On Delivery Times in Packet Networks under Adversarial Traffic *

Adi Rosén[†] Michael S. Tsirkin[‡]

Abstract

We consider packet networks and make use of the "adversarial queuing theory" model [10]. We are interested in the question of guaranteeing that all packets are actually delivered to destination, and of having an upper bound on the delivery times of all packets. Whether this is possible against all adversarial queuing theory rate-1 adversaries was previously posed as an open question [13, 10].

Among other things, we give a queuing policy that guarantees bounded delivery time whenever the rate-1 adversary injects a sequence of packets for which there exists a schedule with a finite upper bound on the delivery times of all packets, and adheres to certain additional conditions. On the negative side we show that there exist rate-1 sequences of packets for which there is no schedule with a finite upper bound on the delivery times of all packets. We thus answer an open question posed by Gamarnik [13]. We further show that delivering *all* packets while maintaining stability (we coin the term "reliability" for this property) can be done by an offline scheduler whenever the injection of packets is done at rate of at most 1. But, on the other hand, we also show that there is no online protocol (even centralized) that can achieve that property against all rate-1 adversaries. We thus answer an open question of Borodin et al. [10].

* An early version of this paper appeared in the proceedings of the 16th SPAA, 2004.

[†] Dept. of Computer Science, Technion, Haifa 32000, Israel. e-mail: adiro@cs.technion.ac.il.

[‡] Dept. of Computer Science, Technion, Haifa 32000, Israel. e-mail: misha@cpan.org.

1 Introduction

The analysis of packet networks under adversarial injection of traffic received increasing attention in recent years (see e.g., [10, 5, 15, 3, 13, 14, 4, 12, 17, 9, 11]). Much of this work makes use of the model of "adversarial queuing theory", introduced by Borodin et al. [10]. This model can be briefly described as follows. Time proceeds in discrete steps. In each step, packets are injected into the network with their routes. Each packet traverses its respective route hop by hop in a store-and-forward manner. In each time step, one packet may cross each link, and all other packets waiting for that link are stored in a queue at the tail of that link. The behavior of the system is determined by the employed *queuing policy*, which chooses, at each time step, for each link, which of the competing packets is forwarded over that link. In the framework of adversarial queuing theory, the injection of packets into the network is modeled as being done by an *adversary*. The adversary is characterized by a *rate* at which packets are injected. Intuitively, the rate of injection is said to be r if for every link e in the network, the average number of packets requiring e injected by the adversary per time step, is at most r (a formal definition of the model is given in Section 2).

When analyzing the system one is interested in two main questions. Whether upper bounds on the queue sizes can be given, and whether upper bounds on the delivery times (a.k.a. delays) of the packets can be given. In particular, the question of *stability* received considerable attention in the literature. The system is said to be stable when there is a finite upper bound on the size of the queues, as opposed to their sizes growing to infinity as time proceeds. A considerable number of important results have been obtained in analyzing under what conditions stability can be achieved (see e.g., [10, 5, 15, 13, 12, 9]). The question of upper bounds on packet delays received less of direct attention, and results pertaining to this quantity were usually given in conjunction with results on queue sizes. In fact, when the injection rate into the network is strictly less than 1, and a greedy (i.e., work conserving) queuing policy is used, stability implies certain bounds on the delay of the packets [10, 5]. Greedy protocols that achieve stability on any network topology when the injection rate is bounded away from 1 (i.e., $r < 1$) are known (e.g., Longest-In-System) [5], and thus upper bounds on packet delays can be given in this case. However, the resulting bounds depend on the injection rate, and grow as $\frac{1}{1-r}$ grows. Gamarnik showed that there are greedy protocols that achieve stability as long as the injection rate does not exceed the link capacity (i.e., for any injection rate $r \leq 1$) [13], with bounds on queue size that do not depend on the injection rate r. However, all protocols that are presently known to preserve stability against any rate $r \leq 1$ adversary suffer from the drawback that they cannot guarantee that all packets eventually arrive to their destinations (i.e., some packets may be left undelivered forever). In fact, Gamarnik [13] gave a scenario with injection rate 1, such that when applying the Nearest-To-Origin protocol some packets never reach their destinations (This scenario can be modified to suit the similar Furthest-To-Go protocol). Clearly, guaranteeing that all packets are delivered, and preferably with an upper bound on the delivery times of all packets, is a desirable property. Gamarnik thus posed the open problem *"whether there exists a policy with bounded delivery time for every packet when $r = 1$"* [13]. Borodin et al. slightly rephrased the question and asked *"if there is a scheduling rule ... that can guarantee bounded or at least finite delivery time with respect to deterministic rate 1 adversaries"* [10]. This is the starting point of our work.

We show that there are sequences of packets injected at rate $r = 1$ such that there is no finite M such that all packets can be delivered (even by a centralized, clairvoyant scheduler) within M time steps from their injection time. Therefore, a protocol as suggested in the open question of Gamarnik does not exist, and we answer in the negative his question. We then turn to the question whether there is a protocol that can deliver *all* packets to their destinations (not leaving in the network any packet undelivered forever) whenever $r \leq 1$. To treat this question we coin, in parallel to the term stability, the term *reliability*. Reliability is a stronger notion than stability, guaranteeing that all packets are actually delivered, while stability is maintained. We first show that for any sequence of packets injected at rate $r \leq 1$, an offline scheduler can maintain reliability. It follows that a sequence of packets has a schedule that maintains reliability if and only if it has a schedule that maintains stability (those sequences characterized by [13, 18]). But we also show that, as opposed to stability, there are network topologies on which there is no online protocol (even centralized) that achieves reliability against all rate-1 adversaries. Thus we answer in the negative the open question posed by Borodin et al. [10], whether stability and finite delivery times for all packets can be achieved against all adversaries of rate $r = 1$.[1]

As discussed above, we show that for some sequences of packets injected at rate $r = 1$, it is impossible (even by an offline scheduler) to have a finite upper bound on the delivery times of all packets. That is, an adversary that is committed to have itself a schedule for the packets such that there is some finite bound on the delivery times of all packets, is a weaker adversary than the rate-1 adversary. We call this adversary a *bounded delivery time adversary*. The aim is thus to design protocols that will have a finite upper bound on the delivery times of all packets against such adversaries. We show a class of network topologies, and a protocol that on this class of topologies achieves bounded delivery time against any bounded

[1] We note that it is easy to guarantee that all packets are delivered if stability is not in question, e.g., by using FIFO or LIS.

delivery time adversary. We then design a new protocol, called *Estimated-Rare First* (ERF), that *on any network topology* achieves bounded delivery time against any adversary in a class that is somewhat weaker than the class of all bounded delivery time adversaries.

Summary of results First, we coin the term *reliability* to be the property that all packets are delivered to their destinations while stability is maintained (Definition 3). We give the following results concerning reliability.

1. We characterize the sequences of packets that allow reliability (i.e., sequences for which there exists a schedule that maintains reliability). Specifically, we show that the set of sequences that allow reliability is equal to the set of sequences that can be injected by rate 1 adversaries (Theorem 5).

2. We characterize the set of network topologies on which there is any online protocol (centralized or not) which achieves reliability whenever reliability can be maintained by an offline scheduler (i.e., whenever the adversary injects a sequence of packets for which reliability can be maintained). We show that on networks that contain simple cycles of length $N > 2$ there is no online protocol (even centralized) that achieves reliability against all rate $r \leq 1$ adversaries (Theorem 12). And for networks not containing simple cycles of length $N > 2$ we demonstrate a protocol that achieves reliability against all rate $r \leq 1$ adversaries (Theorem 20).

For *bounded delivery time* we give the following results.

1. We show that (on certain networks) the set of sequences of packets for which there exists a schedule with a finite upper bound on the delivery times of all packets, is a *proper subset* of the set of sequences given by rate $r \leq 1$ adversaries (Theorem 22 and Observation 21). We thus coin the term *bounded delivery time adversary* for the adversary which is committed to have itself such a schedule for the injected packets (Definition 24).

2. We give a set of network topologies (networks that do not contain simple cycles of length $N > 2$) for which there is a protocol that achieves bounded delivery time against any bounded delivery time adversary, and demonstrate such a protocol. We design a new protocol, called *Estimated-Rare First* (ERF), which achieves bounded delivery time, *on any network topology*, whenever the sequence of packets is given by a bounded delivery time adversary, which is also a "frequent adversary", a term that we define in the sequel (Theorem 36).

Organization The rest of the paper is organized as follows. In Section 2 we formally define the model and give several other preliminary definitions. In Section 3 we give our results for reliability, and in Section 4 we give our results for bounded delivery time. We close the paper in Section 5 with a discussion and open problems.

2 Model and Preliminaries

We model the network as a directed graph $G = (V, E)$, where the nodes represent switches, and the edges represent communication links. We denote by d the length of the longest simple path in G. At the tail of each edge there is a *buffer* (we sometimes use the term *queue* instead). Packets of uniform size are *injected* into the network over time with a prescribed *simple path* to follow from their respective source to their respective destination. The system is synchronous, and time proceeds in *discrete time steps* $t \in \{0, 1, 2, 3, \ldots\}$. Each time step is divided into two sub-steps. In the first sub-step, for each edge, at most one packet is extracted from the buffer associated with the edge and is sent across that edge. In the second sub-step each packet sent in the first sub-step arrives to the node on the other end of the edge; the packet is absorbed (i.e., eliminated from the network) if this node is its destination; otherwise, the packet is placed in the buffer at the tail of the next edge on its path. In addition, new packets are injected into the network in the second sub-step, and are then placed in the buffer at the tail of the first edge on their path. We assume that at time $t = 0$ the buffers of all edges are empty; then the adversary injects packets into the network starting at time $t = 1$. Our proofs will go through with minor changes for the case that the system may start with non-empty buffers.

A *protocol* is an online, and typically local control, algorithm which, at each time step t, and for each edge e, selects a packet to cross edge e from among the packets in the buffer of e (or decides to leave the link idle). A protocol is said to be *greedy* if it does not leave an edge e idle unless the buffer of e is empty.

The injection of packets into the network is modeled as being done by an *adversary*. Following [10, 5], we use the following parameterized definition for the adversary.

Definition 1: Let A be an adversary. A is called an (r, b) adversary, for $r \geq 0$ and $b \geq 0$, if for each edge $e \in E$ and for each time interval $I = (t_1, t_2]$, the number of packets injected by A during I with paths that include e is at most $r(t_2 - t_1) + b$.

For a given sequence of packets, σ, a *schedule* is a specification for the movement of all the packets in σ. A schedule is *valid* if (1) at most one packet is sent across each edge in each time step; (2) a packet is sent across any given edge on its path only once, and only after it arrives to the tail of that edge, according to the same schedule. We note that a protocol generates, in an online and typically local control manner, a schedule for the injected packets.

We define a *network system* as the tuple (G, σ, S) where G is a directed graph, σ a sequence of packets, and S a valid schedule for σ. Following [10] we define the notion of *stability* as follows.

Definition 2: A network system (G, σ, S) is *stable* if there exists $B < \infty$ such that the number of packets in the network at any given time is at most B.

As noted by Gamarnik [13] even when a system is stable some packets may stay in the network indefinitely, and never reach their destinations. We therefore define the notions of eternal packets and of reliability, as follows.

Definition 3: Let (G, σ, S) be a network system. A packet p is *eternal* in (G, σ, S) if it never arrives to its destination in (G, σ, S). The system (G, σ, S) is said to be *reliable* if it is stable and there are no eternal packets in (G, σ, S).

We further give the following definition for bounded delivery time, which is a stronger notion than reliability.

Definition 4: A network system (G, σ, S) has *bounded delivery time* if there exists $M < \infty$ such that each packet arrives in (G, σ, S) to its destination within M time steps from its injection time. We call M the *maximum delivery time*.

In the sequel we sometimes say that a *sequence σ allows reliability* (resp. allows bounded delivery time). By this we mean that there exists a schedule S such that the network system (G, σ, S) is reliable (resp. has bounded delivery time). We also sometimes abuse notation and for a protocol P write (G, σ, P), instead of $(G, \sigma, S^P(\sigma))$, where $S^P(\sigma)$ is the schedule generated by P for σ. We further say that a graph G is universally stable against all rate $r = 1$ adversaries, if for any σ given by a $(1, b)$ adversary, and any greedy protocol P, (G, σ, P) is stable.

3 Reliability

We now study reliability. We start with the following question: for which sequences of packets is there a schedule that actually delivers *all* packets, while using buffers of bounded size, i.e., while maintaining stability. We then study the question of the existence of a protocol that can generate such a schedule, whenever such a schedule exists.

3.1 Sequences that allow reliability

In this section we show that the set of sequences that allow reliability is equal to the set of sequences that allow stability. This set, in turn, is known to be equal to the set of sequences that can be injected by $(1, b)$ adversaries [13, 18]. More formally, we show the following theorem.

Theorem 5: A sequence σ has a reliable schedule if and only if it has a stable schedule.

One direction of the proof is trivial. To prove the second direction we start with the following.

Observation 6: Let (G, σ, S) be a stable system. The number of eternal packets in (G, σ, S) is at most B, where B is the maximum number of packets present at any given time in (G, σ, S).

Proof: Assume towards a contradiction that there are more than B eternal packets in (G, σ, S). Let t be the time step when $B + 1$ eternal packets have been injected. Since an eternal packet never arrives to its destination, all $B + 1$ eternal packets are in the network at time t. A contradiction. □

We now build for σ a schedule S' that maintains reliability, based on (any) schedule S such that (G, σ, S) is stable. Informally, S' will mimic S until all eternal packets of (G, σ, S) stop moving in (G, σ, S); then it will deliver all the eternal packets while blocking all other packets in the network; and then resume the behavior of S, delayed by a certain number of time steps. To define S' formally, let $\mathcal{E}$ be the set of eternal packets in (G, σ, S). Observe that for every eternal packet $p \in \mathcal{E}$ there is a time step $T_p < \infty$ such that after $T_p - 1$ p does not cross any edge in (G, σ, S). If p does not cross any edge at all in (G, σ, S) we define T_p such that $T_p - 1$ is the injection time of p. Let $T = \max_{p \in \mathcal{E}} T_p$, so that at T all eternal packets are already injected, and starting at T no eternal packet crosses an edge in (G, σ, S). Note that $T < \infty$. Let $D = |\mathcal{E}| \cdot d$.

The schedule S' is defined as follows. If (G, σ, S) is reliable itself, then S' is equal to S. In what follows we assume that (G, σ, S) is *not* reliable, and we build a schedule S' which is different from S.

1. For $0 \leq t < T$, S' is identical to S. That is, at any $0 \leq t < T$, S' sends across each edge $e \in E$ the same packet that crosses the edge e at time t in (G, σ, S) .

2. For $T \leq t < T + D$, S' schedules the packets in $\mathcal{E}$ according to an arbitrary greedy schedule. No other packets are sent across any edge in the network.

3. For $T + D \leq t$, for each edge e and time t, S' sends a packet p across e at t if and only if S sends the packet p across e at $t - D$.

We now give the following theorem.

Theorem 7: S' is a valid schedule for σ. (G, σ, S') is reliable.

The theorem follows from Lemmas 9 and 11 that we prove below. Informally, the validity and stability of S' follow from the fact that it mimics S (although sometimes at a finite delay D). The reliability follows from the fact that all the eternal packets of (G, σ, S) are delivered during the second phase of S' (using Observation 6), and that all non-eternal packets are delivered at a delay of at most D time steps relative to S. We note that schedule S' is not necessarily greedy.

We now give a formal proof of the above theorem. We first claim the following.

Lemma 8: For any $t \geq T + D - 1$, any packet $p \notin \mathcal{E}$ that is at node v at $t - D$ in (G, σ, S), is in the same node v at t in (G, σ, S'). In particular, for v being the destination node of p, if p leaves the system (G, σ, S) by time $t - D$, it leaves the system (G, σ, S') by time t.

Proof: We prove the lemma by induction on t. The base of the induction for $t = T + D - 1$ follows from the fact that before time T all packets move identically according to S and S'. For the induction step, let $t \geq T + D$. There are three cases, according to the location of p at $t - D$ in (G, σ, S):

1. p is injected in node v at time $t - D$.
   Since S does not send p by time $t - D$, S' does not send p by time t, and therefore p is still in v in (G, σ, S') at t.

2. p is in node v in (G, σ, S) at time $t - D - 1$.
   By the induction hypothesis, p is in v at $t - 1$ in (G, σ, S'). Clearly S does not send p at time $t - D$, and therefore S' does not send p at time t. We conclude that p is still in v in (G, σ, S') at t.

3. p is in a node $u \neq v$ in (G, σ, S) at time $t - D - 1$.
   By the induction hypothesis, p is in u at $t - 1$ in (G, σ, S'). So that p is in v at $t - D$ in (G, σ, S), S must send p across (v, u) at time t. Therefore, by the definition of S', p arrives at v in (G, σ, S') at t.

□

We now claim the following.

Lemma 9: S' is a valid schedule.

Proof: First, we note that $0 < T < \infty$, by Observation 6. Now observe that until time $T-1$, S and S' behave identically. For $T \leq t < T+D$, S' is valid by construction.

For $T + D \leq t$, we note that S' sends at most one packet across each edge at any time. By Lemma 8, we have that each packet that S' has to send across an edge e at some time t is indeed at the tail of e at the beginning of that time step. □

In the following we prove the desired properties of S'.

Lemma 10: (G, σ, S') is stable.

Proof: Since (G, σ, S) is stable, there exists some $B < \infty$ such that there are at most B packets in (G, σ, S) at any given time. By Lemma 8, each non-eternal packet leaves (G, σ, S') with a delay of at most D time steps compared to (G, σ, S). Since σ can inject at most $B + D|E|$ packets during D time steps, we conclude that the number of packets in (G, σ, S') at any time t may exceed the number of packets in (G, σ, S) at time $t - D$ by at most $B + D|E|$. Since (G, σ, S) is stable the lemma follows. □

Lemma 11: (G, σ, S') is reliable.

Proof: First note that (G, σ, S') is stable by Lemma 10. We now claim that there are no eternal packets in (G, σ, S').

Consider an arbitrary packet p in σ. If $p \in \mathcal{E}$, p is injected in (G, σ, S') by time $T-1$. During the time interval $[T, T+D)$, only packets in $\mathcal{E}$ are sent across edges by a greedy schedule. Since $D = |\mathcal{E}|d$, all these packets arrive to their destinations by $T + D$.

If $p \notin \mathcal{E}$, there are two cases:

1. p leaves the system (G, σ, S) at some time $t < T$. Since S and S' behave identically for $t < T$, p is not eternal in (G, σ, S').

2. p leaves the system (G, σ, S) by time t, for $t \geq T$. By Lemma 8, p leaves the system (G, σ, S') by time $t + D$.

□

3.2 Achieving reliability by protocols

We now consider the question of the existence of an online protocol that achieves reliability whenever the sequence of packets allows reliability. As we show in the previous section, the set of sequences that allow reliability is equal to the set of sequences that can be injected by $(1, b)$ adversaries. We now show that on certain networks (namely, cycles of length greater than 2) there is no online protocol (even centralized) that achieves reliability against all $(1, b)$ adversaries. On the other hand, we show that for any network that does not contain simple cycles of length greater than 2, there are protocols (online and local control) that achieve reliability whenever the sequence of packets allows reliability. Thus we characterize the network topologies on which reliability is achievable by an online protocol, and give a protocol that achieves reliability on these networks. We start with the negative result.

Theorem 12: For any unidirectional cycle of length $N > 2$, there is no online protocol (even centralized) that achieves reliability against all $(1, b)$ adversaries.

To prove the theorem, given a unidirectional cycle of length $N > 2$, C_N, and an arbitrary online protocol P, we build the following adversary A of rate $r \leq 1$. Denote the nodes of the cycle v_0 to v_{N-1}, and denote the edge emanating from v_i by e_i. In each time step the adversary injects one packet of length 2 edges. Therefore A is an (r, b) adversary for $r \leq 1$ and $b = 0$. A packet that is injected at node v_i (and has path (v_i, v_{i+1}, v_{i+2})) is called *a type-i packet*. The adversary proceeds in phases, starting with phase number 1 that starts at time step $t = 1$. In phase 1 the adversary injects one packet of type 1, the phase immediately ends and phase 2 starts at time step $t = 2$. In phase $k > 1$, the adversary injects in each time step one packet of type $(k \bmod N)$. The phase ends at the end of the first time step in which P does not hold in its buffers packets of any type other than type $(k \bmod N)$. (Note that this rule applies also for phase number 1). Phase $k + 1$ starts at the time step immediately after the end of phase k. Note that the actions of the adversary, in particular the length of each phase, are determined as a function of the actions of the online protocol P.

Let T_k be the first time step of phase k. If the adversary does not reach phase k we set $T_k = \infty$.

Lemma 13: If P is reliable then for any $k \geq 1$, $T_k < \infty$.

Proof: The proof is by induction on k. For $k = 1$, by definition, the adversary starts in phase 1, and therefore $T_1 = 1$. For $k = 2$, by definition, phase 1 ends at time 1, and therefore $T_2 = 2$.

For $k > 2$, by the induction hypothesis we have that $T_{k-1} < \infty$. Consider the set of packets $\mathcal{F}_{k-1}$ that are present in the network at time $T_{k-1} - 1$, i.e. at the end of the last time step of phase $k - 2$. Since P is reliable, there are no eternal packets and therefore each of the packets in $\mathcal{F}_{k-1}$ eventually arrives to its destination. Let

T be the *last* time step when a packet from $\mathcal{F}_{k-1}$ arrives at its destination. Since $\mathcal{F}_{k-1}$ is finite $T < \infty$.

If phase $k-1$ ends before time T, we are done since $T_k \leq T < \infty$. Otherwise, all the packets present in the network at time T (if any) were injected during phase $k-1$, and therefore they are all of type $((k-1) \bmod N)$. Therefore phase $k-1$ ends at time T and $T_k = T + 1 < \infty$. □

Lemma 14: Phase k lasts at least k time steps. If the phase ends, then at the end of phase k there are at least k packets of type $(k \bmod N)$ present at their injection point.

Proof: The proof is by induction on the phase number k. The base of the induction is for $k = 1$. By definition, one type-1 packet is injected in node v_1, and the adversary immediately switches to phase 2. Therefore this packet is still in node v_1 at the end of phase 1. This phase lasts exactly one time step.

We now prove the claim for $k > 1$. Assume w.l.o.g. that $((k-1) \bmod N) = 0$, so by the induction hypothesis, at the end of phase $k-1$ there are at least $k-1$ type-0 packets in node v_0. To ease the presentation of the proof, we call these packets *red packets*.

We note that when phase k ends, at time $T_{k+1} - 1$, there are no type-0 packets in the network. This means that at least $k-1$ red packets which were present in the network at the end of time step $T_k - 1$, must traverse edge e_1 during time interval $[T_k, T_{k+1} - 1]$. Further, we note that at the end of time step $T_k - 1$, all red packets are at node v_0. Therefore, no red packet traverses edge e_1 at time step T_k, since red packets cannot reach this edge at this time. We conclude that $(T_{k+1} - 1) - T_k \geq k - 1$, i.e.,

$$T_{k+1} - T_k \geq k \; . \tag{1}$$

Therefore, the length of phase k is at least k.

We now argue that at the end of phase k there are at least k type-1 packets at v_1. Observe that during phase k, the adversary injects $T_{k+1} - T_k$ type-1 packets at node v_1. We note that at the beginning of the first time step of the phase, T_k, there are no type-1 packets in the network, therefore no type-1 packet traverses e_1 at T_k [2]. Therefore, all type-1 packets that leave node v_1 by the end of phase k, must traverse edge e_1 at a time step which is on the one hand in time interval $[T_k + 1, T_{k+1} - 1]$, and on the other hand a time step not used by a red packet to cross e_1. As noted above no red packet can traverse e_1 at T_k and when phase k

[2] Recall that packets traverse edges in the first sub-step, while new packets are injected in the second sub-step of a time step.

ends (at time $T_{k+1} - 1$) there are no red packets in the network (since the phase can end only when there are no type-0 packets in the network). This means that at least $k - 1$ red packets must traverse edge e_1 during time interval $[T_k + 1, T_{k+1} - 1]$. Thus, the number of type-1 packets that do not leave their injection node by time $T_{k+1} - 1$ is at least

$$N_k \geq (T_{k+1} - T_k) - \max\{0, (T_{k+1} - 1 - (T_k + 1) + 1) - (k-1)\} \, .$$

Using (1) we have,

$$N_k \geq (T_{k+1} - T_k) - ((T_{k+1} - T_k - 1) - (k-1)) = k \, .$$

□

We can now conclude the proof of Theorem 12.

Proof of Theorem 12: Assume towards a contradiction that P is reliable. Then it is stable and there exists $B < \infty$ such that there are at most B packets in the network at any time. By Lemma 13, $T_{B+2} < \infty$. In particular, the number of packets in the network at time T_{B+2} is at most B. But, by Lemma 14, at the end of phase $B + 1$ there are at least $B + 1$ packets of type $((B + 1) \bmod N)$ stored at their injection point. A contradiction. □

We note that the adversary used in the proof of Theorem 12 is an (r, b) adversary with $r = 1$, $b = 0$. Therefore, for any integer w and during any interval of w time steps, the adversary injects at most w packets that require any given edge. Thus, our negative results hold also against (w, r) adversaries as considered sometimes in the literature (e.g. [10]).

We now proceed to show that on any network that does not contain any simple cycle of length greater than 2, there is a local control protocol that achieves reliability whenever traffic allows reliability (i.e., for any sequence given by an (r, b) adversary for $r \leq 1$).

Definition 15: We call a directed graph a *directed almost-acyclic graph* if it has no directed simple cycles of length $N > 2$.

In particular, any directed acyclic graph (DAG) is a directed almost-acyclic graph (hence the name). We show that all directed almost-acyclic graphs are universally stable against all rate $r = 1$ adversaries. We use in our proof the fact that all packet paths are simple (and therefore no packet path uses both edges of a cycle). Our proof is an extension of the proof of [10] that any directed acyclic graph is universally stable against all rate $r = 1$ adversaries.

Theorem 16: Let G be an arbitrary directed almost-acyclic graph, P an arbitrary greedy protocol, and σ a sequence of packets given by an (r, b) adversary for $r \leq 1$. Then (G, σ, P) is stable.

Proof: For the purpose of the proof we define the dual graph G' of G.

Definition 17: Given a directed almost-acyclic graph $G = (V, E)$, its dual directed graph $G' = (V', E')$ is defined as follows: $V' = E$, and for each $e_1, e_2 \in E$ there is an edge $(e_1, e_2) \in E'$ if and only if there is a simple path in G which includes the sequence (e_1, e_2) (i.e., if and only if (e_1, e_2) is a simple path in G). If there is an edge (e_1, e_2) in G' we say that e_1 is *a parent* of e_2.

Observation 18: The dual G' of a directed almost-acyclic graph G is a DAG.

Proof: Let $G = (V, E)$ be a directed almost-acyclic graph, and let $G' = (V', E')$ be its dual graph.

First note that by construction, G' does not contain any self-loops.

Now, assume towards a contradiction that there is a cycle in G',
$\pi' = (e_1, e_2, \ldots, e_i, e_{i+1}, \ldots, e_N, e_1)$, for $N > 1$, $e_i \in V'$.

Recall that a node in V' corresponds to an edge in E. We can therefore denote an edge on π' as $(e_i = (v_i, u_i), e_{i+1} = (v_{i+1}, u_{i+1}))$, for $1 \leq i \leq N-1$, $v_k, u_k \in V$ (and similarly $(e_N = (v_N, u_N), e_1 = (v_1, u_1))$). By construction of G' we have that $u_i = v_{i+1}$ for any $1 \leq i \leq N-1$ (and similarly $u_N = v_1$). Therefore the existence of the cycle π' in G' implies the existence of a cycle π in G where $\pi = (v_1, v_2, \ldots, v_i, v_{i+1}, \ldots, v_{N-1}, v_N, v_1)$.

Now consider any two consecutive nodes along π, and denote them v and v'. We claim that v and v' must be distinct nodes in V. This is because (v, v') is a node on π', and by construction of G' a self-loop in G is an isolated node in G'.

Now consider any 3 consecutive nodes along π, and denote them v,v', and v''. We claim that v and v'' must be distinct nodes in V. To see this observe that (v, v') and (v', v'') are two consecutive nodes on π'. But, by construction of G' this means that the path (denoted by its edges) $(v, v'), (v', v'')$ is a *simple* path in G. Therefore, v and v'' cannot be the same node.

It follows that cycle π in G does not contain simple cycles of length 1 or 2 as a sub-path. Since π is a cycle, it must contain a simple cycle of length at least 3, a contradiction to the assumption that G is a directed almost-acyclic graph. □

We now define a function on the edges of G, making use of the graph G' in the definition.

Definition 19: Given an (r, b) adversary, a directed almost-acyclic graph $G = (V, E)$, and its dual $G' = (E', V')$, we define a function $\psi(e)$, for any $e \in E$. For $e \in E$ which is a source (i.e., has in-degree 0) in the DAG G' we define $\psi(e) = b + 1$. For $e \in E$ which is not a source in G' we define $\psi(e) = b + 1 + \sum_{(f,e) \in E'} \psi(f)$.

Note that since the dual graph is a DAG, ψ is well defined.

We now show that any directed almost-acyclic graph is universally stable against all rate $r = 1$ adversaries.

For an edge of G, $e \in E$, let $Q_t(e)$ denote the number of packets in the queue at the tail of edge e at the end of time step t, and let $R_t(e)$ denote the number of packets present in the network at the end of time step t and have e on the remainder of their path. We claim that, for all $t \geq 0$ and all $e \in E$,

$$R_t(e) \leq \psi(e) \ . \tag{2}$$

The theorem will then follow, since $\sum_e \psi(e)$ gives an upper bound on the number of packets in the network.

For an arbitrary $e \in E = V'$ and $t \geq 0$, let $t' \leq t$ be the last time step before t such that $Q_{t'}(e) = 0$ (note that such time exists since the system starts with empty buffers at time $t = 0$). Since $Q_{t'}(e) = 0$, it follows that $R_{t'}(e) \leq \sum_{(f,e)\in E'} R_{t'}(f)$. Since σ is given by an (r, b) adversary for $r \leq 1$, the number of packets injected in $(t', t]$ and require e is at most $t - t' + b$. On the other hand, P is greedy, and since the queue of e is not empty at the end of each time step in the interval (t', t) at least $t - t' - 1$ packets cross e during $(t' + 1, t]$. Thus at the end of time step t,

$$\begin{aligned} R_t(e) &\leq R_{t'}(e) + t - t' + b - (t - t' - 1) \\ &\leq b + 1 + \sum_{(f,e)\in E'} R_{t'}(f) \ . \end{aligned} \tag{3}$$

Denote by l_e the *level* of $e \in V'$, that we define to be the length of the longest path leading to e in G'. The proof of Inequality (2) now proceeds by induction on l_e. The base of the induction for $l_e = 0$ is for a source $e \in V'$. Since it has no parents, we have $\psi(e) = b + 1$ and by (3) $R_t(e) \leq b + 1 = \psi(e)$.

Consider $e \in V'$ that has parents. Clearly the level of e is bigger than that of each of its parents. Thus, by the induction hypothesis, $R_{t'}(f) \leq \psi(f)$ for each parent f of e. By (3) we have

$$R_t(e) \leq b + 1 + \sum_{(f,e)\in E'} \psi(f) = \psi(e) \ .$$

□

Now, we can use First-In-First-Out (FIFO) [3] to show that on directed almost-acyclic graphs there is a protocol that achieves reliability whenever traffic allows

[3]FIFO is an online, local control protocol that serves the packets in each queue according to their arrival time to the queue.

reliability. The relevant property of FIFO is the fact that FIFO does not starve any packet. More formally, for every edge e, every packet p that arrives to the queue of e, eventually crosses e. Clearly, when a non-starving protocol is used, there are no eternal packets in the system. Since FIFO is non-starving and greedy, together with Theorem 16, we have that on directed almost-acyclic graphs FIFO achieves reliability whenever the traffic allows reliability. Thus we have the following.

Theorem 20: For any directed almost-acyclic graph G, and any sequence of packets σ given by an (r, b) adversary for $r \leq 1$, $(G, \sigma, FIFO)$ is reliable.

4 Bounded delivery time

We now turn to consider the question of having bounded delivery time for all packets. That is, the requirement is that there is a finite $M < \infty$ such that each packet is delivered within M time steps from its injection time (rather than just being eventually delivered, as required by the reliability property). We first show that there are sequences of packets given by (r, b) adversaries, $r = 1$, $b = 0$, such that there is no schedule with bounded delivery time. We then turn to the question of achieving bounded delivery time by protocols, whenever traffic allows bounded delivery time.

4.1 Sequences that allow bounded delivery time

We start by considering the following question. For which sequences σ is there a schedule S such that (G, σ, S) has bounded delivery time. The following observation in [10] implies that for any graph G, the set of such sequences is a *subset* of the set of sequences that allow stability.

Observation 21: If no packet is delayed more than M time steps, then the maximum queue size is at most M.

We now show that for any unidirectional cycle of length $N > 2$, C_N, the set of sequences σ for which there is a schedule S such that (C_N, σ, S) has bounded delivery time is a *proper subset* of the set of sequences that allow stability. In particular, on such graphs, there are sequences that allow stability, but there is no finite M such that all packets can be delivered within M time steps.

Theorem 22: For any unidirectional cycle of length $N > 2$, C_N, there are sequences σ that can be given by an (r, b) adversary, $r = 1$, $b = 0$, such that there is no schedule S for which (C_N, σ, S) has bounded delivery time.

We observe that the sequences guaranteed by Theorem 22 are such that, for any integer w and for any interval of w time steps, there are at most w packets injected in that time interval, that require any given edge (because $r = 1$ and $b = 0$). Thus, these sequence can be given by (w, r) adversaries as considered sometimes in the literature (e.g. [10]).

To prove Theorem 22, given a unidirectional cycle C_N, $N > 2$, we build the following sequence σ. (A sequence built on similar ideas was used in [10] to show the instability of FIFO on the cycle with $r = 1$).

Denote the nodes of the cycle v_0 to v_{N-1}, and denote the edge emanating from v_i by e_i. In any time step one packet of length 2 edges is injected. A packet injected at node v_i is called ***a type-i packet***. The sequence is given in phases, starting with phase number 1 that starts at time 1. The duration of each phase $k \geq 1$ is k time steps. In phase k, the packets that are injected are of type $(k \bmod N)$ (recall that in each time step one packet is injected). [4]

We now show that there is no schedule that maintains bounded delivery time for this sequence. We start with the following lemma.

Lemma 23: Let S be a schedule such that (C_N, σ, S) has bounded delivery time, and let $0 < M < \infty$ be the maximum delivery time. Then for each $j \geq 0$, at the end of phase $M + j$ there are at least j packets of type $((M + j) \bmod N)$ stored at their injection point.

Proof: The proof is by induction on j. The base of the induction is trivial for $j = 0$.

We now prove the claim for $j > 0$. Let T_i be the first time step of phase i, for $i \geq 1$. Assume w.l.o.g. that $((M + j - 1) \bmod N) = 0$, so by the induction hypothesis, at the end of phase $M + j - 1$ (at the end of time step $T_{M+j} - 1$) there are at least $j - 1$ type-0 packets in node v_0.

We note that the duration of phase $M + j$ is $M + j > M$, therefore all packets present in the network when this phase begins must leave the network during this phase. This means that at least j type-0 packets which are stored at v_0 at the end of time step $T_{M+j} - 1$, must traverse edge e_1 by time T_{M+j+1}. Note that no such packet can traverse e_1 at time step T_{M+j} since they cannot reach this edge by this time. Further note that no type-1 packet injected during phase $M + j$ can traverse e_1 at time step T_{M+j}, since the first such packet is injected only at time step T_{M+j} [5]. Now, to count the number of type-1 packets stored at v_1 at the end of phase

[4] We note that while this sequence has similarities to the sequence used in the proof of Theorem 12, here we use a single fixed sequence, which is not a function of the action of any online protocol.

[5] Recall that packets traverse edges in the first sub-step, while new packets are injected in the second sub-step of a time step.

$M + j$, we observe that $M + j$ such packets are injected during this phase. For such a packet not to be at v_1 at the end of the phase, it must traverse e_1. But it can traverse e_1 only during time interval $[T_{M+j} + 1, T_{M+j+1} - 1]$, and at time steps where no type-0 packet traverses this edge. We have that the number of type-1 packets that *do not* leave their injection node by time T_{M+j+1} is at least

$$N_{M+j} \geq T_{M+j+1} - T_{M+j} - \max\{0, T_{M+j+1} - 1 - (T_{M+j} + 1) + 1 - j\} .$$

Since the duration of phase $M+j$ is $M+j$, we have $T_{M+j+1} - T_{M+j} = M+j > j$, and therefore,

$$N_{M+j} \geq T_{M+j+1} - T_{M+j} - (T_{M+j+1} - T_{M+j} - 1 - j) = j + 1 .$$

□

We now conclude the proof of the main theorem of this section.

Proof of Theorem 22: Let σ be as defined above. Assume towards a contradiction that there exists a schedule S such that (C_N, σ, S) has bounded delivery time, and let $M < \infty$ be the maximum delivery time. Let $j = M + 1$. By Lemma 23, the number of type-$((M + j) \bmod N)$ packets at their injection point at the end of phase $M + j = 2M + 1$ is at least $j = M + 1$. At this time, again by Lemma 23, all of these packets are in the same queue at edge $e_{(M+j) \bmod N}$. A contradiction to Observation 21. □

4.2 Achieving bounded delivery time by protocols

Since we saw that *not all* sequences that allow stability (and reliability), also allow bounded delivery time, the natural question that arises is whether we can devise a protocol that guarantees bounded delivery time whenever traffic allows bounded delivery time. As noted in Observation 21 bounded delivery time implies stability.

To consider the sequences of packets for which it could at all be possible to achieve bounded delivery time, we define the *bounded delivery time adversary* as follows.

Definition 24: A bounded delivery time adversary with parameter $\mathcal{D}$, called the maximum delay, is an adversary that can inject any sequence of packets as long as it can itself deliver each packet within $\mathcal{D}$ time steps after the packet's injection time.

Observation 25: A bounded delivery time adversary with maximum delay $\mathcal{D}$ is also an (r, b) adversary with $r = 1$ and $b = \mathcal{D}$.

Proof: We claim that any sequence of packets injected by a bounded delivery time adversary with parameter $\mathcal{D}$ is such that for any edge e, the number of packets that use e that are injected in any time interval $(t_1, t_2]$ is at most $(t_2-t_1)+\mathcal{D}$. To see that observe that if for some edge e and some interval $(t_1, t_2]$ the adversary injects more than $(t_2 - t_1) + \mathcal{D}$ packets that use e, then at the end of time step t_2 there are more than $\mathcal{D}$ packets requiring e in the network. At least one of these packets will not be delivered within the next $\mathcal{D}$ time steps, and therefore such sequence cannot be injected by a bounded delivery time adversary with parameter $\mathcal{D}$. □

We note that if one employs FIFO, then when stability is achieved, bounded delivery time is guaranteed. Therefore, using Observation 25 and Theorem 16, we have that FIFO on a directed almost-acyclic graph has bounded delivery time against any bounded delivery time adversary. Thus we have a class of network topologies for which, using FIFO, bounded delivery time is achieved whenever possible.

For general topologies we define a class of adversaries that is somewhat weaker than the class of all bounded delivery time adversaries, and design a protocol that achieves bounded delivery time against any adversary in this class.

Let $A_\pi(t_1, t_2)$ be the number of packets injected during time interval $(t_1, t_2]$ and use path π. Let $A_\pi(t) = A_\pi(0, t)$.

Definition 26: An adversary A is a *frequent adversary*, with parameters $\tau > 0$, called the period, and $z \geq 0$, called the peak, if for each path π, one of the following holds.

- Adversary A injects at least one packet that uses π in each τ time steps. That is, for any $t_2 - t_1 \geq \tau$, $A_\pi(t_1, t_2) > 0$.
- Adversary A injects at most z packets that use π, overall. That is, for all t, $A_\pi(t) \leq z$.

We call the paths of the first kind *frequent paths*, and the paths of the second kind *rare paths*.

We note that this class of adversaries is a generalization of the class of "path-wise constant arrival rates" adversaries as defined by Gamarnik [14].

Definition 27: We say that the network has *path-wise constant arrival rates* if for each path π there exists $r_\pi \geq 0$, such that, for some b_π and for all $t \geq 0$

$$A_\pi(t) \leq r_\pi t + b_\pi \ , \tag{4}$$

and in addition, for some $b'_\pi \geq 0$ and for all $t \geq 0$

$$A_\pi(t) \geq r_\pi t - b'_\pi \ , \tag{5}$$

We consider here path-wise constant arrival rates adversaries that are also (r, b) adversaries for $r \leq 1$ (contrary to [14] which only considers adversaries of rate $r < 1$). By simple calculation it is easy to see that a path-wise constant arrival rates adversary is also a frequent adversary, by setting the appropriate parameters τ and z for the the frequent adversary. For a path π, if $r_\pi = 0$ then the path is a rare path, and if $r_\pi > 0$ then the path is a frequent path. We also note that there are sequences of packets that can be injected by a frequent adversary but not by a path-wise constant arrival rates adversary, for example, a sequence where a packet is injected every second time step, and in addition a packet is injected every time step n^2, for $n \geq 1$.

We now present an online, local control protocol ERF (*Estimated-Rare First*) that has bounded delivery time on any topology and for any sequence given by a bounded delivery time adversary that is also a frequent adversary.

We first give a general description of the protocol and an intuitive idea why it works. First observe that on frequent paths, the protocol FTG will guarantee bounded delivery time. Therefore, if we knew for each path π whether it is frequent or rare, we could give priority to the packets of the rare paths and apply FTG to the packets of the frequent paths. However, we do not have a way to distinguish a priori between frequent and rare paths. The protocol therefore estimates at any point in time, and for each path, whether the path is rare or frequent, based on the number of packets injected on that path until that time. This is done at the source node of each path, and as we prove in the sequel, this can be done in such a way that the estimation stabilizes to the correct value in finite time. If the protocol was centralized, this would be enough. However, since we build a distributed protocol, we use control packets in order to convey changes in the estimation to the nodes along the path. In the sequel, we show how this can be done in such a way that the control packets do not overload the network, and do not interfere with the bounded delivery time property.

Protocol ERF

The protocol labels all the packets with a *priority* label. Each packet is labeled as either *high priority*, or *low priority*. At each time step, and for each edge e, ERF selects a packet to send across e as follows:

- If there is any high priority packet in the queue of e, ERF sends an arbitrary high priority packet.

- Otherwise, ERF selects the packet to send from the queue of e according to the Furthest-To-Go (FTG) rule (i.e., the packet that has the longest remaining path to its destination is selected). Ties are broken according to the FIFO rule (i.e., according to the order of arrival to the queue).

We now describe how the priority labels are managed. In order to manage the labels, the protocol creates at times *control packets* to which it itself assigns a path and introduces them to a queue at their creation node. These packets are then forwarded by the protocol together with the data packets [6].

At the tail of each edge e, for each simple path π such that e is the first edge on π, ERF keeps the current *estimation* for whether π is a frequent path or a rare path. This estimation is in fact a function of $A_\pi(t)$. At any time t, ERF estimates that π is a rare path if $A_\pi(t) < \lfloor t^{1/2} \rfloor$, [7] and that it is a frequent path, otherwise. When a packet is injected, it is labeled as high priority if the path for this packet is estimated at the injection time to be rare; the packet is labeled low priority, otherwise.

When the estimation for π changes from a frequent path to a rare path, ERF:

1. Labels as high priority all packets (if any) in the queue of e that use π.

2. Creates a new control packet with the path π if, since the last creation of a control packet on π, the adversary injected at least one new packet that uses π. All control packets are labeled high priority.

When a control packet that uses path π arrives at some node, ERF labels high priority all packets present in this node that use the same path π.

We now proceed to prove the properties of ERF. The following is immediate from the definition of ERF.

Observation 28: For any time t and any path π, the number of control packets created by ERF on path π by time t is at most the number of packets injected by time t on π.

The following claims shows that ERF eventually correctly identifies each frequent path as such, and each rare path as such.

Claim 29: Let A be a frequent adversary, and let τ be its period. Then for any $t \geq \tau^2$ and for any frequent path π, at time t ERF correctly estimates π to be a frequent path. Further, no control packet that uses π is created by ERF at time t.

Proof: Consider a frequent path π. A injects at least one packet that uses the path π in each τ time steps. Therefore, the total number of packets that use the path π injected by the adversary by t is at least $A_\pi(t) \geq \lfloor t/\tau \rfloor$. Thus for any $t \geq \tau^2$, $A_\pi(t) \geq \lfloor t^{1/2} \rfloor$ and ERF estimates π to be a frequent path. □

[6] In this sense, the protocol is not greedy, since sometimes it sends a control packet and not a packet injected by the adversary.

[7] Any other function $f(t) = o(t)$ could also be used instead.

Claim 30: Let A be a frequent adversary, and let z be its peak. Then for any $t > (z+1)^2$, and for any rare path π, at time t ERF correctly estimates π to be a rare path.

Proof: Since π is a rare path, there are at most z packets ever injected on π. Therefore for $t > (z+1)^2$ we have $A_\pi(t) \leq z < \lfloor t^{1/2} \rfloor$. Thus, for $t > (z+1)^2$, ERF correctly estimates π to be a rare path. □

In what follows we assume a given arbitrary directed graph $G = (V, E)$, and a given arbitrary packet sequence σ injected by a bounded delivery time adversary with maximum delay $\mathcal{D}$, which is also a frequent adversary with period τ and peak z. We further assume that the network is controlled by ERF. We start by proving the stability of the system. This property will be later used to prove that the system has bounded delivery time. In what follows we do not attempt to prove tight bounds on the number of packets in the network or on the delivery times of the packets. Rather, we only prove the existence of finite bounds for these quantities, which yields the required properties of stability and bounded delivery time.

Lemma 31: The overall number of packets that are ever labeled high priority is at most $\mathcal{H} = O(|E|^d(\tau^2 + \mathcal{D} + z))$.

Proof: First consider a frequent path π. By Claim 29, at any $t \geq \tau^2$ ERF correctly estimates π to be a frequent path. Therefore all packets that use π and are injected at $t \geq \tau^2$ are labeled low priority at injection. Furthermore, by Claim 29 no control packet that uses π is created at any $t \geq \tau^2$. Since control packets have priority over low priority packets, for a control packet q on π to arrive to a node v where there is a low priority packet p on π, p must be injected before q is created. Therefore no packet of π injected after τ^2 can ever be labeled high priority. We conclude that all packets that use path π and are at some time labeled high priority must be injected by time τ^2 (or created by this time if they are control packets). Using Observation 25 the adversary can inject at most $|E|(\tau^2 + \mathcal{D})$ packets by τ^2, and as already noted, ERF may create at most as many control packets.

Now consider a rare path π. The adversary injects at most z packets that use π, and ERF can create at most as many control packets that use π.

Summing for both types of paths, we have that the total number of packets ever labeled high priority is at most $\mathcal{H} \leq 2|E|(\tau^2 + \mathcal{D}) + 2N_0 z$, where N_0 is the number of rare paths in σ. Note that $N_0 < |E|^d$. □

Lemma 32: For any given time t, the number of packets labeled low priority present in the network at t is at most $\mathcal{L} = O(|E|^{2d}(\tau^2 + \mathcal{D} + z))$.

Proof: We base our proof technique on the proof in [13] for the stability of Furthest-To-Go, with several adaptations. We begin by introducing the following notations, which deal with the low priority packets residing in the buffer of an edge e at a certain time t.

Let $X_e(t)$ denote the set of low priority packets which are stored at the buffer of edge e at the end of time step t. Let $L_e(p)$ denote the number of edges which a packet p, stored at the buffer of edge e, must still traverse before reaching its destination. Let $X_e^i(t)$ be the following set:

$$X_e^i(t) = \{p | p \in X_e(t) \text{ and } L_e(p) \geq i\} \ .$$

By the definition of ERF, whenever there are no high priority packets in the buffer of e at the end of time step t, and $X_e(t) \neq \emptyset$, then at time $t+1$ the packet p, $p \in X_e(t)$, that is forwarded on e is such that

$$\forall p' \in X_e(t), L_e(p) \geq L_e(p') \ .$$

Now define the following sequence of constants:

$$k_i = \begin{cases} 0 & \text{if } i > d \\ |E|(k_{i+1} + \mathcal{D} + \mathcal{H} + 1) & \text{if } 1 \leq i \leq d \end{cases}$$

Note, for later use, that for $1 \leq i \leq d$, $k_i = (\mathcal{D} + \mathcal{H} + 1)|E|\frac{|E|^{d+1-i}-1}{|E|-1}$.

The lemma will follow from the following claim.

Claim 33: For all t, $k_i \geq \sum_{e \in E} |X_e^i(t)|$. That is, the number of low priority packets in the network at any time t which are at least i hops away from their destinations is at most k_i.

Proof: We prove the claim by decreasing induction on i. Since there are no simple paths of length longer than d in G, the claim trivially holds for all $i > d$ at all times. Assume that the claim holds for any $j > i$, and consider a certain edge $e \in E$. Let $t \geq 0$ such that $X_e^i(t) \neq \emptyset$, and denote by t' the latest time prior to t which satisfies $X_e^i(t') = \emptyset$ (note that such time exists since the system starts with empty buffers at time $t = 0$). We note that ERF never changes the packet label from high priority to low priority. Therefore the packets of $X_e^i(t)$ can be partitioned into the following two disjoint sets.

1. Low priority packets which are injected into the network no earlier than time $t'+1$. Using Observation 25, we can bound the number of these packets by $(t-t') + \mathcal{D}$.

2. Low priority packets which were already active in the system at time t'. Since t' is chosen such that $X_e^i(t') = \emptyset$, we conclude that at the end of time step t' these packets are all stored in buffers other than that of e. Upon arrival at e's buffer (at some time between t' and t), these packets still require at least i hops. We thus conclude that at the end of time step t' they are still at least $i+1$ hops away from their destination. By the induction hypothesis, we can bound the number of these packets by k_{i+1}.

Also, by the choice of t', we know that at the end of each of the time steps in the interval (t', t) the buffer of e contains at least one low priority packet requiring at least i hops. By the definition of ERF, either a high priority packet, or a low priority packet requiring at least i hops, crosses e at each of the time steps in the interval $(t'+1, t]$.

Since by Lemma 31 at most $\mathcal{H}$ packets are ever labeled high priority, we have that

$$\begin{aligned} |X_e^i(t)| &\leq (t-t') + \mathcal{D} + k_{i+1} - (t - t' - 1 - \mathcal{H}) \\ &\leq k_{i+1} + \mathcal{D} + 1 + \mathcal{H} , \end{aligned}$$

and therefore,

$$\sum_{e \in E} |X_e^i(t)| \leq |E|(k_{i+1} + \mathcal{D} + 1 + \mathcal{H}) = k_i .$$

□

The number of low priority packets in the network at a given time t can be expressed as $\sum_{e \in E} |X_e^1(t)|$. Therefore by Claim 33 we have that the number of low priority packets present in the network at any given time is at most

$$\mathcal{L} \leq k_1 = O(|E|^{2d}(\tau^2 + \mathcal{D} + z)) .$$

□

Using Lemmas 32 and 31, we can now bound the number of packets (of all types) present in the network at any given time to be at most $\mathcal{L} + \mathcal{H} = O(|E|^{2d}(\tau^2 + \mathcal{D} + z))$. Thus (G, σ, ERF) is stable. We now proceed to consider the delivery times of the packets.

Claim 34: Let p be a packet labeled at time t as high priority. Packet p leaves the network by time $t + \mathcal{H}d$.

Proof: Recall that ERF always sends a high priority packet if present. Therefore beginning at time step t, and until p is absorbed in its destination node, at least one high priority packet will cross some edge in each time step. By Lemma 31 there is a finite number $\mathcal{H}$ of high priority packets. Each of them has to cross at most d edges to arrive at its destination. Therefore p arrives to its destination in at most $\mathcal{H}d$ time steps from t. □

Since control packets are labeled high priority at creation, we have the following.

Corollary 35: Every control packet leaves the network within $\mathcal{H}d$ time steps after its creation.

We conclude this section with the following theorem.

Theorem 36: (G, σ, ERF) has bounded delivery time $M = O(z^2 + |E|^{2d}\tau(\tau^2 + \mathcal{D} + z))$.

Proof: First, consider a rare path π. Let p be a packet using path π that is injected at time t_p. There are two cases for t_p.

1. If at t_p the estimation for π was rare, p is labeled high priority and by Claim 34, p leaves the network by $t_p + \mathcal{H}d$.

2. If at time t_p the estimation for π was wrong (i.e., π was estimated to be a frequent path), it follows from Claim 30 that $t_p \leq (z+1)^2$. Using Claim 30 again, we have that the estimation for π becomes accurate at some time t_q such that $t_p \leq t_q \leq (z+1)^2$. Thus, if p is still at its injection point at the end of time step t_q, p is labeled high priority and it arrives at its destination by $t_q + \mathcal{H}d$. Otherwise (if p leaves its injection point by time t_q) at time t_q ERF creates a control packet q for π. By Corollary 35, q arrives to its destination by time $t_q + \mathcal{H}d$. Therefore, either p leaves the network by $t_q + \mathcal{H}d$, or it is in a node along its path when q reaches that node. Packet p will then be labeled high priority, and will leave the network within another $\mathcal{H}d$ time steps. In all, p arrives at its destination by $t_q + 2\mathcal{H}d$.

We conclude that the delivery time for packet p that uses a rare path is at most $M_0 \leq (z+1)^2 + 2\mathcal{H}d$.

Now consider a frequent path π. No packet of π changes its label to high priority after time $\tau^2 + \mathcal{H}d$. This is because by Claim 29 after time τ^2 no such packet is labeled high priority at its injection point, and by Corollary 35 after time $\tau^2 + \mathcal{H}d$ there are no control packets on π in the network. Therefore, by Claim 34, all high priority packets that use the path π leave the network by time $T \leq \tau^2 + 2\mathcal{H}d$.

Consider the system at the end of time step $t \geq T$. All packets of a frequent path π present in the network at t must be low priority packets. They are served according to FTG with ties broken by FIFO. Since all of them use the same path, they move in the network in FIFO order among themselves. By Lemma 32, the total number of low priority packets in the network at any time is at most $\mathcal{L}$. After time T, at least one low priority packet that uses the path π is injected every τ time steps, and no low priority packets become high priority. Therefore, each such packet must leave the network within $\tau\mathcal{L}$ time steps, otherwise there would be more than $\mathcal{L}$ low priority packets in the network. A packet that uses a frequent path is therefore delivered within $M_1 \leq T + \tau\mathcal{L}$ time steps.

We have that the maximum delivery time in the system is

$$\begin{aligned} M &= \max(M_0, M_1) \\ &\leq \max((z+1)^2 + 2\mathcal{H}d, \tau^2 + 2\mathcal{H}d + \tau\mathcal{L}) \\ &= O(z^2 + |E|^{2d}\tau(\tau^2 + \mathcal{D} + z)) \, . \end{aligned}$$

□

5 Conclusions and open problems

In this paper we study the problems of having a finite upper bound on the delivery times of all packets, and of even only eventually delivering all packets while maintaining stability. We study both the mere existence of schedules (generated by the adversary or by an offline scheduler) that achieve these requirements, and the existence of online protocols that achieve these requirements whenever such a schedule exists.

Among other things, we show that there exist sequences of packets injected by $(1, b)$ adversaries, such that there is no finite M with a schedule that delivers all packets within M times steps from their injection times. We thus coin the term "bounded delivery time adversary" for the adversary that is committed to be able itself to schedule the injected packets with some finite upper bound on the delivery times of all packets. We show a class of network topologies and a protocol that maintains bounded delivery time, on this class of topologies, against any bounded delivery time adversary. We further give a protocol that achieves bounded delivery time, on any network topology, against a class of adversaries somewhat weaker than the class of all bounded delivery time adversaries.

In this paper we answer open questions posed by Gamarnik and by Borodin et al. [13, 10]. Our work leaves however open the question whether there exists a protocol that achieves bounded delivery time, on any network topology, against all bounded delivery time adversaries.

References

[1] M. Adler and A. Rosén, Tight Bounds for the Performance of Longest-In-System on DAGs, In *Proc. of 19th International Symposium on Theoretical Aspects of Computer Science* (STACS), pp. 88-89, 2002.

[2] W. Aiello, E. Kushilevitz, R. Ostrovsky, A. Rosén. Adaptive packet Routing for Bursty Adversarial Traffic, *Journal of Computer and System Sciences*, Vol. 60, No. 3, pp. 482-509, 2000.

[3] C. Àlvarez, M.J. Blesa, M.J. Serna, Universal Stability of Undirected Graphs in the Adversarial Queueing Model. In *Proc. of 14th Ann. ACM Symposium on Parallel Algorithms and Architectures* (SPAA), pp. 183-197, 2002

[4] M. Andrews, Instability of FIFO in Session Oriented Networks. In *Proc. of the 11th Ann. ACM-SIAM Symposium on Discrete Algorithms* (SODA), pp.440-447, 2000.

[5] M. Andrews, B. Awerbuch, A. Fernández, J. Kleinberg, T. Leighton, and Z. Liu, Universal Stability Results for Greedy Contention-Resolution Protocols, JACM 48(1), pp. 39–69, 2001.

[6] M. Andrews, A. Fernandez, A. Goel, and L. Zhang. Source Routing and Scheduling in Packet Networks. In *Proc. of the 42nd Ann. IEEE Symposium on Foundations of Computer Science* (FOCS), pp. 168-177, 2001

[7] M. Andrews, and L. Zhang, The Effects of Temporary Sessions on Network Performance. *SIAM Jour. on Computing*, Vol. 33, No. 3, pp. 659-673, 2004.

[8] B. Awerbuch, P. Berenbrink, A. Brinkmann, and C. Scheideler, Simple Routing Strategies for Adversarial Systems. In *Proc. of the 42nd Ann. IEEE Symposium on Foundations of Computer Science* (FOCS), pp. 158-167, 2001.

[9] R. Bhattacharjee, A. Goel, and Z. Lotker, Instability of FIFO at Arbitrarily Low Rates in the Adversarial Queueing Model. *SIAM Jour. on Computing*, Vol. 34, No. 2, pp. 318-332, 2005.

[10] A. Borodin, J. Kleinberg, P. Raghavan, M. Sudan, and D. Wiliamson, Adversarial Queuing Theory, JACM 48(1), pp. 13–38, 2001.

[11] A. Borodin, R. Ostrovsky, and Y. Rabani, Stability Preserving Transformations: Packet Routing Networks with Edge Capacities and Speeds. In *Proc of the 12th Ann. ACM-SIAM Symposium on Discrete Algorithms* (SODA), pp. 601–610, 2001.

[12] J. Díaz, D. Koukopoulos, S. Nikoletseas, M. Serna, P. Spirakis, and D. Thilikos, Stability and non-Stability of the FIFO Protocol. In *Proc. of 13th Ann. ACM Symposium on Parallel Algorithms and Architectures* (SPAA), pp. 48-52, 2001.

[13] D. Gamarnik, Stability of Adaptive and non-Adaptive Packet Routing Policies in Adversarial Queuing Networks. In *Proc. of the 31st Ann. ACM Symposium on Theory of Computing* (STOC), pp. 206-214, 1999.

[14] D. Gamarnik, Using Fluid Models to Prove Stability of Adversarial Queueing Networks. In *IEEE Transactions on Automatic Control*, Vol. 45, No. 4, pp. 741–747, 2000.

[15] A. Goel. Stability of Networks and Protocols in the Adversarial Queueing Model for Packet Routing. *Networks*, 37(4):219-224, 2001

[16] D. Koukopoulos, S. E. Nikoletseas, P. G. Spirakis, The Range of Stability for Heterogeneous and FIFO Queueing Networks. *Electronic Colloquium on Computational Complexity* (ECCC TR01-099), 2001.

[17] Z. Lotker, B. Patt-Shamir, and A. Rosén, New Stability Results for Adversarial Queuing. *SIAM Jour. on Computing*, Vol. 33, No. 2, pp. 286-303, 2004.

[18] A. Rosén, A Note on Models for Non-Probabilistic Analysis of Packet-Switching Networks. *IPL*, Vol. 84, No. 5, pp. 237-240, 2002.