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

D. Koukopoulos[1], M. Mavronicolas[2], S. Nikoletseas[1], and P. Spirakis[1]*

[1] Department of Computer Engineering & Informatics, University of Patras and Computer Technology Institute (CTI), Riga Fereou 61, P. O. Box 1122, 261 10 Patras, Greece. {nikole,koukopou,spirakis}@cti.gr

[2] Department of Computer Science, University of Cyprus, 1678 Nicosia, Cyprus. mavronic@ucy.ac.cy

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 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 specifically address 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. *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 protocols 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 FIFO 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.

* 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]).

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

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

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 contention-resolution protocol is running at all queues. In this work, we embark on a study of the effect of composing contention-resolution protocols on the stability of the resulting system. (By *composition* of contention-resolution protocols, we mean the simultaneous 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 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 1).
– We establish that, surprisingly, the composition of LIS with any of SIS, NTS and FTG is *not* universally stable (Theorem 2).
  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 3).
  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 2). 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).

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 induction 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).

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., [10]). 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 community of Security Protocols, the statement that security is not compositional is considered to be folklore [11].

Adversarial Queueing Theory. The model of *Adversarial Queueing Theory* was developed by Borodin *et al.* [3] as a more realistic 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., [1,2,5,8,12]).

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.

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

	LIS	SIS	NTS	FTG
LIS	**US** ([1])			
SIS	[0.519,1] (Thm. 2)	**US** ([1])		
NTS	[0.519,1] (Thm. 2)	**US** (Thm. 1)	**US** ([1])	
FTG	[0.519,1] (Thm. 2)	**US** (Thm. 1)	**US** (Thm. 1)	**US** ([1])

2 Preliminaries

The definition of a *bounded adversary* $\mathcal{A}$ 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) in which we want to derive lower bounds, using an adversary with zero "burstiness" (that is, taking $b = 0$) results in more simplified proofs. Thus, for these purposes, we say that an adversary $\mathcal{A}$ 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. Clearly, an instability result for an adversary with no burstiness ($b = 0$) applies also to an adversary that may use burstiness ($b \geq 0$). 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, $\mathcal{A}$, P) where G is a network, $\mathcal{A}$ 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, $\mathcal{A}$, 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, $\mathcal{A}$, 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 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 us 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 1. *If the used queueing disciplines in the system are a)* SIS *and* FTG*, then the system (G, $\mathcal{A}$,* 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.*

Proof. (*Sketch*) In order to prove this theorem, we first show the following two lemmas:

Lemma 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*.*

Lemma 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*.*

Lemma 1 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 2, we specialize Lemma 1 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 us 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 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. Therefore, the upper bound for delay 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 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 induction hypothesis.*

Definition 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 2. *Let $r \geq 0.519$. There is a network N_1 and an adversary $\mathcal{A}$ of rate r, such that the system (N_1,$\mathcal{A}$,Queueing Disciplines) is unstable, if the used queueing disciplines are a)* LIS *and* SIS*, b)* LIS *and* NTS *or c)* LIS *and* FTG.

Proof. (*Sketch*) **Part a)** Consider the network N_1 in Figure 1.
Induction 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 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

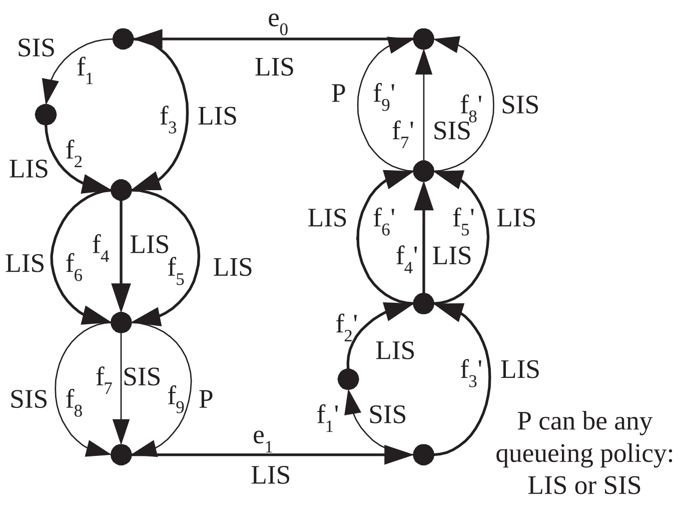

Fig. 1. The network N_1 running a composition of LIS and SIS

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 $\mathcal{A}$ such that the induction step will hold. Proving that the induction step holds, we ensure that the induction 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

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

In order to have instability, we must have $s_{j+1} > s_j$. Therefore from (1), 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.

Part 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,$\mathcal{A}$,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.

Part c) This part of the theorem can be proved similarly to the previous parts. Thus, the construction of the adversary $\mathcal{A}$ for proving instability of the system (N_3,$\mathcal{A}$,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. $\square$

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 3. *Let $r \geq 0.749$. There is a network $\mathcal{N}$ and an adversary $\mathcal{A}$ of rate r, such that the ($\mathcal{N}$, $\mathcal{A}$, FIFO) system is unstable.*

Proof. (*Sketch*) We consider the network $\mathcal{N}$ in Figure 2.
Induction 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.

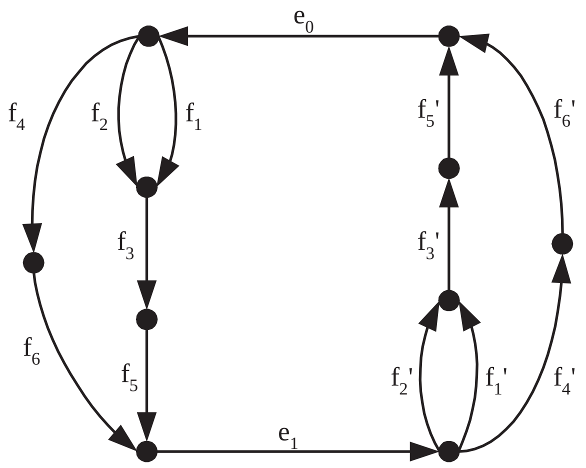

Fig. 2. FIFO network $\mathcal{N}$

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 induction 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 induction 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 induction hypothesis for readability reasons.

We will construct an adversary $\mathcal{A}$ such that the induction step will hold. Proving that the induction step holds, we ensure that the induction hypothesis will hold at the beginning of phase $j+1$ for the symmetric edges with an increased value of s_j packets, $s_{j+1} > s_j$. From the induction 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 mix on the proportion of the sizes with which they arrive in queue f_6 during this round.

The total number of packets in queue f_3 at the beginning of round 3 is

$$|T_1| = |X_{rem,f_3}| + |S_{rem,f_3}| + |S_{3,rem}| = \frac{(r^5 + r^4 + 3r^3 + r^2 + 2r)s_j}{(r+1)(r^2+r+2)} \quad (2)$$

However, $|T_1| \geq r^2 s_j, \forall r$. 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 $|T_2| = \frac{(2r-r^2-r^4)s_j}{(r+1)(r^2+r+2)}$. From this number $|S_{3,rem,f_3}| = \frac{(2r^2-r^3-r^5)s_j}{(r^2+r+2)^2}$ packets belong to the set S_3. In addition, the total number of packets that are in f_2 at the end of round 2 is $|T_3| = r^2 s_j$ 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 queued in 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. Therefore, it should hold

$$|Z| + |Y| + |X_{rem,f_2}| + |X_{rem,f_3}| + |X_{pass,f_3}| - |T_{round_3}| > s_j \tag{3}$$

Substituting in (3), we take

$$r^3 s_j + \frac{r^2 s_j}{r+1} + \frac{r^3 s_j + r s_j}{(r+1)(r^2+r+2)} + \frac{r^4 s_j + r^2 s_j}{(r^2+r+2)(r^3+r^2+2r+2)} > s_j \tag{4}$$

This holds for $r \geq 0.749$. Now in order to conclude the proof we prove:

Lemma 3. *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 induction 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 3, 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. We formally establish:

Lemma 4. *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 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

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,* pp. 168–177, 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. N. Lynch, *Distributed Algorithms,* Morgan Kaufmann, 1996.
11. J. Mitchell, Email Communication to I. Lee, April 2002.
12. P. Tsaparas, *Stability in Adversarial Queueing Theory,* M.Sc. Thesis. Department of Computer Science, University of Toronto, 1997.