Adversary Games in Secure/Reliable Network Routing

Gruia Calinescu, Sanjiv Kapoor, Michael Quinn, and Junghwan Shin

Illinois Institute of Technology, Illinois, Chicago 60616, USA
{calinescu,kapoor,mquinn4,jshin7}@iit.edu

Abstract. In this paper, we consider security aspects of network routing in a game-theoretic framework where an attacker is empowered with an ability for intrusion into edges of the network; on the other hand, the goal of designer is to choose routing paths.

We interpret the secure routing problem as a two player zero sum game. The attacker can choose one or more edges for intrusion, while the designer has to choose paths between source-destination pairs for a given set of pairs. We give polynomial-time algorithms for finding mixed Nash equilibria if 1) the attacker is limited to a one-edge attack (for arbitrary number of source-destination pairs), 2) the designer has two source-destination pairs while the attacker is either limited to c edges, for given c, or the attacker incurs a cost for each edge attacked. Previous work gave an algorithm for one source-destination pair and multiple edge attacks.

Keywords: zero-sum games, Nash equilibrum, network flows, concurrent flows.

1 Introduction

1.1 Motivation

Current routing protocols utilize single paths and are typically prone to failures or malicious attacks. The fast growth of the Internet underscores the need for Network Security. With the current emphasis on distributed computing the network is further exposed to numerous dangerous situations, e.g., security attacks, eavesdropping and so on. An early illustration of this is [24] which considers various *attacks* and *defenses.*

In this paper we consider the security issues modeled as a game between the network designer and the malicious attacker. Attackers may be any malicious users attempting to intercept or eavesdrop packets on the network or impact physical constraints e.g., disconnection, congestion , etc. The network designer has to route data flow between multiple source-sink pairs and is allowed to utilize multiple paths for routing. The attacker selects edges for attack. The attacks can be on either single or multiple edges simultaneously. Similarly, the designer can choose single or multiple paths simultaneously. In either case, the choice of path(s) is considered to be stochastic, i.e. one or more paths are chosen from a given set with a specific probability distribution.

An earlier illustration of this approach in the context of network routing is illustrated by the result on Game Theoretic Stochastic Routing (GTSR), which describes a particular approach to multi-path routing that provides a rigorous way to determine which routes should be used and with what frequency[6][4]. GTSR finds all paths between a source-destination pair and computes next hop probabilities, i.e., the probabilities that a packet takes a particular next-hop. This contrasts with single-path algorithms that simply determine the next-hop. Based on stochastic routing, this paper consider two alternatives : offline games where the attacker starts by selecting one link or a particular node and online games where the attacker scans on physical interface at all nodes [4]. They present results only for single source-sink pairs.

A different consideration of security for network systems via a game theoretic framework can be found in [26]. In this paper they model a security attack as a 2-player stochastic game. They utilize non-linear programming to determine the Nash equilibrium. However, they consider a set of attacks on a node of the computer network instead of an attack on the entire network.

In an earlier paper, Washburn and Wood [25] considers a similar model but in the context of providing security for defense/policing application. They consider game-theoretic models where an *evader* selects and attempts to traverse a path through a network from node s to node t without being detected by an *interdictor*. They have shown that optimal strategies can be obtained by solving a maximum flow problem on the network. Another result that measures the reliability of a general transport network but via only single edge attacks can be found in [3], where a two player game is formulated. The goal of the designer is to find a least cost path and the attacker utilizes a single edge attack. Linear programming formulations are provided when the link costs are fixed.

Multi-path routing is useful for enhanced reliability in the Internet. Traditionally, a single link failure, by a malicious user or a physical reason, can take a small but significant amount time to detect and correct. Further, another disadvantage is that malicious behaviors can be realized easily due to a prediction of paths. On the other hand, multi-path routing provides a set of possible paths that causes hard-prediction of forwarding packets. Moreover, it ensures that data throughput increases[7], traffic congestion decreases[9], network utilization is improved[20][12] and network security increases.[6][21][14][4]. In wireless, adhoc and sensor networks, routing with vulnerable links(interpreted as malicious attackers) is a challenging problem. Multi-path routing can ease unreliability and vulnerability[8][22][17]. Related game theoretic research on computer security includes the work of [10][18][19][8].

As in [25], the results in [5] consider the case of routing a single source-destination path under single edge attacks. Both papers formulate an LP solution and show that the value of the game is $1/f$, where f is a value of max flow.

The case where multiple source-destination paths are to be found is not considered in these papers.

In this paper we consider the case where the designer has to select k paths, each from given source s_i to given destination t_i. The attacker can select a set of c edges, with c also given. We give a polynomial-time algorithm for finding mixed Nash equilibria in directed graphs when $c = 1$ by using concurrent flows linear programs. We also give a polynomial-time algorithm for finding mixed Nash equilibria when the input graph is undirected and $k = 2$, for arbitrary c not exceeding the minimum edge-cut of the graph, using Hu's [11] two commodity flows. We leave open the case $k \geq 3$ and $c \geq 2$. Our approach does not extend to this case due to the fact that maxflow/mincut type conditions do not hold.

We also consider the case where the attacker can select an arbitrary set of edges incurring a cost of attack, which is dependant on the edge, and show how to compute the equilibrum in polynomial time for two commodities in undirected graphs, again using [11].

We remark that all the games considered, here and in previous work, are linear programs, but with exponential-size matrices. The challenge is finding polynomial-time algorithms for solving such linear programs; thus the matrices must be considered implicitly.

The paper outline is as follows. In Section 2, we describe the previously known game [25][5] where a designer uses one path; on the other hand, an attacker attempts to destroy one edge. While the pay-offs for these case are known, we illustrate the methodology linking these games to network flows. In Section 3, we consider the game where a designer is required to find k paths for given source-destination pairs, while the attacker still attempts to destroy one edge. In Section 4 we describe results for two source-destination pairs, while the attacker can choose c edges. In Section 5 we present polynomial-time algorithms for the case where the attacker incurs a cost for each edge attacked (dependant on the edge) while the designer chooses two paths, from given s_1 to t_1 and s_2 to t_2.

1.2 Notations

Given the directed or undirected graph, $G = (V, E)$, where V is a set of vertices and E is a set of edges we consider the network $N(G, S - T)$ where $S - T$ is a set of k source-destination pairs $(s_i, t_i), i = 1 \ldots k, s_i, t_i \in V$. Let us define $\mathcal{P}$ and $\mathcal{P}^i$ as the set of all possible source-destination paths and the set of possible $s_i - t_i$ paths, respectively. We sometimes call an $s_i - t_i$ paths a path for commodity i. We let S_A be the strategy set of the attacker and S_D be the strategy set of the designer. Each strategy $s \in S_A$ of the attacker is a subset of edges of size c, i.e. $s = \{e_1 \ldots e_c\}, e_i \in E$, which he chooses to attack. And each strategy of the designer $p \in S_D$ is a set of paths, one for each source-sink pair, i.e $p = \{P_1, \ldots, P_k\}, P_i \in \mathcal{P}^i$. We use the following payoff matrix, for $s \in S_A$ and $p \in S_D$: the attacker gains $|\{i \in \{1, 2, \ldots, k\} \mid P_i \cap s \neq \emptyset\}|$. That is, the attacker's payoff is the number of intercepted paths (and intercepting a path more than once does not help him). We make this a zero-sum gain so the designer's payoff is the negation of the attacker's payoff; a designer therefore would minimize the number of intercepted paths. A mixed strategy of the attacker is denoted by the vector x, where $x = (x_1, x_2, ..., x_r)$ where $r = |S_A|$ and y denotes the mixed strategy of the designer, where $y = (y_1, y_2, ..., y_t)$ $t = |S_D|$.

In another version of the game, we use the following payoff matrix, for $s \in S_A$ and $p \in S_D$: the attacker gains $|\{i \in \{1, 2, \ldots, k\} \mid P_i \cap s \neq \emptyset\}| - \sum_{e \in s} c(e)$. That is, the attacker's payoff is the number of intercepted paths, less the total cost of the edges attacked. We consider this to be a zero-sum game so the designer's payoff is the negation of the attacker's payoff; the designer, therefore, would minimize the payoff of the attacker. A mixed strategy of the attacker is denoted by the vector x, where $x = (x_1, x_2, ..., x_r)$ where $r = |S_A|$ and y denotes the mixed strategy of the designer, where $y = (y_1, y_2, ..., y_t)$ and $t = |S_D|$.

Note that the Nash equilibrium is the solution of a linear program, which in our case has exponentially many variable and constraints. We are able to find an equilibrium point in polynomial time only for the case of one or two commodities. We directly describe the two-commodity case; the simpler one-commodity case can be solved by maximum flow and the methods of the two-commodity case, and can also be reduced to the two-commodity case by setting $s_2 = t_2$.

The attacker will pick, without loss of generality, all the edges of cost 0. Therefore, for the purpose of computing the equilibrium, we are only concerned with those commodity i for which there exist an $s_i - t_i$ path all whose edges have strictly positive costs. We discard from now on the commodities for which no such path exist as whatever path a designer chooses for such a pair $s_i - t_i$, it will be intercepted. For the remaining commodities, without loss of generality, the designer will never choose a path with an edge with zero cost, and therefore from now on we assume $c(e) > 0$ for all $e \in E$.

2 The (One Commodity, One Edge)-Problem

We consider the simplest version of the problem where $S - T$ comprises one source-sink pair (s, t). The attacker is allowed to attack one edge and the designer has to choose one path. The strategy sets are $S_A = E$ and $S_D = \mathcal{P}$ where $\mathcal{P}$ is the set of paths from s to t. We consider a zero-sum game where the attacker gains 1 unit if his chosen edge successfully intersects the designer path and the designer loses 1 unit. In the case when his chosen edge does not intersect the designer path, the designer and attacker loose or gain nothing.

We consider the pay-off matrix A of size $(m \times q), q = |\mathcal{P}|$, where player 1 (attacker) has m strategies (number of edges) and player 2 (designer) has q strategies (number of $s - t$ paths). Then, we can assign a payoff to each element of $A = a_{ij}$ as follows: when the attacker chooses a strategy i and the designer chooses a strategy j : if $e_i \in P_j$ then $a_{ji} = 1$. Otherwise, $a_{ji} = 0$.

2.1 Reduction to the Max-flow Min-cut Problem

Let us define the LPs for player 1 and player 2 as *LP1* and *LP2*, respectively, as follows :

LP1: Maximize w subject to
$$\sum_{i=1}^{m} x_i a_{ji} \geq w \ \forall 1 \leq j \leq q$$
$$\sum_{i=1}^{m} x_i = 1$$
$$x_i \geq 0 \qquad \forall i \in E$$

LP2: Minimize λ subject to
$$\sum_{j=1}^{q} a_{ji} y_j \leq \lambda \ \forall 1 \leq i \leq m$$
$$\sum_{j=1}^{q} y_j = 1$$
$$y_j \geq 0 \qquad \forall 1 \leq j \leq q$$

where x_i and y_j represent probabilities to choose an edge and a path, respectively. a_{ij} represents the payoff obtained when the designer uses strategy i and the attacker uses strategy j. In this game, a Nash equilibrium set of strategies is a probability distribution on the set of edges corresponding to the attacker choice of edges and a probability distribution on the possible flow paths chosen by the designer.

Theorem 1. *[23] (*Menger's theorem *(directed vertex-disjoint version)). Let $D = (V, A)$ be a digraph and let $S, T \subseteq V$. Then the maximum number of vertex-disjoint $S - T$ paths is equal to the minimum size of an $S - T$ disconnecting vertex set.*

Theorem 2. *[23] A maximum collection of arc-disjoint $s-t$ paths and a minimum-size $s - t$ cut can be found in time $O(m^2)$.*

2.2 Algorithm

1. Attacker : find min-cut, C, which separates V into two disjoint subset of vertices. The attacker chooses a strategy that selects each edge in C with probability $1/|C|$.
2. Designer : construct a undirected graph from a given network instance. choose r-disjoint paths which are equally assigned $1/r$, where $|C| = r$. By Theorem 1, we know that *the maximum number of arc-disjoint s-t paths is equal to the minimum size of an s-t edge-cut.* The theorem holds in the case of undirected graphs as follows. The undirected vertex-disjoint version follows immediately from Theorem 1 by replacing each undirected edge by two oppositely oriented arcs. Next, the undirected edge-disjoint version follows from the undirected vertex-disjoint version. We can find arc-disjoint $s - t$ paths in time polynomial by Theorem 2.

We can show the correctness by showing the solutions are feasible for the LP1 and LP2 and have the same payoff. Recall that a profile S of mixed strategies is a mixed strategy Nash equilibrium if and only if every player's mixed strategy is a best response to the other player's mixed strategies.

Theorem 3. *[5][25] A strategy (α^*, β^*) given by the algorithm above is a mixed Nash equilibrium for the attacker and designer.*

3 The (k Commodities, One Edge)-Problem

In this section we consider the problem where the designer chooses paths for multiple source-sink pairs, while the attacker is allowed one edge for intrusion. We present the result for directed graphs, although the results hold for undirected graphs as well.

Let A be the payoff matrix. With q_i denoting the number of $s_i - t_i$ paths, A has q rows, where $q = \Pi_{i=1}^{k} q_i$, and m columns, where $m = |E|$ (recall that in this section, an attacker's strategy is a single edge). Let P_j^i be the j^{th} path from s_i to t_i, for $1 \leq i \leq k$ and $1 \leq j \leq q_i$. Let r be some correspondence from $\{1, 2, \ldots, q\}$ to $\{1, 2, \ldots, q_1\} \times \{1, 2, \ldots, q_2\} \times \cdots \times \{1, 2, \ldots, q_k\}$, with $r_i(j)$ giving the i^{th} component of r ($1 \leq i \leq k$). We index the columns of A by e, for $e \in E$. Then, based on previous discussion, we have for $e \in E$ and $j \in \{1, 2, \ldots, q\} : A_{je} = |\{i \in \{1, 2, \ldots, k\} \mid e \in P^i_{r_i(j)}\}|$.

Based on standard game theory, to find a Nash equilibrium the attacker must solve the following linear program with variables w and x_e, for $e \in E$:

LP3: Maximize w subject to
$$\sum_{e\in E} A_{je} x_e \geq w \ \forall 1 \leq j \leq q \quad (1)$$
$$\sum_{i=1}^{m} x_e = 1 \quad (2)$$
$$x_e \geq 0 \qquad \forall e \in E. \quad (3)$$

We denote the program above *LP3*. The designer must solve the linear program below, with variable λ and y_j:

LP4: Minimize λ subject to
$$\sum_{j=1}^{q} A_{je} y_j \leq \lambda \ \forall e \in E \quad (4)$$
$$\sum_{j=1}^{q} y_j = 1 \quad (5)$$
$$y_j \geq 0 \qquad \forall j \in \{1, 2, \ldots, q\} \quad (6)$$

We denote the program above *LP4*. LP3 and LP4 are duals. With the goal of solving the exponentially large programs LP3 and LP4, we introduce the following linear program, with variables α and f_P, for $P \in \mathcal{P}$.

LP5: Minimize α subject to
$$\sum_{P\in\mathcal{P} \mid e\in P} f_P \leq \alpha \ \forall e \in E \quad (7)$$
$$\sum_{j=1}^{q_i} f_{P_j^i} = 1 \qquad \forall i \in \{1, 2, \ldots, k\} (8)$$
$$f_P \geq 0 \qquad \forall P \in \mathcal{P} \quad (9)$$

Recall that $\mathcal{P}$ is the set of all source-destination paths, while for $i \in \{1, 2, \ldots, k\}$, the set of $s_i - t_i$ paths is $\{P_1^i, P_2^i, \ldots, P_{q_i}^i\}$. We denote the program above *LP5*. LP5 is also exponentially large, but it is known that it can be solved in polynomial time as it is a concurrent flow problem [23]. Precisely, LP5 is solved by the standard decomposition into path-flows of the solution of the following linear program, with variables f_e^i for $e \in E$ and $i \in \{1, 2, \ldots, k\}$:

LP7: Minimize γ subject to
$$\sum_{e \in \delta^-(u)} f_e^i = \sum_{e \in \delta^+(u)} f_e^i \quad \forall i \wedge \forall u \in V \setminus \{s_i, t_i\}$$
$$\sum_{e \in \delta^+(s_i)} f_e^i - \sum_{e \in \delta^-(s_i)} f_e^i = 1 \; \forall i$$
$$\textstyle\sum_{i=1}^k f_e^i \leq \gamma \qquad \forall e \in E$$
$$f_e^i \geq 0 \qquad \forall e \in E \wedge i$$

Here $\delta^-(u)$ denotes the set of edges entering u, and $\delta^+(u)$ denotes the set of edges leaving u. We denote the program above *LP7*.

The dual of LP5, denoted as *LP6*, given below, can also be solved in polynomial-time by solving the dual of the concurrent flow program above. LP6 has variables d_i, for $i \in \{1, 2, \ldots, k\}$, and l_e for $e \in E$; it has exponentially many constraints:

LP6: Maximize $\sum_{i=1}^k d_i$ subject to
$$d_i \leq \sum_{e \in P_j^i} l_e \;\; \forall i \wedge \forall j\} \tag{10}$$
$$\textstyle\sum_{e \in E} l_e = 1 \tag{11}$$
$$l_e \geq 0 \qquad \forall e \in E \tag{12}$$

As an aside, LP6 is a fractional version of Sparsest Cut [15,13,16,2,1]. After optimally solving LP5 and LP6, we assign to the variables of LP3 and LP4 values as follows. For LP3, we set $w = \sum_{i=1}^k d_i^*$ and $x_e = l_e^*$, where d_i^*, for $i \in \{1, 2, \ldots, k\}$, and l_e^* for $e \in E$ are an optimal solution to LP6. For LP4, we set $\lambda = \alpha^*$ and $y_j = \Pi_{i=1}^k f_{P^i_{r_i(j)}}^*$, where α^* and f_P^* are an optimal solution to LP5. As the number of nonzero y_j can be exponentially large, it has to be remembered implicitly, and indeed the designer can generate one random strategy by drawing independently at random for each $i \in \{1, 2, \ldots, k\}$ one P_j^i with probability $f_{P_j^i}^*$. There is an alternate way for the designer, used in [6] (but not for this game): at each node of the network, the packets going from s_i to t_i are not forwarded deterministically but exit node u on edge e with probability $\frac{f_e^i}{\sum_{e \in \delta^+(u)} f_e^i}$, where f_e^i are from an acyclic optimum to LP7; in this case the game is played separately for each packet.

We have not proved yet that our w and x_e are feasible for LP3, and λ and y_j are feasible for LP4; however once we do so they are both optimal, since the objective functions of the two dual linear programs LP3 and LP4 match: $w = \lambda$ (as $\alpha^* = \sum_{i=1}^k d_i^*$).

Claim. As defined above, w and x_e are feasible for LP3.

Proof. We have to verify the constraints of LP3. Indeed, constraints 3 follow immediately from constraints 12, and 2 follows from 11. As for constraints 1, let $e \in E$ and $j \in \{1, 2, \ldots, q\}$. We have

$$\sum_{e \in E} A_{je} x_e = \sum_{e \in E} x_e |\{i \in \{1, 2, \ldots, k\} \mid e \in P^i_{r_i(j)}\}| = \sum_{i=1}^{k} \sum_{e \in P^i_{r_i(j)}} x_e$$
$$= \sum_{i=1}^{k} \sum_{e \in P^i_{r_i(j)}} l^*_e \geq \sum_{i=1}^{k} d^*_i = w,$$

where the inequality follows from constraint 10 of LP6. ■

Claim. As defined above, λ and y_j are feasible for LP4.

Due to the page limitation, we omit the long technical proof. A similar proof appears later for Lemma 1.

4 The (2 Commodities, *c* Edges)-Problem

In this section we assume the input graph is undirected and c does not exceed the minimum edge-cut of the graph. Let A be the payoff matrix. With q_i denoting the number of $s_i - t_i$ paths, A has q rows, where $q = q_1 \cdot q_2$, and $\hat{m}$ columns, where $\hat{m} = \binom{m}{c}$. Let P^i_j be the j^{th} path from s_i to t_i, for $1 \leq i \leq 2$ and $1 \leq j \leq q_i$. Let r be some correspondence from $\{1, 2, \ldots, q\}$ to $\{1, 2, \ldots, q_1\} \times \{1, 2, \ldots, q_2\}$, with $r_i(j)$ giving the i^{th} component of r $(1 \leq i \leq 2)$.

Let us use $\mathcal{T}$ to denote the set of all sets $S \subset E$ with $|S| = c$. We index the columns of A by $S \in \mathcal{T}$. Then, based on previous discussion, we have for $S \in \mathcal{T}$ and $j \in \{1, 2, \ldots, q\}$: $A_{jS} = |\{i \in \{1, 2\} \mid P^i_{r_i(j)} \cap S \neq \emptyset\}|$.

Based on standard game theory, to find a Nash equilibrium the attacker must solve the following linear program with variables w and x_S, for $S \in \mathcal{T}$:

LP8: Maximize w subject to
$$\textstyle\sum_{S \in \mathcal{T}} A_{jS} x_S \geq w \; \forall 1 \leq j \leq q \quad (13)$$
$$\textstyle\sum_{i=1}^{\hat{m}} x_S = 1 \quad (14)$$
$$x_S \geq 0 \qquad \forall S \in \mathcal{T}. \quad (15)$$

We denote the program above *LP8.* The designer must solve the linear program below, with variable λ and y_j, for $1 \leq j \leq q$:

LP9: Minimize λ subject to
$$\textstyle\sum_{j=1}^{q} A_{jS} y_j \leq \lambda \; \forall S \in \mathcal{T} \quad (16)$$
$$\textstyle\sum_{j=1}^{q} y_j = 1 \quad (17)$$
$$y_j \geq 0 \qquad \forall j \in \{1, 2, \ldots, q\} \quad (18)$$

We denote the program above *LP9.* LP8 and LP9 are duals. With the goal of solving the exponentially large programs LP8 and LP9, we introduce the following linear program, with variables α and f_P, for $P \in \mathcal{P}$.

LP10: Minimize α subject to
$$\textstyle\sum_{P \in \mathcal{P} \mid e \in P} f_P \leq \alpha \; \forall e \in E \quad (19)$$
$$\textstyle\sum_{j=1}^{q_i} f_{P^i_j} = 1 \qquad \forall i \in \{1, 2\} \quad (20)$$
$$f_P \geq 0 \qquad \forall P \in \mathcal{P} \quad (21)$$

Recall that $\mathcal{P}$ is the set of all source-destination paths, while for $i \in \{1, 2\}$, the set of $s_i - t_i$ paths is $\{P_1^i, P_2^i, \ldots, P_{q_i}^i\}$. We denote the program above *LP10* and note that the only difference between LP5 and LP10 is that LP10 has $k = 2$ and an undirected graph instead of arbitrary k and directed graph. LP10 is also exponentially large, but it is known that it can be solved in polynomial time [11], [23].

We show later how to extract a feasible solution for LP8 from an LP10 solution. However, the dual of LP10 does not immediately give a solution to LP9, and we do not use it. Instead, we use a sort of integral dual of LP10, which is what [11] provides. Precisely, let $C_1 \subset E$ be the minimum cut separating s_1 from t_1, let $C_2 \subset E$ be the minimum cut separating s_2 from t_2, and let $C_3 \subset E$ be the minimum cut separating both $s_i - t_i$ pairs. In other words, in the graph $(V, E \setminus C_3)$ there exist neither $s_1 - t_1$ nor $s_2 - t_2$ paths; C_3 can be computed by computing two minimum cuts: one separating s_1, s_2 from t_1, t_2, and the other separating s_1, t_2 from t_1, s_2. Then Hu's theorem states:

Theorem 4. *If* $\min(|C_1|, |C_2|) \leq |C_3|/2$*, then there exist a concurrent two-commodity flow in G shipping* $\min(|C_1|, |C_2|)$ *from* s_1 *to* t_1 *and* $\min(|C_1|, |C_2|)$ *from* s_2 *to* t_2*. Otherwise (*$|C_3|/2 < |C_1|$ *and* $|C_3|/2 < |C_2|$*), there exist a concurrent two-commodity flow in G shipping* $|C_3|/2$ *units of flow from* s_1 *to* t_1 *and* $|C_3|/2$ *units of flow from* s_2 *to* t_2*. Moreover, these two-commodity flows can be found in time polynomial in the size of G.*

It is clear (and known) that in LP10 we have for any feasible solution $\alpha \geq 1/|C_1|$, $\alpha \geq 1/|C_2|$, and $\alpha \geq 2/|C_3|$. Let us consider three cases, with the second being symmetric to the first.
Case 1. In the first case, $|C_1| \leq |C_2|$ and $|C_1| \leq |C_3|/2$. Then Hu's theorem and algorithm finds a feasible solution to LP10 with $\alpha = 1/|C_1|$. In LP9, we assign $\lambda = c \cdot \alpha$ and $y_j = f_{P^1_{r_1(j)}} \cdot f_{P^2_{r_2(j)}}$.

Lemma 1. *λ and y_j as described above make a feasible solution to LP9.*

Proof. The constraints 18 follow immediately from the definition of y_j and constraints 21. Regarding constraint 17, we have:

$$\sum_{j=1}^{q} y_j = \sum_{l_1=1}^{q_1} \sum_{j \in \{1,2,\ldots,q\} \mid r_1(j)=l_1} y_j = \sum_{l_1=1}^{q_1} \sum_{l_2=1}^{q_2} \sum_{j \in \{1,2,\ldots,q\} \mid r_1(j)=l_1 \wedge r_2(j)=l_2} y_j$$
$$= \sum_{l_1=1}^{q_1} \sum_{l_2=1}^{q_2} f_{P^1_{l_1}} \cdot f_{P^2_{l_2}} = \sum_{l_1=1}^{q_1} f_{P^1_{l_1}} \cdot \sum_{l_2=1}^{q_2} f_{P^2_{l_2}}$$
$$= \sum_{l_1=1}^{q_1} f_{P^1_{l_1}} = 1,$$

where we used the constraints 20 of LP10. Finally, to verify Constraints 16, we let $S \in \mathcal{T}$ be arbitrary. For path P and subset $S \subseteq E$, we define the matrix $B_{P,S}$ as follows:

$$B_{P,S} = \begin{cases} 1 \text{ if } P \cap S \neq \emptyset \\ 0 \text{ otherwise} \end{cases}$$

Then:

$$\begin{aligned}
\sum_{j=1}^{q} A_{jS} y_j &= \sum_{j=1}^{q} A_{jS} \cdot f_{P^1_{r_1(j)}} \cdot f_{P^2_{r_2(j)}} \\
&= \sum_{j=1}^{q} \left(B_{P^1_{r_1(j)},S} + B_{P^2_{r_2(j)},S} \right) \cdot f_{P^1_{r_1(j)}} \cdot f_{P^2_{r_2(j)}} \\
&= \sum_{l_1=1}^{q_1} \sum_{l_2=1}^{q_2} \left(B_{P^1_{l_1},S} + B_{P^2_{l_2},S} \right) \cdot f_{P^1_{l_1}} \cdot f_{P^2_{l_2}} \\
&\le \sum_{l_1=1}^{q_1} \sum_{l_2=1}^{q_2} \sum_{e \in S} \left(B_{P^1_{l_1},\{e\}} + B_{P^2_{l_2},\{e\}} \right) \cdot f_{P^1_{l_1}} \cdot f_{P^2_{l_2}} \\
&= \sum_{e \in S} \sum_{l_1=1}^{q_1} \sum_{l_2=1}^{q_2} \left(B_{P^1_{l_1},\{e\}} + B_{P^2_{l_2},\{e\}} \right) \cdot f_{P^1_{l_1}} \cdot f_{P^2_{l_2}} \\
&= \sum_{e \in S} \sum_{l_1=1}^{q_1} \sum_{l_2=1}^{q_2} B_{P^1_{l_1},\{e\}} \cdot f_{P^1_{l_1}} \cdot f_{P^2_{l_2}} \\
&\quad + \sum_{e \in S} \sum_{l_1=1}^{q_1} \sum_{l_2=1}^{q_2} B_{P^2_{l_2},\{e\}} \cdot f_{P^1_{l_1}} \cdot f_{P^2_{l_2}} \\
&= \sum_{e \in S} \sum_{l_1=1}^{q_1} \left(B_{P^1_{l_1},\{e\}} \cdot f_{P^1_{l_1}} \cdot \sum_{l_2=1}^{q_2} f_{P^2_{l_2}} \right) \\
&\quad + \sum_{e \in S} \sum_{l_2=1}^{q_2} \left(B_{P^2_{l_2},\{e\}} \cdot f_{P^2_{l_2}} \cdot \sum_{l_1=1}^{q_1} f_{P^1_{l_1}} \right) \\
&= \sum_{e \in S} \sum_{l_1=1}^{q_1} B_{P^1_{l_1},\{e\}} \cdot f_{P^1_{l_1}} + \sum_{e \in S} \sum_{l_2=1}^{q_2} B_{P^2_{l_2},\{e\}} \cdot f_{P^2_{l_2}} \\
&= \sum_{e \in S} \left(\sum_{l_1=1 \,|\, e \in P^1_{l_1}}^{q_1} f_{P^1_{l_1}} + \sum_{l_2=1 \,|\, e \in P^2_{l_2}}^{q_1} f_{P^2_{l_2}} \right) \\
&= \sum_{e \in S} \sum_{P \in \mathcal{P}} f_P \le \sum_{e \in S} \alpha \le c \cdot \alpha = \lambda
\end{aligned}$$

where we used Constraint 20 to replace $\sum_{l_2=1}^{q_2} f_{P^2_{l_2}}$ and $\sum_{l_2=1}^{q_2} f_{P^2_{l_2}}$ by 1, and Constraint 19 in the last line. ■

Also, in LP8, assign $x_S = 1/\binom{|C_1|}{c}$ to each $S \subseteq C_1$ with $|S| = c$, and $x_S = 0$ to all other $S \in \mathcal{T}$. Note that we made the assumption that c does not exceed any cut in G and therefore $\sum_{S \in \mathcal{T}} x_S = 1$, showing constraint 14 is satisfied. We also assign $w = c/|C_1|$. We proceed to show that this assignment is feasible for LP8; the fact that we found both equilibria follows from the fact that $w = \lambda$. Constraints 14 and 15 are immediate. We must verify constraints 13, so let

$j \in \{1, 2, \ldots, q\}$ be arbitrary, and let e_j be an edge of $C_1 \cap P^1_{r_1(j)}$ (such an edge must exist since C_1 is an $s_1 - t_1$ cut). Then indeed:

$$\sum_{S \in \mathcal{T}} A_{jS} x_S = \sum_{S \in \mathcal{T}} x_S |\{i \in \{1, 2\} | P^i_{r_i(j)} \bigcap S \neq \emptyset\}| \geq \sum_{S \in \mathcal{T}} x_S |\{e_j\} \cap S|$$

$$= \sum_{S \mid e_j \in S} \frac{1}{\binom{|C_1|}{c}} = \frac{|\{S \subseteq C_1 \mid e_j \in S\}|}{\binom{|C_1|}{c}} = \frac{\binom{|C_1|-1}{c-1}}{\binom{|C_1|}{c}} = c/|C_1| = w.$$

Case 2. In the second case, $|C_2| \leq |C_1|$ and $|C_2| \leq |C_3|/2$. This is symmetric to the first case.
Case 3. In the third case, $|C_3| < 2|C_1|$ and $|C_3| < 2|C_1|$. Then Hu's theorem and algorithm finds a feasible solution to LP10 with $\alpha = 2/|C_3|$. In LP9, we assign $\lambda = c \cdot \alpha$ and $y_j = f_{P^1_{r_1(j)}} \cdot f_{P^2_{r_2(j)}}$, just as in the first case.

Claim. λ and y_j as described above make a feasible solution to LP9.

Proof. Let $f^1(e_i) = \sum_{P^1_{r_1(j)} \ni e_i} f_{P^1_{r_1(j)}}$ and $f^2(e_i) = \sum_{P^2_{r_2(j)} \ni e_i} f_{P^2_{r_2(j)}}$. Let $S_1 = \{e \in S : f^1(e) > 0 \text{ and } f^2(e) > 0\}$, and let $k = |S_1|$.

$$\begin{aligned}
\sum_{j=1}^{q} A_{jS} y_j &= \sum_{j=1}^{q} A_{jS} f_{P^1_{r_1(j)}} f_{P^2_{r_2(j)}} \\
&= \sum_{e \in S_1} f^1(e)(1 - f^2(e)) + \sum_{e \in S_1} f^2(e)(1 - f^1(e)) \\
&\quad +2 \sum_{e \in S_1} f^1(e) f^2(e) + \sum_{e \in S \setminus S_1} f^1(e) \sum_{j=1}^{q_2} f_{P^2_{r_2(j)}} \\
&= \sum_{e \in S_1} (f^1(e) + f^2(e)) + \sum_{e \in S \setminus S_1} f^1(e) = 2k/|C_3| + 2(c-k)/|C_3| \\
&= 2c/|C_3| = \lambda
\end{aligned}$$

■

Also, in LP8, assign $x_S = 1/\binom{|C_3|}{c}$ to each $S \subseteq C_3$ with $|S| = c$, and $x_S = 0$ to all other $S \in \mathcal{T}$. Note that we made the assumption that c does not exceed any cut in G and therefore $\sum_{S \in \mathcal{T}} x_S = 1$, showing constraint 14 is satisfied. We also assign $w = 2c/|C_3|$. We proceed to show that this assignment is feasible for LP8; the fact that we found both equilibrium follows from the fact that $w = \lambda$. Constraints 14 and 15 are immediate. We must verify constraints 13, so let $j \in \{1, 2, \ldots, q\}$ be arbitrary. Let e^1_j be an edge of $C_3 \cap P^1_{r_1(j)}$ and e^2_j be an edge of $C_3 \cap P^1_{r_2(j)}$ (such edges must exist since C_3 is separating both pairs $s_1 - t_1$ and $s_2 - t_2$). It may be that $e^1_j = e^2_j$. Then indeed:

$$\begin{aligned}
\sum_{S \in \mathcal{T}} A_{jS} x_S &= \sum_{S \in \mathcal{T}} x_S |\{i \in \{1, 2\} | P^i_{r_i(j)} \bigcap S \neq \emptyset\}| \\
&\geq \sum_{S \in \mathcal{T}} x_S (|\{e^1_j\} \cap S| + |\{e^2_j\} \cap S|)
\end{aligned}$$

$$
\begin{aligned}
&= \sum_{S\in\mathcal{T}} x_S|\{e_j^1\}\cap S| + \sum_{S\in\mathcal{T}} x_S|\{e_j^2\}\cap S|) \\
&= \sum_{S\mid e_j^1\in S} \frac{1}{\binom{|C_3|}{c}} + \sum_{S\mid e_j^2\in S} \frac{1}{\binom{|C_3|}{c}} \\
&= \frac{\left|\{S\subseteq C_3 \mid e_j^1\in S\}\right|}{\binom{|C_3|}{c}} + \frac{\left|\{S\subseteq C_3 \mid e_j^2\in S\}\right|}{\binom{|C_3|}{c}} \\
&= 2\frac{\binom{|C_3|-1}{c-1}}{\binom{|C_3|}{c}} = 2c/|C_3| = w.
\end{aligned}
$$

In conclusion, we provided a polynomial-time algorithm for computing the Nash equilibrium in the game where there are two commodities and the attacker can attack any subset of c edges, provided c does not exceed the minimum cut of undirected graph G.

5 Budgeted Attacks: Costs on Edges

In this case, we consider the problem where $S-T$ consists of two commodities with sources s_1 and s_2, and sinks t_1 and t_2. The attacker may choose to attack any number of edges, and the designer chooses two paths, one for each source-sink pair. Let A be the payoff matrix. Let q_i denote the number of $s_i - t_i$ paths. A will have q rows, where $q = q_1 \cdot q_2$, and h columns, where $h = 2^{|E|}$ is the cardinality of the power set of E. Let P_j^i be the j^{th} path from s_i to t_i, for $i \in \{1,2\}$, where $1 \le j \le q_i$. Let r be some correspondence from $\{1,2,\ldots,q\}$ to $\{1,2,\ldots,q_1\}\times\{1,2,\ldots,q_2\}$, with $r_i(j)$ giving the i^{th} component of r $(1\le i\le 2)$. Let $\mathcal{T}$ denote the power set of E. The columns of A will be indexed by $S\in\mathcal{T}$. Thus: $A_{jS} = |\{i\in\{1,2\} \mid P^i_{r_i(j)} \cap S \neq \emptyset\}| - \sum_{e\in S} c(e)$. We seek a Nash equilibrium. To get one, the attacker should solve the following linear program:

LP11: Maximize w subject to

$$\textstyle\sum_{S\in\mathcal{T}} A_{jS}x_S \ge w \;\forall 1\le j\le q \quad (22)$$

$$\textstyle\sum_{S\in\mathcal{T}} x_S = 1 \quad (23)$$

$$x_S \ge 0 \qquad \forall S\in\mathcal{T}. \quad (24)$$

The above linear program shall be called LP11. The path designer must solve the dual of this linear program, denoted LP12, described below:

LP12: Minimize λ subject to

$$\textstyle\sum_{j=1}^q A_{jS}y_j \le \lambda \;\forall S\in\mathcal{T} \quad (25)$$

$$\textstyle\sum_{j=1}^q y_j = 1 \quad (26)$$

$$y_j \ge 0 \qquad \forall j\in\{1,2,\ldots,q\} \quad (27)$$

Our general plan for solving these programs is as follows. We compute four minimum costs cuts in the network $N=(V,E,c)$. The attacker will choose one of these cuts, or the empty set as his strategy; note that this is a pure strategy. This choice will be function of the costs of these cuts; also this choice gives us a feasible w in LP11. For the designer, we use Hu's [11,23] result to obtain certain flows matching the cuts and use the value of these flows to assign values (probabilities) to the variables y_j from LP12. Finally, we show that LP12 is feasible with $\lambda = w$ and these value for y_j; matching primal and dual objective functions implies we have optimum solutions for both LP11 and LP12.

We continue with the detailed description of this algorithm. Let $C_1 \subset E$ be the minimum cost cut separating s_1 from t_1 in N, let $C_2 \subset E$ be the minimum cost cut separating s_2 from t_2 in N, and let $C_3 \subset E$ be the minimum cost cut separating both $s_i - t_i$ pairs. In other words, in the graph $(V, E \setminus C_3)$ there exist neither $s_1 - t_1$ nor $s_2 - t_2$ paths; C_3 can be computed by computing two minimum cost cuts: one separating s_1, s_2 from t_1, t_2, and the other separating s_1, t_2 from t_1, s_2.

As usual, for $S \subseteq E$, we use $c(S) := \sum_{e \in S} c(e)$ with the usual convention that $c(\emptyset) = 0$. We assign $\lambda = w = \max(0 - c(\emptyset), 1 - c(C_1), 1 - c(C_2), 2 - c(C_3))$, and the attacker picks as his strategy the set of edges above achieving the maximum. We now verify that this is indeed a feasible solution for LP11. Constraints 23 and 24 are clearly satisfied. For Constraint 22, we must consider the four cases, depending on how w is selected. If $w = 0$, then, as for any j $A_{j\emptyset} = 0 - 0 = 0$, we see that 22 is satisfied. If $w = 1 - c(C_1)$, then we note that the strategy of the attacker, C_1, being a set of edges separating s_1 from t_1 in N, satisfies , for any designer strategy indexed by j, $A_{jC_1} \geq 1 - c(C_1)$ and this is the same as 22. The case $w = 1 - c(C_2)$ is symmetric to the previous case, while if $w = 2 - c(C_3)$, then we note that the strategy of the attacker, C_3, being a set of edges separating both s_1 from t_1 and s_2 from t_2 in N, satisfies, for any designer strategy indexed by j, $A_{jC_1} \geq 2 - c(C_3)$ and this is the same as 22. For simplicity of notation, let $c_1 = c(C_1)$, $c_2 = c(C_2)$, and $c_3 = c(C_3)$. For the designer, we use the following result of Hu [11](corollary 71.1b in [23]):

Theorem 5. *Let $G = (V, E)$ be a graph, let s_1, t_1 and s_2, t_2 be pairs of vertices of G, let $c : E \to \mathbb{R}_+$, and let $d_1, d_2 \in \mathbb{R}_+$. Then there exists a 2-commodity flow subject to c and with value d_1, d_2 if and only if the following "cut condition" is satisfied: for any $U \subset V$, we have $\sum_{i \mid |U \cap \{s_i, t_i\}| = 1} d_i \leq \sum_{e \in E \mid |U \cap e| = 1} c(e)$ (here, to make notation compact, we use an undirected edge e as a set of two vertices). Moreover, these two-commodity flows can be found in time polynomial in the size of G.*

When applying this theorem 5, we will choose $0 < d_1 \leq 1$ and $0 < d_2 \leq 1$ depending on how λ is selected. We will verify that the cut condition is satisfied, and get a two-commodity flow f^1 and f^2 as in the theorem. Then in all four cases, we proceed as follows.

We algorithmically decompose the $s_1 - t_1$ flow f^1 into at most $|E|$ path flows f^1_j for some paths $P^1_j \in \mathcal{P}^1$, and the $s_2 - t_2$ flow f^2 into at most $|E|$ path flows f^2_j for some paths $P^2_j \in \mathcal{P}^2$; we implicitly (for the purpose of the proof, and not as part of the algorithm) keep $f^1_j = 0$ if P^1_j is not one of the above paths, and $f^2_j = 0$ if P^2_j is not one of the above paths. Thus we have the flow value constraint for the first and second commodity and the capacity constraint as follows :

$$\forall i \in \{1,2\} : \sum_{j=1}^{q_i} f_j^i = d_i \tag{28}$$

$$\forall e \in E : \sum_{j \mid e \in P_j^1} f_j^1 + \sum_{j \mid e \in P_j^2} f_j^2 \leq c(e) \tag{29}$$

Recall that we have $q = q_1 \cdot q_2$ variables y_j in LP2, and r is a correspondence from $\{1, 2, \ldots, q\}$ to $\{1, 2, \ldots, q_1\} \times \{1, 2, \ldots, q_2\}$, with $r_i(j)$ giving the i^{th} component of r $(1 \leq i \leq 2)$. We set $y_j = \frac{1}{d_1 \cdot d_2} f^1_{r_1(j)} f^2_{r_2(j)}$; algorithmically this is done only for those at most $|E| \cdot |E|$ indices j for which both $f^1_{r_1(j)}$ and $f^2_{r_2(j)}$ are strictly positive.

It is immediate that Constraint 27 holds, and we prove that 26 holds below:

$$\begin{aligned}\sum_{j=1}^{q} y_j &= \frac{1}{d_1 \cdot d_2} \sum_{l_1=1}^{q_1} \sum_{l_2=1}^{q_2} f^1_{l_1} f^2_{l_2} = \frac{1}{d_1 \cdot d_2} \left(\sum_{l_1=1}^{q_1} f^1_{l_1} \right) \left(\sum_{l_2=1}^{q_2} f^2_{l_2} \right) \\ &= \frac{1}{d_1 \cdot d_2} (d_1 \cdot d_2) = 1\end{aligned}$$

where the third equality follows from Equations 28.

Also common to all four cases is the following inequality (whose long proof similar to Lemma 1 we omit), an upper bound on the left-hand side of Constraint 25:

$$\sum_{j=1}^{q} A_{jS} y_j \leq \left(\frac{1}{d_1} - 1 \right) d_1 + \left(\frac{1}{d_2} - 1 \right) d_2 = (1 - d_1) + (1 - d_2) \tag{30}$$

Only now we consider the four cases, function of how λ is selected. In the first case, $\lambda = 0$ and we choose $d_1 = d_2 = 1$; note that $0 < d_i \leq 1$. It is easy to verify then that the cut condition is satisfied, as $c_1 \geq 1$, $c_2 \geq 1$, and $c_3 \geq 2$ (this is how λ was selected). As for Constraint 25, from Equation 30 we deduce that indeed $\sum_{j=1}^{q} A_{jS} y_j \leq 0 = \lambda$.

In the second case, $\lambda = 1 - c_1$ and we choose $d_1 = c_1$ and $d_2 = 1$; note that $0 < d_i \leq 1$ since we made the assumption that $c(e) > 0$ for all e and there exists an $s_1 - t_1$ path in G. For the cut condition, notice that $d_1 \leq c_1$, and $d_2 \leq c_2$ follows from $1 - c_1 \geq 2 - c_3$ (this is how λ was selected) and $c_3 \leq c_1 + c_2$ (as $C_1 \cup C_2$ is a candidate for C_3). Also, $d_1 + d_2 = 1 + c_1 \leq c_3$ (this is how λ was selected). Therefore the cut condition is satisfied. As for Constraint 25, from Equation 30 we deduce that indeed $\sum_{j=1}^{q} A_{jS} y_j \leq 1 - c_1 = \lambda$.

The third case is symmetric with the second one. $\lambda = 1 - c_2$ and we choose $d_1 = 1$ and $d_2 = c_2$, and all the arguments from the second case hold.

In the fourth case, $\lambda = 2 - c_3$. We have two subcases. If $c_3/2 \leq c_1$ and $c_3/2 \leq c_2$, we use $d_1 = d_2 = c_3/2$. Note that $0 < c_3 \leq 2$ (from the selection of λ) so $0 < d_i \leq 1$. It is immediate to verify that the cut condition is satisfied.

As for Constraint 25, from Equation 30 we deduce that indeed $\sum_{j=1}^{q} A_{jS} y_j \leq (1 - c_3/2) + (1 - c_3/2) = 2 - c_3 = \lambda$.

The second case of the fourth case has $\lambda = 2 - c_3$. And either $c_3/2 > c_1$ or $c_3/2 > c_2$, and we assume by symmetry that $c_3/2 > c_1$. We use $d_1 = c_1$ and $d_2 = c_3 - c_1$. It is immediate that $d_1 > 0$, and $d_1 = c_1 < c_3/2 \leq 1$, since the choice of λ implies $2 - c_3 \geq 0$. Also, $c_3 - c_1 > c_3/2$ and therefore $d_2 > 0$. Also $d_2 = c_3 - c_1 \leq 1$ since the choice of λ implies $2 - c_3 \geq 1 - c_1$. The cut condition is satisfied since in the only non-trivial case, $d_2 = c_3 - c_1 \leq c_2$ since as argued above $c_3 \leq c_1 + c_2$ (as $C_1 \cup C_2$ is a candidate for C_3). As for Constraint 25, from Equation 30 we deduce that indeed $\sum_{j=1}^{q} A_{jS} y_j \leq (1 - c_1) + (1 - (c_3 - c_1)) = \lambda$.

We have checked that in all cases, we get in polynomial time a feasible solution to LP12. Moreover, we found in polynomial time in all four cases a feasible solution to LP11 of the same objective vlaue. Thus we have proven:

Theorem 6. *For two commodities, there exists a polynomial-time algorithm to find a mixed Nash equilibrium for the budgeted attacker/designer game.*

6 Conclusions

We note that our approach for two commodities works whenever an equivalent of Theorem 5 holds for the input graph. It is known that Theorem 5 does not hold for three commodities in undirected graphs, or two commodities in directed graphs.

Acknowledgements. This work was supported in part by NSF grant NeTS-0916743 (ARRA - NSF CNS-0916743).

References

1. Arora, S., Rao, S., Vazirani, U.: Expander flows, geometric embeddings and graph partitioning. J. ACM 56, 5:1–5:37 (2009)
2. Aumann, Y., Rabani, Y.: An $O(\log k)$ Approximate Min-Cut Max-Flow Theorem Approximation Algorithm. SIAM J. Comput. 27(1), 291–301 (1998)
3. Bell, M.G.H.: The measurement of reliability in stochastic transport networks. In: Proceedings of 2001 IEEE Intelligent Transportation Systems (2001)
4. Bohacek, S., Hespanha, J., Lee, J., Lim, C., Obraczka, K.: Game theoretic stochastic routing for fault tolerance and security in communication networks. IEEE/ACM Trans. on Parallel and Distributed Systems (2007)
5. Bohacek, S., Hespanha, J.P., Obraczka, K.: Saddle policies for secure routing in communication networks. In: Proc. of the 41st Conf. on Decision and Contr., pp. 1416–1421 (2002)
6. Bohacek, S., Hespanha, J.P., Obraczka, K., Lee, J., Lim, C.: Enhancing security via stochastic routing. In: Proceedings of the 11th IEEE International Conference on Computer Communications and Networks (2002)
7. Chen, J., Chan, S.-H.G., Li, V.O.K.: Multipath routing for video delivery over bandwidth-limited networks. IEEE Journal on Selected Areas in Communications 22(10), 1920–1932 (2004)

8. Chen, L., Leneutre, J.: On multipath routing in multihop wireless networks: security, performance, and their tradeoff. EURASIP J. Wirel. Commun. Netw. 2009, 6:1–6:13 (2009)
9. Elwalid, A., Jin, C., Low, S., Widjaja, I.: Mate: Mpls adaptive traffic engineering (2001)
10. Gueye, A., Walrand, J.C.: Security in networks: A game-theoretic approach. In: 47th IEEE Conference on Decision and Control, CDC 2008, pp. 829–834 (2008)
11. Hu, T.C.: Multi-commodity network flows. Operations Research 11, 344–360 (1963)
12. Lee, S.J., Gerla, M.: Split multipath routing with maximally disjoint paths in ad hoc networks (2001)
13. Klein, P., Rao, S., Agrawal, A., Ravi, R.: An approximate max-flow min-cut relation for undirected multicommodity flow, with applications. Combinatorica 15(2), 187–202 (1995)
14. Lee, P.P.C., Misra, V., Rubenstein, D.: Distributed algorithms for secure multipath routing. In: Proceedings of IEEE INFOCOM (2005)
15. Leighton, F., Rao, S.: Multicommodity max-flow min-cut theorems and their use in designing approximation algorithms. Journal of the ACM, 787–832 (1999)
16. Linial, N., London, E., Rabinovich, Y.: The geometry of graphs and some of its algorithmic applications. Combinatorica 15(2), 215–245 (1995)
17. Lou, W., Liu, W., Zhang, Y., Fang, Y.: Spread: Improving network security by multipath routing in mobile ad hoc networks. Wireless Networks 15, 279–294 (2009)
18. Mavronicolas, M., Papadopoulou, V., Philippou, A., Spirakis, P.: A Graph-Theoretic Network Security Game. In: Deng, X., Ye, Y. (eds.) WINE 2005. LNCS, vol. 3828, pp. 969–978. Springer, Heidelberg (2005)
19. Mavronicolas, M., Papadopoulou, V., Philippou, A., Spirakis, P.: A network game with attackers and a defender. Algorithmica 51, 315–341 (2008)
20. Mirrokni, V.S., Thottan, M., Uzunalioglu, H., Paul, S.: A simple polynomial time framework for reduced-path decomposition in multi-path routing. In: Proceedings of IEEE INFOCOM, pp. 183–194 (2004)
21. Papadimitratos, P., Haas, Z.J.: Secure message transmission in mobile ad hoc networks. Ad Hoc Networks Journal 1(1), 193–209 (2003)
22. Parissidis, G., Lenders, V., May, M., Plattner, B.: Multi-path routing protocols in wireless mobile ad hoc networks: A quantitative comparison
23. Schrijver, A.: Combinatorial Optimization. Springer, Heidelberg (2003)
24. Spafford, E.H.: Crisis and aftermath. Commun. ACM 32, 678–687 (1989)
25. Washburn, A., Wood, K.: Two-Person Zero-Sum Games for Network Interdiction. Operations Research 43(2), 243–251 (1995)
26. Lye, K.W., Wing, J.M.: Game strategies in network security (2002)