Design of Network Topology in an Adversarial Environment

Assane Gueye[1], Jean C. Walrand[1], and Venkat Anantharam[1]
{agueye|wlr|ananth}@eecs.berkeley.edu

University of California at Berkeley, EECS Department, Berkeley CA 94720, USA

Abstract. We study the *strategic* interaction between a network manager whose goal is to choose (as communication infrastructure) a spanning tree of a network given as an undirected graph, and an attacker who is capable of attacking a link in the network. We model their interaction as a zero-sum game and discuss a particular set of Nash equilibria. More specifically, we show that there always exists a Nash equilibrium under which the attacker targets a *critical* set of links. A set of links is called *critical* if it has maximum vulnerability, and the *vulnerability* of a set of links is defined as the minimum fraction of links the set has in common with a spanning tree. Using simple examples, we discuss the importance of critical subsets in the design of networks that are aimed to be robust against attackers. Finally, an algorithm is provided, to compute a critical subset of a given graph.

Keywords: Network Topology, Connectivity, Graph Vulnerability, Spanning Trees, Minimum Cut Set, Game Theory, Nash Equilibrium, Linear Programming, Blocking pairs of polyhedra, Polymatroid, Network Flow Algorithm.

1 Introduction

In this work, we aim to study the *strategic* interaction between a network manager whose goal is to choose a spanning tree of the network as communication infrastructure, and an attacker who tries to disrupt the communication tree by attacking one link in the network.

The network topology is given as a connected undirected graph. The formulation of the problem extends naturally to the manager choosing a k-connected component and the attacker selecting $k' \geq k$ links to attack. For example, if $k = 2$, this problem models the situation where the manager is choosing a primary communication tree and a backup tree in the presence of an attacker who can attack more than 2 links in the network. The discussion in the present paper, however, focuses attention only on the case $k = k' = 1$.

In general, each tree has a given cost which is the loss seen by the manager when one of the edges of that tree is attacked. This cost (or a function of it) goes to the attacker. Also, it is conceivable that the attacker incurs some cost by attacking a link. The goal of the network manager is to minimize the cost of attack while the attacker is trying to maximize the net attack reward.

In a non-adversarial environment, choosing a minimum cost spanning tree (MST) of the graph would be optimal for the network manager. Algorithms for calculating the MST have been studied extensively since the work of Kruskal [15] and Prim [20].

In this paper, we will assume that every tree has equal cost. It is also assumed that the cost of attacking any given link is zero for the attacker. These assumptions will be relaxed in subsequent studies of the problem.

The communication networks community has spent a lot of effort studying the reliability/robustness of networks. The interested reader is referred to [21] and [12] and the references therein. Robustness has mostly been considered against *non-strategic* phenomena (e.g. random failures). However, network disruption can also be due to malicious attackers. The nature of the attack can be varied. In an *availability* attack, the attacker might be launching a denial of service (DoS) attack on some node/link, or simply jam a communication channel. In a *confidentiality* attack, the attacker could be choosing a link and observe/analyze the traffic that it carries. An *integrity* attack could also be launched, where the attacker will try to modify the traffic (or generate traffic) for a target link/node.

These problems have received a lot of attention specially in the area of mobile and *ad-hoc* networks [5], [1] and mostly in a non-strategic framework. In an environment where the adversary is cognitive, most of the results found in the literature do not apply any more. For example, in the graph connectivity problem considered here, when the attacker strategically chooses the edge to attack, it is no longer obvious how the network manager should choose a spanning tree. For example, if the network manager were to always choose a specific MST, the attacker could compute this MST and attack one of its links to disconnect the network.

To understand how the network manager should choose a spanning component as well as how an attacker could break the communication, we model their interaction as a game where:

- the manager's strategy set is the set of spanning trees of the graph;
- the attacker's strategy set is the set of links;
- the goal of the manager is to minimize the average cost of getting attacked while the goal of the attacker is to maximize that cost.

We assume that the network topology is known to both players. All trees have the same cost, and there is no cost of attack. We would like to understand the structure of the (or at least some) set of Nash equilibria of this one-shot, zero-sum game.

The organization of this paper is as follows. In the next section, we present the model of the game considered in this paper. The notion of critical subset is discussed in subsection 2.1, followed by illustrative examples in subsection 2.2. The main result of the paper (the critical subset attack theorem) is presented in subsection 2.3 and a brief discussion of this result is provided in subsection 2.4. A proof of the theorem is provided in section 3. This proof requires the notions of *blocking pairs of polyhedra* and a characterization of the *spanning tree polyhedra*. A tutorial presenting those notions is given in appendix A. Section 4 presents an algorithm to compute a critical subset of a graph. The algorithm is essentially based on the theory of *polymatroids* which we discuss in section B. Concluding remarks and directions for future work are given in section 5.

2 The Game

The network topology is given by a connected undirected graph $G = (\mathcal{V}, \mathcal{E})$ with $|\mathcal{E}| = m$ links and $|\mathcal{V}| = n$ nodes. Let $\mathcal{T}$ be the set of spanning trees, and let N denote $|\mathcal{T}|$.

We consider the 2-player, zero-sum game where player 1 (the network manager) chooses a spanning tree according to some distribution on $\mathcal{T}$ to minimize the probability (which, for equal unit cost of trees, corresponds to the cost) that the spanning tree is disrupted. Player 2 (the attacker) chooses a link to attack according to some distribution on $\mathcal{E}$ to maximize this probability. A tree gets "disconnected" if the attacked link belongs to it. We aim to analyze the set of Nash equilibria of this game.

More precisely, let $\mathcal{A} := \{\alpha \in \Re_+^N \mid \sum_{T\in\mathcal{T}} \alpha_T = 1\}$ be the set of mixed strategies of the network manager and $\mathcal{B} := \{\beta \in \Re_+^m \mid \sum_{e\in\mathcal{E}} \beta_e = 1\}$ the set of mixed strategies of the attacker.

The manager wants to minimize and the attacker wants to maximize $C(\alpha, \beta)$ where

$$C(\alpha, \beta) = \sum_{e\in E} \sum_{T\in\mathcal{T}} \alpha_T \beta_e 1\{e \in T\}. \tag{1}$$

2.1 Critical Set

We first characterize some subsets of edges as being most vulnerable to attack.

Definition 1 (Critical Set). *For any nonempty subset of edges $E \subseteq \mathcal{E}$, define*

$$\mathcal{M}(E) := \min_{T\in\mathcal{T}} \mid T \cap E \mid \text{ and } \vartheta(E) := \frac{\mathcal{M}(E)}{|E|}. \tag{2}$$

We call $\vartheta(E)$ the vulnerability *of E. It is the minimum fraction of links the set E has in common with a spanning tree. A nonempty subset E of edges is said to be* critical *if*

$$\vartheta(E) = max_{E'\subseteq\mathcal{E}} \{\vartheta(E')\}. \tag{3}$$

In other words, a subset of links is critical *if it has maximum* vulnerability. *The vulnerability of a graph G is defined as the vulnerability of its critical subset(s), and is denoted $\vartheta(G)$.*

For each $E \subseteq \mathcal{E}$ we define $\mathcal{T}_E \subseteq \mathcal{T}$ by:

$$T \in \mathcal{T}_E \Longleftrightarrow \mid T \cap E \mid = \mathcal{M}(E). \tag{4}$$

*We will call any $T \in \mathcal{T}_E$ an E-*minimal spanning tree.

Our notion of graph vulnerability is related to a notion which has previously been proposed in the graph theory literature (see [13], [8], [3]). However, to the authors's knowledge it seems to have not received a lot of attention. We briefly discuss those references in section 2.4.

2.2 Examples of Critical Sets

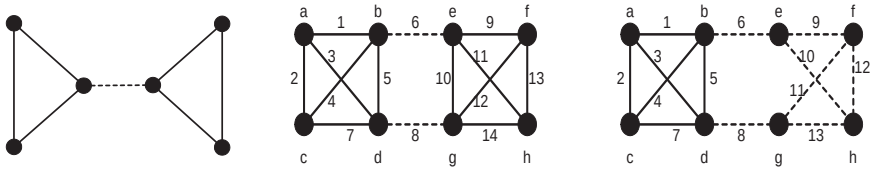
Let us illustrate the definitions with some examples, shown in Fig.1. For the network in Fig.1(a), all spanning trees must go through the middle link (called a *bridge*), so that $\vartheta(E) = 1$ if E is the set with only that link. That set is critical and the attacker can attack it and achieves the maximum cost of one.

In general, an edge that must be part of every spanning tree is called a bridge. Also, it is not difficult to verify that the vulnerability of a subset E is equal to the maximum value of 1 if and only if E is only composed of bridges.

The graph in Fig.1(b) contains 8 nodes and 14 links. It has one minimum cut set composed of the links 6 and 8. If $E = \{6, 8\}$, then any spanning tree contains at least one link in E. Thus, $|T \cap E| \geq 1$ for any tree T. Furthermore, there exists T such that $T \cap E = \{6\}$. Thus, $\mathcal{M}(E) = 1$, giving a vulnerability of $\vartheta(E) = 1/2$. This is the maximum vulnerability of this graph (verification is left as an exercise for the interested reader), which implies that $E = \{6, 8\}$ is a critical subset. If we consider the set of all links $E = \mathcal{E}$, then $|T \cap E| = n - 1 = 7$ for any tree T because any spanning tree contains $n - 1$ links. This set is also critical because $\vartheta(E) = \frac{7}{14} = 1/2$.

In general, there might be many critical subsets for a given graph. For instance, in Fig.1(b), as shown above, $E = \{1, 2, 3, 4, 5, 6, 7, 8\}$ is another critical subset. If $E = \{1, 2, 4\}$, choosing $T = \{3, 6, 7, 8, 9, 13, 14\}$ gives $T \cap E = \emptyset$. Hence, $\mathcal{M}(E) = 0$.

The minimum cut set of a graph is not always critical. In Fig.1(c) if $E = \{6, 8\}$ then $\vartheta(E) = 1/2$. However choosing $E = \{6, 8, 9, 10, 11, 12, 13\}$ gives $\vartheta(E) = 4/7 > 1/2$. One can show that $E = \{6, 8, 9, 10, 11, 12, 13\}$ is critical but $E = \{6, 8\}$ is not.



(a) Graph with bridge

(b) Network where minimum cut set is critical

(c) Minimum cut set is not critical

Fig. 1. Illustrative network examples. Example 1(a) is a network that contains a bridge (dotted link). A bridge is always a critical set. The network in 1(b) is an example of graph where the minimum cut set (dashed links) corresponds to a critical subset. Example 1(c) shows a graph where the minimum cut set is not critical.

2.3 Critical Subset Attack

Next we give the structure of one particular class of Nash equilibria (NE) of the game defined above. First, we let

$$\alpha(e) := \sum_{T \in \mathcal{T}} \alpha_T 1\{e \in T\}, \text{ for } e \in \mathcal{E}. \tag{5}$$

Theorem 1 (Critical Subset Attack Theorem). *For each critical subset of edges, E, there exists a NE under which the attacker* uniformly and exclusively *targets the edges of the critical subset E and the network manager chooses only trees inside the set of E-minimal spanning trees. Specifically, the strategy of the attacker is*

$$\beta_e = \frac{\mathbf{1}_{e \in E}}{|E|}, \tag{6}$$

and the strategy of the manager is $\alpha \in \mathcal{A}$ such that

$$\begin{cases} \alpha_T \geq 0 \text{ if } T \in \mathcal{T}_E \\ \alpha_T = 0 \text{ otherwise} \end{cases} \tag{7}$$

$$\alpha(e) := \sum_{T \in \mathcal{T}} \alpha_T 1\{e \in T\} = \vartheta(E), \forall e \in E \tag{8}$$

$$\alpha(f) \leq \vartheta(E), \forall f \notin E. \tag{9}$$

The corresponding optimal payoff is equal to $\vartheta(E)$.

A proof of the theorem is provided in section 3.

2.4 Comments

A certain number of remarks are to be made about the previous result.

- The equilibrium strategy for the network α is such that each element of its support ($\mathcal{T}_E$) meets the critical set in the minimum number of links. Furthermore, the sum ($\alpha(e)$) of the probability assigned to the trees crossing each link $e \in E$ is the same for all links in the critical subset. This sum is equal to the vulnerability of the subset E.
- As we have seen in the examples of the previous section, a graph has in general many critical subsets. As a consequence, there might be many NE (each with a different α and β). There might even exist other Nash equilibria than the ones isolated above. However, because the game is zero-sum, all equilibria have the same payoff [24]. As a consequence, it is reasonable to use the terminology "vulnerability of a graph" for $\vartheta(G)$, defined earlier as the vulnerability of any critical subset of its links.
- Theorem 1 implies that every critical subset supports some Nash equilibrium (for instance the critical subset attack equilibrium).
- Knowing the critical subsets (the weakest points of the network) is important for the network manager. The example in Fig.2 is an illustration. Consider the network in Fig.2(a) whose vulnerability is equal to $\frac{3}{4}$. In all these figures, the critical subset is represented by the dashed edges. Suppose that the network manager has an extra link to add to this network and would like to know the optimal way to add this link. If the additional link is put in the position as in Fig.2(b), then the vulnerability of the graph becomes $\frac{3}{5} < \frac{3}{4}$ (the graph is always less vulnerable with an additional link). If instead the link is added as in Fig.2(c), the vulnerability of the graph is $\frac{2}{3} > \frac{3}{5}$ leading to a less robust network.

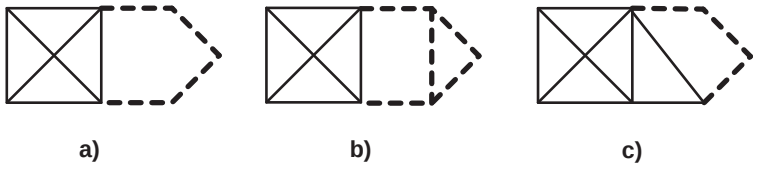


Fig. 2. Critical subset and topology design. Graphs (b) and (c) are two different ways of adding a link to graph (a) which have a vulnerability of 3/4. If it is added as in (b), then the vulnerability is $\frac{3}{5}$. If it is done as in (c), the vulnerability is $\frac{2}{3} > \frac{3}{5}$, which is leads to a less robust network.

– As was mentioned in section 2, the notion of graph vulnerability considered in this paper has been previously (with some differences) defined in a related but slightly different context. In [13], Gusfield discussed the consequences of Tutte [23] and Nash-Williams' theorem [19] and was particularly interested in the maximum number (M) of edge-disjoint spanning trees of a graph G. Two spanning trees of G are called disjoint if they have no edge in common.
Gusfield showed that

$$M = \min_{E \subseteq \mathcal{E}} \lfloor \frac{|E|}{Q(G_{\bar{E}}) - 1} \rfloor \, , \tag{10}$$

where $G_{\bar{E}}$ is the graph resulting from deleting the edges in E from G, and $Q(G_{\bar{E}})$ is the number of connected components in $G_{\bar{E}}$. $\bar{E}$ denotes the complement of E in $\mathcal{E}$.
The quantity $\sigma(G) = \min_{E \subseteq \mathcal{E}} \left(\frac{|E|}{Q(G_{\bar{E}})-1} \right)$ was then used as a measure of the *invulnerability* of the graph, i.e. the smaller this is the more vulnerable the graph is, in the sense of Gusfield. In that paper, any minimizing set for this quantity was interpreted as *a set of edges whose removal from G maximizes the number of additional components created, per edge removed*. The main question that was asked in that paper was whether there exists a polynomial time algorithm to compute $\sigma(G)$.
Cunningham provided such an algorithm in [8]. Considering $\sigma(G)$ as the *strength* of G, he defined (in a non-game theoretic setting) an *optimal attack* problem as well as a *network reinforcement* problem. The optimal attack problem consists of computing the strength of G and determining a minimizing set. Cunningham considered edge-weighted graphs, with edge j having strength s_j; the strength of the graph is defined as $\sigma(G) = \min_{E \subseteq \mathcal{E}} \left(\frac{\sum_{j \in E} s_j}{Q(G_{\bar{E}})-1} \right)$, which corresponds to the invulnerability defined by Gusfield when $s_j = 1$ for all $j \in \mathcal{E}$. The network reinforcement problem of [8] is related to minimizing the cost of increasing the strengths of individual edges in order to achieve a target strength for the graph. For details, see [8].
Using *polymatroid theory* and *network flow analysis*, Cunningham provided polynomial time algorithmic solutions to both problems. In section 4, we discuss this algorithm in the context of the present paper.
A more recent paper by Catlin *et al.* [3] generalizes Gusfield's notion of invulnerability by imposing bounds on the number of connected components, $Q(G_{\bar{E}})$.

In the present paper, the critical subsets, in our sense, have been found to correspond to Nash equilibria of a zero-sum game. It is to be noticed that our definition of vulnerability verifies $\vartheta(G) = \sigma(G)^{-1}$. To see that, one needs to show that,

Lemma 1. *For any $E \subseteq \mathcal{E}$,*

$$\mathcal{M}(E) = Q(G_{\bar{E}}) - 1. \tag{11}$$

Proof. The ideas in the proof is as follows. Consider the different connected components of the graph when the edges in E are removed. Any spanning tree of the original graph has to connected those components, and this connection is done by only using edges in E. Since there are $Q(G_{\bar{E}})$ connected components, one needs *exactly* $Q(G_{\bar{E}})$-1 to connect them in a cycle-free way. A complete proof is given in [11].

It is interesting to note that, despite the fact that this metric ($\sigma(G)$) is more refined than the *edge connectivity* (i.e. size of minimum cut set), it has largely not been used in the graph theory community. One reason suggested by Gusfield is the complexity of its computation. As was stated earlier, Cunningham [8] has subsequently provided a polynomial time algorithm to compute $\sigma(G)$ as well as a minimizing subset.
Our result shows that, in a environment where the adversary is cognitive, $\vartheta(G)$ is indeed an appropriate metric of graph vulnerability.
From the discussion above, we can, by using Cunningham's algorithm, compute a critical set of a given graph. We present the details of the algorithm in section 4.

3 Proof of the critical subset attack theorem

In this section we present a proof of the critical subset attack theorem. The proof is done in two parts. In the first part (section 3.1), we show that the strategy pair given in Theorem 1 forms a pair of best responses to each other. In the second part of the proof (section 3.2), we show that for any critical subset, there indeed exists a probability distribution α that satisfies conditions (7-9).

3.1 Best Responses

Let (α, β) be a strategy pair. Observe that the attack cost is given by

$$C(\alpha, \beta) = \sum_{e \in E} \sum_{T \in \mathcal{T}} \alpha_T \beta_e 1\{e \in T\} = \sum_{e \in E} \beta_e \alpha(e). \tag{12}$$

Let $E \subseteq \mathcal{E}$ be a critical subset and assume that α satisfies the conditions (7-9). Then, any distribution β concentrated on E achieves the cost $\vartheta(E)$. This is the maximum possible cost achievable for the attacker. To see this, observe that for any β,

$$C(\alpha, \beta) = \sum_{f} \beta_f \alpha(f) \leq \sum_{f} \beta_f \vartheta(E) \leq \vartheta(E). \tag{13}$$

Now assume that β is uniform on a critical set E. Then the distribution α achieves the cost $\vartheta(E)$. This is the minimum possible cost. To see this, note that, for any α,

$$C(\alpha,\beta)=\frac{1}{|E|}\sum_{e\in E}\sum_{T}\alpha_T 1\{e\in T\}=\frac{1}{|E|}\sum_{T}\alpha_T|T\cap E|\geq\frac{1}{|E|}\sum_{T}\alpha_T\mathcal{M}(E)=\vartheta(E), \tag{14}$$

where the next-to-last inequality uses the fact that $|T\cap E|\geq\mathcal{M}(E)$ for all T.

3.2 Existence of the Equilibrium Distribution

The claim is that one can find $\alpha\in\mathcal{A}$ that satisfies (7-9). To prove that fact, we formulate an optimization problem and we show in Theorem 2 that the solution is the desired α.

Let A be the edge-tree incidence matrix with $A(f,T)=1\{f\in T\}$ for $f\in\mathcal{E}$ and $T\in\mathcal{T}$. The *spanning tree polyhedron* $\mathcal{P}$ is defined as the vector sum of the convex hull of the columns of A and the nonnegative orthant $\mathcal{R}^m_+$ (see appendix A.2, and references [10], [7]). It is known [4] that

$$\mathcal{P}=\{\mathbf{x}\in R^m_+\mid\mathbf{x}(\mathcal{E}(P))\geq|P|-1,\ \text{for all feasible partitions } P=\{V_1,V_2,\ldots,V_{|P|}\}\}. \tag{15}$$

In (15), $P=\{V_1,V_2,\ldots,V_{|P|}\}$ is *feasible* if each V_i induces a connected subgraph $G(V_i)$ of G (see appendix A). $|P|$ is the size of the partition. The notation $\mathbf{x}(\mathcal{E}(P))$ is defined as $\mathbf{x}(\mathcal{E}(P)):=\sum_{i\in\mathcal{E}(P)}x_i$, where $\mathcal{E}(P)$ is the set of all edges of G having endpoints in different members of the partition.

Theorem 2. *Let E be a critical subset of edges. Let $\mathbf{x}^*\in R^N$ be the solution of the following problem:*

$$\begin{aligned}&\textit{Maximize } \mathbf{1}'\mathbf{x}\\ &\textit{subject to } A\mathbf{x}\leq\vartheta(E)\mathbf{1},\mathbf{x}\geq\mathbf{0}.\end{aligned} \tag{16}$$

Then

a) $\mathbf{1}'\mathbf{x}^*\leq 1$*;*

b) $\mathbf{1}'\mathbf{x}^*\geq 1$*;*

c) $A\mathbf{x}^*(e)=\vartheta(E),\forall e\in E$.

As a consequence, $\alpha=\mathbf{x}^*$ *satisfies (7)-(9).*

Proof. **a)** Let $\mathbf{w}(f)=1\{f\in E\}$ for $f\in\mathcal{E}$. Note that $A'\mathbf{w}\geq\mathcal{M}(E)\mathbf{1}$, by definition of $\mathcal{M}(E)$. Hence, for all $\mathbf{x}\in R^N$ satisfying (16),

$$\mathbf{1}'\mathbf{x}\leq\mathcal{M}(E)^{-1}\mathbf{w}'A\mathbf{x}\leq\mathcal{M}(E)^{-1}\mathbf{w}'\vartheta(E)\mathbf{1}=1, \tag{17}$$

since $\mathbf{w}'\mathbf{1}=|E|$.

b) The dual of the program (see [2]) is

$$\begin{aligned}&\text{Minimize } \vartheta(E)\mathbf{y}'\mathbf{1}\\ &\text{subject to } A'\mathbf{y}\geq\mathbf{1},\mathbf{y}\geq\mathbf{0}.\end{aligned}$$

The constraints of the dual program define the following polyhedron

$$\hat{\mathcal{P}} = \left\{ \mathbf{y} \in R_+^m, \text{ s.t } A'\mathbf{y} \geq 1 \right\} . \tag{18}$$

By results of linear programming (strong duality [2]), the value of the dual program is identical to that of the original program. Now we would like to show that the value of the dual program is at least 1, i.e. $\vartheta(E)\mathbf{y}'\mathbf{1} \geq 1$ for all $\mathbf{y} \in \hat{\mathcal{P}}$.

An equivalent way of saying this is that $\boldsymbol{\gamma} := \vartheta(E)\mathbf{1}$ belongs to the set

$$b(\hat{\mathcal{P}}) = \left\{ \mathbf{z} \in R_+^m, \text{ s.t } \mathbf{z} \cdot \hat{\mathcal{P}} \geq 1 \right\} , \tag{19}$$

where $\mathbf{z} \cdot \hat{\mathcal{P}}$ defines the inner product of $\mathbf{z}$ with any vector in $\hat{\mathcal{P}}$.

According to standard terminology (see Fulkerson [10, pg. 171] or Chopra [4]), this set is called the *blocker* of the polyhedron $\hat{\mathcal{P}}$. Since A is defined as (the transpose of) the incidence matrix of the spanning trees, $\hat{\mathcal{P}}$ in (18) is also the blocker of the spanning tree polyhedron $\mathcal{P}$ [4]. From the theory of blocking pairs of polyhedra (see appendix A), we have: if $\mathcal{B}$ is a polyhedron and $b(\mathcal{B})$ its blocker, then $b\left(b(\mathcal{B})\right) = \mathcal{B}$. ($\mathcal{B}$ and $b(\mathcal{B})$ are said to form a blocking pair of polyhedra.)

Thus, since $\hat{\mathcal{P}}$ is the blocker of $\mathcal{P}$, $b(\hat{\mathcal{P}}) = \mathcal{P}$. Now, $\mathbf{y}'\boldsymbol{\gamma} \geq 1$ for all $\mathbf{y} \in \hat{\mathcal{P}}$ is equivalent to saying that $\boldsymbol{\gamma} \in b(\hat{\mathcal{P}}) = \mathcal{P}$. From (15), this means

$$\boldsymbol{\gamma}\left(\mathcal{E}(P)\right) \geq |P| - 1 \tag{20}$$

for all feasible partitions P, $\mathcal{E}(P) \subseteq \mathcal{E}$.
Now assume that this is not the case, i.e $\boldsymbol{\gamma}\left(\mathcal{E}(P)\right) < |P| - 1$ for some P. Then

$$\sum_{i \in \mathcal{E}(P)} \boldsymbol{\gamma}_i = \frac{\mathcal{M}(E)}{|E|} \sum_{i \in \mathcal{E}(P)} 1 = \frac{\mathcal{M}(E)}{|E|} |\mathcal{E}(P)| < |P| - 1, \tag{21}$$

which implies that

$$\frac{\mathcal{M}(E)}{|E|} < \frac{P-1}{|\mathcal{E}(P)|}. \tag{22}$$

This means that $\mathcal{E}(P)$ is more vulnerable than E. Indeed, $|P| - 1$ is the minimum number of edges in common with $\mathcal{E}(P)$ that a spanning tree of G has.

Now, since the value of the dual program is at least 1, and the value of the primal program is at most 1, we can conclude that the value of the primal problem is one.

c) Note that, $\mathbf{1}'\mathbf{x}^* = 1$ and (17) imply

$$\mathbf{w}' A\mathbf{x}^* = \mathcal{M}(E). \tag{23}$$

By (16), we have $A\mathbf{x}^*(e) \leq \vartheta(E)$ for all $e \in \mathcal{E}$. Thus $\mathbf{w}' A\mathbf{x}^* \leq \mathcal{M}(E)$, but then by (23) we also have $A\mathbf{x}^*(e) = \vartheta(E)$ for all $e \in E$. Finally, we see that if $x^*(T) > 0$ for any $T \notin \mathcal{T}_E$, we would have $\mathbf{w}' A\mathbf{x}^* > \mathcal{M}(E)$, contradicting (23).

4 An Algorithm to Compute a Critical Subset

In this section, we present an algorithm to compute the vulnerability of a graph $\vartheta(G) = max_{E \subseteq \mathcal{E}} \{\vartheta(E)\}$, as well as a maximizing subset (i.e. a critical subset). The algorithm was first presented by Cunningham in [8]. For the sake of completeness, we discuss it here, and adapt it to the context of this paper. A summary of the steps of the algorithm is presented in section 4.1, and its details are discussed in section 4.2.
The discussion of section 4.2 needs the notions of *matroid* and *polymatroid*, which we present in appendix B.1.

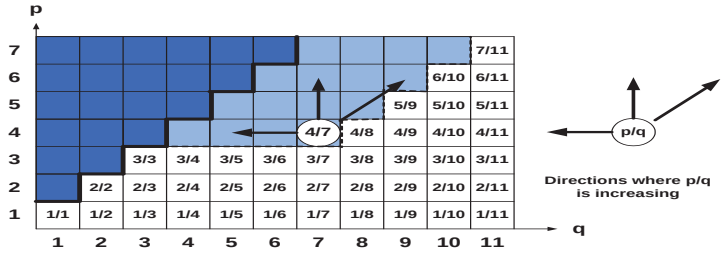


Fig. 3. An illustration of the 2-dimensional search algorithm to find the vulnerability of a graph. The dark (blue) region consists of p and q verifying $p/q > 1$. Since $\vartheta(G) \leq 1$, those values do not need to be tested. The light (blue) consist of values of p and q such that $\frac{p}{q} > \frac{p_0}{q_0}$ (here $\frac{p_0}{q_0} = \frac{4}{7}$). If $\vartheta(G) < \frac{p_0}{q_0}$, then, those values can be discarded from the test. The remaining (uncolored) values are the only ones that need to be tested.

4.1 Summary of the Algorithm

Observing that $\frac{\mathcal{M}(E)}{|E|}$ takes values in a finite set ($0 \leq \mathcal{M}(E) \leq |\mathcal{V} - 1|$ and $1 \leq |E| \leq |\mathcal{E}|$), we propose a *binary search* algorithm to find $\vartheta(G)$ *(BinarySearch2D)*. This requires an *oracle* to perform the test $\vartheta(G) \leq \frac{p}{q}$ for some p and q, $1 \leq p \leq |\mathcal{V} - 1|$ and $1 \leq q \leq |\mathcal{E}|$.

We show that such an oracle is equivalent to one that solves a minimization problem on the subsets of $\mathcal{E}$.
Solving this optimization problem will be further shown to be the same as finding a P-*basis* of some properly defined polymatroid P (those notions are presented in appendix B.1). A simple, greedy algorithm *(CunninghamMin)* will be used to find such P-basis.

The greedy algorithm will successively visit the edges of the graph, and for each edge, solve a minimization problem that is related to the first one. This last minimization can be solved by running a network flow algorithm on an appropriately defined graph.

4.2 Details of the Algorithm

In the process of computing the quantity $\vartheta(G) = \max_{E\subseteq\mathcal{E}} \left(\frac{\mathcal{M}(E)}{|E|}\right)$, we first notice that, if there exists an oracle to test whether $\vartheta(G) \leq \frac{p}{q}$, then one will be able to compute $\vartheta(G)$ using an efficient search algorithm. Indeed, the values of p and q for which one needs to test are in a finite range. We illustrate this 2-dimensional search in Figure 3. Details of the algorithm will be discussed later.

Related to the test $\vartheta(G) \leq \frac{p}{q}$, we define the following problem (that Cunningham calls the *optimal attack* problem)

$$\text{minimize} \left(\frac{p}{q}|E| - \mathcal{M}(E)\right) \ , \tag{24}$$

where the minimization is carried out over all subsets of edges $E \subseteq \mathcal{E}$, and p and q are given numbers. The next lemma shows an equivalence between testing $\vartheta(G) \leq \frac{p}{q}$ and verifying whether the minimum in (24) is greater than or equal to zero.

Lemma 2. *For fixed values of p and q (define $\rho := \frac{p}{q}$), we have*

$$\vartheta(G) \leq \rho \ \ \Leftrightarrow \ \ 0 \leq \min_{E\subseteq\mathcal{E}} \left(\rho|E| - \mathcal{M}(E)\right) \ . \tag{25}$$

Proof. The proof of the lemma is as follows:

$$\begin{aligned}\vartheta(G) \leq \rho \Leftrightarrow \max_{E\subseteq\mathcal{E}} \left(\frac{\mathcal{M}(E)}{|E|}\right) \leq \rho \Leftrightarrow \frac{\mathcal{M}(E)}{|E|} \leq \rho, \ \ \forall \ E \subseteq \mathcal{E} \\ \Leftrightarrow 0 \leq \rho|E| - \mathcal{M}(E), \ \ \forall \ E \subseteq \mathcal{E} \\ \Leftrightarrow 0 \leq \min_{E\subseteq\mathcal{E}} \left(\rho|E| - \mathcal{M}(E)\right).\end{aligned}$$

Now, we show how, by Lemma 1, we can rewrite the minimization using a function on subsets of the edges of the graph G. More precisely, we define $f(\cdot)$ such that $f(E) = |\mathcal{V}| - Q(G_E)$, where $Q(G_E)$ is the number connected components of the subgraph $G_E = (\mathcal{V}, E)$, that only contains the edges in E (in the terminology of Appendix B, $f(\cdot)$ is the rank function of the graphic matroid associated with G).
By definition of $f(\cdot)$, $f(\bar{E}) = |\mathcal{V}| - Q(G_{\bar{E}})$, where $\bar{E}$, denotes the complement of set the E. Using Lemma 1, we can write $\mathcal{M}(E) = |\mathcal{V}| - 1 - f(\bar{E})$.

The minimization in (24) can now be written as

$$\text{minimize}_{E\subseteq\mathcal{E}} \left(\left(\rho|E| + f(\bar{E})\right) - \left(|\mathcal{V}| - 1\right)\right). \tag{26}$$

Thus, we can conclude that testing whether $\vartheta(G) \leq \rho$ is equivalent to testing

$$|\mathcal{V}| - 1 \leq \min_{E\subseteq\mathcal{E}} \left(\rho|E| + f(\bar{E})\right) \ . \tag{27}$$

Since $f(\cdot)$ is the rank function of a matroid, it satisfies the hypothesis of Theorem 3 of appendix B. Using that theorem, the minimum in the RHS is achieved at an $P(f)$-basis of the vector $\rho\mathbf{1} \in R_+^{|\mathcal{E}|}$, where $P(f)$ is the *polymatroid* associated with $f(\cdot)$ (see

appendix B.1). Thus, any oracle that computes a $P(f)$-basis for the polymatroid will suffice to compute a minimizer of (27) (and the minimum). Using such an oracle, we can now implement the following search algorithm that computes $\vartheta(G)$, as well as a critical set which is the minimizer provided by the algorithm when it terminates.

The search algorithm (summarized in Table 1) keeps a set of candidate values Pr for p, and for each $p \in Pr$, a range $\{q_{min}(p), \ldots, |\mathcal{E}|\}$ of values of q for which the test in (27) will be carried out.

At each iteration, for some $p \in Pr$ and $q \in \{q_{min}(p), \ldots, |\mathcal{E}|\}$, a call is made to the oracle; then Pr and q_{min} are updated. Pr is defined as $P_r = \{1, \ldots, |\mathcal{V}| - 1\}$ at initial time, and maintained as follows.
Since the vulnerability of a graph is always less than or equal to 1, the values of p and q for which $p/q > 1$ can be ignored from the test. These values correspond to the "dark" (blue) region above the first diagonal of Figure 3 (if the graph does not contain a bridge, one can eliminate the values in the first diagonal as well). This implies that for each p, there is a minimum value for q, call it $q_{min}(p)$; i.e. when p is considered in a given iteration, only values of q in the range $\{q_{min}(p), \ldots, |\mathcal{E}|\}$ need to be used for testing.

Also, if $\vartheta \leq \frac{p_0}{q_0}$ for some fixed (p_0, q_0), then $\vartheta \leq \frac{p}{q}$ for all $\frac{p}{q} > \frac{p_0}{q_0}$. As such, those values can be safely discarded from the set of values to be tested. In Figure 3, that set is represented by the "light" (blue) region for $p_0 = 4$ and $q_0 = 7$. It is the set of numbers that are located in the 135 degrees range, from the first diagonal to the horizontal axis (traveling counterclockwise). After removing this set, the values of $q_{min}(p)$ need to be updated for all $p \geq p_0$. If q_0 is the first value of q (starting from $|\mathcal{E}|$ going down) for which the test succeeds (i.e. $\vartheta(G) \leq \frac{p_0}{q_0}$), then $q_{min}(p_0) = q_0 + 1$, and for $p \in \{p_0 + 1, \ldots, \mathcal{V} - 1\}$, $q_{min}(p)$ is obtained by adding 1 to $q_{min}(p - 1)$. If $q_{min}(p) > |\mathcal{E}|$, then p can be removed from the set Pr of candidate values for p. If for some p, the test fails for all $q \in \{q_{min}, \ldots, |\mathcal{E}|\}$, then p can also be discarded from Pr. The algorithm stops when the test succeeds and $|Pr| = 1$.

For each value of p, the algorithm makes less than $|\mathcal{E}|$ calls to the oracle, and there are at most $|\mathcal{V}|$ possible values for p (this is the worst case). Thus, computing a critical subset will take a polynomial time provided that Cunningham's algorithm is polynomial. We will see that it is indeed the case.

5 Conclusion and future work

The paper studies a *1-connection* game where a network manager is choosing a spanning tree of a graph as communication infrastructure, and an attacker is trying to disrupt the communication tree by attacking one link of the graph. We discovered that for every critical subset of edges (a subset of edges of maximum vulnerability) there is a Nash equilibrium such that the attacker attacks uniformly at random over this subset of edges. The vulnerability of a subset of links E is defined as the minimum fraction of links it has in common with any spanning tree. More precisely, we show that there always exists a NE under which an attacker targets uniformly and exclusively a critical subset of links. The network manager chooses spanning trees that cross the critical set in the minimum number of edges and such that the sum of the probabilities of all trees going

Table 1. Left: Pseudocode of the *BinarySearch2D* algorithm to compute the vulnerability $\vartheta(G)$ of a graph and a critical subset. The algorithm *CunninghamMin* is discussed in Appendix B. The *update* method is presented in the right Table. **Right**: Pseudocode of the *Update* method.

BinarySearch2D
Input: connected graph $G = (\mathcal{V}, \mathcal{E})$, $\mathcal{V} = n$, $\mathcal{E} = m$
Output:$\vartheta(G)$ of G, $E \subseteq \mathcal{E}$ critical

```
1 begin
2   Pr = {1,2,...,n-1}
3   qmin = {1,2,...,n-1}
4   while |Pr|>0
5      p <-- random(Pr)
6      for q=m downto qmin(p)
7         (E,minpq) = CunninghamMin((p/q)*1,G)
8          if n-1 <= minpq then
9             (Pr,qmin) = update(Pr,p,q)
10            goto 4
11         end  //if
12     end       //for
13     Pr = Pr-p
14  end           //while
15  return E, minpq
16 end             // begin
```

Update
Input: Pr, $p \in Pr$, $q \in \{q_{min}, |\mathcal{E}|\}$
Output: new Pr, q_{min}

```
1   begin
2     qmin(p) = q+1
3     for j=p+1 to |n|-1
4         qmin(j) = qmin(j-1)+1
5         if qmin(j)>m
6              Pr = Pr - j
7         end    //if
8     end        //for
9     return Pr, qmin
10   end           //begin
```

through any link in the critical set is the same. Since there exist, in general, multiple critical subsets, the NE of this game is typically not unique. We show, using a simple example, the importance of the critical subsets in the design of a robust network.

A polynomial time algorithm is presented, to compute the vulnerability of a graph as well as a critical set. The algorithm was previously presented in the literature. We discuss it and adapt it to the context of this paper.

A certain number of future directions are being explored by the authors. In the present paper, results have been obtained by assuming zero-attack cost for the attacker and an equal cost for all spanning trees in the network. Further investigations have shown that the notion of criticality of a set generalizes to the case where the attacker pays a certain cost to attack an edge. In this case, the definition of vulnerability needs a slight change to reflect the cost of attack.

Finally, in this paper, we only discuss the *1-connection* game in a graph. The case where the network chooses a *k-connected* component (for $k \geq 2$) and the attacker simultaneously attacks k or more links will be the subject of subsequent publications.

Acknowledgments

The authors would like to thank members of the Berkeley MURI and Netecon groups for their valuable input. Our special thanks go to Prof. Dorit Hochbaum for suggesting a set of very related papers. The work of the authors was supported by the ARO MURI grant W911NF-08-1-0233 and by the NSF grants CNS-0627161 and its continuation, CNS-0910702.

References

1. Baruch Awerbuch, Andrea Richa, and Christian Scheideler. A Jamming-Resistant MAC Protocol for Single-Hop Wireless Networks. *PODC 08: Proceedings of the 27th ACM symposium on Principles of distributed computing*, pages 45–54, 2008.
2. Stephen Boyd and Lieven Vandenberghe. *Convex Optimization*. Cambridge University Press, March 2004.
3. Paul A. Catlin, Hong-Jian Lai, and Yehong Shao. Edge-Connectivity and Edge-Disjoint Spanning Trees. *Discrete Mathematics*, 309(5):1033 –1040, 2009.
4. S Chopra. On the Spanning Tree Polyhedron. *Operations Research Letters*, 8(1):25 – 29, 1989.
5. Clayton W. Commander, Panos M. Pardalos, Valeriy Ryabchenko, Stan Uryasev, and Grigoriy Zrazhevsky. The Wireless Network Jamming Problem. *J. Comb. Optim.*, 14(4):481–498, 2007.
6. Raul Cordovil, Komei Fukuda, and Maria Leonor Moreira. Clutters and Matroids. *Discrete Math.*, 89(2):161–171, 1991.
7. Gerard Cornuejols. *Combinatorial Optimization: Packing and Covering*. Society for Industrial and Applied Mathematics, Philadelphia, PA, USA, 2001.
8. William H. Cunningham. Optimal Attack and Reinforcement of a Network. *J. ACM*, 32(3):549–561, 1985.
9. J Edmonds and D R Fulkerson. Bottleneck Extrema. *Journal of Combinatorial Theory*, (8):299–306, 1970.
10. D R Fulkerson. Blocking and Anti-Blocking Pairs of Polyhedra. *Math. Programming*, (1):168–194, 1971.
11. Assane Gueye, Jean C. Walrand, and Venkat Anantharam. Understanding the Design of Network Topology in Adversarial Environment. 2010. `http://www.eecs.berkeley.edu/~agueye/index.html`.
12. Anupam Gupta, Amit Kumar, Martin Pal, and Tim Roughgarden. Approximation Via Cost-Sharing: A Simple Approximation Algorithm for the Multicommodity Rent-or-Buy Problem. *J. ACM*, 54(3):11, 2007.
13. D Gusfield. Connectivity and Edge-Disjoint Spanning Trees. *Information Processing Letters*, (16):87–89, 1983.
14. David R. Karger and Clifford Stein. An o(n2) Algorithm for Minimum Cuts. pages 757–765, New York, NY, USA, 1993.
15. Jr. Kruskal, Joseph B. On the Shortest Spanning Subtree of a Graph and the Traveling Salesman Problem. *Proceedings of the American Mathematical Society*, 7(1):48–50, 1956.
16. D Fulkerson L Ford. Flows in Networks. pages 453–460. Princeton Univ. Press, 1962.
17. A. O. Matveev. Maps on Posets, and Blockers. *ArXiv Mathematics e-prints*, November 2004.
18. Andrey O Matveev. On Blockers in Bounded Posets. *Int. J. Math. Math. Sci*, (26):581–588.
19. J A Nash-Williams. Edge-Disjoint Spanning Trees of Finite Graphs. *Journal London Math. Soc*, (36):445–450, 1961.
20. R. C. Prim. Shortest Connection Networks and some Generalizations. *Bell System Technology Journal*, 36:1389–1401, 1957.
21. Vanderbei Robert. *Linear Programming: Foundations and Extensions*. Springer, May 2001.
22. Mechthild Stoer and Frank Wagner. A simple min-cut algorithm. *J. ACM*, 44(4):585–591, 1997.
23. W T Tutte. On the Problem of Decomposing a Graph into N Connected Factors. *Journal of the London Mathematical Society*, (36):221–230, 1961.
24. J. v. Neumann. Zur Theorie der Gesellschaftspiele. *Mathematische Annalen*, 100:295–320, 1928.
25. D Welsh. *Matroid Theory*. Academic Press, London, New York, San Francisco, 1976.

A Blocking Pairs of Polyhedra and the Spanning Tree Polyhedron

A.1 Blockers

Let N be a nonempty set that we will call the *ground set*, and let $\mathcal{J} = \{J_1, \ldots, J_p\}$ be a family of nonempty subsets of N. A subset J_0 of N is said to be a *blocking* set for $\mathcal{J}$ if $|J_0 \cap J_k| > 0$ for all $k \in \{1, \ldots, p\}$. The *blocker* of $\mathcal{J}$ is the family of all inclusion-wise minimal blocking sets of $\mathcal{J}$. As an example consider the graph $G = (\mathcal{V}, \mathcal{E})$ and let $N = \mathcal{E}$ the set of edges of G. Then the set $\mathcal{T}$ of spanning trees of G forms a family of subsets of $\mathcal{E}$. Any edge-cutset of the graph is blocking $\mathcal{T}$. The blocker of $\mathcal{T}$ is the set of all minimal cutsets of G.

In [6], [17], [18], the concept of blocker is defined as a mapping on families of subsets. More precisely:

Definition 2. *Given a ground set N, the* blocker map $b(\cdot)$ *is a function from the class $\mathcal{J}_N$ of all families of subsets on N to itself which associates to each family $\mathcal{J}$, its blocker*

$$b(\mathcal{J}) = \min\{J' : \ \ J' \subseteq N, \ \ J' \cap J \neq \emptyset, \ \ \forall\, J \in \mathcal{J}\}\,. \tag{28}$$

It has been shown [9] that if $\mathcal{J}$ is a family such that each element is not contained in another element (e.g. family of spanning trees), then the blocker map satisfies $b\,(b(\mathcal{J})) = \mathcal{J}$. As a consequence (since $b\,(b(\mathcal{J}))$ uniquely defines $\mathcal{J}$); $\mathcal{J}$ and $b(\mathcal{J})$ are said to form a *blocking pair*.

It is easy to see that if $\mathcal{J} = \{J_1, J_2, \ldots, J_k\}$ is the blocker of $\mathcal{J}' = \{J'_1, J'_2, \ldots, J'_p\}$, then $|J_i \cap J'_j| > 0 \ \forall\ J'_j \in \mathcal{J}'$ and for all $i = 1, \ldots, k$. Thus, any $J_i \in \mathcal{J}$ is actually blocking the family $\mathcal{J}'$.

A.2 Characterization of the spanning tree family

We have seen above that the blocker of the set $\mathcal{T}$ of spanning trees is the set of minimum cuts of the graph. Let $\mathcal{M}$ be the *tree-link* incidence matrix of $\mathcal{T}$. It characterizes the *spanning tree polyhedron* $\mathcal{P}$ which is defined as the vector sum of the convex hull of the rows of $\mathcal{M}$ and the nonnegative orthant:

$$\mathcal{P} = conv\{x \mid x \ \text{ is a row of } \mathcal{M}\} + R^m_+ \tag{29}$$

where $m = |\mathcal{E}|$.

Next we give another characterization of $\mathcal{P}$. Recall that for a connected graph $G = (\mathcal{V}, \mathcal{E})$, a minimum cut partitions the node set $\mathcal{V}$ into two subsets $\mathcal{V}_1$ and $\mathcal{V}_2$, and includes all the edges having one end point in $\mathcal{V}_1$ and the other one in $\mathcal{V}_2$. Furthermore, the subgraphs, $G_i = (\mathcal{V}_i, E(\mathcal{V}_i)),\ i = 1, 2$ are connected. This notion can be generalized. Consider a partition $P = (\mathcal{V}_1, \ldots, \mathcal{V}_{k_P})$ of the nodes of G such that each subgraph $G_i = (\mathcal{V}_i, E(\mathcal{V}_i)),\ i = 1, \ldots, k_P$ is connected. Such partition is said to be *feasible*.

The spanning tree polyhedron of the graph G is characterized by the following proposition [4].

Proposition 1. *The spanning tree polyhedron of the graph G corresponds to the set*

$$\mathcal{P} = \left\{ x \in R_+^m \mid \sum_{e \in \mathcal{E}(P)} x_e \geq k_p - 1, \ \forall \ P \text{ feasible partition} \right\} ,$$

where $\mathcal{E}(P)$ denotes the subset of edges that go between vertices in distinct elements of the partition P.

The *blocking polyhedron* of $\mathcal{P}$ (corresponding to the minimal cuts) is given by (see [10],[4], [7])

$$\hat{\mathcal{P}} = \left\{ y \in R_+^m \mid y \cdot \mathcal{P} \geq 1 \right\} .$$

In other words, $\hat{\mathcal{P}}$ consists of all nonnegative m-vectors y such that $y \cdot x \geq 1$ for all $x \in \mathcal{P}$.

Let $\hat{\mathcal{M}}$ be the $K \times m$ matrix whose rows correspond to the extreme points of $\mathcal{P}$.

Proposition 2. *The polyhedron $\hat{\mathcal{P}}$ is given by*

$$\hat{\mathcal{P}} = \left\{ y \in R_+^m \mid \hat{\mathcal{M}} y \geq 1 \right\} .$$

B Matroids, Polymatroids, and Network Flow

B.1 Matroids and Polymatroids

Let N be a finite set, and let $r : \ 2^N \rightarrow \mathbb{N}$ be a function from the family of subsets of N to the set of non-negative integers $\mathbb{N}$.

Definition 3. *$M = (N, r)$ is called a matroid if it satisfies the following properties:*

r.0: For all $J \subseteq N$, $r(J) \leq |J|$,
r.1: If $J' \subseteq J \subseteq N$, then $r(J') \leq r(J)$,
r.2: If $J, J' \subseteq N$, then $r(J \cup J') + r(J \cap J') \leq r(J) + r(J')$ (i.e. $r(\cdot)$ is submodular).

The subsets $I \subseteq N$ that verify $r(I) = |I|$ are called the independent sets of the matroid. Let $\mathcal{I}$ be the family of all independent sets. Sometime, the matroid is referred to by using the notation $M = (N, \mathcal{I})$

An example of a matroid is the collection of cycle-free subsets of edges of a graph $G = (\mathcal{V}, \mathcal{E})$ on the ground set $\mathcal{E}$. It is called the *graphic matroid* of the graph. Its rank function is given by letting $r(E)$ be defined as the maximum size of a subset of edges in E that does not contain a loop. It is known to be equal to $r(E) = |\mathcal{V}| - Q(G_E)$, where $Q(G_E)$ is the number of connected components of the subgraph $G_E = (\mathcal{V}, E)$. The graphic matroid and its rank function will be very useful in the rest of this appendix.

More details about matroids can be found in [25].

In section 4, we have seen that, to compute the vulnerability of a graph, the search algorithm needs an oracle that solves

$$\min_{E \subseteq \mathcal{E}} \left(\mathbf{y}_0(E) + f(\bar{E}) \right) , \tag{30}$$

where $\mathbf{y}_0 = \frac{p}{q}\mathbf{1}$ for p and q given by the search algorithm. Notice that $\mathbf{y}_0(E) = \frac{p}{q}|E|$ for any subset of edges $E \subseteq \mathcal{E}$ of the graph. In this section of the appendix, we discuss how such an oracle can be built. We start by defining the notion of a *polymatroid*.

Definition 4. *A real-valued function $f(\cdot)$, defined on subsets of N, is called a* **polymatroid function** *if it verifies*

P.0: $f(\emptyset) = 0$,
P.1: If $J \subseteq J' \subseteq N$, then $f(J) \leq f(J')$ (i.e. $f(\cdot)$ is non-decreasing),
P.2: If $J, J' \subseteq N$, then $f(J \cup J') + f(J \cap J') \leq f(J) + f(J')$ (i.e. $f(\cdot)$ is submodular).

Given a polymatroid function $f(\cdot)$, the following polyhedron is called the **polymatroid** *associated to f:*

$$P(f) = \left\{ \mathbf{x} \in R_+^{|N|},\ \mathbf{x}(J) \leq f(J),\ \forall J \subseteq N \right\}. \tag{31}$$

*For any $\mathbf{y} \in R_+^{|N|}$, $\mathbf{x} \in P(f)$ is called a $P(f)$-**basis** of $\mathbf{y}$ if $\mathbf{x}$ is a componentwise maximal vector of the set $\{\mathbf{x},\ \mathbf{x} \in P$ and $\mathbf{x} \leq \mathbf{y}\}$.*

The matroid rank function defined above is an example of polymatroid function.

The following (max-min) theorem relates the minimizing subsets of (30) to the $P(f)$-basis of $\mathbf{y}_0$. The proof of the theorem can be found in [8].

Theorem 3. *Let $f(\cdot)$ be a polymatroid function on subsets of N. Then, for any $\mathbf{y} \in R_+^{|N|}$ and any $P(f)$-basis $\mathbf{x}$ of $\mathbf{y}$, we have*

$$\mathbf{x}(N) = \min \left(\mathbf{y}(J) + f(\bar{J}),\ J \subseteq N \right) . \tag{32}$$

From this theorem, we see that an oracle that computes a $P(f)$-basis of $\mathbf{y}_0$ suffices for the minimization in (30). Let's see how such an oracle can be built.

The definition of $P(f)$-basis implies a very simple method for finding a $P(f)$-basis of any $\mathbf{y} \in R_+^{|N|}$. Namely,
start with $\mathbf{x} = 0$ and successively increase each component of $\mathbf{x}$ as much as possible while still satisfying $\mathbf{x} \leq \mathbf{y}$, and $\mathbf{x} \in P(f)$.

Implementing this simple and greedy algorithm might, however, not be so simple. In fact, it requires one to be able to compute, for a given $\mathbf{x} \in P(f)$ and any $j \in N$, the quantity

$$\epsilon_{max}(j) = \max(\epsilon\ :\ \mathbf{x} + \epsilon \mathbf{1}_j \in P(f)) , \tag{33}$$

where $\mathbf{1}_j$ is the incidence vector of subset $\{j\}$. $\epsilon_{max}(j)$ is the maximum amount by which component j of $\mathbf{x}$ can be increased while keeping $\mathbf{x}$ in $P(f)$.

Verifying that a vector $\mathbf{x}$ belongs to the polymatroid can be done using the following idea: if $\mathbf{x} \notin P(f)$, then one can find a subset J for which $\mathbf{x}(J) \leq f(J)$ is violated. If $\mathbf{x} \in P(f)$ and $j \in N$, then any ϵ such that $\epsilon > \min_{J \subseteq N} \left(f(J) - \mathbf{x}(J),\ j \in J \right)$ will send $\mathbf{x} + \epsilon \mathbf{1}_j$ out of $P(f)$.

Also, if $\mathbf{x}$ is a $P(f)$-basis of $\mathbf{y}$, then for any $j \in N$, either $\mathbf{x}(j) = \mathbf{y}(j)$ or $\mathbf{x}(J) = f(J)$ for some subset J containing j. In fact, for all $j \in N$

$$\epsilon_{max}(j) = \min \left\{ \mathbf{y}(j) - \mathbf{x}(j), \min_J \left(f(J) - \mathbf{x}(J),\ j \in J \subseteq N \right) \right\} . \tag{34}$$

If the minimum is achieved at $\mathbf{y}(j) - \mathbf{x}(j)$, then $\mathbf{x} \leftarrow \mathbf{x} + \epsilon_{max}(j)\mathbf{1}_j$ will satisfy $\mathbf{x}(j) = \mathbf{y}(j)$. Otherwise, there exists some $J_j \ni j$, such that $\mathbf{x}(J_j) = f(J_j)$ (J_j is said to be *tight*). Letting $\bar{J} = \bigcup_j J_j$, and $\mathbf{x}$ being the $P(f)$-basis obtained after running the greedy algorithm, it can be shown (see [8]) that $f(\bar{J}) = \mathbf{x}(\bar{J})$ (union of tight set is tight). For such $\bar{J}$, we have that

$$\mathbf{x}(N) = \mathbf{x}(J) + \mathbf{x}(\bar{J}) = \mathbf{y}(J) + f(\bar{J}) \; . \tag{35}$$

This is because $\mathbf{x}(\bar{J}) = f(\bar{J})$ and if $j \notin \bar{J}$, $\mathbf{x}(j) = \mathbf{y}(j)$.

Based on these observations, Cunningham [8] proposed a modified version of the greedy algorithm to compute a $P(f)$-basis, as well as a minimizing subset for the minimization in (32). The algorithm is presented in Table 2.

It starts with $\mathbf{x} = 0$ and $\bar{J} = \emptyset$. For each $j \in N$, the component $\mathbf{x}(j)$ is increased as much as possible: $\mathbf{x} \leftarrow \mathbf{x} + \epsilon_{max}(j)\mathbf{1}_j$. If the minimum in (34) is achieved at $\min_{J'} (f(J') - \mathbf{x}(J'),\ j \in J')$, then update $\bar{J} \leftarrow \bar{J} \cup J'$ where J' is a minimizer. At the end of the algorithm, $\bar{J}$ is a tight set and $\mathbf{x}$ is maximal. Also, it satisfies $\mathbf{x} \in P(f)$ and $\mathbf{x} \leq \mathbf{y}$, with $\mathbf{x}(N) = \mathbf{y}(J) + f(\bar{J})$.

To find a $P(f)$-basis, Cunningham's algorithm performs $|\mathcal{E}|$ computations of the the minimization below:

$$\min_J \left(f(J) - \mathbf{x}(J),\ j \in J \subseteq N\right) \; . \tag{36}$$

Now, all that remains is to find an algorithm that computes the minimization in polynomial time. This is the subject of the next section.

Table 2. Pseudocode of the oracle *CunninghamMin* that solves the minimization (36).

Cunningham
Input: Polymatroid function f, $\mathbf{y} \in R_+^{|N|}$
Output: minimum eps, minimizer T

```
1 begin
2  x = 0
3  J := {}
4  for j in N
5     eps := min(f(J')-x(J'): j in J')
6     J'(j) := a minimizer
7     if eps <= y(j)-x then J:=J U J'(j)
8     else eps:= y(j)-x(j)
9     end        //if
10    x=x+eps*1(j)
11  end   //for
12 end    //begin
```

B.2 Network Flow

In the notation of the last two sections, $\mathcal{E}$ below will be a ground set (N above), and subsets of $\mathcal{E}$ will be referred to using E (J and I above).

Let $G = (\mathcal{V}, \mathcal{E})$ be a connected graph and let $f(\cdot)$ the rank function of the graphic matroid that is associated to G. We have seen above that $f(E) = |\mathcal{V}| - Q(G_E)$. Let $P(f)$ be the polymatroid associated with $f(\cdot)$. An equivalent description of $P(f)$ is given as follows (see [8]):

$$P(f) = \left\{ \mathbf{x} \in R_+^{|\mathcal{E}|},\ \mathbf{x}(\gamma(B)) \leq |B| - 1 \text{ for all } B,\ \emptyset \neq B \subseteq \mathcal{V} \right\}, \tag{37}$$

where $\gamma(B)$ denotes the set of edges with both ends in B.

Recall that our goal is, for a given j, to find a subset E, $j \in E \subseteq \mathcal{E}$ that minimizes $f(E) - \mathbf{x}(E)$. This is equivalent to finding B that minimizes $|B| - 1 - \mathbf{x}(\gamma(B))$, with $j \in \gamma(B)$.

To find the minimizing subset of nodes, B, we define the following graph G' for a given polymatroid function $f(\cdot)$, $\mathbf{x} \in P(f)$, and edge $j \in \mathcal{E}$. The vertices of G' are $\mathcal{V} \cup (r, s)$ for new vertices r and s. Each $e \in \mathcal{E}$ is an edge of G', having the same ends and having capacity $\frac{1}{2}\mathbf{x}_e$. There is an edge joining v to s for each $v \in \mathcal{V}$, it has capacity 1. There is an edge joining r to v for each $v \in \mathcal{V}$. It has capacity ∞ if v is an end of j, and otherwise it has capacity $\mathbf{x}(\delta(v))$. (Here $\delta(B) = \{e \in \mathcal{E},\ e \text{ has exactly one end in } B \subseteq \mathcal{V}\}$, $\delta(v)$ is shorthand for $\delta(\{v\})$. This construction is illustrated in Figure 4(a). Its motivation is to ensure that $j \in \gamma(B)$ as can be seen next.

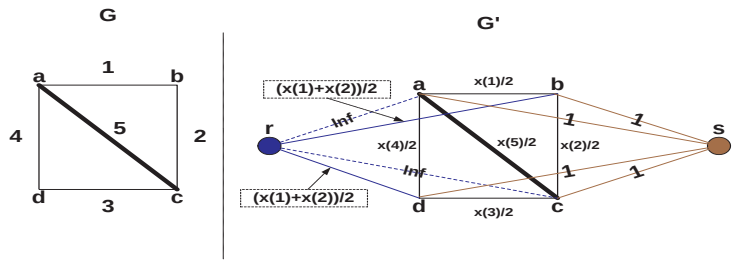
Now consider a cut in G' induced by the set $B \cup \{r\}$, where $j \in B \subseteq \mathcal{V}$. It is the set of links that have one end in $B \cup \{r\}$ and the other end in the complement of $B \cup \{r\}$. The capacity of such cut is (see an illustration in Figures 4(b))

$$|B| + \frac{1}{2}\mathbf{x}\left(\delta(B)\right) + \mathbf{x}\left(\gamma(\bar{B})\right) + \frac{1}{2}\mathbf{x}\left(\delta(B)\right) = |B| + \mathbf{x}(\mathcal{E}) - \mathbf{x}(\gamma(B)) \tag{38}$$

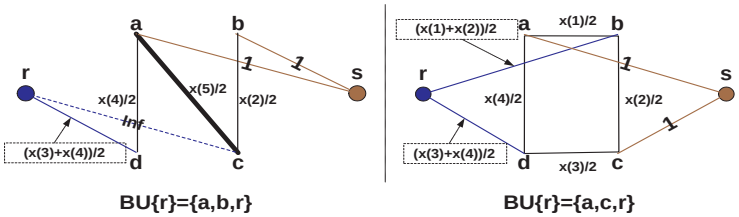
$$= |B| - 1 - \mathbf{x}(\gamma(B)) + (\mathbf{x}(\mathcal{E}) + 1). \tag{39}$$

The first term in the LHS of equation (38) corresponds to edges going from nodes in B to the sink s. There are $|B|$ of them, each having capacity 1. The next term corresponds to edges going from a node in B to a node in $\bar{B}$. The last two terms correspond to edges going from the root r to nodes in $\bar{B}$. For each such edge (r, u), the capacity is defined as $\frac{1}{2}\delta(\{u\})$. Let $e = (u, v) \in \delta(\{u\})$. Then, if $v \in B$ (i.e. $e \in \delta(B)$), then $\mathbf{x}(e)$ appears only in the capacity of (r, u); implying the term $\frac{1}{2}\mathbf{x}\left(\delta(B)\right)$. If, on the other hand, $v \notin B$ (i.e. $e \in \gamma(\bar{B})$), then $\mathbf{x}(e)$ appears both in the capacity of (r, u), and in that of (r, v), thus the term $\mathbf{x}\left(\gamma(\bar{B})\right)$.

Now, since a cut induced by a subset of edges B will have infinite capacity if $j \notin \gamma(B)$, a minimum cut in G' will indeed have the form $B \cup \{r\}$ with $j \in B$, hence, minimizing $|B| - 1 - \mathbf{x}(\gamma(B))$. As a consequence, any network flow algorithm can serve as an oracle for Cunningham's algorithm. Many polynomial implementations of network flow algorithms ([22], [14]) have been proposed since the proof of the Max-Flow Min-Cut theorem by Ford and Fulkerson [16] in 1962.



(a) Constructing the graph G' from G for the network flow algorithm.



(b) Illustrating the cut induced by $B \cup \{r\}$

Fig. 4. Constructing the graph G' for the network flow algorithm. Figure 4(a) shows the construction of G' from G. The edge under consideration in this example is $j = 5$. Examples in Figures 4(b) show the cut induced by $B \cup \{r\}$ for $B \subseteq \mathcal{V}$. In the left figure, $B = \{a, b\}$ does not contain $j = 5$. The capacity of this cut is equal to infinity. In the right figure, $B = \{a, c\}$ which contains edge $j = 5$ (the only edge). As can be seen in the figure, the capacity of the cut induced by this choice of B is $2 + \mathbf{x}(1) + \mathbf{x}(2) + \mathbf{x}(3) + \mathbf{x}(4)$ which is finite.