Adversarial models for priority-based networks*

C. Àlvarez† M. Blesa† J. Díaz† A. Fernández‡ M. Serna†

Abstract

We propose several variations of the adversarial queueing model. The *priority* model takes into account the case in which the packets can have different priorities, assigned by the adversary at injection time. The *variable priority* model is an extension of the priority model in which the adversary may change the priority of packets at each time step. The *failure* and *reliable* models are designed to cope with dynamic networks in which the adversary controls, under different constraints, the edge failures.

We address stability issues in the proposed adversarial models. We show that the set of *universally stable* networks in the adversarial model remains the same in the priority, variable priority, failure and reliable models.

From the point of view of queueing policies we show that several queueing policies that are universally stable in the adversarial model remain so in the priority, failure and reliable models. However, we show that LIS, a universally stable queueing policy in the adversarial model, is not universally stable in any of the other models. Moreover, we show that no greedy queueing policy is universally stable in the variable priority model.

Finally we analyze the problem of deciding stability of a given network under a fixed protocol. We provide a characterization of the networks that are stable under FIFO and LIS in the failure model (and therefore in the reliable and priority models). This characterization allows us to show that the stability problem under FIFO and LIS in the failure model can be solved in polynomial time.

1 Introduction

The model of Adversarial Queueing Theory (AQT) proposed by Borodin et al. [9] considers the time evolution of a packet-routing network as a game between an adversary and a queue policy. At each time step the adversary may inject a set of packets to some of the nodes. For each packet the adversary specifies the sequence of edges that it must traverse, after which the packet will be absorbed. If more than one packet try to cross an edge e at the same time step, then the queueing policy chooses one of these packets to be sent across e. The remaining packets wait in the queue. This game then advances to the next time step. The main goal of the model is to study *stability issues* of the network, under different *greedy* queueing policies. Stability is the property of deciding whether at any time the maximum number of packets present in the system is bounded by a constant that may depend on system parameters. Recall that a protocol is *greedy* if whenever there is at least one packet waiting to use an edge, the protocol advances a packet through the edge.

In the *adversarial* model the adversary is restricted by a pair (r, b) where $b \geq 0$ is *the burstiness*, and $0 < r < 1$ is the *injection rate*. The adversary must obey the following rule [9, 4],

$$N_e(I) \leq r|I| + b, \tag{1}$$

*Work partially supported by the FET Programme of the EU under contract number IST-2001-33116 (FLAGS), and by the Spanish CICYT project TIC-2001-4917-E. The second author was also supported by the Catalan government with the pre-doctoral grant 2001FI-00659.

†Dept. Llenguatges i Sistemes Informàtics, Universitat Politècnica de Catalunya. Jordi Girona 1–3, 08034 Barcelona, Spain. {alvarez,mjblesa,diaz,mjserna}@lsi.upc.es

‡Dept. Ciencias Experimentales e Ingeniería, Universidad Rey Juan Carlos. Tulipan s/n, Campus de Móstoles, 28933 Madrid, Spain anto@gsyc.escet.urjc.es

where $N_e(I)$ denotes the number of packets injected by the adversary during a time interval I that have paths containing the edge e[1] .

In this paper we consider a generalization of the adversarial model which takes into account the possibility that packets may have different priorities and we explore some dynamic network settings.

Priority models Considering priorities is a natural approach to model nowadays' networks. Today's networked applications, such as data mining, e-commerce and multimedia, are bandwidth hungry and time sensitive. These applications need networks that accommodate to these requirements and guarantee some *Quality of Service* (QoS). Classifying and prioritizing network traffic flow is a basic task to fulfill these goals. In nowadays' networks this is supported by mapping into the packets information about the type and the priority of their contents. Internet Protocol (IP) supports QoS in the IP header of the packets, e.g. by using the IP Precedence/ToS field. The current Internet protocol IPv4 uses 3 bits of the *Type of Service* (ToS) byte to designate the IP Precedence. Eight priorities are possible, from 0 (default) to 7; the higher the number, the higher the priority. Then different queues are supported (*priority queueing*), which are serviced in strict order of queue priority. With these services, the routers and switches can police the traffic entering the network, assign priority, and ensure optimal paths through the IP network.

We are interested in analyzing the power of an adversary that can prioritize the packets. We will consider two settings, in the first one packets have fixed priority and in the second one the adversary is allowed to modify the priority of a packet. Consequently, we define two new models for adversarial queueing theory, the *priority* and the *variable priority*. When packets have priorities, each edge has a queue associated to every possible priority. If at a certain time, more than one packet tries to cross the same edge e, the queueing policy chooses one of these packets to send across e, from the non-empty queue with highest priority. The limitations on the adversary are the same as in the adversarial model.

In the priority model the priority of a packet is fixed at injection time and never changes, while in the variable priority model at each time step the adversary can decide the priority of a packet. We will assume that the number of possible priorities that the adversary can use is pre-fixed.

Models for dynamic networks Inspired by the priority models and by the growing importance of wireless mobile networks, where some connections between nodes may fail or change quickly and unpredictably. We also consider some variations of the adversarial model for *dynamic networks*, in which edges can appear and disappear arbitrarily. Note that in the priority model, we can *simulate* the failure of an edge e by injecting a packet of length 1 in e with a priority higher than any other packet in the queue of e. Once this packet has been injected none of the remaining packets of the queue can be served until the next time step. It seems natural to introduce models for dynamic networks in which the adversary controls not only the packet arrivals, but also the edge failures. At any time step, the adversary can produce the failure of an edge. If an edge e fails, the packets in e's queue wait until the moment when the edge recovers. The constraints of an adversary are defined taking into consideration the number of time steps during any interval I that the edge e is down. We assume that a packet cannot cross a link when it fails, and that during an edge failure the packets that arrive wait at the queue a the head of the edge.

Let $F_e(I)$ be the number of steps during time interval I for which the edge e is down. We propose suitable restrictions on the adversary, obeying the rule: the number of packets introduced by the adversary during interval I, which have paths containing e, can not be greater than the number of times that e is alive in this interval. Furthermore, as packets must follow a pre-specified path, the adversary should not be able to make an edge fail perpetually. To guarantee that we keep a bound on the maximum number of failures of an edge e in any time interval, we propose a new model, the *failure* model, in which the adversary is controlled by a common bound on both

[1] Recall that in [9] the model is defined over windows of fixed size w and the equation $N_e(I) \leq r|I|$, for $|I| = w$. It is known that both models are equivalent [15]

the injection and the edge failures, according to the restriction

$$N_e(I) + F_e(I) \leq r|I| + b. \tag{2}$$

Observe than in this case, during a given interval, the injection rate limits the maximum number of failures and the maximum number of packet injections per edge.

With the aim of allowing a higher amount of edge failures, we define a new dynamic model. To do so, we introduce an additional parameter α. The adversary is characterized by (r, b, α) , where r, b are defined as before and $r \leq \alpha \leq 1$. For any edge e and any interval I, the adversary must obey the constraint

$$N_e(I) + \alpha F_e(I) \leq r|I| + b. \tag{3}$$

In the model defined by equation (3), we can consider two extreme cases: For $\alpha = 1$ we obtain the constraint (2) defining the failure model. For $\alpha = r$ we get a model in which an edge can be down all the time. Notice that in the case when $r < \alpha \leq 1$, if the adversary produces a failure of an edge, then it is forced to recover the edge after $b/(\alpha - r)$ steps, otherwise it will violate inequality (3). We are interested in the latter property and will use the term *reliable model* to denote an adversary with parameters (r, b, α) where $0 < r < 1$, $b > 1$, and $r < \alpha \leq 1$ and constrained by inequality (3).

Greedy protocols Through this paper the term *network* will refer to digraphs which may have multiple edges but no loops. As in [9, 4] we will only consider greedy protocols which apply their policies to the queues at the edges according to some local or global criteria. The systems acts synchronously. The main queueing protocols we consider are: FIFO, LIFO, SIS, LIS, NTG, FTG, NFS and FFS.

The protocol LIFO (Last In First Out) gives priority to the packet which joined later the edge; in FIFO (First In First Out), precedence is given to the packet that has arrived first in the queue of the edge. The protocol which gives precedence to the packet last introduced into the system is SIS (Shortest In System), while in LIS (Longest In System) every queue gives precedence to the packet that has been in the system the longest time. The protocol NTG (Nearest To Go) assigns precedence to the packet that is closest to its destination and FTG (Furthest To Go) selects the packet that is farthest away from its destination. NFS (Nearest From Source) and FFS (Farthest From Source) consider the same policies but taking the distance to the source node of the packets as reference point. NFS is sometimes called NTS (Nearest To Source). Aside from these protocols, we also consider the NTG-LIS protocol, which works as NTG, but resolving ties using the LIS protocol.

It is known that FTG, NTS, SIS and LIS are universally stable in the adversarial model while FIFO, LIFO, NTG and FFS are not [4].

Related work Two adversarial models for dynamic networks have been proposed in [6] and [5]. In both models the injected packets are defined by specifying only source and destination, and thus are not forced to follow a pre-specified path. In both cases the adversary is restricted to guarantee that a static multi-commodity flow problem has a solution. Stability results are obtained using a load balancing algorithm, for the case that the adversary injection rate is one, and the packets have a unique common destination. The main difference in both models is that in [6] the adversary has to provide a solution to the associated multi-commodity flow problem, while in [5] the injection pattern must obey a condition that guarantees the existence of the solution.

The dynamic models proposed in this paper consider the case in which the injected packets are defined specifying the sequence of edges that they must traverse. Our models and the dynamic models proposed in [6] and in [5] have the common characteristic that, for every interval I, the adversary can not inject to any edge e (or to any set S of nodes for the model in [5]), more packets than the number of packets that e can absorb (or the edges with only one extreme in S).

An interpretation of system failure as a *slowdown* in the transmission or *link capacity*, instead of link failure was studied in [8], in both models packets are injected with a pre-specified path.

In the introduced *slowdown model*, packets are injected with a pre-specified path and might suffer delays while crossing a link. A packet suffers delay d while crossing edge e if it starts to traverse the link at time t and arrives to the tail of e at time $t+d$. During this transfer time the packets that want to cross e wait in the queue of e. The paper considers two cases, the *static* case in which every link e has a fixed slowdown s_e that gives the delay incurred by any packet that crosses e, and the *dynamic* case in which the slowdown in link e depends of the time, that is, if a packet starts to cross e at time t it suffers delay $s_e(t)$.

In the static slowdown, the adversary in is controlled by a pair $(r, b)^2$ and

$$N_e(I) \leq r\frac{|I|}{s_e} + b, \tag{4}$$

whereas in the dynamic slowdown model, for any edge e and any time step t', the adversary is controlled by a pair (r, b) and

$$N_e(I) \leq r\sum_{t \in I} \frac{1}{\overline{s}_e(t)} + b, \tag{5}$$

where for any edge e and any time step t', define $\overline{s}_e(t') = \max_t\{s_e(t) | t \leq t' \leq t + s_e(t)\}$.

This situation has some similarities with the failure model. In the static case, the slowdown s_e of an edge e can be interpreted as "link e *fails* during $s_e - 1$ steps". However, there is a difference, in the slowdown model a packet p is delayed after leaving e's queue, while in the failure model p waits in e's queue. This means that when e is recovered, the next packet to be served might not be p.

In the *capacity* model every edge e in a network has capacity $c_e(t)$ at time step t. They also consider a static case in which the capacity does not depend on time, $c_e(t) = c_e$. In step t a link is able to transmit simultaneously up to $c_e(t)$ packets. The adversary is controlled by a pair (r, b) and

$$N_e(I) \leq r|I|c_e(t) + b. \tag{6}$$

The main results in the paper are that every universally stable network remains universally stable in the slowdown and the capacity models, even in the dynamic case. That SIS, NTS and FTG remain universally stable in all the models. The situation is different for LIS since it is universally stable in the static slowdown model but it is not in the dynamic slowdown and capacity models. Even though we can interpret that a link fails at any time step with $c_e(t) = 0$ the proof that LIS is not universally in the dynamic capacity model uses two non zero capacities (see Theorem 3.1 [8]).

Our contributions We address stability issues in the proposed adversarial models. Let us recall that a network is *stable under a protocol and an adversary* if the number of packets in the system at any time step remains bounded.

Our first results concern with *universal stability* of networks. First we show that the property that a network is stable under any adversary and queueing policy remains the same in the adversarial, priority, variable priority, failure and reliable models.

From the point of view of queueing policies, we show that NFS, SIS and FTG, that are universally stable in the adversarial model [4], remain so in the failure, reliable, and priority models. However, we show that LIS, a universally stable queueing policy in the adversarial model [4], is not universally stable in the failure, reliable and priority models. Moreover, we show that no greedy protocol is universally stable in the variable priority model.

Finally we analyze the problem of deciding stability of a given network under a fixed protocol. We provide a characterization of the networks that are stable under FIFO and LIS in the failure model. This characterization is the same as the one given in [2] in the adversarial model for universal stability and for stability under NTG-LIS. Thus, our results show that for FIFO and LIS the stability problem in the failure model can be solved in polynomial time. Let us observe that the characterization of FIFO stability in the adversarial model remains an open problem [3].

[2] Although in [8] the adversary is restricted in the window model for simplicity we express it here in the (r, b) model.

2 Universal stability of networks

A communication system is formed by three main components: A network G, a scheduling protocol $\mathcal{P}$ and a traffic pattern $\mathcal{A}$ which is represented by an adversary. The concept of *universal stability* applies either to networks or protocols.

Let $\mathcal{M}$ denote a model in the set {*adversarial*, *reliable*, *failure*, *priority*, *variable priority* } as defined in the previous section. For all of them, we can formally define different concepts around universal stability.

Given a network G, a queuing policy $\mathcal{P}$ and a model $\mathcal{M}$, we say that for a given adversary $\mathcal{A}$ following the restrictions of $\mathcal{M}$, the system $S = (G, \mathcal{A}, \mathcal{P})$ is *stable in the model* $\mathcal{M}$ if, at any time step, the maximum number of packets in the system is bounded by a fixed value that may depend on system parameters. The pair $(G, \mathcal{P})$ is *stable in the model* $\mathcal{M}$ if, for any adversary $\mathcal{A}$ following the restrictions of $\mathcal{M}$, the system $S = (G, \mathcal{A}, \mathcal{P})$ is stable in $\mathcal{M}$. A network G is *universally stable in the model* $\mathcal{M}$ if, for any greedy queuing policy $\mathcal{P}$, the pair $(G, \mathcal{P})$ is stable in $\mathcal{M}$. A greedy protocol $\mathcal{P}$ is *universally stable in the model* $\mathcal{M}$ if, for any digraph G, the pair $(G, \mathcal{P})$ is stable in $\mathcal{M}$.

We start by comparing the relative power of the adversaries in the different proposed models.

We say that adversary $\mathcal{A}$ in model $\mathcal{M}$ *simulates* adversary $\mathcal{A}'$ in model $\mathcal{M}'$ when for any network G and any protocol $\mathcal{P}$ if $(G, \mathcal{A}, \mathcal{P})$ is stable in $\mathcal{M}$, then $(G, \mathcal{A}', \mathcal{P})$ is stable in $\mathcal{M}'$.

Lemma 1
1. Any adversary in the adversarial model with parameters (r, b) can be simulated by an adversary in the failure model with parameters (r, b).
2. Any adversary in the failure model with parameters (r, b) can be simulated by an adversary in the reliable model with parameters $(r, b, 1)$.
3. Any adversary in the failure model with parameters (r, b) can be simulated by an adversary in the priority model with parameters (r, b) and two different priorities.
4. Any adversary in the reliable model with parameters (r, b, α) can be simulated by an adversary in the failure model with parameters $(r + 1 - \alpha, b)$.

Proof.
1. Any adversary that obeys (1) can be seen as an adversary constrained by (2) that never forces an edge failure.
2. By definition.
3. The priority model adversary $\mathcal{A}'$ will inject packets with two priorities, *low* and *high*. At each time step, $\mathcal{A}'$ injects with low priority the same packets that $\mathcal{A}$ injects. If an edge e fails, $\mathcal{A}'$ injects a packet (e) with *high* priority. Observe that the parameters (r, b) for $\mathcal{A}$ are also valid for $\mathcal{A}'$.
4. Given (r, b, α), with $r < \alpha \leq 1$, an adversary in the reliable model obeys the constraint (3). Since $\alpha \leq 1$, $N_e(I) + F_e(I) \leq r|I| + (1 - \alpha)F_e(I) + b$. Moreover, as $F_e(I) \leq |I|$,

$$N_e(I) + F_e(I) \leq r|I| + (1 - \alpha)|I| + b \leq (r + 1 - \alpha)|I| + b. \tag{7}$$

Thus the adversary obeys the restriction (2) of the failure model with $r' = 1 + r - \alpha < 1$. ■

Observe that the failure and reliable models are equivalent. So any stability or instability result for one model applies to the other as well. We will consider in the following only the failure and priority models.

Now we can state our main result in this section.

Theorem 2 *Given a digraph G, the following properties are equivalent*
1. *G is universally stable in the adversarial model,*
2. *G is universally stable in the failure model,*
3. *G is universally stable in the priority model, and*
4. *G is universally stable in the variable priority model.*

Proof.

1 $\Rightarrow$ **4** For any system $S = (G, \mathcal{A}, \mathcal{P})$ in the variable priority model, we define the following system $S' = (G, \mathcal{A}', \mathcal{P}')$ in the adversarial model: At each time step, $\mathcal{A}'$ injects the same packets that $\mathcal{A}$ injects (without priority). The new queue policy $\mathcal{P}'$ will determine the packet to be served, simulating the system S.
4 $\Rightarrow$ **3** by definition.
3 $\Rightarrow$ **2** and **2** $\Rightarrow$ **1** follow from Lemma 1.
■

3 Universal stability of protocols

In this section we address the universal stability property in the failure and priority models, from the point of view of the queuing policy. We will consider the six basic protocols presented in the introduction. Recall that FTG, NTS, SIS and LIS are universally stable in the adversarial model while FIFO, LIFO, NTG, and FFS are not [4]. Since any adversary in the adversarial model can be seen as an adversary in the other models, FIFO, LIFO, NTG, and FFS are not universally stable in the failure and priority models.

First we show how any adversary with parameters (r, b) in the priority model can be simulated by an adversary in the adversarial model with the same injection rate (r, b), by changing slightly the network, but preserving the same protocol $\mathcal{P}$ in the case that $\mathcal{P} \in \{\text{FTG}, \text{NTG}, \text{NFS}, \text{FFS}\}$.

Let G be a directed graph and A_π an adversary in the priority model, assigning at most π priorities and with injection rate (r, b), where $b \geq 0$ and $0 < r < 1$. Every injected packet p has a priority π_p in the ordered interval $[1, \ldots, \pi]$, being one the lowest priority.

Lemma 3 *For any system $S = (G, \mathcal{A}_\pi, \mathcal{P})$ in the priority model, for $\mathcal{P} \in \{\text{FTG}, \text{NTG}\}$, there is a system $S' = (G', \mathcal{A}', \mathcal{P})$ in the adversarial model such that,*

- *G is a subgraph of G',*
- *$\mathcal{A}'$ has injection rate (r, b),*
- *if a packet p is injected in S at time t with route r, a packet p' is injected in S' at time t with a route r' obtained by concatenating r and a path of edges not in G, and*
- *if p crosses edge e at time t' in S, p' crosses e at time t' in S'.*

Proof. We first consider FTG and construct G' by attaching to every node of G as many outgoing disjoint paths as edges are incident to the node. Each outgoing path will be associated to a different incoming edge and has length $(\pi - 1)d$, where d is the length of the longest directed path in G and π is the number of different priorities used by $\mathcal{A}_\pi$.

Then each packet p injected in S with priority π_p and route r is replaced in S' by a packet p' that first follows the same route r and then follows $(\pi_p - 1)d$ edges of the outgoing path associated to the last edge of r. Observe that this packet, while it remains in the queues of the original network G still has at least $(\pi_p - 1)d + 1$ edges to cross. Similarly, in those queues, any packet q' that replaced a packet q with priority $\pi_q < \pi_p$ still has to cross at most $\pi_q d < (\pi_p - 1)d + 1$ edges, and hence will always be blocked by p'. Similarly, any packet that replaced one with higher priority than p will always block p'.

The proof for NTG is similar, but in this case p' has to cross $(\pi - \pi_p)d$ edges of the outgoing path. ■

As a consequence of the previous theorem, and the results in [4] we get,

Theorem 4 FTG *is universally stable in the priority, failure and reliable models.*

To get a similar result for NFS and FFS we have to relate two different networks.

Lemma 5 *For any system $S = (G, \mathcal{A}_\pi, \mathcal{P})$ in the priority model, for $\mathcal{P} \in \{\text{NFS}, \text{FFS}\}$, there is a system $S' = (G', \mathcal{A}', \mathcal{P})$ in the adversarial model such that,*

- *G is a subgraph of G',*
- *$\mathcal{A}'$ has injection rate (r, b'), where $b' = r(\pi - 1)d + b$,*
- *if a packet p is injected in S with route r, a packet p' is injected in S' with a route r' obtained by concatenating and a path of edges not in G and r, and*
- *if p crosses edge e at time t in S, p' crosses e at time t in S'.*

Proof. We first consider NFS and construct G' by attaching to every node of G incoming disjoint paths as follows. For each node s, each edge e leaving s, and each priority $\pi_p \in \{1, ..., \pi - 1\}$, G' will have b disjoint incoming paths associated to the pair (e, π_p) of length $(\pi - \pi_p)d$ whose last node is s.

Then each packet p injected in S with priority $\pi_p < \pi$ at time t in the source node s and with edge e as the first edge in its route is replaced in S' by a packet p' injected in the first node of one of the b incoming paths associated to (e, π_p) at time $t - (\pi - \pi_p)d$. If several packets are injected in S at the same time with the same initial edge e and the same priority π_p, each uses a different disjoint incoming path from those associated to (e, π_p). Notice that p' follows the incoming path and then the same route as p. A packet p with priority $\pi_p = \pi$ is replaced in S' by a packet p' with the same injection time and route as in S.

Observe that any packet p' in S', while in the queues of the original network G has already crossed at least $(\pi - \pi_p)d$ edges. Similarly, in those queues, any packet q' that replaced a packet q with priority $\pi_q > \pi_p$ has crossed at most $(\pi - \pi_q)d + d - 1 < (\pi - \pi_p)d$ edges, and hence will always block p'. Similarly, any packet that replaced one with lower priority than p will always be blocked by p'.

Note that in S' a packet p' can be injected up to $(\pi - 1)d$ steps before the packet p it is replacing was injected in S. Hence, in any time interval I in S' the maximum number of injections for each edge is $r|I| + b + r(\pi - 1)d$. The first part $r|I| + b$ corresponds with the packets injected in I in S, and the second part $r(\pi - 1)d$, with the packets of low priority that have been injected early in S'.

The proof for FFS is similar, but in this case priority π_p is associated with incoming paths of length $(\pi_p - 1)d$. ■

As a consequence of the previous theorem, and the results in [4] we get,

Theorem 6 NFS *is universally stable in the priority, failure and reliable models.*

In order to show the universal stability of SIS in the priority model we follow similar arguments to that of Lemma 2.2 in [4] for showing the universal stability of SIS in the adversarial model .

Lemma 7 *For any network G and any adversary $\mathcal{A}_\pi$, the system $(G, \mathcal{A}_\pi, \text{SIS})$ is stable under the priority model.*

Proof. The proof is by induction on the number of priorities. When $\pi = 1$ the SIS protocol was shown in [4] to be universally stable under the adversarial model.

Let us assume that the system $(G, \mathcal{A}_{\pi-1}, \text{SIS})$ is stable. Observe that in $(G, \mathcal{A}_\pi, \text{SIS})$ the system formed by the packets with $\pi_p > 1$ are independent of the packets with priority 1, and therefore they form an stable system. Let S be the maximum number of packets in the system requiring any particular edge, with priority bigger than 1, at any time.

Based on [4], we define the following recurrence.

$$K_1 = b + S \quad ; \quad K_{j+1} = \frac{K_j + b}{1 - r}$$

Let d be the length of the longest simple path in G. By a similar argument to that of Lemma 2.2 in [4], but replacing b by $b + S$ in the base case of the induction, we can show that when a packet p, with priority one, arrives at the queue of the jth edge e_j on its path there are at most $K_j - 1$ packets requiring any edge e in the path of p with priority over p.

Then K_d denotes the maximum number of packets with priority one, in any queue. Therefore the system with π priorities is stable. ■

Theorem 8 SIS *is universally stable in the priority, failure and reliable models.*

Even though the universal stability property of FTG, NFS, and SIS in the adversarial model is preserved in the priority model, we will see that this is not the case for LIS which is universally stable in the adversarial. The next result states the non universal stability of LIS in the failure model, and therefore in all the other models. We consider graph U_1 of Figure 1(a) and we show that U_1 is not stable under LIS in the failure model. The proof is given in the appendix as Lemma 22. Putting this result together with Lemma 1 we get,

Theorem 9 LIS *is not universally stable in the failure, reliable and priority models.*

Finally, we consider universal stability of protocols in the variable priority model. In this case the adversary is so powerful that no greedy protocol is universally stable. In Lemma 17 of the appendix we show that for any greedy protocol $\mathcal{P}$, the pair $(U_1, \mathcal{P})$ is not stable in the variable priority model. Hence we can state,

Theorem 10 *There is no greedy protocol universally stable in the variable priority model*

4 Stability under a protocol

In this section we analyze the complexity of the problem of deciding whether a given network G is stable under a fixed protocol. Few results are known for this problem. Deciding stability under NTG-LIS is polynomially solvable [2]. The polynomial time decidability of stability under FFS in the case that the adversary can solve ties arbitrarily was shown in [1]. Further results for undirected graphs and other variations can be found in [4] and [3]. Our results are obtained through the same characterization, using a basic graph family an the exclusion of a set of forbidden subgraphs.

Before formally stating our results, we need to introduce some graph theoretical definitions. We will consider the following subdivision operations over digraphs:

- The *subdivision of an arc* (u, v) in a digraph G consists in the addition of a new vertex w and the replacement of (u, v) by the two arcs (u, w) and (w, v).
- The *subdivision of a 2-cycle* (u, v), (v, u) in a digraph G consists in the addition of a new vertex w and the replacement of (u, v), (v, u) by the arcs (u, w), (w, u), (v, w) and (w, v).

Given a digraph G, $\mathcal{E}(G)$ denotes the family of digraphs formed by G and all the digraphs obtained from G by successive arc or 2-cycle subdivisions. Given a family of digraphs $\mathcal{F}$, $\mathcal{S}(\mathcal{F})$ denotes the family of digraphs that contain a graph in $\mathcal{F}$ as a subgraph.

Figure 1(a) provides the two basic graphs needed to characterize universal stability, and Figure 1(b) gives the shape of the extensions of those graphs. This basic family provides the characterization of the stability properties. It is known that a digraph is universally stable in the adversarial model if and only if $G \notin \mathcal{S}(\mathcal{E}(U_1) \cup \mathcal{E}(U_2))$ [2] . The same property characterizes network stability under NTG-LIS [2] and FFS [1]. It is also known that, for a given digraph G, checking whether $G \notin \mathcal{S}(\mathcal{E}(U_1) \cup \mathcal{E}(U_2))$ can be done in polynomial time [2].

Nothing is known about the complexity of deciding stability in the adversarial model for any other queueing policy. In the following we will provide a similar characterization of stability in the failure model under FIFO and LIS.

4.1 FIFO stability under the failure model

The FIFO protocol gives precedence to the packet that arrived first to the queue it schedules. One of the first results relating the adversarial model and this protocol was done in [4], where the FIFO protocol was shown not to be universally stable.

Much effort have been devoted to study stability and instability properties of FIFO recently. A network-dependent absolute constant is provided in [11] such that FIFO is stable against any adversary with smaller injection rate. A lower bound of 0.749 for the instability is calculated in [12]. This bound was decreased to 0.5 [14]. In [10] it is shown that FIFO is stable if the injection rate is smaller than $1/(d-1)$. Recently, it has been proved that FIFO can become unstable at arbitrarily low injections rates [7, 13].

We show that the two basic graphs given in Figure 1(a), as well as their extensions (see Figure 1(b)), are not stable under FIFO in the failure model. As a more general result, for network U_2 we show instability in the adversarial model. The corresponding proofs can be found in the appendix.

Lemma 11 *Any graph in $\mathcal{E}(U_1) \cup \mathcal{E}(U_2)$ is not stable under* FIFO *in the failure model.*

As we have pointed out before, all networks $G \notin \mathcal{S}(\mathcal{E}(U_1) \cup \mathcal{E}(U_2))$ are universally stable in the adversarial model. Taking into account that if a network has an unstable subnetwork it is also unstable we get the following result.

Theorem 12 *Let G be a digraph, the pair (G, FIFO) is stable in the failure model if and only if G is universally stable in the adversarial model.*

A corollary of this result is the equivalence between FIFO stability in the failure model and universal stability in the adversarial model. Furthermore, as instability in the failure model implies instability in the priority and reliable models, the characterization of FIFO stability remains the same in the priority and reliable models. Observe also that stability under FIFO can be checked in polynomial time for the failure, priority and reliable models.

4.2 LIS stability under the failure model

The LIS protocol gives priority to the packet that was longer in the system, i.e., that joined the network earlier. In [4], the LIS protocol was shown to be universally stable in the adversarial model, with $O(b/(1-r)^d)$ queue size per edge and delay of the packets in the order $O(b/(1-r)^d)$. However, as we have shown the protocol is not universally stable in the failure model. Now we will show that deciding stability under LIS in the failure model can be performed in polynomial time.

We proceed as in the case of FIFO by showing, respectively, the instability of the basic graphs given in Figure 1(a) and their extensions in Figure 1(b). The corresponding proofs can be found in the appendix.

Lemma 13 *Any graph in $\mathcal{E}(U_1) \cup \mathcal{E}(U_2)$ is not stable under* LIS *in the failure model.*

Therefore, as in the case of FIFO, we have

Theorem 14 *A digraph G is stable under* LIS *in the failure model if and only if G is universally stable in the adversarial model.*

Again, stability under LIS in the failure, reliable and priority models coincides with universal stability in the adversarial model, and therefore it can be checked in polynomial time for the failure, priority and reliable models.

4.3 The variable priority model

For the variable priority model the situation is simpler. It is easy to adapt the proof of Lemma 19 to get instability results for the graph U_2. This together with Lemma 17 gives the following result.

Lemma 15 *Let $\mathcal{P}$ be any greedy protocol. Any graph in $\mathcal{E}(U_1) \cup \mathcal{E}(U_2)$ is not stable under $\mathcal{P}$ in the variable priority model.*

Therefore, we have

Theorem 16 *Let $\mathcal{P}$ be any greedy protocol. A digraph G is stable under $\mathcal{P}$ in the failure model if and only if G is universally stable in the adversarial model.*

5 Conclusions and open problems

We have proposed several variations on the adversarial model to cope with packet priorities and link failures. We have studied universal stability from the point of view of both, the network and the queueing policy. We have also addressed the complexity of deciding stability under a fixed protocol.

We have shown that in the adversarial, failure, reliable, priority and variable priority models, the set of networks that are universally stable remains the same. The models present a different behavior with respect to the universal stability of protocols, since LIS is universally stable in the adversarial model, but it is not universally stable in the other models. In contrast, we have shown that there are no universally stable protocols for the variable priority model.

We have proposed a new and natural way to model the behavior of queueing systems in dynamic networks. Our results compared to the slowdown models introduced in [8] show that the power of an adversary in the failure and in the dynamic slowdown model is quite similar. In both cases the LIS protocol is not universally stable. However the static slowdown model is less powerful than the failure model as LIS remains universally stable [8].

The argument used in the proof of Theorem 4.1 in [8] can be used to show how to construct an adversary in the variable priority model that simulates an adversary in the dynamic slowdown model. It would be of interest to find constructions, similar to those given in Lemmas 3 and 5 to relate the power of the slowdown and failure models without changing the protocol.

Regarding the dynamic capacity model, the authors frequently use the trick of injecting $c - c_e(t)$ dummy packets which only need to traverse link e. This can be done without violating the load condition for a network with static capacity c provided that $c_e(t) > 0$ (see Theorems 3.3 and 3.4 [8]). It will be of interest to analyze the case with zero capacities.

It remains as an open problem to show the existence of a protocol that is universally stable in the failure model but it is not in the priority model.

All the already known characterizations of stability under a protocol are equivalent to universal stability in the adversarial model, even in the variable priority model. It is an interesting open question to know whether there is any protocol $\mathcal{P}$, not universally stable, for which there are networks that are not universally stable but that are stable under $\mathcal{P}$. Further open questions concerning this property are given in Figure 2.

References

[1] C. Àlvarez, M. Blesa, J. Díaz, A. Fernández and M. Serna. The complexity of deciding stability under FFS in the adversarial model. Technical Report, Software department, Universitat Politècnica de Catalunya, 2003.

[2] C. Àlvarez, M. Blesa, and M. Serna. A characterization of universal stability in the adversarial queueing model. Manuscript in preparation, 2003.

[3] C. Àlvarez, M. Blesa, and M. Serna. Universal stability of undirected graphs in the adversarial queueing model. In *14th ACM Symposium on Parallel Algorithms and Architectures* (SPAA'02), pages 183–197, Winnipeg, Canada, August 2002. ACM Press New York, USA.

[4] M. Andrews, B. Awerbuch, A. Fernández, J. Kleinberg, T. Leighton, and Z. Liu. Universal stability results for greedy contention–resolution protocols. *Journal of the ACM*, 48(1):39–69, 2001. Previously at FOCS'96, pages 380–389.

[5] E. Anshelevich, D. Kempe, and J. Kleinberg. Stability of load balancing algorithms in dynamic adversarial systems. In *34th. ACM Symposium on Theory of Computing* (STOC'02), pages 399–406, 2002.

[6] B. Awerbuch, P. Berenbrink, A. Brinkmann, and C. Scheideler. Simple routing strategies for adversarial systems. In *42th. IEEE Symposium on Foundations of Computer Science* (FOCS'01), pages 158–167, 2001.

[7] R. Bhattacharjee and A. Goel. Instability of FIFO at arbitrarily low rates in the adversarial queueing model. Technical Report 02-776, Department of Computer Science, University of Southern California, Los Angeles, USA, 2002.

[8] A. Borodin, R. Ostrovsky, Y. Rabani. Stability Preserving Transformations: Packet Routing Networks with Edge Capacities and Speeds. In *ACM-SIAM Symposium on Discrete Algorithms* (SODA'01), pages 601–610, 2001.

[9] A. Borodin, J. Kleinberg, P. Raghavan, M. Sudan, and D. Williamson. Adversarial queueing theory. *Journal of the ACM*, 48(1):13–38, 2001. Previously at STOC'96, pages 376–385.

[10] V. Cholvi, J. Echagüe, and A. Fernández. Universal stability results for low rate adversaries in packet switched networks. Manuscript, 2002.

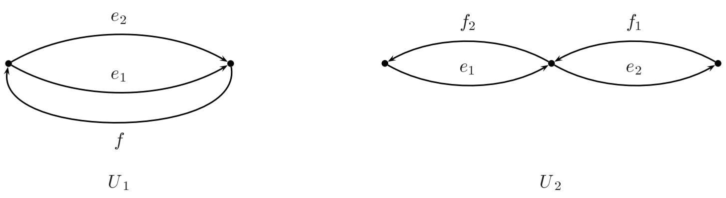
[11] J. Diaz, D. Koukopoulos, S. Nikoletseas, M. Serna, P. Spirakis, and D. Thilikós. Stability and non-Stability of the FIFO Protocol. In *13th annual ACM Symposium on Parallel Algorithms and Architectures* (SPAA'01), pages 48–52, Crete Island, Greece, July 2001.

[12] D. Koukopoulos, M. Mavronicolas, S. Nikoletseas, and P. Spirakis. On the stability of compositions of universally stable, greedy contention-resolution protocols. In D. Malkhi, editor, *Distributed Computing, 16th International Conference*, volume 2508 of *Lecture Notes in Computer Science*, pages 88–102, Toulouse, France, October 2002. Springer.

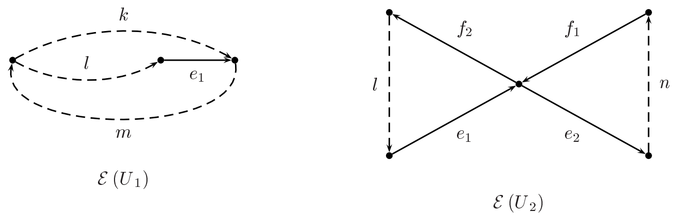
[13] D. Koukopoulos, M. Mavronicolas and P. Spirakis. FIFO is unstable at arbitrarily low rates. ECCC, TR-03-16, 2003.

[14] Z. Lotker, B. Patt-Shamir, and A. Rosén. New stability results for adversarial queuing. In *14th ACM Symposium on Parallel Algorithms and Architectures* (SPAA'02), pages 175–182, Winnipeg, Canada, August 2002. ACM Press New York, USA.

[15] A. Rosén. A note on models for non-probabilistic analysis of packet switching networks. *Information Processing Letters*, 84:237–240, 2002.



(a) Basic digraphs characterizing universal stability in the adversarial model (see [2]).



(b) Extensions of the basic digraphs in Figure 1(a), where edges have been extended to paths. Let us denote by p_k, p_l, p_m and p_n those paths, where $k, m \geq 1$ and $l, n \geq 0$.

Figure 1: Family of subgraphs characterizing universal stability in the adversarial model.

	Adversarial	Failure	Priority	Reliable
LIS	T [4]	P	P	P
NTG	P [3]	P	P	P
FFS	P [1]	P	P	P
LIFO	O	O	O	O
FIFO	O	P	P	P

Figure 2: Complexity of deciding stability under a protocol. Where T means trivial property, P means decidable in polynomial time, and O means open. The unrefered results are provided in this paper

A Appendix

A.1 Protocol universal stability under the variable priority model

Lemma 17 *For any greedy protocol $\mathcal{P}$, the pair $(U_1, \mathcal{P})$ is not stable in the variable priority model.*

Proof. We will construct an adversary that uses only two priorities, 0 and 1, with 1 being the highest. The adversary is described by rounds, during a round the priority of a packet will not change, but it might be reassigned at the beginning of the next round.

At the beginning there are s packets that want to traverse edge f, all of them get priority 1. The adversary $\mathcal{A}$ will play injections in five rounds:

Round 1: for s steps, the adversary injects rs packets of the form (fe_2) with priority 0. These injections get mixed with the initial packets at edge f, and are blocked because their priority is lower, independently of the greedy queuing policy.

Round 2: the adversary assigns priority 1 to the packets waiting at f's queue.
For the next rs steps, the adversary injects a set of r^2s packets of the form (fe_1) and r^2s packets of the form (e_2), all of them with priority 0. Injections (fe_1) are blocked by the packets (fe_2) from the first round. The r^2s injections of the form (e_2) get mixed with packets (fe_2), so that r^2s (e_2) packets are queued at e_2 at the end of the round.

Round 3: the adversary assigns priority 1 to the packets queued at f and 0 to the packets queued at e_2.
For the next r^2s steps, the adversary injects r^3s packets of the form (e_2) with priority 0 and r^3s of the form (e_1f) with priority 0. The simple injections on e_2 get mixed with the packets at e_2 from the previous round, at the end of the round they will be r^3s such packets queued at e_2. The (e_1f) injections get mixed with the packets of the form (fe_1) at edge e_1. This edge at the end of the round will have in total r^3s packets of the form (e_1f).

Round 4: the adversary assigns priority 0 to the packets queued at e_1 and priority 1 to the packets queued at e_2.
For the next r^3s steps, the adversary injects r^4s packets of the form (e_2f) with priority 0 and r^4s packets of the form (e_1) with priority 1. The injections at e_1 block the last r^4s packets queued at e_1 at the beginning of the round. The (e_2f) injections are blocked by the (e_2) packets from the previous round. So at the end of the round the number there will be r^4s packets (e_2f) and r^4s packets (e_1f) queued at e_2 and e_1 respectively.

Round 5: the adversary assigns priority 1 to the packets queued at e_1 and e_2.
For the next r^4s steps, the adversary injects r^5s packets of the form (f).

At the end there are $(r^5+r^4)s$ packets (f) in the system. The adversary $\mathcal{A}$ described above makes the network U_1 unstable when $r^5 + r^4 > 1$, i.e. $r \geq 0.857$. ■

A.2 Stability under a protocol

In the following, we show how the proofs for instability of U_1, U_2 (see Figure 1(a)) and their corresponding extensions $\mathcal{E}(U_1)$ and $\mathcal{E}(U_2)$ (see Figure 1(b)), under the FIFO and LIS protocols. These results lead to Lemmas 11 and 13.

The extended graphs are formed by transforming the edges of U_1 and U_2 to paths. Let us denote by p_k, p_l, p_m and p_n those paths, where $k, m \geq 1$ and $l, n \geq 0$.

The quantity of packets in the initial configuration is considered also to be big enough to guarantee that, the time given to every round is sufficient for the packets to cross the possible paths (p_k, p_l, p_m or p_n) in their packet trajectory and arrive where they are supposed to arrive. For simplicity, we will omit floors and ceilings, and sometimes will count steps and packets roughly; by carrying these through the computations one loses some additive constants, which are offset by the fact that s_0 was a large constant.

A.2.1 FIFO stability under the failure model

Lemma 18 *The pair (U_1, FIFO) is not stable in the failure model.*

Proof. At the beginning there are s packets that want to traverse edge f. The adversary $\mathcal{A}$ will play injections in five rounds:

Round 1: for s steps, the adversary injects rs packets of the form (fe_2). These injections get mixed with the initial packets at edge f, and are blocked there because the queueing protocol is FIFO.

Round 2: for the next rs steps, the adversary injects a set of r^2s packets of the form (fe_1) and r^2s packets of the form (e_2). Injections (fe_1) are blocked by the packets (fe_2) from the first round. The r^2s injections of the form (e_2) get mixed with packets (fe_2), so that r^2s (e_2) packets are queued at e_2.

Round 3: for the next r^2s steps, the adversary injects r^3s packets of the form (e_2) and r^3s of the form (e_1f). The simple injections on e_2 will be blocked by the packets at e_2 from the previous round. The (e_1f) injections get blocked by the packets of the form (fe_1) at edge e_1. This edge at the end of the round will have in total r^3s packets of which $r^3s/(1+r)$ are (e_1) and $r^4s/(1+r)$ are (e_1f).

Round 4: for the next r^3s steps, the adversary injects r^4s packets of the form (e_2f) and makes edge e_1 fail for r^4s steps. The failures at e_1 block the last r^4s packets queued at e_1 at the beginning of the round. As the proportion is maintained, at the end of the round there are $r^4s/(1+r)$ packets (e_1) and $r^5s/(1+r)$ packets (e_1f). The (e_2f) injections are blocked by the (e_2) packets from the previous round.

Round 5: for the next r^4s steps, the adversary injects r^5s packets of the form (f).

At the end there are $(r^5 + \frac{r^5}{1+r})s$ packets (f) in the system. The adversary $\mathcal{A}$ described above makes the network U_1 unstable when $r^5\frac{2+r}{1+r} > 1$, i.e. $r \geq 0.920$. ∎

Lemma 19 *The pair (U_2, FIFO) is not stable in the adversarial model.*

Proof. We will not use failures. At the beginning there are s packets that want to traverse edge f_2. The adversary $\mathcal{A}$ will play injections in five rounds:

Round 1: for s steps, the adversary injects rs packets of the form $(f_2e_1e_2)$. These injections get mixed with the initial packets at edge f_2, and are blocked there because the queueing protocol is FIFO.

Round 2: for the next rs steps, the adversary injects a set of r^2s packets of the form (f_2e_1) and r^2s packets of the form (e_2). All the injections (f_2e_1) are blocked by the older packets and stay at the end of the round. The injections (e_2) get mixed with the original packets, so at the end of the round there are r^2s packets that want to cross (e_2).

Round 3: for the next r^2s steps, the adversary injects r^3s packets of the form (e_1) and r^3s of the form $(e_2f_1f_2)$. The simple injections on e_1 get mixed with the old packets arriving at e_1, at the end they will remain r^3s packets that want to traverse e_1. The second set is delayed by the old packets.

Round 4: for the next r^3s steps, the adversary injects r^4s packets of the form (e_1f_2) and r^4s packets of the form (f_1). At the end of the round there are r^4s packets that want to traverse e_1f_2 and $\frac{r^4s}{r+1}$ packets that want to traverse f_1f_2 and $\frac{r^5s}{r+1}$ packets that want to traverse only f_1.

Round 5: for the next r^4s steps, the adversary injects r^5s packets of the form (f_2).

At the end there are $(r^5 + \frac{r^4}{1+r})s$ packets (f_2) in the system. This implies instability for $r \geq 0.914$.
∎

Lemma 20 *Any graph in $\mathcal{E}(U_1)$ is not stable under* FIFO *in the failure model.*

Proof. Observe that, in the FIFO protocol, neither the length of the path nor the distances to source or destination are not important criteria. Only the order on joining the queues matters. Due to this, proofs for instability of U_1 and U_2 can be easily adapted just by using an adversary that replaces edges by paths. Notice however that, the longer the paths the less packets the network accumulates but, for big enough initial configurations, instability can always hold. The quantity of packets in the initial configuration is considered also to be big enough to guarantee that, the time given to every round is sufficient for the packets to cross the possible paths (p_k, p_l, p_m or p_n) in their packet trajectory and arrive where they are supposed to arrive.

Let G_1 be the graph $\mathcal{E}(U_1)$ described in Figure 1(b). At the beginning there are s packets that want to traverse edge p_m. The adversary $\mathcal{A}$ will play injections in five rounds:

Round 1: for s steps, the adversary injects rs packets of the form $(p_m p_k)$. These injections get mixed with the initial packets at the first edge of p_m, and are blocked there because the queueing protocol is FIFO.

Round 2: for the next rs steps, the adversary injects a set of $r^2 s$ packets of the form $(p_m p_l e_1)$ and $r^2 s$ packets of the form (p_k). Injections $(p_m p_l e_1)$ are blocked by the packets $(p_m p_k)$ from the first round. An amount of rm injections of the form (p_k) are not accumulated. The rest get mixed with packets $(p_m p_k)$, so that $r^2 s - rm$ packets of the form (p_k) are queued at the first edge of p_k.

Round 3: for the next $r^2 s$ steps, the adversary injects $r^3 s$ packets of the form (p_k) and $r^3 s$ of the form $(p_l e_1 p_m)$. Then, $r^3 s - rm$ injections on p_k will be blocked by the packets at the first edge of p_k remaining from the previous round. All the $(p_l e_1 p_m)$ injections get mixed with the packets of the form $(p_m p_l e_1)$ at the first edge of p_l, except rm of them. At the end of the round this edge will have in total $r^3 s - rm$ packets of which $r^3 s - rm/(1+r)$ are $(p_l e_1)$ and $r^4 s - r^2 m/(1+r)$ are $(p_l e_1 p_m)$.

Round 4: for the next $r^3 s - rm$ steps, the adversary injects $r^4 s - r^2 m$ packets of the form $(p_k p_m)$ and makes edge e_1 fail for $r^4 s - r^2 m$ steps. The failures at e_1 block the last $r^4 s - r^2 m - rl$ packets queued at the first edge of p_l at the beginning of the round. As the proportion is maintained, at the end of the round there are $r^4 s - r^2 m - rl/(1+r)$ packets $(p_l e_1)$ and $r^5 s - r^3 m - r^2 l/(1+r)$ packets $(p_l e_1 p_m)$. The $(p_k p_m)$ injections are blocked by the (p_k) packets from the previous round.

Round 5: for the next $r^4 s - r^2 m$ steps, the adversary injects $r^5 s - r^3 m$ packets of the form (p_m).

At the end there are $r^5 s - r^3 m + \frac{r^5 s - r^3 m - r^2 l}{1+r}$ packets (p_m) in the system. The adversary $\mathcal{A}$ described above makes the network $\mathcal{E}(U_1)$ unstable when $r^5 s - r^3 m + \frac{r^5 s - r^3 m - r^2 l}{1+r} > s$. Note that $\frac{r^4 m + r^2 l}{1+r} < \frac{m+l}{1+r} = C$, then for big enough s, an injection rate r can be found such that $(r^5 + \frac{r^5}{1+r})s - C > s$ holds and thus $G_1 \in \mathcal{E}(U_1) \cup \mathcal{E}(U_2)$ is not stable under FIFO in the failure model. ■

Lemma 21 *Any graph in $\mathcal{E}(U_2)$ is not stable under* FIFO *in the failure model.*

Proof. Let now G_2 be the graph $\mathcal{E}(U_2)$ described in Figure 1(b). At the beginning there are s packets that want to traverse edge f_2. The adversary $\mathcal{A}$ will play injections in five rounds:

Round 1: for s steps, the adversary injects rs packets of the form $(f_2 p_l e_1 e_2)$. These injections get mixed with the initial packets at edge f_2, and are blocked there because the queueing protocol is FIFO.

Round 2: for the next rs steps, the adversary injects a set of $r^2 s$ packets of the form $(f_2 p_l e_1)$ and $r^2 s$ packets of the form (e_2). All the injections $(f_2 p_l e_1)$ are blocked by the older packets and stay at the end of the round. The injections (e_2) get mixed with the original packets, so at the end of the round there are $r^2 s - rl$ packets that want to cross e_2.

Round 3: for the next $r^2 s$ steps, the adversary injects $r^3 s$ packets of the form $(p_l e_1)$ and $r^3 s$ of the form $(e_2 p_n f_1 f_2)$. The former get mixed with the old packets arriving at the first edge of p_l, at the end they will remain r^3s packets. From the second set, $r^3s - rl$ packets are delayed by the old packets.

Round 4: for the next r^3s steps, the adversary injects r^4s packets of the form $(p_l e_1 f_2)$ and provokes r^4s failures in e_2. At the end of the round there are r^4s packets that want to traverse $p_l e_1 f_2$ and $r^4s - r^2l$ packets from the previous round blocked in e_2 due to the failures.

Round 5: for the next r^4s steps, the adversary injects r^5s packets of the form (f_2). Because of the possible different length of the paths p_l and p_n, there are $r \mid l - n \mid$ injections that will not allow to accumulate.

At the end there are $(r^4 + r^5)s - r^2l - r \mid l - n \mid$ packets in the system waiting to traverse f_2. The adversary $\mathcal{A}$ makes the network $\mathcal{E}(U_2)$ unstable when $(r^4 + r^5)s - r^2l - r \mid l - n \mid > s$. Notice that $r^2l + r \mid l - n \mid < 2l + n = C$, then for big enough s, an injection rate r can be found such that $(r^4 + r^5)s - C > s$ holds and thus $G_2 \in \mathcal{E}(U_1) \cup \mathcal{E}(U_2)$ is not stable under FIFO in the failure model. ∎

A.2.2 LIS stability under the failure model

Lemma 22 *The pair* (U_1, LIS) *is not stable in the failure model.*

Proof. At the beginning there are s packets that want to traverse edge f. An adversary $\mathcal{A}$ playing injections exactly as does the adversary in the analogous Lemma 18 under FIFO would accumulate $r^4s + r^5s$ packets in the queue of edge f at the end of the 5th round. This would make the system unstable for $r \geq 0.857$. ∎

Lemma 23 *The pair* (U_2, LIS) *is not stable in the failure model.*

Proof. Starting with s packets that want to traverse edge f_2, let $\mathcal{A}$ be an adversary playing injections exactly as the adversary in the analogous Lemma 19 under FIFO. But the adversary we used in Lemma 19 was not producing failures, then substitute there the single injections of the form (f_1) at round 4 by failures of edge f_1. Thus, at the end of the 5th round, this would allow to accumulate $r^4s + r^5s$ packets in the queue of edge f_2 and prove that the system can become unstable for $r \geq 0.857$. ∎

Lemma 24 *Any graph in* $\mathcal{E}(U_1)$ *is not stable under* LIS *in the failure model.*

Proof. Let G_1 be the graph $\mathcal{E}(U_1)$ described in Figure 1(b). At the beginning there are s packets that want to traverse edge p_m. The adversary $\mathcal{A}$ will play injections in five rounds exactly as the adversary for FIFO (see Lemma 18) but modifying the packet trajectories by extending them to paths: where e_1 appears put $p_l e_1$, and replace e_2 and f in paths by p_k and p_m respectively.

Also, instead of making edge e_1 fail at the fourth round, this adversary makes the first edge of p_l fail. At round fifth, because of the possible different length of the paths p_l and p_k, there are $r|l-k|$ injections that will not allow to accumulate. The quantity of packets in the initial configuration is considered to be big enough to guarantee that packets arrive to the first edge of p_m. This adversary would allow to make the system under LIS unstable when $r^4s - r^2m + r^5s - r^3m - r \mid k - l \mid > s$. Notice that $r^2m + r^3m + r \mid k - l \mid < 2m + l + k = C$ and, for big enough s, an injection rate r can be found such that $(r^4 + r^5)s - C > s$ holds, thus showing instability of G_1 under FIFO in the failure model. ∎

Lemma 25 *Any graph in* $\mathcal{E}(U_2)$ *is not stable under* LIS *in the failure model.*

Proof. Let now G_2 be the graph $\mathcal{E}(U_2)$ described in Figure 1(b). At the beginning there are s packets that want to traverse edge f_2. The adversary $\mathcal{A}$ will play injections in five rounds. The first three rounds it injects the same kind of packets than the adversary for FIFO (see Lemma 19) but modifying the packet trajectories by extending them to paths: where e_1 appears put $p_l e_1$, and replace e_2 in paths by $e_2 p_n$. We show the last two rounds:

Round 4: at the beginning there are $r^3 s$ packets of the form $(p_l e_1)$ and $r^3 s - rl$ packets of the form $(e_2 p_n f_1 f_2)$. For the next $r^3 s$ steps, the adversary injects $r^4 s$ packets of the form $(e_1 f_2)$ and causes $r^4 s$ failures in edge f_1. From the injections in e_1, only $r^4 s - rl$ will remain at the end of the round. The failures will accumulate $r^4 s - rn - r^2 l$ packets that want to traverse $f_1 f_2$.

Round 5: During $r^4 s - rl$ steps, the adversary injects $r^5 s - r^2 l$ packets (f_2).

When the fifth round finishes, there are $(r^4 + r^5)s - 2r^2 l - rn$ packets at f_2, thus instability would hold under LIS unstable when $(r^4 + r^5)s - 2r^2 l - rn > s$. Notice that $2r^2 l + rn < 2l + n = C$ and, for big enough s, an injection rate r can be found such that $(r^4 + r^5)s - C > s$ holds, thus showing instability of G_2 under LIS in the failure model. ■