Preliminary Results in Routing Games*: Technical Report

João P. Hespanha† Stephan Bohacek‡
hespanha@usc.edu bohacek@math.usc.edu

†*Dept. Electrical Engineering–Systems, Univ. of Southern California*
3740 McClintock Ave., EEB 318, Los Angeles, CA 90089-2563

‡*Department of Mathematics, Univ. of Southern California*
1042 West 36th Place, DRB 155, Los Angeles, California 90089-1113

June 22, 2001†

Abstract

In this paper we determine routing polices for a data transmission network that are robust with respect to attempts of packet interception by an adversary. This problem is formulated as a zero-sum game between the designer of the routing algorithm and an adversary that attempts to intersect packets. We show that for some versions of the game, the optimal routing policies also maximize the throughput between the source to the destination node. In this paper we also list problems in this area that remain open.

1 Introduction

In modern networking, game theory has been used to investigate flow control [1], [2], allocation of link capacities [3], server allocation [4], the trade-off between delay and throughput along virtual circuits [5], [6], [7], competitive routing, where multiple users are sharing the network and each is trying to minimize the flow cost [8]. However, there has been little work on secure routing where packets are under threat of being corrupted or filtered by an attacker.

In this paper we are interested in determining routing policies for the network that are robust with respect to attempts of packet interception by an adversary. To this effect, we will formalize routing design as a game between two players: the designer of the routing algorithm and an adversary that attempts to intersect packets. This game is investigated under different rules and information structures. In some cases, the game reduces to the well known max flow problem for which there exists computationally efficient algorithms.

We consider a data transmission network with nodes $\mathcal{N} := \{1, 2, \dots, n\}$ connected by unidirectional links. We denote by $\mathcal{L}$ the set of all links and use the notation $\vec{ji}$ to represent a link from node j to node i. For a given link $\ell \in \mathcal{L}$ we denote by b_ℓ the link's bandwidth in packets per second[1] and by τ_ℓ the time it takes for a packet to transverse that link. We assume here that all the nodes in the network are connected in the sense that it is possible to reach any node from any other node through a finite sequence of links.

*This research was supported by the Defense Advanced Research Projects Agency. The views presented here are those of the authors and do not represent the views of the funding agencies.

†This report supersedes a previous report with the same title, dated October 2000.

[1] For simplicity, we assume here that all packets have the same length.

By a routing policy for a network it is usually meant an algorithm that determines which sequence of links

$$\{\vec{i_1 i_2}, \vec{i_2 i_3}, \dots, \vec{i_{k-1} i_k}\} \subset \mathcal{L}$$

should be used to direct (route) a packet from a source node $i_1 \in \mathcal{N}$ to a destination node $i_k \in \mathcal{N}$. Without loss of generality, we take the source and destinations nodes to be 1 and n, respectively. Because of this we do not need to consider links coming out of node n and for simplicity we will assume that no link in $\mathcal{L}$ exits node n. Most routing policies used in data transmission network have no memory in the sense that, when a packet arrives at some node $i \in \mathcal{N}$ with final destination $i_k \in \mathcal{N}$, the routing algorithms selects a path from i to i_k independently of where the packet is coming from. Here, we will not restrict our attention solely to this class of routing algorithms. In particular, some of the policies considered will take into account the node where the packets started (by convention node 1).

In the next section the max flow problem is posed in an uncommon setting. Then, different types of games are investigated. The first is an on-line game where the attacker has full information and is able to attack a new link at every step. This game is solved with dynamic programming. The second game is off-line, where the attacker must choose one link to attack at the beginning of the game. The routing algorithm is designed to minimize the effect of the attack. In some cases, these games reduces to the max flow problem of Section 2. Finally, the routing algorithm is adjusted in order to not only minimize the effect of the attack, but also reduce the delay (e.g. number of hops). This problem will reduce to the max flow with gain problem [9].

2 Maximum Flow

Before formalizing the routing problem we derive some basic results on the maximum number of packets per second that can flow in a network, without violating the bandwidth constraints.

2.1 Maximum Throughput

Suppose that we want to compute the maximum number of packets per second that can be transferred between two nodes in the network without drops. Without loss of generality, we take the source and destinations nodes to be 1 and n, respectively, and we denote by x_ℓ the number of packets per second that transverses link $\ell \in \mathcal{L}$. Each x_ℓ must be in the interval $[0, b_\ell]$.

In this section, we consider routing policies that distribute packets among several possible alternative paths according to a rule defined in terms of the percentage of packages sent through each path. Here, we take as given a routing policy that enforces that $r_{\vec{ik}}$ percent of all the packets arriving at node i are routed through link $\vec{ik} \in \mathcal{L}$. We ignore the quantization of data and assume that this percentage is exact at all times. Policies of this type are called *deterministic multi-path routing policies.* We denote by $\mathcal{R}_{\text{det}}$ the set of all such policies. As far as the traffic flow is concerned, each deterministic routing policy is characterized by the list $R := \{r_\ell : \ell \in \mathcal{L}\}$, where the r_ℓ satisfy

$$\sum_{k:\vec{ik}\in\mathcal{L}} r_{\vec{ik}} = 1, \qquad \forall i \in \mathcal{N}.$$

In the sequel, we actually equate $\mathcal{R}_{\text{det}}$ to the set of lists R with the above property. The maximum number of packets per second that can be transfered without drops by a deterministic multi-path routing policy $R := \{r_\ell : \ell \in \mathcal{L}\}$ is called the *maximum throughput of* R.

Assuming that μ packets per second are sent from node 1 to n without drops, we must have

$$x_{\vec{ik}} = r_{\vec{ik}} \Big(\delta_{i1}\mu + \sum_{j:\vec{ji}\in\mathcal{L}} x_{\vec{ji}} \Big), \qquad \vec{ik} \in \mathcal{L}, \tag{1}$$

where $\delta_{ij} := 1$ when $i = j$ and $\delta_{ij} := 0$ otherwise. Equation (1) can also be written as

$$x = A_R x + \mu\, c_R, \tag{2}$$

where $x := \{x_\ell : \ell \in \mathcal{L}\}$, A_R is an appropriately defined matrix, and c_R is an appropriately defined vector. The maximum throughput of R is then equal to

$$\mu_R := \max_{\mu} \max_{\substack{x_\ell \in [0, b_\ell]: \\ x = A_R x + \mu\, c_R}} \mu. \tag{3}$$

Remark 1. We say that a routing policy $R \in \mathcal{R}_{\mathrm{det}}$ is *cycle-free* if under this policy a packet will never pass through the same node twice. Formally, this means that there is no sequence of links

$$\mathcal{S} := \{\vec{i_1 i_2}, \vec{i_2 i_3}, \ldots, \vec{i_{k-1} i_k}, \vec{i_k i_1}\} \subset \mathcal{L},$$

with $r_\ell > 0$ for all $\ell \in \mathcal{S}$. For a cycle-free policy R, the matrix $I - A_R$ is nonsingular[2] and the input flow μ completely defines the traffic $x = \mu(I - A_R)^{-1} c_R$ at every link.

2.2 Maximum Bandwidth

The maximum throughput computed above depends on the routing policy R. One can also pose the question, what is the maximum throughput attainable by any routing policy. We call this the *maximum bandwidth* of the network between nodes 1 and n.

Assuming that μ packets per second are sent from node 1 to n without drops, we must have

$$\delta_{i1}\mu + \sum_{j:\vec{ji} \in \mathcal{L}} x_{\vec{ji}} = \sum_{j:\vec{ij} \in \mathcal{L}} x_{\vec{ij}}, \qquad i \in \mathcal{N} \setminus \{n\}. \tag{4}$$

Equation (4) can also be written as

$$Ax + \mu c = 0,$$

where $x := \{x_\ell : \ell \in \mathcal{L}\}$, A is an appropriately defined matrix, and c an appropriately defined vector. The following can then be proved:

Theorem 1 (Maximum Bandwidth). *For each deterministic multi-path routing policy $R \in \mathcal{R}_{\mathrm{det}}$, let μ_R be defined by* (3). *Then*

$$\max_{R \in \mathcal{R}_{\mathrm{det}}} \mu_R = \mu_{\max} := \max_{\mu} \max_{\substack{x_\ell \in [0, b_\ell]: \\ Ax + \mu c = 0}} \mu \tag{5}$$

and the maximum is attained at any[3] *policy R^* defined by*

$$r^*_{\vec{ik}} := \frac{x^*_{\vec{ik}}}{\sum_{j:\vec{ij} \in \mathcal{L}} x^*_{\vec{ij}}}, \qquad \vec{ik} \in \mathcal{L}, \tag{6}$$

where the $x^* := \{x^*_\ell : \ell \in \mathcal{L}\}$ maximizes (5). *Moreover, it is always possible to find a cycle-free optimal policy R^*.*

[2]When R is cycle-free $A_R^n = 0$ and therefore the series $\sum_{k=0}^{\infty} A_R^k$ converges and is equal to $(I - A_R)^{-1}$ [10].
[3]Since, in general, the x^* that maximize (5) are not unique, the bandwidth maximizing routing policy R^* is also not unique.

Proof of Theorem 1. First note that the policy R^* defined by (6) satisfies (1) with $\mu = \mu_{\max}$ and $x = x^*$, because $\mu_{\max}$ and x^* satisfy (4). This means that μ_{R^*} can achieve a throughput of at least $\mu_{\max}$ without drops and therefore

$$\max_{R \in \mathcal{R}_{\text{det}}} \mu_R \geq \mu_{R^*} \geq \mu_{\max}.$$

By contradiction assume that the above inequality is strict and therefore that there exists a policy $R \in \mathcal{R}_{\text{det}}$ such that $\mu_R > \mu_{\max}$. This policy satisfies (1) for appropriate x and a μ larger than $\mu_{\max}$. Adding (1) over k leads to (4) so we conclude that (4) holds for the same x but a μ larger than $\mu_{\max}$. This contradict the maximality of $\mu_{\max}$.

It is shown in [9] that a flow can be uniquely decomposed into flows along elementary cycles and a cycle-free flow. The flow along the cycles has no effect on the cost and can be taken to be zero. Hence, the optimal flow can be taken to have no cycles. ∎

Remark 2. Equation (4) can be generalized to multiple flows, i.e., sources of traffic. Assuming there are m flows $\{\mu_k : k \in \mathcal{M}\}$, $\mathcal{M} := \{1, 2, \ldots, m\}$, with the flow μ_k from node 1_k to node n_k, equations (4) becomes

$$\delta_{i1_k}\mu_k + \sum_{j:\vec{ji} \in \mathcal{L}} x_{\vec{ji}}^{(k)} = \sum_{j:\vec{ij} \in \mathcal{L}} x_{\vec{ij}}^{(k)}, \qquad i \in \mathcal{N} \setminus \{n\},\ k \in \{1, 2, \ldots, m\}.$$

In this case, the bandwidth constraints are

$$\sum_{k \in \mathcal{M}} x_\ell^{(k)} \in [0, b_\ell], \qquad \ell \in \mathcal{L}.$$

3 Routing Games

Take the routing problem as a game between the network designer that specifies the routing algorithm and an adversary that attempt to intercept data in the network. We consider here a zero-sum game in which the designer wants to minimize the time it takes for a packet to be sent from node 1 to node n, and the adversary wants to maximize this time. To accomplish this, the adversary attempts to intercept the packet at particular links in the network. For short we say that the adversary *scans* link $\ell \in \mathcal{L}$ when she attempts to intercept the packet at that link. Several versions of the game are possible depending on how the game is defined:

Offline v.s. online The adversary can choose the link to be scanned only once before the packet is sent out of node 1, or she can select a new link every time the packet arrives at a new node. In the latter case, we assume that she knows where the packet is when she makes her selection.

Effectiveness In case the adversary scans a link over which the packet is traveling, the probability that she will intercept the packet maybe smaller than 1, i.e., she may not produce any effect on the routing of the packet with some nonzero probability. This probability may be fixed or it may vary from link to link.

Delay v.s. resend In case the adversary intercepts the packet, it may just suffer an additional delay and proceed from the same node after that, or it can be sent back to node 1 after some delay (timeout).

In this paper we assume that the interception is done at the links but similar results can be derived when the interception occurs at the nodes.

3.1 Online Game

We start by considering an on-line game in which the adversary selects a new link to be scanned every time the packet arrives at a new node and makes the selection knowing where the packet is. For generality, we take the probability of intercepting a packet to be link dependent and denote by p_ℓ the probability of intercepting a packet traveling in link $\ell \in \mathcal{L}$, given that link ℓ is being scanned by the adversary. Here, we will assume that there are no drops.

We start by considering the case in which intercepting a packet simply results in a fixed[4] extra delay T. The routing of the packet over the network can then be regarded as a stationary Markov chain whose state $\mathbf{q}_t \in \mathcal{N}$ is a random variable denoting the node where the packet is before the hop $t \in \{1, 2, \dots\}$. Denoting by $\mathbf{a}_t \in \mathcal{L}$ the next link as determined by the routing algorithms and by $\mathbf{b}_t \in \mathcal{L}$ the link scanned by the adversary, we have the following transition probability function for the Markov chain

$$\mathrm{P}(\mathbf{q}_{t+1} = q' \mid \mathbf{q}_t = q,\ \mathbf{a}_t = \vec{qa},\ \mathbf{b}_t = \ell) = \delta_{q'a}, \tag{7}$$

$q \in \mathcal{N} \setminus \{n\}$, $q' \in \mathcal{N}$, $\vec{qa}, \ell \in \mathcal{L}$, $t \in \{1, 2, \dots\}$. The state n is an absorbing state, i.e.,

$$\mathrm{P}(\mathbf{q}_{t+1} = q' \mid \mathbf{q}_t = n,\ \mathbf{a}_t = \ell_1,\ \mathbf{b}_t = \ell_2) = \delta_{q'n},$$

$q' \in \mathcal{N}$, $\ell_1, \ell_2 \in \mathcal{L}$, $t \in \{1, 2, \dots\}$. The cost to be optimized is the average time it takes to send the package from node 1 to node n and can be written as

$$J = \mathrm{E}\Big[\sum_{t=1}^{\infty} l(\mathbf{q}_t, \mathbf{a}_t, \mathbf{b}_t)\Big],$$

where

$$l(q, a, b) = \begin{cases} 0 & q = n \\ \tau_a + p_a T \delta_{ab} & q \neq n \end{cases}$$

To optimize this cost, for each node $i \in \mathcal{N} \setminus \{n\}$, the player that designs the routing chooses the distribution $a(i) := \{a_k : \vec{ik} \in \mathcal{L}\}$ of links to route the packet out of node i and the adversary chooses the distribution $b(i) := \{b_\ell : \ell \in \mathcal{L}\}$ of links to be scanned.

The two-person zero-sum game just defined falls in the class of stochastic shortest path games considered in [11]. In particular, it satisfies the following SSP assumptions:

SSP 1 There exists at least one proper policy for the minimizer, i.e., a policy that will lead to a finite cost regardless of the policy chosen by the maximizer. This is true because we assume that there exists a sequence of link that connects node 1 to node n. Note that even considering the version of the game in which an intercepted packet is sent back to node 1, this assumption holds provided that $p_\ell < 1$ for every $\ell \in \mathcal{L}$, because the packet will eventually reach the destination with probability one.

SSP 2 For any policies for which there is a non-zero probability that the packet will not reach the destination node, the corresponding cost is infinite. This is true provided that $\tau_\ell > 0$ for every $\ell \in \mathcal{L}$, because all links that do not reach node n will contribute to the final cost with a positive marginal cost.

To find a saddle solution to the game defined above, let us denote by V_i, $i \in \mathcal{N}$, the *cost-to-go from node* i, defined to be the average time it takes to send a packet from node i to node n

$$V_i = \mathrm{E}\Big[\sum_{t=1}^{\infty} l(\mathbf{q}_t, \mathbf{a}_t, \mathbf{b}_t) \,\Big|\, \mathbf{q}_1 = i\Big], \tag{8}$$

[4] This could be easily generalized for link-dependent extra delays.

using optimal routing policies for each player. Because we are dealing with a stationary Markov chain and an infinite horizon cost, starting the summation at time $k = 1$ is completely arbitrary. Because of the way the cost was defined,

$$V_n = 0.$$

Suppose now that a packet just arrived at node $i \in \mathcal{N} \setminus \{n\}$, the designer decides to route it through the link $\vec{ik} \in \mathcal{L}$ and the adversary decides to scan link $\ell \in \mathcal{L}$. For these particular choices, the average cost will be

$$\tau_{\vec{ik}} + p_{\vec{ik}} T \, \delta_{\ell \, \vec{ik}} + V_k. \tag{9}$$

Denoting by $a(i) := \{a_k : \vec{ik} \in \mathcal{L}\}$ the distribution of links with which the designer decides to route the packet out of node i and by $b(i) := \{b_\ell : \ell \in \mathcal{L}\}$ the distribution of the links to be scanned, the average cost will then be

$$a(i)' M_i[V] b(i)$$

where $M_i[V]$ is a matrix defined by

$$M_i[V] := \Big[\tau_{\vec{ik}} + p_{\vec{ik}} \, T \, \delta_{\ell \, \vec{ik}} + V_k\Big]_{k,\ell}. \tag{10}$$

Consider now the operators $T_{\min\max}$ and $T_{\max\min}$ that transform a set of costs-to-go $\bar{V} := \{\bar{V}_i : i \in \mathcal{N}\}$ into $V' := \{V_i' : i \in \mathcal{N}\}$ and $V'' := \{V_i'' : i \in \mathcal{N}\}$, respectively, with

$$V_i' = \inf_a \sup_b a' M_i[\bar{V}] b, \quad i \in \mathcal{N} \setminus \{n\}, \qquad V_n' = 0 \tag{11}$$

$$V_i'' = \sup_b \inf_a a' M_i[\bar{V}] b, \quad i \in \mathcal{N} \setminus \{n\}, \qquad V_n'' = 0. \tag{12}$$

These operators satisfy the regularity assumptions in [11]. In particular,

R 1 The sets over which the controls $a(i), b(i)$, $i \in \mathcal{N} \setminus \{n\}$, take values are compact.

R 2 The maps $\{a(i), b(i)\} \mapsto a' M_i[V] b$ are continuous.

R 3-4 The infima and suprema in (11)–(12) are actually minima and maxima and the two operators $T_{\min\max}$ and $T_{\max\min}$ are equal [12, Minimax Theorem].

The following Theorem is then a consequence of the results in [11].

Theorem 2. *The operator $T_{\min\max}$ (which is the same as $T_{\max\min}$) has a unique fixed point $V^* := \{V_i^* : i \in \mathcal{N}\}$ and, for any $\bar{V} := \{\bar{V}_i : i \in \mathcal{N}\}$,*

$$V^* = \lim_{k \to \infty} T_{\min\max}^k \bar{V}.$$

This fixed point V^* is equal to the cost-to-go $V := \{V_i : i \in \mathcal{N}\}$ corresponding to a pair of saddle policies for the game. Moreover, any pair of policies for which the outer inf *and* sup *in* (11)–(12) *are achieved, forms a saddle solution for the game.*

Figure 1 show the routing policies obtained using this algorithm for a test network topology. All links in this topology are bidirectional. This figure shows the policies for all $p_\ell = .5$, $\ell \in \mathcal{L}$, and three distinct values of the delay parameter T: 0, 9, and 100. For $T = 0$ we get the policy that minimizes the number of hops.

When we consider the case in which an intercepted packet is sent back to node 1 after a delay T, the transition probability for the Markov chain in (7) becomes

$$\mathrm{P}(\mathbf{q}_{t+1} = q' \mid \mathbf{q}_t = q, \; \mathbf{a}_t = \vec{qa}, \; \mathbf{b}_t = \ell) = \begin{cases} \delta_{q'a} & \ell \neq \vec{qa} \\ p_\ell \delta_{q'1} + (1 - p_\ell)\delta_{q'a} & \ell = \vec{qa} \end{cases}$$

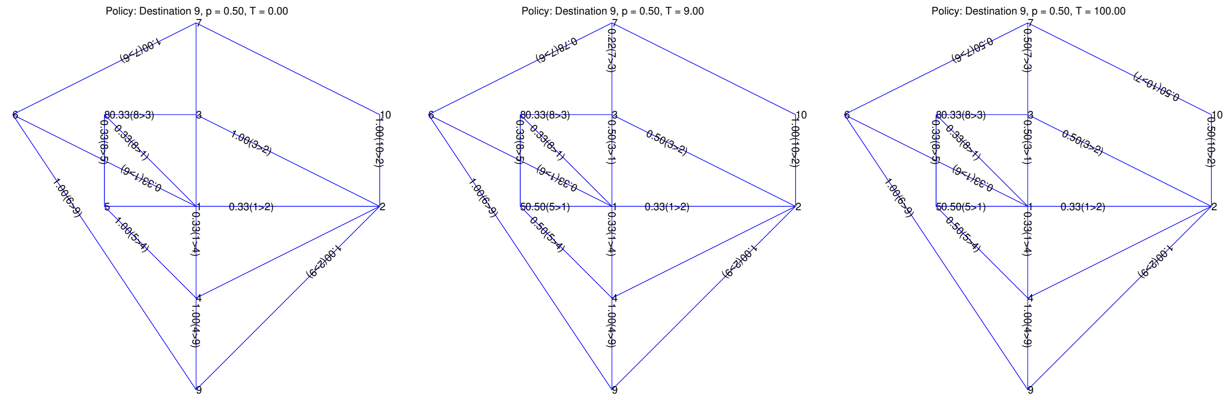

Figure 1: Routing policies for three values of T: 0 (left), 9 (middle), and (100) right. The destination node is always node 9. The label superimposed in each link is of the form $r_{\vec{ik}}(i > k)$. The lack of label on a link means that no packets will be routed through that link.

The cost in (9) must then be replaced by

$$\begin{cases} p_{\vec{ik}}(T + V_1) + (1 - p_{\vec{ik}})(\tau_{\vec{ik}} + V_k) & \vec{ik} = \ell \\ \tau_{\vec{ik}} + V_k & \vec{ik} \neq \ell \end{cases} \qquad = \qquad \tau_{\vec{ik}} + V_k + p_{\vec{ik}}(T - \tau_{\vec{ik}} + V_1 - V_k)\delta_{\ell\,\vec{ik}}$$

and therefore the matrix $M_i[V]$ must be replaced by

$$M_i[V] := \left[\tau_{\vec{ik}} + V_k + p_{\vec{ik}}(T - \tau_{\vec{ik}} + V_1 - V_k)\delta_{\ell\,\vec{ik}}\right]_{k,\ell}.$$

However, all the assumptions mentioned before still hold and Theorem 2 is also true for this version of the game.

Remark 3. When the intercepted packet is sent back to the source node, the cost-to-go V_i (and therefore the optimal routing policy) depends on the source node (chosen to be 1 in this derivation). This does not happen when interception just introduces an extra delay.

3.2 Off-line Game

We consider now an offline game in which the adversary selects which link to be scanned before routing starts, but the player responsible for designing the routing policy does not know which link was selected.

This problem can be solved by converting it into its extensive form by building a matrix M with one row for each possible path from node 1 to node n and one column for each possible link that the adversary can scan. If one allows for circular paths (as could in principle happen with stochastic routing) the matrix may have an infinite (but countable) number of rows. This method to compute the optimal solution does not scale well because potentially the number of distinct paths grows exponentially with the number of links.

Another method to solve this problem consists in converting it into a game in a Markov chain as done in the previous section. For the Markov chain to "memorize" the initial decision of the adversary, its state must be a pair $(\mathbf{q}_t, \mathbf{s}_t)$ where $\mathbf{q}_t$ denotes the node where the packet is before the tth hop and $\mathbf{s}_t$ the link being scanned by the adversary. In this game $\mathbf{s}_t$ remains constant after $t > 1$, since we assume the adversary cannot change is mind after the initial choice. The resulting game is then played in a partially observable Markov chain because the player that designs the routing cannot measure the component $\mathbf{s}_t$ of the state. However, we can assume that it is a nested information game because the adversary can have full state information (of course there is nothing she can do with this information) and therefore a solution to this game that avoids the combinatorial explosion by making use of dynamic programming seems possible. Unfortunately, even with nested information, solving this game seems computationally hard because of partial information.

In search for solutions to the game that are computationally more attractive, we start by considering a simpler version of the game in which the cost that defines the zero-sum game is simply

$$J = \mathrm{E}[\chi] = \mathrm{P}(\chi = 1),$$

where χ is a random variable that is equal to 1 if the packet is intercepted and 0 otherwise. This cost assumes that all paths are equal and therefore all that matters is to make sure that the packet reaches its destination. This can be viewed as the limiting case when a packet being intercepted means that it will suffer a fixed extra delay T that is much larger than any of the delays τ_ℓ, $\ell \in \mathcal{L}$, incurred when the packet is not caught.

We consider here *stochastic routing policies* and denote by $\mathcal{R}_{\text{sto}}$ this class of policies. Under such policies, whenever a packet arrives at node $k \in \mathcal{N}$, it will be routed through link $\vec{ki} \in \mathcal{L}$ with probability $r_{\vec{ki}}$. As far as the routing is concerned, each routing policy is characterized by the list $R := \{r_\ell : \ell \in \mathcal{L}\}$, where the r_ℓ satisfy

$$\sum_{k:\vec{ik}\in\mathcal{L}} r_{\vec{ik}} = 1, \qquad \forall i \in \mathcal{N}.$$

In the sequel we equate $\mathcal{R}_{\text{sto}}$ to the set of lists R with the above property.

Let us consider a fixed routing policy $R := \{r_\ell : \ell \in \mathcal{L}\} \in \mathcal{R}_{\text{sto}}$ and a choice of link $\ell \in \mathcal{L}$ to be scanned by the adversary. The corresponding cost $J_{R\ell}$ is then given by

$$J_{R\ell} = \mathrm{P}_{R\ell}(\chi = 1),$$

where the subscript $_{R\ell}$ in the probability measure emphasizes its dependency on the choices of both players. Since all we are interested in is determining if the packet is caught (in which case $\chi = 1$), without loss of generality we can assume that, once a packet is caught it will not be routed anymore. Under this assumption, denoting by $x_{\bar{\ell}}(t)$ the probability that the packet will be sent to link $\bar{\ell} \in \mathcal{L}$ for the hop $t \in \{1, 2, \dots\}$, we have that

$$x_{\vec{ik}}(1) = r_{\vec{ik}}\delta_{i1}, \qquad x_{\vec{ik}}(t+1) = \sum_{j:\vec{ji}\in\mathcal{L}} r_{\vec{ik}}(1 - p_\ell \delta_{\ell\,\vec{ji}}) x_{\vec{ji}}(t), \quad t > 1, \qquad \vec{ik} \in \mathcal{L}. \tag{13}$$

Using the fact that the conditional probability that a packet is caught in the tth hop, given that it was sent to link ℓ in that hop, is equal to p_ℓ, we can then write the cost as

$$J_{R\ell} = \sum_{t=1}^{\infty} p_\ell x_\ell(t).$$

Equation (13) can also be written as

$$x(1) = c_R, \qquad x(t+1) = A_R(I - P_\ell)x(t), \quad t > 1, \tag{14}$$

where $x(t) := \{x_\ell(t) : \ell \in \mathcal{L}\}$, $t \geq 1$, P_ℓ is a matrix whose entries are all zero except for the ℓth diagonal element that is equal to p_ℓ, and A_R, c_R as in Section 2.1. The probability of the packet being intersected can then be written as

$$J_{R\ell} = \mathbf{1}P_\ell \sum_{t=1}^{\infty} x(t) = \mathbf{1}P_\ell \sum_{t=1}^{\infty} \big(A_R(I - P_\ell)\big)^{t-1} c_R, \tag{15}$$

where $\mathbf{1}$ denotes a row vector with all entries equal to 1.

Suppose now that the adversary scans each link $\ell \in \mathcal{L}$, with probability d_ℓ and let $D := \{d_\ell : \ell \in \mathcal{L}\}$. We then have

$$J_{RD} = \sum_{\ell \in \mathcal{L}} d_\ell J_{R\ell} = \sum_{\ell \in \mathcal{L}} d_\ell \mathbf{1} P_\ell \sum_{t=1}^{\infty} \big(A_R(I-P_\ell)\big)^{t-1} c_R.$$

A saddle solution for this zero-sum game exists in case

$$\min_{R \in \mathcal{R}_{\text{sto}}} \max_{D=\{d_\ell\}} \sum_{\ell \in \mathcal{L}} d_\ell \mathbf{1} P_\ell \sum_{t=1}^{\infty} \big(A_R(I-P_\ell)\big)^{t-1} c_R = \max_{D=\{d_\ell\}} \min_{R \in \mathcal{R}_{\text{sto}}} \sum_{\ell \in \mathcal{L}} d_\ell \mathbf{1} P_\ell \sum_{t=1}^{\infty} \big(A_R(I-P_\ell)\big)^{t-1} c_R.$$

Since the cost is not bilinear on the policies, it is not clear if a saddle solution does exist[5]. We conjecture that it does but leave this issue for a future paper. Here, we will seek for a security policy R^* for the player that designs the routing policy. The policy R^* policy is defined by

$$J^* := \min_{R \in \mathcal{R}_{\text{sto}}} \max_{D=\{d_\ell\}} \sum_{\ell \in \mathcal{L}} d_\ell \mathbf{1} P_\ell \sum_{t=1}^{\infty} \big(A_R(I-P_\ell)\big)^{t-1} c_R = \max_{D=\{d_\ell\}} \sum_{\ell \in \mathcal{L}} d_\ell \mathbf{1} P_\ell \sum_{t=1}^{\infty} \big(A_{R^*}(I-P_\ell)\big)^{t-1} c_{R^*}\ . \tag{16}$$

3.2.1 Cycle-free Routing

To solve (16), we restricting ourselves to *stochastic cycle-free routing policies.* These are stochastic routing policies for which $R := \{r_\ell : \ell \in \mathcal{L}\}$ is chosen so that a packet will not return to a node where it has been before with probability one. Formally, this means that there is no sequence of links $\mathcal{S} := \{\overrightarrow{i_1 i_2}, \overrightarrow{i_2 i_3}, \dots, \overrightarrow{i_{k-1} i_k}, \overrightarrow{i_k i_1}\} \subset \mathcal{L}$, with $r_\ell > 0$ for all $\ell \in \mathcal{S}$. We denote by $\mathcal{R}_{\text{nocycle}}$ the set of lists $R \in \mathcal{R}_{\text{sto}}$ with the above property[6]. The following basic property of cycle-free policies is proved in the appendix.

Lemma 1. *For any cycle-free routing policy $R := \{r_\ell : \ell \in \mathcal{L}\} \in \mathcal{R}_{\text{nocycle}}$ and any $\ell \in \mathcal{L}$,*

$$P_\ell \sum_{t=1}^{\infty} \big(A_R(I-P_\ell)\big)^{t-1} = P_\ell \sum_{t=1}^{\infty} A_R^{t-1} = P_\ell (I-A_R)^{-1}. \tag{17}$$

Once we restrict ourselves to stochastic cycle-free routing policies, the security policy R^* for the player that designs the routing policy can be defined by

$$J^* := \min_{R \in \mathcal{R}_{\text{nocycle}}} \max_{D=\{d_\ell\}} \sum_{\ell \in \mathcal{L}} d_\ell \mathbf{1} P_\ell x_R = \max_{D=\{d_\ell\}} \sum_{\ell \in \mathcal{L}} d_\ell \mathbf{1} P_\ell x_{R^*},$$

where $x_R := (I-A_R)^{-1} c_R$ and therefore x_R is the unique solution to

$$x_R = A_R x_R + c_R. \tag{18}$$

Because of linearity we have that

$$J^* = \min_{R \in \mathcal{R}_{\text{nocycle}}} \max_{\ell \in \mathcal{L}} \mathbf{1} P_\ell x_R.$$

Moreover, since $\max\{c_1, c_2\} = \min_{c_1 \le \mu,\ c_2 \le \mu} \mu$ and all the x_ℓ are nonnegative, we also have

$$J^* = \min_{R \in \mathcal{R}_{\text{nocycle}}} \min_{\substack{\mathbf{1} P_\ell x_R \in [0,\mu]: \\ x_R = A_R x_R + c_R}} \mu.$$

[5]Actually, since the cost is linear on D for fixed R, to apply the Minimax Theorem [13] it would be sufficient that the cost be convex on R for fixed D. However, it is unclear if this is the case.

[6]This set turns out to be nonconvex.

Suppose now that we make the change of variables $\bar{\mu} := \frac{b}{\mu}$ and $\bar{x}_R := \bar{\mu} x_R$ in the above optimization, for some constant $b > 0$. This yields

$$J^* = \min_{R \in \mathcal{R}_{\text{nocycle}}} \min_{\substack{\mathbf{1} P_\ell \bar{x}_R \in [0,b]: \\ \bar{x}_R = A_R \bar{x}_R + \bar{\mu} c_R}} \frac{b}{\bar{\mu}},$$

and therefore

$$\frac{b}{J^*} = \max_{R \in \mathcal{R}_{\text{nocycle}}} \max_{\substack{\mathbf{1} P_\ell \bar{x}_R \in [0,b]: \\ \bar{x}_R = A_R \bar{x}_R + \bar{\mu} c_R}} \bar{\mu} = \max_{R \in \mathcal{R}_{\text{nocycle}}} \mu_R,$$

where μ_R denotes the maximum throughput defined by (3), when all links have bandwidths b_ℓ, $\ell \in \mathcal{L}$, equal to b. Security policies for this game can then be computed using the linear program defined in Theorem 1, provided that one selects an optimal policy R^* that is cycle-free. Note that the optimal policy R^* in Theorem 1 should now be interpreted as a stochastic cycle-free routing policy.

3.2.2 Bias Towards Short Paths

In the previous sections we considered a cost $J = \mathrm{E}[\boldsymbol{\chi}]$ that ignores the length of the path taken. We show here how this cost can be modified to bias the routing towards short paths and still keeping the computations simple. The cost proposed is defined by

$$J_\epsilon := \mathrm{E}[\boldsymbol{\chi}_\epsilon],$$

where $\boldsymbol{\chi}_\epsilon$, $\epsilon \geq 0$, is a random variable that is equal to $(1+\epsilon)^{t-1}$ if the packet is intercepted at the tth hop and 0 otherwise. For $\epsilon = 0$, this random variable degenerates in the random variable $\boldsymbol{\chi}$ defined before. This new cost function bias the solution sought by the player that designs the routing policy towards shorter paths since, when being caught is inevitable, it incurs in less cost if it is caught sooner than later. In fact, as $\epsilon \to \infty$, the burden of an extra hop is so large that the optimal solution will minimize the number of hops.

It turns out that solving the game for the new cost J_ϵ is conceptually the same as solving the game for the previous cost J. To see why this is so, let us expand J_ϵ as

$$J_\epsilon = \sum_{t=1}^{\infty} p_\ell (1+\epsilon)^{t-1} x_\ell(t) = \sum \mathbf{1} P_\ell \sum_{t=1}^{\infty} \big((1+\epsilon) A_R (I - P_\ell)\big)^{t-1} c.$$

Here we have used (14). This means that this new game can be solved similarly to the previous ones, provided that we replace A_R by $(1+\epsilon)A_R$.

Remark 4. For the case of no cycles, the solution to this game corresponds to a flow constrain of the form

$$x = (1+\epsilon) A_R x + \mu\, c_R,$$

instead of (2). This can be view as a multiplicative flow amplification of $(1+\epsilon)$ at each node and it will make input flows that use many hops smaller because they are more amplified as they travel to the destination node. This means that the matrix A and the vector c in (5) must now be defined by

$$\delta_{i1} \mu + (1+\epsilon) \sum_{j: \vec{ji} \in \mathcal{L}} x_{\vec{ji}} = \sum_{j: \vec{ij} \in \mathcal{L}} x_{\vec{ij}}, \qquad i \in \mathcal{N} \setminus \{n\}.$$

Figure 2 show the routing policies obtained using this algorithm for a test network topology. All links in this topology are bidirectional. This figure shows the policies for three values of the ϵ parameter: 0, 1, and 500. For $\epsilon = 0$ we get the policy that maximizes the flow from source to destination, whereas for $\epsilon = 500$ we basically get the policy that minimizes the number of hops. For $\epsilon = 1$ we get a compromise solution.

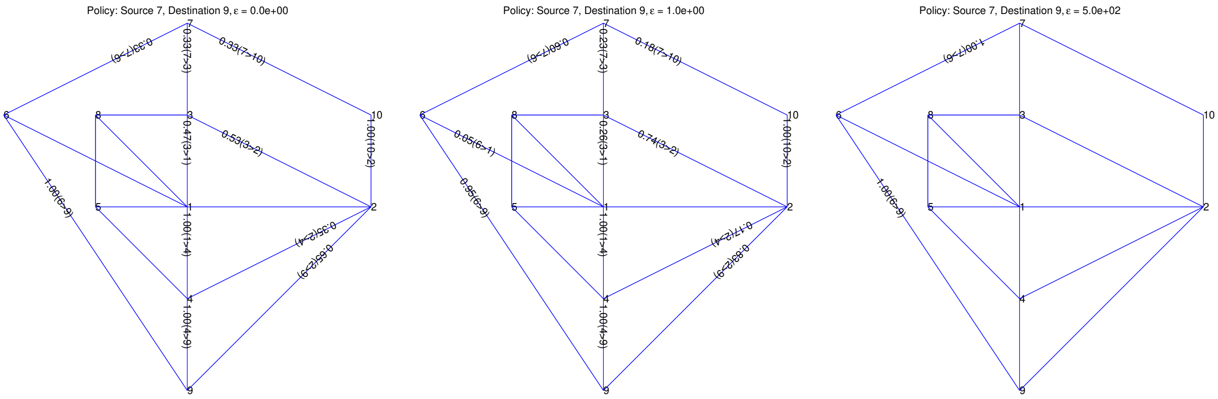

Figure 2: Routing policies for three values of ϵ: 0 (left), 1 (middle), and (500) right. The source and destination nodes are 7 and 9, respectively. The label superimposed in each link is of the form $r_{\vec{i}\vec{k}}(i > k)$. The lack of label on a link means that no packets will be routed through that link.

4 Conclusions

In this paper we determined routing polices for a data transmission network that are robust with respect to attempts of packet interception by an adversary. This problem is formulated as a zero-sum game between the designer of the routing algorithm and an adversary that attempts to intersect packets. We show that for some versions of the game, the optimal routing policies also maximize the bandwidth between the source to the destination node.

Several problems remain open. For online games, numerical computations seem to indicate that policy iteration seems to converge to a fixed point in a finite number of steps. We have also observed that the optimal policies computed numerically are cycle-free. We are now investigating if these are general properties of these games. This seems to be so, at least, for the case when interception just introduces an extra delay. For offline game, we have so far only computed security policies that are worst case solutions for the player that designs the routing policies. We believe that the solutions that we derived are actually saddle solutions but that remains to be proved. Finally, so far we restricted our attention to cycle-free policies for the offline game. It remains to investigate if policies with cycles can yield more favorable costs.

Appendix

Proof of Lemma 1. To verify the first equality in (17) we prove by induction that

$$P_\ell\big(A_R(I - P_\ell)\big)^k = P_\ell A_R^k, \qquad k \geq 0. \tag{19}$$

The above equality is trivial for $k = 0$. Suppose now that it holds for some $k \geq 0$ and therefore

$$P_\ell\big(A_R(I - P_\ell)\big)^{k+1} = P_\ell\big(A_R(I - P_\ell)\big)^k\big(A_R(I - P_\ell)\big) P_\ell A_R^{k+1}(I - P_\ell). \tag{20}$$

When the network has not loops

$$P_\ell A_R^{k+1} P_\ell = 0 \tag{21}$$

because the ℓth link is not connected to itself through $k + 1$ hops. From (20)–(21) we conclude that

$$P_\ell\big(A_R(I - P_\ell)\big)^{k+1} = P_\ell A_R^{k+1},$$

which conclude the induction step. See footnote 2 in page 3 regarding the second equality in (17). ■

References

[1] E. Altman, "Fow control using the theory of zero-sum markov games," *IEEE Transaction on Automatic Control*, vol. 39, pp. 814–818, 1994.

[2] R. Maheswaran and T. Basar, "Multi-user flow control as a nash game: Performance of various algorithms," in *Proc. 37th IEEE Conference on Decision and Control*, (Tampa, Florida), pp. 1090–1095, 1998.

[3] Y. A. Korilis, A. A. Lazar, and A. Orda, "Capacity allocation under noncooperative routing," *IEEE Transactions on Automatic Control*, vol. 42, no. 3, pp. 309–325, 1997.

[4] S. Olafsson, "Resource allocation as an evolving strategy," *Evolutionary Computation*, vol. 4, no. 1, pp. 33–55, 1996.

[5] K. Bharath-Kumar and J. M. Jaffe, "A new approach to preformance-oriented flow control," *IEEE Transactions on Communications*, vol. 29, pp. 427–435, 1981.

[6] A. Orda, N. Rom, and N. Shimkin, "Competitive routing in multi-user communication networks," *IEEE/ACM Transaction on Networking*, vol. 1, pp. 614–627, 1993.

[7] S. Shenker, "Making greed work in networks: A game-theoretic analysis of switch service disciplines," in *SIGCOMM Symposium on Communications Architectures and Protocols*, (London, UK), pp. 47–57, 1994.

[8] E. Altman, T. Basar, T. Jimenez, and N. Shimkin, "Competitive routing in networks with polynomial costs," in *IEEE INFOCOM 2000*, (Tel-Aviv, Israel), 2000.

[9] M. Gondran and M. Minoux, *Graphs and Algorithms*. New York: John Wiley, 1979.

[10] R. A. Horn and C. R. Johnson, *Matrix Analysis*. Cambridge: Cambridge University Press., 1993.

[11] S. D. Patek and D. P. Bertsekas, "Stochastic shortest path games," *SIAM J. Contr. Optimization*, vol. 37, no. 3, pp. 804–824, 1999.

[12] T. Başar and G. J. Olsder, *Dynamic Noncooperative Game Theory*. No. 23 in Classics in Applied Mathematics, Philadelphia: SIAM, 2nd ed., 1999.

[13] J. Aubin, *Optima and Equilibria: An Introduction to Nonlinear Analysis*. Graduate Texts in Mathematics, Berlin: Springer-Verlag, 2nd ed., 1998.