Alcom-FT Technical Report Series
ALCOMFT-TR-02-74

On the Stability of Compositions of Universally Stable, Greedy Contention-Resolution Protocols*

Dimitrios Koukopoulos[†] *Marios Mavronicolas*[‡] *Sotiris Nikoletseas*[§] *Paul Spirakis*[§]

Abstract

A distinguishing feature of today's large-scale platforms for distributed computation and communication, such as the *Internet,* is their *heterogeneity,* predominantly manifested by the fact that a wide variety of *communication protocols* are simultaneously running over different distributed hosts. A fundamental question that naturally poses itself for such common settings of heterogeneous distributed systems concerns the preservation (or loss) of important correctness and performance properties of the individual protocols when they are *composed* in a large network. In this work, we address this foundational question for the specific case of stability properties of greedy, contention-resolution protocols operating over a *packet-switched* communication network.

We focus on a basic adversarial model for packet arrival and path determination for which the time-averaged arrival rate of packets requiring a single edge is no more than 1. Roughly speaking, a *contention-resolution protocol* arbitrates the cases where more than one packet wants to advance over a given queue in a single time step. *Stability* requires that the number of packets in the system remains bounded, as the system runs for an arbitrarily long period of time. It is known that several commonly used contention-resolution protocols, such as LIS (*Longest-in-System*), SIS (*Shortest-in-System*), NTS (*Nearest-to-Source*), and FTG (*Furthest-to-Go*) are *universally stable* in this setting – that is, they are stable over all networks. We investigate the preservation of universal stability under compositions for these four greedy, contention-resolution protocols. We discover:

- The composition of any two policies among SIS, NTS and FTG is universally stable.
- The composition of LIS with any of SIS, NTS and FTG is *not* universally stable: we provide interesting combinatorial constructions of networks over which the composition is unstable when the adversary's injection rate is at least 0.519.
- Through an involved combinatorial construction, we significantly improve the current state-of-the-art record for the adversary's injection rate that implies instability for the FIFO (*First-In-First-Out*) protocol to 0.749. Since 0.519 is significantly below 0.749, this last result suggests that the potential for instability incurred by the composition of *two* universally stable protocols may be *worse* than that of some *single* protocol that is not universally stable.

<u>*Note 1:*</u> Due to lack of space, many of our proofs are only sketched in this extended abstract; full proofs are included in a clearly marked Appendix that may be read at the discretion of the Program Committee.

[2]**Contact Author.** Department of Computer Engineering and Informatics, University of Patras, Rion, 265 00 Patras, Greece, & Computer Technology Institute, P. O. Box 1122, 261 10 Patras, Greece. Email: Dimitrios.Koukopoulos@cti.gr

[3]Departemnt of Computer Science, University of Cyprus, 1678 Nicosia, Cyprus. Email: mavronic@ucy.ac.cy

[4]Department of Computer Engineering and Informatics, University of Patras, Rion, 265 00 Patras, Greece, & Computer Technology Institute, P. O. Box 1122, 261 10 Patras, Greece. Email: {nikole,spirakis}@cti.gr

*This work has been partially supported by the IST Program of the European Union under contract numbers IST-1999-14186 (*ALCOM-FT*) and IST-2001-33116 (*FLAGS*).

1 Introduction

1.1 Motivation-Framework

Heterogeneous Networks. A key feature of contemporary large-scale platforms for distributed communication and computation, such as the *Internet,* is their *heterogeneity.* Heterogeneity comes around in many different flavors. For example, the specifics of how the computers in different parts of the network are connected (directly or indirectly) with each other, and the properties of the links that foster the interconnection, is difficult to characterize uniformly. Second, different traffic sources over the Internet (due to varying mechanisms for supporting different classes and qualities of service) result in a heterogeneous mix of traffic traces. Third but not least, although, conceptually, the Internet uses a unified set of protocols, in practice each protocol has been implemented with widely varying features (and of course bugs). (See the recent interesting article by Floyd and Paxson [7] for an extended discussion on the heterogeneity of Internet.) Thus, heterogeneity is a crucial feature that makes it difficult to model, verify and analyze the behavior of such large-scale communication networks.

Objectives. In this work, we embark on a study of the impact of heterogeneity of distributed systems on their correctness and performance properties. More specifically, we wish to pose the general question of which correctness and performance properties of individual, different modules of a distributed system are maintained and which are not when such modules are *composed* into a larger, heterogeneous distributed system. We choose, as a test-bed, the case of distinct *communication protocols* that are simultaneously running on different hosts in a distributed system. We ask, in particular, which (and how) stability properties of greedy, contention-resolution protocols operating over a *packet-switched* communication network are maintained under composition of such protocols.

Framework of Adversarial Queueing Theory. We consider a packet-switched communication network in which packets arrive dynamically at the nodes with predetermined paths, and they are routed at discrete time steps across the edges. We focus on a basic adversarial model for packet arrival and path determination that has been recently introduced in a pioneering work by Borodin *et al.* [3], under the name *Adversarial Queueing Theory.* Roughly speaking, this model views the time evolution of a packet-switched communication network as a game between an *adversary* and a *protocol.* At each time step, the adversary may inject a set of packets into some nodes. For each packet, the adversary specifies a simple path (including an *origin* and *destination*) that the packet must traverse; when the packet arrives to destination, it is absorbed by the system. When more than one packets wish to cross a queue at a given time step, a *contention-resolution* protocol is employed to resolve the conflict. A crucial parameter of the adversary is its *injection rate* r, where $0 < r < 1$. Among the packets that the adversary injects in any time interval I, at most $\lceil r|I| \rceil$ can have paths that contain any particular edge. Such a model allows for adversarial injection of packets, rather than for injection according to a randomized, oblivious process (cf. [4]).

Stability. *Stability* requires that the number of packets in the system remains bounded, as the system runs for an arbitrarily long period of time. Naturally, achieving stability in a packet-switched communication network depends on the *rate* at which packets are injected into the system, and on the employed contention-resolution protocol. Till our work, the study of stability has focused on *homogeneous* networks, that is, on networks in which the same contetntion-resolution protocol is running at all queues. In this work, we embark on a stydy of the effect of composing contention-resolution protocols on the stability of the resulting system. (By *composition* of contention-resolution protocols, we mean the simulytaneous use of different such protocols at different queues of the system.)

Greedy Contention-Resolution Protocols. We consider only *greedy* protocols– ones that always advance a packet across a queue (but one packet at each discrete time step) whenever there resides at least one packet in the queue. The protocol specifies which packet will be chosen. We study five greedy protocols (all of which enjoy simple implementations): Say that a protocol is *stable* [3] on a given

Protocol name	Which packet it advances:	US
Shortest-in-System (SIS)	The one that was most recently injected into the network	$\surd$
Longest-in-System (LIS)	The one that was least recently injected into the network	$\surd$
Furthest-to-Go (FTG)	The one that is furthest from its destination	$\surd$
Nearest-to-Source (NTS)	The one that its nearest to its origin	$\surd$
First-In-First-Out (FIFO)	The one that arrived earliest at the queue	**X**

Table 1: Contention-resolution protocols considered in this paper. (**US** stands for universally stable.)

network if it induces a *bounded* number of packets in the network against any adversary with injection rate less than 1. (Note that the bound may depend on parameters of the network.) The first four of these protocols (namely, SIS, LIS, FTG and NTS) are *universally stable*– each is stable on *all* networks [1, Section 2.1]. In contrast, FIFO (one of the most popular queueing disciplines, because of its simplicity) is not universally stable [1, Theorem 2.10].

Approach. We consider all combinations of two from the four universally stable protocols, and we examine whether the corresponding composition is universally stable. We either show that it is, or we demonstrate a network and an adversary (with some specific injection rate less than 1) such that the composition is not stable on the network (against the adversary). In addition, in order to qualitatively evaluate how unstable are the compositions that turn out not to be universally stable, we also consider the FIFO protocol, which is known not be universally stable; we measure the instability of the composition against that of FIFO by establishing the best lower bound we can on the adversary's injection rate that implies instability for the composition and for FIFO, and we compare the two resulting lower bounds.

1.2 Contribution

Summary of Results. In this work, we initiate the study of the stability properties of heterogeneous networks with compositions of greedy contention-resolution protocols, such as SIS, LIS, FTG, NTS and FIFO, running on top of them. Our results are three-fold; they are summarized as follows:

- We establish universal stability for compositions of any two among the (universally stable) SIS, FTG, and NTS protocols (Theorem 3.1).
- We establish that, surprisingly, the composition of LIS with any of SIS, NTS and FTG is *not* universally stable (Theorem 4.1).

  To show this, we provide interesting combinatorial constructions of networks, for each queue of which we specify the contention-resolution protocol to be used, so that the composition of the protocols is unstable if the injection rate of the adversary is at least 0.519.
- We establish a new lower bound on the instability threshold of the FIFO protocol. More specifically, we provide an involved combinatorial construction of a network containing only FIFO queues and an adversary with injection rate 0.749 that result to instability (Theorem 5.1).

  This result not only significantly improves the current record (that is, the lowest known) instability threshold for FIFO [5, Theorem 3.1]. More importantly perhaps, it provides, as we argue, a standard for evaluating the lower bound (0.519) on the instability thresholds we established for the not universally stable compositions (Theorem 4.1). Since 0.519 is substantially less than 0.749 (in the climax [0.1]), and since lowering the instability threshold for FIFO has undergone a series of subsequent improvements in recent papers in the literature [1, 8, 5] culminating to the 0.749 shown in this work, these together may modestly suggest that composing two universally stable protocols may, surprisingly, turn out to exhibit more unstable behavior than a single protocol that is already known to not be universally stable (such as FIFO).

Tools for Establishing Stability of Compositions. We significantly extend ideas and techniques used by Andrews *et al.* [1, Section 2.1] for establishing universal stability of *individual* protocols to our setting that accomodates two different protocols running in the network.

Tools for Establishing Instability of Compositions. The combinatorial constructions of networks and adversaries that we have employed for showing that certain compositions of universally stable protocols are not universally stable significantly extend ones that appeared before in [1, 3, 5]. In more detail, some of the tools we devise in order to obtain constructions of networks and adversaries that imply improved bounds are the following:

- We employ combinatorial constructions of networks with multiple "parallel" paths between a *common* origin and destination; we judiciously use such paths for the simultaneous injection of various non-overlapping flows.
- We introduce and use the technical notions of *investing flow* and *short intermediate flow*; these are some special cases of packet flows that we use in our adversarial constructions that consist of inductive *phases*. Roughly speaking, an investing flow injects packets in a phase which will remain in the system till the beginning of the next phase, in order to guarantee the inductive hypothesis for the next phase; on the other hand, short intermediate flows consist of packets injected on judiciously chosen paths of the network and their role is to block *all* packets of the investing flows (so that the latter will indeed remain in the system).

Tools for Establishing Instability of FIFO. We present involved combinatorial constructions of a network and an associated adversary that leads to instability. In order to create instability, we heavily exploit the *fair mixing* property of FIFO stating that if two packet flows arrive at the same queue simultaneously for a time period, they will get mixed according to their initial proportions. (This property appears to characterize FIFO, and it does not hold for the other contention-resolution protocols considered in this paper.) Moreover, we pursue a tight analysis of the time evolution of the various packet flows by exactly computing the number of packets remaining at various queues of the network, and their corresponding delays. The network construction we employ contains multiple "parallel" paths in order to facilitate further delay of packets initially residing in the system.

General Contribution. We feel that our study of the stability and instability properties of networks hosting compositions of multiple contention-resolution protocols (within the theoretical context of Adversarial Queueing Theory) provides an important first step towards understanding the impact of heterogeneity on the correctness and performance properties of large-scale communication networks such as the Internet.

1.3 Related Work and Comparison

Composing Protocols and Objects. The issue of composing distributed protocols (resp., objects) to obtain other protocols (resp., objects), and the properties of the resulting (*composed*) protocols (resp., objects), has a rich record in Distributed Computing Theory (see, e.g., [11]). For example, Fernández *et al.* [6] study techniques for the composition of (identical) *causal* DSM systems from smaller modules each individually satisfying *causality*. Herlihy and Wing [9] establish that a composition of *linearizable* memory objects (possibly distinct), each managed by its own protocols, preserves linearizability. In the

	LIS	SIS	NTS	FTG
LIS	**US** ([1])			
SIS	[0.519,1] (Thm. 4.1)	**US** ([1])		
NTS	[0.519,1] (Thm. 4.1)	**US** (Thm. 3.1)	**US** ([1])	
FTG	[0.519,1] (Thm. 4.1)	**US** (Thm. 3.1)	**US** (Thm. 3.1)	**US** ([1])

Table 2: Range of injection rates for which the composition of the two protocols is unstable on some network. We denote **US** the universally stable compositions, i.e., the compositions for which the range is [0,1].

community of Security Protocols, the statement that security is not compositional is considered to be folklore [12].

Adversarial Queueing Theory. The model of *Adversarial Queueing Theory* was developed by Borodin *et al.* [3] as a more realsistic model that replaces traditional stochastic assumptions made in Queueing Theory (cf. [4]) by more robust, worst-case ones. Subsequently, the Adversarial Queueing Theory model, and corresponding stability and instability issues, received a lot of interest and attention (see, e.g., [1, 2, 5, 8, 13]).

Stability and Instability Results. The universal stability of SIS, LIS, NTS and FTG was established by Andrews *et al.* [1, Section 2.1]. The instability of FIFO (on a specific network) was first established by Andrews *et al.* [1, Theorem 2.10]. Lower bounds of 0.85, 0.84 and 0.8357 on the instability threshold of FIFO (in the model of Adversarial Queueing Theory) were presented before by Andrews *et al.* [1, Theorem 2.10], Goel [8] and Diaz *et al.* [5, Theorem 3]. To the best of our knowledge, no previous work addressed the stability and instability properties of networks consisting of queues using multiple contention-resolution protocols.

Summary. For purpose of completeness and comparison, we summarize, in Table 2, all results shown in this work and in [1] that provide bounds on stability and instability properties of the universally stable, greedy contention-resolution protocols, and their compositions, that we considered.

1.4 Road Map

The rest of this paper is organized as follows. In Section 2, we discuss some preliminaries that will be useful in some of our later constructions. In Section 3, we establish the universal stability of certain compositions of universally stable protocols. The instability of some other compositions of universally stable protocols is established in Section 4. In Section 5 we present our instability result for the FIFO protocol. We conclude, in Section 6, with a discussion of our results and some open problems.

2 Preliminaries

The definition of a *bounded adversary* A of rate (r, b) (where $b \geq 1$ is a natural number and $0 < r < 1$) in the Adversarial Queueing Theory model [3] requires that for any edge e and any interval I, the adversary injects no more than $r|I| + b$ packets during I that require edge e at their time of injection. Such a model allows for adversarial injection of packets that are "bursty" using the integer $b > 0$. Say that a packet p *requires* an edge e at time t if e lies on the path from its position at time t to its destination.

This definition for the adversary is used in Section 3 for proving that specific compositions of protocols are universally stable. On the other hand, when we consider adversarial constructions for proving instability of compositions of specific protocols (Section 4) and FIFO protocol (Section 5.1) in which we want to derive lower bounds, it is advantageous to have an adversary that is as weak as possible. Thus, for these purposes, we say that an adversary A has injection rate r if for every $t \geq 1$, every interval I of t steps, and every edge e, it injects no more than $r|t|$ packets during I that require edge e at the time of their injection. Also, for simplicity, and in a way similar to that in [1], we omit floors and ceilings and sometimes count time steps and packets roughly. This only results to loosing small additive constants while we gain in clarity.

In order to formalize the behavior of a network under the Adversarial Queueing model, we use the notions of *system* and *system configuration*. A triple of the form (G, A, P) where G is a network, A is the adversary and P is the used protocol on the network queues is called a system. Furthermore, the configuration C^t of a system (G, A, P) in every time step t is a collection of sets $\{S_e^t : e\epsilon G\}$, such that S_e^t is the set of packets waiting in the queue of the edge e at the end of step t. If the current system configuration is C^t, then we can go to the system configuration C^{t+1} for the next time step as follows: (i) Addition of new packets to some of the sets S_e^t, each of which has an assigned path in G, and (ii) for each non-empty set S_e^t deletion of a single packet $p\epsilon S_e^t$ and its insertion into the set S_f^{t+1} where f is the edge following e on its assigned path (if e is the last edge on the path of p, then p is not inserted into any set.) The time evolution of the system is a sequence of such configurations $C^1, C^2, \ldots$, such that for all edges e and all intervals I, no more than $r|I| + b$ packets are introduced during I with an assigned path containing e. An execution of the adversary's construction on a system (G, A, P) determines the time evolution of the system configuration.

In the constructions of executions in Sections 4 and 5, we split time into *phases*. In each phase, we study the evolution of the *system configuration* by considering corresponding *time rounds*. For each phase, we inductively prove that the number of packets of a specific subset of queues in the system increases in order to guarantee instability. This inductive argument can be applied repeatedly, thus showing instability. In addition, our constructions use networks that can be split into two symmetric parts. Thus, the inductive argument needs to be applied twice to establish increase in the number of packets residing at *two* different queues.

Also, in order to make our inductions work, we assume that there is a sufficiently large number of packets s_0 in the initial system configuration. This will imply instability results for networks with an *empty* initial configuration, as established by Andrews *et al.* [1, Lemma 2.9].

3 Universally Stable Compositions of Universally Stable Protocols

In this section, we present compositions of universally stable protocols that remain universally stable. In order to prove the following theorem we make the following assumptions and definitions. Let $0 < \epsilon < 1$ be a real number. We assume that $r = 1 - \epsilon$, m is the number of network edges, and d is the length of the longest simple directed path in the network. Let's now define a sequence of numbers by the recurrence $k_j = \frac{mk_{j-1}+mb}{\epsilon}$, where $k_1 = \frac{mb}{\epsilon}$. Our techniques here are motivated by corresponding techniques in Andrews *et al.* [1].

Theorem 3.1 *If the used queueing disciplines in the system are a)* SIS *and* FTG*, then the system (G, A,* SIS*,* FTG*) is stable, no queue ever contains more than k_d packets and no packet spends more than $\frac{1}{\epsilon}(db + \sum_{i=1}^{d} k_i)$ steps in the system, while if they are b)* NTS *and* FTG *or c)* SIS *and* NTS*, then there are never more than k_d packets in the system and no queue contains more than $\frac{1}{\epsilon}(k_{d-1} + b)$ packets, where d is the length of the longest simple directed path in G and k_i is an appropriately defined sequence of numbers.*

Sketch of proof: In order to prove this theorem, we first show the following two lemmas:

Lemma 3.2 *Let p be a packet waiting in a queue e at time t and suppose there are currently $k-1$ other packets in the system requiring e that have priority over p. Then p will cross e within the next $\frac{k+b}{\epsilon}$ steps if the queueing discipline in queue e is* SIS, NTS *or* FTG.

Lemma 3.3 *When a packet p crossing its path arrives at the j^{th} edge on its path, there are at most $k_j - 1$ other packets requiring to traverse this edge with priority over p, if the used queueing protocol is a)* SIS *or* NTS, *b)* SIS *or* FTG, *and c)* NTS *or* FTG.

Lemma 3.2 claims that if a packet p waits in a queue e at time t and there are currently $k-1$ other packets in the system requiring to traverse edge e that have priority over p, then p will cross e within the next $\frac{k+b}{\epsilon}$ steps either the queueing discipline of queue e is SIS or NTS or FTG. In Lemma 3.3, we specialize Lemma 3.2 taking into account the distance of the queue e, in which packet p arrives at time t crossing its specified path, in relation to the first queue in its path. In this case, we prove that if queue e has distance j from the first queue on $p's$ path then there are at most $k_j - 1$ other packets in the system requiring to traverse the edge e that have priority over p when the system queues have as protocols (a) SIS or FTG, (b)NTS or FTG and (c) SIS or NTS.

Then based on these two lemmas, we prove this theorem using contradiction. Firstly, let's assume that there are $k_d + 1$ packets at some time all requiring the same edge. Then, the packet with the lowest priority of the $k_d + 1$ packets contradicts Lemma 3.3. Combining both results, a packet p takes at most $\frac{k_j+b}{\epsilon}$ steps to cross the j^{th} edge on its path. Therefore, the delay bound is $D = \frac{db+\sum_{i=1}^{d} k_i}{\epsilon}$. No packet spends more than D steps in the system. ∎

4 Unstable Compositions of Universally Stable Protocols

In this section, we show that the composition of LIS protocol with any of SIS, NTS and FTG protocols is *not* universally stable. Before proceeding to the adversary constructions for proving instability we give two basic definitions.

Definition 4.1 *We denote X_i the set of packets that are injected by the adversary into the system in the i_{th} round of a phase. These packet sets are characterized as investing flows because the number of their packets that will remain into the system at the end of the phase in which they have been injected, will be a portion of the packets that will be used as the initial ones in the next phase ensuring the reproduction of the inductive hypothesis.*

Definition 4.2 *We denote $S_{i,k}$ the k_{th} set of packets the adversary injects into the system in the i_{th} round of a phase. These packet sets are characterized as short intermediate flows because their only purpose is to block other packet sets by using suitable paths.*

Theorem 4.1 *Let $r \geq 0.519$. There is a network N_1 and an adversary A of rate r, such that the system (N_1,A,Queueing Disciplines) is unstable, starting from a non-empty initial configuration if the used queueing disciplines are a)* LIS *and* SIS, *b)* LIS *and* NTS *or c)* LIS *and* FTG.

Sketch of proof: Consider the network N_1 in Figure 1.

Inductive Hypothesis: At the beginning of phase j, there are s_j packets that are queued in the queues f_4', f_7', f_8' (in total) requiring to traverse the edges e_0, f_1, f_2, f_4, f_7.

Induction Step: At the beginning of phase $j+1$ there will be more than s_j packets that will be queued in the queues f_4, f_7, f_8, requiring to traverse the edges $e_1, f_1', f_2', f_4', f_7'$.

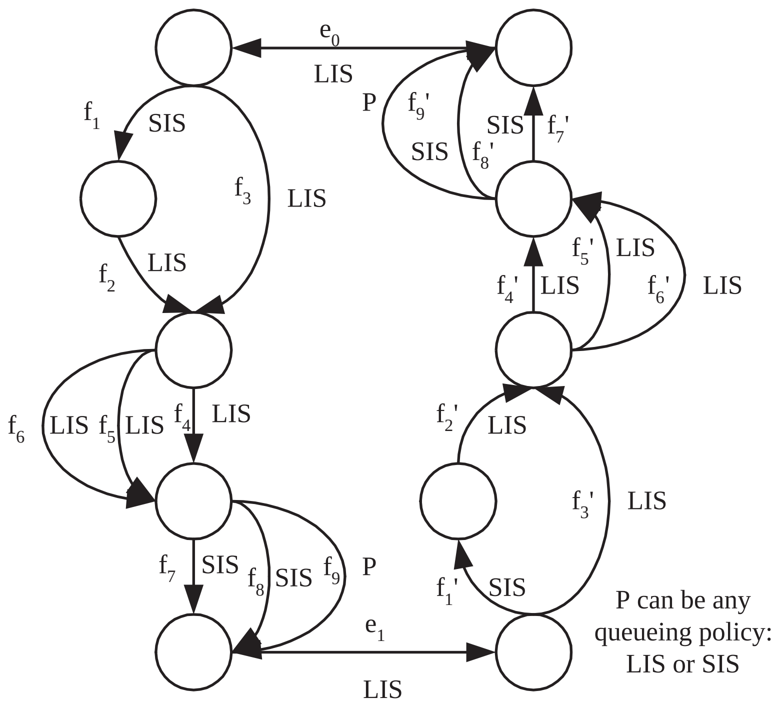

Figure 1: The network N_1 running a composition of LIS and SIS

The intuition behind this adversary construction and network topology is basically the preservation of the newly injected investing flows in each round of a phase inside the network until the end of the phase during which they have been injected. The tools we use to achieve this goal are the injections of short intermediate flows ($S_{i,k}$) and the introduction of parallel edges (f_4, f_5, f_6 and f_7, f_8, f_9) in the network topology. Although the short intermediate flows do not contribute actually to the number of packets that will be used as the initial flow for the next phase, their role is essential as they are used for blocking investing flows of packets in consecutive rounds. Moreover, some of them have another role, too. They block short intermediate flows that have been injected in certain rounds in order they can be used for blocking investing flows in the next round of the one in which they have been injected. Also, an important point we should mention is the use of the short intermediate flow that is injected in the first round of a phase j ($S_{1,1} - flow$) to block a portion of the initial packets that are in the system at the beginning of the phase (s_j) in order these packets to be used twice for blocking investing flow X_1 to the first and the next round of the current phase. As far as concerns the parallel edges, the purpose of their presence in the network topology is to be guaranteed that the paths of the packet flows injected in the same round do not overlap. The first set of parallel edges f_4, f_5, f_6 is used in order short intermediate flows that are injected at the same round to not overlap, while the second set of parallel edges f_7, f_8, f_9 is used mainly in order the paths of investing flows and the paths of short intermediate flows that are injected at the same round to not overlap.

We will construct an adversary A such that the induction step will hold. Proving that the induction step holds, we ensure that the inductive hypothesis will hold at the beginning of phase $j+1$ for the symmetric edges with an increased value of s_j, $s_{j+1} > s_j$. In order to prove that the induction step works, we should consider that there is a large enough number of packets s_j in the initial system configuration. During phase j the adversary plays four rounds of injections. The sequence of injections is as follows:

Round 1: It lasts s_j time steps.

At the beginning of this round, there are s_j packets ($S - flow$) in the queues f_4', f_7', f_8' (in total) requiring to traverse the edges e_0, f_1, f_2, f_4, f_7.

Adversary's behavior. During this round the adversary injects in queue e_0 a set X_1 of $|X_1| = rs_j$ packets wanting to traverse the edges $e_0, f_3, f_4, f_7, e_1, f_1^{'}, f_2^{'}, f_4^{'}, f_7^{'}$. At the same time, the adversary injects a set $S_{1,1}$ of $|S_{1,1}| = rs_j$ packets in queue f_1 that require to traverse only the edge f_1.

At the end of this round, there are rs_j packets of $S - flow$ in queue f_1 wanting to traverse the edges f_1, f_2, f_4, f_7. Also, there is a set X_1 of $|X_1| = rs_j$ packets in queue e_0 wanting to traverse the edges $e_0, f_3, f_4, f_7, e_1, f_1^{'}, f_2^{'}, f_4^{'}, f_7^{'}$.

Round 2: It lasts rs_j time steps.

Adversary's behavior. During this round the adversary injects a set X_2 of $|X_2| = r^2 s_j$ packets in queue e_0 requiring to traverse the edges $e_0, f_3, f_4, f_8, e_1, f_1^{'}, f_2^{'}, f_4^{'}, f_7^{'}$. In addition, the adversary injects a set $S_{2,1}$ of $|S_{2,1}| = r^2 s_j$ packets in queue f_2 wanting to traverse the edges f_2, f_5, f_7.

At the end of this round, there are rs_j packets of $X_1 - flow$ in queue f_4 wanting to traverse the edges $f_4, f_7, e_1, f_1^{'}, f_2^{'}, f_4^{'}, f_7^{'}$. Also, there are $r^2 s_j$ packets of $X_2 - flow$ in queue e_0 requiring to traverse the edges $e_0, f_3, f_4, f_8, e_1, f_1^{'}, f_2^{'}, f_4^{'}, f_7^{'}$ and $r^2 s_j$ packets of $S_{2,1} - flow$ in queue f_2 wanting to traverse the edges f_2, f_5, f_7.

Round 3: It lasts $r^2 s_j$ time steps.

Adversary's behavior. During this round the adversary injects a set X_3 of $|X_3| = r^3 s_j$ packets in queue f_4 wanting to traverse the edges $f_4, f_9, e_1, f_1^{'}, f_2^{'}, f_4^{'}, f_7^{'}$. Also, the adversary injects a set $S_{3,1}$ of $|S_{3,1}| = r^3 s_j$ packets in queue f_5 wanting to traverse the edges f_5, f_7. Finally, the adversary injects a set $S_{3,2}$ of $|S_{3,2}| = r^3 s_j$ packets in queue f_2 wanting to traverse the edges f_2, f_6, f_8.

At the end of this round, there are rs_j packets of $X_1 - flow$ in queues f_4, f_7 in total ($rs_j - r^2 s_j$ packets in queue f_4 and rs_j packets in queue f_7) wanting to traverse the edges $f_4, f_7, e_1, f_1^{'}, f_2^{'}, f_4^{'}, f_7^{'}$. Also, in queue f_4 there are $r^2 s_j$ packets of $X_2 - flow$ requiring to traverse the edges $f_4, f_8, e_1, f_1^{'}, f_2^{'}, f_4^{'}, f_7^{'}$ and $r^3 s_j$ packets of $X_3 - flow$ wanting to traverse the edges $f_4, f_9, e_1, f_1^{'}, f_2^{'}, f_4^{'}, f_7^{'}$. Moreover, there are $r^3 s_j$ packets of $S_{3,1} - flow$ in queue f_5 requiring to traverse the edges f_5, f_7 and $r^3 s_j$ packets of $S_{3,2} - flow$ in queue f_2 requiring to traverse the edges f_2, f_6, f_8.

Round 4: It lasts $r^3 s_j$ time steps.

Adversary's behavior. During this round, the adversary injects a set X_4 of $|X_4| = r^4 s_j$ packets in queue f_3 wanting to traverse the edges $f_3, f_4, f_7, e_1, f_1^{'}, f_2^{'}, f_4^{'}, f_7^{'}$. Besides the short intermediate flows ($S_{3,1}, S_{3,2}$) that have been injected into the system during the previous round and still remain into the system at the beginning of this round, the system configuration at the beginning of round 4 also consists of the investing flows X_1, X_2, X_3 that have been injected into the system during the previous rounds. During this round, these investing flows along with the investing flow that is injected into the system during this round ($X_4 - flow$) continue to remain into the system independently of the adversarial injection rate r because they are blocked by the short intermediate flows $S_{3,1}, S_{3,2}$.

At the end of this round, the number of packets in queues f_4, f_7, f_8 requiring to traverse the edges $e_1, f_1^{'}, f_2^{'}, f_4^{'}, f_7^{'}$ is $s_{j+1} = |X_1| + |X_2| + |X_3| + |X_4| = rs_j + r^2 s_j + r^3 s_j + r^4 s_j$.

In order to have instability, we must have $s_{j+1} > s_j$. Therefore, we should have $rs_j + r^2 s_j + r^3 s_j + r^4 s_j > s_j$, i.e. $r \geq 0.519$. This argument can be repeated for an infinite number of phases ensuring that the number of packets at the end of a phase will be bigger than at the beginning of the phase.

b) This part of the theorem can be proved similarly to the previous part. One difference here is the replacement of the protocol of queues that use SIS by NTS. The topology of the used network N_2 and the adversary construction for proving instability of the system (N_2,A,LIS, NTS) are similar to the first part of the theorem. Especially, short intermediate flows have the same blocking effects as before over investing flows because their injection in the same queues as before are enough to guarantee their priority over investing flows when they conflict in queues that use NTS.

c) This part of the theorem can be proved similarly to the previous parts. The construction of the adversary A for proving instability of the system (N_3,A,LIS, FTG) is similar to the other parts. As far as concerns the network topology a difference here is that the queues, that have as protocol SIS in the first part, are now using FTG. Another difference that concerns the used network topology is that it contains additional paths that start at FTG queues and have sufficient lengths, such that the injected short intermediate packet flows have the same blocking effects over the injected investing packet flows, as in the proofs of the previous parts, when they conflict in queues that use FTG. ■

5 A Lower Bound for FIFO Instability

In this section, we present an adversary construction that lowers significantly the injection rate bound for which FIFO is unstable to 0.749 on the network that we consider in Figure 2.

Theorem 5.1 *Let $r \geq 0.749$. There is a network $\mathcal{N}$ and an adversary A of rate r, such that the ($\mathcal{N}$, A,* FIFO*) system is unstable, starting from a non-empty initial configuration.*

Sketch of proof: We consider the network $\mathcal{N}$ in Figure 2.

Inductive Hypothesis: At the beginning of phase j, there are s_j packets that are queued in the queues $e_0, f_3^{'}, f_4^{'}, f_5^{'}, f_6^{'}$ (in total) requiring to traverse the edges e_0, f_1, f_3, f_5, all these packets are able to depart from their initial edges to the symmetric part of the network (f_1, f_3, f_5) as a continuous flow in s_j time steps, and the number of packets that are queued in queues $f_4^{'}, f_6^{'}$ is bigger than the number of packets that are queued in queues $f_3^{'}, f_5^{'}$.

Induction Step: At the beginning of phase $j+1$ there will be more than s_j packets (s_{j+1} packets) that will be queued in the queues f_3, f_5, f_4, f_6, e_1 (in total) requiring to traverse the edges $e_1, f_1^{'}, f_3^{'}, f_5^{'}$, all of which will be able to depart from their initial edges to the symmetric part of the network ($f_1^{'}, f_3^{'}, f_5^{'}$) in s_{j+1} time steps as a continuous flow and the number of packets that will be queued in queues f_4, f_6 will be bigger than the number of packets that will be queued in queues f_3, f_5.

Notice that our inductive argument claims that if at the beginning of phase j all s_j packets, that are queued in queues $e_0, f_3^{'}, f_4^{'}, f_5^{'}, f_6^{'}$ requiring to traverse the edges e_0, f_1, f_3, f_5, manage to traverse their initial edges in s_j time steps as a continuous flow, then at the beginning of phase $j+1$ all s_{j+1} packets, that will be queued in queues f_3, f_5, f_4, f_6, e_1 requiring to traverse the edges $e_1, f_1^{'}, f_3^{'}, f_5^{'}$, will be able to traverse their initial edges in s_{j+1} time steps as a continuous flow. This argument guarantees the reproduction of the inductive hypothesis in queues f_3, f_5, f_4, f_6, e_1 even if there are flows (in particular in queues f_3, f_4, f_5) that do not want to traverse the edges $e_1, f_1^{'}, f_3^{'}, f_5^{'}$, the packets of which are regularly spread among the packets that want to traverse these edges. Furthermore, this argument implies the third part of the inductive argument, which claims that if at the beginning of phase j, the number of packets that are queued in queues $f_4^{'}, f_6^{'}$ is bigger than the number of packets that are queued in queues $f_3^{'}, f_5^{'}$, then at the beginning of phase $j+1$ the number of packets that will be queued in queues f_4, f_6 will be bigger than the number of packets that will be queued in queues f_3, f_5. This happens because in the first round of the adversary's construction we inject packets in queue $f_4^{'}$ and if the third part of the inductive hypothesis does not hold, then we cannot guarantee that all the initial s_j packets will depart their initial edges to the edges f_1, f_3, f_5 in s_j time steps as a continuous flow. However, we include it into the inductive hypothesis for readability reasons.

We will construct an adversary A such that the induction step will hold. Proving that the induction step holds, we ensure that the inductive hypothesis will hold at the beginning of phase $j+1$ for the

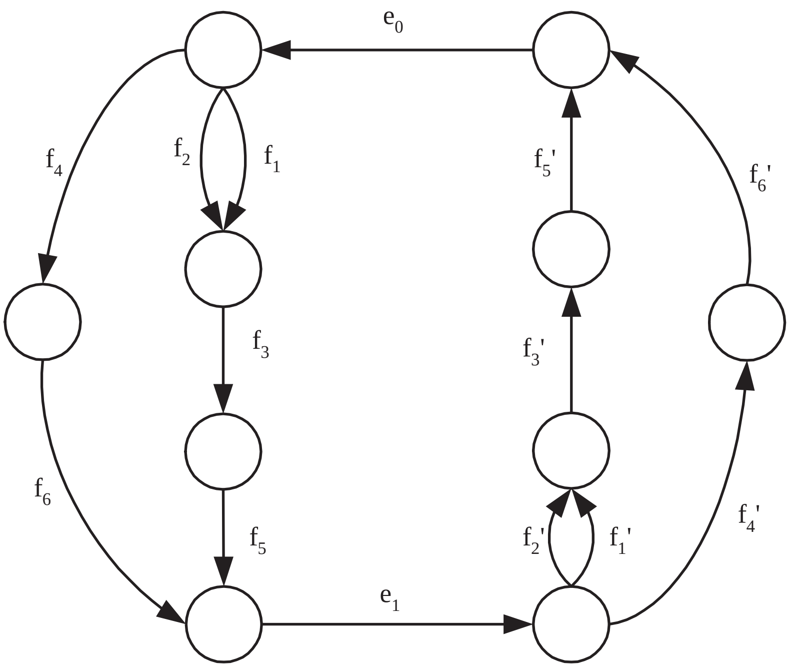

Figure 2: FIFO network $\mathcal{N}$

symmetric edges with an increased value of s_j packets, $s_{j+1} > s_j$. From the inductive hypothesis, initially, there are s_j packets (called $S - flow$) in the queues $e_0, f_3^{'}, f_4^{'}, f_5^{'}, f_6^{'}$ requiring to traverse the edges e_0, f_1, f_3, f_5. In order to prove the induction step, it is assumed that there is a set S with a large enough number of $|S| = s_j$ packets in the initial system configuration. During phase j the adversary plays three rounds of injections. The sequence of injections is as follows:

Round 1: It lasts s_j time steps.

Adversary's behavior. During these steps, the adversary injects a set X of $|X| = rs_j$ packets in queue $f_4^{'}$ wanting to traverse the edges $f_4^{'}, f_6^{'}, e_0, f_2, f_3, f_5, e_1, f_1^{'}, f_3^{'}, f_5^{'}$ and a set S_1 of $|S_1| = rs_j$ packets in f_1 wanting to traverse f_1.

At the end of this round, all the packets of the set X are queued in e_0, while in queue f_1 remains a set S_{rem} of $|S_{rem}| = \frac{rs_j}{r+1}$ packets from the set S and a set $S_{1,rem}$ of $|S_{1,rem}| = \frac{r^2 s_j}{r+1}$ packets from the set S_1 mixed on a proportion equal to their initial proportion of their sizes (fair mixing property).

Round 2: It lasts rs_j steps.

Adversary's behavior. The adversary injects a set Y of $|Y| = r^2 s_j$ packets in queue $f_4^{'}$ requiring to traverse the edges $f_4^{'}, f_6^{'}, e_0, f_4, f_6, e_1, f_1^{'}, f_3^{'}, f_5^{'}$. At the same time, the adversary injects a set S_2 of $|S_2| = r^2 s_j$ packets in f_2 wanting to traverse f_2, a set S_3 of $|S_3| = r^2 s_j$ packets in f_3 wanting to traverse f_3, and a set S_4 of $|S_4| = r^2 s_j$ packets in f_5 wanting to traverse f_5.

At the end of this round all the packets of the set Y are queued in queue e_0. Also, a set X_{rem,f_2} of $|X_{rem,f_2}| = \frac{r^2 s_j}{r+1}$ packets from the set X and a set S_{2,rem,f_2} of $|S_{2,rem,f_2}| = \frac{r^3 s_j}{r+1}$ packets from the set S_2 remain in queue f_2 mixed on a proportion equal to their initial proportion of their sizes. Furthermore a set X_{rem,f_3} of $|X_{rem,f_3}| = \frac{r^3 s_j + rs_j}{(r+1)(r^2+r+2)}$ packets from the set X, a set S_{rem,f_3} of $|S_{rem,f_3}| = \frac{r^3 s_j + rs_j}{(r+1)(r^2+r+2)}$ packets from the set S and a set $S_{3,rem}$ of $|S_{3,rem}| = \frac{r^4 s_j + r^2 s_j}{r^2+r+2}$ packets from the set S_3 remain in queue f_3 mixed on the proportion of the sizes with which they arrive in queue f_3 during this round. Finally, $\frac{r^4 s_j + r^2 s_j}{(r^2+r+2)(r^3+r^2+2r+2)}$ packets from the set X, $\frac{r^4 s_j + r^2 s_j}{(r^2+r+2)(r^3+r^2+2r+2)}$ packets from the set S, and $\frac{r^5 s_j + r^3 s_j}{r^3+r^2+2r+2}$

packets from the set S_4 remain in queue f_5 mixed on the proportion of the sizes with which they arrive in queue f_3 during this round.

Round 3: It lasts $r^2 s_j$ time steps.

Adversary's behavior. During this round the adversary injects a set S_5 of $|S_5| = r^3 s_j$ packets in queue f_4 requiring to traverse the edge f_4 and a set Z of $|Z| = r^3 s_j$ packets in queue f_6 requiring to traverse the edges $f_6, e_1, f_1^{'}, f_3^{'}, f_5^{'}$.

At the end of this round a set Y_{rem} of $|Y_{rem}| = \frac{r^3 s_j}{r+1}$ packets from the set Y and a set $S_{5,rem}$ of $|S_{5,rem}| = \frac{r^4 s_j}{r+1}$ packets from the set S_5 remain in queue f_4 mixed on a proportion equal to their initial proportion of their sizes. Furthermore, a set Y_{rem,f_6} of $|Y_{rem,f_6}| = \frac{r^4 s_j}{(r+1)(r^2+r+1)}$ packets from the set Y and a set Z_{rem,f_6} of $|Z_{rem,f_6}| = \frac{r^5 s_j}{r^2+r+1}$ packets from the set Z are queued in queue f_6 mixed on the proportion of the sizes with which they arrive in queue f_6 during this round. Notice also that the total number of packets that are queued in queue f_3 at the beginning of round 3 is $|T_2| = |X_{rem,f_3}| + |S_{rem,f_3}| + |S_{3,rem}| = \frac{(r^5+r^4+3r^3+r^2+2r)s_j}{(r+1)(r^2+r+2)}$. However, $|T_2| \geq r^2 s_j, \forall r$. Thus, a number of $X_{rem,f_3}, S_{rem,f_3}, S_{3,rem}$ packets will remain in queue f_3 at the end of round 3. This number is $|T_3| = \frac{(2r-r^2-r^4)s_j}{(r+1)(r^2+r+2)}$. From this number, the number of packets that belong to the set S_3 is $|S_{3,rem,f_3}| = \frac{(2r^2-r^3-r^5)s_j}{(r^2+r+2)^2}$. In addition, the total number of packets that are in f_2 at the end of round 2 is $|T_5| = r^2 s_j$ that is equal to the round's time duration. Thus, all the packets that belong to the set X and were in f_2 at the end of round 2 ($|X_{rem,f_2}|$) traverse now f_2 and arrive to f_3 where they are blocked.

In order to have instability the number of packets that are in queues f_3, f_4, f_5, f_6, e_1 requiring to traverse the edges $e_1, f_1^{'}, f_3^{'}, f_5^{'}$ at the end of this round, s_{j+1}, should be more than the initial s_j packets that were queued in the system in corresponding queues at the beginning of round 1 ($s_{j+1} > s_j$). Therefore, it should hold $|Z| + |Y| + |X_{rem,f_2}| + |X_{rem,f_3}| + |X_{pass,f_3}| - |T_{round3}| > s_j$. Substituting properly we take $r^3 s_j + r^2 s_j + \frac{r^2 s_j}{r+1} + \frac{r^3 s_j + r s_j}{(r+1)(r^2+r+2)} + \frac{r^4 s_j + r^2 s_j}{(r^2+r+2)(r^3+r^2+2r+2)} - r^2 s_j > s_j$. This holds for $r \geq 0.749$. Now in order to conclude the proof we should also consider the following lemma:

Lemma 5.2 *For $r \geq 0.686$, the number of packets that remain in queues f_3, f_5 ($Q(f_3), Q(f_5)$) at the end of round 3 is less than or equal to the number of packets that remain in queues f_4, f_6 at the end of round 3 ($Q(f_4), Q(f_6)$).*

Notice that we have, till now, managed to reproduce the inductive hypothesis in queues f_3, f_5, f_4, f_6, e_1 but with some packet flows (in particular in queues f_4, f_3, f_5) having empty spaces (packets that don't want to traverse the edges $e_1, f_1^{'}, f_3^{'}, f_5^{'}$). In order for the induction step to work we must show that all the packets in these queues will manage to depart to the symmetric part of the network ($f_1^{'}, f_3^{'}, f_5^{'}$) in s_{j+1} time steps as a continuous flow. As we have shown above all the packets that are queued in queue e_1 want to traverse the edges $e_1, f_1^{'}, f_3^{'}, f_5^{'}$ and their flow is continuous without empty spaces (packets that do not want to traverse the edges $e_1, f_1^{'}, f_3^{'}, f_5^{'}$). Also, from Lemma 5.2, the number of packets in queues f_4, f_6 is bigger than the number of packets in queues f_3, f_5. Furthermore, the packets that are queued in queue f_6 can be seen as a continuous flow that wants to traverse the edges $e_1, f_1^{'}, f_3^{'}, f_5^{'}$, while the set of packets in queue f_4 consists of packets that want to traverse the edges $e_1, f_1^{'}, f_3^{'}, f_5^{'}$ (Y_{rem}) and packets that are injections which require to traverse a single edge ($S_{5,rem}$), which can be considered as empty spaces. Because of that we should show that all the Y_{rem} packets manage to leave the edge f_4 during the s_{j+1} time steps. This holds from Lemma 5.3.

Lemma 5.3 *For any injection rate r, all the Y_{rem} packets manage to leave the edge f_4 during s_{j+1} time steps.*

We have so far established two (non-trivial) sufficient constraints on r for instability, namely that $r \geq 0.749$ and $r \geq 0.686$. Clearly, taking $r \geq \max\{0.749, 0.686\} = 0.749$ suffices for instability of the network $\mathcal{N}$ in the constructed execution. This concludes our proof. ■

6 Discussion and Directions for Further Research

In this work we have studied preservation patterns for stability properties of several universally stable, greedy contention-resolution protocols in the framework of Adversarial Queueing Theory [3]. We have shown that specific compositions of such protocols are universally stable, while others are unstable for adversary's injection rate $r \in [0.519, 1]$. We also presented a combinatorial construction of a network and an adversary that lead FIFO protocol to instability for any injection rate $r \in [0.519, 1]$; this result significantly improves the current record of 0.8357 on the instability threshold of FIFO, established by Diaz *et al.* [5]. The involved analysis that has been employed to improve this result, together with the corresponding lower bound of 0.519 on instability threshold for compositions of protocols that we showed, suggests that certain compositions of universally stable protocols exhibit probably more unstable behavior than single individual protocols, such as FIFO, that are not universally stable.

Our work opens up the study of the stability and instability properties of heterogeneous communication networks with multiple contention-resolution protocols running on top of them. A fundamental question that arises in this setting is whether there exists a structural explanation of the differences we have observed regarding the stability of different compositions of universally stable protocols. Is there any deep reason for the composition of LIS with another contention-resolution protocol to be unstable? Also, for the unstable compositions of pairs of protocols, can we characterize the graphs on which the composition is unstable? A corresponding characterization for networks on which a *single* greedy protocol is running (as opposed to a composition of greedy protocols) in terms of *graph minors* has been developed in [1, Section 3.2].

References

[1] M. Andrews, B. Awerbuch, A. Fernandez, J. Kleinberg, T. Leighton, and Z. Liu, "Universal Stability Results for Greedy Contention-Resolution Protocols," *Journal of the ACM,* Vol. 48, No. 1, pp. 39-69, January 2001.

[2] M. Andrews, A. Férnandez, A. Goel and L. Zhang, "Source Routing and Scheduling in Packet Networks," *Proceedings of the 42nd Annual IEEE Symposium on Foundations of Computer Science,* October 2002.

[3] A. Borodin, J. Kleinberg, P. Raghavan, M. Sudan and D. Williamson, "Adversarial Queueing Theory," *Journal of the ACM,* Vol. 48, No. 1, pp. 13–38, January 2001.

[4] H. Chen and D. D. Yao, *Fundamentals of Queueing Networks,* Springer, 2000.

[5] J. Diaz, D. Koukopoulos, S. Nikoletseas, M. Serna, P. Spirakis and D. Thilikos, "Stability and Non-Stability of the FIFO Protocol," *Proceedings of the 13th Annual ACM Symposium on Parallel Algorithms and Architectures,* pp. 48–52, July 2001.

[6] A. Férnandez, E. Jiménez and V. Cholvi, "On the Interconnection of Causal Memory Systems," *Proceedings of the 19th Annual ACM Symposium on Principles of Distributed Computing,* pp. 163–170, July 2000.

[7] S. Floyd and V. Paxson, "Difficulties in Simulating the Internet," *IEEE/ACM Transactions on Networking,* Vol. 9, No. 4, pp. 392–403, August 2001.

[8] A. Goel, "Stability of Networks and Protocols in the Adversarial Queueing Model for Packet Routing," *Networks,* Vol. 37, No. 4, pp. 219-224, 2001.

[9] M. P. Herlihy and J. Wing, "Linearizability: A Correctness Condition for Concurrent Objects," *ACM Transactions on Programming Languages and Systems,* Vol. 12, No. 3, pp. 463–492, 1990.

[10] F. P. Kelly, *Reversibility and Stochastic Networks,* Wiley, New York, 1979.

[11] N. Lynch, *Distributed Algorithms,* Morgan Kaufmann, 1996.

[12] J. Mitchell, Email Communication to I. Lee, April 2002.

[13] P. Tsaparas, *Stability in Adversarial Queueing Theory,* M.Sc. Thesis. Department of Computer Science, University of Toronto, 1997.

Appendix

A Proof of Theorem 3.1

We first prove two lemmas.

Lemma A.1 *Let p be a packet waiting in a queue e at time t and suppose there are currently $k-1$ other packets in the system requiring e that have priority over p. Then p will cross e within the next $\frac{k+b}{\epsilon}$ steps if the queueing discipline in queue e is* SIS, NTS *or* FTG.

Proof: Let's assume that p does not cross e in the next $\frac{k+b}{\epsilon}$ steps from the moment it arrives in the queue of edge e. It will be proved that this assumption results in a contradiction. In order for packet p not to cross edge e in the next $\frac{k+b}{\epsilon}$ steps, other packets cross e in these steps (one distinct packet crosses queue e in each step because of the greedy nature of the protocol). These packets must either belong to the set of $k-1$ packets existing in the system at time t requiring edge e that have priority over p or belong to the (at most) $(1-\epsilon)\frac{k+b}{\epsilon}+b$ newly-injected packets (from the definition of bounded adversary [3]) requiring queue e that can be injected in the system during the time period of $\frac{k+b}{\epsilon}$ steps.

Notice that *all* newly-injected packets have priority over packet p if SIS protocol is used. Furthermore, if NTS is used, *in the worst case* new injections are nearest to their source than packet p. Finally if FTG is used, in the *worst case* all new injections have longer paths to cross than packet p. Therefore at most

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

packets have priority over p during the time period of $\frac{k+b}{\epsilon}$ steps. Hence, there is a contradiction. Thus, p will cross e within the next $\frac{k+b}{\epsilon}$ steps. ■

Definition A.1 *We denote an edge, in which a packet p arrives e_j, if it is the j^{th} edge that packet p arrives crossing its path.*

Let's now define a sequence of numbers by the recurrence $k_j=\frac{mk_{j-1}+mb}{\epsilon}$, where $k_1=\frac{mb}{\epsilon}$.

Lemma A.2 *When a packet p crossing its path arrives at the j^{th} edge on its path, there are at most k_j-1 other packets requiring to traverse this edge with priority over p, if the used queueing protocol is a)* SIS *or* NTS, *b)* SIS *or* FTG, *and c)* NTS *or* FTG.

Proof: Part a)

Induction will be used to prove the claim of this lemma for all j. If the queue of edge e_j uses SIS then the claim holds for $j=1$, since for any edge e_j the only packets requiring to traverse it, which initially could have priority over p, are the (at most) $b-1$ packets injected in the same time step as p $(b-1\leq k_1-1)$.

If queue e_j uses NTS then we will prove that the claim holds for $j=1$. Let's define $X_i(t)$ the set of packets in the queue e_j that have crossed less than i edges at time t and let's assume that l_i is (at most) the number of packets in the system that have crossed less than i edges $(i=j)$. Let t be the current time and let $t^{'}$ be the most recent time in which $X_i(t^{'})$ was empty. Any packet in $X_i(t)$ must either have had crossed less than $i-1$ edges at time $t^{'}$ or else it must have been injected after time $t^{'}$. But, at every time step $t^{''}$ between times $t^{'}$ and t a packet from $X_i(t^{''})$ must have crossed edge e_j. Hence,

$$\begin{aligned}|X_i(t)| &\leq l_{i-1}+(t-t^{'})(1-\epsilon)+b-(t-t^{'}) \\ &= l_{i-1}-(t-t^{'})\epsilon+b\end{aligned}$$

The above inequality can be written

$$|X_i(t)|+(t-t^{'})\epsilon \leq l_{i-1}+b$$

But,

$$(t-t^{'})\epsilon \leq |X_i(t)|+(t-t^{'})\epsilon$$

Therefore, we conclude that

$$t-t^{'} \leq \frac{l_{i-1}+b}{\epsilon}$$

This inequality shows an upper bound on the number of packets in queue e_j that have crossed less than i edges. So, this result holds for one queue. However, there are m queues in the network. Thus, the total number of packets in the system that have crossed less than i edges is always at most $\frac{ml_{i-1}+mb}{\epsilon}$.

We claim that, at most $\frac{ml_0+mb}{\epsilon}-1$ packets could have priority over p in queue e_1 when NTS is used because in the queue e_1 packet p hasn't crossed any edge. Therefore, no packet from the old ones have priority over p in queue e_1. So, $l_0=0$. Therefore, the expression $\frac{ml_{i-1}+mb}{\epsilon}$ that has been proved before becomes $\frac{0+mb}{\epsilon}-1 \leq k_1-1$. This is the number of packets that could have priority over p in queue e_1 when NTS is used. Thus, the claim holds for $j=1$ if NTS is used.

Now suppose that the claim holds for some j. Then by Lemma A.1 p will arrive at the tail of e_{j+1} in at most another $\frac{k_j+b}{\epsilon}$ steps, during which at most $(1-\epsilon)\frac{k_j+b}{\epsilon}+b$ packets arrive requiring the edge e_{j+1} with priority over p. These packets along with the k_j-1 packets that had priority over p in the previous step of the induction constitute the set of packets that will have priority over p when it arrives at the tail of e_{j+1}.

So, at most

$$\begin{aligned}k_j-1+(1-\epsilon)\frac{k_j+b}{\epsilon}+b &= \frac{k_j+b}{\epsilon}-1 \\ &\leq \frac{mk_j+mb}{\epsilon}-1=k_{j+1}-1\end{aligned}$$

packets requiring queue e_{j+1} have priority over p and hence the claim holds.

Part b)

We will employ induction to prove the claim of this lemma for all j. If the j^{th} edge e_j on a packet's p path uses SIS protocol then the claim holds for $j=1$, since for any edge the only packets requiring to traverse it, which initially could have priority over p, are the (at most) $b-1$ packets injected in the same time step as p ($b-1 \leq k_1-1$).

If the j^{th} edge e_j on a packet's p path uses FTG as protocol we will prove that the claim holds for $j=1$. Let's define as $X_i(t)$ the set of packets in the queue e_j that still have to cross at least i edges at time t and let's assume that l_i is (at most) the number of packets in the system that still have to cross at least i edges. Let's t be the current time and let $t^{'}$ be the most recent time in which $X_i(t^{'})$ was empty. Any packet in $X_i(t)$ must either have had at least $i+1$ edges to cross at time $t^{'}$ or else it must have been injected after time $t^{'}$. But, at every step $t^{''}$ between times $t^{'}$ and t a packet from $X_i(t^{''})$ must have crossed edge e_j. Hence,

$$|X_i(t)| \leq l_{i+1} + (t - t^{'})(1 - \epsilon) + b - (t - t^{'}) = l_{i+1} - (t - t^{'})\epsilon + b$$

The above inequality can be written

$$|X_i(t)| + (t - t^{'})\epsilon \leq l_{i+1} + b$$

But,

$$(t - t^{'})\epsilon \leq |X_i(t)| + (t - t^{'})\epsilon$$

Therefore, we conclude that

$$t - t^{'} \leq \frac{l_{i+1} + b}{\epsilon}$$

This inequality shows an upper bound on the number of packets in queue e_j that have to cross i or more edges. So, this result holds for one queue. However, there are m queues in the network. Thus, the total number of packets in the system that have to cross i or more edges is always at most $\frac{ml_{i+1}+mb}{\epsilon}$.

In the case of $j = 1$ and FTG as used protocol, we claim that $k_1 = l_d$ because in the queue e_1, that has distance $d - 1$ from the final edge on its path in the worst case, packet p will have to cross d edges. Therefore, only packets that have to cross $d + 1$ edges have priority over p in queue e_1. But, no packet has to cross $d + 1$ edges because d is the length of the longest path in the network. So, $l_{d+1} = 0$. Therefore, the expression $\frac{ml_{i+1}+mb}{\epsilon}$ that has been proved before becomes $\frac{0+mb}{\epsilon} - 1 = k_1 - 1$. Thus, $k_1 - 1$ packets could have priority over p in queue e_d when FTG is used. Thus, the claim holds for $j = 1$ if FTG is used.

Now suppose that the claim holds for some j. Then by Lemma A.1, p will arrive at the tail of e_{j+1} on its path in at most another $\frac{k_j+b}{\epsilon}$ steps, during which at most $(1 - \epsilon)\frac{k_j+b}{\epsilon} + b$ packets requiring edge e_{j+1} arrive with priority over p. These packets along with the $k_j - 1$ packets that had priority over p in the previous step of the induction constitute the set of packets that will have priority over p when it arrives at the tail of e_{j+1}. So, at most

$$k_j - 1 + (1 - \epsilon)\frac{k_j + b}{\epsilon} + b = \frac{k_j + b}{\epsilon} - 1 \leq \frac{mk_j + mb}{\epsilon} - 1 = k_{j+1} - 1$$

packets requiring queue e_{j+1} have priority over p and hence the claim holds.

Part c)

Induction will be used to prove the claim of this lemma for all j. If queue e_j uses NTS then we will prove that the claim holds for $j = 1$. Let's define as $X_i(t)$ the set of packets in the queue e_j that have crossed less than i edges at time t and let's assume that l_i is (at most at any time) the number of packets in the system that have crossed less than i edges ($i = j$). Let t be the current time and let $t^{'}$ be the most recent time in which $X_i(t^{'})$ was empty. Any packet in $X_i(t)$ must either have had crossed less than $i - 1$ edges at time $t^{'}$ or else it must have been injected after time $t^{'}$. But, at every step $t^{''}$ between times $t^{'}$ and t a packet from $X_i(t^{''})$ must have crossed edge e_j. Hence,

$$\begin{aligned} |X_i(t)| &\leq l_{i-1} + (t-t^{'})(1-\epsilon) + b - (t-t^{'}) \\ &= l_{i-1} - (t-t^{'})\epsilon + b \end{aligned}$$

The above inequality can be written

$$|X_i(t)| + (t-t^{'})\epsilon \leq l_{i-1} + b$$

But,

$$(t-t^{'})\epsilon \leq |X_i(t)| + (t-t^{'})\epsilon$$

Therefore, we conclude that

$$t - t^{'} \leq \frac{l_{i-1}+b}{\epsilon}$$

This inequality shows an upper bound on the number of packets in queue e_j that have crossed less than i edges. So, this result holds for one queue. However, there are m queues in the network. Thus, the total number of packets in the system that have crossed less than i edges is always at most $\frac{ml_{i-1}+mb}{\epsilon}$. We claim that, at most $\frac{ml_0+mb}{\epsilon} - 1$ packets could have priority over p in queue e_1, when NTS is used, because in the queue e_1 packet p hasn't crossed any edge. Therefore, no packet from the old ones have priority over p in queue e_1. So, $l_0 = 0$. Therefore, the expression $\frac{ml_{i-1}+mb}{\epsilon}$ that has been proved before becomes $\frac{0+mb}{\epsilon} - 1 \leq k_1 - 1$. This is the number of packets that could have priority over p in queue e_1 when NTS is used. Thus, the claim holds for $j = 1$ if NTS is used.

If queue e_j uses FTG then we will prove that the claim holds for $j = 1$. Let's define as $L_h(t)$ the set of packets in the queue e_j that still have to cross at least h edges at time t and let's assume that l_h is (at most) the number of packets in the system that still have to cross at least h edges. Let t be the current time and let $t^{'}$ be the most recent time in which $L_h(t^{'})$ was empty. Any packet in $L_h(t)$ must either have had at least $h+1$ edges to cross at time $t^{'}$ or else it must have been injected after time $t^{'}$. But, at every step $t^{''}$ between times $t^{'}$ and t a packet from $L_h(t^{''})$ must have crossed edge e_j. Hence,

$$\begin{aligned} |L_h(t)| &\leq l_{h+1} + (t-t^{'})(1-\epsilon) + b - (t-t^{'}) \\ &= l_{h+1} - (t-t^{'})\epsilon + b \end{aligned}$$

The above inequality can be written

$$|L_h(t)| + (t-t^{'})\epsilon \leq l_{h+1} + b$$

But,

$$(t-t^{'})\epsilon \leq |L_h(t)| + (t-t^{'})\epsilon$$

Therefore, we conclude that

$$t - t^{'} \leq \frac{l_{h+1}+b}{\epsilon}$$

This inequality shows an upper bound on the number of packets in queue e_j that have to cross h or more edges. So, this result holds for one queue. However, there are m queues in the network. Thus, the total number of packets in the system that have to cross h or more edges is always at most $\frac{ml_{h+1}+mb}{\epsilon}$.

In the case of $j = 1$ and FTG as used protocol, we claim that $k_1 = l_d$ because in the queue e_1 packet p will have to cross in the worst case d edges. Therefore, only packets that have to cross $d + 1$ edges have priority over p in queue e_1. But, no packet have to cross $d+1$ edges because d is the length of the longest path in the network. So, $l_{d+1} = 0$. Therefore, the expression $\frac{ml_{h+1}+mb}{\epsilon}$ that has been proved before becomes $\frac{0+mb}{\epsilon} - 1 = k_1 - 1$ packets could have priority over p in queue e_1 when FTG is used. Thus, the claim holds for $j = 1$ if FTG is used.

Now suppose that the claim holds for some j. Then by Lemma A.1 p will arrive at the tail of e_{j+1} in at most another $\frac{k_j+b}{\epsilon}$ steps, during which at most $(1-\epsilon)\frac{k_j+b}{\epsilon} + b$ packets requiring edge e_{j+1} arrive with priority over p. These packets along with the $k_j - 1$ packets that had priority over p in the previous step of the induction constitute the set of packets that will have priority over p when it arrives at the tail of e_{j+1}. So, at most

$$k_j - 1 + (1-\epsilon)\frac{k_j + b}{\epsilon} + b = \frac{k_j + b}{\epsilon} - 1 \leq \frac{mk_j + mb}{\epsilon} - 1 = k_{j+1} - 1$$

packets requiring queue e_{j+1} have priority over p and hence the claim holds. ■

We continue with the proof of Theorem 3.1.

a) Assume there are $k_d + 1$ packets at some time all requiring the same edge. Then, the packet with the lowest priority of the $k_d + 1$ packets contradicts the claim of Lemma A.2. Combining both lemmas, a packet p takes at most $\frac{k_j+b}{\epsilon}$ steps to cross the edge e_j, once it is in the queue of this edge. Therefore, the delay bound is $D = \frac{db+\sum_{i=1}^{d} k_i}{\epsilon}$. No packet spends more than D steps in the system.

b) Assume there are $k_d + 1$ packets at some time all requiring the same edge. Then, the packet with the lowest priority of the $k_d + 1$ packets contradicts the claim of Lemma A.2. Combining both lemmas, a packet p takes at most $\frac{k_j+b}{\epsilon}$ steps to cross the e_j edge, once it is in the queue of this edge.

c) Assume there are $k_d + 1$ packets at some time all requiring the same edge. Then, the packet with the lowest priority of the $k_d + 1$ packets contradicts the claim of Lemma A.2. Combining both lemmas, a packet p takes at most $\frac{k_j+b}{\epsilon}$ steps to cross the j^{th} edge on its path, once it is in the queue of this edge.

B Proof of Theorem 4.1

Part a)

Inductive Hypothesis: At the beginning of phase j, there are s_j packets that are queued in the queues $f_4^{'}, f_7^{'}, f_8^{'}$ (in total) requiring to traverse the edges e_0, f_1, f_2, f_4, f_7.

Induction Step: At the beginning of phase $j + 1$ there will be more than s_j packets that will be queued in the queues f_4, f_7, f_8, requiring to traverse the edges $e_1, f_1^{'}, f_2^{'}, f_4^{'}, f_7^{'}$.

Consider first the network N_1 in Figure 1. All the queues of the network use LIS protocol except the queues that correspond to the edges $f_1, f_1^{'}, f_7, f_7^{'}, f_8, f_8^{'}$ that use SIS protocol. Moreover, we should mention that the edges $f_9, f_9^{'}$ can use any protocol of LIS or SIS because the packets that want to traverse them in the last round of each phase are blocked in their previous edge on their path (let P be any of these protocols). Thus, there is no need to use a specific protocol at queues $f_9, f_9^{'}$ for blocking these packets with a suitable injection.

The intuition behind this adversary construction and network topology is basically the preservation of the newly injected investing flows, in each round of a phase, inside the network until the end of the phase during which they have been injected. The tools we use to achieve this goal are the injections of short intermediate flows ($S_{i,k}$) and the introduction of parallel edges (f_4, f_5, f_6 and f_7, f_8, f_9) in the network topology. Although the short intermediate flows do not contribute actually to the number of packets that will be used as the initial flow for the next phase, their role is essential as they are used for blocking investing flows of packets in consecutive rounds. Moreover, some of them have another role, too. They block short intermediate flows that are injected into the system at certain rounds in order to be used for blocking investing flows in the next round of the round in which they have been injected.

Also, an important point we should mention is the use of the short intermediate flow that is injected in the first round of a phase j ($S_{1,1} - flow$) to block a portion of the initial packets that are in the system at the beginning of the phase (s_j) in order these packets to be used twice for blocking investing flow X_1 in the next round. As far as concerns the parallel edges, the purpose of their presence in the network topology is to be guaranteed that the paths of the packet flows injected in the same round do not overlap. The first set of parallel edges f_4, f_5, f_6 is used in order short intermediate flows that are injected at the same round to not overlap, while the second set of parallel edges f_7, f_8, f_9 is used mainly in order the paths of investing flows and the paths of short intermediate flows that are injected at the same round to not overlap.

We will construct an adversary A such that the induction step will hold. Proving that the induction step holds, we ensure that the inductive hypothesis will hold at the beginning of phase $j+1$ for the symmetric edges with an increased value of s_j packets, $s_{j+1} > s_j$. In order to prove that the induction step works, we should consider that there is a large enough number of packets s_j in the initial system configuration.

During phase j the adversary plays four rounds of injections. The sequence of injections is as follows:

Round 1: It lasts s_j time steps.

At the beginning of this round, there are s_j packets ($S - flow$) in the queues f_4', f_7', f_8' (in total) requiring to traverse the edges e_0, f_1, f_2, f_4, f_7.

Adversary's behavior. During this round the adversary injects a set X_1 of $|X_1| = rs_j$ packets in queue e_0 wanting to traverse the edges $e_0, f_3, f_4, f_7, e_1, f_1', f_2', f_4', f_7'$. At the same time, the adversary injects a set $S_{1,1}$ of $|S_{1,1}| = rs_j$ packets in queue f_1 that require to traverse only the edge f_1.

Evolution of the system configuration. The set X_1 of packets, that are injected into the system during this round, are blocked in queue e_0 by the $S - flow$ packets because e_0 uses LIS as protocol and therefore for every arrival of an injected X_1 packet in queue e_0, there is at least one $S - flow$ packet there that has been in the system for a longer time, so it has priority. At the same time, the $S - flow$ packets are delayed in queue f_1 by the set $S_{1,1}$ of packets that are injected into the system during this round and require to traverse only the edge f_1. This happens because queue f_1 uses SIS protocol and $S_{1,1} - flow$ has been in the system for a shorter time than $S - flow$. Notice that because of SIS protocol, when a packet from $S_{1,1} - flow$ is injected in queue f_1, it has priority over all packets from $S - flow$ that are queued in f_1. Thus, all the $S_{1,1}$ packets traverse the edge f_1 along with some packets from $S - flow$, while the rest of $S - flow$ packets remain in queue f_1. The number of $S - flow$ packets that traverse the edge f_1 is $s_j - rs_j$.

At the end of this round, there are rs_j packets of $S - flow$ in queue f_1 wanting to traverse the edges f_1, f_2, f_4, f_7. Also, there is a set X_1 of $|X_1| = rs_j$ packets in queue e_0 wanting to traverse the edges $e_0, f_3, f_4, f_7, e_1, f_1', f_2', f_4', f_7'$.

Round 2: It lasts rs_j time steps.

Adversary's behavior. During this round the adversary injects a set X_2 of $|X_2| = r^2 s_j$ packets in queue e_0 requiring to traverse the edges $e_0, f_3, f_4, f_8, e_1, f_1^{'}, f_2^{'}, f_4^{'}, f_7^{'}$. In addition, the adversary injects a set $S_{2,1}$ of $|S_{2,1}| = r^2 s_j$ packets in queue f_2 wanting to traverse the edges f_2, f_5, f_7.

Evolution of the system configuration. During the rs_j time steps of this round, X_2 packets, that are injected into the system during this round, are blocked in queue e_0 that uses LIS protocol by the $X_1 - flow$ packets that have been in the system for a longer time. At the same time, all $X_1 - flow$ packets traverse the edges e_0, f_3, but they are blocked in f_4 that uses LIS protocol because of the remaining rs_j packets of $S - flow$ in queue f_1 at the end of the previous round. The remaining packets of $S - flow$ in queue f_1 want to traverse the edges f_1, f_2, f_4, f_7 and they have priority over the packets of $X_1 - flow$ because they have been in the system for a longer time. At the end of this round all the remaining rs_j packets of $S - flow$ arrive at the queue of the edge f_7, where they are absorbed after they traverse it. In addition, $S_{2,1}$ packets, that are injected into the system during this round, are blocked in queue f_2 that uses LIS protocol by the remaining rs_j packets of $S - flow$ that have been in the system for a longer time and consequently they have priority over $S_{2,1} - flow$ packets.

At the end of this round, there are rs_j packets of $X_1 - flow$ in queue f_4 wanting to traverse the edges $f_4, f_7, e_1, f_1^{'}, f_2^{'}, f_4^{'}, f_7^{'}$. Also, there are $r^2 s_j$ packets of $X_2 - flow$ in queue e_0 requiring to traverse the edges $e_0, f_3, f_4, f_8, e_1, f_1^{'}, f_2^{'}, f_4^{'}, f_7^{'}$ and $r^2 s_j$ packets of $S_{2,1} - flow$ in queue f_2 wanting to traverse the edges f_2, f_5, f_7.

Round 3: It lasts $r^2 s_j$ time steps.

Adversary's behavior. During this round the adversary injects a set X_3 of $|X_3| = r^3 s_j$ packets in queue f_4 wanting to traverse the edges $f_4, f_9, e_1, f_1^{'}, f_2^{'}, f_4^{'}, f_7^{'}$. Also, the adversary injects a set $S_{3,1}$ of $|S_{3,1}| = r^3 s_j$ packets in queue f_5 wanting to traverse the edges f_5, f_7. Finally, the adversary injects a set $S_{3,2}$ of $|S_{3,2}| = r^3 s_j$ packets in queue f_2 wanting to traverse the edges f_2, f_6, f_8.

Evolution of the system configuration. During this round, a portion of the $X_1 - flow$ packets $(r^2 s_j)$ traverse the edge f_4, while the rest $rs_j - r^2 s_j$ packets remain in queue f_4. The packets of $X_1 - flow$ that traverse the edge f_4 are blocked in queue f_7 that uses SIS protocol by the $S_{2,1} - flow$ packets that have been injected later than $X_1 - flow$ into the system, were blocked in queue f_2 at the end of the previous round and traverse now the edges f_2, f_5 arriving at the queue of the edge f_7 where they are absorbed after they traverse it. Furthermore, during this round the $X_2 - flow$ packets, that have been injected into the system during the previous round, traverse the edges e_0, f_3 and they are blocked in queue f_4 that uses LIS protocol by $X_1 - flow$ packets that have arrived there earlier and consequently they have been in the system for a longer time than $X_2 - flow$ packets.

Furthermore, X_3 packets that are injected into the system during this round are blocked in queue f_4 that uses LIS protocol by the $X_2 - flow$ packets that have been in the system for a longer time than $X_3 - flow$ packets. Also, the set $S_{3,1}$ of packets, that are injected into the system during this round, are blocked in queue f_5 that uses LIS protocol by $S_{2,1} - flow$ packets that have been in the system for a longer time than $S_{3,1} - flow$ packets. Finally, the set $S_{3,2}$ of packets, that are injected into the system during this round, are blocked in queue f_2 that uses LIS protocol by the $S_{2,1} - flow$ packets that have been in the system for a longer time than $S_{3,2} - flow$ packets.

At the end of this round, there are rs_j packets of $X_1 - flow$ in queues f_4, f_7 in total ($rs_j - r^2 s_j$ packets in queue f_4 and rs_j packets in queue f_7) wanting to traverse the edges $f_4, f_7, e_1, f_1^{'}, f_2^{'}, f_4^{'}, f_7^{'}$. Also, in queue f_4 there are $r^2 s_j$ packets of $X_2 - flow$ requiring to traverse the edges $f_4, f_8, e_1, f_1^{'}, f_2^{'}, f_4^{'}, f_7^{'}$ and $r^3 s_j$ packets of $X_3 - flow$ wanting to traverse the edges $f_4, f_9, e_1, f_1^{'}, f_2^{'}, f_4^{'}, f_7^{'}$. Moreover, there are $r^3 s_j$ packets of $S_{3,1} - flow$ in queue f_5 requiring to traverse the edges f_5, f_7 and $r^3 s_j$ packets of $S_{3,2} - flow$ in queue f_2 requiring to traverse the edges f_2, f_6, f_8.

Round 4: It lasts $r^3 s_j$ time steps.

Adversary's behavior. During this round, the adversary injects a set X_4 of $|X_4| = r^4 s_j$ packets in queue f_3 wanting to traverse the edges $f_3, f_4, f_7, e_1, f_1^{'}, f_2^{'}, f_4^{'}, f_7^{'}$. Besides the short intermediate flows $(S_{3,1}, S_{3,2})$ that have been injected into the system during the previous round and still remain into the system at the beginning of this round, the system configuration at the beginning of round 4 also consists of the investing flows X_1, X_2, X_3 that have been injected into the system during the previous rounds. During this round, these investing flows along with the investing flow that is injected into the system during this round $(X_4 - flow)$ continue to remain into the system independently of the adversarial injection rate r because they are blocked by the short intermediate flows $S_{3,1}, S_{3,2}$.

Evolution of the system configuration. We can distinguish two cases. In the first case, the injection rate r is such that all the $X_1 - flow$ packets $(rs_j - r^2 s_j)$ that were queued in the queue of the edge f_4 traverse it and they are queued with the rest $X - 1 - flow$ packets in queue f_7 where they continue to be blocked by $S_{3,1} - flow$ packets that were in queue f_5 at the end of the previous round and traverse during this round the edges f_5, f_7. These packets have been in the system for a shorter time than $X - 1 - flow$ packets and thus they have priority in edge f_7 that uses SIS protocol. As far as concerns the other "useful flows" injected in previous rounds (X_2 and X_3), if there are remaining time steps until the end of round 4 (after the departure of all $X_1 - flow$ packets that were queued in f_4 at the end of the previous round), then a portion of $X_2 - flow$ packets traverse the edge f_4, while the rest packets of $X_2 - flow$ remain in queue f_4. The portion of the $X_2 - flow$ packets that traverse the edge f_4 are blocked in queue f_8 that uses SIS protocol by the $S_{3,2} - flow$ packets that traverse the edges f_2, f_6, f_8, which have arrived later than $X_2 - flow$ into the system. Also, the $X_3 - flow$ packets remain blocked by $X_2 - flow$ in queue f_4. In the second case the injection rate r is such that a portion of the $X_1 - flow$ packets, that were queued in the queue of the edge f_4, traverse it and queued with the rest $X - 1 - flow$ packets in queue f_7 where they continue to be blocked by the $S_{3,1} - flow$ packets for the same reason as in the previous case. Then, $X_2 - flow$ and $X_3 - flow$ packets remain blocked in queue f_4 by $X_1 - flow$ packets.

Therefore, we have shown that independently of the adversarial injection rate the investing flows X_1, X_2, X_3 continue to remain into the system at the end of round 4. Finally, during this round, the set X_4 of packets, that are injected into the system during this round, are blocked in queue f_4 that uses LIS protocol by $X_3 - flow$ packets that have arrived there earlier than $X_4 - flow$ packets.

At the end of this round, the number of packets in queues f_4, f_7, f_8 requiring to traverse the edges $e_1, f_1^{'}, f_2^{'}, f_4^{'}, f_7^{'}$ is

$$s_{j+1} = |X_1| + |X_2| + |X_3| + |X_4| = rs_j + r^2 s_j + r^3 s_j + r^4 s_j$$

In order to have instability, we must have $s_{j+1} > s_j$. Thus,

$$\begin{array}{lcl}
s_{j+1} & > & s_j \\
|X_1| + |X_2| + |X_3| + |X_4| & > & s_j \\
r^3 s_j + r^2 s_j + \frac{r^2 s_j}{r+1} + \frac{r^3 s_j + r s_j}{(r+1)(r^2+r+2)} + \frac{r^4 s_j + r^2 s_j}{(r^2+r+2)(r^3+r^2+2r+2)} - r^2 s_j r s_j + r^2 s_j + r^3 s_j + r^4 s_j & > & s_j \\
r & \geq & 0.519
\end{array}$$

Therefore, the inequality $s_{j+1} > s_j$ holds for $r \geq 0.519$. This argument can be repeated for an infinite number of phases ensuring that the number of packets at the end of a phase will be bigger than at the beginning of the phase.

Part b)

Inductive Hypothesis: At the beginning of phase j, there are s_j packets that are queued in the queues $f_4^{'}, f_7^{'}, f_8^{'}$ (in total) requiring to traverse the edges e_0, f_1, f_2, f_4, f_7.

Induction Step: At the beginning of phase $j + 1$ there will be more than s_j packets that will be queued in the queues f_4, f_7, f_8, requiring to traverse the edges $e_1, f_1^{'}, f_2^{'}, f_4^{'}, f_7^{'}$.

The network N_2 that is used here is the same with the network in Figure 1. The only difference is that in this case the queues that correspond to the edges $f_1, f_1^{'}, f_7, f_7^{'}, f_8, f_8^{'}$ use NTS protocol instead of SIS protocol. Also all the other queues of the network use LIS protocol except the queues that correspond to the edges $f_9, f_9^{'}$ that can use any protocol of LIS or NTS because the packets that want to traverse them in the last round of each phase are blocked in their previous edge on their path (let P be any of these protocols). Thus, there is no need to use a specific protocol at queues $f_9, f_9^{'}$ for blocking these packets with a suitable injection.

The intuition behind this adversary construction and network topology is basically the preservation of the newly injected investing flows in each round of a phase inside the network until the end of the phase during which they have been injected. The tools we use to achieve this goal are the injections of short intermediate flows ($S_{i,k}$) and the introduction of parallel edges (f_4, f_5, f_6 and f_7, f_8, f_9) in the network topology. Although the short intermediate flows do not contribute actually to the number of packets that will be used as the initial flow for the next phase, their role is essential as they are used for blocking investing flows of packets in consecutive rounds. Moreover, some of them have another role, too. They block short intermediate flows that are injected in certain rounds in order to be used for blocking investing flows in the next round of the round in which they have been injected.

Also, an important point we should mention is the use of the short intermediate flow that is injected in the first round of a phase j ($S_{1,1} - flow$) to block a portion of the initial packets that are in the system at the beginning of the phase (s_j) in order these packets to be used twice for blocking investing flow X_1 in the next round. As far as concerns the parallel edges, the purpose of their presence in the network topology is to be guaranteed that the paths of the packet flows that are injected in the same round do not overlap. The first set of parallel edges f_4, f_5, f_6 is used in order short intermediate flows that are injected at the same round to not overlap, while the second set of parallel edges f_7, f_8, f_9 is used mainly in order the paths of investing flows and the paths of short intermediate flows that are injected at the same round to not overlap.

We will construct an adversary A such that the induction step will hold. Proving that the induction step holds, we ensure that the inductive hypothesis will hold at the beginning of phase $j+1$ for the symmetric edges with an increased value of s_j, $s_{j+1} > s_j$. In order to prove that the induction step works, we should consider that there is a large enough number of packets s_j in the initial system configuration.

During phase j the adversary plays four rounds of injections. The sequence of injections is as follows:

Round 1: It lasts s_j time steps.

Adversary's behavior. During this round the adversary injects a set X_1 of $|X_1| = rs_j$ packets in queue e_0 wanting to traverse the edges $e_0, f_3, f_4, f_7, e_1, f_1^{'}, f_2^{'}, f_4^{'}, f_7^{'}$. At the same time, the adversary injects a set $S_{1,1}$ of $|S_{1,1}| = rs_j$ packets in queue f_1 that require to traverse only the edge f_1.

At the beginning of this round, there are s_j packets ($S - flow$) in the queues $f_4^{'}, f_7^{'}, f_8^{'}$ (in total) requiring to traverse the edges e_0, f_1, f_2, f_4, f_7.

Evolution of the system configuration. The packets of set X_1 that are injected into the system during this round are blocked by the $S - flow$ in queue e_0 because e_0 uses LIS protocol and therefore for every arrival of an injected X_1 packet in queue e_0, there is at least one $S - flow$ packet there, that is longer time in the system so it has priority. At the same time, the $S - flow$ is delayed by the packets of set $S_{1,1}$ that are injected into the system during this round. This happens because queue f_1 uses NTS protocol and $S_{1,1} - flow$ packets are nearest to their source than $S - flow$ packets. Notice that because of NTS protocol, when a packet from $S_{1,1} - flow$ is injected in queue f_1, it has priority over all packets from $S - flow$ that are queued in f_1. Thus, all the $S_{1,1}$ packets traverse the edge f_1 along with some packets from $S - flow$, while the rest of $S - flow$ packets remain in queue f_1. The number of $S - flow$ packets that traverse the edge f_1 is $s_j - rs_j$.

At the end of this round, there are rs_j packets of $S - flow$ in queue f_1 wanting to traverse the edges f_1, f_2, f_4, f_7. Also, there is a set X_1 of $|X_1| = rs_j$ packets in queue e_0 wanting to traverse the edges $e_0, f_3, f_4, f_7, e_1, f_1^{'}, f_2^{'}, f_4^{'}, f_7^{'}$.

Round 2: It lasts rs_j time steps.

Adversary's behavior. During this round the adversary injects a set X_2 of $|X_2| = r^2 s_j$ packets in queue e_0 requiring to traverse the edges $e_0, f_3, f_4, f_8, e_1, f_1^{'}, f_2^{'}, f_4^{'}, f_7^{'}$. In addition, the adversary injects a set $S_{2,1}$ of $|S_{2,1}| = r^2 s_j$ packets in queue f_2 wanting to traverse the edges f_2, f_5, f_7.

Evolution of the system configuration. The packets of the set X_2, that are injected into the system during this round, are blocked in queue e_0 that uses LIS by $X_1 - flow$ packets that have been in the system for a longer time. At the same time, all $X_1 - flow$ packets traverse the edges e_0, f_3, but they are blocked in f_4 that uses LIS by the remaining rs_j packets of $S - flow$ in queue f_1 at the end of the previous round. The remaining packets of $S - flow$ want to traverse the edges f_1, f_2, f_4, f_7 and have priority over the packets of $X_1 - flow$ because they have been in the system for a longer time. At the end of this round all the remaining rs_j packets of $S - flow$ arrive at edge f_7, where they are absorbed after they traverse it. In addition, the packets of the set $S_{2,1}$, that are injected into the system during this round, are blocked in queue f_2 that uses LIS protocol by the remaining rs_j packets of $S - flow$ that have been in the system for a longer time and consequently they have priority over $S_{2,1} - flow$ packets.

At the end of this round, there are rs_j packets of $X_1 - flow$ in queue f_4 wanting to traverse the edges $f_4, f_7, e_1, f_1^{'}, f_2^{'}, f_4^{'}, f_7^{'}$. Also, there are $r^2 s_j$ packets of $X_2 - flow$ in queue e_0 requiring to traverse the edges $e_0, f_3, f_4, f_8, e_1, f_1^{'}, f_2^{'}, f_4^{'}, f_7^{'}$ and $r^2 s_j$ packets of $S_{2,1} - flow$ in queue f_2 wanting to traverse the edges f_2, f_5, f_7.

Round 3: It lasts $r^2 s_j$ time steps.

Adversary's behavior. During this round the adversary injects a set X_3 of $|X_3| = r^3 s_j$ packets in queue f_4 wanting to traverse the edges $f_4, f_9, e_1, f_1^{'}, f_2^{'}, f_4^{'}, f_7^{'}$. Also, the adversary injects a set $S_{3,1}$ of $|S_{3,1}| = r^3 s_j$ packets in queue f_5 wanting to traverse the edges f_5, f_7 and a set $S_{3,2}$ of $|S_{3,2}| = r^3 s_j$ packets in queue f_2 wanting to traverse the edges f_2, f_6, f_8.

Evolution of the system configuration. During this round, a portion of the $X_1 - flow$ packets $(r^2 s_j)$ traverse the edge f_4, while the rest $rs_j - r^2 s_j$ packets remain in queue f_4. The packets of $X_1 - flow$ that traverse the edge f_4 are blocked in queue f_7 that uses NTS by the $S_{2,1} - flow$ packets that are nearest to their source than $X_1 - flow$ packets, were blocked in queue f_2 at the end of the previous round and traverse now the edges f_2, f_5 arriving at edge f_7 where they are absorbed after they traverse it. Furthermore, during this round the $X_2 - flow$ packets, that have been injected into the system during the previous round, traverse the edges e_0, f_3 and they are blocked in queue f_4 that uses LIS protocol by $X_1 - flow$ packets that have arrived there earlier and consequently they have been in the system for a longer time than $X_2 - flow$ packets.

As far as concerns X_3 packets that are injected into the system during this round they are blocked in queue f_4 that uses LIS protocol by $X_2 - flow$ packets that have been in the system for a longer time than $X_3 - flow$ packets. Also, $S_{3,1}$ packets, that are injected into the system during this round, are blocked in queue f_5 that uses LIS protocol by $S_{2,1} - flow$ packets that have been in the system for a longer time than $S_{3,1} - flow$ packets. Finally, $S_{3,2}$ packets, that are injected into the system during this round, are blocked in queue f_2 that uses LIS protocol by the $S_{2,1} - flow$ packets that have been in the system for a longer time than $S_{3,2} - flow$ packets.

At the end of this round, there are rs_j packets of $X_1 - flow$ in queues f_4, f_7 in total ($rs_j - r^2 s_j$ packets in queue f_4 and rs_j packets in queue f_7) wanting to traverse the edges $f_4, f_7, e_1, f_1^{'}, f_2^{'}, f_4^{'}, f_7^{'}$. Also, in queue f_4 there are $r^2 s_j$ packets of $X_2 - flow$ requiring to traverse the edges $f_4, f_8, e_1, f_1^{'}, f_2^{'}, f_4^{'}, f_7^{'}$ and

$r^3 s_j$ packets of $X_3 - flow$ wanting to traverse the edges $f_4, f_9, e_1, f_1^{'}, f_2^{'}, f_4^{'}, f_7^{'}$. Moreover, there are $r^3 s_j$ packets of $S_{3,1} - flow$ in queue f_5 requiring to traverse the edges f_5, f_7 and $r^3 s_j$ packets of $S_{3,2} - flow$ in queue f_2 requiring to traverse the edges f_2, f_6, f_8.

Round 4: It lasts $r^3 s_j$ time steps.

Adversary's behavior. During this round the adversary injects a set X_4 of $|X_4| = r^4 s_j$ packets in queue f_3 wanting to traverse the edges $f_3, f_4, f_7, e_1,\ f_1^{'}, f_2^{'}, f_4^{'}, f_7^{'}$. Besides the short intermediate flows $(S_{3,1}, S_{3,2})$ that have been injected into the system during the previous round and still remain into the system at the beginning of this round, the system configuration at the beginning of round 4 also consists of the investing flows X_1, X_2, X_3 that have been injected into the system during the previous rounds. During this round, these investing flows along with the investing flow that is injected into the system during this round ($X_4 - flow$) continue to remain into the system independently of the adversarial injection rate r because they are blocked by the short intermediate flows $S_{3,1}, S_{3,2}$.

Evolution of the system configuration. We can distinguish two cases. In the first case, the injection rate r is such that all the $X_1 - flow$ packets $(rs_j - r^2 s_j)$ that were queued in the queue of the edge f_4 traverse it and they are queued with the rest $X - 1 - flow$ packets in queue f_7 where they continue to be blocked by $S_{3,1} - flow$ packets that were in queue f_5 at the end of the previous round and traverse during this round the edges f_5, f_7. These packets are nearest to their source than $X_1 - flow$ packets and thus they have priority in edge f_5 that uses NTS protocol. As far as concerns the other investing flows that have been injected into the system in previous rounds (X_2 and X_3), if there are remaining time steps until the end of round 4 (after the departure of all $X_1 - flow$ packets that were queued in f_4 at the end of the previous round), then a portion of $X_2 - flow$ packets traverse the edge f_4, while the rest packets of $X_2 - flow$ remain in queue f_4. The portion of the $X_2 - flow$ packets that traverse the edge f_4 are blocked in queue f_8 that uses NTS protocol by $S_{3,2} - flow$ packets that traverse the edges f_2, f_6, f_8 because the $S_{3,2} - flow$ packets are nearest to their source than $X_2 - flow$ packets and consequently they have priority. Also, $X_3 - flow$ packets remain blocked by $X_2 - flow$ packets in queue f_4. In the second case the injection rate r is such that a portion of the $X_1 - flow$ packets that were queued in the queue of the edge f_4 at the end of round 3 traverse it and they are queued with the rest $X - 1 - flow$ packets in queue f_7 where they continue to be blocked by $S_{3,1} - flow$ packets for the same reason as in the previous case. Also, $X_2 - flow$ and $X_3 - flow$ packets remain blocked in queue f_4 by $X_1 - flow$ packets.

Therefore, we have shown that independently of the adversarial injection rate the investing flows X_1, X_2, X_3 continue to remain into the system at the end of round 4. Finally, the set X_4 of packets that are injected into the system during this round are blocked in queue f_4 that uses LIS protocol by the $X_3 - flow$ packets that have arrived there earlier than $X_4 - flow$ packets.

At the end of this round, the number of packets in queues f_4, f_7, f_8 requiring to traverse the edges $e_1, f_1^{'}, f_2^{'}, f_4^{'}, f_7^{'}$ is

$$s_{j+1} = |X_1| + |X_2| + |X_3| + |X_4| = rs_j + r^2 s_j + r^3 s_j + r^4 s_j$$

In order to have instability, we must have $s_{j+1} > s_j$. Thus,

$$\begin{array}{lcl}
s_{j+1} & > & s_j \\
|X_1| + |X_2| + |X_3| + |X_4| & > & s_j \\
r^3 s_j + r^2 s_j + \frac{r^2 s_j}{r+1} + \frac{r^3 s_j + r s_j}{(r+1)(r^2+r+2)} + \frac{r^4 s_j + r^2 s_j}{(r^2+r+2)(r^3+r^2+2r+2)} - r^2 s_j r s_j + r^2 s_j + r^3 s_j + r^4 s_j & > & s_j \\
r & \geq & 0.519
\end{array}$$

Therefore, the inequality $s_{j+1} > s_j$ holds for $r \geq 0.519$. This argument can be repeated for an infinite number of phases ensuring that the number of packets at the end of a phase will be bigger than at the beginning of the phase.

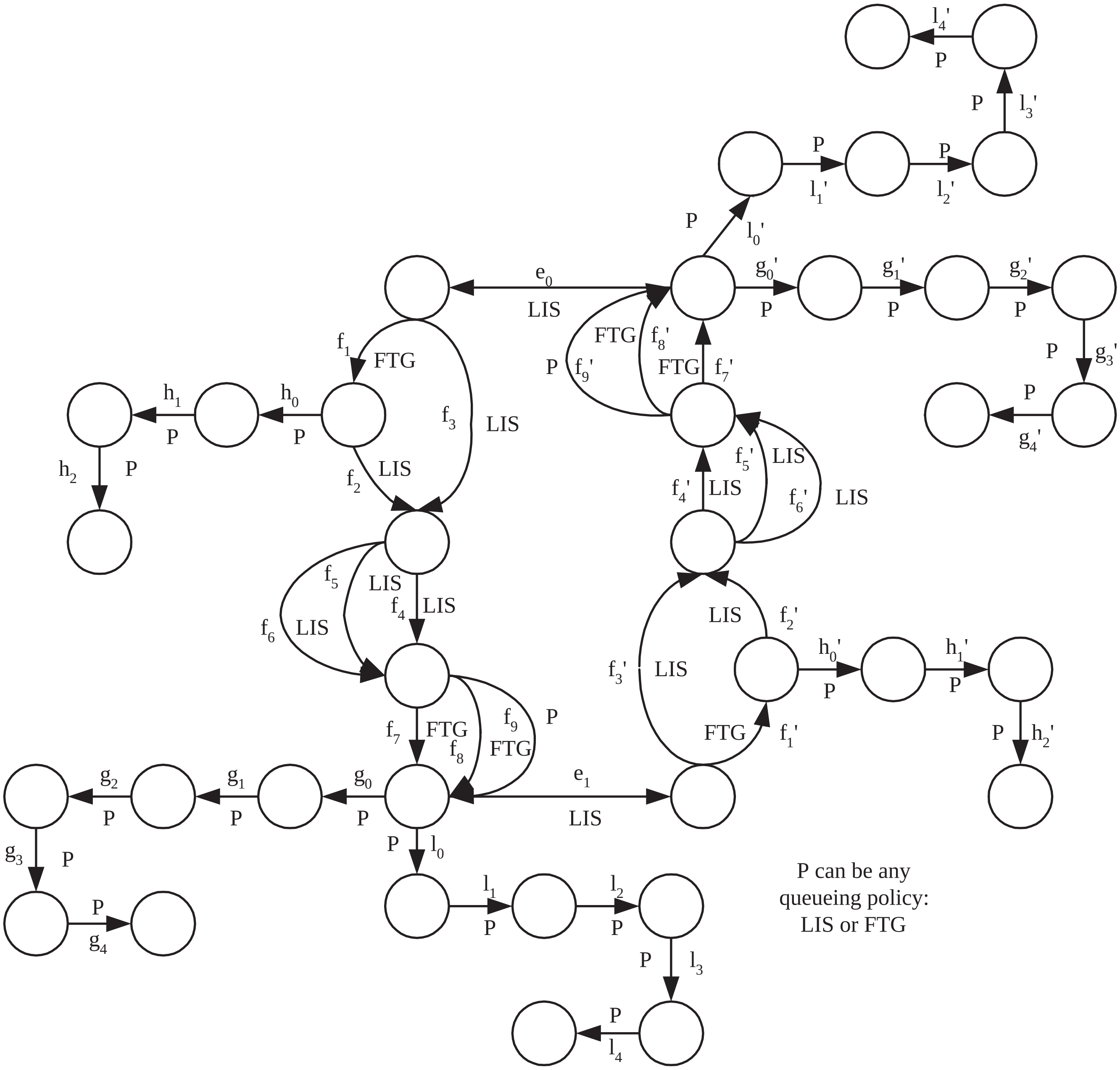

Figure 3: A network running a composition of LIS and FTG

Part c)

Inductive Hypothesis: At the beginning of phase j, there are s_j packets that are queued in the queues f_4', f_7', f_8' (in total) requiring to traverse the edges e_0, f_1, f_2, f_4.

Induction Step: At the beginning of phase $j+1$ there will be more than s_j packets that will be queued in the queues f_4, f_7, f_8, requiring to traverse the edges e_1, f_1', f_2', f_4'.

Let's consider the network N_3 in Figure 3.

All the queues of the network use LIS protocol except the queues that correspond to the edges $f_1, f_1', f_7, f_7', f_8, f_8'$ that use FTG protocol. Moreover, we should mention that edges f_9, f_9' can use any protocol of LIS or FTG because the packets that want to traverse them at the last round of each phase are blocked in their previous edge on their path (let P be any of these protocols). Thus, there is no need to use a specific protocol at queues $f_9, f_9^{'}$ for blocking these packets with a suitable injection. Also, the edges h_0, h_1, h_2, $h_0^{'}, h_1^{'}, h_2^{'}$, g_0, g_1, g_2, g_3, g_4, $g_0^{'}, g_1^{'}, g_2^{'}, g_3^{'}, g_4^{'}$, l_0, l_1, l_2, l_3, l_4 and $l_0^{'}, l_1^{'}, l_2^{'}, l_3^{'}, l_4^{'}$ can use any protocol of LIS or FTG because it is not observed any conflict among packets that belong to different sets in any of them.

The intuition behind this adversary construction and network topology is basically the preservation of the newly injected investing flows in each round of a phase inside the network, until the end of the phase during which they have been injected. The tools we use to achieve this goal are the injections of short intermediate packet flows ($S_{i,k}$) and the introduction of parallel edges (f_4, f_5, f_6 and f_7, f_8, f_9) in the network topology. Although the short intermediate flows do not contribute actually to the number of packets that will be used as the initial flow for the next phase, their role is essential as they are used for blocking investing flows of packets in consecutive rounds. Moreover, some of them have another role, too. They block short intermediate flows that are injected in certain rounds in order to be used for blocking investing flows in the next round of the round in which they have been injected.

Also, an important point we should mention is the use of the short intermediate flow that is injected in the first round of a phase j ($S_{1,1} - flow$) to block a portion of the initial packets that are in the system at the beginning of the phase (s_j) in order these packets to be used twice for blocking investing flow X_1 in the next round. As far as concerns the parallel edges, the purpose of their presence in the network topology is to be guaranteed that the paths of the packet flows injected in the same round do not overlap. The first set of parallel edges f_4, f_5, f_6 is used in order short intermediate flows that are injected at the same round to not overlap, while the second set of parallel edges f_7, f_8, f_9 is used mainly in order the paths of investing flows and the paths of short intermediate flows that are injected at the same round to not overlap. Moreover, the chains of edges h_0, h_1, h_2, $h_0^{'}, h_1^{'}, h_2^{'}$, g_0, g_1, g_2, g_3, g_4, $g_0^{'}, g_1^{'}, g_2^{'}, g_3^{'}, g_4^{'}$, l_0, l_1, l_2, l_3, l_4 and $l_0^{'}, l_1^{'}, l_2^{'}, l_3^{'}, l_4^{'}$, are used to ensure that the short intermediate flows $S_{i,k}$ have priority over the investing flows when they conflict at queues that use FTG protocol. This is achieved because the chains of edges add in the paths of the flows $S_{i,k}$ the appropriate number of edges in order to have priority when they conflict with other packet flows in FTG edges.

We will construct an adversary A such that the induction step will hold. Proving that the induction step holds, we ensure that the inductive hypothesis will hold at the beginning of phase $j+1$ for the symmetric edges with an increased value of s_j packets, $s_{j+1} > s_j$. In order to prove that the induction step works, we should consider that there is a large enough number of packets s_j in the initial system configuration.

During phase j the adversary plays four rounds of injections. The sequence of injections is as follows:

Round 1: It lasts s_j time steps.

At the beginning of this round, there are s_j packets ($S-flow$) in queues $f_4^{'}, f_7^{'}, f_8^{'}$ (in total) requiring to traverse the edges e_0, f_1, f_2, f_4.

Adversary's behavior. During this round the adversary injects a set X_1 of $|X_1| = rs_j$ packets in queue e_0 wanting to traverse the edges $e_0, f_3, f_4, f_7, e_1, f_1^{'}, f_2^{'}, f_4^{'}$. At the same time, the adversary injects a set $S_{1,1}$ of $|S_{1,1}| = rs_j$ packets in queue f_1 that require to traverse the edges f_1, h_0, h_1, h_2.

Evolution of the system configuration. The X_1 packets that are injected into the system during this round are blocked by the $S-flow$ in queue e_0 because e_0 uses LIS as protocol and therefore for every arrival of an injected X_1 packet in queue e_0, there is at least one $S-flow$ packet there that has been in the system for a longer time so it has priority. At the same time, the $S-flow$ is delayed by the set $S_{1,1}$ of packets that are injected during this round into the system. This happens because queue f_1 uses FTG protocol and $S_{1,1}$ packets have more edges to traverse than $S-flow$ packets. Notice that because of FTG, when a packet from $S_{1,1}$ is injected in queue f_1, it has priority over all $S-flow$ packets that are queued in f_1. Thus, all the $S_{1,1}$ packets traverse the edge f_1 along with some packets from $S-flow$, while the rest $S - flow$ packets remain in queue f_1. The number of $S - flow$ packets that traverse the edge f_1 is $s_j - rs_j$.

At the end of this round, there are rs_j packets of $S - flow$ in queue f_1 wanting to traverse the edges f_1, f_2, f_4. Also, there is a set X_1 of $|X_1| = rs_j$ packets in queue e_0 wanting to traverse the edges $e_0, f_3, f_4, f_7, e_1, f_1^{'}, f_2^{'}, f_4^{'}$.

Round 2: It lasts rs_j time steps.

Adversary's behavior. During this round the adversary injects a set X_2 of $|X_2| = r^2 s_j$ packets in queue e_0 requiring to traverse the edges $e_0, f_3, f_4, f_8, e_1, f_1^{'}, f_2^{'}, f_4^{'}$. In addition, the adversary injects a set $S_{2,1}$ of $|S_{2,1}| = r^2 s_j$ packets in queue f_2 wanting to traverse the edges $f_2, f_5, f_7, g_0, g_1, g_2, g_3, g_4$.

Evolution of the system configuration. The X_2 packets that are injected into the system during this round are blocked in queue e_0 that uses LIS protocol by $X_1 - flow$ packets that have been in the system for a longer time. At the same time, all $X_1 - flow$ packets traverse the edges e_0, f_3, but they are blocked in f_4 that uses LIS protocol by the remaining rs_j packets of $S - flow$ in queue f_1 at the end of the previous round. The remaining packets of $S - flow$ want to traverse the edges f_1, f_2, f_4 and have priority over the packets of $X_1 - flow$ because they have been in the system for a longer time. At the end of this round all the remaining rs_j packets of $S - flow$ arrive at edge f_4, where they are absorbed after they traverse it. In addition, the set $S_{2,1}$ of packets that are injected into the system by the adversary during this round are blocked in queue f_2 that uses LIS protocol by the remaining rs_j packets of $S - flow$ that have been in the system for a longer time and consequently they have priority over $S_{2,1} - flow$ packets.

At the end of this round, there are rs_j packets of $X_1 - flow$ in queue f_4 wanting to traverse the edges $f_4, f_7, e_1, f_1^{'}, f_2^{'}, f_4^{'}$. Also, there are $r^2 s_j$ packets of $X_2 - flow$ in queue e_0 requiring to traverse the edges $e_0, f_3, f_4, f_8, e_1, f_1^{'}, f_2^{'}, f_4^{'}$ and $r^2 s_j$ packets of $S_{2,1} - flow$ in queue f_2 wanting to traverse the edges $f_2, f_5, f_7, g_0, g_1, g_2, g_3, g_4$.

Round 3: It lasts $r^2 s_j$ time steps.

Adversary's behavior. During this round the adversary injects a set X_3 of $|X_3| = r^3 s_j$ packets in queue f_4 wanting to traverse the edges $f_4, f_9, e_1, f_1^{'}, f_2^{'}, f_4^{'}$. Also, the adversary injects a set $S_{3,1}$ of $|S_{3,1}| = r^3 s_j$ packets in queue f_5 wanting to traverse the edges $f_5, f_7, g_0, g_1, g_2, g_3, g_4$ and a set $S_{3,2}$ of $|S_{3,2}| = r^3 s_j$ packets in queue f_2 wanting to traverse the edges $f_2, f_6, f_8, l_0, l_1, l_2, l_3, l_4$.

Evolution of the system configuration. During this round, a portion of the $X_1 - flow$ packets $(r^2 s_j)$ traverse the edge f_4, while the rest $rs_j - r^2 s_j$ packets remain in queue f_4. The packets of $X_1 - flow$ that traverse the edge f_4 are blocked in queue f_7 that uses FTG protocol by the $S_{2,1} - flow$ packets that have more edges to traverse than $X_1 - flow$ packets, were blocked in queue f_2 at the end of the previous round and traverse now the edges $f_2, f_5, f_7, g_0, g_1, g_2, g_3$ arriving at edge g_4 where they are absorbed after they traverse it. Furthermore, during this round the $X_2 - flow$ packets, that have been injected into the system during the previous round, traverse the edges e_0, f_3 and they are blocked in queue f_4 that uses LIS protocol by the $X_1 - flow$ packets that have arrived there earlier and consequently they have been in the system for a longer time than $X_2 - flow$ packets.

As far as concerns the X_3 packets that are injected into the system during this round they are blocked in queue f_4 that uses LIS protocol by $X_2 - flow$ packets that have been in the system for a longer time than $X_3 - flow$ packets. Also, the set of $S_{3,1}$ packets, that are injected into the system during this round, are blocked in queue f_5 that uses LIS protocol by the $S_{2,1} - flow$ packets that have been in the system for a longer time than $S_{3,1} - flow$ packets. Finally, the set $S_{3,2}$ of packets that are injected into the system during this round are blocked in queue f_2 that uses LIS protocol by the $S_{2,1} - flow$ packets that have been in the system for a longer time than $S_{3,2} - flow$ packets.

At the end of this round, there are rs_j packets of $X_1 - flow$ in queues f_4, f_7 in total ($rs_j - r^2 s_j$ packets in queue f_4 and rs_j packets in queue f_7) wanting to traverse the edges $f_4, f_7, e_1, f_1^{'}, f_2^{'}, f_4^{'}$. Also, in queue f_4 there are $r^2 s_j$ packets of $X_2 - flow$ requiring to traverse the edges $f_4, f_8, e_1, f_1^{'}, f_2^{'}, f_4^{'}$ and $r^3 s_j$ packets of $X_3 - flow$ wanting to traverse the edges $f_4, f_9, e_1, f_1^{'}, f_2^{'}, f_4^{'}$. Moreover, there are $r^3 s_j$ packets of $S_{3,1} - flow$ in queue f_5 requiring to traverse the edges $f_5, f_7, g_0, g_1, g_2, g_3, g_4$ and $r^3 s_j$ packets of $S_{3,2} - flow$ in queue f_2 requiring to traverse the edges $f_2, f_6, f_8, l_0, l_1, l_2, l_3, l_4$.

Round 4: It lasts $r^3 s_j$ time steps.

Adversary's behavior. During this round the adversary injects a set X_4 of $|X_4| = r^4 s_j$ packets in queue f_3 wanting to traverse the edges $f_3, f_4, f_7, e_1, f_1^{'}, f_2^{'}, f_4^{'}$. Besides the short intermediate flows ($S_{3,1}, S_{3,2}$) that have been injected into the system during the previous round and still remain into the system at the beginning of this round, the system configuration at the beginning of round 4 also consists of the investing flows X_1, X_2, X_3 that have been injected into the system during the previous rounds. During this round, these investing flows along with the investing flow that is injected into the system during this round ($X_4 - flow$) continue to remain into the system independently of the adversarial injection rate r because they are blocked by the short intermediate flows $S_{3,1}, S_{3,2}$.

Evolution of the system configuration. We can distinguish two cases. In the first case, the injection rate r is such that all the $X_1 - flow$ packets ($rs_j - r^2 s_j$) that were queued in the queue of the edge f_4 traverse the edge f_4 and they are queued with the rest $X - 1 - flow$ packets in queue f_7 where they continue to be blocked by $S_{3,1} - flow$ packets that were in queue f_5 at the end of the previous round and traverse during this round the edges f_5, f_7. These packets have more edges to traverse than $X_1 - flow$ packets and thus they have priority in edge f_7 that uses FTG protocol. As far as concerns the other investing flows injected in previous rounds (X_2 and X_3), if there are remaining time steps until the end of round 4 (after the departure of all $X_1 - flow$ packets that were queued in f_4 at the end of the previous round), then a portion of $X_2 - flow$ packets traverse the edge f_4, while the rest packets of $X_2 - flow$ remain in queue f_4. The portion of $X_2 - flow$ packets, that traverse the edge f_4, are blocked in queue f_8 that uses FTG protocol by the $S_{3,2} - flow$ packets that traverse the edges f_2, f_6, f_8 because $S_{3,2} - flow$ packets have more edges to traverse than $X_2 - flow$ and consequently they have priority. Also, $X_3 - flow$ packets remain blocked by $X_2 - flow$ in queue f_4. In the second case the injection rate r is such that a portion of the $X_1 - flow$ packets that were queued in the queue of the edge f_4 traverse it and they are queued with the rest $X - 1 - flow$ packets in queue f_7 where they continue to be blocked by $S_{3,1} - flow$ packets for the same reason as in the previous case. Then, $X_2 - flow$ and $X_3 - flow$ packets remain blocked in queue f_4 by $X_1 - flow$ packets.

Therefore, we have shown that independently of the adversarial injection rate the investing flows X_1, X_2, X_3 continue to remain into the system at the end of round 4. Finally, the X_4 packets that are injected into the system during this round are blocked in queue f_4 that uses LIS protocol by the $X_3 - flow$ packets that have been in the system for a longer time than $X_4 - flow$ packets.

At the end of this round, the number of packets in queues f_4, f_7, f_8 requiring to traverse the edges $e_1, f_1^{'}, f_2^{'}, f_4^{'}$ is

$$s_{j+1} = |X_1| + |X_2| + |X_3| + |X_4| = rs_j + r^2 s_j + r^3 s_j + r^4 s_j$$

In order to have instability, we must have $s_{j+1} > s_j$. Thus,

$$\begin{array}{lcl}
s_{j+1} & > & s_j \\
|X_1| + |X_2| + |X_3| + |X_4| & > & s_j \\
r^3 s_j + r^2 s_j + \frac{r^2 s_j}{r+1} + \frac{r^3 s_j + r s_j}{(r+1)(r^2+r+2)} + \frac{r^4 s_j + r^2 s_j}{(r^2+r+2)(r^3+r^2+2r+2)} - r^2 s_j r s_j + r^2 s_j + r^3 s_j + r^4 s_j & > & s_j \\
r & \geq & 0.519
\end{array}$$

Therefore, the inequality $s_{j+1} > s_j$ holds for $r \geq 0.519$. This argument can be repeated for an infinite number of phases ensuring that the number of packets at the end of a phase will be bigger than at the beginning of the phase.

C Proof of Theorem 5.1

We consider the network $\mathcal{N}$ in Figure 2.

Inductive Hypothesis: At the beginning of phase j, there are s_j packets that are queued in the queues $e_0, f_3^{'}, f_4^{'}, f_5^{'}, f_6^{'}$ (in total) requiring to traverse the edges e_0, f_1, f_3, f_5, all these packets are able to depart from their initial edges to the symmetric part of the network (f_1, f_3, f_5) as a continuous flow in s_j time steps, and the number of packets that are queued in queues $f_4^{'}, f_6^{'}$ is bigger than the number of packets that are queued in queues $f_3^{'}, f_5^{'}$.

Induction Step: At the beginning of phase $j+1$ there will be more than s_j packets (s_{j+1} packets) that are queued in the queues f_3, f_5, f_4, f_6, e_1 (in total) requiring to traverse the edges $e_1, f_1^{'}, f_3^{'}, f_5^{'}$, all of which will be able to depart from their initial edges to the symmetric part of the network $(f_1^{'}, f_3^{'}, f_5^{'})$ in s_{j+1} time steps as a continuous flow and the number of packets that will be queued in queues f_4, f_6 will be bigger than the number of packets that will be queued in queues f_3, f_5.

Notice that our inductive argument claims that if at the beginning of phase j all s_j packets, that are queued in queues $e_0, f_3^{'}, f_4^{'}, f_5^{'}, f_6^{'}$ requiring to traverse the edges e_0, f_1, f_3, f_5, manage to traverse their initial edges in s_j time steps as a continuous flow, then at the beginning of phase $j+1$ all s_{j+1} packets, that will be queued in queues f_3, f_5, f_4, f_6, e_1 requiring to traverse the edges $e_1, f_1^{'}, f_3^{'}, f_5^{'}$, will be able to traverse their initial edges in s_{j+1} time steps as a continuous flow. This argument guarantees the reproduction of the inductive hypothesis in queues f_3, f_5, f_4, f_6, e_1 even if there are flows (in particular in queues f_3, f_4, f_5) that do not want to traverse the edges $e_1, f_1^{'}, f_3^{'}, f_5^{'}$ the packets of which are regularly spread among the packets that want to traverse these edges.

Furthermore, this argument implies the third part of the inductive argument, which claims that if at the beginning of phase j, the number of packets that are queued in queues $f_4^{'}, f_6^{'}$ is bigger than the number of packets that are queued in queues $f_3^{'}, f_5^{'}$, then at the beginning of phase $j+1$ the number of packets that will be queued in queues f_4, f_6 will be bigger than the number of packets that will be queued in queues f_3, f_5. This happens because in the first round of the adversary's construction we inject packets in queue $f_4^{'}$ and if the third part of the inductive hypothesis doesn't hold then we cannot guarantee that all the initial s_j packets will depart their initial edges to the edges f_1, f_3, f_5 in s_j time steps as a continuous flow. However, we include it into the inductive hypothesis for readability reasons.

We will construct an adversary A such that the induction step will hold. Proving that the induction step holds, we ensure that the inductive hypothesis will hold at the beginning of phase $j+1$ for the symmetric edges with an increased value of s_j, $s_{j+1} > s_j$.

From the inductive hypothesis, initially, there are s_j packets (called $S-flow$) in the queues $e_0, f_3^{'}, f_4^{'}$, $f_5^{'}$, $f_6^{'}$ requiring to traverse the edges e_0, f_1, f_3, f_5. In order to prove the induction step, it is assumed that there is a set S with a large enough number of $|S| = s_j$ packets in the initial system configuration.

During phase j the adversary plays three rounds of injections. The sequence of injections is as follows:

Round 1: This round lasts s_j time steps.

Adversary's behavior. During this round the adversary injects a set X of $|X| = rs_j$ packets in queue $f_4^{'}$ wanting to traverse the edges $f_4^{'}, f_6^{'}, e_0, f_2$, $f_3, f_5, e_1, f_1^{'}, f_3^{'}, f_5^{'}$. Also, the adversary injects a set S_1 of $|S_1| = rs_j$ packets in queue f_1 wanting to traverse the edge f_1.

Evolution of the system configuration. The packets of set X are blocked into the system (in queue e_0) by the packets of set S because the packets of S that are queued in queues f_4', f_6' at the beginning of this round are more than the packets of S that are queued in queues f_3', f_5'. Thus, all the S packets will traverse their initial edges in s_j time steps blocking the packets of set X.

At the same time, the packets of set S are delayed in queue f_1 by the packets of set S_1. The packets of set S_1 get mixed with the packets of the set S. Notice that because of FIFO protocol, the packets of sets S, S_1 mix in consecutive blocks according to their initial proportion of their sizes (fair mixing property). Since $|S| = s_j$ and $|S_1| = rs_j$, these proportions are $\frac{1}{r+1}$ and $\frac{r}{r+1}$, respectively. Thus, during the s_j steps of round 1, the packets of sets S, S_1, which cross f_1 are, respectively, $\frac{s_j}{r+1}$ and $\frac{rs_j}{r+1}$. Therefore, the remaining packets in queue f_1 are:

- for packet set S: a set S_{rem} of $|S_{rem}| = s_j - \frac{s_j}{r+1} = \frac{rs_j}{r+1}$ packets,
- for packet set S_1: a set $S_{1,rem}$ of $|S_{1,rem}| = rs_j - \frac{rs_j}{r+1} = \frac{r^2 s_j}{r+1}$ packets.

Round 2: It lasts rs_j steps.

Adversary's behavior. The adversary injects a set Y of $|Y| = r^2 s_j$ packets in queue f_4' requiring to traverse the edges $f_4', f_6', e_0, f_4, f_6, e_1, f_1', f_3', f_5'$. At the same time, the adversary injects a set S_2 of $|S_2| = r^2 s_j$ packets in queue f_2 wanting to traverse the edge f_2, a set S_3 of $|S_3| = r^2 s_j$ packets in queue f_3 wanting to traverse the edge f_3, and a set S_4 of $|S_4| = r^2 s_j$ packets in queue f_5 wanting to traverse the edge f_5.

Evolution of the system configuration. The packets of set Y are blocked by the set X in queue e_0. At the same time, the packet sets X, S_2 mix in consecutive blocks according to their initial proportion of their sizes because of FIFO protocol. Since $|X| = rs_j$ and $|S_2| = r^2 s_j$, these proportions are $\frac{1}{r+1}$ and $\frac{r}{r+1}$, respectively. Thus, during the rs_j steps of this round, the packets of sets X, S_2, that traverse the edge f_2 are:

- for set X: a set X_{pass,f_2} of $|X_{pass,f_2}| = \frac{rs_j}{r+1}$ packets,
- for set S_2: a set $S_{2,pass,f_2}$ of $|S_{2,pass,f_2}| = \frac{r^2 s_j}{r+1}$ packets. These packets are absorbed.

On the other hand, the remaining packets in queue f_2 are

- for set X: a set X_{rem,f_2} of $|X_{rem,f_2}| = rs_j - \frac{rs_j}{r+1} = \frac{r^2 s_j}{r+1}$ packets,
- for set S_2: a set S_{2,rem,f_2} of $|S_{2,rem,f_2}| = r^2 s_j - \frac{r^2 s_j}{r+1} = \frac{r^3 s_j}{r+1}$ packets.

Notice that in queue f_1, at the beginning of this round, there are the packets of sets $S_{rem}, S_{1,rem}$ that have remained there from the previous round. Since their total number is rs_j, which is equal to the duration of this round, the packets of $S_{1,rem}$ do not delay the packets of S_{rem}. In addition, the $S_{1,rem}$ packets are absorbed after they traverse the edge f_1, therefore only the S_{rem} packets require the edge f_3. As a result the stream arriving from f_1 to f_3 contains empty spaces at the positions of the $S_{1,rem}$ packets. However, these empty spaces are uniformly spread for the duration of the time period. Thus, during round 2, three different sets of packets (packet flows) arrive at queue f_3:

- X_{pass,f_2} where $|X_{pass,f_2}| = \frac{rs_j}{r+1}$ packets. These packets mixed with packets of set $S_{2,pass,f_2}$. But, since their total number is rs_j (that is equal to the duration of the round), $S_{2,pass,f_2}$ packets do not delay X_{pass,f_2} packets. Also because the $S_{2,pass,f_2}$ packets are absorbed after they traverse the edge f_2, only the X_{pass,f_2} packets require the edge f_3. As a result the stream arriving from f_2 to f_3 contains empty spaces at the positions of the $S_{2,pass,f_2}$ packets.
- S_{rem}, where $|S_{rem}| = \frac{rs_j}{r+1}$,
- S_3, where $|S_3| = r^2 s_j$.

Since the total number of packets of the three flows arriving to queue f_3 is

$$|T| = \frac{r^3 s_j + r^2 s_j + 2rs_j}{r+1}$$

the corresponding proportions are:

- for X_{pass,f_2}: $\frac{|X_{pass,f_2}|}{T} = \frac{1}{r^2+r+2}$,
- for S_{rem}: $\frac{|S_{rem}|}{T} = \frac{1}{r^2+r+2}$,
- for S_3: $\frac{|S_3|}{T} = \frac{r^2+r}{r^2+r+2}$.

Thus, the remaining packets in queue f_3 from each flow at the end of round 2 are:

- for X_{pass,f_2}: a set X_{rem,f_3} of $|X_{rem,f_3}| = \frac{rs_j}{r+1} - \frac{rs_j}{r^2+r+2} = \frac{r^3 s_j + rs_j}{(r+1)(r^2+r+2)}$ packets,
- for S_{rem}: a set S_{rem,f_3} of $|S_{rem,f_3}| = \frac{rs_j}{r+1} - \frac{rs_j}{r^2+r+2} = \frac{r^3 s_j + rs_j}{(r+1)(r^2+r+2)}$ packets,
- for S_3: a set $S_{3,rem}$ of $|S_{3,rem}| = r^2 s_j - rs_j \frac{r^2+r}{r^2+r+2} = \frac{r^4 s_j + r^2 s_j}{r^2+r+2}$ packets.

The technique of proportions can still be used even if some flow of packets has empty spaces, since the empty spaces are uniformly spread. Notice that during round 2 the stream arriving to the edge f_5 contains 3 different flows:

- the $S_4 - flow$, where $|S_4| = r^2 s_j$,
- the portion S_{pass,f_3} of $S_{rem} - flow$ that traverses the edge f_3, where $|S_{pass,f_3}| = \frac{rs_j}{r^2+r+2}$,
- the portion X_{pass,f_3} of $X_{pass,f_2} - flow$ that traverse the edge f_3, where $|X_{pass,f_3}| = \frac{rs_j}{r^2+r+2}$.

Notice also that the S_3 packets that traverse the edge f_3 are absorbed after they traverse edge f_3. Since the total number of packets in the three flows is $|T'| = \frac{(r^4+r^3+2r^2+2r)s_j}{r^2+r+2}$ the corresponding proportions are:

- for X_{pass,f_3}: $\frac{|X_{pass,f_3}|}{|T'|} = \frac{rs_j}{(r^4+r^3+2r^2+2r)s_j} = \frac{1}{r^3+r^2+2r+2}$,
- for S_{pass,f_3}: $\frac{|S_{pass,f_3}|}{|T'|} = \frac{1}{r^3+r^2+2r+2}$,

- for S_4: $\frac{|S_4|}{|T'|} = \frac{r^2 s_j (r^2+r+2)}{(r^4+r^3+2r^2+2r)s_j} = \frac{r^3+r^2+2r}{r^3+r^2+2r+2}$.

Thus, the remaining packets from each flow in queue f_5 at the end of round 2 are:

- for X_{pass,f_3}: $\frac{rs_j}{r^2+r+2} - \frac{rs_j}{r^3+r^2+2r+2} = \frac{r^4 s_j + r^2 s_j}{(r^2+r+2)(r^3+r^2+2r+2)}$ for X_{pass,f_3},
- for S_{pass,f_3}: $\frac{rs_j}{r^2+r+2} - \frac{rs_j}{r^3+r^2+2r+2} = \frac{r^4 s_j + r^2 s_j}{(r^2+r+2)(r^3+r^2+2r+2)}$
- for S_4: $r^2 s_j - \frac{(r^4+r^3+2r^2)s_j}{r^3+r^2+2r+2} = \frac{r^5 s_j + r^3 s_j}{r^3+r^2+2r+2}$.

Round 3: It lasts $|T_{round_3}| = r^2 s_j$ time steps.

Adversary's behavior. During this round the adversary injects a set S_5 of $|S_5| = r^3 s_j$ packets in queue f_4 requiring to traverse only the edge f_4 and a set Z of $|Z| = r^3 s_j$ packets in queue f_6 requiring to traverse the edges $f_6, e_1, f_1', f_3', f_5'$.

Evolution of the system configuration. During this round the packet set S_5 get mixed with the set Y in consecutive blocks according to their initial proportion of their sizes. These proportions are for $Y : \frac{1}{r+1}$ and for $S_5 : \frac{r}{r+1}$. Thus, during the $r^2 s_j$ steps of this round, the packets of packet sets Y, S_5 that remain in queue f_4 are respectively:

- for Y: a set Y_{rem} of $|Y_{rem}| = r^2 s_j - \frac{r^2 s_j}{r+1} = \frac{r^3 s_j}{r+1}$ packets,
- for S_5: a set $S_{5,rem}$ of $|S_{5,rem}| = r^3 s_j - \frac{r^3 s_j}{r+1} = \frac{r^4 s_j}{r+1}$ packets.

On the other hand the packets of packet sets Y, S_5 that traverse the edge f_4 are respectively:

- for Y: a set Y_{pass} of $|Y_{pass}| = \frac{r^2 s_j}{r+1}$ packets,
- for S_5: a set $S_{5,pass}$ of $|S_{5,pass}| = \frac{r^3 s_j}{r+1}$ packets. The $S_{5,pass}$ packets are absorbed after they traverse the edge f_4 because they are single-edge injections.

Furthermore, during this round the set of packets Z get mixed with the set Y_{pass} in consecutive blocks according to their initial proportion of their sizes. These proportions are:

- for Y_{pass}: $\frac{\frac{r^2 s_j}{r+1}}{\frac{r^2 s_j}{r+1} + r^3 s_j} = \frac{1}{r^2+r+1}$
- for Z: $\frac{r^3 s_j}{r^3 s_j + \frac{r^2 s_j}{r+1}} = \frac{r(r+1)}{r^2+r+1}$.

Thus, during the $r^2 s_j$ steps of this round, the packets that traverse the edge f_6 are respectively:

- for Y_{pass}: a set Y_{pass,f_6} of $|Y_{pass,f_6}| = \frac{r^2 s_j}{r^2+r+1}$ packets,
- for Z: a set Z_{pass,f_6} of $|Z_{pass,f_6}| = \frac{r^3 s_j (r+1)}{r^2+r+1}$ packets.

Therefore, the remaining packets in queue f_6 are:

- for Y_{pass}: a set Y_{rem,f_6} of $|Y_{rem,f_6}| = \frac{r^2 s_j}{r+1} - \frac{r^2 s_j}{r^2+r+1} = \frac{r^4 s_j}{(r+1)(r^2+r+1)}$ packets,
- for Z: a set Z_{rem,f_6} of $|Z_{rem,f_6}| = r^3 s_j - \frac{r^3 s_j (r+1)}{r^2+r+1} = \frac{r^5 s_j}{r^2+r+1}$ packets.

Notice that the total number of packets that are queued in queue f_5 at the end of round 2 is equal to

$$|T_1| = |X_{pass,f_3}| + |S_{pass,f_3}| + |S_4| = \frac{(r^7 + r^6 + 3r^5 + 3r^4 + 2r^3 + 2r^2)s_j}{(r^2 + r + 2)(r^3 + r^2 + 2r + 2)}$$

However, $|T_1| < r^2 s_j, \forall r > 0$. Thus, all the packets in queue f_5 at the end of the previous round will traverse the edge f_5. Therefore, the remaining time during which packets arriving to edge f_5 from edge f_3 can traverse the edge f_5 is $t_{rem} = r^2 s_j - |T_1|$. Assigning the value of T_1 in the previous equation we take

$$t_{rem} = \frac{(r^6 + 2r^5 + 3r^4 + 4r^3 + 2r^2)s_j}{r^5 + 2r^4 + 5r^3 + 6r^2 + 6r + 4}$$

Notice also that the total number of packets that are queued in queue f_3 at the beginning of round 3 is

$$|T_2| = |X_{rem,f_3}| + |S_{rem,f_3}| + |S_{3,rem}| = \frac{(r^5 + r^4 + 3r^3 + r^2 + 2r)s_j}{(r + 1)(r^2 + r + 2)}$$

However, $|T_2| \geq r^2 s_j, \forall r$. Thus, a number of $X_{rem,f_3}, S_{rem,f_3}, S_{3,rem}$ packets can remain in queue f_3 at the end of round 3. This number is

$$|T_3| = |T_2| - r^2 s_j = \frac{(2r - r^2 - r^4)s_j}{(r + 1)(r^2 + r + 2)}$$

From this number, the number of packets that belong to set $S_{3,rem}$ is

$$|S_{3,rem,f_3}| = |T_3| \frac{r^2 + r}{r^2 + r + 2} = \frac{(2r^2 - r^3 - r^5)s_j}{(r^2 + r + 2)^2}$$

where $\frac{r^2+r}{r^2+r+2}$ is the mixing proportion of the sets $S_{3,rem}$, X_{rem,f_3} and S_{rem,f_3} in queue f_3.

Except the remaining packets in queue f_3, there is a number of packets of sets X_{rem,f_3}, S_{rem,f_3} and $S_{3,rem}$ that traverse the edge f_3 during round 3. This number of packets is $r^2 s_j$ because the duration of round 3 is $r^2 s_j$. From these packets, the packets that belong to $S_{3,rem}$ are absorbed because they are single-edge injections. The number of these packets is $r^2 s_j \frac{r^2+r}{r^2+r+2}$, where $\frac{r^2+r}{r^2+r+2}$ is the mixing proportion of $S_{3,rem}$ with X_{rem,f_3}, S_{rem,f_3} in queue f_3. In $r^2 s_j$ steps, the same portion of X_{rem,f_3} and S_{rem,f_3} packets traverse the edge f_3, $X_{rem,f_3,pass}$ and $S_{rem,f_3,pass}$ correspondingly, because they get mixed in queue f_3 with the same proportion. The size of the portion of X_{rem,f_3} and S_{rem,f_3} packets that traverse edge f_3 is $|X_{pass,f_3,pass}| = |S_{rem,f_3,pass}| = \frac{r^2 s_j}{r^2+r+2)}$. Consequently, the sum of these packets is

$$|T_4| = |X_{rem,f_3,pass}| + |S_{rem,f_3,pass}| = \frac{2r^2 s_j}{r^2+r+2)}$$

However, $|T_4| \geq t_{rem}\ \forall r$. Therefore, during the remaining t_{rem} time steps of this round a number of $X_{pass,f_3,pass}$ and $S_{rem,f_3,pass}$ packets, which arrive to the edge f_5 from the edge f_3, will traverse the edge f_5. Because $X_{pass,f_3,pass}$, and $S_{rem,f_3,pass}$ packets arrive to the edge f_5 from the edge f_3 with the same proportion and their sum T_4 is greater or equal to the remaining time steps t_{rem}, the same number of $X_{pass,f_3,pass}$ and $S_{rem,f_3,pass}$ packets will traverse the edge f_5 (X_{pass,f_3,f_5} and S_{absorb,f_3,f_5} correspondingly). This number is

$$|X_{pass,f_3,f_5}| = |S_{absorb,f_3,f_5}| = \frac{t_{rem}}{2} = \frac{(r^6+2r^5+3r^4+4r^3+2r^2)s_j}{2(r^5+2r^4+5r^3+6r^2+6r+4)}$$

In addition, the total number of packets that are in queue f_2 at the end of round 2 is $|T_5| = r^2 s_j$ that is equal to the round's time duration. Thus, all the $X - flow$ packets traverse the edge f_2 and arrive to the edge f_3 where they are blocked because of the packets that are already queued in queue f_3 at the end of the previous round whose number is greater than or equal to the time duration of the current round as we have shown previously.

The first part of the proof is concluded as follows. In order to have instability the number of packets that are queued in queues f_3, f_4, f_5, f_6, e_1 requiring to traverse the edges $e_1, f_1^{'}, f_3^{'}, f_5^{'}$ at the end of this round, s_{j+1}, should be more than the initial s_j packets that were queued in the system at corresponding queues at the beginning of round 1. Thus,

$$\begin{array}{lcl}
s_{j+1} & > & s_j \\
|Z| + |Y| + |X_{rem,f_2}| + |X_{rem,f_3}| + |X_{pass,f_3}| - |T_{round_3}| & > & s_j \\
r^3 s_j + r^2 s_j + \frac{r^2 s_j}{r+1} + \frac{r^3 s_j + r s_j}{(r+1)(r^2+r+2)} + \frac{r^4 s_j + r^2 s_j}{(r^2+r+2)(r^3+r^2+2r+2)} - r^2 s_j & > & s_j \\
r & \geq & 0.749
\end{array}$$

Therefore, the inequality $s_{j+1} > s_j$ holds for $r \geq 0.749$. Now in order to conclude the proof we should also consider the following technical lemma:

Lemma C.1 *For $r \geq 0.686$, the number of packets that remain in queues f_3, f_5 ($Q(f_3), Q(f_5)$) at the end of round 3 is less than or equal to the number of packets that remain in queues f_4, f_6 at the end of round 3 ($Q(f_4), Q(f_6)$).*

Proof: Consider the following definitions:

- $A = |X_{rem,f_3}| - |X_{pass,f_3,f_5}|$,
- $B = |S_{rem,f_3}| - |S_{absorb,f_3,f_5}|$,
- $C = |X_{rem,f_2}| + |S_{3,rem,f_3}|$.

where $X_{rem,f_3}, X_{pass,f_3,f_5}, S_{rem,f_3}, S_{absorb,f_3,f_5}, X_{rem,f_2}, S_{3,rem,f_3}$ have been defined and estimated previously in the main body of the proof at round 3. Then

$$Q(f_3) + Q(f_5) = A + B + C$$

On the other hand

$$Q(f_4) + Q(f_6) = (|Y_{rem}| + |S_{5,rem}|) + (|Y_{rem,f_6}| + |Z_{rem,f_6}|)$$

where $Y_{rem}, S_{5,rem}, Y_{rem,f_6}, Z_{rem,f_6}$ have been defined and estimated previously in the main body of the proof at round 3. Assigning the appropriate values to the above equations we take:

$$Q(f_4) + Q(f_6) = r^3 s_j + \frac{r^6 s_j + r^5 s_j + r^4 s_j}{(r+1)(r^2+r+1)}$$

and

$$Q(f_3)+Q(f_5) = \frac{r^2 s_j}{r+1} + 2\left(\frac{r^3 s_j + r s_j}{(r+1)(r^2+r+2)} - \frac{r^6 s_j + 2r^5 s_j + 3r^4 s_j + 4r^3 s_j + 2r^2 s_j}{2(r^5 + 2r^4 + 5r^3 + 6r^2 + 6r + 4)}\right) + \frac{2r^2 s_j - r^3 s_j - r^5 s_j}{(r^2+r+2)^2}$$

For $r \geq 0.686$, the inequality $Q(f_3) + Q(f_5) \leq Q(f_4) + Q(f_6)$ holds. If this constraint holds, the third part of the inductive hypothesis of the theorem 5.1 holds too. ■

Notice that we have, till now, managed to reproduce the inductive hypothesis in queues f_3, f_5, f_4, f_6, e_1 but with some packet flows (in particular in queues f_4, f_3, f_5) having empty spaces (packets that don't want to traverse the edges $e_1, f_1^{'}, f_3^{'}, f_5^{'}$). In order for the induction step to work we must show that all the packets in these queues will manage to depart to the symmetric part of the network $(f_1^{'}, f_3^{'}, f_5^{'})$ in s_{j+1} time steps as a continuous flow.

As we have shown above all the packets that are queued in queue e_1 at the end of round 3 want to traverse the edges $e_1, f_1^{'}, f_3^{'}, f_5^{'}$ and their flow is continuous without empty spaces (packets that don't want to traverse the edges $e_1, f_1^{'}, f_3^{'}, f_5^{'}$). Also, from Lemma C.1, the number of packets in queues f_4, f_6 is bigger than the number of packets in queues f_3, f_5 at the end of round 3. The packets that are queued in queue f_6 can be seen as a continuous flow that wants to traverse the edges $e_1, f_1^{'}, f_3^{'}, f_5^{'}$, while the set of packets in queue f_4 consists of packets that want to traverse the edges $e_1, f_1^{'}, f_3^{'}, f_5^{'}$ (Y_{rem}) and packets that are single-edge injections ($S_{5,rem}$), which can be considered as empty spaces. Because of that, we should show that all the Y_{rem} packets manage to leave the edge f_4 during the s_{j+1} time steps. This is established in Lemma C.2.

Lemma C.2 *For any injection rate r, all the Y_{rem} packets manage to leave the edge f_4 during s_{j+1} time steps.*

Proof: In order to show that, it suffices to estimate the number G of Y_{rem} packets that manage to traverse the edge f_4 within the time steps, the packets in queue f_6 (Z_{rem,f_6}, Y_{rem,f_6} at the end of round 3) need to traverse the edges f_6, e_1. After the estimation of number G, if we show that it is bigger than or equal to the number of Y_{rem}, $S_{5,rem}$ packets that remain in queue f_4, we have actually proved that the $S_{5,rem}$ packets do not delay the Y_{rem} packets. Therefore, it is proved that in s_{j+1} time steps, all the s_{j+1} packets that want to traverse the edges $e_1, f_1^{'}, f_3^{'}, f_5^{'}$ leave their initial edges to $f_1^{'}, f_3^{'}, f_5^{'}$ as a continuous flow.

In the estimation of number G, we will take into account the number of packets in queues f_6, e_1 at the end of round 3. It is not needed to take into account the packets arriving to the edge e_1 from the edges f_3, f_5 during the estimation of G because they simply increase the time, which is given to the packets in f_6 to traverse the edges f_6, e_1. This has as a result, the time that is given to the Y_{rem} packets to traverse the edge f_4 to be increased and consequently more Y_{rem} packets (bigger G) to traverse the edge f_4. Therefore, if we prove our argument without taking into account the packets arriving to the edge e_1 from the edges f_3, f_5, this also holds for the case we take into account the previous packet quantity.

The number of time steps that is needed in order all the initial packets in queue f_6 to traverse the edges f_6, e_1 is equal to the sum of the number of Z_{rem,f_6}, Y_{rem,f_6} packets in queue f_6 and the number of packets in queue e_1 at the end of round 3. This number is

$$|T_6| = D + E + F$$

where

- $D = |Z_{rem,f_6}| + |Y_{rem,f_6}|$ is the number of remaining packets in queue f_6 at the end of round 3,
- $E + F$ with $E = |Z_{pass,f_6}| + |Y_{pass,f_6}| + |X_{pass,f_3}|$ and $F = |X_{pass,f_3,f_5}| - T_{round3}$ is the number of the packets that are queued in queue e_1 at the end of round 3.

Replacing D, E, F with their corresponding values:

$$|T_6| = \frac{r^4 s_j}{r+1} + \frac{r^4 s_j + r^2 s_j}{(r^2+r+2)(r^3+r^2+2r+2)} + \frac{r^6 s_j + 2r^5 s_j + 3r^4 s_j + 4r^3 s_j + 2r^2 s_j}{2(r^5 + 2r^4 s + 5r^3 + 6r^2 + 6r + 4)}$$

Therefore, in T_6 time steps, the number of Y_{rem} packets that traverse the edge f_4 is

$$G = |T_6| \frac{1}{r+1}$$

and the number of $S_{5,rem}$ packets that traverse the edge f_4 is

$$H = |T_6| \frac{r}{r+1}$$

where $\frac{1}{r+1}$ and $\frac{r}{r+1}$ are the mixing proportions of Y_{rem} and $S_{5,rem}$ correspondingly. In addition, the number of Y_{rem} and $S_{5,rem}$ packets that remain in queue f_4 after T_6 time steps are $|Y_{rem}| - G$ and $|S_{5,rem}| - H$, correspondingly.

We should show that the number G of Y_{rem} packets that manage to traverse queue f_4 within $|T_6|$ time steps is bigger than or equal to the number of $Y_{rem}, S_{5,rem}$ packets that remain in queue f_4 after T_6 time steps. Therefore,

$$G \geq |Y_{rem}| - G + |S_{5,rem}| - H$$

This holds for $r \geq 0$. ■

We have so far established two (non-trivial) sufficient constraints on r for instability, namely that $r \geq 0.749$ and $r \geq 0.686$. Clearly, taking $r \geq \max\{0.749, 0.686\} = 0.749$ suffices for instability of the network $\mathcal{N}$ in the constructed execution. This concludes our proof.