Stability of Adversarial Routing with Feedback*

Bogdan S. Chlebus[†]

Vicent Cholvi[‡]

Dariusz R. Kowalski[§]

Abstract

We consider the impact of scheduling disciplines on performance of routing in the framework of adversarial queuing. We propose an adversarial model which reflects stalling of packets due to transient failures and explicitly incorporates feedback produced by a network when packets are stalled. This adversarial model provides a methodology to study stability of routing protocols when flow-control and congestion-control mechanisms affect the volume of traffic. We show that any scheduling policy that is universally stable, in the regular model of routing that additionally allows packets to have two priorities, remains stable in the proposed adversarial model.

Keywords: adversarial queueing, adversary with feedback, packet routing, scheduling policy, stability, universal stability, link fault, transient fault.

^{*}The results of this paper were announced in a preliminary form in [14] and in its final form in [15].

[†]Department of Computer Science and Engineering, University of Colorado Denver, Denver, Colorado, USA. Supported by the NSF Grant 1016847.

[‡]Department of Computer Science, Universitat Jaume I, Castellón, Spain. Supported by the MEC Grant TIN2011-28347-C01-02.

[§]Department of Computer Science, University of Liverpool, Liverpool, UK. Supported by the EPSRC grant EP/G023018/1.

1 Introduction

We consider routing in communication networks when transient transmission failures and congestioncontrol mechanisms affect the number of packets handled by nodes. The overall goal is to have fluid traffic and avoid a congestive collapse, as means to optimize the use of a network.

The following are general comments relevant to the methodological approach considered in this work. First, routing protocols operate in environments involving flow control and congestion control. In the real-world implementations of routing, packets are dropped when the time assigned for processing has been surpassed, which has a stabilizing effect on performance. The algorithms we consider do not drop packets before they are delivered to their destinations. This requires assuming potentially unbounded private memory at nodes to accommodate any number of packets in transit. Second, stochastic approaches to model performance usually rely on strong assumptions about how traffic is generated, which may be considered too constrained. In spite of that, stochastic approaches encounter technical challenges to assess factors underlying a network's performance [28]. In this paper, we consider worst-case bounds on the performance of routing algorithms in adversarial frameworks of packet generation. This allows one to obtain insights into traffic efficiency phenomena while avoiding making ad-hoc stochastic assumptions and abstracting from low level mechanisms implemented on the network and transport layers.

Adversarial queuing was proposed as a methodology to analyze worst-case bounds on routing performance in a framework of traffic environment determined by parameters like injection rates and burstiness [5, 12]. The basic aspect of a satisfactory behavior of a system in adversarial queuing is stability, which is defined to mean that the number of packets handled simultaneously is bounded at all times.

Our goal is to extend the model of adversarial queuing by incorporating features representing congestion control. This includes the feedback provided by the network to the nodes to notify them that some packets have been stalled. Delayed feedback is more realistic than assuming that failures are known in advance when an execution of a routing protocol starts.

The routing protocols that we consider do not drop packets intentionally, but still some packets may be delayed due to malfunctioning of a communication infrastructure. This includes transient wire-link failures and interferences on wireless links, to the effect that packets may fail to be successfully transmitted between nodes.

Another related situation occurs when scheduled packets are delayed due to nodes being switched off for energy savings. The recently adopted IEEE 802.3az Energy Efficient Ethernet (EEE) standard, as described in [22], is expected to be conducive to energy savings in local area networks by implementing mechanisms to have nodes temporarily unavailable to cooperate in routing [19].

Our contributions. Now we summarize the contributions of this paper. We propose an adversarial model to study routing in faulty systems, which incorporates feedback about packets that are stalled on their itineraries. The model is an extension of leaky-bucket regulations to model injecting packets into a system subject to constraints on injection rate and burstiness. In the proposed adversarial model, the adversary learns of transient failures on links that caused packet delay after some time rather than instantaneously, which represents interaction of routing protocols with congestion-control mechanisms. Knowledge of the adversary is interpreted as an additional constraint on the power to inject packets. Protocols are considered stable when they are universally stable and unstable otherwise. We show that any scheduling policy that is unstable under the regular adversarial queueing model remains such in the proposed adversarial model. Next, we demonstrate that any scheduling policy universally stable in the 2-priority model remains stable in the adversarial model that we propose. This extends our understanding of universal stability in adversarial queuing with prioritized packets, as obtained by Àlvarez et al. [1]. Finally, we show a possible extension of the proposed adversarial model to capture permanent link faults.

The related work. The adversarial methodology to study store-and-forward routing in wired networks was proposed by Andrews et al. [5] and Borodin et al. [12]. Adversarial communication in wireless networks was considered by Andrews and Zhang [6]. Stability of broadcast protocols in adversarial multiple-access channels was studied by Anantharamu et al. [3, 4], Bender et al. [8], and Chlebus et al. [16, 17].

Adversarial models capturing failures have been proposed in the literature in various network settings. Borodin et al. [13] considered slowdowns associated with links. The papers [11, 18, 23, 24] considered dynamic changes in the link capacities, with the intention to interpret such transient decreasing of capacity of a link as a transient failure of the link.

Álvarez et al. [1] proposed a model that allows transient disruptions of the connectivity of the system. That model was extended by Álvarez et al. [2] to incorporate node failures. The models mentioned above assume that the adversary can make a link fail at any round. The papers [1, 2, 25] assume that, at each round, the adversary knows when links fail. It is then natural for adversaries to be equipped with the power to adjust the injection of packets to such events, possibly even before link failures occur. We propose an approach in which the constraints on the adversary, in terms of the injection rate and burstiness, are modified after some time delay triggered by malfunctioning of links.

The adversarial approach has been applied to modeling malfunctioning of wireless networks, including single-hop multi-channels. Bhandari and Vaidya [9, 10] considered broadcast protocols in multi-hop wireless networks with nodes prone to failures. Gilbert et al. [20] considered a multi-channel where the adversary controls how information flows on subsets of channels. Meier et al. [27] considered adversarial multi-channel single-hop networks when some t channels out of m could be disrupted in a round, with m known and t not known.

Adversarial queuing was applied when studying interference and jamming in wireless networks, including single-hop multi-channels and multiple access channels. Lim et al. [25] proposed an adversarial model to capture interferences among the links in wireless networks. In this case, at each round the adversary assigns specific edge rate vectors that are assumed to keep the network stable. These vectors can be interpreted as reflecting the degree to which edges fail by not providing their full capacity. Anantharamu et al. [4] considered multiple access channels with adversarial jamming, when the attached stations perceive jamming as colliding attempts by different stations to access the channel. Awerbuch et al. [7] studied saturation throughput of randomized protocols in adversarial multiple access channels subject to jamming. Gilbert et al. [21] studied single-hop multi-channel networks with communication subject to adversarial jamming.

For a general discussion of topics related to the mechanisms of flow and congestion control, see [26, 28].

2 The adversarial model with feedback

We propose an adversarial approach to study stability of routing which captures packet delays due to failures of network elements. This methodology is an extension of the regular leaky-bucket adversarial model determined by injection rate and burstiness. The new component is the feedback from the network after an unsuccessful transmission. This feedback restricts the adversary's capability to inject packets.

We will model networks as directed graphs G = (V, E), where the vertices in V represent the nodes of the network and the edges in E are the links connecting nodes. The orientation of an edge represents the direction in which the link can transmit data. The networks we consider are *synchronous*, in that an execution of a communication protocol is partitioned into rounds.

Each packet is injected by the adversary into some node and assigned a path through the network to traverse. Such paths cannot contain the same link more than once. If more than one packet wishes to cross an edge e in a round, then a routing protocol chooses one of these packets to send across e, while the remaining packets are kept in a queue at the tail of the edge e. When a packet reaches the destination node then it is absorbed, which means that it disappears.

A packet travels through the network with additional information associated with it, like the destination address or the round when it was injected into the network. A packet encapsulated with this information makes an atomic unit of data to be transmitted through links, which we call simply a *message*. Messages and rounds are scaled to each other, in that it takes one round to transmit a message through a link.

In this work, we consider routing environments in which a packet scheduled to be transmitted over a link in a round may fail to traverse the link, this possibly happening for a consecutive number of rounds at a time. When we use the terms "faulty round" and "faulty link," these refer to situations where failures in messages traversing links occur. We assume that messages are never lost in transmissions, in that they are successfully handed over from a node to the next neighbor on the traversed path, until the packets reach their destination.

2.1 A leaky-bucket regulation

We define the adversarial model by how traffic is regulated. We use a traffic descriptor using the notion of a leaky-token bucket, as proposed by Turner [29]. Traffic demand is determined by packets injected into the network, each packet being assigned a path to traverse. The notion of packets becoming stalled in their journeys is to represent a general malfunctioning of the system that results in a packet getting delayed, when an attempted transmission on a link fails, regardless of what is the reason of failure. A packet is *stalled* in round t if it is attempted to be transmitted but the transmission fails.

An adversary is defined by three parameters: *injection rate* r, such that $0 < r \le 1$, *burstiness* b, which is a positive integer, and *feedback delay* δ , which is also a positive integer. These three parameters together determine the *adversarial type* (r, b, δ) .

We will consider two kinds of virtual objects called tokens and antitokens. A *token* is in a bucket and represents the ability to inject a packet. An *antitoken* represents a stalled packet, and so the need to decrease the traffic by one packet to avoid congestion.

A single *bucket* is a variable K storing a number; K is initialized to 0. When $K \ge 0$, then $\lfloor K \rfloor$ is interpreted as the number of tokens in the bucket. The bucket's capacity is b. In general, K may assume negative values; in such a case the bucket does not contain any tokens. The operations performed on K are as follows.

- 1) In each round, first r is added to the bucket K by the assignment $K \leftarrow K + r$. If at this point K > b then K is immediately modified by the assignment $K \leftarrow b$, which is interpreted as the bucket's overflow.
- 2) The adversary injects some i packets into the system and simultaneously removes i tokens from the bucket K by performing $K \leftarrow K - i$. For this to be possible to perform, the inequalities $0 < i \le K$ need to hold.
- 3) The adversary controls when packets are stalled. Each round a packet is stalled, an antitoken g is created, carrying a value. The value of a newly created antitoken is initialized to δ . The value of an antitoken gets decremented by 1 in each round. An antitoken disappears in two possible ways, as decided by the adversary. One results in removing the antitoken and simultaneously modifying $K \leftarrow K r$ while the token's value is still positive. Another is when the value becomes 0, then the antitoken disappears, and simultaneously $K \leftarrow K r$; this represents the maximally delayed feedback.

Intuitively, when the adversary decides to annihilate a token, then this represents a moment when the adversary obtains feedback from the network of a stalled packet. One token and one antitoken disappear simultaneously, as by annihilation resulting from their getting mixed together, which explains the terminology.

This completes the specification how a single bucket operates, when it is considered in isolation independently from other buckets. The complete picture is such that we associate a bucket K_e with each directed edge e of the network.

The operations on these buckets are coordinated as follows:

- 1) Every bucket gets incremented by r in each round, subject to possible overflow which makes a bucket store precisely b tokens.
- 2) A packet has a path assigned to traverse; when a packet gets stalled then an antitoken is created for each bucket K_e associated with an edge e that the packet is still to traverse; all these antitokens are said to be *related*.
- 3) When the adversary injects a packet to traverse a path, then it removes a token from each bucket K_e associated with any edge e of the path. For this to be possible to be performed, each bucket on the path needs to include at least one token.
- 4) When the adversary destroys an antitoken g created by a stalled packet p on some link, then such a destruction, and the matching operation $K \leftarrow K - r$, is performed on each related token g' created on a link which p was still to traverse when g was created, and the bucket associated with a link that p was still to traverse when g was created.

This completes the specification of how the adversary can inject packets into the network.

2.2 Comparison with the regular adversary

The regular leaky-bucket adversary is defined by two parameters: *injection rate* r, such that $0 < r \leq 1$, and a positive integer *burstiness b*. These two parameters together determine the *adversarial type* (r, b).

Lemma 1 A delayed-feedback leaky-bucket adversary of type (r, b, δ) is at least as powerful as the regular leaky-bucket adversary of the type (r, b).

Proof: We compare the two adversarial models as regulated by a leaky bucket of tokens. When the delayed-feedback adversary of the type (r, b, δ) does not induce any stalling among the packets, then the regulatory properties of a bucket of tokens determine the regular adversary of the type (r, b).

Theorem 1 If a scheduling policy is unstable in a network G under a scheduling policy S against the regular adversary of type (r, b) then this same scheduling policy S is unstable in the network G against a delayed-feedback adversary of the type (r, b, δ) , for any positive integer δ .

Proof: Consider an unstable execution of routing against the adversary of the type (r, b) when the scheduling policy S is applied. A similar unstable execution can be produced by the adversary of the type (r, b, δ) , by Lemma 1.

We consider the following specific scheduling policies: First-In-First-Out (FIFO), Nearest-To-Go (NTG), Farthest-From-Source (FFS), and Slowest-Previous-Link with ties broken using the Nearest-From-Source policy (SPL-NFS).

Corollary 1 Each of the scheduling policies FIFO, NTG, FFS and SPL-NFS is unstable in some network against a delayed-feedback adversary with injection rate less than 1.

Proof: The argument is based on Theorem 1. We rely on the respective instability results obtained for the regular adversary. The instability of the scheduling policies FIFO, NTG, and FFS follows from the instabilities obtained by Andrews et al. [5], and the instability of SPL-NFS follows from the related result given by Blesa et al. [11]. \Box

3 Properties of the adversarial model with feedback

In this section, we investigate properties of the adversarial model with feedback presented in the previous section. A key point of the model is the concept of an antitoken, which represents a stalled packet and thus the need to decrease the bound on the future traffic by one packet to avoid congestion. We re-define the adversary by expressing its power in a more analytical way to facilitate the future technical analysis. More precisely, we describe our adversary in terms of an admissibility condition, which involves a delay function accounting for the impact of each antitoken's annihilation on decreasing the amount of traffic that could be injected.



Figure 1: An illustration for the admissibility condition (1). The number $s_q^{\delta}(t)$ represents the rounds when the adversary reacts to stalled packets in q. The injected data $I_q^{\delta}(t)$ represents the admissible amount of data that the adversary can inject due to the extra rounds incurred by stalling.

3.1 Reformulation of the adversarial model

In each round, the adversary may inject packets into some of the nodes in the network. In order for stability to be achievable in principle, the adversary needs to be somehow restricted. Such restrictions imposed on the regular leaky-bucket adversary are represented by its adversarial type (b, r), where $b \ge 1$ is a natural number and r satisfies $0 \le r < 1$. For each link, the injection rate rmodels an upper bound on the frequency with which packets that eventually need to traverse the link can be injected into the network. The burstiness b represents the maximum number of packets, among those that need to traverse the same link, that the adversary can inject into the network in one round. The precise interpretation of such a type (b, r) is that in any time interval τ of length $|\tau|$ the adversary may inject at most $r|\tau| + b$ packets that need to traverse the same edge. An adversary is free to choose both the source and the destination node for any injected packet. The adversary also determines the individual path from the source to the destination that any specific packet needs to traverse.

An extension to the adversarial model with delays, which we refer to as \mathcal{DF}_{δ} (feedback delayed by up to δ), is as follows:

Let $I_q^{\delta}(t)$ represent the total number of packets which the adversary injects at round t that have

queue q on their path. We say that the packet injections are *admissible for rate* r *and burstiness* b if the following *admissibility condition* holds for all q:

$$\sum_{t\in T} I_q^{\delta}(t) \le r \sum_{t\in T} (1 - s_q^{\delta}(t)) + b, \tag{1}$$

where T represents a contiguous time interval and s_q^{δ} is a delay function. A formal specification of s_q^{δ} is provided in Figure 3 in Section 3.2.

The function s_q^{δ} represents the rounds when the adversary becomes constrained by packets getting stalled in queue q. We interpret them as the rounds where the adversary "reacts" to stalled packets. At any round t, the adversary takes into account the values of $s_q^{\delta}(t')$, for $t' \leq t$, based only on the notifications received up to that round, as reflected in Figure 3. This means that the admissibility condition in Equation (1) refers only to the received notifications and the packets injected into the system. Figure 1 gives an example of the admissibility condition in a queue q.

Next, we compare the adversarial power in \mathcal{DF}_{δ} with its *matching adversary* in (r, b, δ) , that is, adversaries with the same respective parameters.

Proposition 1 Any adversary in \mathcal{DF}_{δ} is equivalent to its matching adversary in (r, b, δ) for all $0 \leq r < 1$ and $b \geq 1$.

Proof: What distinguishes the adversary in \mathcal{DF}_{δ} and in (r, b, δ) is that, on the one hand, while in \mathcal{DF}_{δ} the adversary obtains the feedback (regarding stalled packets) provided by the underlying system, in (r, b, δ) the adversary has the power to control any malfunctioning of the network. A second difference is how the adversary can inject packets into the network. While in (r, b, δ) that is specified by means of a leaky-bucket mechanism; in \mathcal{DF}_{δ} that is done by means of an admissibility condition.

Regarding the first difference, we have that these two approaches are equivalent, as the adversarial model is to capture a worst-case behavior of the network.

Regarding the second difference, we have that they are also equivalent. Let us first start by considering the admissibility condition. It captures the following intuitions:

1. When $s_q^{\delta}(t) = 0$: This can be interpreted as if there are no annihilations of antitokens at time t in q.

In this case, the value of the bucket K is not modified due to some annihilation, as also happens in the admissibility condition. Since the regulatory properties of the bucket of tokens are equivalent to the regulatory properties of the admissibility condition [5], we are done.

2. When $s_q^{\delta}(t) = 1$: This can be interpreted as if an antitoken has been annihilated at time t in q.

In this case, the value of the bucket is reduced by r, and since $1 - s_q^{\delta}(t) = 0$, then the component corresponding to time t in the admissibility condition is equal to 0. In other words, there is a reduction of r in the admissibility condition with respect to the case where $s_q^{\delta}(t) = 0$. The regulatory properties of the bucket of tokens is equivalent to the regulatory properties of the admissibility condition [5], so we are also done.

The reasoning in the case where we consider the leaky-bucket mechanism is similar: an annihilated antitoken at time t in q can be interpreted as the case where $s_a^{\delta}(t) = 1$.

The definition of stability in \mathcal{DF}_{δ} is similar to the definition stated under other adversarial models.

Definition 1 Let G be a network, \mathcal{P} a scheduling policy and \mathcal{A} an adversary of type (r, b). Let \mathcal{D} be an execution of protocol \mathcal{P} against \mathcal{A} in G. For a positive integer t, let $Q_{\mathcal{D}}(t)$ be the number of packets that are queued in the system at time t. Protocol \mathcal{P} is stable on G against \mathcal{A} if, in each such execution \mathcal{D} , all the numbers $Q_{\mathcal{D}}(t)$ are bounded. Protocol \mathcal{P} is universally stable if it is stable against any adversary with injection rate r < 1 and in any network.

3.2 Delay functions and reactive functions

We say that a queue q is *stalled* in round t if some packet in the queue is stalled in this round. Next, we introduce the function w_q which represents the rounds where delays occur at queue q.

Definition 2 Consider an execution of a system up to round t. Given a queue q, we define the function $w_q(t)$ such that $w_q(t) = 1$ if the queue q is stalled at round t and $w_q(t) = 0$ otherwise.

The rounds that are added to a stalled packet's itinerary occur as a side effect of the adversary's actions. The adversary receives information about the extra rounds of stalled packets occurring at the different queues. We consider the case where the adversary becomes constrained by the stalled packets after some time delay; the parameter δ is used to bound such a maximum delay.

Next, we introduce the notations T_q , D_q^{δ} , and w_q^{δ} . We want w_q^{δ} to model the rounds where the adversary becomes constrained by the queue q getting stalled and it also provides the number of notifications.

Definition 3 We will use the following terminology and notations:

- 1. Let T_q be the set of those rounds t for which $w_q(t) = 1$ holds.
- 2. For a given function w_q , let the function $D_q^{\delta}: T_q \to \mathbb{N}$ be such that $D_q^{\delta}(t) = t'$, for $t \in T_q$ and $t \leq t' \leq t + \delta$, where t' is the time when the feedback about the queue q being stalled at time t arrives.
- 3. Let $T_q^{\delta}(t)$ be the subset of T_q such that $t' \in T_q^{\delta}(t)$ if $D_q^{\delta}(t') = t$.
- 4. For a given w_q and a given D_q^{δ} , let the delay function w_q^{δ} be determined by the equality $w_q^{\delta}(t) = |T_q^{\delta}(t)|$.

Figure 2 provides a graphical representation of the notions introduced in Definition 3 in a specific example.

Reactive functions. When the adversary receives a notification of a queue q getting delayed, then this indicates that some packet in this queue will need an extra round. In order to maintain stability, the adversary needs to take such an eventuality into account. A reaction can be interpreted



Figure 2: An illustration of how the function D_q^{δ} is determined.

as if temporarily the injection rate were reduced, which is implemented by the mechanism of tokens and antitokens. Several notifications could be received at the same time, which needs to be reflected in a cumulative reaction. For instance, if at some time t the adversary receives three notifications of stalling for some given link, it will reduce the long term injection rate of a packet that will cross such a link for three rounds after time t, provided such rounds have not been already reduced because of previous notifications, in which case the next "available" rounds will be chosen.

To formalize this, we specify the reactive function s_q^{δ} in Figure 3. It is intended to model the rounds when the adversary will react to a delayed notification of stalling at a queue q. This function s_q^{δ} is determined by w_q^{δ} , which gives the rounds when the adversary receives notifications of delays occurring at queue q, as represented by annihilations of antitokens.

4 Stability of scheduling policies

In this section, we show that some scheduling policies are stable in the adversarial-queuing model \mathcal{DF}_{δ} . We will use an auxiliary model, known as the *priority model*, which was introduced in [1]. We refer

initialization:

| $s_q^{\delta}(t) \leftarrow 0 \text{ for all } t;$ | | | | |
|--|---|------|----|----------------|
| $t \leftarrow 1;$ | % | time | in | w_q^{δ} |
| $t' \leftarrow 1;$ | % | time | in | s_a^{δ} |

repeat forever

| $\mathbf{if} \; w_q^\delta(t) \neq 0 \; \mathbf{then}$ | % when one or more notifications are received at time step t |
|--|--|
| $t' \leftarrow \max(t, t');$ | $\ensuremath{\mathscr{K}}$ set up the next "available" round in s_q^δ after t |
| for $t_{aux} = t'$ to $(t' +$ | $w_q^\delta(t)-1)\;{f do}$ |
| $s_q^{\delta}(t_{aux}) \leftarrow 1;$ | $\%$ mark t_{aux} as "reactive" |
| $t' \leftarrow t' + w_q^{\delta}(t) - 1;$ | $\%$ set up the next "available" round in s_q^δ |
| $t \leftarrow t + 1;$ | % increment the time step |

Figure 3: A specification of the reactive function s_q^{δ} . It is determined by a given delayed function w_q^{δ} .

to the model as a *c*-priority model when there are c priorities. It is obtained by modifying the regular adversarial model [5, 12] so that packets have priorities in the following sense. If, at a certain time, more than one packet located at the same queue is ready to be transmitted, then the scheduling policy chooses the packet of the highest priority.

The following fact provides a relationship between the number of extra rounds due to delays and the number of reactive rounds at a given time interval.

Lemma 2 $\sum_{t \in T} w_q(t) \leq \sum_{t \in T} s_q^{\delta}(t) + \delta$.

Proof: By the definition of function w_q^{δ} , for each round t where $w_q(t) = 1$, there is a round t', which is not necessarily different for each t, such that $w_q^{\delta}(t') \neq 0$ and $t \leq t' \leq t + \delta$.

From the mechanism used to construct s_q^{δ} and specified in Figure 3, the function s_q^{δ} takes the value 1 as many times as the values taken by the function w_q^{δ} . It follows that the number of times where w_q takes the value 1 is the same as the number of times where s_q^{δ} takes the value 1, although not necessarily at the same rounds.

This means that there exists a bijective increasing function rD_q^{δ} such that, for all t, when $w_q(t) = 1$ then $rD_q^{\delta}(t) = t'$, where $s_q^{\delta}(t') = 1$.

We proceed with considering the following two possible cases.

The case of t > t':

This case cannot happen by the construction of s_q^{δ} , because t_{aux} in Figure 3 is always at least t. The case of $t' > t + \delta$:

We prove this case by contradiction. Let t_1 be the first round such that $rD_q^{\delta}(t_1) = t_1^*$ and $t_1^* > t_1 + \delta$. From the construction of rD_q^{δ} described above and specified in Figure 3, we have that the equality $s_q^{\delta}(t'') = 1$ holds for all $t'' \in [D_q^{\delta}(t_1), rD_q^{\delta}(t_1) - 1]$. Since rD_q^{δ} is a bijective increasing function, there must exist some round t_m such that $t_m < t_1$ and $rD^{\delta}(t_m) = t_1^* - 1$. Let $t_m^* = t_1^* - 1$.

The following two facts hold. One is that $t_m < t_1$ and $t_m^* = t_1^* - 1$, which means $t_m^* > t_m + \delta$. The other is that $rD^{\delta}(t_m) = t_m^*$. All this contradicts our assumption that t_1 is the first round such that $rD_q^{\delta}(t_1) = t_1^*$ and $t_1^* > t_1 + \delta$. The fact that the inequality $t \le t' \le t + \delta$ holds means that for each t such that $w_q(t) = 1$, the corresponding image in s_q^{δ} will be for a round t' that is delayed by at most δ rounds. We conclude that the inequality $\sum_{t \in T} w_q(t) \le \sum_{t \in T} s_q^{\delta}(t) + \delta$ holds.

The following Lemma 3 shows that if a given scheduling policy is unstable in \mathcal{DF}_{δ} then it is also unstable in the 2-priority model.

Lemma 3 If a given scheduling policy is unstable against an adversary with injection rate r and burstiness b in \mathcal{DF}_{δ} , then such a scheduling policy is unstable against some adversary of injection rate r' and burstiness b' in the 2-priority model, where 0 < r' < 1.

Proof: Let us take an adversary \mathcal{A} in \mathcal{DF}_{δ} with parameters (r, b). Then, according to the admissibility condition in Equation (1) and by Lemma 2, the following estimates hold:

$$\sum_{t \in T} I_q^{\delta}(t) \leq r \sum_{t \in T} (1 - s_q^{\delta}(t)) + b$$

$$\leq r \sum_{t \in T} 1 - (\sum_{t \in T} w_q(t) - \delta)) + b$$

$$\leq r \sum_{t \in T} (1 - w_q(t)) + r\delta + b.$$

The right-hand side of this bound equals $r \sum_{t \in T} (1 - w_q(t)) + b'$ for $b' = r\delta + b$. We obtain by algebraic manipulations the following bound:

$$\begin{split} \sum_{t \in T} I_q^{\delta}(t) + \sum_{t \in T} w_q(t) &\leq r \sum_{t \in T} (1 - w_q(t)) + b' + \sum_{t \in T} w_q(t) \\ &= r \sum_{t \in T} 1 + (1 - r) \sum_{t \in T} w_q(t) + b' \\ &\leq r \sum_{t \in T} 1 + ((1 - r) \sum_{t \in T} \frac{\tau}{\tau + 1}) + b' \\ &= \left(\frac{r + \tau}{\tau + 1}\right) \sum_{t \in T} 1 + b' \\ &= \frac{r + \tau}{\tau + 1} \mid T \mid + b' \\ &= r' \mid T \mid + b' \;. \end{split}$$

In this derivation, we used $w_q(t) \leq \frac{\tau}{\tau+1}$ and $r' = \frac{r+\tau}{\tau+1}$. Observe that 0 < r' < 1.

Consider an adversarial pattern for injection rate r and burstiness b that results in unstable execution of the given scheduling policy. Based on the obtained estimate on

$$\sum_{t \in T} I_q^{\delta}(t) + \sum_{t \in T} w_q(t)$$

we define the corresponding adversarial pattern in the 2-priority model, as specified in the claim of the lemma. The constructed specific adversarial behavior follows the same injection pattern as defined by the adversary \mathcal{A} in \mathcal{DF}_{δ} , with a low priority given to all these packets, and additionally it injects a high priority packet at the starting queue q in each round t such that $w_q(t) = 1$.

Consider the execution of the original scheduling policy under the defined adversarial pattern in the 2-priority adversarial model. By the inequality

$$\sum_{t \in T} I_q^{\delta}(t) + \sum_{t \in T} w_q(t) \le r' |T| + b' ,$$

which holds in the execution in \mathcal{DF}_{δ} , we obtain that queue-congestion of the injected packets, whether of a high or low priority, is constrained by the injection rate r', with r' < 1, and burstiness b'_{v} . This is because $\sum_{t \in T} I_q^{\delta}(t)$ from the original execution in \mathcal{DF}_{δ} corresponds to the node congestion of the low-priority packets, and $\sum_{t \in T} w_q(t)$ corresponds to the queue-congestion of the high-priority packets, both in the 2-priority execution.

It remains to argue that the newly defined execution is also unstable in the 2-priority model. We observe that the following invariant holds:

Invariant:

There is at most one high-priority packet at a queue in any round in the 2-priority execution, and the transmissions of low-priority packets are the same in both considered executions.

This fact follows by induction on the round numbers. Any such a packet is injected in the beginning of each round when a extra round due to stalling occurs in the execution in \mathcal{DF}_{δ} through some queue. Since the execution in \mathcal{DF}_{δ} results in unbounded queues, the other one also does.

Theorem 2 Any scheduling policy that is universally stable in the 2-priority model is universally stable in \mathcal{DF}_{δ} .

Proof: Let us suppose it is otherwise, in order to arrive at a contradiction. This means that there is a scheduling policy S that is universal in the 2-priority model but for any burstiness b there is some rate r such that the inequalities 0 < r < 1 hold and such that S is unstable against the adversary with rate r and burstiness b in \mathcal{DF}_{δ} . Let us consider such an unstable execution. By Lemma 3, there is an unstable execution of the scheduling policy S in the 2-priority model, for some injection rate r' such that 0 < r' < 1 and for burstiness b'. This contradicts the universal stability of S in the 2-priority model.

Alvarez et al. [1] showed that the scheduling policies Farthest-To-Go (FTG), Nearest-From-Source (NFS) and Shortest-In-System (SIS) are universally stable for the 2-priority model. By this and Theorem 2, we obtain the following corollary.

Corollary 2 Scheduling policies FTG, NFS and SIS are all universally stable in the adversarial model \mathcal{DF}_{δ} .

5 Permanent Failures

In the previous sections it has been assumed that failures occur only in a transient manner. In this section, we extend the delayed feedback adversarial model (\mathcal{DF}_{δ}) to incorporate permanent failures into the model of packet injection. We denote such an extended model as $\mathcal{DF}_{\delta}^{e}$.

A scenario of failures. In \mathcal{DF}^e_{δ} we consider the case where, in addition to temporary failures, links may also fail in a permanent fail-stop manner, while nodes are not prone to faults. We assume that packets are never lost.

When a link q permanently fails, the adversary will be aware of that in a finite amount of time τ' . That can be modeled by using a notification of the form $fail_q(t)$ to be sent to the adversary when the link fails, which will receive it in at most τ' rounds.

We remark that $\mathcal{DF}_{\delta}^{e}$ allows scenarios where both permanent and temporal link failures occur simultaneously. Furthermore, a temporary faulty link could also fail in a permanent manner. For instance, if the link undergoes a transient fault for τ consecutive rounds, then the failure could be assumed to be permanent, in the sense that the link stays faulty forever.

An adversarial model. In addition to the behavior of the adversary in \mathcal{DF}_{δ} , in $\mathcal{DF}_{\delta}^{e}$ the adversary also reacts to occurrences of permanent link failures. This is specified as follows:

- Once the adversary receives the notification of a permanent failure, it stops injecting new packets to traverse the respective faulty link. Starting from such a notification, the adversary must choose such routes for injected packets that do not include any permanently faulty link.
- It could happen that packets injected prior to a permanent failure notification haven't crossed the failed link. In order to make it possible for these packets to reach their destinations, when a packet reaches a node such that the outgoing link to traverse is faulty, the adversary simply re-routes each such a packets using a new simple path. Note that in order to re-route a packet, it is necessary that the new path be able to reach the destination, which could impose some restrictions on the failure pattern. While the new path will not use the same link more than once, it could use links that have been previously used in the original path.

Furthermore, when the adversary re-routes a packet, it also *updates* the packet with the new path; namely, the concatenation of its original path up to the failed link, and the new path.

Let us make two observations at this point: First, at each queue, the scheduling policy will treat in the same way all packets, regardless of whether they are re-routed packets or not. Second, since re-routed packets are already inside the system, when the adversary re-routes a packet it does not take into account any additional admissibility condition, other than choosing a simple path.

Lemma 4 The maximum number of re-routed packets in \mathcal{DF}^e_{δ} is bounded, provided the used scheduling policy is universally stable in \mathcal{DF}_{δ} . **Proof:** On the one hand, since the scheduling policy is universally stable in \mathcal{DF}_{δ} , when a link fails in a permanent manner after failing in a temporary manner the number of waiting packets (at the queue of that link) that will be re-routed is guaranteed to be bounded. On the other hand, the adversary will stop injecting packets to pass through such a link after receiving the corresponding notification, which will occur in a bounded number of rounds. Then, the number of new injected packets that will be re-routed because of that failure is also bounded. Since the number of links is bounded, the overall number of packets that will be re-routed is bounded.

Now, we present our main result regarding permanent link failures in $\mathcal{DF}^{e}_{\delta}$.

Theorem 3 Any universally stable scheduling policy in \mathcal{DF}_{δ} remains universally stable in $\mathcal{DF}_{\delta}^{e}$.

Proof: Let us consider an arbitrary network and consider an adversarial system execution in $\mathcal{DF}_{\delta}^{e}$ using an universally stable scheduling policy in \mathcal{DF}_{δ} . Let us also consider that, in addition to the original notifications of delays, when a re-routed packet is transmitted at some time t at given queue q, a notification $w_q(t)$ is sent to the adversary. This means we consider the transmission of re-routed packets as failures.

Suppose the adversary does nothing upon the reception of the notifications for re-routed packets. This means that, at each round when the adversary receives a notification, the admissibility condition will allow the adversary to inject one more packet compared with the execution when the adversary reacts to these notifications.

This execution will be equivalent to an execution in \mathcal{DF}_{δ} where the adversary has a larger burst. Since by Lemma 4 the number of these notifications is bounded, then the increase in that burst is also bounded. Therefore, since this execution is stable in \mathcal{DF}_{δ} , it will also be stable in \mathcal{DF}_{δ}^e . \Box

On the use of re-routing for temporary failures. Since the use of re-routing has been proposed for permanent failures, a natural question that arises is whether or not such a technique could be used for temporary failures.

However, the use of re-routing with temporary faulty links presents some problems that may lead to instability. Indeed, re-routed packets are not subject to any admissibility condition regarding the links of the new paths. Therefore, if links disappear and appear over time, that could provoke an accumulation of re-routed packets at some queues, whose occupancy could grow unboundedly.

Lemma 5 If temporary faulty links are treated as permanent, any scheduling policy is unstable in $\mathcal{DF}_{\delta}^{e}$.

Proof: Let us consider the network topology in Figure 4 and assume that links fail in a temporary manner but are treated as permanent so this results in rerouting. Assume moreover that bursts of 10 packets are injected at each node e_i with destination f_i and passing through e'_i . Assume also that immediately after injecting these packets all links $e'_i \to f_i$ fail for 10 rounds, and packets are re-routed through link $h \to h'$. After all packets have been re-routed, faulty links recover and the process is repeated forever. It is easy to see that the buffer occupancy of link $h \to h'$ will grow unbounded.

Recovering from permanent failures. A consequence of the previously stated fact is that if permanent faulty links are allowed to recover then this could provoke instability, since a series



Figure 4: The network topology used in the proof of Lemma 5 to show that the use of re-routing with temporary faulty links may lead to instability.

of permanent failures and recoveries make the links behave as if experiencing transient faults. However, under some circumstances it is possible to allow permanent faulty links to recover and still guarantee stability.

Theorem 4 Any universally stable scheduling policy in $D\mathcal{F}^e_{\delta}$ remains universally stable when permanent faulty links are allowed to recover, provided each link recovers only after all packets that have been re-routed because of its failure have reached their destinations.

Proof: From Lemma 4, the number of re-routed packets is bounded up to the round when no permanent faulty link recovers. That means that the number of re-routed packets caused by the permanent failure of each link is also bounded. Since each link is allowed to recover only when all re-routed packets caused by the permanent failure of that link have reached their destination, the number of re-routed packets in \mathcal{DF}^e_{δ} remains bounded, by Lemma 4. Therefore, we can apply Theorem 3, which guarantees that the scheduling policy will remain universally stable.

6 Conclusion

We study routing in the suitable adversarial frameworks. We investigate how unexpected packet delays may affect routing's performance. Packet delays represent either malfunctioning of the network's infrastructure, implemented below the network layer, or transient unavailability of nodes due to energy saving policies.

We assume that routing protocols are embedded into flow control and connection control mechanisms. These mechanisms react to packet delays by spreading the suitable information through the network with the goal to decrease packet injection rates. We propose how to study stability of various classes of scheduling policies in such network settings. To this end, we propose a new adversarial model that has delays built into its machinery. The model we consider is an extension of the regular leaky-bucket model, which is determined only by the injection rate and burstiness.

Each transmission that fails results in feedback, which abstracts the flow control and connection control mechanisms. We treat this feedback as if given to the adversary, because it decreases the adversary's capability to inject packets.

We demonstrated that all scheduling policies stable in the 2-priority wireline adversarial model are also stable in the new proposed model. That includes such popular scheduling policies as FTG, NFS and SIS.

The model is flexible enough to cover permanent failures. We found that stable scheduling policies in the setting with temporal failures remain stable when some failures are permanent.

References

- C. Álvarez, M. J. Blesa, J. Díaz, M. J. Serna, and A. Fernández. Adversarial models for priority-based networks. *Networks*, 45(1):23–35, 2005.
- [2] C. Alvarez, M. J. Blesa, and M. J. Serna. The robustness of stability under link and node failures. *Theoretical Computer Science*, 412(50):6855–6878, 2011.
- [3] L. Anantharamu and B. S. Chlebus. Broadcasting in ad hoc multiple access channels. *Theoretical Computer Science*, 584:155–176, 2015.
- [4] L. Anantharamu, B. S. Chlebus, D. R. Kowalski, and M. A. Rokicki. Packet latency of deterministic broadcasting in adversarial multiple access channels. *Journal of Computer and System Sciences*, 99:27–52, 2019.
- [5] M. Andrews, B. Awerbuch, A. Fernández, F. T. Leighton, Z. Liu, and J. M. Kleinberg. Universal-stability results and performance bounds for greedy contention-resolution protocols. *Journal of the ACM*, 48(1):39–69, 2001.
- [6] M. Andrews and L. Zhang. Routing and scheduling in multihop wireless networks with timevarying channels. ACM Transactions on Algorithms, 3(3):article number 33, 2007.
- [7] B. Awerbuch, A. W. Richa, and C. Scheideler. A jamming-resistant MAC protocol for singlehop wireless networks. In *Proceedings of the 27th ACM Symposium on Principles of Distributed Computing (PODC)*, pages 45–54, 2008.
- [8] M. A. Bender, M. Farach-Colton, S. He, B. C. Kuszmaul, and C. E. Leiserson. Adversarial contention resolution for simple channels. In *Proceedings of the 17th ACM Symposium on Parallel Algorithms and Architectures (SPAA)*, pages 325–332, 2005.
- [9] V. Bhandari and N. H. Vaidya. Reliable broadcast in wireless networks with probabilistic failures. In Proceedings of the 26th IEEE International Conference on Computer Communications (INFOCOM), pages 715 – 723, 2007.

- [10] V. Bhandari and N. H. Vaidya. Reliable broadcast in radio networks with locally bounded failures. *IEEE Transactions on Parallel and Distributed Systems*, 21(6):801–811, 2010.
- [11] M. J. Blesa, D. Calzada, A. Fernández, L. López, A. L. Martínez, A. Santos, M. J. Serna, and C. Thraves. Adversarial queueing model for continuous network dynamics. *Theory of Computing Systems*, 44(3):304–331, 2009.
- [12] A. Borodin, J. Kleinberg, P. Raghavan, M. Sudan, and D. Williamson. Adversarial queueing theory. *Journal of the ACM*, 48(1):13–38, 2001.
- [13] A. Borodin, R. Ostrovsky, and Y. Rabani. Stability preserving transformations: Packet routing networks with edge capacities and speeds. *Journal of Interconnection Networks*, 5(1):1–12, 2004.
- [14] B. S. Chlebus, V. Cholvi, and D. R. Kowalski. Stability of adversarial routing with feedback. In Proceedings of the First International Conference on Networked Systems (NETYS), volume 7853 of Lecture Notes in Computer Science, pages 206–220. Springer, 2013.
- [15] B. S. Chlebus, V. Cholvi, and D. R. Kowalski. Stability of adversarial routing with feedback. *Networks*, 66(2):88–97, 2015.
- [16] B. S. Chlebus, D. R. Kowalski, and M. A. Rokicki. Maximum throughput of multiple access channels in adversarial environments. *Distributed Computing*, 22(2):93–116, 2009.
- [17] B. S. Chlebus, D. R. Kowalski, and M. A. Rokicki. Adversarial queuing on the multiple access channel. ACM Transactions on Algorithms, 8(1):article number 5, 2012.
- [18] V. Cholvi. Stability bounds in networks with dynamic link capacities. Information Processing Letters, 109(2):151–154, 2008.
- [19] K. Christensen, P. Reviriego, B. Nordman, M. Bennett, M. Mostowfi, and J. A. Maestro. IEEE 802.3az: The road to energy efficient Ethernet. *IEEE Communications Magazine*, 48(11):50– 56, 2010.
- [20] S. Gilbert, R. Guerraoui, D. R. Kowalski, and C. Newport. Interference-resilient information exchange. In *Proceedings of the 28th IEEE International Conference on Computer Communications (INFOCOM)*, pages 2249–2257, 2009.
- [21] S. Gilbert, R. Guerraoui, and C. C. Newport. Of malicious motes and suspicious sensors: On the efficiency of malicious interference in wireless networks. *Theoretical Computer Science*, 410(6-7):546–569, 2009.
- [22] IEEE P802.3az Energy Efficient Ethernet. Task force public area. http://grouper.ieee.org/groups/802/3/az/public/index.html, 2008.
- [23] D. Koukopoulos, M. Mavronicolas, and P. G. Spirakis. The increase of the instability of networks due to quasi-static link capacities. *Theoretical Computer Science*, 381(1-3):44–56, 2007.
- [24] D. Koukopoulos, M. Mavronicolas, and P. G. Spirakis. Performance and stability bounds for dynamic networks. *Journal of Parallel and Distributed Computing*, 67(4):386–399, 2007.

- [25] S. Lim, K. Jung, and M. Andrews. Stability of the max-weight protocol in adversarial wireless networks. In Proceedings of the 31st IEEE International Conference on Computer Communications (INFOCOM), pages 1251–1259, 2012.
- [26] L. Mamatas, T. Harks, and V. Tsaoussidis. Approaches to congestion control in packet networks. *Journal of Internet Engineering*, 1(1):22–33, 2007.
- [27] D. Meier, Y. A. Pignolet, S. Schmid, and R. Wattenhofer. Speed dating despite jammers. In Proceedings of the 5th IEEE International Conference on Distributed Computing in Sensor Systems (DCOSS), volume 5516 of Lecture Notes in Computer Science, pages 1–14. Springer, 2009.
- [28] R. Srikant. The Mathematics of Internet Congestion Control. Birkhäuser, 2004.
- [29] J. S. Turner. New directions in communications (or which way to the information age?). IEEE Communications Magazine, 40(5):50–57, 2002.