Theory Comput Syst
DOI 10.1007/s00224-007-9046-1

Adversarial Queueing Model for Continuous Network Dynamics

Maria Blesa · Daniel Calzada · Antonio Fernández · Luis López · Andrés L. Martínez · Agustín Santos · Maria Serna · Christopher Thraves

Abstract In this paper we initiate the generalization of the Adversarial Queueing Theory (AQT) model to capture the dynamics of continuous scenarios in which the

This work was partially supported by EU IST-2004-15964 (AEOLUS), by the Spanish Ministry of Education and Science under CICYT TIN2005-09198 (ASCE) and CICYT TIN2005-25859-E, by the Regional Government of Madrid under contract No. 07T/0022/2003, and by the Universidad Rey Juan Carlos under project No. PPR-2004-42. Also partially supported by the Universidad de Chile (Mecesup fellowship and Anillo en Redes).

A preliminary version of this work was presented in the *30th International Symposium on Mathematical Foundations of Computer Science* (MFCS), Gdansk, Poland, LNCS, vol. 3618, pp. 144–155, Springer.

M. Blesa (✉) · M. Serna
ALBCOM, LSI, Universitat Politècnica de Catalunya, 08034 Barcelona, Spain
e-mail: mjblesa@lsi.upc.edu

M. Serna
e-mail: mjserna@lsi.upc.edu

D. Calzada
ATC, EUI, Universidad Politécnica de Madrid, 28031 Madrid, Spain
e-mail: dcalzada@eui.upm.es

A. Fernández · L. López · A.L. Martínez · A. Santos
LADyR, GSyC, ESCET, Universidad Rey Juan Carlos, 28933 Madrid, Spain

A. Fernández
e-mail: anto@gsyc.escet.urjc.es

L. López
e-mail: llopez@gsyc.escet.urjc.es

A.L. Martínez
e-mail: aleonar@gsyc.escet.urjc.es

A. Santos
e-mail: asantos@gsyc.escet.urjc.es

C. Thraves
DIM, Universidad de Chile, 837-0459 Santiago, Chile
e-mail: thraves@dim.uchile.cl

usually assumed synchronicity of the evolution is not required anymore. We propose an asynchronous model, named *continuous* AQT (CAQT), in which packets can have arbitrary lengths, and the network links may have different speeds (or bandwidths) and propagation delays. With respect to the standard AQT model, these new features turn out to be significant for the stability of packet scheduling policies that take them into account, but not so much for the stability of networks.

From the network point of view, we show that networks with directed acyclic topologies are universally stable, i.e., stable independently of the scheduling policies and traffic patterns used in it. Interestingly enough, this even holds for traffic patterns that make links to be fully loaded. Finally, it turns out that the set of universally stable networks remains the same as in the AQT model and, therefore, the property of universal stability of networks is decidable in polynomial time.

Concerning packet scheduling policies, we show that the well-known LIS, SIS, FTGand NFSscheduling policies remain universally stable in the CAQT model. We introduce other scheduling policies that, although being universally stable in the AQT model, they are unstable under the CAQT model.

Keywords Adversarial queueing theory · Packet-switched networks · Stability of networks · Stability of queueing policies

1 Introduction

The Adversarial Queueing Theory (AQT) model [4, 8] has been used in the latest years to study the stability and performance of packet-switched networks. The AQT model, (like other adversarial models) allows to analyze the system in a worst-case scenario, since it replaces traditional stochastic arrival assumptions in the traffic pattern by worst-case inputs. In this model, the arrival of packets to the network (i.e., the traffic pattern) is controlled by an adversary that defines, for each packet, the place and time in which the packet joins the system and, additionally it might decide the path it has to follow. In order to study non-trivial overloaded situations, the adversary is restricted so that it cannot overload any link (in an amortized sense). Under these assumptions, the *stability* of network systems $(\mathcal{G}, \mathcal{P}, \mathcal{A})$ is studied. A network system $(\mathcal{G}, \mathcal{P}, \mathcal{A})$ is represented by the network topology $\mathcal{G}$, the policy (or protocol) $\mathcal{P}$ used for scheduling the packets at every link, and the adversary $\mathcal{A}$, which defines the traffic pattern. Stability is the property that at any time the maximum number of packets present in the system is bounded by a constant that may depend on system parameters.

The original AQT model assumes a synchronous behavior of the network, that evolves in steps. In each step at most one packet crosses each link. Implicitly, this assumption means that all the packets have the same size and all the links induce the same delay in each packet transmission. To the best of our knowledge, this is the first work that removes the restriction of synchronism.

In this paper we propose a generalization of the AQT model allowing arbitrary packet lengths, link speeds (bandwidths), and link propagation delays. The network traffic flow is considered to be continuous in time. Since we do not restrict a synchronous system evolution anymore, we call this model *continuous* AQT (CAQT). Note that all the results for the AQT model which are concerned with instability, also hold for our CAQT model, e.g., the instability of the FIFO policy at any constant rate [7]. The CAQT model is inspired in the traffic conditions of the session oriented model proposed by Cruz [11], which is widely studied in the communication networks literature. The synchronous assumptions of the AQT model limit the capacity of the adversary as well. In the CAQT model the adversary is more powerful, and any instability result shown in the AQT model can be reproduced in ours.

We show that several results from the AQT model still hold in the CAQT model. First, we show that in the case of non-overloading traffic injection rates, having bounded queue size implies having bounded packet end-to-end delays and vice versa. Then, we show that networks with a directed acyclic graph (DAG) topology are always stable even if the links are fully loaded. Concerning packet scheduling policies, we show that the well-known LIS, SIS, FTG and NFS scheduling policies remain universally stable in our model. Finally, we show that some scheduling policies which are based on criteria concerning the length of the packets, the bandwidth of the links or their propagation delay, can configure unstable systems.

There have been previous generalizations of the AQT model to other synchronous models for networks with links with different (and possibly variable) capacities, or links with delays [9, 13, 14]. These works consider packets of unitary length, while the model we present allows packets of arbitrary lengths. Moreover, they still assume a synchronous network evolution, to the point that, for instance in [9] all capacities and slow-downs must have an integral positive value. Another significant difference between those existing models and the model presented here, is the possibility of the latter of considering systems with both link capacities (a.k.a. bandwidths) and delays at the same time. Probably, among the existing cited models, the model presented in [9] is the closest one. However, there are important differences among that model and the model we present here. First, as mentioned above, the model in [9] does always consider capacities and delays as separate features. Second, most of the results obtained in [9] allow to change dynamically those features; the situation of static modifications (like the ones we tackle here) is not clear in [9]. In static scenarios, only results concerning scheduling policies are provided, namely, that "well defined" universal stable scheduling policies maintain their universal stability under static capacities, and that common universally stable policies maintain their universal stability under static speeds. Other generalizations of the AQT model are those considering dynamic networks, like networks with failures [2, 3, 5, 6]. These models that consider failures can be seen as models in which the capacities of the links can change dynamically over time, between a fixed positive value (when there is no failure), and zero (when a failure occurs). However, since we consider static capacities, failures are not considered in this work and remains open the general model with variable (and possibly null) capacities.

Concerning the length of the packets, the work included in [12] is (to the best of our knowledge) the only generalization of the AQT model considering packets of arbitrary lengths (up to a maximum) or links of arbitrary (not integral) speeds and propagation delays. In that model the adversary is more powerful than in the AQT model, and a sufficient condition on the adversary injection rate for assuring network stability is presented.

2 System Model

Like AQT, the CAQT model represents a network as a finite directed graph $\mathcal{G}$ in which the set of nodes $V(\mathcal{G})$ represent the hosts, and the set of edges $E(\mathcal{G})$ represent the links between those hosts. Each link $e \in E(\mathcal{G})$ in this graph has associated a positive but not infinite transmission speed (a bandwidth), denoted as B_e, which does not change over time. The bandwidth of a link establishes how many bits can be transmitted in the link per time unit. Instead of considering the bandwidth as a synonym for parallel transmission, we relate the bandwidth to the transmission velocity. We consider that only one bit can be put in a link $e \in E(\mathcal{G})$ at each time, and that conceptually the sender puts the associated signal level to the corresponding bit for $1/B_e$ seconds for each bit. This means that a bit can be partially transmitted or partially received at a given time. Let us denote as $B_{\min} = \min_{e \in E(\mathcal{G})} B_e$ and as $B_{\max} = \max_{e \in E(\mathcal{G})} B_e$ the minimum and maximum bandwidth, respectively, of the edges in $\mathcal{G}$.

Each link $e \in E(\mathcal{G})$ has also associated a propagation delay, denoted here as P_e, being $P_e \geq 0$. The propagation delay of an edge does not change over time. This delay, measured in seconds, establishes how long it takes for a signal (the start of a bit, for instance) to traverse the link. This parameter has to do with the propagation speed of the changes in the signal that carry the bits along the physical medium used for the transmission. We will denote as $P_{\min} = \min_{e \in E(\mathcal{G})} P_e$ and $P_{\max} = \max_{e \in E(\mathcal{G})} P_e$ the minimum and maximum propagation delay, respectively, of the edges in $\mathcal{G}$.

Like in the AQT model, we assume the existence of an adversary that defines the traffic pattern of the system by choosing when and where to inject packets into the system, and the path to be followed by each of them. We assume that a packet path is edge-simple, in the sense that it does not contain the same edge more than once (it can visit the same vertex several times, though). Again, we restrict the adversary so that it cannot trivially overload any link. To do so, we also define two system-wide parameters: the *injection rate* r (with $0 < r \leq 1$), and the *burstiness* b (with $b \geq 1$). For every link $e \in E(\mathcal{G})$, if we denote by $N_e(I)$ the total size (in bits) of the packets injected by the adversary in the interval I whose path contains link e, it must be satisfied that

$$N_e(I) \leq r|I|B_e + b.$$

We call an adversary $\mathcal{A}$ that satisfies this restriction an (r, b)-adversary. The injection rate r is sometimes expressed alternatively as $(1 - \varepsilon)$, with $\varepsilon \geq 0$.

Regarding packet injections, we assume that the adversary injects packets instantaneously. From the above restriction, this implies that packets have a maximum size of b bits. In general, we will use L_p to denote the length (in bits) of a packet p, and $L_{\max} = \max_p L_p \leq b$ to denote the maximum packet length. Observe that, once a packet p starts being transmitted through a link $e \in E(\mathcal{G})$, it will only take $P_e + L_p/B_e$ units of time more until it crosses it completely.

Let us now look at the packet switching process. We assume that each link has associated an *output queue*, where the packets that have to be sent across the link are held. The still unsent portion of a packet that is being transmitted is also held in this queue. In fact, in order to simplify the analysis, if a bit has only been partially

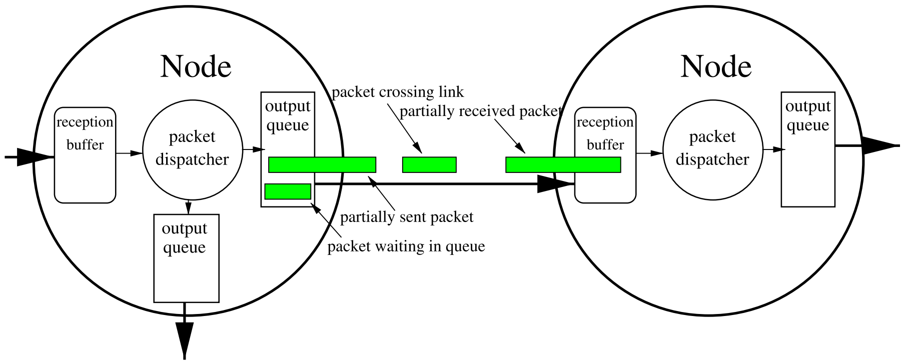

Fig. 1 Elements involved in the nodes and links of the network in the CAQT model

sent, we assume that the still unsent portion of the bit still resides in this queue.[1] A packet can arrive to a node either by direct injection of the adversary or by traversing some incoming link. In the latter case we assume that only full packets are dispatched (moved to an output queue). Hence, we assume that each link has a *reception buffer* in the receiving node where the portion of a partially received packet is held. As soon as the very last bit of a packet is completely received, the packet is dispatched instantaneously (by a packet dispatcher) to the corresponding output queue (or removed, if this is the final node of the packet). Figure 1 shows these network elements.

The definition of *stability* in the CAQT model is analogous to the definitions stated under other adversarial models.

Definition 1 Let G be a network with a bandwidth B_e and a propagation delay P_e associated to each link e, $\mathcal{P}$ be a scheduling policy (or protocol), and $\mathcal{A}$ an (r,b)-adversary, with $0 < r \leq 1$ and $b \geq 1$. The system $(\mathcal{G},\mathcal{P},\mathcal{A})$ is stable if, at every moment, the total number of packets (or, equivalently, the total number of bits) in the system is bounded by a value C, that can depend on the system parameters.

We also use common definitions of *universal stability*. We say that a scheduling policy $\mathcal{P}$ is universally stable if the system $(\mathcal{G},\mathcal{P},\mathcal{A})$ is stable for each network $\mathcal{G}$ and each (r,b)-adversary $\mathcal{A}$, with $0 < r < 1$ and $b \geq 1$. Similarly, we say that a network $\mathcal{G}$ is universally stable if the system $(\mathcal{G},\mathcal{P},\mathcal{A})$ is stable for each *greedy* scheduling policy[2] $\mathcal{P}$ and each (r,b)-adversary $\mathcal{A}$, with $0 < r < 1$ and $b \geq 1$.

[1]Clearly, a queue can only store "whole" bits. However, since the transmission of a bit over a link of finite bandwidth is not done instantaneously, between the start and the end of the transmission it can be considered that the bit has been partially sent. Then the assumption that the unsent portion is still held in the output queue of the link allows to account for the whole bit at all times.

[2]Greedy (or work-conserving) protocols are those forwarding a packet across a link e whenever there is at least one packet waiting to traverse e. Three types of packets may wait to traverse a link in a particular instant of time: the incoming packets arriving from adjacent links, the packets injected directly into the link, and the packets that could not be forwarded in previous steps. At each time step, only one packet from those waiting is forwarded through the link; the rest are kept in the output queue.

Some additional notation is needed to describe the state of the queues and the packets at a specific time step. We will use $Q_t(e)$ to denote the queue size (in bits) of edge $e \in E(\mathcal{G})$ at time t, and define $Q_{\max}(e) = \max_t Q_t(e)$. Similarly, we will use $R_t(e)$ to denote the number of bits at time t that are crossing link e, or already crossed it but are still in its reception buffer at the target node of e. Then, we define $R_{\max}(e) = \max_t R_t(e)$. Observe that $R_{\max}(e) < P_e B_e + L_{\max}$ and is hence bounded. $A_t(e)$ will denote the number of bits in the system that require to cross e and still have to be transmitted across link e at time t. The bits in $Q_t(e)$ are included in $A_t(e)$, but those in $R_t(e)$ are not.

3 General Results

We point out some general results that apply to every system $(\mathcal{G}, \mathcal{P}, \mathcal{A})$ in the CAQT model, independently of which is the network topology, the scheduling policy used and the traffic pattern.

3.1 Relation between Maximum Queue Size and Maximum Delay

We show that for injection rate $r < 1$, having bounded queues is equivalent to having bounded end-to-end packet delay. This generalizes a result from the AQT model to the stronger CAQT model.

Theorem 1 *Let $\mathcal{G}$ be a network, $\mathcal{P}$ a greedy scheduling policy, and $\mathcal{A}$ an (r,b)-adversary with $r \le 1$ and $b \ge 1$. If the maximum end-to-end delay is bounded by D in the system $(\mathcal{G}, \mathcal{P}, \mathcal{A})$, then the maximum queue size of an edge e is bounded by $(D - P_e)B_e$.*

Proof We prove the claim by contradiction. Suppose there is some time t at which e has $Q_t(e) > (D - P_e)B_e$ bits in its queue. Then, the last packet p to completely cross e (out of those with bits in the queue at time t) will do so at a time

$$t' \ge t + \frac{Q_t(e)}{B_e} + P_e > t + \frac{(D - P_e)B_e}{B_e} + P_e = t + D.$$

Therefore, the end-to-end delay of p cannot be bounded by D. □

Theorem 2 *Let $\mathcal{G}$ be a network with $m = |E(\mathcal{G})|$ links, $\mathcal{P}$ a greedy scheduling policy, and $\mathcal{A}$ an (r,b)-adversary, with $r = 1 - \varepsilon < 1$ and $b \ge 1$. If the maximum queue size is bounded by Q in the system $(\mathcal{G}, \mathcal{P}, \mathcal{A})$, then the end-to-end delay of a packet p with path $e_1, \ldots, e_d$ is bounded by*

$$\sum_{i=1}^{d} \frac{mQ + \sum_{e \in E(\mathcal{G})} R_{\max}(e) + b}{\varepsilon B_{e_i}} + P_{e_i}.$$

Proof We bound the time p takes to cross every edge e_i. Let us assume that p arrives at the queue of e_i at time t. Note that all the bits in the system are either in the output queues, crossing links, or in the reception buffers. Then, there are at most $mQ + \sum_e R_{\max}(e)$ bits in the whole system at time t. Hence, the queue of e_i will be empty after at most an interval of time of length Δ, such that

$$\Delta B_e = mQ + \sum_e R_{\max}(e) + r\Delta B_e + b.$$

Hence, p will completely cross e by time

$$t' \le t + \frac{mQ + \sum_e R_{\max}(e) + b}{(1-r)B_e} + P_e. \qquad \square$$

Then, the following corollary follows from the above two lemmas.

Corollary 3 *Let $\mathcal{G}$ be a network, $\mathcal{P}$ a greedy scheduling policy, and $\mathcal{A}$ an (r,b)-adversary, with $r<1$ and $b \ge 1$. In the system $(\mathcal{G},\mathcal{P},\mathcal{A})$ the maximum end-to-end delay experienced by any packet is bounded if and only if the maximum queue size is bounded.*

3.2 Initial Configurations

The moment in which a system $(\mathcal{G},\mathcal{P},\mathcal{A})$ starts its dynamics is usually denoted as t_0. Moreover, the system can start either with no packet placed at any element of the network or with some kind of initial configuration. Usually, an initial configuration C_0 consists of a set S of packets located in the output queues of the network links. The following theorem formalizes the fact that starting with or without initial configuration is not a very relevant matter for the stability of the system.

Theorem 4 *A system $(\mathcal{G},\mathcal{P},\mathcal{A})$, where $\mathcal{G}$ is a network, $\mathcal{P}$ a greedy scheduling policy, and $\mathcal{A}$ an (r,b)-adversary with $r \le 1$ and $b \ge 1$, that starts with an initial configuration C_0 (consisting of a set S of packets in the network output queues) can be simulated by a system $(\mathcal{G},\mathcal{P},\mathcal{A}')$ starting from an empty configuration, where $\mathcal{A}'$ is an $(r, A_S + b)$-adversary and $A_S = \max_e A_0(e)$ is the maximum number of bits that have to be transmitted across any given edge in the paths of the set S of packets.*

Proof Trivially, any initial configuration C_0 for a system $(\mathcal{G},\mathcal{P},\mathcal{A})$ can be built from an empty initial configuration at time t_0 if we allow a burstiness which is large enough to inject all the packets in the initial configuration at the first step. Thus, any system $(\mathcal{G},\mathcal{P},\mathcal{A})$ that starts with a non-empty initial configuration as described can be simulated by another system $(\mathcal{G},\mathcal{P},\mathcal{A}')$ that starts with an empty one. $\square$

Corollary 5 *A policy or network that is universally stable for systems with empty initial configurations is also universally stable for initial configurations in which there are initially packets in the network output queues.*

4 Stability of Networks

We focus now on the study of stability of networks. In Sect. 4.1, we show that networks with a directed acyclic graph topology are universally stable, even when the traffic pattern can fully load the links, i.e., even for the injection rate $r = 1$. Moreover, as it turned out to be in the AQT model, the systems with topologies in the shape of a directed ring are also universally stable in the CAQT model for injection rates smaller than the unity. Sect. 4.2 covers this latter result. Both results, together with some result concerning the stability maintenance of acyclically connected digraphs, leads us towards the characterization of the property of universal stability of networks, which we fully study in Sect. 4.3.

4.1 Directed Acyclic Graphs

We study first the stability of networks whose topology define a directed acyclic graphs. Observe that as it happens for the AQT model, the absence of cycles together with the restrictions to which the adversary is subjected, makes the emergence of bottlenecks and the accumulation of packets not possible. Although inspired on the technique used in [8] under the AQT model for proving an analogous result, the technique used here is not a direct adaptation of it.

Theorem 6 *Let $\mathcal{G}$ be a directed acyclic graph, $\mathcal{P}$ any greedy scheduling policy, and $\mathcal{A}$ any (r,b)-adversary with $r \le 1$ and $b \ge 1$. The system $(\mathcal{G},\mathcal{P},\mathcal{A})$ is stable.*

Proof Let us first denote with T_e the node at the tail of link e (i.e., the node that contains the output queue of e), for every edge $e \in E(\mathcal{G})$. Let us also denote with $in(v)$ the set of incoming links to node v, for all $v \in V(\mathcal{G})$. Let us define the function Ψ on the edges of $\mathcal{G}$ as

$$\Psi(e) = Q_0(e) + b + R_{\max}(e) + \sum_{e' \in in(T_e)} \Psi(e').$$

If we call nodes without incoming links *sources*, we will show that $A_t(e) + R_t(e)$ is bounded by $\Psi(e)$, for all e and all $t \ge 0$, by induction on the maximum distance of T_e to a source (i.e., the length of the longest directed path from any source to T_e). Then, stability follows.

The base case of the induction is when T_e is a source. In this case, $A_t(e) = Q_t(e)$ and $\Psi(e) = Q_0(e) + b + R_{\max}(e)$. Let us fix a time t and consider two cases, depending on whether in the interval $[0,t]$ the output queue of e was empty at any time. If it was never empty, then by the restriction on the adversary and the fact that $\mathcal{P}$ is greedy we have that

$$Q_t(e) \le Q_0(e) + rtB_e + b - tB_e \le Q_0(e) + b.$$

Otherwise, if time t' was the last time in interval $[0,t]$ that the queue of e was empty (i.e., $Q_{t'}(e) = 0$), by the same facts,

$$Q_t(e) \le Q_{t'}(e) + r(t-t')B_e + b - (t-t')B_e \le b.$$

Clearly, in either case,

$$A_t(e) + R_t(e) \leq Q_t(e) + R_{\max}(e) \leq Q_0(e) + b + R_{\max}(e) = \Psi(e).$$

Now, let us assume that the maximum distance of T_e to any source is $k > 0$. Note that for any edge $e' \in in(T_e)$, the maximum distance of $T_{e'}$ to a source is at most $k-1$. Then, by induction hypothesis, we assume that $(A_t(e') + R_t(e')) \leq \Psi(e')$ for all $t \geq 0$ and all $e' \in in(T_e)$. Note that $A_t(e) \leq Q_t(e) + \sum_{e' \in in(T_e)} (A_t(e') + R_t(e'))$. Again, we fix t and consider separately the case when the output queue of e was never empty in the interval $[0, t]$ and the case when it was. In the first case we have that

$$\begin{aligned} A_t(e) &\leq Q_0(e) + rtB_e + b - tB_e + \sum_{e' \in in(T_e)} (A_0(e') + R_0(e')) \\ &\leq Q_0(e) + b + \sum_{e' \in in(T_e)} \Psi(e'). \end{aligned}$$

In the second case, if time t' was the last time in interval $[0, t]$ that the queue of e was empty (i.e., $Q_{t'}(e) = 0$), we have that

$$\begin{aligned} A_t(e) &\leq Q_{t'}(e) + r(t-t')B_e + b - (t-t')B_e + \sum_{e' \in in(T_e)} (A_{t'}(e') + R_{t'}(e')) \\ &\leq b + \sum_{e' \in in(T_e)} \Psi(e'). \end{aligned}$$

In either case, we have that

$$A_t(e) + R_t(e) \leq Q_0(e) + b + R_{\max}(e) + \sum_{e' \in in(T_e)} \Psi(e') = \Psi(e).$$

□

4.2 Rings

We study now the simplest networks which include some cyclicity: the directed rings. Let $\mathcal{G}$ denote the n-node ring with $V(\mathcal{G}) = \{0, 1, \ldots, n-1\}$, $E(\mathcal{G}) = \{(i, ((i+1) \bmod n) : i \in V(\mathcal{G})\}$ and $\mathcal{A}$ any (r, b)-adversary with $r = 1 - \varepsilon < 1$ and $b \geq 1$. We show that the rings are universally stable in the CAQT model, i.e., that any system $(\mathcal{G}, \mathcal{A}, \mathcal{P})$ is stable whatever adversary $\mathcal{A}$ and greedy scheduling policy $\mathcal{P}$ is considered. The proof follows the lines of that for the AQT model presented in [4].

Previously, we need to state some partial results in the form of lemmas. Let us consider some packet p in the system $(\mathcal{G}, \mathcal{A}, \mathcal{P})$. We suppose that it was injected in instant T_0 at the node 0 with some node d as destination. Let $T' > T_0$ be some time at which it was not yet absorbed. Let $0, 1, \ldots, s$ be the nodes visited by p in the interval $[T_0, T']$. The edge $(i, i+1)$ will be named edge i for simplicity. For $l = 0, \ldots, s$, let T_l denote the time at which p arrives at the output buffer of the edge l; by abuse of notation, we will also write $T_{s+1} = T'$.

For j an edge of $\mathcal{G}$, and $t \in [T_0, T']$, we define $A_t(j)$ as in Sect. 2, i.e., as the number of bits in the system that require to cross j and still have to be transmitted across link j at time t (recall that the bits in $Q_t(j)$ are included in $A_t(j)$ but those in $R_t(j)$ are not). Based on the maximum amount of packet's bits requiring any edge j that are injected by the adversary in an interval a time, and based on the amount bits consumed by j in that interval of time, we have the following basic property of $A_t(j)$, which follows from the definition:

Lemma 7 *Let t and t' be such that $t' \leq t$. Then*

$$A_t(j) \leq A_{t'}(j) + (1-\varepsilon)(t-t')B_j + b - z,$$

where z is the number of bits sent across edge j in the interval $[t', t]$.

We define

$$Q = \max_{j \in G,\ t \in [T_0, T')} A_t(j). \tag{1}$$

For j and t as before, we now define the function f as follows:

$$f(j,t) = \begin{cases} Q + bj, & \text{for } t = T_0, \\ Q - \varepsilon(t-T_0)B_{\min} + (b + 2P_{\max}B_{\max} + L_{\max})(j+1), & \text{for } t > T_0. \end{cases}$$

This function f is just a technicality that, taking time T_0 as reference, tries to quantify the maximum amount of packets in the system that require edge j at any certain time $t \geq T_0$. We note the following properties that follow from the definition of this function f.

Lemma 8

(i) $f(j,t) = f(j,T_0) - \varepsilon(t-T_0)B_{\min} + b + (1+j)(2P_{\max}B_{\max} + L_{\max})$, $\forall t > T_0$.

(ii) $f(j,t) = f(j,t') - \varepsilon(t-t')B_{\min}, \quad \forall t > t' > T_0$.

(iii) $f(j,t) = f(j-1,t) + b + 2P_{\max}B_{\max} + L_{\max}, \quad \forall t > T_0$.

Proof Property (i) is obtained from the following reasoning: if $t > T_0$, then

$$\begin{aligned} f(j,t) &= Q - \varepsilon(t-T_0)B_{\min} + (b + 2P_{\max}B_{\max} + L_{\max})(j+1) \\ &= Q - \varepsilon(t-T_0)B_{\min} + b(j+1) + (2P_{\max}B_{\max} + L_{\max})(j+1) \\ &= Q + bj - \varepsilon(t-T_0)B_{\min} + b + (2P_{\max}B_{\max} + L_{\max})(j+1) \\ &= f(j,T_0) - \varepsilon(t-T_0)B_{\min} + b + (2P_{\max}B_{\max} + L_{\max})(j+1), \end{aligned}$$

where the first equality states the definition of f, the second equality and the third equality apply distributiveness relative to b and $(j+1)$, and the forth equality applies the definition of $f(j,T_0)$. Property (ii) is obtained from the following reasoning: if $t>T_0$, then

$$\begin{aligned}f(j,t)&=Q-\varepsilon(t-T_0)B_{\min}+(b+2P_{\max}B_{\max}+L_{\max})(j+1)\\&=Q-(\varepsilon(t-t')+\varepsilon(t'-T_0))B_{\min}+(b+2P_{\max}B_{\max}+L_{\max})(j+1)\\&=f(j,t')-\varepsilon(t-t')B_{\min},\end{aligned}$$

by applying the definition of $f(j,t)$ in the first equality, replacing $(t-T_0)$ by $((t-t')+(t'-T_0))$ and applying distributiveness in the second equality, and by the definition of $f(j,t')$ in the third equality. Property (iii) is obtained from the following reasoning: if $t>T_0$, then

$$\begin{aligned}f(j,t)&=Q-\varepsilon(t-T_0)B_{\min}+(b+2P_{\max}B_{\max}+L_{\max})(j+1)\\&=Q-\varepsilon(t-T_0)B_{\min}+(b+2P_{\max}B_{\max}+L_{\max})((j-1+1)+1)\\&=(Q-\varepsilon(t-T_0)B_{\min}+(b+2P_{\max}B_{\max}+L_{\max})(j-1+1))\\&\quad+(b+2P_{\max}B_{\max}+L_{\max})\\&=f(j-1,t)+(b+2P_{\max}B_{\max}+L_{\max}),\end{aligned}$$

where the first equality states the definition of f, the second equality replaces $(j+1)$ by $(j-1+1+1)$, and the third equality applies distributiveness over $(j-1+1+1)$. □

Definition 2 If j is an edge of $\mathcal{G}$ and t is a time step, we say that the pair (j,t) is *applicable* if either

$$(j\in\{0,1,\dots,s\}\quad\text{and}\quad t\in[T_0,T_{j+1}])$$

or

$$(j>s\quad\text{and}\quad t\in[T_0,T']).$$

Note the following basic property of applicability.

Lemma 9 *If j's output buffer is empty at a time $t'\in[T_0,T_{j+1}-P_j)$, then $(j-1,t')$ is applicable.*

Proof First we can notice that t' cannot be in $[T_j,T_{j+1}-P_j)$, because the packet p arrived to j's output buffer at time T_j and the last bit of p fully leaves it at time $T_{j+1}-P_j$, then it cannot be empty during this interval. Now, if $t'\in[T_0,T_j]$ then the pair $(j-1,t')$ is applicable by Definition 2. □

The crux of our analysis is the following lemma.

Lemma 10 *For all applicable pairs (j,t), we have $A_t(j)\le f(j,t)$.*

Proof We prove the lemma by induction on $j \geq 0$, and for fixed j for all t.

First, we prove for $j = 0$ and for all t, which is the first step of the induction. In this case you cannotice that $(0, t)$ is applicable if and only if $t \in [T_0, T_1]$. In the case $(0, T_0)$ we have $f(0, T_0) = Q \geq A_{T_0}(0)$. Note that there is a continuous flow of bits sent across $j = 0$ during the interval $[T_0, T_1 - P_0]$, because the packet p arrives to $j = 0$ at time T_0 and the last bit of p completely leaves it at time $T_1 - P_0$. Then,

$$\begin{aligned}
A_t(0) &\leq A_{T_0}(0) + (1 - \varepsilon)(t - T_0)B_0 + b - (t - P_0 - T_0)B_0 \\
&= A_{T_0}(0) - \varepsilon(t - T_0)B_0 + b + P_0 B_0 \\
&\leq f(0, T_0) - \varepsilon(t - T_0)B_{\min} + b + P_{\max} B_{\max} \\
&= f(0, t) - P_{\max} B_{\max} - L_{\max} \leq f(0, t).
\end{aligned}$$

The first inequality follows from Lemma 7, while the next equality is just a manipulation of the expression right above. The second inequality follows from the definition of $B_{\min}$, $P_{\max}$ and $B_{\max}$ and the induction hypothesis. The next equality follows from Lemma 8, and the last inequality is a trivial deduction.

Now we want to prove the claim for all $j > 0$, for this we use induction on j. We assume, by induction hypothesis, that $A_t(j - 1) \leq f(j - 1, t)$ for all applicable $(j - 1, t)$, and we will prove it for j. For T_0 we have that (j, T_0) is applicable, and we have $f(j, T_0) = Q + bj \geq Q \geq A_{T_0}(j)$.

Now, consider any applicable pair (j, t), with $t > T_0$. If for all $t' \in [T_0, t - P_j)$ j's output buffer is not empty at time t', then

$$\begin{aligned}
A_t(j) &\leq A_{T_0}(j) + (1 - \varepsilon)(t - T_0)B_j + b - (t - P_j - T_0)B_j \\
&= A_{T_0}(j) - \varepsilon(t - T_0)B_j + b + P_j B_j \\
&\leq f(j, T_0) - \varepsilon(t - T_0)B_{\min} + b + P_{\max} B_{\max} \\
&= f(j, t) - (2P_{\max} B_{\max} + L_{\max})j - (P_{\max} B_{\max} + L_{\max}) \leq f(j, t).
\end{aligned}$$

As before, the first inequality follows from Lemma 7, while the next equality is just a manipulation of the expression right above. The second inequality follows from the definition of $B_{\min}$, $P_{\max}$ and $B_{\max}$ and the induction hypothesis. The next equality follows from Lemma 8, and the last inequality is a trivial deduction.

Otherwise, there is some step $t' \in [T_0, t - P_j)$ in which the j's output buffer is empty. Let us take the longest such $t' < t - P_j$. Notice that; by definition, (j, t) is applicable when $t \in [T_0, T_{j+1}]$. Then, in this case $t' \in [T_0, T_{j+1} - P_j)$. Now by Lemma 9 we have that the pair $(j - 1, t')$ is also applicable. Therefore,

$$\begin{aligned}
A_t(j) &\leq A_{t'}(j) + (1 - \varepsilon)(t - t')B_j + b - (t - P_j - t')B_j \\
&\leq A_{t'}(j - 1) + P_{j-1}B_{j-1} + L_{\max} + b + P_j B_j - \varepsilon(t - t')B_j \\
&\leq f(j - 1, t') + b + 2P_{\max} B_{\max} + L_{\max} - \varepsilon(t - t')B_{\min} \\
&= f(j, t') - \varepsilon(t - t')B_{\min} = f(j, t).
\end{aligned}$$

The first inequality follows from Lemma 7. The second inequality follows from skipping some terms of the expression right above and from the fact that, since the output buffer of j is empty at time t', then $A_{t'}(j) \leq A_{t'}(j-1) + P_{j-1}B_{j-1} + L_{\max}$. The induction hypothesis is applied in the third inequality, as well as the definition of $B_{\min}$, $P_{\max}$ and $B_{\max}$. The next two equalities are obtained from Lemma 8 and by manipulation of the previous expressions. □

Using this lemma, we now prove the following results:

Theorem 11 $(\mathcal{G}, \mathcal{A}, \mathcal{P})$ *is stable*, *and there are never more than*

$$(1-\varepsilon)\frac{(b+2P_{\max}B_{\max}+L_{\max})n}{\varepsilon B_{\min}}B_{\max}+b$$

bits in the system that require any given edge.

Proof The second statement implies the first, so we will concentrate on proving the second statement. Set

$$Q' = ((1-\varepsilon)(b+2P_{\max}B_{\max}+L_{\max})nB_{\max}/\varepsilon B_{\min}) + b,$$

and suppose that the theorem is not true. Let T' be the first time at which $Q'+1$ bits in the system require any edge. Let $T_0 < T'$ denote the time at which the oldest of these bits, which belongs to a packet p, was injected. Relabel the nodes to make 0 the node at which p was injected. The link of interest will become $k \leq n-1$. Then, at time T' packet p has not crossed edge k yet, and hence (k, T') is applicable. Note that for Q as defined in (1) we have $Q \leq Q'$. In interval $[T_0, T']$ at most

$$(T'-T_0)(1-\varepsilon)B_{\max}+b$$

bits can be injected requiring any edge. Therefore,

$$Q' \leq (T'-T_0)(1-\varepsilon)B_{\max}+b,$$

and hence

$$\begin{aligned}\varepsilon(T'-T_0)B_{\min} &\geq \varepsilon\frac{Q'-b}{(1-\varepsilon)B_{\max}}B_{\min}\\ &= \varepsilon\frac{(b+2P_{\max}B_{\max}+L_{\max})n}{\varepsilon B_{\min}}B_{\min}\\ &= (b+2P_{\max}B_{\max}+L_{\max})n. \end{aligned} \tag{2}$$

By our assumption we have $Q' < A_{T'}(k)$ and, by Lemma 10, we have $A_{T'}(k) \leq f(k, T')$. Then, by transitivity, we have that $Q' < f(k, T')$. Thus,

$$\begin{aligned}&Q' < f(k,T')\\ &\quad = Q - \varepsilon(T'-T_0)B_{\min} + (b+2P_{\max}B_{\max}+L_{\max})(1+k)\end{aligned}$$

$$\begin{aligned}&\le Q' - \varepsilon(T' - T_0)B_{\min} + (b + 2P_{\max}B_{\max} + L_{\max})n\\&\le Q' - (b + 2P_{\max}B_{\max} + L_{\max})n + (b + 2P_{\max}B_{\max} + L_{\max})n = Q'\end{aligned}$$

is a contradiction. In this latter reasoning, the first equality follows from the definition of $f(k, T')$. The second inequality follows from the fact that $k \le n - 1$, and thus $k + 1 \le n$; while the third inequality follows from the previously stated Inequality (2). Finally, the last equality is trivially correct. □

Theorem 12 *The maximum number of steps a packet spends in the system is*

$$\frac{B_{\max}((b + 2P_{\max}B_{\max} + L_{\max})n)}{\varepsilon^2 B_{\min}^2} + \frac{b}{\varepsilon B_{\min}} + P_{\max}.$$

Proof Suppose that a packet p is injected at time T_0, with origin 0 (w.l.o.g.) and destination $k + 1$ (mod n). Then k is the last edge it has to traverse. Suppose that p did not completely leave the output buffer of edge k at time T', where

$$T' = T_0 + \frac{B_{\max}((b + 2P_{\max}B_{\max} + L_{\max})n)}{\varepsilon^2 B_{\min}^2} + \frac{b}{\varepsilon B_{\min}}.$$

We can manipulate this expression and obtain the following equality,

$$T' - T_0 = \frac{1}{\varepsilon B_{\min}}\left(\frac{B_{\max}((b + 2P_{\max}B_{\max} + L_{\max})n)}{\varepsilon B_{\min}} + b\right),$$

and thus also the following one,

$$\varepsilon(T' - T_0)B_{\min} = \frac{B_{\max}((b + 2P_{\max}B_{\max} + L_{\max})n)}{\varepsilon B_{\min}} + b, \tag{3}$$

which will be useful later. By Theorem 11 we can apply Lemma 10 with

$$Q = (1 - \varepsilon)\frac{(b + 2P_{\max}B_{\max} + L_{\max})n}{\varepsilon B_{\min}}B_{\max} + b,$$

and we have

$$\begin{aligned}A_{T'}(k) &\le f(k, T')\\&= Q - \varepsilon(T' - T_0)B_{\min} + (b + 2P_{\max}B_{\max} + L_{\max})(1 + k)\\&\le Q - \frac{B_{\max}((b + 2P_{\max}B_{\max} + L_{\max})n)}{\varepsilon B_{\min}} - b + (b + 2P_{\max}B_{\max} + L_{\max})n\\&= (1 - \varepsilon)\frac{(b + 2P_{\max}B_{\max} + L_{\max})n}{\varepsilon B_{\min}}B_{\max} + b\\&\quad - \frac{B_{\max}((b + 2P_{\max}B_{\max} + L_{\max})n)}{\varepsilon B_{\min}} - b + (b + 2P_{\max}B_{\max} + L_{\max})n\\&= -\frac{(b + 2P_{\max}B_{\max} + L_{\max})n}{B_{\min}}B_{\max} + (b + 2P_{\max}B_{\max} + L_{\max})n\end{aligned}$$

$$= ((b + 2P_{\max}B_{\max} + L_{\max})n)\left(1 - \frac{B_{\max}}{B_{\min}}\right) \leq 0.$$

which is a contradiction. In this latter reasoning, the first inequality follows from Lemma 10, and the next equality follows from the definition of f. The second inequality follows from the fact that $k \leq n - 1$, and thus $k + 1 \leq n$, and from (3). The equalities coming afterwards result from the manipulation of the expressions and, finally, the last inequality is a triviality since $B_{\max}/B_{\min} \geq 1$. □

4.3 Characterization of Universal Stability

In this section we give a characterization for the property of universally stability of networks, thus identifying which networks are universally stable under CAQT model. Recall that a network $\mathcal{G}$ is universally stable if the system $(\mathcal{G}, \mathcal{P}, \mathcal{A})$ is stable for every greedy scheduling policy $\mathcal{P}$ and every (r, b)-adversary $\mathcal{A}$ (see Sect. 2). To characterize the property, we proceed in a similar way as done in [1], where a full characterization of the same property was provided under the AQT model.

In the following we will consider bounded (r, b)-adversaries with $r = 1 - \varepsilon < 1$.

Lemma 13 *If digraphs $\mathcal{G}_1$ and $\mathcal{G}_2$ are universally stable in the CAQT model, then so is any graph formed by joining them with edges that go only from $\mathcal{G}_1$ to $\mathcal{G}_2$.*

Proof Assume that we are working against an $(1 - \varepsilon, b)$ adversary. Since $\mathcal{G}_1$ is universally stable, any bit which is injected in $\mathcal{G}_1$ gets out of $\mathcal{G}_1$ within T_1 times steps, where T_1 is some constant that depend on $1 - \varepsilon$ and b. Some of these bits may then enter to $\mathcal{G}_2$. Now consider a time window T_2 units long. Any new bits that enter to $\mathcal{G}_2$ during this window must have been introduced during $T_1 + T_2$ continuous units. The number of bits introduced during this interval for an edge can be at most $(T_1 + T_2)(1 - \varepsilon)B_{\max} + b = T_2B_{\max}(1 - \varepsilon) + T_1B_{\max}(1 - \varepsilon) + b$.

Then we can say that these bits could have been introduced by an $(1 - \varepsilon, b')$ adversary, where $b' = b + T_1B_{\max}(1 - \varepsilon)$. By definition of universal stability, $\mathcal{G}_2$ is then stable against such adversary and therefore the traversal time for bits is bounded. Due to the finite upper bound on the length of the packets, this bound applies to packets and therefore also to the queue lengths in $\mathcal{G}$. □

As a consequence of the previous result we have the following theorem.

Theorem 14 *A digraph $\mathcal{G}$ is universally stable in the CAQT model if and only if all its strongly connected components are universally stable.*

Observe that this theorem, together with the previous lemma, assure the universal stability of unicyclic digraphs, i.e., those digraphs with only one cycle and possibly, directed subdigraphs rooted in the nodes of that unique cycle.

Once all these types of graphs (namely directed acyclic digraphs, rings and unicyclic digraphs, see Sects. 4.1 and 4.2) have been identified as universally stable, the next step towards the characterization of the property is to detect the simplest

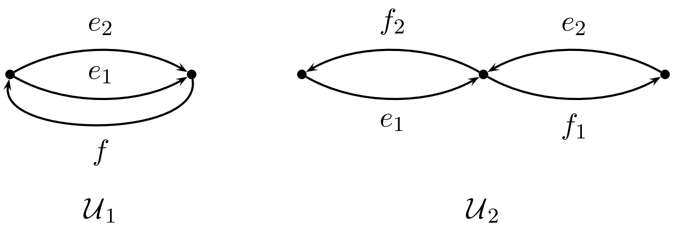

Fig. 2 Minimum forbidden subdigraphs characterizing the property of universal stability of networks [1]

digraphs that are not universally stable in the CAQT model. Then, we must consider the smallest non-unicyclic digraphs whose cycles are bilaterally connected, i.e., with edges that go from one cycle to another and vice versa. Those graphs are $\mathcal{U}_1$ and $\mathcal{U}_2$ as depicted in Fig. 2. Observe that $\mathcal{U}_1$ is the smallest non-unicyclic digraph whose cycles share an edge, and $\mathcal{U}_2$ is the smallest non-unicyclic digraph whose cycles share a vertex.

After these two digraphs $\mathcal{U}_1$ and $\mathcal{U}_2$, we want to consider the whole family of digraphs that contain the essence of their structure, i.e., those digraphs which are defined by iteratively extending their topologies. Given a digraph $\mathcal{G}$, we use the subdivision operations defined in [1] to build that family. We will denote as $\mathcal{E}(\mathcal{G})$ the family of digraphs formed by $\mathcal{G}$ and all the digraphs obtained from $\mathcal{G}$ by successive subdivision of any of the following two types to it:

Definition 3 (Subdivision operations [1])

– The *subdivision of an arc* (u, v) in a digraph $\mathcal{G}$ consists in the addition of a new vertex w and the replacement of (u, v) by (u, w) and (w, v).
– The *subdivision of a 2-cycle* (u, v), (v, u) in a digraph $\mathcal{G}$ 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).

In the AQT model, we know that neither the digraphs $\mathcal{U}_1, \mathcal{U}_2$ nor the families that they define by subdivision operations are universally stable. Even more, we know that these digraphs characterize the property of universal stability of networks. These two results are stated in the following lemmas from [1]:

Theorem 15 ([1]) *All digraphs in* $\mathcal{E}(\mathcal{U}_1) \cup \mathcal{E}(\mathcal{U}_2)$ *are not stable under the* NTG-LIS *protocol* (*in the* AQT *model*).

Theorem 16 ([1]) *A digraph is universally stable* (*in the* AQT *model*) *if and only if it does not contain as a subdigraphs any of the digraphs in* $\mathcal{E}(\mathcal{U}_1) \cup \mathcal{E}(\mathcal{U}_2)$.

Since the instability results from the AQT model can be extended to the CAQT model, we have that all digraphs in $\mathcal{E}(\mathcal{U}_1) \cup \mathcal{E}(\mathcal{U}_2)$ are not stable under the same scheduling policy in the CAQT model. Moreover, since the previously considered digraphs (namely acyclic ones, rings and unicyclic ones) are universally stable under the CAQT model, we can state that the property of universal stability of networks is characterized by the same family of digraphs as stated in Theorem 16.

Corollary 17 *A digraph is universally stable in the* CAQT *model if and only if it does not contain as a subdigraphs any of the digraphs in* $\mathcal{E}(\mathcal{U}_1) \cup \mathcal{E}(\mathcal{U}_2)$.

Corollary 18 *The property of universal stability of networks in the* CAQT *model can be decided in polynomial time.*

5 Stability of Queueing Policies

Stability can also be studied from the point of view of the protocols. Unstable scheduling policies in the AQT model are also unstable in the CAQT model. In the following, we show that the so-called LIS, SIS, FTG and NFS protocols are universally stable in the CAQT model, as they were in the AQT model. The techniques used here to show their universal stability under CAQT are inspired on those used in [4], however some of them introduce significant variations.

5.1 Universal Stability of LIS

The LIS (*longest-in-system*) scheduling policy gives priority to the packet which was earliest injected in the system. Independently of the network topology and the (r, b)-adversary, any system $(\mathcal{G}, \text{LIS}, \mathcal{A})$ is stable.

We first show a bound on the time that a packet needs to cross his path. Consider some packet p, injected at time T_0, and whose path crosses edges $e_1, e_2, \ldots, e_d$, in this order. We use T_i to denote the time instant in which p finishes crossing edge e_i, for $i = 1, \ldots, d$. Let t denote some time in $[T_0, T_d]$. Let us denote by g_t the injection time of the oldest packet in the system at time t. We define $c = \max_{t\in[T_0,T_d]} t - g_t$.

Lemma 19 $T_d - T_0 \leq (1-\varepsilon^d)c + \frac{(1-\varepsilon^d)(\frac{b}{B_{\min}}+P_{\max})}{1-\varepsilon}$.

Proof Packet p reaches the tail of edge e_i at time T_{i-1}. Thus, from the definition of c, only packets injected in the interval $[T_{i-1} - c, T_0]$ can block p in the queue of e_i. The packets injected in that interval include p and have at most $r(T_0 - T_{i-1} + c)B_{e_i} + b$ bits. Hence,

$$\begin{aligned}
T_i &\leq T_{i-1} + \frac{r(T_0 - T_{i-1} + c)B_{e_i} + b - L_p}{B_{e_i}} + \frac{L_p}{B_{e_i}} + P_{e_i} \\
&= \varepsilon T_{i-1} + (1-\varepsilon)(c + T_0) + \frac{b}{B_{e_i}} + P_{e_i} \\
&\leq \varepsilon T_{i-1} + (1-\varepsilon)(c + T_0) + \frac{b}{B_{\min}} + P_{\max}.
\end{aligned}$$

Thus, solving the recurrence, we obtain

$$\begin{aligned} T_d &\le \left((1-\varepsilon)(c+T_0)+\frac{b}{B_{\min}}+P_{\max}\right)\sum_{i=0}^{d-1}\varepsilon^i+\varepsilon^d\,T_0\\ &= \left((1-\varepsilon)(c+T_0)+\frac{b}{B_{\min}}+P_{\max}\right)\frac{1-\varepsilon^d}{1-\varepsilon}+\varepsilon^d\,T_0\\ &= (1-\varepsilon^d)\,c+\frac{(1-\varepsilon^d)(\frac{b}{B_{\min}}+P_{\max})}{1-\varepsilon}+T_0 \end{aligned}$$

and the claim follows. □

Theorem 20 *Let $\mathcal{G}$ be a network, $\mathcal{A}$ an (r,b)-adversary with $r=1-\varepsilon<1$, and d the length of the longest simple directed path in $\mathcal{G}$. Then all packets spend less than $(\frac{b}{B_{\min}}+P_{\max})/(r\varepsilon^d)$ time in the system $(\mathcal{G},\textsc{lis},\mathcal{A})$.*

Proof Let $c'=(\frac{b}{B_{\min}}+P_{\max})/((1-\varepsilon)\varepsilon^d)$ and assume that at time t is the first time that a packet p satisfies $t-g_t=c'$, i.e., the first time that a packet p has been in the system for c' time. We apply the previous Lemma 19 to this packet p. From that lemma, p should have been absorbed in at most

$$\begin{aligned} &(1-\varepsilon^d)\,c'+\frac{(1-\varepsilon^d)(\frac{b}{B_{\min}}+P_{\max})}{1-\varepsilon}\\ &\quad=c'-\frac{(\frac{b}{B_{\min}}+P_{\max})-(1-\varepsilon^d)(\frac{b}{B_{\min}}+P_{\max})}{1-\varepsilon}\quad<c' \end{aligned}$$

time, and a contradiction is reached. □

Corollary 21 *Let $\mathcal{G}$ be a network, $\mathcal{A}$ an (r,b)-adversary with $r=1-\varepsilon<1$, and d the length of the longest edge-simple directed path in $\mathcal{G}$. Then, the system $(\mathcal{G},\textsc{lis},\mathcal{A})$ is stable, and there are always less than $(\frac{b}{B_{\min}}+P_{\max})\,\varepsilon^{-d}B_{\max}+b$ bits trying to cross any edge e.*

5.2 Universal Stability of SIS

The SIS (*shortest-in-system*) scheduling policy gives priority to the packet which was injected the latest in the system. In the case of the SIS protocol, bounding the size of the packets recently injected is related to bounding the time that a packet p requires to cross the edge e. The following lemma provides us with such a bound:

Lemma 22 *Let p be a packet that, at time t, is waiting in the queue of edge $e\in E(\mathcal{G})$. At that instant, let $k-1$ be the total size in bits of the packets in the system that also require e and that may have priority over p (i.e., that were injected later in the system). Then p will start crossing e in at most $(k+b)/(\varepsilon B_e)$ units of time.*

Proof The lemma can be proved by contradiction. Suppose that the packet p is not transmitted across the edge e in the claimed units of time. Then, other packets different from p must have crossed the edge meanwhile. Those packets, must have also had priority over p; however, during that time, the only packets in the system that have priority over p are, either those comprising the $k-1$ bits existing at time t, or those that have been injected meanwhile. During that time, a total size of at most

$$(k-1)+\left((1-\varepsilon)\frac{k+b}{\varepsilon B_e}B_e+b\right)$$

bits belong to packets that have priority over p. The packet p would be transmitted right after these bits are transmitted; since these bits would require at most

$$\frac{(k-1)+((1-\varepsilon)\frac{k+b}{\varepsilon B_e}B_e+b)}{B_e}<\frac{k+b}{\varepsilon B_e}$$

units of time to be sent across e, this leads to a contradiction. □

Observe that, once the packet p starts being transmitted through the link e, it will only take $P_e + L_p/B_e$ units of time more until it crosses it completely. Using the bound obtained in Lemma 22 in a recursive way, we can derive more general bounds, thus proving the universal stability of the SIS scheduling policy.

Theorem 23 *Let $\mathcal{G}$ be a network, $\mathcal{A}$ an (r,b)-adversary with $r=1-\varepsilon<1$ and $b\geq 1$, and d the length of the longest edge-simple directed path in $\mathcal{G}$. The system $(\mathcal{G}$, SIS, $\mathcal{A})$ is stable and, moreover:*

- *No queue ever contains $k_d+L_{\max}$ bits, and*
- *No packet spends more than $(d(b+\varepsilon L_{\max})+\sum_{i=1}^{d}k_i)/(\varepsilon B_{\min})+dP_{\max}$ time in the system,*

where k_i is defined according to the following recurrence:

$$k_i=\begin{cases} b & \text{for } i=1,\\ k_{i-1}+(1-\varepsilon)\left(\dfrac{k_{i-1}+b}{\varepsilon B_{\min}}+\dfrac{L_{\max}}{B_{\min}}+P_{\max}\right)B_{\max}+b & \text{for } 1<i\leq d.\end{cases}$$

Proof We first show that, when a given packet p, with path $\Pi_p=\{e_1,\dots,e_{|\Pi_p|}\}$, $|\Pi_p|\leq d$, arrives at the queue of e_i, the total size of the packets with priority over p that also require some edge $e_j\in\Pi_p$ is at most k_i-1.

This can be proved by induction on the position i of the edge e_i in the path Π_p. The claim holds for $i=1$, since the only packets requiring any $e_j\in\Pi_p$ that at the time of p's injection could have priority over p are those injected at the same time; whose total size is at most $b-1$ bits (since $L_p\geq 1$). Let us now assume that the claim holds for some $i\geq 1$ (inductive hypothesis). Then, using the bound in Lemma 22, p will completely arrive at the queue of e_{i+1} in at most

$$\frac{k_i+b}{\varepsilon B_{e_i}}+\frac{L_p}{B_{e_i}}+P_{e_i}$$

units of time more. During that time, at most another

$$(1-\varepsilon)\left(\frac{k_i+b}{\varepsilon B_{e_i}}+\frac{L_p}{B_{e_i}}+P_{e_i}\right)B_{e_j}+b \le (1-\varepsilon)\left(\frac{k_i+b}{\varepsilon B_{\min}}+\frac{L_{\max}}{B_{\min}}+P_{\max}\right)B_{\max}+b$$

bits of packets requiring an edge e_j could be injected. According to the SIS policy, those packets have priority over p. Thus, when p has completely arrived to e_{i+1}, there is a total amount of at most

$$(k_i-1)+(1-\varepsilon)\left(\frac{k_i+b}{\varepsilon B_{\min}}+\frac{L_{\max}}{B_{\min}}+P_{\max}\right)B_{\max}+b=k_{i+1}-1$$

bits (belonging to packets in the system) which require an edge e and have priority over p. This validates the induction and proves the claim.

Suppose now that, at some point, there are $k_d+L_{\max}$ bits in the queue of some edge e. Then, the latest packet arrived into the queue found at least k_d bits there, which contradicts the above inductive proof. This proves the first claim of the theorem.

Also, by combining the initial claim with the bound in Lemma 22, we get that a packet p takes at most $\frac{k_i+b}{\varepsilon B_{e_i}}+\frac{L_p}{B_{e_i}}+P_{e_i}$ units of time to cross the ith edge in its path Π_p. Since every path defined over the graph $\mathcal{G}$ has length at most d, then no packet spends more than

$$\sum_{i=1}^{d}\left(\frac{k_i+b}{\varepsilon B_{e_i}}+\frac{L_p}{B_{e_i}}+P_{e_i}\right)\le\frac{d(b+\varepsilon L_{\max})+\sum_{i=1}^{d}k_i}{\varepsilon B_{\min}}+dP_{\max}$$

units of time in the system. This proves the second claim of the theorem. □

5.3 Universal Stability of FTG

The FTG (*farthest-to-go*) scheduling policy gives priority to the packet which still has to traverse the longest path until reaching its destination. We show that FTG is universally stable by using the fact that all the packets have to traverse at least one edge, and that all the packet go at most d edges further.

Theorem 24 *Let $\mathcal{G}$ be a network with $m=|E(\mathcal{G})|$ links, $\mathcal{A}$ an (r,b)-adversary with $r<1$ and $b\ge 1$, and d the length of the longest edge-simple directed path in $\mathcal{G}$. The system $(\mathcal{G}$, FTG, $\mathcal{A})$ is stable and:*

- *There are never more than k_1 bits in the system,*
- *No queue ever contains more than k_2+b bits, and*
- *No packet spends more than $dP_{\max}+(d(b+\varepsilon L_{\max})+\sum_{i=2}^{d}k_i)/(\varepsilon B_{\min})$ time in the system,*

where k_i is defined according to the following recurrence:

$$k_i=\begin{cases}0 & \textit{for } i>d,\\ mk_{i+1}+mb+\sum_{e\in E(\mathcal{G})}R_{\max}(e) & \textit{fo } 1\le i\le d.\end{cases}$$

Proof We claim that, for all i, the size of packets in the system that still have to cross at least i edges (they could be crossing one of these edges) is at most k_i. The proof is done by a backward induction on i.

The claim is trivial for $i > d$, since each packet has to cross at most d edges. By induction hypothesis we consider the claim true for $i + 1$. Consider now a particular edge e. We use $Q_t^i(e)$ to denote the number of bits in the queue of edge e that belong to packets that still have to cross *at least i* edges at time t. Let t' be the latest time no later than t such that $Q_{t'}^i(e) = 0$.[3] Then any packet p, part of whose bits are accounted in $Q_t^i(e)$, either was in some other edge at time t' (its bits could be in the output queue, in the reception buffer, or crossing the edge itself), and hence had at least $i + 1$ edges to cross, or else it has been injected after t'. Since the policy is FTG, the packets that have to cross at least i edges continuously cross edge e in the time frame $(t', t]$. Hence,

$$\begin{aligned} Q_t^i(e) &\leq k_{i+1} + ((1-\varepsilon)(t-t')B_e + b) - (t-t')B_e \\ &\overset{\text{(ind.hyp.)}}{=} k_{i+1} + b - \varepsilon(t-t')B_e. \end{aligned}$$

With this result, we can calculate the following:

(i) An upper bound on the total size of the packets *in the system* that still have to cross at least i edges:

Knowing that for a concrete edge e, $Q_t^i(e) \leq k_{i+1} + b - \varepsilon B_e(t - t')$, and that $R_t(e)$ bits are either crossing e or in its reception buffer at time t, we can conclude that the size of the packets in the system that still have to cross i or more edges at any time t is at most

$$\begin{aligned} \sum_{e\in E(\mathcal{G})} (Q_t^i(e) + R_t(e)) &\leq mk_{i+1} + mb + \sum_{e\in E(\mathcal{G})} (R_t(e) - \varepsilon(t-t')B_e) \\ &\leq mk_{i+1} + mb + \sum_{e\in E(\mathcal{G})} R_{\max}(e) = k_i, \end{aligned}$$

and hence the inductive step holds.

This trivially shows the first claim of the theorem, since no more than k_1 bits belong to packets that need to traverse at least one edge, which are all the packets in the system.

(ii) An upper bound on the size of the queue of any edge e:

The maximum queue size of any edge e can be calculated when considering $i = 1$, since all packets in the queue need to cross at least one edge (e itself). Then, independently of the time instant, $Q_{\max}(e) = \max_t Q_t^1(e) \leq k_2 + b$.

(iii) An upper bound on $t - t'$:

[3] Observe that, w.l.o.g. we can assume that the system had empty initial configuration (see Corollary 5), and thus such a time t' always exists.

Since for any concrete edge e, $Q_t^i(e) \geq 0$, we obtain that

$$t - t' \leq \frac{k_{i+1} + b}{\varepsilon B_e} \leq \frac{k_{i+1} + b}{\varepsilon B_{\min}},$$

and so the maximum amount of time that a packet p with path $e_1, e_2, \ldots, e_d$ spends in the system is bounded by

$$\sum_{i=1}^{d} \left(\frac{k_{i+1} + b}{\varepsilon B_{e_i}} + \frac{L_p}{B_{e_i}} + P_{e_i} \right) \leq \frac{d(b + \varepsilon L_{\max}) + \sum_{i=2}^{d} k_i}{\varepsilon B_{\min}} + d P_{\max}. \qquad \square$$

5.4 Universal Stability of NFS

The NFS (*nearest-to-source*) scheduling policy gives priority to the packet which is closest to its origin, i.e., which has traversed the less portion of its whole path. We show that NFS is universally stable by using a similar argument as the one used for FTG; however the bounds will be provided now taking the length of the longest path as a reference point.

Theorem 25 *Let $\mathcal{G}$ be a network with $m = |E(\mathcal{G})|$ links, $\mathcal{A}$ an (r, b)-adversary with $r < 1$ and $b \geq 1$, and d the length of the longest edge-simple directed path in $\mathcal{G}$. The system $(\mathcal{G}$, NFS, $\mathcal{A})$ is stable and:*

- *There are never more than k_d bits in the system,*
- *No queue ever contains more than $k_{d-1} + b$ bits, and*
- *No packet spends more than $d P_{\max} + (d(b + \varepsilon L_{\max}) + \sum_{i=1}^{d-1} k_i)/(\varepsilon B_{\min})$ time in the system,*

where k_i is defined according to the following recurrence:

$$k_i = \begin{cases} 0 & \text{for } i = 0, \\ m k_{i-1} + m b + \sum_{e \in E(\mathcal{G})} R_{\max}(e) & \text{for } 1 \leq i \leq d. \end{cases}$$

Proof We claim that, for all i, the total size of the packets in the system that have (completely) crossed less than i edges is at most k_i. The proof is done by induction on i.

The claim is trivial for $i = 0$, since it is not possible to cross less than 0 edges. By induction hypothesis we consider the claim true for $i - 1$. Consider now a particular edge e. We use $Q_t^i(e)$ to denote the bits in the queue of edge e that belong to packets that have crossed *less than i* edges at time t. Let t' be the latest time no later than t such that $Q_{t'}^i(e) = 0$.[4] Then, any packet p whose bits are accounted in $Q_t^i(e)$ either was at some other edge at time t' (its bits could be in the output queue, in the reception buffer, or crossing the edge itself), and hence had crossed less than $i - 1$ edges, or

[4]Observe that, w.l.o.g. we can assume that the system had empty initial configuration (see Corollary 5), and thus such a time t' always exists.

else it has been injected after t'. Since we consider a greedy protocol, packets have been continuously sent across edge e in the interval $(t', t]$. Hence,

$$\begin{aligned} Q_t^i(e) \quad &\leq \quad k_{i-1} + ((1-\varepsilon)(t-t')B_e + b) - (t-t')B_e \\ &\overset{\text{(ind.hyp.)}}{=} k_{i-1} + b - \varepsilon(t-t')B_e. \end{aligned}$$

With this result, we can calculate the following:

(i) An upper bound on the total size of the packets *in the system* that have crossed less than i edges:

Knowing that for a concrete edge e, $Q_t^i(e) \leq k_{i-1} + b - \varepsilon B_e(t-t')$, and that $R_t(e)$ bits are either crossing e or in its reception buffer at time t, we can conclude that the size of the packets in the system that have crossed less than i edges at any time t is at most

$$\begin{aligned} \sum_{e\in E(\mathcal{G})} (Q_t^i(e) + R_t(e)) &\leq mk_{i-1} + mb + \sum_{e\in E(\mathcal{G})} (R_t(e) - \varepsilon(t-t')B_e) \\ &\leq mk_{i-1} + mb + \sum_{e\in E(\mathcal{G})} R_{\max}(e) \\ &= k_i, \end{aligned}$$

and hence the inductive step holds.

This trivially shows the first claim of the theorem, since no more than k_d bits belong to packets that have traversed less than d edges, which are all the packets in the system.

(ii) An upper bound on the size of the queue of any edge e:

The maximum queue size of any edge e can be calculated when considering $i = d$, since no packet in the queue has crossed d edges. Then, independently of the time instant, $Q_{\max}(e) = \max_t Q_t^d(e) \leq k_{d-1} + b$.

(iii) An upper bound on $t - t'$:

Since for any concrete edge e, $Q_t^i(e) \geq 0$, we obtain that

$$t - t' \leq \frac{k_{i-1}+b}{\varepsilon B_e} \leq \frac{k_{i-1}+b}{\varepsilon B_{\min}},$$

and so the maximum amount of time that a packet p with path $e_1, e_2, \dots, e_d$ spends in the system is bounded by

$$\sum_{i=1}^{d}\left(\frac{k_{i-1}+b}{\varepsilon B_{e_i}} + \frac{L_p}{B_{e_i}} + P_{e_i}\right) \leq \frac{d(b+\varepsilon L_{\max}) + \sum_{i=1}^{d-1} k_i}{\varepsilon B_{\min}} + dP_{\max}. \qquad \square$$

6 Instability of Queueing Policies

In this section we introduce some new scheduling policies, namely LPL-LIS, SPL-NFS, and SPP-NFS, that base their policies in the main features of the CAQT model, namely, the length of the packets, the edge bandwidths and the edge propagation delays, respectively. We show that the CAQT model differs from the AQT model in some aspects concerning the stability of these scheduling policies. More precisely, we will see that these policies are unstable under the CAQT model, while they are universally stable under the AQT model. Thus it turns out that, apart from allowing more scenarios for instability, the consideration of the CAQT model transforms into instability scenarios which were universally stable in the AQT model. Additionally, these scheduling policies do not only represent a distinction between the AQT and the CAQT models, but also inside the CAQT model itself: the LPL-LIS, SPL-NFS, and SPP-NFS are universally stable scheduling policies when uniform values are given to the packet lengths, bandwidths, and propagation delays, respectively; but result in unstable systems when the values considered for those features differ.

6.1 Instability by Difference in Packet Length

Consider the LPL (*longest-packet-length*) scheduling policy, which gives priority to the packet with longest length. Let us denote as LPL-LIS the same policy when ties are broken according to the LIS policy. Note that LPL-LIS is universally stable under the AQT model, since in this model all packets have the same length and hence the policy simply becomes LIS [4]. However, we show here that LPL-LIS is unstable in an extension of AQT with multiple packet lengths just by considering two different packet lengths (1 and 2). For simplicity we will assume that time advances in synchronous steps (as in AQT). Packets of length 2 take 2 steps to cross each link. In the LPL-LIS policy, these double packets will have priority over the single packets. Note that this model is trivially included in CAQT. To show the instability of the LPL-LIS policy, we use the baseball network presented in [4] (see Fig. 3).

Theorem 26 *Let $\mathcal{G}_B$ be the graph with nodes $V(\mathcal{G}_B) = \{v_0, v_1, w_0, w_1\}$, and edges $E(\mathcal{G}_B) = \{(v_0, w_0), (v_1, w_1), (w_1, v_0), (w_1, v_0), (w_0, v_1), (w_0, v_1)\}$. All the edges in $E(\mathcal{G}_B)$ have bandwidth 1 and null propagation delay. For $r > 1/\sqrt{2}$ there is an (r, b)-adversary $\mathcal{A}$ that makes the system $(\mathcal{G}_B$, LPL-LIS, $\mathcal{A})$ to be unstable only with packets of length 1 and 2.*

Proof We assume an initial configuration in which there are s_0 packets of length 1, distributed roughly evenly among w_1 and v_1, and all of them trying to cross e_0, for

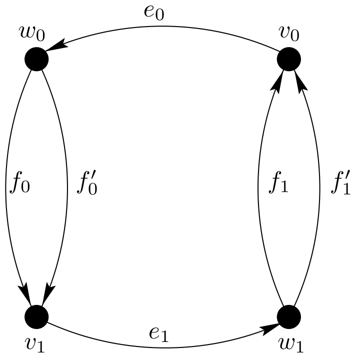

Fig. 3 Baseball network $\mathcal{G}_B$ presented in [4]

a large enough s_0. This is the base case of the induction. Then, we show that at the end of phase j there are at least $s_0 + j$ packets of length 1 distributed evenly among w_{1-i} and v_{1-i} that want to cross edge e_i, where $i = j \bmod 2$. This clearly holds for phase 0.

Then, we consider phase j. Without loss of generality we assume j to be odd. By induction hypothesis at the beginning of phase j there is a set S of s packets of length 1 that want to cross e_0 in w_1 and v_1. The sequence of injections is divided into subphases as follows.

(i) We let one step go to allow the first packet in S to reach the queue of e_0 (note that after then all the packets will follow without interruption). Then, for the next s steps, we inject a set S_1 of rs packets of length 1 that want to traverse edges $e_0 f_0' e_1$. These are blocked by the packets in S, since they all have the same length, but those in S are older.

(ii) Then, for the next rs steps we inject a set of $r^2 s$ packets of length 1 that want to traverse edges $e_0 f_0 e_1$. These are blocked by the packets in S_1.

We also delay the flow of packets in S_1 through f_0' using single–edge injections of packets of length 2. The new packets block the packets in S_1, since they are longer. Roughly $r^2 s/2$ packets of length 2 can be injected, and hence roughly $rs - 2r^2 s/2$ packets of S_1 get to cross f_0'. Then, at the end of this sub-phase there are still $r^2 s$ packets of S_1 in w_0.

This completes phase j. Clearly, at the end of it there are $r^2 s$ packets in w_0 and $r^2 s$ packets in v_0, all of length 1 and trying to traverse e_1. Since $2r^2 s > s$ for large enough s, the induction hypothesis holds. □

A deeper understanding of the LPL-LIS scheduling policy, together with the previous theorem, leads us to the following observation:

Observation 1 *The* LPL-LIS *scheduling policy is universally stable when all the packet lengths are given the same value because, in that case, it behaves like* LIS. *However, with different packet length values, it can become unstable.*

6.2 Instability by Difference in Bandwidth

Consider the SPL (*slowest-previous-link*) scheduling policy, which gives priority to the packet whose last crossed link was the slowest, i.e., had the smallest bandwidth. This policy aims to equilibrate the lost in transmission velocity suffered in previous links. Let us denote as SPL-NFS this policy, when ties are broken according to the NFS policy. Observe that the SPL-NFS scheduling policy is equivalent to NFS in the AQT model and thus universally stable [4]. However, we show that in a similar way as shown for the LPL-LIS policy, the SPL-NFS policy can be made unstable in the CAQT model.

Theorem 27 *Let $\mathcal{G}_B$ be the graph with nodes $V(\mathcal{G}_B) = \{v_0, v_1, w_0, w_1\}$ and edges $E(\mathcal{G}_B) = \{(v_0, w_0), (v_1, w_1), (w_1, v_0), (w_1, v_0), (w_0, v_1), (w_0, v_1)\}$. Let $\mathcal{G}$ be the graph obtained from $\mathcal{G}_B$ whose set of nodes is $V(\mathcal{G}) = V(\mathcal{G}_B) \cup \{v_0', v_1', w_0', w_1'\}$,* *and whose set of edges is* $E(\mathcal{G}) = E(\mathcal{G}_B) \cup \{(v_0', v_0), (v_1', v_1), (w_0', w_0), (w_1', w_1)\}$. *Those edges inciding to* v_0 *and* v_1 *have bandwidth* 2, *while the rest have bandwidth* 1. *All the edges have null propagation delays. For* $r > 1/\sqrt{2}$ *there is an* (r, b)*-adversary* $\mathcal{A}$ *that makes the system* $(\mathcal{G}, \text{SPL-NFS}, \mathcal{A})$ *to be unstable.*

Proof The proof is similar to the proof of Theorem 26 above but the injections are done only at the queues of the new 4 edges. Using the same induction hypothesis and sequence of injections we have that in the first sub-phase the packets in S still block those in S_1 because the latest edge crossed by the latter is faster, while in the second sub-phase the single injections (now injections of path length 2) block the packets in S_1 by NFS. □

A deeper understanding of the SPL-NFS scheduling policy, together with the previous theorem, leads us to the following observation:

Observation 2 *The* SPL-NFS *scheduling policy is universally stable when all the edge bandwidths are given the same value because, in that case, it behaves like* NFS. *However, with different edge bandwidth values, it can become unstable.*

6.3 Instability by Difference in Propagation Delays

Consider the SPP (*smallest-previous-propagation*) scheduling policy, which gives priority to the packet whose previously traversed edge had smallest propagation delay, and combine it with NFS to break ties. Let us denote this policy as SPP-NFS. Observe that the SPP-NFS policy is equivalent to NFS in the AQT model and thus universally stable [4]. However, we show with the SPP-NFS policy as example, that just the fact of considering propagation delays can make a policy unstable in CAQT.

Theorem 28 *Let* $\mathcal{G}_B$ *be the graph with nodes* $V(\mathcal{G}_B) = \{v_0, v_1, w_0, w_1\}$ *and edges* $E(\mathcal{G}_B) = \{(v_0, w_0), (v_1, w_1), (w_1, v_0), (w_1, v_0), (w_0, v_1), (w_0, v_1)\}$. *Let* $\mathcal{G}$ *be the graph obtained from* $\mathcal{G}_B$ *whose set of nodes is* $V(\mathcal{G}) = V(\mathcal{G}_B) \cup \{v_0', v_1', w_0', w_1'\}$, *and whose set of edges is* $E(\mathcal{G}) = E(\mathcal{G}_B) \cup \{(v_0', v_0), (v_1', v_1), (w_0', w_0), (w_1', w_1)\}$. *Those edges inciding to* v_0 *and* v_1 *have propagation delay* 1, *while the rest have null propagation delay. All the edges have unary bandwidth. For* $r > 1/\sqrt{2}$ *there is an* (r, b)*-adversary* $\mathcal{A}$ *that makes the system* $(\mathcal{G}, \text{SPP-NFS}, \mathcal{A})$ *to be unstable.*

Proof The proof is similar to the proof of Theorem 26 above but the injections are done only at the queues of the new 4 edges. Packets in S block those in S_1 because the latter crossed an edge with larger propagation delay. Then, the single (two-edge in this case) injections block the packets in S_1 by NFS. □

A deeper understanding of the SPP-NFS scheduling policy, together with the previous theorem, leads us to the following observation:

Observation 3 *The* SPP-NFS *scheduling policy is universally stable when all the edge propagation delays are given the same value because, in that case, it behaves like* NFS. *However, with different edge propagation delay values, it can become unstable.*

7 Conclusions and Open Questions

We consider a networking scenario in which packets can have arbitrary lengths, and the network links may have different speeds and propagation delays. Taking into account these features, we have presented a generalization of the well-known Adversarial Queueing Theory (AQT) model which does not assume anymore synchronicity in the evolution of the system, and makes it more appropriate for more realistic continuous scenarios. We called it the CAQT model, which states for *continuous* AQT model.

We have shown that, in the CAQT model having bounded queues is equivalent to having bounded packet end-to-end delays. From the network point of view, we show that networks with a directed acyclic topologies are universally stable even when the traffic pattern fully loads the links. Networks with a ring topology are also universally stable in the CAQT model for injection rates smaller than the unity. In fact, we have provided a characterization of the property of universal stability of networks under the CAQT model, which turns out to coincide with the characterization under the AQT model and, thus, of polynomial decidability.

From the protocol point of view, we have also shown that the well-known LIS, SIS, FTG and NFS scheduling policies remain universally stable in the CAQT model. New scheduling policies have also been proposed which are universally stable in the AQT model but unstable in the CAQT model. It would be of interest to find other scheduling policies that do not use specific properties of the AQT model and which might be stable in that model, but unstable in the CAQT model. For those policies that turn out to be stable, lower and upper bounds on their stability (w.r.t. the injection ratio) could be studied. Also from the protocol point of view, it would be of interest to establish a relation between systems with and without propagation delays. This would provide us with more intuitions on the power of the different features, i.e., packet lengths, bandwidths and propagation delays, involved in the CAQT model.

Finally, it would also be of interest to know something about the queue sizes to be expected (as it was studied in [4, 16] for the AQT model), as well as which conditions guarantee that all the packets are actually delivered to destination (as it was studied in [15] for AQT).

References

1. Àlvarez, C., Blesa, M., Serna, M.: A characterization of universal stability in the adversarial queueing model. SIAM J. Comput. **34**, 41–66 (2004)
2. Àlvarez, C., Blesa, M., Serna, M.: The impact of failure management on the stability of communication networks. In: 10th International Conference on Parallel and Distributed Systems, pp. 153–160. IEEE Computer Society Press, Los Alamitos (2004)
3. Àlvarez, C., Blesa, M., Díaz, J., Fernández, A., Serna, M.: Adversarial models for priority-based networks. Networks **45**, 23–35 (2005)
4. Andrews, M., Awerbuch, B., Fernández, A., Kleinberg, J., Leighton, T., Liu, Z.: Universal stability results for greedy contention–resolution protocols. J. ACM **48**, 39–69 (2001)
5. Anshelevich, E., Kempe, D., Kleinberg, J.: Stability of load balancing algorithms in dynamic adversarial systems. In: 34th Annual ACM Symposium on Theory of Computing, pp. 399–406. ACM Press, New York (2002)

6. Awerbuch, B., Berenbrink, P., Brinkmann, A., Scheideler, C.: Simple routing strategies for adversarial systems. In: 42th IEEE Symposium on Foundations of Computer Science, pp. 158–167. IEEE Computer Society Press, Los Alamitos (2001)
7. Bhattacharjee, R., Goel, A., Lotker, Z.: Instability of FIFO at arbitrarily low rates in the adversarial queueing model. SIAM J. Comput. **34**, 318–332 (2004)
8. Borodin, A., Kleinberg, J., Raghavan, P., Sudan, M., Williamson, D.: Adversarial queueing theory. J. ACM **48**, 13–38 (2001)
9. Borodin, A., Ostrovsky, R., Rabani, Y.: Stability preserving transformations: Packet routing networks with edge capacities and speeds. In: 12th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA'01), pp. 601–610 (2001). Full version in [10]
10. Borodin, A., Ostrovsky, R., Rabani, Y.: Stability preserving transformations: packet routing networks with edge capacities and speeds. J. Interconnect. Netw. **5**(1), 1–12 (2004)
11. Cruz, R.: A calculus for network delay. Part I (network elements in isolation) and II (network analysis). IEEE Trans. Inform. Theory **37**, 114–141 (1991)
12. Echagüe, J., Cholvi, V., Fernández, A.: Universal stability results for low rate adversaries in packet switched networks. IEEE Commun. Lett. **7**, 578–580 (2003)
13. Koukopoulos, D., Mavronicolas, M., Spirakis, P.: Instability of networks with quasi-static link capacities. In: 10th International Colloquium on Structural Information Complexity. Proceedings in Informatics, vol. 17, pp. 179–194. Carleton Scientific (2003)
14. Koukopoulos, D., Mavronicolas, M., Spirakis, P.: Performance and stability bounds for dynamic networks. In: 7th International Conference on Parallel Architectures, Algorithms and Networks, pp. 239–246. IEEE Computer Society Press, Los Alamitos (2004)
15. Rosén, A., Tsirkin, M.: On delivery times in packet networks under adversarial traffic. In: 16th ACM Symposium on Parallel Algorithms and Architectures, pp. 1–10. ACM Press, New York (2004)
16. Weinard, M.: The necessity of timekeeping in adversarial queueing. In: 4th International Workshop on Efficient and Experimental Algorithms, LNCS, vol. 3503, pp. 440–451. Springer, Berlin (2005)