IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received July 2, 2020, accepted July 9, 2020, date of publication July 16, 2020, date of current version July 28, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.3009625

Optimal *k Cut-Sets* in Attack/Defense Strategies on Networks

MEHDI MRAD[1], UMAR S. SURYAHATMAJA[1], (Graduate Student Member, IEEE), 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 The paper investigates the most critical *k cut-sets* to target in the context of attacks on networks. The targeted network fails if the attack successfully disables a full cut-set. The attack process is dynamic and targets one link at a time. We assume perfect information, so that the attacker picks the next link to target after identifying the outcome of the attack on the previous one. The attack may continue to reach up to *k cut-sets*. The problem is to identify which cut-sets to target (referred to as the *k-critical cut-sets* of the network) that maximize the probability of a successful attack. We distinguish both cases of disjoint and non-disjoint cut-sets. We develop an algorithm for each case with illustrations. We explore the large scale of networks and offer guidelines on the corresponding defensive strategies.

INDEX TERMS Cut-set, defense, interdiction networks, multiple-attacks, path-set.

I. INTRODUCTION

The increased frequency of intelligent threats has given rise to a number of new concepts to approach the problem including survivability of systems under attack, adversarial risk analysis, interdiction network and adversarial network. Many of the developed models arise in a game-theoretic framework to investigate the interaction between attackers and defenders. In such situations, the defender seeks to strengthen the targeted system while the attacker attempts to use the system's weaknesses in order to disable it.

Networks represent attractive targets for intentional attacks. Attacking a network may occur in various ways such as disturbing the flow, decreasing the network capacity, or totally disconnecting nodes/links. Examples of attacks on networks may include cyber-attacks on computer systems, missile attacks on water distribution networks, and blocking roads in front of trucks in a transportation context. Attacks on network models usually account for an attacker (an individual or an organization) and a defender (often the body in charge of protecting and preserving the functionality of the network). In this paper, we will consider a failure of a node/link if it is totally disabled and hence unable to receive/send flow.

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

A. MOTIVATION

Consider for instance that a government, facing some instability, attempts to block rebels based initially at a given city from backing their allies at the capital. Hence, the government's forces seek to control the traffic in the main roads preventing the rebels from advancing toward their destination. The arrival of the rebels to the capital may be viewed as sending a flow from the source to the sink of the network. Given that it might be difficult to distinguish between regular citizens and rebels, strengthening the control system over various cut-sets of the roads of interests may give higher reliability to the success of the government's mission.

Another interesting application would be to extend the classical minimum bisection problem by partitioning the network of interest into several rather than two disconnected sub-networks as previously treated by Ben Yaghlane *et al.* [1]. In fact, they consider an integer program that turns out to be reasonably efficient even for large networks. The challenge is whether it is still possible to suggest an efficient tool when the network is to be partitioned into a larger number of sub-networks.

A third application; which may arise in a business setting, is as follows. Suppose that a business firm (the attacker in our context) would like to sabotage some alliance (the attack process in this situation) between a local competitor

and a famous international company. Suppose also that the attacker has received some information about the arrival of a delegation of the foreign company to sign the necessary papers. The attack strategy would be to prevent this delegation from traveling to the local competitor's head-quarter. The idea is to make a large number of fake reservations through selected connecting flights in a way to make it impossible for the delegation to find enough seats on any path of correspondences linking the source and the destination cities. This is equivalent to attacking one full cut-set. In anticipation of overbooking as practiced by many airline companies, the attacker may opt for disabling several cut-sets to increase the chance of preventing the delegation from arriving on time.

B. PAPER ORGANIZATION

The remainder of the paper is organized as follows. Section II reviews the literature. Section III provides the problem statement. Section IV suggests a solution methodology with illustrations. Section V explores the large scale of networks through an optimization-simulation approach. Finally, section VI serves for conclusions.

II. LITERATURE REVIEW

In the context of protecting against intelligent threats, Ben Yaghlane and Azaiez [2] develop the concept of system survivability upon attack as opposed to the classical concept of system reliability. Moreover, Yaghlane *et al.* [1] adapt system survivability to networks and distinguish two cases of perfect and absence of information. In the latter case, they develop two distinct definitions of network survivability; one related to the attacker, and the other to the defender. They investigate the relationship between these two definitions as well as their position with respect to network reliability. Gharbi *et al.* [3] consider the attacker problem to fully disable a network. They identify the optimal cut-set to target. Minimizing the expected attack cost represents the optimality criterion for this model. The authors 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.* [4] identify an exact method for the problem treated by Gharbi *et al.* [3].

Cox [5] discusses the resilience of telecommunication networks subjected to intentional threats. In this context, the failure of a link occurs when the received flow exceeds the link capacity. Cox [5] elaborates on various solution methods relying basically on increasing capacities of links and rerouting of flow upon failure of a particular link. Bellmore and Ratliff [6] and Frank *et al.* [7] investigate similar problems. Away from telecommunication networks, a variety of other network problems are investigated. Ausseil *et al.* [8] consider some military deception strategies that include falsifying targets and hiding routes on the network to deceive the attacker. Alderson, *et al.* [9] assess the node interdiction impact on the global maritime transportation. They consider the node as the seaports and maritime chokepoints where a disruption will lead to increase the re-route cost.

Recently, the literature on capacitated survivable network design problems has considerably expanded. In particular, various models for simultaneously attacking k links are considered. Attacks consist on disturbing flow circulation. Defensive strategies call for increasing the capacity of some selected links at the lowest cost so that any flow at a failing link can be rerouted. The problem is strongly NP-Hard as shown by Tomatore *et al.* [10]. Brightwell *et al.* [11] apply relaxations to derive polynomial time solutions through successive shortest path problems. Kanturska *et al.* [12] discuss a transportation network model where multi-path routing and link defense are used for reliability improvement.

In interdiction network literature, various models are investigated. Wei *et al.* [13] propose algorithms to optimize the consumption of attacker's resources on the shortest path interdiction model by limiting the network capacities. They approach the problem using various tools including Benders decomposition, Lagrange duality, set-covering, and Lagrange approximation algorithms. Magliocca *et al.* [14] consider counter-drug interdiction to disrupt the cocaine distribution. Zhang *et al.* [15] study the stochastic shortest-path interdiction problem using probabilistic detection likelihood to solve Ports-of-Entry problem; such as human trafficking, illegal drug distribution, and terrorism. Fang *et al.* [16] develop a framework for the electrical power grid interdiction that balances the difficulties in predicting the hazard with the over conservatism in the attacker-defender models. Smith and Song [17] provide a recent review on interdiction networks. It focuses on the mathematical formulation and solution approaches using the dual model and the row generation methods for the network interdiction problem. The paper also summarizes the basic assumptions of the problem and the recently developed concepts.

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.* [18] discuss a framework to detect the set of critical nodes that maximize the possibility of disconnecting the network. Karakose and McGarvey [19] propose a path-based formulation and multi-commodity flow-based formulations to identify the optimal *k-nodes* 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.* [20] design a single allocation hub network reliable to massive disruption using bi-objective quadratic model. Lei [21] determines the critical air transportation hub facilities to be protected to reduce the transit time using integer linear programming formulation. Ramamoorthy *et al.* [22] identify n critical hubs from a set of candidate hubs to reduce the routing cost using Bender decomposition. Quadros *et al.* [23] 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.

Many of the interdiction network models are approached using game theory. Ben Yaghlane *et al.* [1] investigate Nash equilibrium in various setting of the network survivability problem considering both zero and nonzero sum games. Xiao *et al.* [24] 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 action is made subjectively.

Bricha and Nourelfath [25] 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. Casorran *et al.* [26] consider multi attackers and one defender in general and security Stackelberg games. Li *et al.* [27] studies Stackelberg games with two attackers and one defender that fight over the hub nodes by considering cost-sensitive parameters.

In the same spirit of identifying the *k-critical node* problem and the *n-critical hub problem*, the current paper attempts to identify the *k-critical cut-sets* in a network that an attacker may target. The attacker objective would be to increase the chance of a successful attack that may completely prevent a flow from reaching its destination. The problem is highly challenging and the focus on this paper will be limited to the case of perfect information related to the outcome of the attack on a single link. That is, the attacker strategy is dynamic and accounts for the results of previous trials when identifying the next link to target.

III. THE PROBLEM

We will start by introducing the general context of the problem. We will elaborate on the min-cut problem treated in Yaghlane *et al.* [1]. Then, we will discuss some drawbacks of the model consisting of targeting a single cut-set. Next, we will introduce the *k cut problem*.

A. THE PROBLEM SITUATION

We consider a network subjected to the threat of attacks. It is considered as operating if it succeeds to send some flow from its sources to its destinations. We assume one attacker targeting the network and one defender attempting to protect it. An attack is considered as successful if it allows disabling the network by preventing any flow from reaching the destinations from its various sources. This occurs if a full cut-set is disabled. In this context, Yaghlane *et al.* [1] determine the optimal cut-set to attack. Optimality is considered with respect to maximizing the probability of disabling the network; or equivalently, to minimizing the network survivability. Without loss of generality, Yaghlane *et al.* [1] show that the network can be reduced to a one with a single source and a single destination.

We display below the linear programming (LP) formulation suggested by Yaghlane *et al.* [1] to generate the *min cut-set*:

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

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

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

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

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

In the above formulation, V is the set of nodes, E is the set of arcs between nodes i and j, and p_{ij} is the probability of each $arc(i,j)$ to survive an attack. The nodes start from the source which is node 1 and sink at a destination which is node n. A cut-set can be defined as a set of arcs that is partitioning the set of nodes V into two disjoint subsets named 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 and zero else.

At the same time, Yaghlane *et al.* [1] solve the defender problem by determining the optimal breakthrough paths (i.e., those having the highest probability; referred to as network survivability) to operate.

One of the limitations of the models by Yaghlane *et al.* [1] consists on considering a single cut-set to target by the attacker. In fact, contenting by attacking one cut-set does not seem to be favorable to the attacker given that the probability of a successful attack is still low in general. Indeed, disabling a full cut-set is equivalent to disconnecting all corresponding links. It follows that the probability of success is obtained as a product of probabilities in case of independent failure probabilities of links in the context of distinct attacks. Therefore, we assume, in this paper, that the attacker may target several cut-sets. We are interested in determining which critical k *cut-sets* to attack. Here, k might be dictated by a budget or time constraint.

B. THE PROBLEM STATEMENT

We consider an attacker trying to disable a network by disconnecting one cut-set among k *cut-sets* to be targeted over time while observing the result of an attack on a given link of a selected cut-set. This is a situation of perfect information on the outcome of the previous attacks. This may arise in situations where it is easy to detect the result of an attack such as destroying a road or a pipeline through a missile attack or checking if the delegation of the motivation above has obtained boarding passes in a given connection flight. It should be clear however that this need not be true in other situations where the outcomes of an attack may not be observable or may take some time to be detected. In the context of our problem of perfect information, the attack process is dynamic and attack decisions are taken sequentially.

We assume one attack per link on each considered cut-set. The choice of the next link/cut-set to be tried accounts for the previous observed results. Each link has some survival

probability upon attack. The attack ends in one of three possible situations, whichever occurs first:

1) The network is disabled and therefore the attack succeeds
2) The network is guaranteed to send flow to the destination; in which case the attack fails
3) All selected *k cut-sets* are unsuccessfully tried; in which case the attack fails

The attacker objective is to determine the best attack strategy that maximizes the chance of disabling the network. That is, the attacker will gradually select and the cut-sets to target (up to *k cut-sets*) and the order in which cut-sets and links within a cut-set will be tried. Note that disabling a cut-set is equivalent to disabling a parallel system (Azaiez and Bier [28]). Ben Yaghlane and Azaiez [2] identify the sequence in which a parallel system is to be attacked in order to minimize the attacker cost. Essentially, when attacks consume comparable resources, the result is reduced to starting by targeting the most reliable component and continuing in this fashion until one component resists to an attack or all components are disabled. In this paper, we will use the same strategy once a cut-set is identified for attack.

IV. SOLUTION METHODOLOGY

We will distinguish two cases. The first one deals with the situation where the eligible cut-sets to select are disjoint. The second one is the general case of any cut-sets to target. At a first glance, it looks awkward to opt for an attack on disjoint cut-sets. In fact, more resources are likely to be consumed. In addition, potentially failing links in previously attacked cut-sets might be unexploited in selecting the next cut-sets to target. However, it might be justified to opt for disjoint cut-sets in some situations. For instance, assume that attacks on different cut-sets require different tools, trainings, and hence teams. Then, it would be natural that each team takes care of a separate cut-set. Other justifications may include security reasons (such as not simultaneously putting different teams under the risk of retaliations), or space/logistics reasons and so on.

In order to show the effectiveness of our proposed methods, we will compare each algorithm with a naive approach. The naive approach will select randomly the cut-set to be attacked and a successful attack will be determined only on the current cut-set. The details of the naive approach will be shown in the next sub section.

A. CASE OF DISJOINT CUT-SETS

We start by the simple case where all cut-sets to be targeted must be disjoint. The subset of all disjoint cut-sets can clearly be represented through a series-parallel system. In this situation, when a cut-set is unsuccessfully tried, all corresponding links will be entirely ignored from consideration when identifying the next cut-set to target. However, disabled links within an attacked cut-set will be kept in memory to identify when the entire network fails. This is different from the case where the cut-sets are non-disjoint and failing links should be kept in memory and for identification of network functionality and for potential consideration with other cut-sets. Moreover, untried links in the non-disjoint case can be considered for other cut-sets to target.

Algorithm 1 Naive Approach for Disjoint Cut-Set

```
Let G(V, E) is the graph where V is the set of nodes and E
is the set of links
set C = ∅ (the generated cut-set)
succeed = false;
n = a number of cut-sets to be targeted ({3, 5, 10})
i = 0;
while succeed = false and i < n do
   generate randomly new survival probabilities for the arcs
   that have not been attacked
   set C = solve the min-cut problem on graph G(V, E) to
   get a cut-set based on the randomly generated survival
   probabilities
   if C = ∅ then
      succeed = false;
      break;
   end if
   for all link a in cut-set C do
      Attack the link a;
      if attack failed then
         succeed = false;
         break;
      else
         succeed = true;
      end if
   end for
   Update the probability of the arcs in the cut-set C to
   1.0 in order not to be considered in the next cut-set
   i++;
end while
if succeed = true then
   Attack succeed;
else
   Attack failed;
end if
```

Algorithms 1 and 2 will identify the naive and optimal attack strategy, respectively, targeting up to *k cut-sets* in an attempt to fully partition the network under the disjoint cut-set framework. Given that we approach the problem in the dynamic case, we will sequentially generate a cut-set. This is obtained optimally using the min-cut approach as identified in Yaghlane *et al.* [1].

Note that the dynamic case is supposed to use the information on the output of previous attacks when targeting the next cut-set. However, when cut-sets are disjoint, there is no opportunity to use failed links of previous survived cut-sets to determine a new cut-set susceptible of having low survival probability.

Algorithm 2 Disjoint Cut-Set

Let $G(V, E)$ is the graph where V is the set of nodes and E is the set of links
Stop = false; succeed = false; empty = true; l = 1
set $W = \emptyset$ (set of disabled links)
set $A = \emptyset$ (current cut-set)
while stop = false **do**
  set A = generate the new optimal cut from the graph $G(V, E)$ using the min-cut problem as identified in Yaghlane *et al.* [1]
  if $A = \emptyset$ **then**
    stop = true; succeed = false;
  else
    sort A using the non-descending rule of survival probability P_a of its links
    for all link a in A based on the above ordering **do**
      attack the link;
      if attack fails **then**
        break;
      else
        update W with the current link
        if link a is the last link in A **then**
          stop = true; succeed = false;
        else
          if empty = false **then**
            if W contains a full cut-set **then**
              stop = true; succeed = true;
              break;
            end if
          end if
        end if
      end if
    end for
    if $W = \emptyset$ **then**
      empty = false;
    end if
  end if
  if stop = false **then**
    for all link a in A **do**
      $P_a = 1.0$;
    end for
    if $l < k$ **then**
      $l = l + 1$;
    else
      stop = true; succeed = false;
    end if
  end if
  clear A;
end while

1) EXAMPLE OF DISJOINT CUT-SETS

Consider the network depicted in Figure 1. It has seven nodes and nine arcs with their survival probabilities. A cut-set will be generated and attacked sequentially by iterating the procedure as explained in Algorithm 2. As soon as a cut-set is fully disabled, the network fails to supply the destination node with flow. Based on Yaghlane *et al.* [1], the attack will start by the weakest cut-set through the min-cut problem.

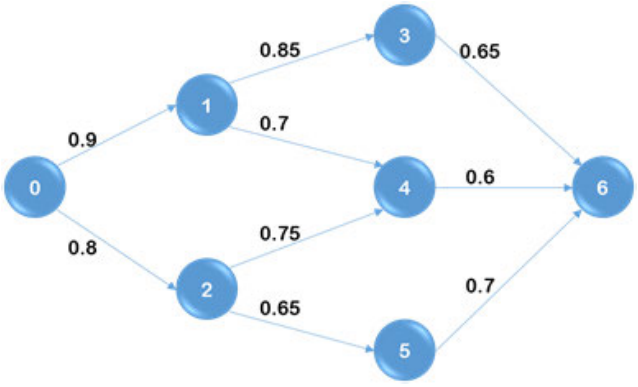


FIGURE 1. Initial network.

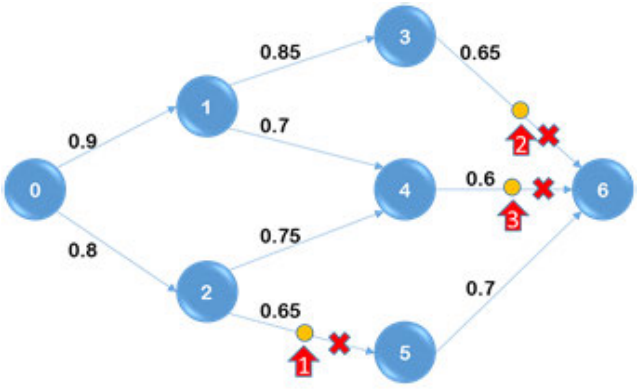


FIGURE 2. Iteration 1 of first scenario.

The yellow dot shows the identified cut-set $\{2-5, 3-6, 4-6\}$ and the numbered arrow shows the sequence of attacks so that the attack starts with the highest survival probabilities (evens are broken randomly). The cross sign means that the attack is successful on the corresponding link. From the scenario exhibited in Figure 2, the network is fully disabled at the first attempt. As a result, the attacker stops and the attack is considered as successful.

In a second scenario of the same example, the attack on the first identified cut-set fails (Figure 3) upon the failure on link $\{2-5\}$ as shown by the green mark. Thus, a new cut-set is to be determined using the min-cut problem after discarding the tried one (by artificially changing the survival probabilities of its various links to 1.0 as specified in Figure 4-6 below).

This new cut-set at the new iteration is $\{0-1, 0-2\}$ (Figure 4). The attempt on link $\{0-1\}$ is assumed to fail. We update the survival probabilities of this cut-set and generate a new cut-set (Figure 5).

The third iteration generates the cut-set $\{1-3, 1-4, 2-4, 5-6\}$. Assume that the attacks on links $\{1-3\}$, $\{2-4\}$, and $\{1-4\}$ are successful. If the attack on link $\{5-6\}$ also succeeds, then the full cut-set is disconnected and the attack is successful (Figure 5). Stopping criterion 1 applies.

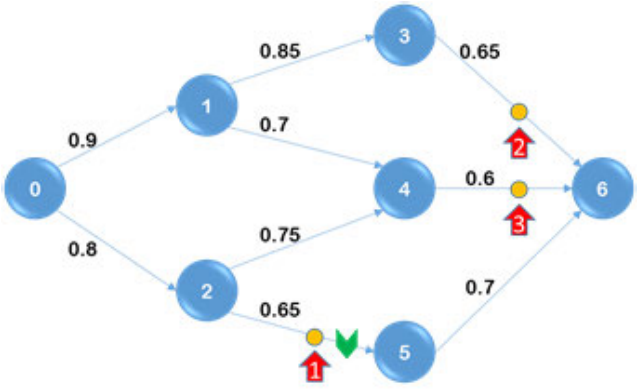


FIGURE 3. Iteration 1 of second scenario.

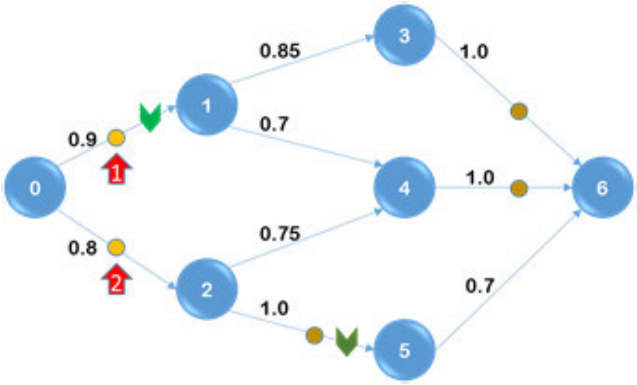


FIGURE 4. Iteration 2 of second scenario.

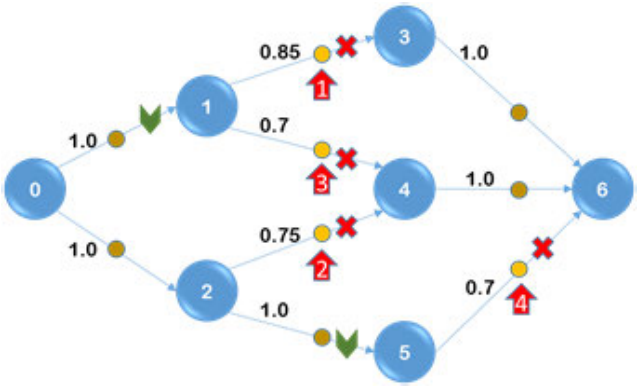


FIGURE 5. Iteration 3 of second scenario (successful attack).

However, suppose that link $\{5-6\}$ resists to the attack. It follows that the overall attack fails (Figure 6) because a full path-set $\{0-2-5-6\}$ will not be attacked any further and will be able to send flow from the source to the destination. Hence, stopping criterion 2 occurs.

For this problem, assuming that $k = 3$, then the most critical cut-sets to target are identified to be respectively $\{2-5, 3-6, 4-6\}$, $\{0-1, 0-2\}$, and $\{1-3, 1-4, 2-4, 5-6\}$.

B. CASE OF NON-DISJOINT CUT-SETS

We consider now the general case of networks where targeted cut-sets need not be disjoint. That is, the same link may belong to several cut-sets. Hence, a non-attacked link of a targeted cut-set can be considered in another cut-set. Moreover, a link that belongs to more than one cut-set, once failing, will give rise to an update (based on the state of functionality) of these cut-sets. We display below an appropriate algorithm to identify the naive (Algorithm 3) and the optimal (Algorithm 4) attack policy in this case.

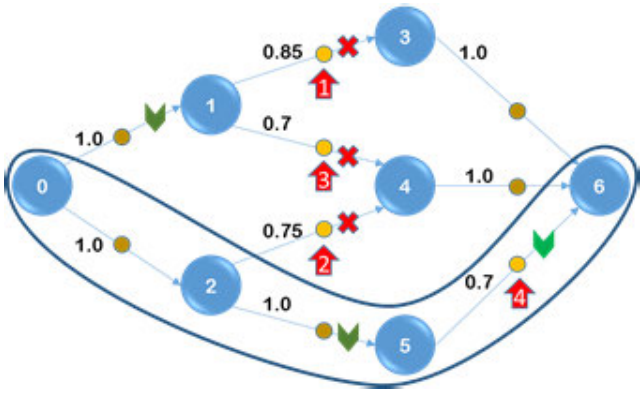


FIGURE 6. Alternative iteration 3 of second scenario (failed attack).

The difference between Algorithms 1 and 3 (the naive approaches) is in the update of the artificial survival probability (p') in order to generate a random cut-set. However, the difference between Algorithms 2 and 4 resides in the update of the survival probability (p) of an arc in the targeted cut-set. In the Algorithms 1 and 2, p' and p are updated to 1.0 for all arcs in the current generated cut-set. This procedure ensures that all arcs in the cut-set whether attacked or not will not be considered in the next iterations to warrant that all considered cut-sets will be disjoint. However, in the Algorithms 3 and 4, only the attacked arc will have its survival probability (p' and p, respectively) updated either to 1.0 or to 0.0 based on whether the arc has survived or not (respectively). In fact, when the arc survives, it will not be attempted any further when considering other cut-sets. If failed, any other cut-set containing this arc might be tempting as it has already a smaller number of attacks to carry out. Not tried arcs within a particular cut-set can still be targeted within other cut-sets (without any change in their survival probabilities) as we treat here the non-disjoint case.

1) EXAMPLE OF NON-DISJOINT CUT-SETS

Consider the network example of Figure 1. The first iteration is shown in Figure 7. The yellow dots and arrows with numbers represent the cut-set $\{2-5, 3-6, 4-6\}$ and the sequence of the attacks to be carried out. In the current iteration, the first attack on arc $\{2-5\}$ succeeds but not on arc $\{3-6\}$. The attack fails on that cut-set. The survival probabilities will be updated for arcs $\{2-5\}$ and $\{3-6\}$ to take the values of 0.0 and 1.0, respectively (Figure 8).

Figure 8 shows the second iteration with the generated cut-set $\{1-3, 2-5, 4-6\}$. Here, the current cut-set intersects the previous one at arcs $\{2-5, 4-6\}$ (as we treat the non-disjoint case). The darker color represents the previous iteration. While arc $\{2-5\}$ has already been disconnected

Algorithm 3 Naive Approach for Non-Disjoint Cut-Set

```
Let G(V, E) is the graph where V is the set of nodes and E
is the set of links
set C = ∅ (the generated cut-set)
succeed = false
n = a number of cut-sets to be targeted ({3, 5, 10})
i = 0
while succeed = false and i < n do
  generate randomly new survival probabilities for the arcs
  except those who resisted in the previous attacks
  set C = solve the min-cut problem on graph G(V, E)
  to get a cut-set based on the new randomly generated
  survival probabilities.
  if C = ∅ then
    succeed = false;
    break;
  end if
  for all link a in cut-set C do
    Attack the link a;
    if attack failed then
      succeed = false;
      p_a = 1.0;
      break;
    else
      succeed = true;
      p_a = 0.0;
    end if
  end for
  i++;
end while
if succeed = true then
  Attack succeed;
else
  Attack failed;
end if
```

at the previous iteration (and therefore will be ignored in the current attack), it is an integral link in the identified cut-set. In the current iteration, suppose that the first attempt targets arc $\{1-3\}$ and succeeds. Suppose also that the second attack on arc $\{4-6\}$ fails. Thus, the attack on the identified cut-set stops and a new cut-set will be generated at the next iteration.

The next generated cut-set is $\{1-3, 1-4, 2-4, 2-5\}$ as shown in Figure 9. This cut-set intersects with the second iteration at arcs $\{1-3, 2-5\}$. Suppose that the attack at arc $\{2-4\}$ succeeds but not at arc $\{1-4\}$. Consequently, the current iteration stops and a fourth iteration (Figure 10) is launched generating cut-set $\{0-1, 1-3, 2-4, 2-5\}$. The required attack on this cut-set is only at arc $\{0-1\}$. If the corresponding attack is successful, then the overall attack on the network will be successful; as a full cut-set is disabled. However, if the attack on $\{0-1\}$ fails (Figure 11), then the overall attack on the network fails because a full path-set from node 0 to 6 is identified to resist; namely path-set $\{0-1-4-6\}$.

Algorithm 4 Non-Disjoint Cut-Set

```
Let G(V, E) is the graph where V is the set of nodes and E
is the set of links
Stop = false; succeed = false; empty = true; l = 1
set W = ∅ (set of disabled links)
set A = ∅ (current cut-set)
while stop = false do
  set A = generate the new optimal cut from the graph
  G(V, E) using the min-cut problem as identified in
  Yaghlane et al. [1]
  if A = ∅ then
    stop = true; succeed = false;
  else
    sort A using the non-descending rule of survival
    probability P_a of its links
    for all link a in A based on the above ordering do
      attack the link;
      if attack fails then
        P_a = 1.0;
        break;
      else
        update W with the current link
        P_a = 0.0;
        if link a is the last link in A then
          stop = true; succeed = false;
        else
          if empty = false then
            if W contains a full cut-set then
              stop = true; succeed = true;
              break;
            end if
          end if
        end if
      end if
    end for
    if W = ∅ then
      empty = false;
    end if
  end if
  if stop = false then
    if l < k then
      l = l + 1;
    else
      stop = true; succeed = false;
    end if
  end if
  clear A;
end while
```

Assuming $k \geq 3$, then the most critical cut-sets identified in this example are $\{2-5, 3-6, 4-6\}$, $\{1-3, 1-4, 2-4, 2-5\}$, $\{0-1, 1-3, 2-4, 2-5\}$, respectively. Next, we attempt to identify the *k-critical cut-sets* of the example investigated above by respectively running Algorithm 2 and 4. Recall that the attack strategy accounts for a sequential decision problem

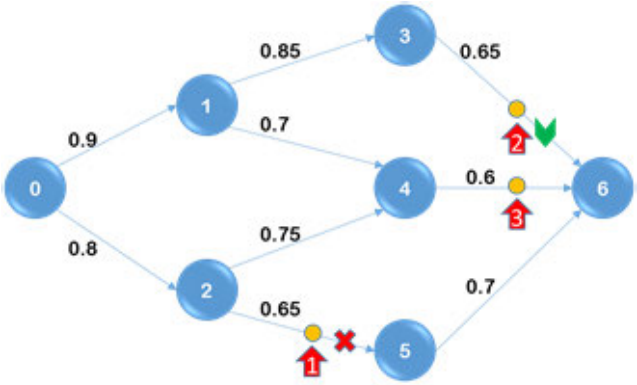


FIGURE 7. Iteration 1 of the non-disjoint case example.

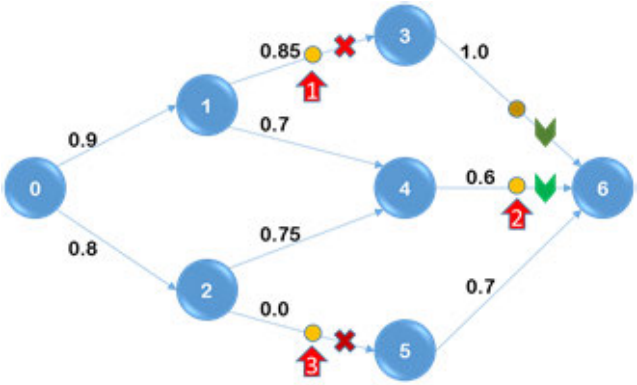


FIGURE 8. Iteration 2 of the non-disjoint case example.

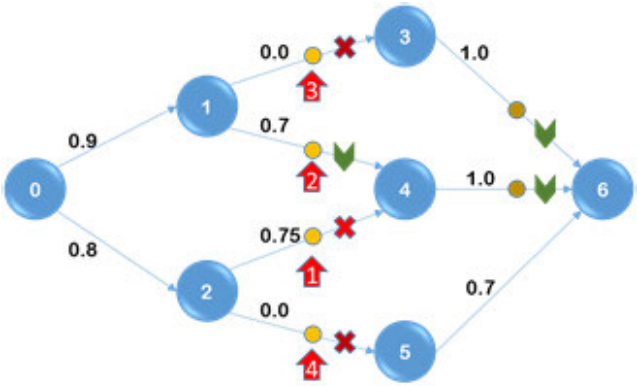


FIGURE 9. Iteration 3 of the non-disjoint case example.

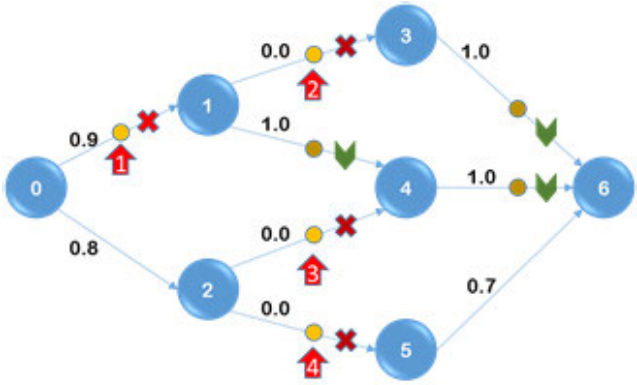


FIGURE 10. Iteration 4 of the non-disjoint case example (successful attack).

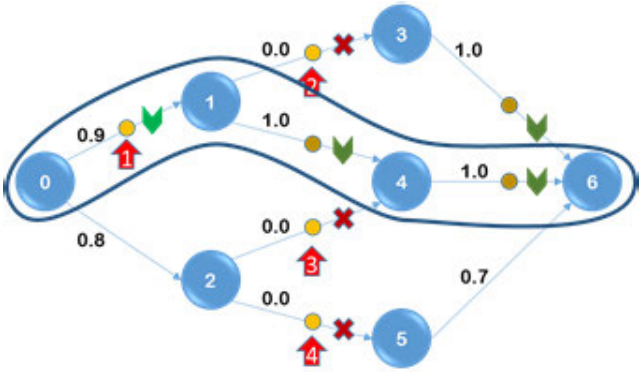


FIGURE 11. Alternative iteration 4 of the non-disjoint case example (failed attack).

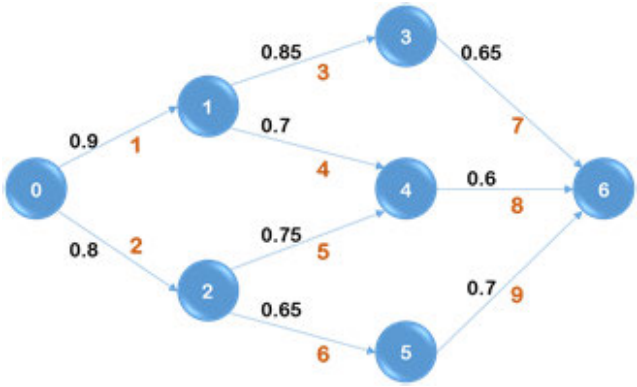


FIGURE 12. Numbered links of the network.

based on the random result of a single attack on a given link. Therefore, the algorithm uses a simulation-optimization model. Since the result of an attack on a given link follows a Bernoulli distribution (Ben Yaghlane and Azaiez [2]), we generate probability values from a Uniform distribution over [0, 1] in the simulation process.

We conduct 100 runs of both the disjoint and the non-disjoint cases of the example above. Figure 12 numbers the links of the network as specified below. Table 1 displays the results related to the number of times each cut-set has been targeted over the 100 runs of the simulation.

The simulation results state that the three most critical disjoint cut-sets are respectively $\{6, 7, 8\}$, $\{1, 2\}$, and $\{3, 4, 5, 9\}$. Note that out-of-100 simulated attacks on the network, cut-set $\{6, 7, 8\}$ has always been targeted first (i.e., 100 times). This is only natural as it is the solution of the min-cut problem. In 95 cases, the attack on $\{6, 7, 8\}$ fails

TABLE 1. Number of times each cut-set is targeted (disjoint case).

Cut-set	Count
6,7,8	100
1,2	95
3,4,5,9	93
Total	**288**

TABLE 2. Number of times each cut-set is targeted and related probabilities (non-disjoint case).

Cut-set	Count	Theoretical probability	Empirical probability	Gap
6,7,8	100	1	1	0
7,8,9	71	0.65	0.71	0.060
2,7,8	45	0.455	0.45	0.005
3,6,8	17	0.2275	0.17	0.058
1,6,8	14	0.1933	0.14	0.053
4,5,7,9	10	0.0409	0.1	0.059
3,8,9	10	0.1267	0.1	0.027
4,5,6, 7	9	0.0735	0.09	0.017
2,4,7	6	0.0551	0.06	0.005
3,4,5,6	1	0.020	0.01	0.10
1,5,6	1	0.0128	0.01	0.003
Total	**284**			

and therefore cut-set $\{1, 2\}$ is targeted next. This cut-set is the second most vulnerable cut-set in the disjoint case. The attack succeeds twice and fails 93 times; in which case the next most vulnerable disjoint cut-set; namely, $\{3, 4, 5, 9\}$ is targeted. Note that regardless of the outcome of the attack on this cut-set, there are no other disjoint cut-sets that can further be considered and therefore, the attack process stops. The simulation gives rise to 7 successful attacks in this last scenario. Observe that the exact probability that the attack process targets cut-set $\{1, 2\}$ is 0.951. Also, the exact probability that the attack process targets cut-set $\{3, 4, 5, 9\}$ is 0.93198. It follows that the gaps between the simulated probabilities of targeting any of the critical cut-sets and the exact corresponding probabilities are extremely low.

For the non-disjoint case, the problem is not so obvious and is not a sequential application of the min-cut problem (Table 2). Rather, the new iterations depend both on the identified cut-set to target and on the sequence of attacks to be carried-out on its various links.

Cut-set $\{6, 7, 8\}$, which is the solution of the min-cut problem of the entire network, is always targeted first. Therefore, over 100 runs, it is targeted 100 times. It is therefore the most critical cut-set. Given that we consider the non-disjoint case, when link 6 resists and hence the attack fails on cut-set $\{6, 7, 8\}$, there is still room to target links 7 and 8 under another cut-set which is $\{7, 8, 9\}$.

Based on the simulated results, the *3-critical cut-sets* are $\{6, 7, 8\}$, $\{7, 8, 9\}$, and $\{2, 7, 8\}$, respectively while the *4-critical cut-sets* are $\{6, 7, 8\}$, $\{7, 8, 9\}$, $\{2, 7, 8\}$, and $\{3, 6, 8\}$, respectively.

When comparing the simulated results with the theoretical ones, we observe that the gap is reasonably low and hence, the simulation optimization model can be reliable for large scale problems. One should also note that the next most vulnerable cut-set is not necessarily the next critical cut-set. For instance, cut-set $\{6, 7, 8\}$ is more vulnerable than the second critical cut-set $\{7, 8, 9\}$. This says that the results are not intuitive and hence unpredictable beforehand.

V. IMPLEMENTATION FOR LARGE SCALE NETWORKS

In order to investigate the performance of Algorithm 2 and 4, we carry out experimentations using instances generated by Gharbi *et al.* [3] for small size and by Yaghlane *et al.* [4] for large sizes. The small instances consist of networks with sizes ranging from 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. The optimized algorithms are compared to the naive approach for both disjoint and non-disjoint cases. Based on the results shown in Tables 3 and 4, it is obvious that the optimized approach outperforms the naive approach.

We also modify the instances into three parts by gradually increasing the range of values of the survival probabilities of each instance by withdrawing random numbers in the

TABLE 3. Results of comparison between the naive approach and proposed algorithm for the disjoint cut-set.

		$k = 3$				$k = 5$				$k = 10$			
		Naive		Optimal		Naive		Optimal		Naive		Optimal	
Probability range	#Nodes	Avg. kReach	Success Rate	Avg. kReach	Success Rate	Avg. kReach	Success Rate	Avg. kReach	Success Rate	Avg. kReach	Success Rate	Avg. kReach	Success Rate
ORIGINAL	10	1.4	26.3%	1.3	29.9%	1.4	26.3%	1.3	29.9%	1.4	26.3%	1.3	29.9%
ORIGINAL	15	1.6	26.5%	1.5	31.6%	1.6	26.3%	1.5	31.6%	1.6	26.3%	1.5	31.6%
ORIGINAL	20	1.6	30.7%	1.5	34.9%	1.6	31.2%	1.5	34.9%	1.6	31.2%	1.5	34.9%
ORIGINAL	25	1.8	31.4%	1.7	37.8%	1.8	31.6%	1.7	37.8%	1.8	31.6%	1.7	37.8%
ORIGINAL	30	1.9	30.4%	1.8	37.2%	1.9	30.3%	1.8	37.3%	1.9	30.3%	1.8	37.3%
ORIGINAL	50	1.8	10.2%	1.8	13.4%	1.8	10.6%	1.8	13.4%	1.8	10.6%	1.8	13.4%
ORIGINAL	100	2.0	15.0%	2.0	18.7%	2.1	14.6%	2.0	18.8%	2.1	14.6%	2.0	18.8%
ORIGINAL	150	2.5	9.2%	2.4	12.5%	2.6	9.2%	2.5	12.7%	2.6	9.2%	2.5	12.7%
ORIGINAL	200	2.6	11.5%	2.5	14.6%	2.6	11.7%	2.5	14.7%	2.6	11.7%	2.5	14.7%

TABLE 4. Results of comparison between the naive approach and proposed algorithm for the non-disjoint cut-set.

		$k=3$				$k=5$				$k=10$			
		Naive		Optimal		Naive		Optimal		Naive		Optimal	
Probability range	#Nodes	Avg. kReach	Success Rate	Avg. kReach	Success Rate	Avg. kReach	Success Rate	Avg. kReach	Success Rate	Avg. kReach	Success Rate	Avg. kReach	Success Rate
ORIGINAL	10	1.7	47.3%	2.0	51.7%	1.8	48.6%	2.3	59.5%	1.8	48.5%	2.3	60.8%
ORIGINAL	15	1.9	44.7%	2.2	51.9%	2.0	49.5%	2.6	63.2%	2.0	50.4%	2.7	67.5%
ORIGINAL	20	2.0	50.6%	2.1	55.6%	2.1	55.2%	2.5	68.0%	2.1	56.5%	2.8	73.0%
ORIGINAL	25	2.1	52.8%	2.2	58.0%	2.3	59.8%	2.7	72.0%	2.3	61.6%	3.0	77.7%
ORIGINAL	30	2.2	48.3%	2.2	55.0%	2.4	55.7%	2.8	69.5%	2.5	57.6%	3.3	78.4%
ORIGINAL	50	2.2	16.9%	2.6	23.2%	2.4	17.3%	3.9	29.3%	2.5	17.7%	6.0	35.0%
ORIGINAL	100	2.4	22.3%	2.6	29.0%	2.7	22.9%	3.9	35.2%	2.7	23.5%	6.3	42.1%
ORIGINAL	150	2.7	14.6%	2.7	20.8%	3.2	15.4%	4.3	26.0%	3.2	16.0%	7.3	36.1%
ORIGINAL	200	2.7	16.4%	2.7	24.7%	3.2	18.7%	4.2	29.5%	3.2	18.3%	7.2	37.4%

TABLE 5. Calculation results for Algorithms 2 and 4.

		Algorithm 2 (Disjoint case)						Algorithm 4 (Non-disjoint case)					
		$k=3$		$k=5$		$k=10$		$k=3$		$k=5$		$k=10$	
Probability range	#Nodes	Avg. kReach	Success Rate	Avg. kReach	Success Rate	Avg. kReach	Success Rate	Avg. kReach	Success Rate	Avg. kReach	Success Rate	Avg. kReach	Success Rate
ORIGINAL	10	1.3	29.9%	1.3	29.9%	1.3	29.9%	2.0	51.7%	2.3	59.5%	2.3	60.8%
ORIGINAL	15	1.5	31.6%	1.5	31.6%	1.5	31.6%	2.2	51.9%	2.6	63.2%	2.7	67.5%
ORIGINAL	20	1.5	34.9%	1.5	34.9%	1.5	34.9%	2.1	55.6%	2.5	68.0%	2.8	73.0%
ORIGINAL	25	1.7	37.8%	1.7	37.8%	1.7	37.8%	2.2	58.0%	2.7	72.0%	3.0	77.7%
ORIGINAL	30	1.8	37.2%	1.8	37.3%	1.8	37.3%	2.2	55.0%	2.8	69.5%	3.3	78.4%
ORIGINAL	50	1.8	13.4%	1.8	13.4%	1.8	13.4%	2.6	23.2%	3.9	29.3%	6.0	35.0%
ORIGINAL	100	2.0	18.7%	2.0	18.8%	2.0	18.8%	2.6	29.0%	3.9	35.2%	6.3	42.1%
ORIGINAL	150	2.4	12.5%	2.5	12.7%	2.5	12.7%	2.7	20.8%	4.3	26.0%	7.3	36.1%
ORIGINAL	200	2.5	14.6%	2.5	14.7%	2.5	14.7%	2.7	24.7%	4.2	29.5%	7.2	37.4%
[0.01-0.25]	10	1.2	63.9%	1.2	63.9%	1.2	63.9%	1.5	83.0%	1.6	86.9%	1.6	87.9%
[0.01-0.25]	15	1.3	65.0%	1.3	65.0%	1.3	65.0%	1.6	83.0%	1.7	88.9%	1.8	90.4%
[0.01-0.25]	20	1.3	68.9%	1.3	68.9%	1.3	68.9%	1.6	85.6%	1.7	91.0%	1.7	92.9%
[0.01-0.25]	25	1.4	71.4%	1.4	71.4%	1.4	71.4%	1.6	86.6%	1.8	93.3%	1.8	95.3%
[0.01-0.25]	30	1.4	74.0%	1.4	74.0%	1.4	74.0%	1.6	87.2%	1.7	92.7%	1.8	95.3%
[0.01-0.25]	50	1.5	46.3%	1.5	46.3%	1.5	46.3%	2.0	60.8%	2.7	68.9%	3.6	76.1%
[0.01-0.25]	100	1.5	55.1%	1.6	55.1%	1.6	55.1%	1.9	67.6%	2.5	74.9%	3.5	80.3%
[0.01-0.25]	150	1.9	49.6%	1.9	49.5%	1.9	49.5%	2.1	62.2%	2.8	69.6%	4.0	78.2%
[0.01-0.25]	200	1.9	51.2%	1.9	51.1%	1.9	51.1%	2.0	64.2%	2.7	71.9%	3.8	80.5%
[0.01-0.50]	10	1.3	35.2%	1.3	35.2%	1.3	35.2%	2.0	52.7%	2.3	60.5%	2.4	62.1%
[0.01-0.50]	15	1.5	38.0%	1.5	38.0%	1.5	38.0%	2.1	54.0%	2.5	63.6%	2.8	68.0%
[0.01-0.50]	20	1.5	41.0%	1.5	41.0%	1.5	41.0%	2.0	56.6%	2.6	66.7%	3.0	72.3%
[0.01-0.50]	25	1.7	41.6%	1.7	41.6%	1.7	41.6%	2.1	57.1%	2.6	67.9%	3.1	74.3%
[0.01-0.50]	30	1.7	41.8%	1.7	41.8%	1.7	41.8%	2.1	57.2%	2.7	67.4%	3.5	75.3%
[0.01-0.50]	50	1.7	21.3%	1.7	21.3%	1.7	21.3%	2.5	28.2%	3.6	31.8%	5.9	32.5%
[0.01-0.50]	100	1.9	22.2%	1.9	21.8%	1.9	21.8%	2.5	29.8%	3.8	33.5%	6.6	35.8%
[0.01-0.50]	150	2.3	19.8%	2.3	19.8%	2.3	19.8%	2.6	26.1%	4.0	29.4%	7.4	32.2%
[0.01-0.50]	200	2.4	20.7%	2.4	20.6%	2.4	20.6%	2.6	26.0%	3.9	31.3%	7.4	29.9%
[0.01-0.99]	10	1.4	7.6%	1.4	7.6%	1.4	7.6%	2.2	10.4%	2.6	11.8%	2.7	12.1%
[0.01-0.99]	15	1.7	9.4%	1.7	9.4%	1.7	9.4%	2.5	12.4%	3.4	14.2%	3.8	14.9%
[0.01-0.99]	20	1.7	12.8%	1.7	12.8%	1.7	12.8%	2.4	17.3%	3.3	19.7%	4.1	20.0%
[0.01-0.99]	25	1.9	9.9%	1.9	9.9%	1.9	9.9%	2.5	14.3%	3.6	16.0%	4.8	16.8%
[0.01-0.99]	30	2.1	9.5%	2.1	9.5%	2.1	9.5%	2.6	13.3%	4.0	14.2%	6.0	15.6%
[0.01-0.99]	50	1.9	0.9%	1.9	0.9%	1.9	0.9%	3.0	1.8%	4.8	1.8%	8.3	1.7%
[0.01-0.99]	100	2.1	3.2%	2.2	3.3%	2.2	3.3%	2.9	3.5%	4.6	4.5%	8.3	4.8%
[0.01-0.99]	150	2.6	2.4%	2.7	2.1%	2.7	2.1%	3.0	2.8%	4.8	3.1%	9.3	2.7%
[0.01-0.99]	200	2.7	3.3%	2.7	3.5%	2.7	3.5%	2.9	3.5%	4.8	3.7%	9.5	3.2%

intervals [0.01, 0.25], [0.01, 0.50], and [0.01, 0.99] for the first, the second, and the third part, respectively. We solve all instances through an Intel(R) Core (TM) i7 2.00 GHz Personal Computer with 32GB RAM using C++ and CPLEX version 12.6. The simulation uses 3 different values of k; namely, 3, 5, and 10. We repeat the experiment 100 times

TABLE 6. Results for Algorithm 2 on networks with larger sizes of the shortest path.

		$k=3$		$k=5$		$k=10$	
Probability range	**#Nodes**	**Avg. kReach**	**Success Rate**	**Avg. kReach**	**Success Rate**	**Avg. kReach**	**Success Rate**
ORIGINAL	10	1.3	95.1%	1.3	95.5%	1.3	95.5%
ORIGINAL	15	1.2	97.4%	1.3	98.6%	1.3	98.6%
ORIGINAL	20	1.3	94.6%	1.4	97.1%	1.4	97.4%
ORIGINAL	25	1.3	94.3%	1.4	97.5%	1.4	97.6%
ORIGINAL	30	1.6	87.1%	1.8	92.2%	1.8	92.6%
ORIGINAL	50	1.6	82.8%	2.0	86.9%	2.1	88.0%
ORIGINAL	100	2.2	48.9%	3.3	49.8%	4.0	49.0%
ORIGINAL	150	2.2	48.2%	3.3	52.2%	4.3	52.4%
ORIGINAL	200	2.5	36.1%	3.8	36.5%	4.8	36.7%
[0.01-0.25]	10	1.2	98.5%	1.2	98.8%	1.2	98.8%
[0.01-0.25]	15	1.1	99.1%	1.1	99.6%	1.1	99.6%
[0.01-0.25]	20	1.2	98.7%	1.2	99.5%	1.2	99.5%
[0.01-0.25]	25	1.2	98.6%	1.2	99.6%	1.2	99.6%
[0.01-0.25]	30	1.3	97.2%	1.3	98.8%	1.3	98.8%
[0.01-0.25]	50	1.3	95.5%	1.4	97.5%	1.4	97.7%
[0.01-0.25]	100	1.6	80.9%	2.0	81.6%	2.1	82.3%
[0.01-0.25]	150	1.6	81.5%	2.0	83.9%	2.3	82.9%
[0.01-0.25]	200	1.7	73.4%	2.3	73.2%	2.6	73.2%
[0.01-0.50]	10	1.3	94.7%	1.3	95.1%	1.3	95.2%
[0.01-0.50]	15	1.3	96.5%	1.3	97.8%	1.3	97.9%
[0.01-0.50]	20	1.4	93.0%	1.5	95.1%	1.5	95.3%
[0.01-0.50]	25	1.3	93.9%	1.4	96.2%	1.4	96.5%
[0.01-0.50]	30	1.5	87.7%	1.7	90.5%	1.7	90.8%
[0.01-0.50]	50	1.6	83.1%	1.8	86.7%	2.1	86.3%
[0.01-0.50]	100	2.1	51.8%	3.1	53.4%	3.6	53.9%
[0.01-0.50]	150	2.0	60.6%	2.7	62.2%	3.6	61.3%
[0.01-0.50]	200	2.3	40.6%	3.4	43.3%	4.3	42.0%
[0.01-0.99]	10	1.8	68.8%	1.9	69.4%	1.9	69.4%
[0.01-0.99]	15	1.5	84.2%	1.8	86.0%	1.8	86.1%
[0.01-0.99]	20	1.6	79.2%	2.0	81.8%	2.0	82.0%
[0.01-0.99]	25	1.6	79.4%	1.9	81.9%	2.1	81.9%
[0.01-0.99]	30	1.9	63.4%	2.6	65.2%	2.8	65.5%
[0.01-0.99]	50	2.0	55.7%	2.8	58.7%	3.5	58.3%
[0.01-0.99]	100	2.5	25.8%	4.0	26.6%	5.0	26.6%
[0.01-0.99]	150	2.5	26.2%	4.0	26.2%	5.5	28.2%
[0.01-0.99]	200	2.7	18.5%	4.3	18.7%	5.5	19.0%

for each instance. Table 5 above provides the calculation results for Algorithms 2 and 4.

All results are obtained instantly. This says that both algorithms are very computationally efficient. One can obviously detect that the non-disjoint case strongly dominates the disjoint case with respect to the success rate. This is only natural as it contains a larger set of feasible solutions to try. Clearly, the success rate tends to decrease, as the network gets very large in either case. For the disjoint case, Algorithm 2 stops after trying very few cut-sets (within 3 on the average) even when k takes the values 5 and 10. For the non-disjoint case however, Algorithm 4 stops after attempting almost exhaustively the *k-critical cut-sets* (particularly for large networks and relatively important survival probabilities). This says that when there are enough resources to target more than one cut-set in the non-disjoint case, then this is beneficial for the attacker. For the disjoint case, however, more attacking resources do not seem to help.

Another important observation to explore has to do with the major decline in the success rate of attacks for the non-disjoint case when the survival probability of links may take reasonably large values (exceeding perhaps 0.5 as witnessed by the last instances). This holds true even when attacks may target up to 10 cut-sets.

When considering the disjoint case however, we surprisingly note that the algorithm stops yielding most frequently unsuccessful attacks after attempting no more than four cut-sets; even for the case where up to ten cut-sets may be tried. In situations of small networks, this might be reasonable as the algorithm may fail to identify additional disjoint cut-sets. In contrast, when the size of the network gets large, certainly the algorithm is expected to identify many disjoint cut-sets to explore. In order to investigate this finding, we calculate the shortest-path for each of the generated networks. It turns out that this path possesses up to four links for all instances. We know that the size of the shortest path

problem represents an upper bound of the number of disjoint cut-sets. This says that the algorithm reaches a stopping criterion by identifying a surviving set of arcs covering all the disjoint cuts and hence stops without considering available untried cut-sets. To further verify this outcome, we generate other networks possessing larger sizes of the shortest path. The results fully support our claim. In fact, we observe that the algorithm considers more cut-sets as shown in table 6.

The defender may exploit this finding by designing resilient networks containing small paths joining the sources to the destinations if the attack may only concern disjoint cut-sets.

VI. CONCLUSION

In this paper, we extend the work by Yaghlane *et al.* [1] of interdiction networks by considering attack strategies on the most critical cut-sets of the network rather than limiting focus on a unique cut-set. We distinguish both cases of disjoint and non-disjoint cut-sets. We assume perfect information on the outcome of an attack on a targeted link before considering the next one. Consequently, the attack strategy is a sequential attack. The attacker may try any link at most once. The attack strategy stops if one full cut-set among the most critical ones fails, or if a path-set fully resists to the attack, or else the attack unsuccessfully targets all *k cut-sets*. In the last two situations, the attack on the network fails.

We develop two algorithms to determine optimally the attack strategy both for the disjoint and non-disjoint case, respectively. Both algorithms heavily rely on the solution of the min-cut problem. We compare our strategy with the naive approaches and find out that the proposed methods not only outperform by far the naive approach, but also turn out to be computationally very efficient even for large networks. The paper provides detailed illustrations at some small scale to explain the process. It also conducts some simulation-optimization to investigate the large scale of networks.

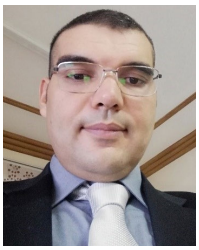
For the disjoint case, the critical cut-sets can only be limited. Therefore, both offensive and defensive resources should be devoted to only few cut-sets. Further, the defender may design the network to include some small path sets to force the attack to be devoted only to a limited number of cut-sets regardless of the size of k. For the non-disjoint case however, increasing the potential number of targeted cut-sets can improve the success rate of the attack.

Among avenues for future investigations, one may constrain the attack strategies by some resource or budget availability. This constraint may dictate the value of k. Another possible extension would be to model the problem in a game-theoretic setting. A third challenging extension would consider the absence of information on the outcomes of the attacks on successive links. In other words, the attack process would no further be dynamic and the attacker must plan it all at once.

REFERENCES

[1] 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.
[2] A. Ben 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.
[3] 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.
[4] 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* (Springer Series in Reliability Engineering), M. Ram and H. Pham, Eds. Cham, Switzerland: Springer, 2020, pp. 343–359.
[5] L. A. T. Cox, "Making telecommunications networks resilient against terrorist attacks," in *Game Theoretic Risk Analysis of Security Threats* (International Series in Operations Research & Management Science), 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] R. Ausseil, R. Gedik, A. Bednar, and M. Cowan, "Identifying sufficient deception in military logistics," *Expert Syst. Appl.*, vol. 141, Mar. 2020, Art. no. 112974.
[9] D. L. Alderson, D. Funk, and R. Gera, "Analysis of the global maritime transportation system as a layered network," *J. Transp. Secur.*, pp. 1–35, Nov. 2019.
[10] M. Tomatore, G. Maier, and A. Pattavina, "WDM network optimization by ILP based on source formulation," in *Proc. 21th Annu. Joint Conf. IEEE Comput. Commun. Societies*, Jun. 2002, pp. 1813–1821.
[11] G. Brightwell, G. Oriolo, and F. B. Shepherd, "Reserving resilient capacity in a network," *SIAM J. Discrete Math.*, vol. 14, no. 4, pp. 524–539, Jan. 2001.
[12] 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* (International Series in Operations Research & Management Science), vol. 128, V. M. Bier and M. N. Azaiez, Eds. Boston, MA, USA: Springer, 2009, pp. 199–227.
[13] 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.
[14] 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, pp. 7784–7792, Apr. 16 2019.
[15] 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.
[16] 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.
[17] 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.
[18] 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.
[19] 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.
[20] 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.
[21] 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.
[22] 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.

[23] 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.
[24] 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.
[25] 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.
[26] 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.
[27] 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.
[28] M. N. Azaiez and V. M. Bier, "Optimal resource allocation for security in reliability systems," *Eur. J. Oper. Res.*, vol. 181, no. 2, pp. 773–786, Sep. 2007.



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. He is attracted by the application of different optimization techniques to model and solve real life complicated problems from different industrial fields. His research interests include network design, vehicle routing, scheduling, and cutting problems.

UMAR S. SURYAHATMAJA (Graduate Student Member, IEEE) 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.

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

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.