Resilience of Routing in Parallel Link Networks

Eitan Altman, Aniruddha Singhal, Corinne Touati, Jie Li

Resilience of Routing in Parallel Link Networks

Eitan Altman[1,2], Aniruddha Singhal[2], Corinne Touati[2], and Jie Li[3]

[1] Université Côte d'Azur
[2] Inria, France, Email: {Eitan.Altman, corinne.touati}@inria.fr
[3] Faculty of Engineering, Information and Systems, University of Tsukuba, Japan, Email: lijie@cs.tsukuba.ac.jp

Abstract. We revisit in this paper the resilience problem of routing traffic in a parallel link network model with a malicious player using a game theoretic framework. Consider that there are two players in the network: the first player wishes to split its traffic so as to minimize its average delay, which the second player, i.e., the malicious player, tries to maximize. The first player has a demand constraint on the total traffic it routes. The second player controls the link capacities: it can decrease by some amount the capacity of each link under a constraint on the sum of capacity degradation. We first show that the average delay function is convex both in traffic and in capacity degradation over the parallel links and thus does not have a saddle point. We identify best responses strategies of each player and compute both the max-min and the min-max values of the game. We are especially interested in the min max strategy as it guarantees the best performance under worst possible link capacity degradation. It thus allows to obtain routing strategies that are resilient and robust. We compare the results of the min-max to those obtained under the max-min strategies. We provide stable algorithms for computing both max-min and min-max strategies as well as for best responses.

1 Introduction

The current computer networks such as Internet architecture remain remarkably vulnerable to different security attacks and failures which may cause system unaivabilities or performance degradation. It is a great challenge to provide services under such security attacks and failures in computer networks. Resiliency is the ability to provide and maintain an acceptable level of service in the face of faults and challenges to normal operation [1].

In this paper, we study the resilience problem of routing traffic in a parallel link network model with a malicious player using game theory. Although the network model looks simple, it could be taken as a typical one for a computer network with general network configuration in which there are many paths between a source node and a destination node and a path consists of several communications.

Although our network is a simple one, the network resilience problem in the network model is not a trivial one. We study the resilience problem of routing

traffic in a parallel link network model with a malicious player using a game theoretic framework. Consider that there are two players in the network: the first player wishes to split its traffic so as to minimize its average delay, which the second player, i.e., the malicious player, tries to maximize. The first player has a demand constraint on the total traffic it routes. The second player controls the link capacities: it can decrease some amount of the capacity of each link under a constraint on the sum of capacity degradation. We first show that the average delay function is convex both in traffic and the capacity degradation over the parallel links and thus does not have a saddle point. We identify best responses strategies of each player and compute both the max-min and the min-max value of the game. We are especially interested in the min-max strategy as it guarantees the best performance under worst possible unknown link capacity degradation. It thus allows to obtain routing strategies that are resilient and robust. We compare the results of min-max to those obtained at max-min. We provide numerical algorithms for computing both max-min and min-max values and strategies as well as for best responses.

1.1 Related work

We restrict in this paper our analysis to the framework of routing in a "parallel link" network. This topology has long been a basic framework for the study of routing, as it is a natural generic framework of load balancing among servers in a network. The study of competitive routing in networks with parallel links using game theory goes back to [2]. They were further studied in [3], [4] and many others. The only reference we know that studied adversarial behavior in routing in a model similar to the max-min scenario is [5] but they do not propose an algorithmic solution as we do here. On the other hand, to the best of our knowledge, the min-max setting has not been studied before. While the max-min problem has a water-filling structure, we show that the min-max policy has a form which extends the water filling policy and we call it the "water distribution" policy. We provide an algorithm for computing it.

2 System Model and Problem Formulation

Consider a set $\mathcal{L} = \{1, ..., \mathbf{L}\}$ of parallel links between a common source s and destination d as shown in Fig. 1.

Let the delay density over link $\ell \in \mathcal{L}$ of capacity C_ℓ be given by the following function of the link flow x_l:

$$D(x_\ell, C_\ell) \triangleq \begin{cases} \dfrac{1}{C_\ell - x_\ell} & \text{if } x_\ell < C_\ell, \\ +\infty & \text{otherwise.} \end{cases} \tag{1}$$

Let $\boldsymbol{x}$ be the flow vector, $\boldsymbol{x} = (x_\ell, 1 \leq \ell \leq \mathbf{L})$. Define the system delay as the average delay experienced by the flow on the different links:

$$\hat{D}(\boldsymbol{x}, \boldsymbol{C}) \triangleq \sum_{\ell \in \mathcal{L}} x_\ell D_\ell(x_\ell, C_\ell). \tag{2}$$

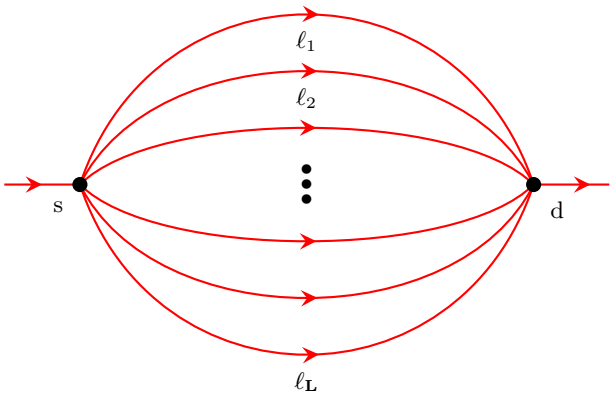


Fig. 1: A system of parallel links

Such delay system model was already widely used to describe delay in telecommunication network (see, e.g. [2]). In this paper, we address resilience in routing for such networks.

The vector $\boldsymbol{x}$ is controlled so as to minimize the system delay under the demand constraint: $\sum_{\ell \in \mathcal{L}} x_\ell = \mathbf{X}$. Meanwhile, suppose that the capacity C_ℓ of link ℓ is decreased to $C_\ell - \delta_\ell$ where $\delta_\ell \in \mathbb{R}_+$. In this case, the delay of link ℓ becomes

$$D(x_\ell, C_\ell - \delta_\ell) = \begin{cases} \dfrac{1}{C_\ell - \delta_\ell - x_\ell} & \text{if } x_\ell < C_\ell - \delta_\ell, \\ +\infty & \text{otherwise.} \end{cases}$$

Let $\boldsymbol{\delta}$ be the degradation vector, $\boldsymbol{\delta} = (\delta_\ell, 1 \leq \ell \leq \mathbf{L})$. The average system delay is therefore given by

$$\hat{D}(\boldsymbol{x}, \boldsymbol{C} - \boldsymbol{\delta}) \triangleq \sum_{\ell \in \mathcal{L}} x_\ell D_\ell(x_\ell, C_l - \delta_\ell), \tag{3}$$

and the worst degradation is therefore $\max_{\boldsymbol{\delta}} \hat{D}(\boldsymbol{x}, \boldsymbol{C} - \boldsymbol{\delta})$ subject to the constraint $\sum_{\ell \in \mathcal{L}} \delta_\ell = \boldsymbol{\Delta}$.

In this paper, we study a routing that would be robust under the worst possible impact of removing an amount $\boldsymbol{\Delta}$ of link capacities. Our objectives are as follows:

Objective 1 For a given load vector $\boldsymbol{x}$, identify the vector $\boldsymbol{\delta}$ which is the most harmful. This is addressed in Section 3.

Objective 2 Conversely, for a given attack $\boldsymbol{\delta}$, identify the best possible response $\boldsymbol{x}$ to capacity degradation. This is addressed in Section 4.

Objective 3 Determine the worst possible capacity degradation to arbitrary flow vector. This is addressed in Section 5. More precisely, our problem can be interpreted as a zero-sum game with $\boldsymbol{x}$ playing the role of the minimization player's action and $\boldsymbol{\delta}$ the maximization player's. If player $\Pi_{\boldsymbol{\delta}}$ is playing first, then it will aim at finding the attack which reduces the system capacity most:

$$\max_{\boldsymbol{\delta}} \min_{\boldsymbol{x}} \hat{D}(\boldsymbol{x}, \boldsymbol{C} - \boldsymbol{\delta}) \quad \text{subject to} \quad \begin{cases} \sum_{\ell \in \mathcal{L}} x_\ell = \mathbf{X} \text{ and} \\ \sum_{\ell \in \mathcal{L}} \delta_\ell = \boldsymbol{\Delta}. \end{cases} \tag{4}$$

Objective 4 Determine the flow vector $\boldsymbol{x}^*$ which guaranties the best possible performance under any possible capacity degradation response. This is addressed in Section 6. That is, if player $\Pi_{\mathbf{x}}$ is playing first, it will aim at choosing the flow vector $\boldsymbol{x}^*$ that guarantees the best possible performance under the worst possible reaction of attack $\boldsymbol{\delta}$ of player $\Pi_{\boldsymbol{\delta}}$:

$$\min_{\boldsymbol{x}} \max_{\boldsymbol{\delta}} \hat{D}(\boldsymbol{x}, \boldsymbol{C} - \boldsymbol{\delta}) \quad \text{subject to} \quad \begin{cases} \sum_{\ell \in \mathcal{L}} x_\ell = \mathbf{X} \text{ and} \\ \sum_{\ell \in \mathcal{L}} \delta_\ell = \boldsymbol{\Delta}. \end{cases} \tag{5}$$

A crucial question is whether the solutions of the latter two problems coincide. The following result gives a clue:

Proposition 1. *The average delay function $\hat{D}$ is convex both in $\boldsymbol{x}$ and $\boldsymbol{\delta}$.*

The proof is available in the technical report [6]

A very well studied class of games is that of concave-convex games, for which the maximizing player's optimization function is concave while the minimizer player's is convex. These games are known to have a *value*, that is their maximin optimization (4) coincide with their minimax (5). However, in our scenario, the game is convex-convex, and therefore the order at which the players are taking decisions can affect the resulting equilibrium.

In the following, we shall obviously restrict to the case that $\sum_{\ell \in \mathcal{L}} C_\ell > \mathbf{X} + \boldsymbol{\Delta}$ so that there exists a routing strategy with finite cost.

3 Optimal Attack in response to Link Utilization

In this section, we consider the optimal strategy for player $\Pi_{\boldsymbol{\delta}}$ in response to a given link usage $\boldsymbol{x}$. That is:

$$\boldsymbol{\delta}^*(\boldsymbol{x}) \triangleq \arg\max_{\boldsymbol{\delta} \geq 0} D(\boldsymbol{x}, \boldsymbol{C} - \boldsymbol{\delta}), \text{ s.t } \sum_{\ell} \delta_\ell = \boldsymbol{\Delta}.$$

The next theorem gives a characterization of the optimal reaction of player $\Pi_{\boldsymbol{\delta}}$: it is such that only a single link should be attacked, that is, the one inducing the higher throughput degradation:

Theorem 1 (Optimal attack response). *For any load vector $\boldsymbol{x}$, there exists a unique optimal reaction of player $\Pi_{\boldsymbol{\delta}}$. It is such that:*

$$\delta^*_\ell(\boldsymbol{x}) = \begin{cases} \boldsymbol{\Delta} & \text{if } \ell = \ell^* \\ 0 & \text{otherwise,} \end{cases} \quad \text{with} \quad \ell^* = \arg\max_{\ell \in \mathcal{L}} \frac{x_\ell}{(C_\ell - x_\ell)(C_\ell - \boldsymbol{\Delta} - x_\ell)}. \tag{6}$$

Proof. For a given $\boldsymbol{x}$ vector, note that $\hat{D}$ is convex in $\boldsymbol{\delta}$ and defined on the convex polytope P:

$$P \triangleq \{(\delta_\ell)_{\ell \in \mathcal{L}}, \forall \ell, 0 \leq \delta_\ell \leq \boldsymbol{\Delta} \text{ and } \sum_\ell \delta_\ell = \boldsymbol{\Delta}\}$$

Define $\boldsymbol{e_i}$ the unit vector, i.e. the vector of dimension $\mathbf{L}$ with all elements being equal to 0 except for the ith element which is 1. Then P is the convex hull of a set of $\mathbf{L}+1$ extreme points: $\{\mathbf{0} \cup \boldsymbol{\Delta} \boldsymbol{e_i}, i \in \mathcal{L}\}$.

Hence, for any point p of P, there exists non-negative $\alpha_0, ..., \alpha_L$ such that $1 = \sum_{\ell \in \mathcal{L}} \alpha_\ell$ and $p = \boldsymbol{\Delta} \sum_\ell \alpha_\ell e_\ell$. Let $\ell^* = \arg\max_\ell \hat{D}(\boldsymbol{x}, \boldsymbol{C} - \boldsymbol{e_\ell})$.

As $\hat{D}$ is convex, then $\hat{D}(\boldsymbol{x}, \boldsymbol{C} - p) = \hat{D}(\boldsymbol{x}, \boldsymbol{C} - \boldsymbol{\Delta} \sum_\ell \alpha_\ell e_\ell) \leq \boldsymbol{\Delta} \sum_\ell \alpha_\ell \hat{D}(\boldsymbol{x}, \boldsymbol{C} - \boldsymbol{e_\ell}) \leq \boldsymbol{\Delta} \sum_\ell \alpha_\ell \hat{D}(\boldsymbol{x}, \boldsymbol{C} - \boldsymbol{e^*_\ell}) = \boldsymbol{\Delta} \hat{D}(\boldsymbol{x}, \boldsymbol{C} - \boldsymbol{e^*_\ell})$ which gives Eq. (6).

The degradation of the delay induced by attacking link ℓ is $x_\ell \left(\hat{D}_\ell(\boldsymbol{\Delta}, x_\ell) - \hat{D}_\ell(0, x_\ell) \right) = \frac{x_\ell \boldsymbol{\Delta}}{(C_\ell - \boldsymbol{\Delta} - x_\ell)(C_\ell - x_\ell)}$ which leads to the desired result.

Corollary 1. *The degradation induced by player $\Pi_{\boldsymbol{\delta}}$ on the total delay equals to*

$$\frac{\boldsymbol{\Delta} x_{\ell^*}}{(C_{\ell^*} - x_{\ell^*})(C_{\ell^*} - \boldsymbol{\Delta} - x_{\ell^*})}.$$

Therefore, from Theorem 1, a straightforward algorithm can give the exact optimal attack in $3 \times \mathbf{L}$ multiplications and $\mathbf{L}$ comparisons.

4 Optimal Link Utilization as Response to some Attack

We now analyze the optimal link utilization in response to a choice of $\boldsymbol{\delta}$ from player $\Pi_{\boldsymbol{\delta}}$, which is denoted by $\boldsymbol{x}^*(\boldsymbol{\delta})$. Then, we seek:

$$\boldsymbol{x}^*(\boldsymbol{\delta}) = \arg\min_{\boldsymbol{x} \geq 0} D(\boldsymbol{x}, \boldsymbol{C} - \boldsymbol{\delta}), \text{ subject to } \sum_\ell x_\ell = \mathbf{X}.$$

The next theorem gives a characterization of $\boldsymbol{x}^*(\boldsymbol{\delta})$:

Theorem 2 (Optimal link usage response). *There exists a unique real value K such that:*

$$x^*_\ell(\boldsymbol{\delta}) = \begin{cases} C_\ell - \delta_\ell - K\sqrt{C_\ell - \delta_\ell} & \text{if } \ell \in \mathcal{X}, \\ 0 & \text{otherwise,} \end{cases} \tag{7}$$

$$\text{with} \quad \mathcal{X} \triangleq \{\ell, C_\ell - \delta_\ell \geq K^2\} \qquad \text{and} \qquad K \triangleq \frac{\sum_{\ell \in \mathcal{X}} (C_\ell - \delta_\ell) - \mathbf{X}}{\sum_{\ell \in \mathcal{X}} \sqrt{C_\ell - \delta_\ell}}. \tag{8}$$

(Note the fixed point equation between the set of links used at the optimal response $\mathcal{X}$, and the quantity K.)

Proof. For given $\boldsymbol{C}$ and $\boldsymbol{\delta}$, consider the Lagrangian function

$$L(\lambda, \boldsymbol{x}) = \hat{D}(\boldsymbol{x}, \boldsymbol{C} - \boldsymbol{\delta}) - \lambda \left(\sum_{\ell \in \mathcal{L}} x_\ell - \mathbf{X} \right). \tag{9}$$

Then, the optimal link usage response $\boldsymbol{x}^*$ is solution of the optimization problem $\min_{\boldsymbol{x} \geq 0} L(\lambda, \boldsymbol{x})$. Since $x \mapsto L(\lambda, \boldsymbol{x})$ is convex, then

$$\boldsymbol{x}^* = \arg\min_x L(\lambda, \boldsymbol{x}) \Leftrightarrow \frac{\partial L(\lambda, \boldsymbol{x}^*)}{\partial x_\ell^*} \begin{cases} \geq 0 \ \forall \ell \\ = 0 \text{ if } x_\ell^* > 0. \end{cases}$$

$$\text{Then} \quad \begin{cases} x_\ell^* = 0 \Leftrightarrow \dfrac{\partial \hat{L}}{\partial x_\ell}(\lambda, \boldsymbol{x} | x_\ell = 0) \geq 0 \Leftrightarrow \dfrac{1}{C_\ell - \delta_\ell} \geq \lambda, \\ x_\ell > 0 \Leftrightarrow C_\ell - \delta_\ell - x_\ell = \sqrt{\dfrac{C_\ell - \delta_\ell}{\lambda}} \end{cases} \tag{10}$$

which gives (7) by taking $K = 1/\sqrt{\lambda}$. Then, summing Eq. (7) over $\mathcal{X}$ yields $\mathbf{X} = \sum_{\ell \in \mathcal{X}} \left(C_\ell - \delta_\ell - K\sqrt{C_\ell - \delta_\ell} \right)$, which allows us to express K as Eq. (8).

From this theorem, we can then derive the performance achieved at the optimal link usage response:

Proposition 2 (Performance at the optimal link usage). *At the optimal $\boldsymbol{x}^*(\boldsymbol{\delta})$, the total delay on any used link ℓ (i.e. such that $x_\ell^* > 0$) is given by*

$$x_\ell^*(\boldsymbol{\delta}) D_\ell(x_\ell, C_\ell - \delta_\ell) \triangleq \frac{x_\ell^*(\boldsymbol{\delta})}{C_\ell - \delta_\ell - x_\ell^*(\boldsymbol{\delta})} = \frac{\sqrt{C_\ell - \delta_\ell}}{K} - 1$$

and the total delay is

$$\hat{D}(\boldsymbol{x}^*(\boldsymbol{\delta}), \boldsymbol{C} - \boldsymbol{\delta}) = \frac{\left(\sum_{\ell \in \mathcal{X}} \sqrt{C_\ell - \delta_\ell} \right)^2}{\sum_{\ell \in \mathcal{X}} (C_\ell - \delta_\ell) - \mathbf{X}} - |\mathcal{X}|. \tag{11}$$

Proof. From (7), we have:

$$\frac{x_\ell^*(\boldsymbol{\delta})}{C_\ell - \delta_\ell - x_\ell^*(\boldsymbol{\delta})} = \frac{C_\ell - \delta_\ell - K\sqrt{C_l - \delta_\ell}}{K\sqrt{C_\ell - \delta_\ell}} = \frac{\sqrt{C_\ell - \delta_\ell}}{K} - 1.$$

Thus

$$\hat{D}(\boldsymbol{x}^*(\boldsymbol{\delta}), \boldsymbol{C} - \boldsymbol{\delta}) = \sum_{\ell \in \mathcal{X}} \left(\frac{\sqrt{C_\ell - \delta_\ell}}{K} - 1 \right) = \frac{(\sum_{\ell \in \mathcal{X}} \sqrt{C_\ell - \delta_\ell})^2}{\sum_{\ell \in \mathcal{X}} (C_\ell - \delta_\ell) - \mathbf{X}} - |\mathcal{X}|.$$

In order to derive a powerful algorithmic solution, we need the following characterization of the optimal link usage solution:

Proposition 3 (Optimal Usage Characterization). *For each link ℓ, define the normalized delay as*

$$\mathcal{ND}_\ell(\boldsymbol{x}, \boldsymbol{C} - \boldsymbol{\delta}) = \sqrt{C_\ell - \delta_\ell}.D_\ell(\boldsymbol{x}, \boldsymbol{C} - \boldsymbol{\delta}). \tag{12}$$

Then, at the optimal $\boldsymbol{x}^*(\boldsymbol{\delta})$:

$$\mathcal{ND}_\ell(\boldsymbol{x}^*(\boldsymbol{\delta}), \boldsymbol{C} - \boldsymbol{\delta}) \begin{cases} = K(\mathcal{X}) & \text{if } C_\ell - \delta_\ell \geq 1/K^2 \\ \geq K(\mathcal{X}) & \text{if } C_\ell - \delta_\ell \leq 1/K^2 \end{cases} \tag{13}$$

Proof. At the optimal response $\boldsymbol{x}^*(\boldsymbol{\delta})$, we have, from Eq. (7), for any used link:

$$\begin{aligned} \mathcal{ND}_\ell(\boldsymbol{x}^*(\boldsymbol{\delta}), \boldsymbol{C} - \boldsymbol{\delta}) &= \frac{\sqrt{C_\ell - \delta_\ell}}{C_\ell - \delta_\ell - x_\ell^*(\boldsymbol{\delta})} \\ &= \frac{\sqrt{C_\ell - \delta_\ell}}{K\sqrt{C_\ell - \delta_\ell}} = 1/K. \end{aligned}$$

For any unused link, we have:

$$\begin{aligned} \mathcal{ND}_\ell(\boldsymbol{x}^*(\boldsymbol{\delta}), \boldsymbol{C} - \boldsymbol{\delta}) &= \sqrt{C_\ell - \delta_\ell}.D_\ell(\boldsymbol{x}^*(\boldsymbol{\delta}), \boldsymbol{C} - \boldsymbol{\delta}) \\ &= \frac{\sqrt{C_\ell - \delta_\ell}}{C_\ell - \delta_\ell} = \frac{1}{\sqrt{C_\ell - \delta_\ell}} \end{aligned}$$

But from Eq. (10), $\frac{1}{C_\ell - \delta_\ell} \geq \lambda$, i.e. $\frac{1}{\sqrt{C_\ell - \delta_\ell}} \geq \sqrt{\lambda} = 1/K$ which concludes the proof.

The proposed water-filling mechanism for the strategy of player $\Pi_{\mathbf{x}}$ is given in Algorithm 1. The links are initially sorted by decreasing capacity. The mechanism gradually increases the amount of x_ℓ of the various links until reaching $\mathbf{X}$.

More precisely, the algorithm proceeds with initialization of $\boldsymbol{x}$ as zero. At each iteration of the algorithm, the set $\mathcal{X}$ is updated by checking for some potential new candidates.

Algorithm 1: The algorithm of player $\Pi_{\mathbf{x}}$ which defines an optimal strategy in response to some attack.

Input: Vector C of channel capacities, Attack vector $\boldsymbol{\delta}$
Output: Load vector $\boldsymbol{x}$

Sort links with decreasing capacity $C_\ell - \delta_\ell$
$TA \leftarrow 0$ // The traffic allocated so far
Link $\leftarrow 1$ // The link index up to which we inject data
$\varepsilon \leftarrow 0.001$ // Set it to the desired accuracy
$\boldsymbol{x} \leftarrow \mathbf{0}$ // The traffic vector
while $TA < \mathbf{X}$ **do**
  while Link $< \mathbf{L}$ *and* $\mathcal{ND}_1(\boldsymbol{x}, \boldsymbol{C} - \boldsymbol{\delta}) \geq \mathcal{ND}_{\text{Link}+1}(\boldsymbol{x}, \boldsymbol{C} - \boldsymbol{\delta})$ **do**
    Link++
  $x_1 \leftarrow x_1 + \varepsilon$
  $K \leftarrow \dfrac{\sqrt{C_1 - \delta_1}}{C_1 - \delta_1 - x_1}$
  for $j = 2$ *to* Link **do**
    $x_j \leftarrow C_j - \delta_j - \frac{\sqrt{C_j - \delta_j}}{K}$
  Update $TA \leftarrow \sum_{\ell \in \mathcal{L}} x_\ell$
return $\boldsymbol{x}$

One can use a direct water-filling algorithm by using ε, a very small quantity, representing the discretization of level increase in the water-filling algorithm. The algorithm would be a direct implementation of Proposition 3, that is, if the current link was filled up to a level (in terms of $\mathcal{ND}$) that is greater or equal than that of next link, then variable Link is incremented so as to start filling the next link. Then, the "for" loop would fill each link j by a small amount η_j which is such that $\mathcal{ND}_j(\boldsymbol{x} + \eta_j \boldsymbol{e_j}, \boldsymbol{C} - \boldsymbol{\delta}) - \mathcal{ND}_j(\boldsymbol{x}, \boldsymbol{C} - \boldsymbol{\delta}) = \varepsilon$ until $\mathbf{X}$ is exhausted.

The performance of such algorithm exhibits average performance though, as the numerical precision errors in the level increases of the different links are summed up over the different iterations and can end up in large inaccuracy if the ratio x_ℓ/η_ℓ turns out to be large. Performance is significantly improved by using one link (for instance that of greater capacity) as the point of reference of the link level and setting up the other links levels accordingly. We propose another variant of the algorithm where ε represents the discretization of x_1. Then, at each iteration of the algorithm, x_1 is increased by ε, then K is updated and then all links in $\mathcal{X}$.

Then, the maximal error is $Err \leq \sum_{\ell \in \mathcal{L}} |x_\ell(x_1) - x_\ell(x_1 + \varepsilon)| = \sum_{\ell \in \mathcal{L}} |\frac{\sqrt{C_j - \delta_j}}{K(x_1)} - \frac{\sqrt{C_j - \delta_j}}{K(x_1 + \varepsilon)}| = \frac{\varepsilon}{\sqrt{C_1 - \delta_1}} \sum_{\ell \in \mathcal{L}} \sqrt{C_\ell - \delta_\ell}$. Since the links are ordered by decreasing capacity, then the error is bounded by $\mathbf{L}\varepsilon$.

5 Optimal Link Degradation Strategy to Unknown Link Usage

Let us now consider that player $\Pi_{\boldsymbol{\delta}}$ is to choose its attack vector, without knowing the link usage chosen by player $\Pi_{\mathbf{x}}$. Then, a natural strategy of player $\Pi_{\boldsymbol{\delta}}$ is to choose the attack vector that would guarantee the highest value of delay under any action load $\boldsymbol{x}$. Such strategy of player $\Pi_{\boldsymbol{\delta}}$ is commonly known as the *maxmin strategy* and is given by Definition 1.

Definition 1. *$\boldsymbol{\delta}^*$ is a maxmin strategy if it is solution of*

$$\mathbf{Mm}(\mathbf{C},\mathbf{X},\boldsymbol{\Delta}): \quad \max_{\boldsymbol{\delta}\geq 0}\min_{\boldsymbol{x}\geq 0} D(\boldsymbol{x},\boldsymbol{C}-\boldsymbol{\delta}), \quad s.t\ \textstyle\sum_\ell x_\ell = \mathbf{X},\ \sum_\ell \delta_\ell = \boldsymbol{\Delta}. \tag{14}$$

Note that this is equivalent to a two-player sequential game where player $\Pi_{\boldsymbol{\delta}}$ plays first, followed by player $\Pi_{\mathbf{x}}$, after it observes the action of player $\Pi_{\boldsymbol{\delta}}$.

5.1 Existence and Characterization of the Optimal Strategy

Theorem 3 shows that there exists a unique strategy for player $\Pi_{\boldsymbol{\delta}}$ and provides a characterization of it.

Theorem 3. *There exists a unique real value α such that the optimal strategy for player $\Pi_{\boldsymbol{\delta}}$ is given by:*

$$\delta_\ell = \begin{cases} C_\ell - \alpha & \text{if } \ell \in \mathcal{D}, \\ 0 & \text{otherwise} \end{cases} \quad \text{with } \alpha = \frac{\sum_{\ell\in\mathcal{D}} C_\ell - \boldsymbol{\Delta}}{|\mathcal{D}|} \quad \text{and } \ \mathcal{D} = \{\ell | C_\ell \geq \alpha\}. \tag{15}$$

The proof is given in the technical report [6].

Note that the optimal strategy for player $\Pi_{\boldsymbol{\delta}}$ is therefore to attack the links of greater capacity in a way so that their remaining capacities $(C_\ell - \delta_\ell)$ are all equal to α. Hence, the optimal strategy for player $\Pi_{\boldsymbol{\delta}}$ is independent on the weight $\mathbf{X}$ of player $\Pi_{\mathbf{x}}$.

5.2 A decreasing water filling algorithm

Based on Theorem 3, we can derive an algorithm to compute the optimal strategy of player $\Pi_{\boldsymbol{\delta}}$, which is given in Algorithm 2.

Similarly to Algorithm 1, at each step of the algorithm, the links 1 to Link are being filled. The algorithm ends whenever all links have been attacked or when the attack level $\boldsymbol{\Delta}$ has been exhausted. More precisely, at any stage of the loop, the links 1 to Link are being filled until either the attack has been exhausted (Line 9-10) or the water-level reaches that of the next link (Line 11-12).

Yet, the algorithm differs drastically from Algorithm 1 in its form and complexity. Indeed, from Equation 15, all links $\ell \in \mathcal{D}$ are such that $C_\ell - \delta_\ell$'s are

Algorithm 2: The algorithm of player $\Pi_{\boldsymbol{\delta}}$ which defines an optimal strategy for unknown link usage.

Input: Vector C of channel capacities, of size $\mathbf{L}$
Output: Attack vector $\boldsymbol{\delta}$

Sort links with decreasing capacity C_ℓ
Attack $\leftarrow \boldsymbol{\Delta}$ // Amount of $\boldsymbol{\Delta}$ left to be allocated
Link $\leftarrow 1$ // The link index up to which we attack
Diff $\leftarrow 0$ // Extra capacity of the current link to the next
$\eta \leftarrow 0$ // Amount to be allocated in each link
$\boldsymbol{\delta} \leftarrow \mathbf{0}$ // The attack vector
while Link $\leq \mathbf{L}$ *and* Attack > 0 **do**
    Diff $= C_{\text{Link}} - C_{\text{Link}+1}$
    if *(*Link $= \mathbf{L}$ *or* Attack $<$ Link $\times$ Diff*)* **then**
        $\eta \leftarrow$ Attack / Link
    else
        $\eta \leftarrow$ Diff
    Attack $\leftarrow$ Attack $-\eta$. Link
    for $j = 1$ *to* Link **do**
        $\delta_j \leftarrow \delta_j + \eta$
    Link ++
return $\boldsymbol{\delta}$

equal, which amounts to say that for i, j $\mathcal{D}$, we have $\delta_i - \delta_j = C_i - C_j$. Hence, the different links are being filled *at the same rate* η, which allows us to simply derive the level of exhaustion of $\boldsymbol{\Delta}$ or when the set $\mathcal{D}$ is to be modified. As opposed to Algorithm 1 which computes the solution with arbitrary precision, Algorithm 2 gives the exact solution. Further, the loop runs for at most $\mathbf{L}$ times and the solution is obtained after at most $\mathcal{O}(\mathbf{L})$ multiplications and $\mathcal{O}(\mathbf{L}^2)$ additions.

Figure 2 shows a typical run of the algorithm, with a set of 5 links. There, the algorithm terminates after 3 loops in the "while" command, as $\sum_\ell \delta_\ell = \boldsymbol{\Delta}$.

6 Optimal Link Usage Strategy with Unknown Degradation Attack

We finally proceed to the case where player $\Pi_{\mathbf{x}}$ chooses its routing strategy without knowledge of the attack performed by player $\Pi_{\boldsymbol{\delta}}$. Then, we consider its strategy to be the one that has the best delay guarantee, i.e. the one such that the delay it will suffer from is the lowest possible one in the worst case scenario (i.e. where player $\Pi_{\boldsymbol{\delta}}$ has the strongest attack). The problem is referred to as *minmax* and given below:

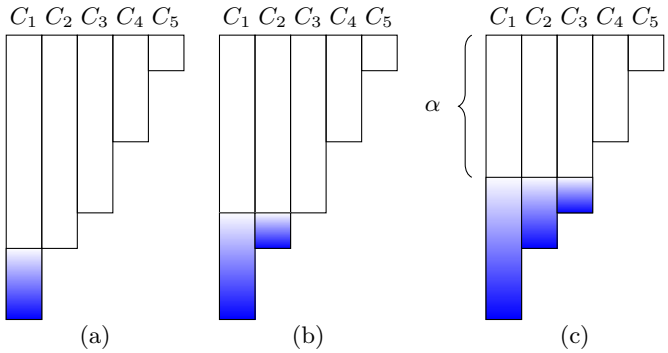


Fig. 2: Typical run of the water-filling algorithm. First, in (a), the channel with highest capacity is filled so as to reach the level of the second. Then in (b) the two channels of largest capacities are filled. Finally, in (c) as the total attack is exhausted before reaching the level of C_4, then channels 1 to 3 are equally filled with the remaining attack.

Definition 2. *$\boldsymbol{x}^*$ is a minmax strategy if it is solution of:*

$$\mathbf{mM}(\mathbf{C},\mathbf{X},\mathbf{\Delta}): \quad \min_{\boldsymbol{x}\geq 0}\max_{\boldsymbol{\delta}\geq 0} D(\boldsymbol{x},\boldsymbol{C}-\boldsymbol{\delta}), \quad s.t \sum_{\ell} x_\ell = \mathbf{X},\ \sum_{\ell}\delta_\ell = \mathbf{\Delta}. \tag{16}$$

Note that the minmax strategy is also the optimal one in a scenario of a two-player sequential game with perfect information where player $\Pi_{\mathbf{x}}$ plays first followed by player $\Pi_{\boldsymbol{\delta}}$.

6.1 Existence and Characterization of the Optimal Strategy

The following theorem states the uniqueness of the solution and gives a characterization:

Theorem 4. *There exists a unique $\boldsymbol{x}^*$ solution of Eq.* (16)*. It is such that there exists a unique α and λ such that*

$$x^*_\ell = \begin{cases} C_\ell - \frac{\mathbf{\Delta}}{2} + \frac{\mathbf{\Delta}}{2\alpha}\left(1-\sqrt{4\alpha\frac{C_\ell}{\mathbf{\Delta}}+(\alpha-1)^2}\right) & \text{if } \ell \in C_M, \\ C_\ell - \sqrt{C_\ell/\lambda} & \text{if } \ell \in C_I, \\ 0 & \text{otherwise.} \end{cases} \tag{17}$$

with

$$\begin{aligned} C_M &\triangleq \left\{\ell\in\mathcal{L},\quad C_\ell \geq \frac{1}{\lambda}\left(\frac{\mathbf{\Delta}-\alpha\mathbf{\Delta}}{\mathbf{\Delta}-\alpha/\lambda}\right)^2\right\} \\ C_I &\triangleq \left\{\ell\in\mathcal{L},\quad \frac{1}{\lambda}\left(\frac{\mathbf{\Delta}-\alpha\mathbf{\Delta}}{\mathbf{\Delta}-\alpha/\lambda}\right)^2 \geq C_\ell \geq 1/\lambda\right\} \end{aligned} \tag{18}$$

and the set of optimal responses $\boldsymbol{\delta}^*$ *of player* $\Pi_{\boldsymbol{\delta}}$ *are:*

$$\{\boldsymbol{\delta} | \exists \ell^* \in C_M, \forall \ell \neq \ell^*, \delta_\ell = 0 \text{ and } \delta_{\ell^*} = \boldsymbol{\Delta}\}.$$

Proof. From Theorem 1, we can write

$$\begin{aligned} x^* &= \arg\min \max_{\ell \in \mathcal{L}} \hat{D}(\boldsymbol{x}, \boldsymbol{C} - \boldsymbol{\Delta} \boldsymbol{e_\ell}) \\ &= \arg\min \left(\hat{D}(\boldsymbol{x}, \boldsymbol{C}) + \max_{\ell \in \mathcal{L}} D_\ell(x_\ell, C_\ell) - D_\ell(x_\ell, C_\ell - \boldsymbol{\Delta}) \right). \end{aligned}$$

Hence, problem (16) is equivalent to the following equivalent constrained optimization problem:

$$\min_{\boldsymbol{x}, \alpha} D(\boldsymbol{x}, \boldsymbol{C}) + \alpha \quad \text{s.t.} \quad \begin{cases} \forall x_\ell, \dfrac{\boldsymbol{\Delta} x_\ell}{(C_\ell - \boldsymbol{\Delta} - x_\ell)(C_\ell - x_\ell)} \leq \alpha, \\ x_\ell \geq 0, \text{ and } \sum_\ell x_\ell = \mathbf{X}. \end{cases} \tag{19}$$

The corresponding Lagrangian is

$$\begin{aligned} L(\boldsymbol{x}, \alpha, \lambda, \mu) = \sum_l \frac{x_\ell}{C_\ell - x_\ell} + \alpha + \lambda(\mathbf{X} - \sum_\ell x_\ell) \\ - \sum_\ell \mu_\ell \left(\alpha + \frac{x_\ell}{C_\ell - x_\ell} - \frac{x_\ell}{C_\ell - \boldsymbol{\Delta} - x_\ell} \right) \end{aligned}$$

with $\forall \ell, x_\ell \geq 0, \mu_\ell \geq 0$.

Let C^M be the set of links for which $\alpha + \frac{x_\ell}{C_\ell - x_\ell} - \frac{x_\ell}{C_\ell - \boldsymbol{\Delta} - x_\ell} = 0$ and x_ℓ^M the corresponding loads. Then, x_ℓ^M satisfies

$$\frac{x_\ell^M \Delta}{(C_\ell - \boldsymbol{\Delta} - x_\ell^M)(C_\ell - x_\ell^M)} = \alpha$$

$$\text{i.e. } x_\ell^M = C_\ell - \frac{\boldsymbol{\Delta}}{2} + \frac{\boldsymbol{\Delta}}{2\alpha} \left(1 - \sqrt{4\alpha \frac{C_\ell}{\boldsymbol{\Delta}} + (\alpha - 1)^2} \right).$$

If $\ell \notin C^M$, then the Karush Kuhn Tucker conditions give that $\mu_\ell = 0$ and hence the lagrangian reduce to Eq. (9). Then, Eq. 10 leads to Eq. 17.

Finally, $\ell \in C^M$ iff

$$x_\ell^M \leq x_\ell^I \text{ i.e. } \frac{x_\ell^M \Delta}{(C_\ell - \boldsymbol{\Delta} - x_\ell^M)(C_\ell - x_\ell^M)} \geq \alpha$$

But

$$\frac{x_\ell^M \Delta}{(C_\ell - \boldsymbol{\Delta} - x_\ell^M)(C_\ell - x_\ell^M)} = \frac{\lambda \Delta \sqrt{C_\ell} - \Delta \sqrt{\lambda}}{\sqrt{C_\ell} - \boldsymbol{\Delta} \sqrt{\lambda}}.$$

Therefore

$$\ell \in C^M \text{ iff } C_\ell \geq \frac{1}{\lambda} \left(\frac{\boldsymbol{\Delta} - \alpha \boldsymbol{\Delta}}{\boldsymbol{\Delta} - \alpha / \lambda} \right)^2.$$

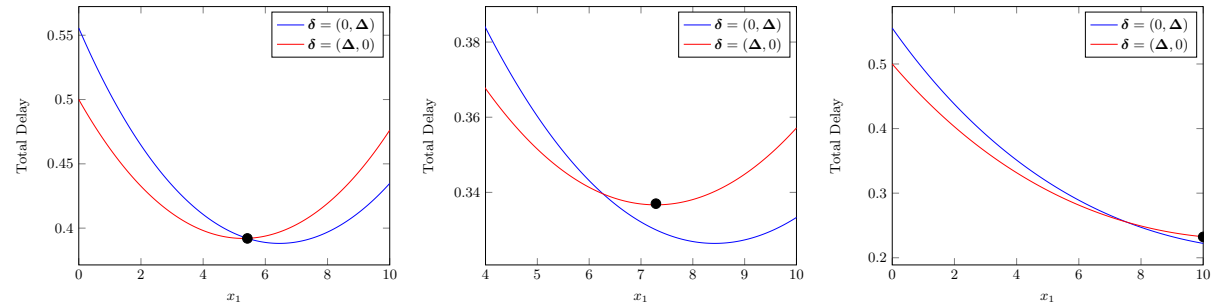
Note that α represents the degradation induced by player $\Pi_{\boldsymbol{\delta}}$. One can readily check that $\alpha = 0 \Leftrightarrow C_M = \emptyset$ which leads to Eq. 8, that is the optimal strategy for $\Pi_{\boldsymbol{\delta}}$ when there is no capacity degradation.

Algorithm 3: A water-distributed algorithm for optimal strategy of player $\Pi_{\mathbf{x}}$ with unknown attack

Sort links with decreasing capacity C_ℓ
$\varepsilon \leftarrow 0.01$, $\varepsilon_\alpha \leftarrow 0.1$ // Set it to the desired accuracy
$TA \leftarrow 0$ // The traffic to be redistributed
$\ell_M \leftarrow 1$ // The link index up to which we reduce the flow
$\ell_I \leftarrow 1$ // The link index up to which we increase the flow
$\boldsymbol{x} \leftarrow$ Solution of Algorithm 1 with no attack ($\boldsymbol{\delta} = \mathbf{0}$)
$\alpha \leftarrow \frac{x_1}{C_1 - \boldsymbol{\Delta} - x_1} - \frac{x_1}{C_1 - x_1}$
$value \leftarrow \frac{x_1}{C_1 - \boldsymbol{\Delta} - x_1} + \sum_{l=2}^{L} \frac{x_l}{C_l - x_l}$
$prec \leftarrow value + 1$
while $value < prec$ **do**
  $\alpha \leftarrow \alpha - \varepsilon_\alpha$, $prec \leftarrow value$, $\ell_I \leftarrow \ell_M$
  for $\ell = 1$ *to* ℓ_M **do** // Reduce all Links in M
    $TA \leftarrow TA + x_\ell$
    $x_\ell \leftarrow C_\ell - \frac{\boldsymbol{\Delta}}{2} + \frac{\boldsymbol{\Delta}}{2\alpha}\left(1 - \sqrt{4\alpha\frac{C_\ell}{\boldsymbol{\Delta}} + (\alpha - 1)^2}\right)$
    $TA \leftarrow TA - x_\ell$
  while $TA > 0$ **do** // Redistribute TA among the links
    while $\ell_M < \mathbf{L}$ *and* $\alpha \leq D(x_{\ell_M+1}, C_{\ell_M+1} - \boldsymbol{\Delta}) - D(x_{\ell_M+1}, C_{\ell_M+1})$ **do**
      $\ell_M ++$
    $\ell_I \leftarrow \ell_M$
    while $\ell_I < \mathbf{L}$ *and* $\mathcal{ND}_{\ell_I}(\boldsymbol{x}, \boldsymbol{C}) \geq \mathcal{ND}_{\ell_I+1}(\boldsymbol{x}, \boldsymbol{C})$ **do**
      $\ell_I ++$
    for $j = \ell_M + 1$ *to* ℓ_I **do**
      $\eta \leftarrow \frac{\varepsilon(C_j - x_j)^2}{\sqrt{C_j} + \varepsilon(C_j - x_j)}$
      $x_j \leftarrow x_j + \eta$
      $TA \leftarrow TA - \eta$
  $value \leftarrow \frac{x_1}{C_1 - \boldsymbol{\Delta} - x_1} + \sum_{l=2}^{L} \frac{x_l}{C_l - x_l}$

6.2 An Algorithmic Solution

We use the equivalent optimization problem given in Eq. (19). Since it is a convex optimization problem, standard optimization tools (e.g. a projected gradient descent on the Lagrangian) can be used, although they exhibit poor performance, in particular because of the nature of the needed projection and the system's size.



(a) Case $C_1 = 33$: then $C_M = \{1, 2\}$, $C_I = \emptyset$

(b) Case $C_1 = 40$: then $C_M = \{1\}$, $C_I = \{2\}$

(c) Case $C_1 = 55$: then $C_M = \{1\}$, $C_I = \emptyset$

Fig. 3: Different sets C_M and C_I. In the three examples, $C_2 = 30$, $\mathbf{X} = 10$ and $\mathbf{\Delta} = 2$. The min-max solution is represented with the black point. In each plot, the two graphs (red and blue) represent the overall delay that would be experienced by the user if the attack was concentrated on a single link (1 and 2 respectively).

Therefore, we propose an algorithm, in a similar vein to water-filling algorithms, which we refer to as *water-distributed algorithm*. It can be seen as a water-filling algorithm with a top cap on the water level (represented by α).

The mechanism is given by Algorithm 3. We initialize it by using Algorithm 1 to compute the optimal allocation $\boldsymbol{x}$ if there was no attack. We deduce the initial value of α.

We then iteratively decrease the value of α and compute the corresponding allocation $\boldsymbol{x}$. The algorithm ends when no gain in the delays is obtained.

Instead of computing the whole allocation at each iteration of the algorithm, we compute the amount of flow which is removed to the links of C_M as a consequence of the decrease of α (lines 21 to 25) and then redistribute this amount to the other links (Line 26 to 36).

6.3 Different C_M and C_I

Note that the set C_M and C_I both depend on the parameter $\mathbf{X}$, $\mathbf{\Delta}$ and the link capacities C_ℓ, $1 \leq \ell \leq \mathbf{L}$.

As long as $\mathbf{\Delta} > 0$, the set C_M is always non-empty (as it should include the link of highest capacity). In contrast, the set C_I can be empty or not. Further, the set $C_M \cup C_I$ may cover all links or not. The different situations are illustrated in the scenario of Figure 3. The system has a set of two links. In Figure 3a both of them are in C_M. In this case, the set C_I is empty. Figure 3c shows a scenario where C_I is also empty and C_M consists of only the link of highest capacity. Finally, Figure 3b shows a case where C_M only contains the link of higher capacity, while the other one is in C_I.

7 Conclusion

We have studied in this paper a game between a router that has a fixed demand to ship and a malicious controller that affects the system capacity by some fixed amount and can decide how to split this among different links. It turned out to be a non standard zero-sum game since the cost is convex for both the minimizer and maximizer and thus does not have a saddle point. We thus focused on computing the max-min and the min-max value and proposed efficient solution algorithms. While the max-min problem is solved using a water-filling algorithm, the solution of the minmax problem requires a more complex algorithm which we call water-distributing algorithm. We plan in the future to extend the problem to several players that try selfishly to minimize their cost in the presence of adversarial capacity degradation controller.

References

1. P. Smith, D. Hutchison, J. Sterbenz, M. Scholler, A. Fessi, M. Karaliopoulos, C. Lac, and B. Plattner, "Network resilience: a systematic approach," *Communications Magazine, IEEE*, vol. 49, no. 7, pp. 88–97, July 2011.
2. A. Orda, R. Rom, and N. Shimkin, "Competitive routing in multiuser communication networks," *IEEE/ACM Trans. Netw.*, vol. 1, no. 5, pp. 510–521, Oct. 1993. [Online]. Available: http://dx.doi.org/10.1109/90.251910
3. T. Harks, "Stackelberg strategies and collusion in network games with splittable flow," *Approximation and Online Algorithms*, 2009.
4. E. Koutsoupias and Papadimitriou, "Worst-case equilibria," in *STACS*, 1999.
5. G. Blocq and A. Orda, "Worst-case coalitions in routing games," *Arxiv*, Aug. 2014.
6. E. Altman, A. Singhal, C. Touati, and J. Li, "Resilience of routing in parallel link networks," Hal, Tech. Rep., 2015, https://hal.inria.fr/hal-01249188.