij}/P_{ij}$. The corresponding explanations and justifications are provided in Yaghlane *et al.* [3]. Algorithm 2 will be used in the current work as a subroutine for the *MCLEC* version.

We develop now a compound algorithm (Algorithm 3) to treat all three versions of the problem for the disjoint case and a second one (Algorithm 4) for the general non-disjoint case. In each situation, we let the algorithm repeatedly call the adequate subroutine to generate the optimal solution corresponding to the considered version. The disjoint case

Algorithm 2 Constraint Generation for Min-Cut With Expected Least Cost

budget = budget of the attack
$k = 0$
set $A = \emptyset$ (cut-set with the expected least cost)
stop = false
attackCost = 0
minExpectedCost = budget
currExpectedCost = budget
while stop = false **do**
  attackCost = **solve** the min cut problem that maximizes the success probability of the attack as provided in subsection IV-A
  if *attackCost* > *budget* **or** no feasible solution exists **then**
    if $k = 0$ **then**
      The problem is infeasible
    end if
    stop = true
  else
    calculate the *currExpectedCost* using equation (18)
    if *currExpectedCost* < *minExpectedCost* **then**
      minExpectedCost = *currExpectedCost*
      set A = the current cut-set
    end if
    Add constraint (17)
    $k = k + 1$
  end if
end while

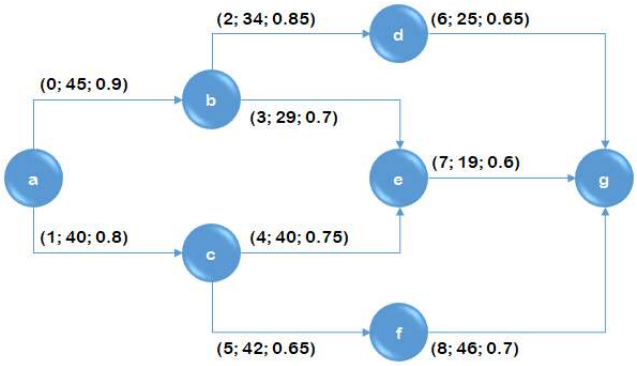
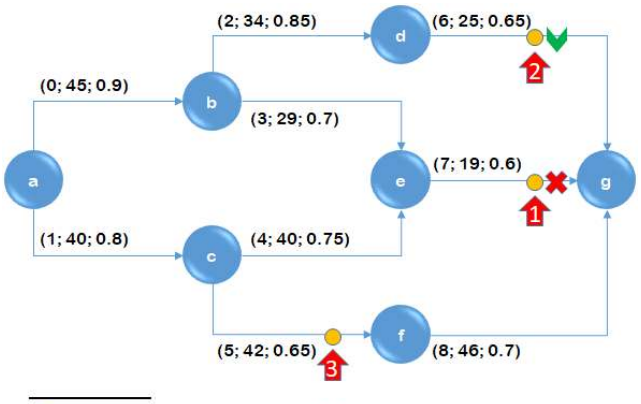
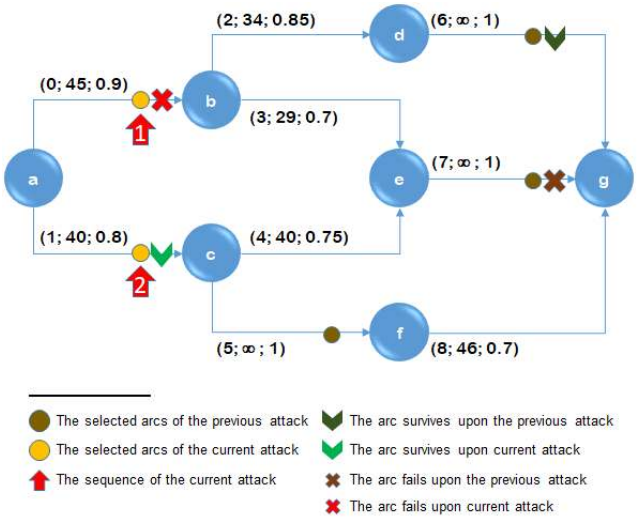


FIGURE 2. Initial network.

of cut-sets refers to the situation of cut-sets to be targeted with empty intersections; while the non-disjoint case is the general case where the same link may belong to more than one targeted cut-set as illustrated in Figure 1. That is, a failed link of a surviving cut-set may be considered in another cut-set and hence may reduce the trials on the new cut-set and at the same time increase the probability of its failure. In either case, the budget constraint plays the role of a new stopping criterion. In fact, the algorithm keeps track of the remaining budget at each iteration, so that when the residual budget is not enough to conduct an attack on any full cut-set, the algorithm stops and the overall attack ceases. It should also be clear that the remaining budget at any iteration plays



(a) Iteration 1, targeting cut-set $\{5, 6, 7\}$.



(b) Iteration 2, targeting cut-set $\{0, 1\}$. Overall attack failed.

FIGURE 3. The attacking process by using *MCSP* method with disjoint cut-sets.

an important role in selecting the next cut-set to target, which need not be the most attractive one as for the case of unlimited resource situation. Furthermore, the budget limitation may dictate at the final solution a smaller number of cut-sets to identify rather than all k *critical cut-sets*.

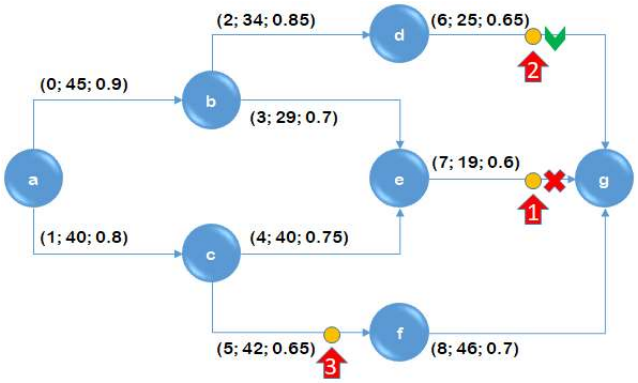
Algorithm 3 below deals with the disjoint cut-set case. In this situation, whenever a cut-set is unsuccessfully tried, all corresponding links will never be further considered for future cut-sets to be identified, even if these links have not been attacked. Nevertheless, the failing links of previously attempted cut-sets are kept in memory to help identify when the network is fully disabled. The algorithm generates the best cut-set to target based on the appropriate objective undertaken (among the three suggested methods). This cut-set must comply with the budget availability restriction. Simulated attacks are sequentially carried out on the links of this cut-set according to the specified order of the non-descending rule of $cost_a/P_a$ of its links. If the cut-set is disabled, the attack

Algorithm 3 Disjoint Cut-Set

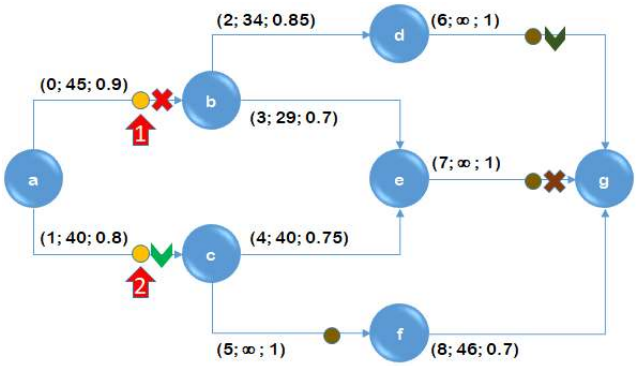
Let $G(V, E)$ be the graph where V is the set of nodes and E is the set of links
$B =$ budget of the attack
$C_a =$ cost to break an arc a
$R =$ available resources
$K =$ the maximum number of the allowed attacks
Stop = false; (the algorithm will stop if the attack succeeds or it is impossible to find more cuts to attack)
succeed = false; (the algorithm succeeds if the arcs of one cut are broken)
$k = 1$; (a counter of the cuts to attack)
set $W = \emptyset$ (set of disabled links)
set $A = \emptyset$ (current targeted cut-set)
set $\Omega = \emptyset$ (set of cuts to attack)
$R = B$
while stop = false **do**
  set A = generate the new optimal cut from the graph $G(V, E)$ using one of the methods explained in sub-sections IV-A, IV-B, or IV-C (as appropriate) where $B = R$
  set $\Omega = \Omega \cup A$
  if $A = \emptyset$ **then**
    stop = true;
  else
    sort A using the non-descending rule of C_a/P_a of its links
    for all link a in A based on the above ordering **do**
      attack the link (Generate random $p \in [0, 1]$)
      $R = R - C_a$;
      if attack fails ($p < P_a$) **then**
        break;
      else
        $W = W \cup \{a\}$
        if link a is the last link in A **then**
          stop = true; succeed = true;
        else
          if $W \neq \emptyset$ **and** W contains a full cut-set **then**
            stop = true; succeed = true;
            break;
          end if
        end if
      end if
    end for
  end if
  if stop = false **then**
    for all link a in A **do**
      $P_a = 1.0$;
      $C_a = +\infty$;
    end for
    if $k < K$ **then**
      $k = k + 1$;
    else
      stop = true;
    end if
  end if
  clear A;
end while

Algorithm 4 Non-Disjoint Cut-Set

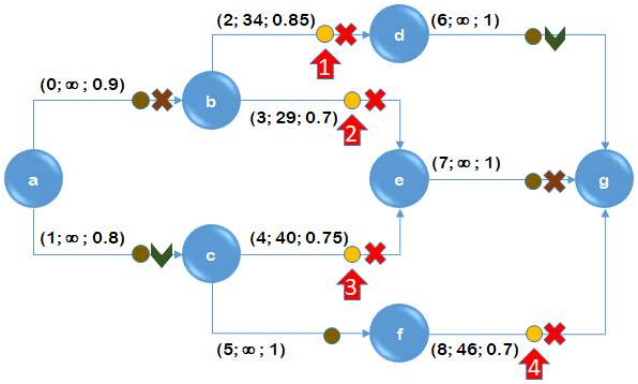
Let $G(V, E)$ be the graph where V is the set of nodes and E is the set of links
B = budget of the attack
C_a = cost to break an arc a
R = available resources
K = the maximum number of the allowed attacks
Stop = false; (the algorithm will stop if the attack succeeds or it is impossible to find more cuts to attack)
succeed = false; (the algorithm succeeds if the arcs of one cut are broken)
$k = 1$; (a counter of the cuts to attack)
set $W = \emptyset$ (set of disabled links)
set $A = \emptyset$ (current targeted cut-set)
set $\Omega = \emptyset$ (set of cuts to attack)
$R = B$
while stop = false **do**
  set A = generate the new optimal cut from the graph $G(V, E)$ using one of the methods explained in sub-sections IV-A, IV-B, or IV-C (as appropriate) where $B = R$
  set $\Omega = \Omega \cup A$
  if $A = \emptyset$ **then**
    stop = true;
  else
    sort A using the non-descending rule of C_a/P_a of its links
    for all link a in A based on the above ordering **do**
      attack the link (Generate random $p \in [0, 1]$)
      $R = R - C_a$;
      if attack fails ($p < P_a$) **then**
        $P_a = 1.0$; $C_a = +\infty$;
        break;
      else
        $W = W \cup \{a\}$
        $P_a = 0.0$; $C_a = 0$;
        if link a is the last link in A **then**
          stop = true; succeed = true;
        else
          if $W \neq \emptyset$ **and** W contains a full cut-set **then**
            stop = true; succeed = true;
            break;
          end if
        end if
      end if
    end for
  end if
  if stop = false **then**
    if $k < K$ **then**
      $k = k + 1$;
    else
      stop = true;
    end if
  end if
  clear A;
end while



(a) Iteration 1, targeting cut-set $\{5, 6, 7\}$.



(b) Iteration 2, targeting cut-set $\{0, 1\}$.

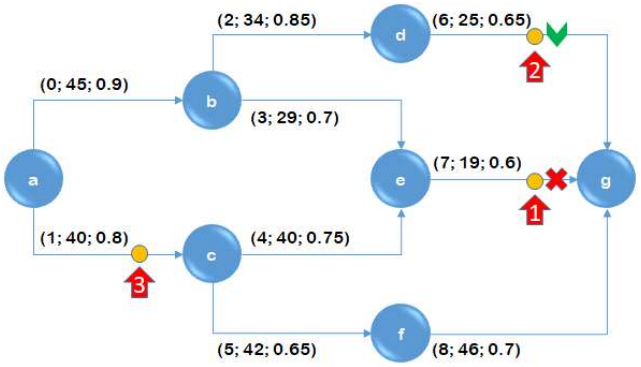


(c) Iteration 3, targeting cut-set $\{2, 3, 4, 8\}$. Overall attack succeed.

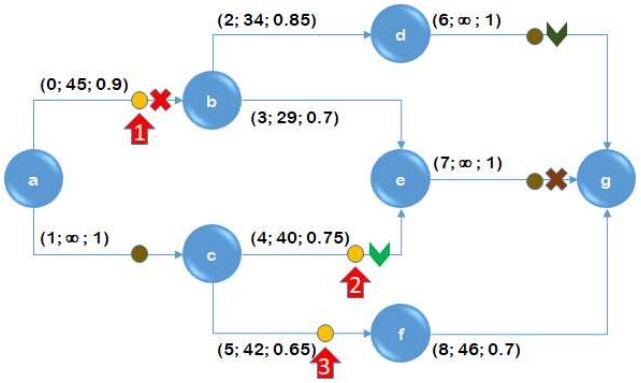
FIGURE 4. The attacking process by using *MCSP* method with disjoint cut-sets and 500 as the total budget.

stops, and the network is disconnected. Otherwise, all links of the identified cut-set are removed from further consideration while keeping in memory failed ones. The process starts over till one of the stopping criteria is met. It is worthwhile mentioning that the dynamic nature of the attack makes full use of the perfect information on the outcomes of previous attacks in selecting the next cut-set to target.

Algorithm 4 below will generate non-disjoint cut-sets until one is fully disabled, all k *cut-sets* are identified, the budget is depleted, or a path set is identified to survive regardless the outcomes of future attacks, whichever occurs first.

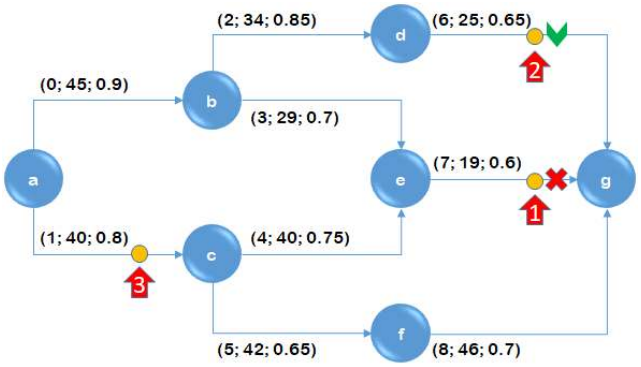


(a) Iteration 1, targeting cut-set $\{1, 6, 7\}$.

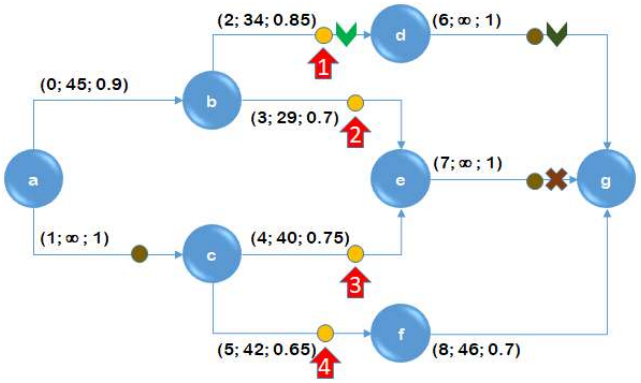


(b) Iteration 2, targeting cut-set $\{0, 4, 5\}$. Overall attack failed.

FIGURE 5. The attacking process by using *MCC* method with disjoint cut-sets.



(a) Iteration 1, targeting cut-set $\{1, 6, 7\}$.



(b) Iteration 2, targeting cut-set $\{2, 3, 4, 5\}$. Overall attack failed.

FIGURE 6. The attacking process by using *MCLEC* method with disjoint cut-sets.

The algorithm keeps track of the failed links that could be used to form a new cut-set to try and/or to identify a fully disconnected cut-set.

We provide in the two subsections below illustrative examples for all methods in the situation of disjoint and non-disjoint cut-sets. The network depicted in Figure 2 is used as an initial network for each example. It has seven nodes and nine arcs where each $arc(i, j)$ has triple (id_{ij}, $cost_{ij}$, p_{ij}). A cut-set will be generated and attacked sequentially by iterating the procedure as explained in Algorithms 3 and 4. The total budget for the attack is 200 and the allowed k is 3. As soon as a cut-set is fully disabled, the network fails to supply the destination node with flow.

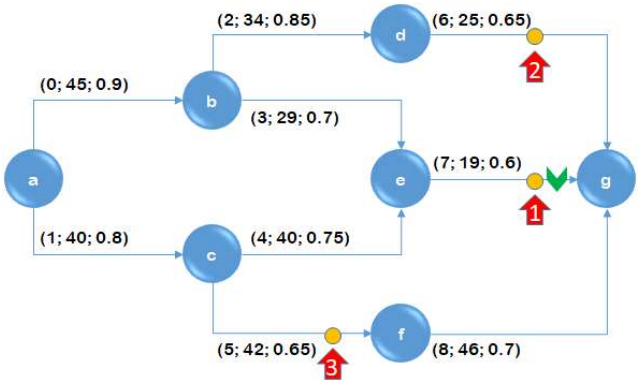
D. CASE OF DISJOINT CUT-SETS

We will start with the *MCSP* version of the problem. The first iteration is shown in Figure 3a. The yellow dot shows the identified arcs on a cut-set {5, 6, 7} and the numbered arrow shows the sequence of attacks so that the attack starts with the lowest ratio of cost/survival probability. The cross sign on arc {7} means that the attack is successful while the green mark on arc {6} means a failed attack (i.e., a surviving arc).

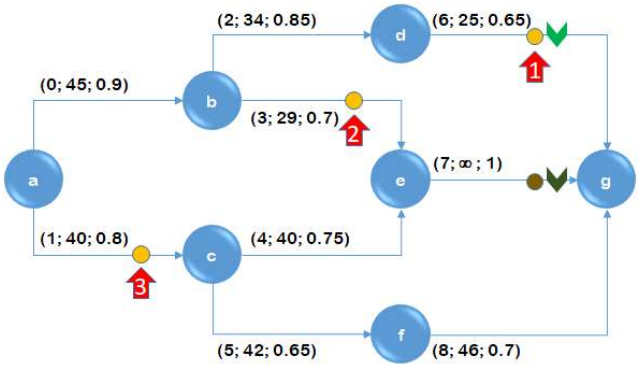
Consequently, the attack on the current cut-set stops. The remaining budget after the current attack is 156. Therefore, a new cut-set is to be identified using *MCSP* after artificially removing the tried one by changing the cost to infinity and the survival probability to 1.

The new identified cut-set by the next iteration is {0, 1} (Figure 3b). The attempt is successful on arc {0} but not on arc {1}. Here, the new cut-set; namely, {2, 3, 4, 8} with total cost of 149 is identified. However, the remaining budget of only 71 is not sufficient to cover the attack on this new cut-set. Therefore, the overall attack stops and fails. For comparison purposes, in Figure 4, we show the attacking process if the attacker's budget were 500 instead of 200. In iteration 2 (Figure 4b), the attack fails while the remaining budget is 371. Such a budget is enough to cover the 149 cost of targeting cut-set {2, 3, 4, 8}. The attempt on this cut set turns out to be successful (Figure 4c) so that the flow cannot reach destination g from source node a. The overall attack succeeds in this case while it initially failed with the reduced budget.

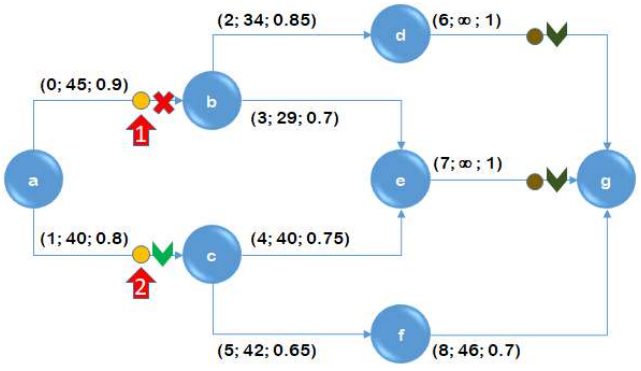
Now, we treat the version of the problem related to *MCC*. Considering the lowest cost combination, the first identified cut-set is {1, 6, 7}. The simulated attacks yield a success on arc 7 followed by a failure on arc {6} (Figure 5a). The new cut-set is identified as {0, 4, 5}. Figure 5b shows the result of the attack on the new cut-set in which the attack on arc {4} is unsuccessful. A new cut-set needs to be identified but the remaining available arcs will not form any cut-set. Thus, the overall attack fails (Figure 5b).



(a) Iteration 1, targeting cut-set $\{5, 6, 7\}$.

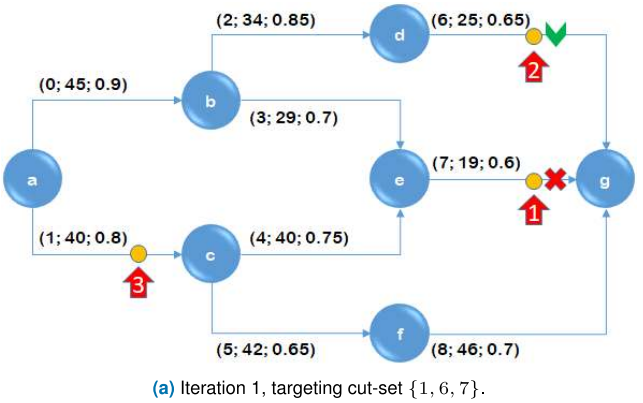


(b) Iteration 2, targeting cut-set $\{1, 3, 6\}$.

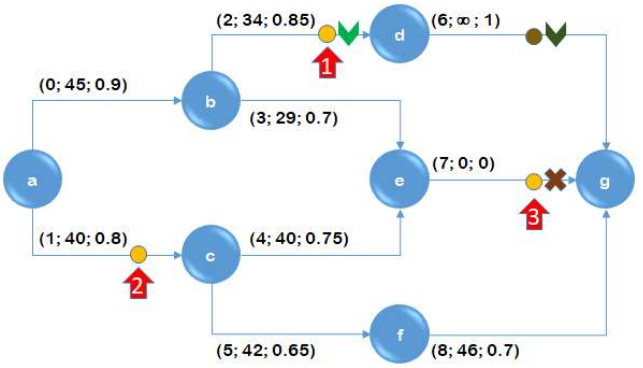


(c) Iteration 3, targeting cut-set $\{0, 1\}$. Overall attack failed.

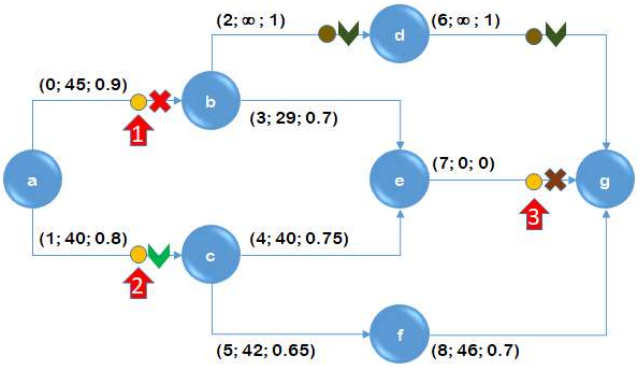
FIGURE 7. The attacking process by using *MCSP* method with non-disjoint cut-sets.



(a) Iteration 1, targeting cut-set $\{1, 6, 7\}$.



(b) Iteration 2, targeting cut-set $\{1, 2, 7\}$.



(c) Iteration 3, targeting cut-set $\{0, 1, 7\}$. Overall attack failed.

FIGURE 8. The attacking process by using *MCC* method with non-disjoint cut-sets.

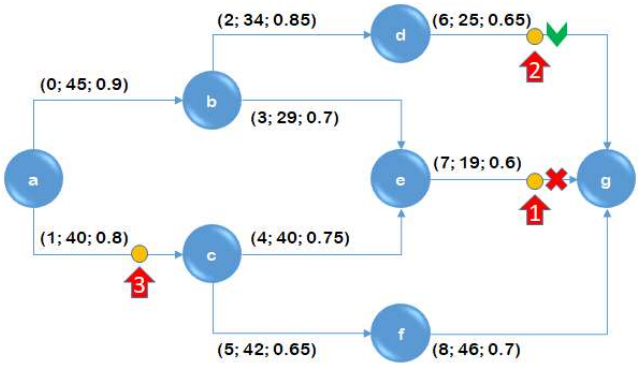
Next, we consider the *MCLEC* version (exhibited on Figure 6a). As in the *MCC* example (Figure 5a), cut-set {1, 6, 7} is identified. The simulated attack disables arc {7} but not arc {6}. The new cut-set {2, 3, 4, 5} is then identified (Figure 6b). Observe that this cut-set is different from the one determined in the *MCC* version of the problem (Figure 5b). The attack on arc {2} is unsuccessful so that the overall attack fails because no new cut-set can be identified.

E. CASE OF NON-DISJOINT CUT-SETS

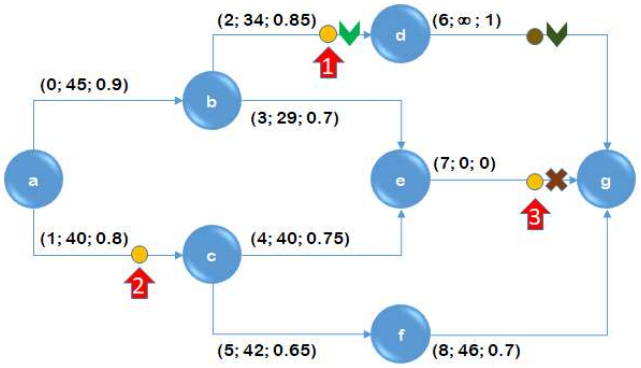
In this section, we treat the example above in all three versions of the problem using the corresponding proposed solution methods in the situation of non-disjoint cut-sets (adopting Algorithm 4 rather than Algorithm 3). The *MCSP* version

TABLE 1. *kOpt* by using min-cut algorithm Mrad *et al.* [2].

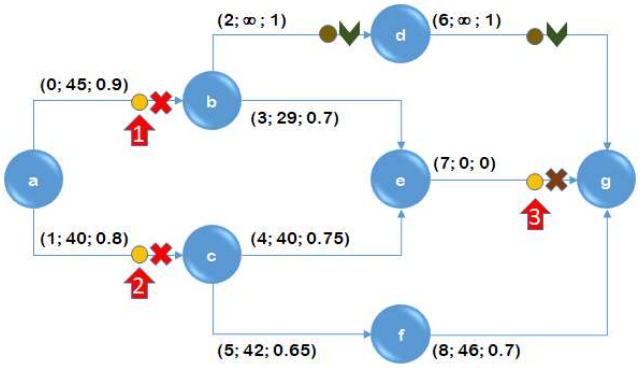
Instance Type	Probability Range	Disjoint			Non-Disjoint		
		k=3	*k=5*	*k=10*	*k=3*	*k=5*	*k=10*
small	ORIGINAL [1]	2	2	2	3	3	4
	[0.01-0.25]	2	2	2	2	2	2
	[0.01-0.50]	2	2	2	3	3	4
	[0.01-0.99]	2	2	2	3	4	5
large	ORIGINAL [3]	3	3	3	3	5	8
	[0.01-0.25]	2	2	2	3	3	4
	[0.01-0.50]	3	3	3	3	4	7
	[0.01-0.99]	3	3	3	3	5	9



(a) Iteration 1, targeting cut-set $\{1, 6, 7\}$.

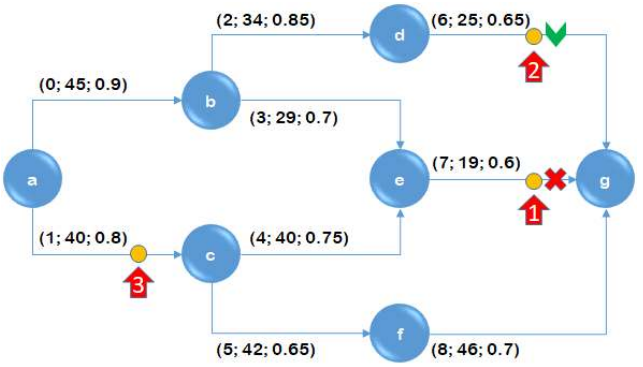


(b) Iteration 2, targeting cut-set $\{1, 2, 7\}$.

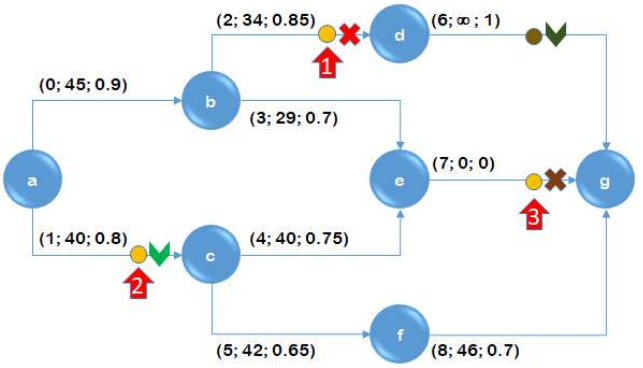


(c) Iteration 3, targeting cut-set $\{0, 1, 7\}$. Overall attack succeed.

FIGURE 9. The attacking process by using *MCLEC* method with non-disjoint cut-sets.

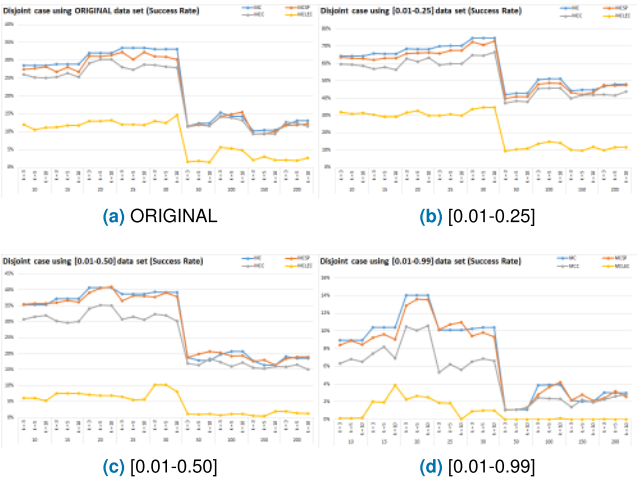


(a) Iteration 1, targeting cut-set $\{1, 6, 7\}$.



(b) Iteration 2, targeting cut-set $\{1, 2, 7\}$. Overall attack failed.

FIGURE 10. The attacking process by using *MCLEC* method with non-disjoint cut-sets and 150 of budget.



(a) ORIGINAL (b) [0.01-0.25]

(c) [0.01-0.50] (d) [0.01-0.99]

FIGURE 11. Success rate of the disjoint case for each survival probability range.

identifies cut-set {5, 6, 7} as shown in Figure 7a. The first attempt unsuccessfully targets arc {7} so that the attack fails on the current cut-set. The survival probability and cost of arc {7} are then artificially modified to values 1 and ∞, respectively to omit the arc from the next considerations. As opposed to the disjoint case, the survival probability and cost of arc {5, 6} will remain the same, so that it can be used in future cut-set identifications.

The second iteration (Figure 7b) generates cut-set {1, 3, 6}, where arc {6} is the intersection of the current cut-set with the previous one. The attempt on arc {6} fails and a new cut-set needs to be determined. With the remaining budget of 156, cut-set {0, 1} is the one identified in the third iteration (Figure 7c). This new cut-set intersects with the previous one in arc {1}. The attack on arc {0} succeeds while that on arc {1} fails. Two stopping criteria are simultaneously met: the remaining budget cannot cover the cost on any remaining cut-set and the limit *k* is reached. Therefore, the algorithm stops, and the overall attack fails.

The treatment of the *MCC* version of the problem for the non-disjoint case is shown in Figure 8. As in the example

TABLE 2. Average run time (in seconds) and standard deviation for each 100 scenarios.

Case	Instance Type	Probability Range	*MC**	*MCSP*	*MCC*	*MCLEC*
Disjoint	small	ORIGINAL [1]	(7.07; 0.29)	(7.75; 0.30)	(8.18; 0.31)	(304.32; 2.05)
		[0.01-0.25]	(5.67; 0.29)	(6.10; 0.32)	(6.88; 0.33)	(282.13; 3.05)
		[0.01-0.50]	(6.04; 0.25)	(6.32; 0.26)	(7.05; 0.24)	(312.54; 1.55)
		[0.01-0.99]	(5.75; 0.12)	(5.15; 0.11)	(5.27; 0.11)	(334.28; 0.89)
	large	ORIGINAL [3]	(18.21; 0.92)	(35.43; 1.18)	(19.77; 1.00)	(921.79; 7.08)
		[0.01-0.25]	(16.16; 1.24)	(28.96; 1.63)	(17.35; 1.35)	(889.12; 11.00)
		[0.01-0.50]	(14.30; 0.64)	(29.30; 0.83)	(15.39; 0.68)	(975.62; 4.76)
		[0.01-0.99]	(11.82; 0.16)	(28.68; 0.35)	(11.63; 0.27)	(965.90; 2.80)
Non-Disjoint	small	ORIGINAL [1]	(13.95; 0.43)	(33.49; 1.71)	(44.72; 1.95)	(696.69; 13.70)
		[0.01-0.25]	(8.81; 0.34)	(14.79; 0.99)	(23.26; 1.27)	(460.29; 8.11)
		[0.01-0.50]	(12.34; 0.39)	(26.22; 1.30)	(37.54; 1.64)	(841.55; 16.87)
		[0.01-0.99]	(11.12; 0.19)	(18.92; 0.86)	(27.08; 0.89)	(1,107.66; 13.63)
	large	ORIGINAL [3]	(46.29; 1.82)	(121.91; 7.52)	(89.74; 5.82)	(1,890.91; 62.57)
		[0.01-0.25]	(34.48; 2.19)	(86.33; 8.30)	(74.05; 7.68)	(1,520.05; 57.15)
		[0.01-0.50]	(33.43; 1.13)	(90.65; 4.51)	(67.58; 3.81)	(2,455.20; 67.29)
		[0.01-0.99]	(27.28; 0.36)	(67.40; 1.56)	(34.33; 0.99)	(3,134.09; 50.19)

*Min-cut in Mrad et al. [2]

TABLE 3. Success rate of the disjoint case.

Survival Probability Range	#Nodes	$k=3$				$k=5$				$k=10$			
		MC	*MCSP*	*MCC*	*MCLEC*	*MC*	*MCSP*	*MCC*	*MCLEC*	*MC*	*MCSP*	*MCC*	*MCLEC*
ORIGINAL	10	28.5%	27.4%	26.1%	12.0%	28.5%	27.7%	25.2%	10.6%	28.5%	28.2%	25.1%	11.2%
	15	28.9%	26.8%	25.3%	11.3%	28.9%	28.1%	26.4%	11.7%	28.9%	26.7%	25.3%	11.7%
	20	32.1%	31.1%	29.2%	13.0%	32.1%	31.1%	30.3%	13.0%	32.1%	31.4%	30.3%	13.2%
	25	33.5%	32.3%	28.0%	12.0%	33.5%	30.2%	27.4%	12.0%	33.5%	32.3%	28.8%	11.8%
	30	33.1%	31.1%	28.7%	12.9%	33.1%	31.0%	28.2%	12.4%	33.1%	30.2%	27.9%	14.6%
	50	11.5%	11.5%	11.4%	1.6%	12.4%	12.1%	11.9%	1.8%	12.4%	11.6%	11.6%	1.4%
	100	15.4%	14.3%	14.2%	5.7%	14.3%	14.9%	13.9%	5.3%	14.3%	15.4%	13.2%	4.8%
	150	10.2%	9.3%	9.4%	2.0%	10.5%	9.4%	9.5%	3.0%	10.5%	10.1%	9.3%	2.0%
	200	11.9%	11.7%	12.8%	2.1%	13.1%	11.9%	12.3%	1.9%	13.1%	12.2%	11.5%	2.6%
[0.01-0.25]	10	64.3%	63.5%	59.7%	31.8%	64.3%	63.1%	59.4%	30.9%	64.3%	62.9%	58.5%	31.3%
	15	65.7%	62.2%	56.8%	30.3%	65.4%	63.0%	57.7%	29.0%	65.4%	63.2%	56.3%	29.2%
	20	68.5%	65.9%	62.8%	31.6%	68.4%	65.9%	61.2%	32.7%	68.4%	66.3%	63.2%	30.0%
	25	70.0%	65.7%	59.0%	29.8%	70.2%	67.5%	59.9%	30.7%	70.2%	67.6%	59.9%	29.8%
	30	74.6%	72.4%	64.8%	33.5%	74.6%	70.8%	64.5%	34.7%	74.6%	72.8%	66.5%	34.6%
	50	42.1%	39.8%	37.0%	9.3%	42.7%	40.8%	38.2%	10.3%	42.7%	40.9%	37.7%	10.7%
	100	50.8%	48.0%	45.4%	13.5%	51.1%	48.8%	45.8%	14.7%	51.1%	48.4%	45.8%	14.2%
	150	44.2%	43.2%	39.8%	10.0%	44.8%	42.0%	41.8%	9.6%	44.8%	43.4%	41.8%	11.8%
	200	46.8%	47.5%	41.9%	9.8%	48.0%	47.3%	41.4%	11.5%	48.0%	47.5%	43.8%	11.7%
[0.01-0.50]	10	35.4%	35.5%	30.7%	6.0%	35.4%	35.7%	31.6%	6.1%	35.4%	35.8%	32.0%	5.2%
	15	37.3%	36.0%	30.3%	7.6%	37.3%	36.7%	29.7%	7.6%	37.3%	36.1%	30.1%	7.6%
	20	40.7%	39.1%	34.1%	7.2%	40.7%	40.6%	35.1%	6.9%	40.7%	41.0%	35.1%	6.8%
	25	38.7%	36.5%	30.7%	6.5%	38.7%	38.2%	31.5%	5.5%	38.7%	38.0%	30.7%	5.6%
	30	39.4%	37.8%	32.4%	10.3%	39.2%	39.2%	32.0%	10.2%	39.2%	37.9%	30.2%	8.0%
	50	18.8%	18.8%	16.9%	1.2%	17.9%	19.9%	16.5%	1.0%	17.9%	20.8%	18.4%	1.1%
	100	19.7%	20.3%	17.4%	0.7%	20.7%	19.3%	16.0%	1.2%	20.7%	19.4%	17.2%	1.1%
	150	17.8%	17.7%	15.6%	0.6%	16.5%	18.1%	15.3%	0.5%	16.5%	16.5%	16.0%	1.9%
	200	19.1%	18.4%	15.8%	1.9%	18.6%	18.9%	16.5%	1.4%	18.6%	18.9%	15.1%	1.2%
[0.01-0.99]	10	9.0%	8.4%	6.4%	0.2%	9.0%	8.9%	6.8%	0.1%	9.0%	8.5%	6.5%	0.2%
	15	10.4%	9.3%	7.4%	2.0%	10.4%	9.6%	8.2%	1.9%	10.4%	9.0%	6.9%	3.9%
	20	14.0%	12.9%	10.5%	2.3%	14.0%	13.6%	10.1%	2.6%	14.0%	13.5%	10.6%	2.5%
	25	10.1%	10.2%	5.3%	1.9%	10.1%	10.7%	6.3%	1.8%	10.1%	11.0%	5.6%	0.1%
	30	10.3%	9.5%	6.5%	0.9%	10.4%	9.8%	6.9%	1.0%	10.4%	9.3%	6.6%	1.0%
	50	1.1%	1.1%	1.1%	0.0%	1.1%	1.1%	1.1%	0.0%	1.1%	1.4%	1.3%	0.0%
	100	3.9%	2.8%	2.5%	0.0%	3.9%	3.7%	2.4%	0.0%	3.9%	4.2%	2.3%	0.2%
	150	2.2%	2.2%	1.4%	0.0%	2.0%	2.8%	2.2%	0.0%	2.0%	2.1%	1.9%	0.0%
	200	3.1%	2.4%	2.3%	0.1%	3.0%	3.2%	2.6%	0.0%	3.0%	2.6%	2.9%	0.0%

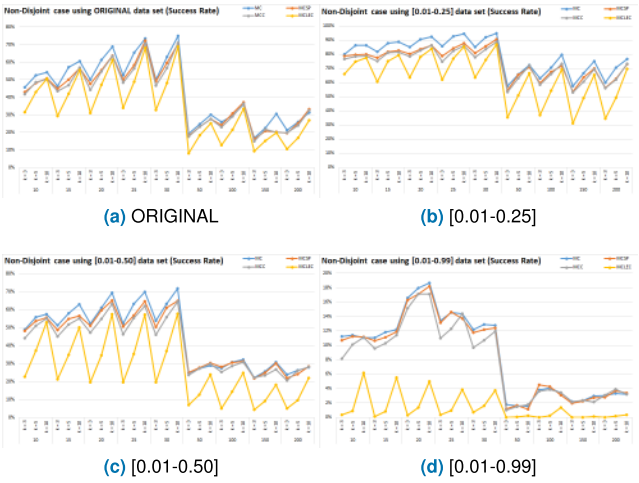
TABLE 4. Success rate of the non-disjoint case.

Survival Probability Range	#Nodes	$k=3$				$k=5$				$k=10$			
		MC	*MCSP*	*MCC*	*MCLEC*	*MC*	*MCSP*	*MCC*	*MCLEC*	*MC*	*MCSP*	*MCC*	*MCLEC*
ORIGINAL	10	45.7%	43.1%	41.8%	31.5%	52.4%	48.2%	48.7%	43.1%	54.3%	50.8%	50.4%	50.5%
	15	46.5%	44.9%	43.5%	29.4%	57.1%	50.1%	46.8%	42.0%	60.6%	56.6%	57.0%	55.3%
	20	50.0%	47.7%	44.1%	31.1%	61.2%	55.4%	54.8%	47.0%	68.8%	63.7%	63.4%	61.3%
	25	52.4%	50.1%	48.2%	33.8%	65.5%	58.4%	56.4%	48.9%	73.6%	72.2%	70.5%	68.2%
	30	50.1%	49.0%	46.7%	32.7%	63.0%	59.6%	56.6%	48.1%	75.0%	70.4%	71.0%	68.8%
	50	19.3%	18.2%	17.6%	8.1%	24.7%	23.3%	23.1%	18.3%	30.1%	27.6%	27.6%	25.2%
	100	25.9%	24.1%	22.9%	12.6%	29.6%	30.7%	29.2%	21.6%	35.9%	37.2%	36.5%	33.4%
	150	16.9%	16.2%	14.8%	9.4%	22.4%	20.7%	21.7%	15.1%	30.6%	20.3%	20.0%	19.7%
	200	21.3%	19.5%	19.8%	10.5%	25.9%	25.1%	23.9%	16.9%	31.3%	33.1%	32.6%	26.8%
[0.01-0.25]	10	80.0%	78.8%	76.5%	66.1%	86.2%	79.6%	78.3%	74.8%	86.2%	80.0%	78.7%	77.3%
	15	81.8%	77.5%	75.0%	60.6%	87.7%	81.7%	81.2%	75.2%	88.7%	82.6%	81.3%	79.4%
	20	85.2%	80.4%	78.3%	63.4%	90.4%	83.7%	82.8%	78.1%	92.4%	86.4%	86.1%	83.8%
	25	85.6%	78.7%	74.6%	61.9%	92.6%	84.1%	82.4%	77.0%	94.4%	87.8%	86.1%	85.3%
	30	85.0%	80.7%	77.8%	63.5%	92.0%	85.8%	83.6%	75.9%	94.6%	90.6%	88.7%	86.9%
	50	57.4%	54.4%	53.2%	35.3%	65.5%	65.3%	62.9%	50.9%	72.5%	70.9%	72.0%	66.6%
	100	63.0%	59.7%	58.3%	36.8%	70.4%	67.5%	65.9%	54.2%	79.7%	71.9%	73.2%	69.3%
	150	57.6%	53.7%	52.9%	31.0%	66.6%	63.5%	60.5%	49.1%	75.0%	69.9%	69.2%	65.4%
	200	59.8%	56.3%	55.9%	34.5%	70.6%	62.8%	62.0%	49.2%	76.5%	73.0%	73.3%	69.6%
[0.01-0.50]	10	48.9%	48.3%	44.1%	22.7%	55.9%	53.8%	51.1%	37.3%	57.5%	55.6%	55.2%	53.0%
	15	51.4%	48.7%	45.1%	21.2%	57.9%	54.8%	51.8%	34.9%	63.0%	56.5%	55.1%	50.1%
	20	52.3%	51.0%	47.2%	19.6%	61.1%	59.6%	54.9%	34.6%	69.5%	65.2%	63.4%	57.6%
	25	52.3%	50.6%	46.2%	19.5%	63.2%	56.8%	55.4%	35.3%	70.0%	64.8%	62.2%	57.3%
	30	54.1%	50.4%	46.1%	19.6%	63.4%	61.0%	56.0%	36.9%	71.9%	65.0%	64.5%	57.7%
	50	24.9%	25.1%	23.6%	6.9%	27.2%	27.7%	27.1%	12.6%	29.0%	30.4%	29.8%	23.8%
	100	27.5%	28.3%	25.3%	5.1%	30.8%	30.5%	28.7%	14.3%	32.3%	31.2%	30.9%	24.9%
	150	22.3%	21.8%	22.2%	4.3%	25.5%	24.8%	23.4%	9.0%	30.8%	30.1%	26.8%	18.2%
	200	23.8%	22.0%	20.6%	5.1%	26.2%	24.2%	26.0%	9.7%	27.9%	28.3%	28.1%	21.9%
[0.01-0.99]	10	11.2%	10.7%	8.2%	0.4%	11.4%	11.3%	10.1%	0.9%	11.0%	11.2%	11.1%	6.2%
	15	11.1%	10.7%	9.6%	0.1%	11.8%	11.1%	10.3%	0.9%	12.2%	11.9%	11.4%	5.5%
	20	16.6%	16.3%	15.2%	0.3%	17.9%	17.1%	17.1%	1.4%	18.6%	18.2%	17.1%	5.0%
	25	13.5%	13.1%	11.0%	0.4%	14.6%	14.6%	12.3%	0.9%	14.4%	13.7%	14.4%	3.9%
	30	12.2%	11.8%	9.7%	0.7%	12.9%	12.2%	10.7%	1.6%	12.8%	12.4%	11.9%	3.7%
	50	1.8%	1.2%	1.0%	0.0%	1.6%	1.7%	1.5%	0.1%	1.6%	1.1%	1.8%	0.2%
	100	3.8%	4.5%	3.6%	0.0%	4.1%	4.3%	3.9%	0.3%	3.4%	3.1%	3.5%	1.4%
	150	2.2%	2.0%	2.1%	0.0%	2.3%	2.3%	2.4%	0.0%	3.0%	2.8%	2.2%	0.1%
	200	3.0%	2.8%	3.1%	0.0%	3.3%	3.7%	3.9%	0.2%	3.2%	3.4%	3.2%	0.4%

above, Algorithm 4 stops with unsuccessful overall attack at iteration 3 (with the same two stopping criteria). Observe that arc {7} is broken at the first iteration (Figure 8a). In contrast, the attack fails on arc {6}. The survival probability and cost of arc {7} are artificially modified to value 0 and 0, respectively, so that it is selected in the next cut-sets of iterations 2 and 3 (Figures 8b and 8c).

The third treatment concerns the *MCLEC* version of the example in the non-disjoint case. The generated cut-sets and attacking results are displayed in Figure 9. The algorithm on the current illustration stops following the network failure (or equivalently, the attack success). Cut-set $\{0, 1, 7\}$ is broken at iteration 3 in Figure 9c preventing any flow from reaching destination g from source node a.

To further emphasize the importance of budget limitation on the choice of the attack strategies, let us reduce the budget from 200 to 150 in one case, say the *MCLEC* method with non-disjoint cut-set (Figure 10). After iteration 2 (Figure 10b), the remaining budget is only 32, which is not sufficient to launch any attack on any cut-set.



(a) ORIGINAL (b) [0.01-0.25]

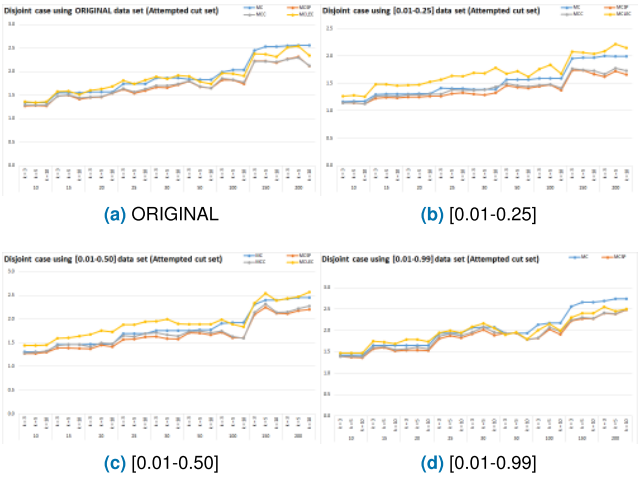
(c) [0.01-0.50] (d) [0.01-0.99]

FIGURE 12. Success rate of the non-disjoint case for each survival probability range.

Thus, the attack stops and fails. Observe that the same attack has reached iteration 3 and was successful by

TABLE 5. Average attempted cut of the disjoint case.

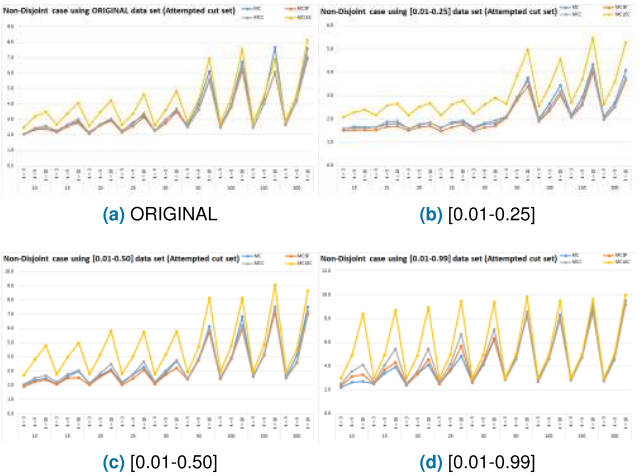
Survival Probability Range	#Nodes	$k=3$				$k=5$				$k=10$			
		MC	*MCSP*	*MCC*	*MCLEC*	*MC*	*MCSP*	*MCC*	*MCLEC*	*MC*	*MCSP*	*MCC*	*MCLEC*
ORIGINAL	10	1.3	1.3	1.3	1.4	1.3	1.3	1.3	1.3	1.3	1.3	1.3	1.4
	15	1.5	1.5	1.5	1.6	1.5	1.5	1.5	1.6	1.5	1.4	1.4	1.5
	20	1.6	1.5	1.5	1.6	1.6	1.5	1.5	1.6	1.6	1.5	1.5	1.7
	25	1.7	1.6	1.6	1.8	1.7	1.5	1.6	1.7	1.7	1.6	1.6	1.8
	30	1.9	1.7	1.7	1.9	1.9	1.7	1.7	1.8	1.9	1.7	1.7	1.9
	50	1.8	1.8	1.8	1.9	1.8	1.7	1.7	1.8	1.8	1.6	1.6	1.7
	100	2.0	1.8	1.9	2.0	2.0	1.8	1.8	1.9	2.0	1.7	1.8	1.9
	150	2.4	2.2	2.2	2.4	2.5	2.2	2.2	2.4	2.5	2.2	2.2	2.3
	200	2.5	2.3	2.3	2.5	2.6	2.3	2.3	2.5	2.6	2.1	2.1	2.3
[0.01-0.25]	10	1.2	1.1	1.1	1.3	1.2	1.1	1.2	1.3	1.2	1.1	1.1	1.3
	15	1.3	1.2	1.3	1.5	1.3	1.2	1.3	1.5	1.3	1.2	1.3	1.5
	20	1.3	1.3	1.3	1.5	1.3	1.3	1.3	1.5	1.3	1.3	1.3	1.5
	25	1.4	1.3	1.3	1.6	1.4	1.3	1.4	1.6	1.4	1.3	1.4	1.6
	30	1.4	1.3	1.4	1.7	1.4	1.3	1.4	1.7	1.4	1.3	1.4	1.8
	50	1.6	1.5	1.5	1.7	1.6	1.4	1.5	1.7	1.6	1.4	1.4	1.6
	100	1.6	1.4	1.5	1.8	1.6	1.5	1.5	1.8	1.6	1.4	1.4	1.7
	150	2.0	1.7	1.8	2.1	2.0	1.7	1.7	2.1	2.0	1.7	1.7	2.0
	200	2.0	1.6	1.7	2.1	2.0	1.7	1.8	2.2	2.0	1.7	1.7	2.1
[0.01-0.50]	10	1.3	1.3	1.3	1.4	1.3	1.3	1.3	1.4	1.3	1.3	1.3	1.4
	15	1.5	1.4	1.4	1.6	1.5	1.4	1.5	1.6	1.5	1.4	1.5	1.6
	20	1.5	1.4	1.4	1.7	1.5	1.5	1.5	1.8	1.5	1.4	1.5	1.7
	25	1.7	1.6	1.6	1.9	1.7	1.6	1.6	1.9	1.7	1.6	1.7	1.9
	30	1.8	1.6	1.7	2.0	1.8	1.6	1.7	2.0	1.8	1.6	1.6	1.9
	50	1.8	1.7	1.7	1.9	1.8	1.7	1.7	1.9	1.8	1.7	1.7	1.9
	100	1.9	1.7	1.8	2.0	1.9	1.6	1.6	1.9	1.9	1.6	1.6	1.8
	150	2.3	2.1	2.1	2.3	2.4	2.3	2.3	2.5	2.4	2.1	2.1	2.4
	200	2.4	2.1	2.1	2.4	2.5	2.2	2.2	2.5	2.5	2.2	2.3	2.6
[0.01-0.99]	10	1.4	1.4	1.4	1.5	1.4	1.4	1.4	1.5	1.4	1.4	1.4	1.5
	15	1.7	1.6	1.6	1.7	1.7	1.6	1.6	1.7	1.7	1.5	1.5	1.7
	20	1.6	1.5	1.6	1.8	1.6	1.5	1.6	1.8	1.6	1.5	1.6	1.7
	25	1.9	1.8	1.9	1.9	1.9	1.9	1.9	2.0	1.9	1.8	1.9	1.9
	30	2.1	1.9	2.0	2.1	2.1	2.0	2.1	2.2	2.1	1.9	2.0	2.1
	50	1.9	1.9	1.9	1.9	1.9	1.9	1.9	2.0	1.9	1.8	1.8	1.8
	100	2.1	1.8	1.8	2.0	2.2	2.0	2.1	2.2	2.2	1.9	2.0	2.0
	150	2.6	2.2	2.2	2.3	2.7	2.3	2.3	2.4	2.7	2.3	2.3	2.4
	200	2.7	2.4	2.4	2.6	2.7	2.4	2.4	2.5	2.7	2.5	2.5	2.5



(a) ORIGINAL (b) [0.01-0.25]

(c) [0.01-0.50] (d) [0.01-0.99]

FIGURE 13. Number of attempted cut-sets of the disjoint case for each survival probability range.



(a) ORIGINAL (b) [0.01-0.25]

(c) [0.01-0.50] (d) [0.01-0.99]

FIGURE 14. Number of attempted cut-sets of the non-disjoint case for each survival probability range.

TABLE 6. Average attempted cut of the non-disjoint case.

Survival Probability Range	#Nodes	$k=3$				$k=5$				$k=10$			
		MC	*MCSP*	*MCC*	*MCLEC*	*MC*	*MCSP*	*MCC*	*MCLEC*	*MC*	*MCSP*	*MCC*	*MCLEC*
ORIGINAL	10	2.1	2.0	2.1	2.5	2.4	2.3	2.4	3.2	2.4	2.4	2.6	3.5
	15	2.2	2.2	2.3	2.7	2.7	2.5	2.7	3.4	2.9	2.8	3.0	4.1
	20	2.2	2.1	2.1	2.6	2.7	2.6	2.7	3.4	2.9	2.9	3.0	4.2
	25	2.2	2.2	2.3	2.7	2.8	2.6	2.7	3.4	3.3	3.1	3.4	4.6
	30	2.3	2.2	2.3	2.6	3.0	2.8	2.9	3.6	3.6	3.5	3.7	4.8
	50	2.6	2.5	2.5	2.8	4.0	3.6	3.7	4.2	6.1	5.5	5.6	6.9
	100	2.6	2.5	2.5	2.8	4.0	3.8	3.9	4.4	6.7	6.2	6.4	7.6
	150	2.8	2.5	2.5	2.8	4.3	4.0	4.0	4.5	7.6	6.1	6.0	6.9
	200	2.7	2.7	2.6	2.9	4.2	4.2	4.2	4.6	7.6	6.9	7.1	8.1
[0.01-0.25]	10	1.6	1.5	1.6	2.1	1.7	1.5	1.6	2.3	1.6	1.5	1.6	2.4
	15	1.6	1.5	1.7	2.2	1.8	1.7	1.9	2.6	1.8	1.7	1.9	2.7
	20	1.6	1.5	1.6	2.2	1.7	1.6	1.8	2.5	1.8	1.7	1.8	2.7
	25	1.6	1.5	1.6	2.1	1.8	1.6	1.9	2.6	1.9	1.8	1.9	2.8
	30	1.6	1.5	1.6	2.2	1.8	1.6	1.8	2.6	1.8	1.7	1.9	2.9
	50	2.1	2.1	2.1	2.6	2.8	2.8	2.9	3.9	3.8	3.4	3.6	5.0
	100	2.0	1.9	1.9	2.5	2.7	2.3	2.5	3.4	3.4	3.0	3.2	4.6
	150	2.2	2.1	2.1	2.7	2.9	2.6	2.7	3.7	4.3	4.0	4.2	5.5
	200	2.1	2.0	2.0	2.6	2.7	2.5	2.5	3.6	4.1	3.7	3.8	5.3
[0.01-0.50]	10	2.0	1.9	2.1	2.7	2.4	2.2	2.5	3.8	2.5	2.4	2.7	4.8
	15	2.1	2.1	2.2	2.8	2.6	2.5	2.8	4.0	3.0	2.5	3.0	5.0
	20	2.1	2.0	2.1	2.8	2.7	2.6	2.8	4.2	3.1	3.0	3.5	5.8
	25	2.1	2.0	2.2	2.8	2.8	2.5	2.8	4.0	3.3	3.1	3.7	5.8
	30	2.1	2.1	2.2	2.8	2.9	2.7	3.1	4.2	3.7	3.2	3.7	5.8
	50	2.5	2.4	2.5	2.9	3.8	3.8	3.9	4.7	6.2	5.8	5.9	8.2
	100	2.5	2.4	2.5	3.0	3.9	3.9	4.0	4.8	6.8	6.0	6.2	8.1
	150	2.6	2.6	2.6	3.0	4.2	4.1	4.2	4.8	7.4	7.2	7.5	9.1
	200	2.6	2.5	2.5	2.9	4.1	3.6	3.6	4.5	7.5	7.0	7.2	8.7
[0.01-0.99]	10	2.2	2.3	2.5	3.0	2.6	3.1	3.6	4.9	2.7	3.3	4.1	8.4
	15	2.5	2.5	2.6	3.0	3.4	3.7	4.0	4.9	3.9	4.3	5.5	8.7
	20	2.4	2.4	2.5	3.0	3.3	3.3	3.6	4.9	4.1	4.6	5.4	8.9
	25	2.5	2.5	2.7	3.0	3.6	3.7	4.2	4.9	4.9	5.7	6.7	9.5
	30	2.7	2.6	2.8	3.0	4.1	4.2	4.4	5.0	6.2	6.3	7.1	9.4
	50	3.0	2.9	2.9	3.0	4.8	4.5	4.6	5.0	8.3	8.4	8.6	9.8
	100	2.9	2.7	2.7	3.0	4.6	4.6	4.6	5.0	8.3	8.0	8.1	9.5
	150	3.0	2.8	2.8	3.0	4.8	4.7	4.7	5.0	9.2	8.6	8.8	9.6
	200	2.9	2.7	2.8	3.0	4.8	4.5	4.5	5.0	9.5	9.1	9.4	10.0

disconnecting cut-set $\{0, 1, 7\}$ when the budget was 200 (Figure 9).

V. IMPLEMENTATION FOR LARGE SCALE NETWORKS

We carry out experimentation using instances generated by Gharbi *et al.* [1] for small sizes and by Yaghlane *et al.* [3] for large sizes to investigate the performance of Algorithms 3 and 4. For each size of instances, we adopt the simulation in Mrad *et al.* [2] to generate survival probabilities in the ranges of [0.01, 0.25], [0.01, 0.50], and [0.01, 0.99], respectively. The small instances consist of networks with sizes between 10 to 30 nodes and each node size has 40 different arc sizes. The large instances consist of networks with sizes varying between 50 and 200 nodes; and each node size has 20 different arc sizes. For each instance, we also calculate a minimum budget to warrant the feasibility of the problem at least for one cut-set to be generated. Hence, we suggest the following formula:

$$budget = f * \alpha \tag{19}$$

In equation (19), f is the value of the min-cost cut-set and α is a random number in the interval [1,2*$kOpt$], where $kOpt$ is the average number of attacks computed based on the simulation performed in Mrad *et al.* [2] for each class of instances (see Table 1 in which the same problem is solved but without budget constraint). This method will ensure that the budget could be sufficiently limiting to have an impactful effect on the success rate of the attack.

We solve all modified instances through an Intel(R) Core(TM) i7 2.00 GHz Personal Computer with 32GB RAM using C++ and CPLEX, version 12.10. The simulation uses 3 different values of k; namely, 3, 5, and 10. We repeat the experiment 100 times for each instance. In addition, we run the approach described in Mrad *et al.* [2] in order to compare the new results with those of unlimited budget. Tables 2-8 below provide the simulation results for Algorithms 3 and 4.

Table 2 reveals that the average run time for each 100 scenarios of *MCSP* and *MCC* is relatively small in general compared to the *MCLEC* method. In fact, in the

TABLE 7. Average run-time (in seconds) of the disjoint case.

Survival Probability Range	#Nodes	$k=3$				$k=5$				$k=10$			
		MC	*MCSP*	*MCC*	*MCLEC*	*MC*	*MCSP*	*MCC*	*MCLEC*	*MC*	*MCSP*	*MCC*	*MCLEC*
ORIGINAL	10	0.01	0.01	0.01	0.23	0.01	0.01	0.01	0.19	0.01	0.01	0.01	0.24
	15	0.01	0.01	0.01	0.38	0.01	0.01	0.01	0.32	0.01	0.01	0.01	0.35
	20	0.01	0.01	0.01	0.55	0.01	0.01	0.01	0.44	0.01	0.01	0.01	0.56
	25	0.01	0.02	0.02	0.69	0.01	0.02	0.02	0.55	0.01	0.02	0.02	0.66
	30	0.02	0.02	0.02	0.82	0.02	0.02	0.02	0.84	0.02	0.02	0.02	0.79
	50	0.03	0.05	0.03	1.86	0.03	0.04	0.03	1.85	0.03	0.04	0.03	1.42
	100	0.05	0.10	0.05	2.96	0.05	0.11	0.06	3.00	0.05	0.09	0.06	2.40
	150	0.09	0.21	0.10	4.93	0.10	0.21	0.11	5.15	0.10	0.19	0.11	4.07
	200	0.12	0.24	0.13	6.22	0.13	0.26	0.14	6.40	0.13	0.23	0.13	5.84
[0.01-0.25]	10	0.01	0.00	0.01	0.20	0.01	0.01	0.01	0.21	0.01	0.01	0.01	0.23
	15	0.01	0.01	0.01	0.34	0.01	0.01	0.01	0.36	0.01	0.01	0.01	0.39
	20	0.01	0.01	0.01	0.43	0.01	0.01	0.01	0.49	0.01	0.01	0.01	0.56
	25	0.01	0.01	0.01	0.51	0.01	0.01	0.01	0.61	0.01	0.01	0.01	0.66
	30	0.01	0.01	0.02	0.61	0.01	0.01	0.02	0.66	0.01	0.01	0.02	0.79
	50	0.02	0.04	0.03	1.67	0.03	0.04	0.03	1.66	0.03	0.04	0.03	1.52
	100	0.04	0.09	0.05	2.74	0.04	0.09	0.05	2.70	0.04	0.08	0.04	2.32
	150	0.08	0.17	0.09	4.73	0.08	0.17	0.09	4.67	0.09	0.15	0.09	4.01
	200	0.11	0.20	0.11	6.07	0.12	0.20	0.13	6.19	0.12	0.18	0.12	6.17
[0.01-0.50]	10	0.01	0.01	0.01	0.24	0.01	0.01	0.01	0.23	0.01	0.01	0.01	0.25
	15	0.01	0.01	0.01	0.41	0.01	0.01	0.01	0.37	0.01	0.01	0.01	0.42
	20	0.01	0.01	0.01	0.56	0.01	0.01	0.01	0.57	0.01	0.01	0.01	0.63
	25	0.01	0.01	0.01	0.69	0.01	0.01	0.01	0.59	0.01	0.01	0.02	0.76
	30	0.01	0.02	0.02	0.76	0.01	0.02	0.02	0.62	0.01	0.01	0.02	0.73
	50	0.02	0.04	0.02	1.82	0.03	0.04	0.02	1.86	0.02	0.04	0.03	1.74
	100	0.04	0.09	0.04	2.93	0.04	0.09	0.04	2.68	0.04	0.08	0.04	2.61
	150	0.07	0.16	0.08	4.55	0.07	0.17	0.09	4.95	0.07	0.15	0.08	4.32
	200	0.09	0.21	0.10	7.06	0.11	0.20	0.11	7.30	0.10	0.19	0.12	6.96
[0.01-0.99]	10	0.01	0.00	0.01	0.22	0.01	0.00	0.01	0.23	0.01	0.00	0.01	0.24
	15	0.01	0.01	0.01	0.39	0.01	0.01	0.01	0.40	0.01	0.01	0.01	0.39
	20	0.01	0.01	0.01	0.56	0.01	0.01	0.01	0.57	0.01	0.01	0.01	0.59
	25	0.01	0.01	0.01	0.68	0.01	0.01	0.01	0.74	0.01	0.01	0.01	0.79
	30	0.01	0.01	0.01	0.75	0.01	0.01	0.01	0.96	0.01	0.01	0.01	0.86
	50	0.02	0.03	0.02	1.81	0.02	0.03	0.02	1.87	0.02	0.03	0.02	1.64
	100	0.04	0.09	0.04	2.63	0.03	0.09	0.04	2.83	0.04	0.08	0.03	2.58
	150	0.06	0.16	0.06	4.42	0.06	0.15	0.06	4.89	0.06	0.16	0.06	4.48
	200	0.08	0.20	0.09	7.52	0.08	0.20	0.08	6.97	0.08	0.20	0.08	6.66

MCLEC, the min-cut algorithm must run repeatedly while the *MCSP* and *MCC* are to be solved only once. It also reveals that when relaxing the budget constraint (*MC*), the execution time is always smaller than those with the budget constraint (which is trivial). Further, the results show that for each algorithm and probability range, the time to solve large instances is almost double than the run time for small instances. It should be clear that the average run time as displayed in Tables 7 and 8 is very small for all methods proving the efficiency of the developed algorithms. Also the standard deviation shows that the variation of the computational time is very small from one scenario to another. Thus, we can conclude that the computational time of our algorithms is stationary.

The success rate for the situation of the disjoint case is almost the same for each method regardless the k values (Figure 11). The MC version outperforms the *MCSP*, *MCC*, and *MCLEC* versions by 76%, 94%, and 100%, respectively. For small instances, the success rate shows the pattern rank of the method given by *MC*, *MCSP*, *MCC*, then *MCLEC*. However, in large instances, the result tends to vary where the pattern is irregular except for *MCLEC* which has the lowest success rate among all scenarios.

In the situation of the non-disjoint case, the values of k affect the success rate for each method (Figure 12) except for large instances with [0.01-0.99] survival probability ranges where the values of k are irrelevant to the success rate. The success rate for all methods in the situation of the non-disjoint case (Figure 12 and Table 4) is mostly outperforming that of the disjoint case (Figure 11 and Table 3). In fact, the set of feasible solutions in the non-disjoint case is larger. The success rates of *MCSP* and *MCC* are comparable in either case. For the *MCLEC* method, the success rates in the disjoint case are much smaller than those of the *MCSP* and *MCC* methods. However, in the non-disjoint case, *MCLEC* success rates get closer to those of the other methods, particularly for large k.

TABLE 8. Average run-time (in seconds) of the non-disjoint case.

Survival Probability Range	#Nodes	$k=3$				$k=5$				$k=10$			
		MC	*MCSP*	*MCC*	*MCLEC*	*MC*	*MCSP*	*MCC*	*MCLEC*	*MC*	*MCSP*	*MCC*	*MCLEC*
ORIGINAL	10	0.01	0.02	0.03	0.42	0.02	0.03	0.05	0.70	0.02	0.04	0.06	1.19
	15	0.02	0.03	0.04	0.65	0.02	0.04	0.06	0.94	0.02	0.06	0.09	1.58
	20	0.02	0.03	0.04	0.76	0.02	0.05	0.07	1.20	0.03	0.08	0.11	1.82
	25	0.02	0.03	0.05	0.85	0.03	0.05	0.07	1.18	0.03	0.11	0.14	1.86
	30	0.02	0.04	0.05	0.86	0.03	0.07	0.09	1.31	0.04	0.14	0.17	2.10
	50	0.04	0.10	0.07	1.98	0.08	0.19	0.15	3.29	0.13	0.37	0.31	4.17
	100	0.07	0.17	0.11	3.67	0.13	0.35	0.26	6.25	0.24	0.73	0.53	8.11
	150	0.11	0.25	0.16	6.46	0.20	0.51	0.37	8.33	0.38	0.98	0.69	13.86
	200	0.15	0.33	0.22	8.49	0.29	0.66	0.49	11.31	0.50	1.45	1.14	18.62
[0.01-0.25]	10	0.01	0.01	0.02	0.33	0.01	0.02	0.03	0.43	0.01	0.02	0.03	0.53
	15	0.01	0.02	0.03	0.50	0.01	0.02	0.04	0.63	0.01	0.03	0.05	0.77
	20	0.01	0.02	0.03	0.63	0.01	0.02	0.04	0.82	0.02	0.03	0.05	0.98
	25	0.01	0.02	0.03	0.70	0.02	0.03	0.05	0.97	0.02	0.04	0.06	1.09
	30	0.02	0.02	0.03	0.78	0.02	0.03	0.05	1.02	0.02	0.04	0.06	1.32
	50	0.04	0.09	0.08	2.37	0.07	0.17	0.16	2.86	0.10	0.27	0.27	3.57
	100	0.06	0.14	0.10	3.82	0.09	0.22	0.19	4.41	0.14	0.39	0.32	6.01
	150	0.10	0.23	0.17	5.59	0.16	0.38	0.31	7.65	0.28	0.78	0.70	10.50
	200	0.13	0.28	0.21	7.51	0.20	0.46	0.38	9.13	0.34	0.90	0.82	12.61
[0.01-0.50]	10	0.01	0.02	0.03	0.46	0.01	0.02	0.04	0.73	0.02	0.03	0.05	1.23
	15	0.01	0.02	0.03	0.64	0.02	0.04	0.06	0.95	0.02	0.05	0.08	1.72
	20	0.01	0.02	0.03	0.88	0.02	0.04	0.06	1.32	0.03	0.07	0.10	2.53
	25	0.02	0.03	0.04	0.99	0.02	0.04	0.06	1.45	0.03	0.08	0.12	2.69
	30	0.02	0.03	0.04	1.02	0.03	0.06	0.07	1.45	0.04	0.10	0.13	2.97
	50	0.03	0.08	0.05	2.81	0.05	0.15	0.12	4.30	0.09	0.26	0.24	7.20
	100	0.06	0.15	0.09	4.95	0.09	0.28	0.19	7.82	0.17	0.48	0.37	12.93
	150	0.09	0.22	0.13	7.53	0.15	0.42	0.30	10.81	0.28	0.83	0.66	19.09
	200	0.11	0.26	0.17	8.78	0.19	0.44	0.31	12.87	0.35	0.97	0.75	23.68
[0.01-0.99]	10	0.01	0.01	0.02	0.58	0.01	0.02	0.03	0.89	0.01	0.02	0.04	1.64
	15	0.01	0.02	0.02	0.79	0.02	0.02	0.04	1.16	0.02	0.03	0.06	2.63
	20	0.01	0.02	0.02	1.00	0.02	0.03	0.04	1.35	0.02	0.05	0.07	3.30
	25	0.01	0.02	0.02	1.13	0.02	0.03	0.04	1.90	0.03	0.06	0.09	4.19
	30	0.02	0.02	0.03	1.04	0.03	0.04	0.05	2.06	0.04	0.08	0.10	4.03
	50	0.03	0.06	0.03	3.09	0.05	0.09	0.05	5.13	0.09	0.16	0.11	9.59
	100	0.05	0.14	0.06	5.56	0.07	0.22	0.10	9.25	0.16	0.39	0.18	16.84
	150	0.07	0.18	0.08	8.56	0.11	0.30	0.14	13.13	0.21	0.58	0.28	25.58
	200	0.09	0.20	0.11	9.75	0.15	0.34	0.18	17.17	0.29	0.71	0.39	33.06

In terms of the number of attempted cut-sets, the disjoint case provides the same results for all methods regardless of the values of k (Figure 13 and Table 5). This is due to the restriction on alternative paths from the source to the destination. This limitation represents an upper bound of the available disjoint cut-sets (see Mrad *et al.* [2]). Thus, increasing resources will not result in increasing the number of attempted cut-sets. In contrast, when dealing with the non-disjoint case, the number of attempted cuts increases with k for all three methods (Figure 14 and Table 6). This number may even get close to k when the links of the network could be sufficiently robust (i.e., in case where the survival probability upon attack may arbitrarily be drawn from the entire range of the interval [0, 1]). In fact, more cut-sets need to be tried to end up disconnecting the network or else ending up with a failed attack. This situation can be beneficial for the attacker when the attacking resources are available to target several cut-sets. It should be clear however that even with a large number of attempted cut-sets, the success rate can be very low. This is due to the resource limitations. In fact, the results in Mrad *et al.* [2] corresponding to the first version of the problem (with no budget constraint) display success rates exceeding 18% as opposed to the 0-3% success rates obtained in Tables 3 and 4.

VI. CONCLUSION AND DIRECTIONS FOR FUTURE RESEARCH

In this paper, we develop two algorithms to approach the problem of the *k vital cut-sets* of a network under limited attacking resources in each of three versions. The first one treats the attack on up to *k cut-sets* that maximizes the probability of disconnecting the network. The second version deals with the min-cost attack of up to *k cut-sets*. The third version combines the two versions above by identifying the *k vital cut-sets* that minimize the expected cost of an attack. Resource limitations are reflected by a budget constraint.

The *k vital cut-set* problem with limited attacking resources may be viewed as an important extension of other interdiction network concepts such as the Most *k vital Arc* problem and the *k-hub* problem discussed above.

Each of the proposed algorithms embeds three subroutines corresponding to the three versions of the problem. The algorithm calls any subroutine when appropriate. One algorithm investigates the case of disjoint cut-sets and the other one takes care of the non-disjoint case.

The attack is conducted sequentially over the different links of the selected cut-sets. A link is tried at most once. The attacker has full information on the attack outcome on a particular link before proceeding to select another link. The attack stops if one cut-set is totally disabled, the budget is depleted, a full path set is identified to survive regardless of future possible attacks, or all identified *k cut-sets* are unsuccessfully tried, whichever occurs first. Obviously, only the first stopping criterion reflects the attack success.

The solution method relies on a simulation-optimization method that extends the approach by Mrad *et al.* [2] which is restricted to the first version of maximizing the probability of an attack with abundant attacking resources. Techniques from Yaghlane *et al.* [3], [4] including the min-cut and the shortest path algorithms are adapted to the current problem accounting for the budget constraint. The solution method proved to be efficient as the run-time is very short even for large networks.

The results clearly reflect the dependence of the success rate of attacks on the resource availability. In fact, the first version of the problem as treated by Mrad *et al.* [2] under the relaxation of the budget constraint yields considerably higher success rates particularly when the network can have high survival probabilities (in the full range of the interval [0, 1]). Further, the illustrations provided in the paper show the effect of budget limitations on the number and the choice of the cut-sets to target. These results can be highly informative both to the attacker and the defender of the network. Indeed, the attackers will be aware that important attacking resources must be available to expect high success rates of their attacks. Defenders on the other hand may opt for defensive strategies that force the attacker to use excessive attacking resources to be able to disconnect the network (such as multiple access barriers, strengthened protecting material, computer firewalls, etc.).

Future work may extend the *k Vital cut* Problem with constrained attacking resources to account for multiple attacks per link rather than a single attack. From defensive viewpoint, it is possible to consider which *vital cut-sets* to reinforce under limited defensive resources. Another important avenue for future research would be to identify the most *vital path sets* to reinforce rather than cut-sets.

REFERENCES

[1] A. Gharbi, M. N. Azaiez, and M. Kharbeche, "Minimizing expected attacking cost in networks," *Electron. Notes Discrete Math.*, vol. 36, pp. 947–954, Aug. 2010.

[2] M. Mrad, U. S. Suryahatmaja, A. B. Yaghlane, and M. N. Azaiez, "Optimal k cut-sets in attack/defense strategies on networks," *IEEE Access*, vol. 8, pp. 131165–131177, 2020.

[3] A. B. Yaghlane, M. Mrad, A. Gharbi, and M. N. Azaiez, "An exact method for solving a least-cost attack on networks," in *Advances in Reliability Analysis and Its Applications*, M. Ram and H. Pham, Eds. Cham, Switzerland: Springer, 2020, pp. 343–359.

[4] A. B. Yaghlane, M. N. Azaiez, and M. Mrad, "System survivability in the context of interdiction networks," *Rel. Eng. Syst. Saf.*, vol. 185, pp. 362–371, May 2019.

[5] L. A. T. Cox, "Making telecommunications networks resilient against terrorist attacks," in *Game Theoretic Risk Analysis of Security Threats*, vol. 128, V. M. Bier and M. N. Azaiez, Eds. Boston, MA, USA: Springer, 2009, pp. 175–197.

[6] M. Bellmore and H. D. Ratliff, "Optimal defense of multi-commodity networks," *Manage. Sci.*, vol. 18, no. 4, pp. B-174–B-185, Dec. 1971.

[7] H. Frank, I. T. Frisch, R. Van Slyke, and W. S. Chou, "Optimal design of centralized computer networks," *Networks*, vol. 1, no. 1, pp. 43–57, 1971.

[8] D. L. Alderson, D. Funk, and R. Gera, "Analysis of the global maritime transportation system as a layered network," *J. Transp. Secur.*, vol. 13, nos. 3–4, pp. 291–325, Dec. 2020.

[9] U. Kanturska, J.-D. Schmöcker, A. Fonzone, and M. G. H. Bell, "Improving reliability through multi-path routing and link defence: An application of game theory to transport," in *Game Theoretic Risk Analysis of Security Threats*, vol. 128, V. M. Bier and M. N. Azaiez, Eds. Boston, MA, USA: Springer, 2009, pp. 199–227.

[10] V. M. Bier and K. Hausken, "Defending and attacking a network of two arcs subject to traffic congestion," *Rel. Eng. Syst. Saf.*, vol. 112, pp. 214–224, Apr. 2013.

[11] Y. Fang, G. Sansavini, and E. Zio, "An optimization-based framework for the identification of vulnerabilities in electric power grids exposed to natural hazards," *Risk Anal.*, vol. 39, no. 9, pp. 1949–1969, Sep. 2019.

[12] J. Obert, I. Pivkina, H. Huang, and H. Cao, "Proactively applied encryption in multipath networks," *Comput. Secur.*, vol. 58, pp. 106–124, May 2016.

[13] F. Hu, Y. Lu, A. V. Vasilakos, Q. Hao, R. Ma, Y. Patil, T. Zhang, J. Lu, X. Li, and N. N. Xiong, "Robust cyber–physical systems: Concept, models, and implementation," *Future Gener. Comput. Syst.*, vol. 56, pp. 449–475, Mar. 2016.

[14] X.-Y. Zhang, Z. Zheng, and K.-Y. Cai, "A fortification model for decentralized supply systems and its solution algorithms," *IEEE Trans. Rel.*, vol. 67, no. 1, pp. 381–400, Mar. 2018.

[15] N. R. Magliocca, K. McSweeney, S. E. Sesnie, E. Tellman, J. A. Devine, E. A. Nielsen, Z. Pearson, and D. J. Wrathall, "Modeling cocaine traffickers and counterdrug interdiction forces as a complex adaptive system," *Proc. Nat. Acad. Sci. USA*, vol. 116, no. 16, pp. 7784–7792, 2019.

[16] J. Zhang, J. Zhuang, and B. Behlendorf, "Stochastic shortest path network interdiction with a case study of Arizona–Mexico border," *Rel. Eng. Syst. Saf.*, vol. 179, pp. 62–73, Nov. 2018.

[17] A. Reilly, L. Nozick, N. Xu, and D. Jones, "Game theory-based identification of facility use restrictions for the movement of hazardous materials under terrorist threat," *Transp. Res. E, Logistics Transp. Rev.*, vol. 48, no. 1, pp. 115–131, Jan. 2012.

[18] N. Bricha and M. Nourelfath, "Critical supply network protection against intentional attacks: A game-theoretical model," *Rel. Eng. Syst. Saf.*, vol. 119, pp. 1–10, Nov. 2013.

[19] A. B. Yaghlane and M. N. Azaiez, "Systems under attack-survivability rather than reliability: Concept, results, and applications," *Eur. J. Oper. Res.*, vol. 258, no. 3, pp. 1156–1164, May 2017.

[20] L. Xiao, D. Xu, N. B. Mandayam, and H. V. Poor, "Attacker-centric view of a detection game against advanced persistent threats," *IEEE Trans. Mobile Comput.*, vol. 17, no. 11, pp. 2512–2523, Nov. 2018.

[21] C. Casorrán, B. Fortz, M. Labbé, and F. Ordóñez, "A study of general and security Stackelberg game formulations," *Eur. J. Oper. Res.*, vol. 278, no. 3, pp. 855–868, Nov. 2019.

[22] Y. Li, S. Qiao, Y. Deng, and J. Wu, "Stackelberg game in critical infrastructures from a network science perspective," *Phys. A, Stat. Mech. Appl.*, vol. 521, pp. 705–714, May 2019.

[23] E. Israeli and R. K. Wood, "Shortest-path network interdiction," *Networks*, vol. 40, no. 2, pp. 97–111, Jan. 2002.

[24] X. Wei, C. Zhu, K. Xiao, Q. Yin, and Y. Zha, "Shortest path network interdiction with goal threshold," *IEEE Access*, vol. 6, pp. 29332–29343, 2018.

[25] A. Caprara, M. Carvalho, A. Lodi, and G. J. Woeginger, "Bilevel knapsack with interdiction constraints," *INFORMS J. Comput.*, vol. 28, no. 2, pp. 319–333, May 2016.
[26] J. C. Smith and Y. Song, "A survey of network interdiction models and algorithms," *Eur. J. Oper. Res.*, vol. 283, no. 3, pp. 797–811, Jun. 2020.
[27] J. L. Walteros, A. Veremyev, P. M. Pardalos, and E. L. Pasiliao, "Detecting critical node structures on graphs: A mathematical programming approach," *Networks*, vol. 73, no. 1, pp. 48–88, Jan. 2019.
[28] G. Karakose and R. G. McGarvey, "Optimal K-node disruption on a node-capacitated network," *Optim. Lett.*, vol. 13, no. 4, pp. 695–715, Jun. 2019.
[29] M. Yahyaei, M. Bashiri, and M. Randall, "A model for a reliable single allocation hub network design under massive disruption," *Appl. Soft Comput.*, vol. 82, Sep. 2019, Art. no. 105561.
[30] T. L. Lei, "Evaluating the vulnerability of time-sensitive transportation networks: A hub center interdiction problem," *Sustainability*, vol. 11, no. 17, p. 4614, Aug. 2019.
[31] P. Ramamoorthy, S. Jayaswal, A. Sinha, and N. Vidyarthi, "Multiple allocation hub interdiction and protection problems: Model formulations and solution approaches," *Eur. J. Oper. Res.*, vol. 270, no. 1, pp. 230–245, Oct. 2018.
[32] H. Quadros, M. C. Roboredo, and A. A. Pessoa, "A branch-and-cut algorithm for the multiple allocation R-hub interdiction median problem with fortification," *Expert Syst. Appl.*, vol. 110, pp. 311–322, Nov. 2018.
[33] Y. Song and S. Shen, "Risk-averse shortest path interdiction," *INFORMS J. Comput.*, vol. 28, no. 3, pp. 527–539, Jul. 2016.
[34] A. Atamtürk, C. Deck, and H. Jeon, "Successive quadratic upper-bounding for discrete mean-risk minimization and network interdiction," *INFORMS J. Comput.*, vol. 32, no. 2, pp. 346–355, Oct. 2019.
[35] R. Powell, "Defending against terrorist attacks with limited resources," *Amer. Political Sci. Rev.*, vol. 101, no. 3, pp. 527–541, Aug. 2007.
[36] R. Zenklusen, "Network flow interdiction on planar graphs," *Discrete Appl. Math.*, vol. 158, no. 13, pp. 1441–1455, Jul. 2010.
[37] M. Ravishankar, T. Stephan, and T. Perumal, "Time dependent network resource optimization in cyber–physical systems using game theory," *Comput. Commun.*, vol. 176, pp. 1–12, Aug. 2021.

MEHDI MRAD received the Ph.D. degree in operations research from the University of Tunis. He is currently an Associate Professor with the Department of Industrial Engineering, King Saud University. His interests include network design, vehicle routing, scheduling, and cutting problems. He is attracted by the application of different optimization techniques to model and solve real life complicated problems from different industrial fields.

UMAR S. SURYAHATMAJA received the B.S. degree in industrial engineering from Gadjah Mada University, Indonesia, in 2008, and the M.S. degree in industrial engineering from King Saud University, Riyadh, Saudi Arabia, in 2017, where he is currently pursuing the Ph.D. degree in industrial engineering.

ASMA BEN YAGHLANE is a Doctor in management science. She also teaches operations research and decision analysis courses with the Tunis Business School. She is also a member of the Business Analytics and DEcision Making Laboratory. She has some publications, including *European Journal of Operational Research* and *Reliability Engineering and System Safety*. She has offered a number of presentations at reputed international conferences. Her research interest includes mainly developing operations research tools in approaching the problems of systems under the threats of intentional attacks. Particularly, she attempts to determine optimal defense/attack strategies in the context of terrorist attacks and security threats.

M. NACEUR AZAIEZ is currently a Professor of operations research with the Business Analytics Department, Tunis Business School, University of Tunis, Tunisia. He is also the dean of the school. His research interests include a variety of applications of operations research in several fields, including security management, water management, operations management, and healthcare.