Available online at www.sciencedirect.com
ScienceDirect
Theoretical Computer Science 381 (2007) 44–56
Theoretical Computer Science
www.elsevier.com/locate/tcs

The increase of the instability of networks due to Quasi-Static link capacities[☆]

Dimitrios Koukopoulos[a,b,*], Marios Mavronicolas[c,d], Paul Spirakis[e,b]

[a] *Department of Applied Informatics in Management and Finance, Technological Education Institute of Ionian Islands, Greece*
[b] *RACTI, 61 Riga Fereou, 26221 Patras, Greece*
[c] *Department of Computer Science, University of Cyprus, 1678 Nicosia, Cyprus*
[d] *Faculty of Computer Science, Electrical Engineering and Mathematics, University of Paderborn, 33102 Paderborn, Germany*
[e] *Department of Computer Engineering and Informatics, University of Patras, Greece*

Received 6 February 2005; received in revised form 13 February 2007; accepted 3 April 2007

Communicated by D. Peleg

Abstract

In this work, we study the impact of the dynamic changing of the network link capacities on the stability properties of *packet-switched* networks. Especially, we consider the *Adversarial, Quasi-Static Queuing Theory* model, where each link capacity may take on only two possible (integer) values, namely 1 and $C > 1$ under a (w, ρ)-*adversary*. We obtain the following results:

- Allowing such dynamic changes to the link capacities of a network with just ten nodes that uses the LIS (*Longest-in-System*) protocol for contention–resolution results in instability at rates $\rho > \sqrt{2} - 1$ and for large enough values of C.
- The combination of dynamically changing link capacities with compositions of contention–resolution protocols on network queues suffices for similar instability bounds: The composition of LIS with any of SIS (*Shortest-in-System*), NTS (*Nearest-to-Source*), and FTG (*Furthest-to-Go*) protocols is unstable at rates $\rho > \sqrt{2} - 1$ for large enough values of C.
- The instability bound of the network subgraphs that are forbidden for stability is affected by the dynamic changes to the link capacities: we present improved instability bounds for all the *directed subgraphs* that were known to be *forbidden* for stability on networks running a certain greedy protocol.

Keywords: Adversarial Queuing Theory; Adversarial Quasi-Static Queuing Theory; Network stability; Greedy protocols

☆ Part of this work appeared in the Proceedings of the 10th International Colloquium on Structural Information and Communication Complexity, Umeå, Sweden, 2003, pp. 179–194. This work has been partially supported by the 6th Framework Programme of the European Union under contract number 001907 (DELIS).

* Corresponding address: RACTI, RU1, George Souri 4, 30100 Agrinio, Aitoloakarnania, Greece. Tel.: +30 2641034473; fax: +30 2641058075.
E-mail addresses: koukopou@ceid.upatras.gr (D. Koukopoulos), mavronic@ucy.ac.cy (M. Mavronicolas), spirakis@cti.gr (P. Spirakis).

0304-3975/$ - see front matter © 2007 Elsevier B.V. All rights reserved.
doi:10.1016/j.tcs.2007.04.008

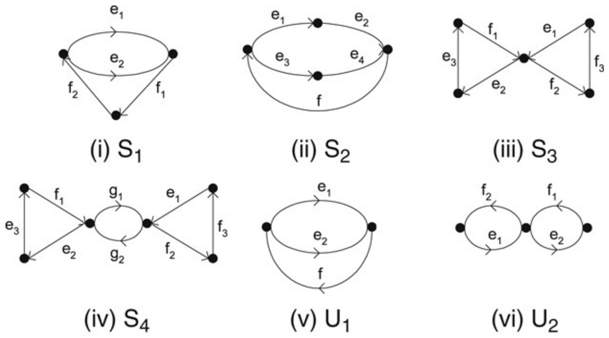

Fig. 1. The edge-simple and node-simple networks $\mathcal{S}_1, \mathcal{S}_2, \mathcal{S}_3, \mathcal{S}_4$, and the edge-simple but not node-simple networks $\mathcal{U}_1, \mathcal{U}_2$.

1. Introduction

1.1. Motivation and framework

Objectives. We are interested in the behavior of *packet-switched networks*, in which (unit-sized) packets arrive dynamically at the *nodes* and they are routed in discrete time steps across the *links*. Recent years have witnessed a vast amount of work on analyzing packet-switched networks under *non-probabilistic* assumptions (rather than probabilistic ones); we work within a model of *worst-case* continuous packet arrivals, originally proposed by Borodin et al. [6] and termed *Adversarial Queuing Theory* to reflect the assumption of an *adversarial* mode of packet generation and path determination.

A major issue that arises in such a setting is that of *stability*—will the number of packets in the network remain bounded at all times? The answer to this question may depend on the *rate* of injecting packets into the network, the *capacity* of the links, the *speed* at which a link forwards outgoing packets, and the *protocol* that is used to resolve the conflict when more than one packet wants to cross a given link in a single time step. The underlying goal of our study is to establish the stability properties of networks and protocols when packets are injected by an adversary and the link capacities are chosen by the same adversary in a dynamic way.

Model of Quasi-Static capacities. Most studies of packet-switched networks have assumed that one packet can cross a network link (an edge) in a single time step. This assumption is well motivated when all network links are identical. However, a packet-switched network can involve many different types of links, and this is common, especially in large-scale networks like the Internet. So, we model this by assigning a *capacity* to each link. We assume that each link capacity takes on values in the two-valued set of integers $\{1, C\}$ for $C > 1$, and each value remains fixed for a long time. This assumption reflects the situation where links are up and down; 1 corresponds to the times where the link is down and C corresponds to the times where the link is up. We chose to use C and 1 (instead of 1 and 0) to model links that go up and down, respectively since C and 1 provide a basis for the comparison of the corresponding rates; indeed, we use 1 to model the very slow rate of a link that goes down, and we use $C > 1$ to model the normal rate. Using C and 1 provides a continuum for the comparison of fast and slow rates, while using 1 and 0 corresponds to just one case of this continuum. So, using C and 1 provides the basis for a tractable stability analysis of a communication network with transient failures.

We consider the impact on the stability behavior of protocols and networks where the adversary, in addition to determining packet injection and paths, also sets the link capacities in each time step. This particular model was initiated by Borodin et al. in [7].

Stability. Roughly speaking, a protocol P is *stable* [6] on a network $\mathcal{G}$ against an adversary $\mathcal{A}$ of rate ρ if there is a constant B (which may depend on $\mathcal{G}$ and $\mathcal{A}$) such that the number of packets in the system is bounded at all times by B. On the other hand, *a protocol* P *is universally stable* [6] if it is stable against every adversary of rate less than 1 and on every network. We also say that *a network* $\mathcal{G}$ *is universally stable* [6] if every greedy protocol is stable against every adversary of rate less than 1 on $\mathcal{G}$. Consider the graphs $\mathcal{S}_1, \mathcal{S}_2, \mathcal{S}_3, \mathcal{S}_4, \mathcal{U}_1$ and $\mathcal{U}_2$ in Fig. 1. It has been shown in [2,9] that these graphs are *forbidden subgraphs* for stability: a network containing any of them as a subgraph is not universally stable.

Table 1
Greedy protocols considered in this paper

Protocol name	Which packet it advances:	US
SIS (*Shortest-In-System*)	The most recently injected packet	√ [1]
LIS (*Longest-In-System*)	The least recently injected packet	√ [1]
FTG (*Furthest-To-Go*)	The furthest packet from its destination	√ [1]
NTS (*Nearest-To-Source*)	The nearest packet to its origin	√ [1]
NTG.LIS (*Nearest-To-Go.LIS*)	The nearest packet to its destination or the one determined by LIS for tie-breaking	X [1,2]

(US stands for universally stable).

Table 2
Instability bounds of forbidden subgraphs in AQT (Adversarial Queuing Theory model) vs. AQSQT (Adversarial Quasi-Static Queuing Theory model)

	Apply to:	Instability (AQM)	Instability (AQSQM)
$\mathcal{S}_1$	Node s.p.	$\rho > 0.87055$ [2, Lemma 12]	$\rho > 0.8191$ [Theorem 5]
$\mathcal{S}_2$	Node s.p.	$\rho > 0.84089$ [2, Lemma 13]	$\rho > 0.8191$ [Theorem 6]
$\mathcal{S}_3$	Node s.p.	$\rho > 0.84089$ [2, Lemma 14]	$\rho > 0.8191$ [Theorem 6]
$\mathcal{S}_4$	Node s.p.	$\rho > 0.84089$ [2, Lemma 15]	$\rho > 0.8191$ [Theorem 6]
$\mathcal{U}_1$	Edge s.p.	$\rho > 0.84089$ [2, Theorem 3]	$\rho > 0.794$ [Theorem 7]
$\mathcal{U}_2$	Edge s.p.	$\rho > 0.84089$ [2, Lemma 9]	$\rho > 0.754$ [Theorem 8]

Greedy protocols. We consider five *greedy* contention–resolution protocols — ones that always advance a packet across a link (but one packet at each discrete time step) whenever there is at least one packet in the queue (Table 1).

1.2. Contribution

Within our model of Adversarial, Quasi-Static Queuing Theory, we present the following results:

- We construct a simple LIS network of only 10 nodes that is unstable at rates $\rho > \sqrt{2} - 1$ for large enough values of C (Theorem 1). This is the first *result* that provides an instability bound on the injection rate less than $1/2$ for a small-size network. Until now, instability bounds of $1/2$ or less have been proved only for families of networks whose size is determined by a parameter [14,5].
- We consider networks where different protocols may run on their nodes; these networks are called heterogeneous, with Internet being the prime example. We prove that the composition of the LIS protocol with any of SIS, NTS and FTG is unstable at rates $\rho > \sqrt{2} - 1$ (for large enough values of C) (Theorems 2–4). To show this, we provide interesting combinatorial constructions of heterogeneous networks.
- Through involved adversarial constructions, we present instability bounds for networks containing each of the forbidden subgraphs in Fig. 1 (Theorems 5–8). These bounds improve upon the corresponding state-of-the-art bounds for the standard AQT model. For the purposes of comparison, we summarize in Table 2 all the results that are shown in this work and in [2] concerning instability bounds on the injection rate for the forbidden subgraphs ($\mathcal{S}_1, \mathcal{S}_2, \mathcal{S}_3, \mathcal{S}_4, \mathcal{U}_1$ and $\mathcal{U}_2$).

1.3. Related work

Adversarial Queuing Model. The Adversarial Queuing Theory model has received a lot of interest and attention in the study of stability and instability issues (see, e.g., [1,2,5,8,11,13,15]). The universal stability of various natural greedy protocols (SIS, LIS, NTS and FTG) was established by Andrews et al. [1]. Also, several greedy protocols such as NTG (Nearest-To-Go) have been proved unstable at arbitrarily small rates of injection in [15].

Stability in heterogeneous networks. The study of stability properties of compositions of universally stable protocols was initiated by Koukopoulos et al. in [11,13,12] where lower bounds of 0.683, 0.519, and 0.5 on the injection rates that guarantee instability for the composition pairs LIS-SIS, LIS-NTS and LIS-FTG were presented.

Instability of forbidden subgraphs. Álvarez et al. [2, Theorems 8, 12] gave a characterization for the universal stability of directed networks when the packets follow edge-simple paths (with no repetition of edges) that are either node-simple (that is, they have no repetition of nodes) or not.[1] According to this characterization, a directed network graph is not edge-simple path universally stable if and only if it does contain as subgraphs any of the extensions of the subgraphs $\mathcal{U}_1$ or $\mathcal{U}_2$ [2, Theorem 8]; it is not node-simple path universally stable if and only if it does contain as subgraphs any of the extensions of the subgraphs $\mathcal{S}_1$ or $\mathcal{S}_2$ or $\mathcal{S}_3$ or $\mathcal{S}_4$ [2, Theorem 12] (see Fig. 1).

Stability issues in dynamic networks. Borodin et al. [7] studied for the first time the impact on stability when the edges of a network may have different capacities. They proved that the universal stability of networks is preserved in this context. Also, it was shown that many well-known universally stable protocols (SIS, NTS, FTG) maintain their universal stability, whereas the universal stability of LIS is not preserved. More specifically Borodin et al. [7, Theorem 1] presented an instability bound of $\rho > C/(2C-1) > 0.5$ for the LIS protocol.

1.4. Road map

The rest of this paper is organized as follows. Section 2 presents model definitions. Section 3 presents our instability bound for LIS. Section 4 demonstrates instability bounds for protocol compositions. Section 5 shows instability bounds for forbidden subgraphs. We conclude, in Section 6, with a discussion of our results and some open problems.

2. The Adversarial Quasi-Static Queuing Theory model

The model definitions are patterned after those in [6, Section 3]; they are adjusted to reflect the fact that the edge capacities may vary arbitrarily as in [7, Section 2].

A network is a directed graph $\mathcal{G} = (V, E)$. Each node $u \in V$ represents a communication switch, and each edge $e \in E$ represents a link between two switches. In each node, there is a buffer (queue) associated with each outgoing link. Time proceeds in discrete time steps. Buffers store packets that are injected into the network with a route, which is a simple directed path in $\mathcal{G}$.

A *packet* is an atomic entity that resides at a buffer at the end of each step. It must travel along a path in the network from its *source* to its *destination,* both of which are nodes. When a packet is injected, it is placed in the buffer of the first link on its route. When a packet reaches its destination, it is *absorbed*. During each step, a packet may be sent from its current node along one of the outgoing edges from that node. Edges can have different integer capacities, which may or may not vary over time. Denote $\mathsf{C}_e(t)$ the *capacity* of the edge e at time step t. That is, we assume that the edge e is capable of simultaneously transmitting up to $\mathsf{C}_e(t)$ packets at time t.

Let $C > 1$ be an integer parameter. We demand that for all the edges e and all times, t $C_e(t) \in \{1, C\}$ (i.e. each edge capacity can get only two values, high and low).

Any packets that wish to travel along an edge e at a particular time step, but are not sent, wait in a queue for edge e. The *delay* of a packet is the number of steps that are spent by the packet while waiting in queues.

At each step, an *adversary* generates a set of requests. A *request* is a *path* specifying the route that will be followed by a packet. We only consider edge-simple paths, where no edge is repeated. We will mostly consider node-simple paths, where no vertex is repeated. (A single exception to this is a construction in Section 5.)

We say that the adversary generates a set of packets when it generates a set of requested paths. Also, we say that a packet p *requires* an edge e at time t if the edge e lies on the path from its position to its destination at time t. We restrict our study to *non-adaptive* routing (or *source* routing), where the path to be traversed by each packet is fixed at the time of injection; in this way, we are able to focus on queuing rather than routing aspects of the problem. (See [3, 4] for an extension of the adversarial model to the case of *adaptive* routing.) There are no computational restrictions on how the adversary chooses its requests at any given time step.

Fix any arbitrary positive integer $w \geq 1$. For any edge e of the network and any sequence of w consecutive time steps, define $N(w, e)$ to be the number of packets injected by the adversary during this time interval of w consecutive time steps needing to traverse the edge e. For any constant ρ, $0 < \rho \leq 1$, a (w, ρ)-*adversary* is an adversary that injects packets subject to the following *load condition*: For every edge e and for every sequence τ of w consecutive

[1] A corresponding characterization for the stability of undirected networks was shown in [1, Theorem 3.16].

time steps, $N(\tau, e) \leq \rho \sum_{t\in\tau} \mathsf{C_e}(t)$. We say that a (w, ρ)-adversary injects packets at rate ρ with *window size* w. The assumption that $\rho \leq 1$ ensures that it is not necessary a priori that some edge of the network becomes overloaded (which happens surely if $\rho > 1$).

A *contention–resolution* protocol specifies, for each pair of an edge e and a time step t, which packet(s) among those waiting at the tail of edge e at time step e will be moved along the edge e. Denote by $W(t, e)$ the number of packets waiting at the tail of edge e at time t. A *greedy* (or *work-conserving*) contention–resolution protocol always specifies $\min\{W(t, e), C_e(t)\}$ packets to move along the edge e. We consider the following (deterministic) greedy protocols:

- SIS (*Shortest-in-System*) gives priority to the most recently injected packet into the network;
- LIS (*Longest-in-System*) gives priority to the least recently injected packet into the network;
- FTG (*Furthest-to-Go*) gives priority to the packet that has to traverse the larger number of edges to its destination;
- NTS (*Nearest-to-Source*) gives priority to the packet that has traversed the smallest number of edges from its origin;
- NTG.LIS (*Nearest-To-Go.LIS*) gives priority to the nearest packet to its destination or the least recently injected packet for tie-breaking.

All these contention–resolution protocols require some tie-breaking rule in order to be unambiguously defined. We will assume any well-determined tie breaking rule for the adversary.

A system is a triple of the form $\langle \mathcal{G}, \mathcal{A}, \mathsf{P}\rangle$ where $\mathcal{G}$ is a network, $\mathcal{A}$ is a (w, ρ)-adversary and P is the used protocol on the network queues. The execution of the system proceeds in global time steps numbered $0, 1, \ldots$. Each time-step t is divided in two sub-steps. In the first sub-step $\min\{W(t, e), C_e(t)\}$ packets are sent from each non-empty buffer at edge e over its corresponding link. In the second sub-step, packets are received by the nodes at the other end of the links; they are absorbed (eliminated) if that node is their destination, or otherwise they are placed in the buffer of the next link on their respective routes. New packets are injected by the adversary in the second sub-step.

At every time step t, the current configuration Q^t of the system $\langle \mathcal{G}, \mathcal{A}, \mathsf{P}\rangle$ is a collection of sets $\{S_e^t : e \in E\}$, where S_e^t is the set of packets waiting in the queue of the edge e at the end of step t. We obtain the configuration Q^{t+1} from Q^t as follows:

- New packets are added to some of the sets S_e^t, each of which has an assigned path in $\mathcal{G}$.
- For each set S_e^t, $\min\{W(t, e), C_e(t)\}$ packets are deleted from S_e^t; each such packet p is inserted into S_f^{t+1} where f is the edge followed by the packet on its assigned path. (If e is the last edge on the path of p, then p is not inserted into any set.)

A time evolution of the system is a sequence of such configurations $Q^1, Q^2, \ldots$.

In our adversarial constructions, we split time into *phases*. Within each phase, we consider corresponding *time rounds*. For each phase, we inductively prove that the number of packets of a specific subset of queues in the system increases. This inductive argument will suffice to imply instability.

We will assume that there are a sufficiently large number of packets s_0 in the initial configuration. By [1, Lemma 2.9] this will imply identical instability bounds for networks with an *empty* initial configuration. For simplicity and in a way similar to [1] and in works following it, we omit floors and ceilings from our analysis, and we often count time steps and packets only roughly. This may only result to losing small additive constants, while it implies a gain in clarity.

3. Instability bound for LIS

In this section, we present an instability bound for the LIS protocol on the network $\mathcal{N}$ in Fig. 2. We show:

Theorem 1. *For the network $\mathcal{N}$, there is a (w, ρ)-adversary $\mathcal{A}$ such that the system $\langle \mathcal{N}, \mathcal{A}, \mathsf{LIS}\rangle$ is unstable for $\rho = 0.422$ and $C \geq 100$. When $C \to \infty$, the system $\langle \mathcal{N}, \mathcal{A}, \mathsf{LIS}\rangle$ is unstable for $\rho > \sqrt{2} - 1$.*

Proof. The construction of the adversary $\mathcal{A}$ is broken into phases.

Inductive hypothesis: At the beginning of phase j, there are in total s_j packets queued in the queues f_1', f_4', f_5', f_7' and needing to traverse the edges e_0, f_2, f_4.

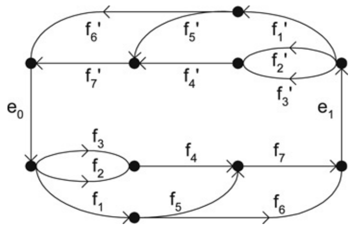

Fig. 2. The network $\mathcal{N}$.

Induction step: We will prove that at the beginning of phase $j+1$, there will be in total $s_{j+1} > s_j$ packets queued in the queues f_1, f_4, f_5, f_7 and needing to traverse the edges $e_1, f_2^{'}, f_4^{'}$.

We start with an informal description of the construction of the adversary $\mathcal{A}$. The main ideas are:

- An accurate tuning of the duration of each round of every phase j (as a function of the capacity C, the injection rate ρ and the number of packets s_j in the system at the beginning of phase j) in order to maximize the growth of the packet population in the system.
- A careful setting of the capacities of some edges to 1 for specified time intervals in order to accumulate packets.
- A careful injection of packets in order to guarantee that the load condition is not violated. In particular, when packets are injected into different queues simultaneously, we assign them to paths that do not overlap in order to preserve the load condition.

During phase j, the adversary uses three rounds of injections:

- **Round 1:** It lasts $|T_1| = s_j/C$ time steps.

  During Round 1, the edges $f_1^{'}, f_2^{'}, f_4^{'}, f_5^{'}, f_7^{'}, e_0, f_1, f_5, f_7, e_1$ have capacity C, while all the other edges have capacity 1. $\mathcal{A}$ injects a set X with $\rho C|T_1|$ packets in the queue e_0 needing to traverse the edges $e_0, f_1, f_5, f_7, e_1, f_2^{'}, f_4^{'}$ and a set S_1 of $|S_1| = \rho|T_1|$ packets in the queue f_2 needing to traverse the edges f_2, f_4. These injections satisfy the load condition because the edges $e_0, f_1, f_5, f_7, e_1, f_2^{'}, f_4^{'}$ have capacity C and the edges f_2, f_4 have capacity 1 during Round 1, and the paths of the injected packet are edge-disjoint.

  The packets in the set S delay the packets in the set X in the queue e_0, and the packets in the set S_1 in the queue f_2 because they are in the system for a longer time than the packets in the sets X and S_1. At the same time, the packets of the set S are delayed in f_2 due to the unit capacity of the edge f_2. At the end of Round 1, the remaining packets from the set S in f_2 form a set $S^{'}$ with $|S| - |T_1|$ packets. The packets in the set S that manage to traverse the edge f_2 continue traversing their remaining path and they are absorbed. So, the number of packets in the queue f_2 at the end of Round 1 needing to traverse the edges f_2 and f_4 form a set S_2 with $|S^{'}| + |S_1|$ packets.
- **Round 2:** It lasts $|T_2| = |S_2|/C$ time steps.

  During Round 2, the edges $f_2, f_4, f_7, e_0, e_1, f_2^{'}, f_4^{'}$ have capacity C, while all the other edges have capacity 1. $\mathcal{A}$ injects a set Y with $\rho C|T_2|$ packets in the queue f_4 needing to traverse the edges $f_4, f_7, e_1, f_2^{'}, f_4^{'}$. These injections satisfy the load condition because all the edges in their assigned paths have capacity C during Round 2.

  The packets in the set Y are delayed by the packets in the set S_2 in the queue f_4, because the latter are in the system longer than the former. The packets in the set S_2 traverse the edge f_4 and they are absorbed. At the same time, the packets in the set X are delayed in the queue f_1 due to its unit capacity. So, the remaining packets from the set X in the queue f_1 form a set $X^{'}$ with $|X| - |T_2|$ packets.
- **Round 3:** It lasts $|T_3| = |X^{'}|/C$ time steps.

  During Round 3, the edges $f_1, f_6, e_1, f_2^{'}, f_4^{'}$ have capacity C, while all the other edges have capacity 1. $\mathcal{A}$ injects a set Z with $\rho C|T_3|$ packets in the queue f_1 needing to traverse the edges $f_1, f_6, e_1, f_2^{'}, f_4^{'}$. These injections satisfy the load condition, because all the edges in their assigned path have capacity C during Round 3.

  Note that the packets in the set $X^{'}$ delay the packets in the set Z in the queue f_1, because they are in the system longer than the latter. At the same time, the packets in the set $X^{'}$ are delayed in f_5 due to the unit capacity of the edge f_5. So, the remaining packets of the set $X^{'}$ in the queue f_5 form a set $X^{''}$ with $|X^{'}| - |T_3|$ packets. Moreover, the packets in the set Y are delayed in f_4 due to the unit capacity of the edge f_4 Round 3. So, the remaining packets of the set Y in the queue f_4 is a set $Y^{'}$ with $|Y| - |T_3|$ packets.

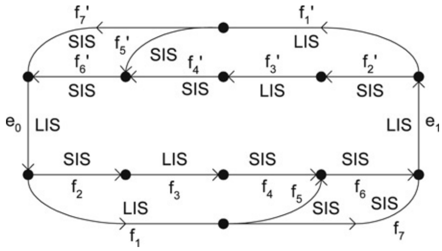

Fig. 3. The network $\mathcal{G}_1$.

Note that during Round 3, $|K| = 2|T_3|$ packets arrive in the queue f_7 from the queues f_4, f_5. Since the edge f_7 has capacity 1 and the duration of Round 3 is $|T_3|$ time steps, it follows that at the end of this Round 3, $|L| = |T_3|$ packets will remain in the queue f_7 needing to traverse the edges $f_7, e_1, f_2^{'}, f_4^{'}$. So, the number of packets in the queues f_1, f_4, f_5, f_7 needing to traverse the edges $e_1, f_2^{'}, f_4^{'}$ at the end of Round 3 is $s_{j+1} = |X^{''}|+|Y^{'}|+|Z|+|L|$. Substituting the quantities $|X^{''}|$, $|Y^{'}|$, $|Z|$ and $|L|$, we obtain that

$$s_{j+1} = \rho s_j - \frac{1+\rho}{C}s_j + \frac{2-\rho}{C^2}s_j + \frac{\rho-1}{C^3}s_j + \rho s_j + \frac{\rho^2-2\rho}{C}s_j + \frac{1}{C^2}s_j + \frac{\rho-1}{C^3}s_j + \rho^2 s_j - \frac{\rho}{C}s_j + \frac{\rho-\rho^2}{C^2}s_j + \frac{\rho}{C}s_j - \frac{1}{C^2}s_j + \frac{1-\rho^2}{C^3}s_j.$$

In order to have instability, we must have $s_{j+1} > s_j$, that is $\rho^2[1+\frac{1}{C}-\frac{1}{C^2}]+\rho[2-\frac{3}{C}+\frac{1}{C^3}]+[-\frac{1}{C}+\frac{2}{C^2}-\frac{1}{C^3}] > 1$. For $C = 100$ and $\rho = 0.422$, the inequality holds. Hence, for $C \geq 100$ and $\rho = 0.422$, the claim follows.

When $C \to \infty$, $\frac{1}{C^k} \to 0$ for all $k \geq 1$. So, the inequality $\rho^2[1+\frac{1}{C}-\frac{1}{C^2}]+\rho[2-\frac{3}{C}+\frac{1}{C^3}]+[-\frac{1}{C}+\frac{2}{C^2}-\frac{1}{C^3}] > 1$ becomes $\rho^2+2\rho-1 > 0$, which holds for $\rho > \sqrt{2}-1$. Hence, for $C \to \infty$ and $\rho > \sqrt{2}-1$, the claim follows. □

4. Instability bounds for protocol compositions

In this section, we present instability bounds for protocol compositions.

First, we show an instability bound for the composition of LIS and SIS protocols on the network $\mathcal{G}_1$ in Fig. 3. The edges $e_0, e_1, f_1, f_1^{'}, f_3, f_3^{'}$ use the LIS protocol, while the remaining edges use the SIS protocol.

Theorem 2. *For the network $\mathcal{G}_1$ there is a (w, ρ)-adversary $\mathcal{A}_1$ such that the system $\langle \mathcal{G}_1, \mathcal{A}_1, (\mathsf{LIS}, \mathsf{SIS})\rangle$ is unstable for $\rho = 0.4209$ and $C \geq 100$. When $C \to \infty$, the system $\langle \mathcal{G}_1, \mathcal{A}_1, (\mathsf{LIS}, \mathsf{SIS})\rangle$ is unstable for $\rho > \sqrt{2}-1$.*

Proof. The construction of the adversary $\mathcal{A}_1$ is broken into phases.

Inductive hypothesis: At the beginning of phase j, there are in total s_j packets queued in the queues $f_1^{'}, f_4^{'}, f_5^{'}, f_6^{'}$ and needing to traverse the edges e_0, f_2, f_3, f_4, respectively. Denote as S the set of these packets.

Induction step: We will prove that at the beginning of phase $j+1$, there will be in total $s_{j+1} > s_j$ packets queued in the queues f_1, f_4, f_5, f_6 and needing to traverse the edges $e_1, f_2^{'}, f_3^{'}, f_4^{'}$.

We will construct an adversary $\mathcal{A}_1$ such that the induction step will hold. During phase j, the adversary uses three rounds of injections as follows:

- **Round 1:** It lasts $|T_1| = s_j/C$ time steps.

  During this round, the edge f_2 has capacity , while all the other edges have capacity C. Also, the adversary injects a set X with $\rho C|T_1|$ packets in the queue e_0 needing to traverse the edges $e_0, f_1, f_5, f_6, e_1, f_2^{'}, f_3^{'}, f_4^{'}$, and a set S_1 with $\rho|T_1|$ packets in the queue f_2 needing to traverse the edge f_2. These injections satisfy the load condition because the edges $e_0, f_1, f_5, f_6, e_1, f_2^{'}, f_3^{'}, f_4^{'}$ have capacity C and the edge f_2 has capacity 1 during Round 1, and the paths of the injected packets are edge-disjoint.

  Note that the packets in S delay the packets in X in the queue e_0 that uses the LIS protocol, because they are present for a longer period in the system. The packets in S are delayed in the queue f_2 that uses the SIS protocol due to the packets in S_1, which are present for a shorter time in the system than the packets in S, and the unit capacity of the edge f_2. At the end of Round 1, the remaining packets from S in f_2 are a set S_2 with $|S| - (|T_1| - |S_1|)$ packets. The packets in S that manage to traverse the edge f_2, will traverse their remaining path and be absorbed.

- **Round 2:** It lasts $|T_2| = |S_2|/C$ time steps.

  During this round, the edge f_1 has capacity 1, while all the other edges have capacity C. $\mathcal{A}_1$ injects a set Y with $\rho C|T_2|$ packets in the queue f_3 needing to traverse the edges $f_3, f_4, f_6, e_1, f_2^{'}, f_3^{'}, f_4^{'}$. These packet injections satisfy the load condition because all the edges in their assigned path have capacity C during Round 2.

  Note that the packets in S_2 delay the packets of the set Y in the queue f_3 that uses the LIS protocol because they are in the system longer. The packets in S_2 traverse the edge f_3 and they are absorbed. The packets in X are delayed in the queue f_1 due to the unit capacity of the edge f_1. So, at the end of Round 2, the remaining packets from X in the queue f_1 form a set $|X^{'}|$ with $|X| - |T_2|$ packets.

- **Round 3:** It lasts $|T_3| = |X^{'}|/C$ time steps.

  During Round 3, the edges f_4, f_5, f_6 have capacity 1, while all the other edges have capacity C. $\mathcal{A}_1$ injects a set Z with $\rho C|T_3|$ packets in the queue f_1 needing to traverse the edges $f_1, f_7, e_1, f_2^{'}, f_3^{'}, f_4^{'}$. Also, $\mathcal{A}_1$ injects a set S_3 with $\rho|T_3|$ packets in the queue f_4 needing to traverse the edge f_4, a set S_4 with $\rho|T_3|$ packets in the queue f_5 needing to traverse the edge f_5, and a set S_5 with $\rho|T_3|$ packets in the queue f_6 needing to traverse the edge f_6. These injections satisfy the load condition because the edges $f_1, f_7, e_1, f_2^{'}, f_3^{'}, f_4^{'}$ have capacity C and the edges f_4, f_5, f_6 have capacity 1 during this round, while the injection paths of the different packet sets are edge-disjoint.

  Note that the packets in $X^{'}$ delay the packets in Z in the queue f_1 that uses the LIS protocol because they are in the system longer. Note also that the packets in $X^{'}$ are delayed in the queue f_5 that uses the SIS protocol due to the unit capacity of the edge f_5 during Round 3 and the packets in S_4 that are shorter in the system. So, the remaining packets of the set $X^{'}$ in the queue f_5 form a set $X^{''}$ with $|X^{'}| - (|T_3| - |S_4)|$ packets. Moreover, the packets in Y are delayed in the queue f_4 that uses the SIS protocol due to the unit capacity of the edge f_4 during Round 3 and the packets of the set S_3 that are in the system for a shorter period. So, the remaining packets of the set Y in the queue f_4 form a set $Y^{'}$ with $|Y| - (|T_3| - |S_3|)$ packets.

  Note that during Round 3, $|K| = 2|T_3| - |S_3| - |S_4|$ packets arrive in the queue f_6 from the queues f_4, f_5. Since, the edge f_6 has capacity 1 and uses the SIS protocol, it will give priority to the packets in the set S_5. Since Round 3 goes for $|T_3|$ steps, at the end of Round 3, the number of packets that remain in the queue f_6 needing to traverse the edges $f_6, e_1, f_2^{'}, f_3^{'}, f_4^{'}$ is $|L| = |K| + |S_5| - |T_3|$. So, the number of packets in the queues f_1, f_4, f_5, f_6 needing to traverse the edges $e_1, f_2^{'}, f_3^{'}, f_4^{'}$ at the end of Round 3 is $s_{j+1} = |X^{''}| + |Y^{'}| + |Z| + |L|$. Substituting the quantities $|X^{''}|, |Y^{'}|, |Z|$ and $|L|$, we take

$$\begin{aligned} s_{j+1} &= \rho s_j + \frac{\rho^2 - \rho - 1}{C} s_j + 2\frac{1-\rho}{C^2} s_j + \frac{-\rho^2 + 2\rho - 1}{C^3} s_j + \rho s_j \\ &+ \frac{2\rho^2 - 2\rho}{C} s_j + \frac{1-\rho}{C^2} s_j + \frac{-\rho^2 + 2\rho - 1}{C^3} s_j + \rho^2 s_j - \frac{\rho}{C} s_j \\ &+ \frac{\rho - \rho^2}{C^2} s_j + \frac{\rho - \rho^2}{C} s_j + 2\frac{\rho - 1}{C^2} s_j + \frac{\rho^2 - 2\rho + 1}{C^3} s_j. \end{aligned}$$

  For instability, it must be that $s_{j+1} > s_j$, that is $\rho^2[1 + \frac{2}{C} - \frac{1}{C^2} - \frac{1}{C^3}] + \rho[2 - \frac{3}{C} - \frac{1}{C^2} + \frac{2}{C^3}] + [-\frac{1}{C} + \frac{2}{C^2} - \frac{1}{C^3}] > 1$. If we let $C = 100$ and $\rho = 0.4209$, the inequality holds. Hence, for $C \geq 100$ and $\rho = 0.4209$ the claim follows.

  When $C \to \infty$, it holds that $\frac{1}{C^k} \to 0$ for all $k \geq 1$. So, the inequality $\rho^2[1 + \frac{2}{C} - \frac{1}{C^2} - \frac{1}{C^3}] + \rho[2 - \frac{3}{C} - \frac{1}{C^2} + \frac{2}{C^3}] + [-\frac{1}{C} + \frac{2}{C^2} - \frac{1}{C^3}] > 1$ becomes $\rho^2 + 2\rho - 1 > 0$, which holds for $\rho > \sqrt{2} - 1$. Hence, the claim follows for $C \to \infty$ and $\rho > \sqrt{2} - 1$. □

Now, consider the network $\mathcal{G}_1$ in Fig. 3 to show an instability bound for the composition of LIS and NTS protocols. Similarly, to Theorem 2, we can prove Theorem 3 [10]. The queues $f_2, f_2^{'}, f_4, f_4^{'}, f_5, f_5^{'}, f_6, f_6^{'}, f_7, f_7^{'}$ use the NTS protocol, while the remaining queues of $\mathcal{G}_1$ use the LIS protocol. We still consider that each phase consists of three distinguished time rounds. The *inductive argument* states that if at the beginning of a phase j, there are s_j packets in the queues $f_1^{'}, f_4^{'}, f_5^{'}, f_6^{'}$ needing to traverse the edges e_0, f_2, f_3, f_4, then at the beginning of phase $j+1$ there will be more than s_j packets in the queues f_1, f_4, f_5, f_6 needing to traverse the edges $e_1, f_2^{'}, f_3^{'}, f_4^{'}$. We obtain:

Theorem 3. *For the network $\mathcal{G}_1$, there is an adversary $\mathcal{A}_2$ of rate ρ such that the system $\langle \mathcal{G}_1, \mathcal{A}_2, (\mathsf{LIS}, \mathsf{NTS})\rangle$ is unstable for $\rho = 0.4209$ and $C \geq 100$. When $C \to \infty$, the system $\langle \mathcal{G}_1, \mathcal{A}_2, (\mathsf{LIS}, \mathsf{NTS})\rangle$ is unstable for $\rho > \sqrt{2} - 1$.*

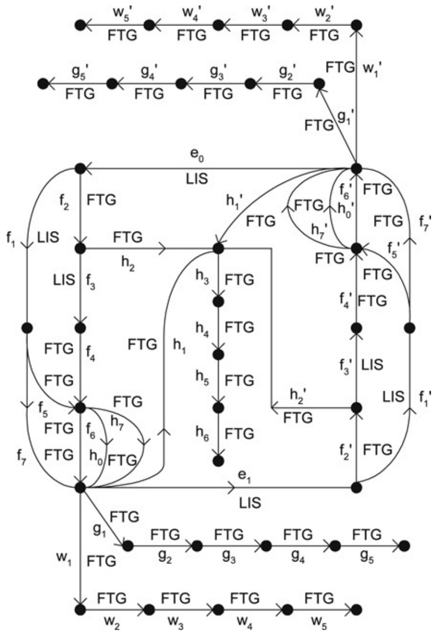

Fig. 4. The network $\mathcal{G}_3$.

We continue to show an instability bound for the composition of LIS and FTG protocols on the network $\mathcal{G}_3$ in Fig. 4. The edges $e_0, e_1, f_1, f_1', f_3, f_3'$ of $\mathcal{G}_3$ use the LIS protocol, while the remaining edges use the FTG protocol. We show:

Theorem 4. *For the network $\mathcal{G}_3$, there is an adversary $\mathcal{A}_3$ of rate ρ such that the system $\langle \mathcal{G}_3, \mathcal{A}_3, (\mathsf{LIS}, \mathsf{FTG})\rangle$ is unstable for $\rho = 0.4209$ and $C \geq 100$. When $C \to \infty$ the system $\langle \mathcal{G}_3, \mathcal{A}_3, (\mathsf{LIS}, \mathsf{FTG})\rangle$ is unstable for $\rho > \sqrt{2} - 1$.*

Proof. We break the construction of the adversary $\mathcal{A}_3$ into phases.

Inductive hypothesis: At the beginning of phase j, there are in total s_j packets that are queued in the queues f_1', f_4', f_5', f_6' (in total) needing to traverse the edges e_0, f_2, f_3, f_4. Denote as S the set of these packets.

Induction step: We will prove that at the beginning of phase $j+1$, there will be in total $s_{j+1} > s_j$ packets that will be queued in the queues f_1, f_4, f_5, f_6 needing to traverse the edges e_1, f_2', f_3', f_4'.

We now construct an adversary $\mathcal{A}_3$ such that the induction step will hold. During phase j, the adversary uses three rounds of injections as follows:

- **Round 1:** It lasts $|T_1| = s_j/C$ time steps.

  During Round 1, the edge f_2 has capacity 1, while all the other edges have capacity C. $\mathcal{A}_3$ injects a set X with $\rho C|T_1|$ packets in the queue e_0 needing to traverse the edges $e_0, f_1, f_5, f_6, e_1, f_2', f_3', f_4'$ and a set S_1 with $\rho|T_1|$ packets in the queue f_2 wanting to traverse the edges f_2, h_2, h_3, h_4. These injections satisfy the load condition because the edges $e_0, f_1, f_5, f_6, e_1, f_2', f_3', f_4'$ have capacity C and the edge f_2 has capacity 1 during Round 1, while the injection paths of the different packet sets are edge-disjoint.

  Note that the packets in S delay the packets of the set X in the queue e_0 that uses the LIS protocol because they are in the system longer. Note also that the packets in S are delayed in the queue f_2 that uses the FTG protocol due to the packets in S_1 (which have furthest to go (to queue h_4) than the packets in S (to queue f_4)) and the unit capacity of the edge f_2. At the end of Round 2, the remaining packets of S in f_2 are $|S_2| = |S| - (|T_1| - |S_1|)$. The packets of S that manage to traverse the edge f_2 will traverse their remaining path and be absorbed.
- **Round 2:** It lasts $|T_2| = |S_2|/C$ steps.

  During Round 2, the edge f_1 has capacity 1, while all the other edges have capacity C. $\mathcal{A}_3$ injects a set Y with $\rho C|T_2|$ packets in the queue f_3 needing to traverse the edges $f_3, f_4, f_6, e_1, f_2', f_3', f_4'$. These injections satisfy the load condition because all the edges in the assigned path have capacity C during Round 3.

Note that the packets of S_2 delay the packets of the set Y in the queue f_3 that uses the LIS protocol because they are in the system longer. Note also that the packets of S_2 traverse the edge f_3 and they are absorbed. Finally, note that the packets of X are delayed in the queue f_1 due to the unit capacity of the edge f_1. So, the remaining packets of X in the queue f_1 at the end of Round 2 form a set X' with $|X| - |T_2|$ packets.

- **Round 3:** It lasts $|T_3| = |X'|/C$ time steps.

During Round 3, the edges f_4, f_5, f_6 have capacity 1, while all the other edges have capacity C. $\mathcal{A}_3$ injects a set Z with $\rho C|T_3|$ packets in the queue f_1 needing to traverse the edges $f_1, f_7, e_1, f_2', f_3', f_4'$, a set S_3 with $\rho|T_3|$ packets in the queue f_4 needing to traverse the edges $f_4, h_0, h_1, h_3, h_4, h_5, h_6$, a set S_4 with $\rho|T_3|$ packets in the queue f_5 needing to traverse the edges $f_5, h_7, g_1, g_2, g_3, g_4, g_5$ and a set S_5 with $\rho|T_3|$ packets in the queue f_6 needing to traverse the edges $f_6, w_1, w_2, w_3, w_4, w_5$.

These injections satisfy the load condition because the edges $f_1, f_7, e_1, f_2', f_3', f_4'$ have capacity C and the edges f_4, f_5, f_6 have capacity 1 during Round 3, and the paths of the injected packet sets are edge-disjoint.

Note that the packets of X' delay the packets of the set Z in the queue f_1 that uses the LIS protocol because they are in the system longer. Note also that the packets of X' are delayed in the queue f_5 that uses the FTG protocol due to the unit capacity of the edge f_5 during Round 3 and the packets of S_4 that have the furthest to go (to queue g_5) than the packets of the set X' (to queue f_4'). So, the remaining packets of X' in the queue f_5 form a set X'' with $|X'| - (|T_3| - |S_4|)$ packets. Finally, note that the packets of Y are delayed in the queue f_4 that uses the FTG protocol due to the unit capacity of the edge f_4 during Round 3 and the fact that the packets of S_3 have furthest to go (to queue h_6) than the packets of Y (to queue f_4'). So, the remaining packets of Y in the queue f_4 at the end of Round 3 form a set Y' with $|Y| - (|T_3| - |S_3|)$ packets.

Note that during Round 3, $|K| = 2|T_3| - |S_3| - |S_4|$ packets arrive at the queue f_6 from the queues f_4, f_5. Since the edge f_6 has capacity 1 and uses the FTG protocol, it will give priority to the packets of S_5. Since Round 3 lasts for $|T_3|$ time steps, it follows that at the end of Round 3, the number of packets that remain in the queue f_6 needing to traverse the edges $f_6, e_1, f_2', f_3', f_4'$ is $|L| = |K| + |S_5| - |T_3|$. Hence, the number of packets in the queues f_1, f_4, f_5, f_6 needing to traverse the edges e_1, f_2', f_3', f_4' at the end of Round 3 is $s_{j+1} = |X''| + |Y'| + |Z| + |L|$. Substituting the quantities $|X''|, |Y'|, |Z|$ and $|L|$, we take

$$\begin{aligned}
s_{j+1} &= \rho s_j + \frac{\rho^2 - \rho - 1}{C} s_j + 2\frac{1-\rho}{C^2} s_j + \frac{-\rho^2 + 2\rho - 1}{C^3} s_j + \rho s_j \\
&\quad + \frac{2\rho^2 - 2\rho}{C} s_j + \frac{1-\rho}{C^2} s_j + \frac{-\rho^2 + 2\rho - 1}{C^3} s_j + \rho^2 s_j - \frac{\rho}{C} s_j \\
&\quad + \frac{\rho - \rho^2}{C^2} s_j + \frac{\rho - \rho^2}{C} s_j + 2\frac{\rho - 1}{C^2} s_j + \frac{\rho^2 - 2\rho + 1}{C^3} s_j.
\end{aligned}$$

For instability, it must be that $s_{j+1} > s_j$ or $\rho^2[1 + \frac{2}{C} - \frac{1}{C^2} - \frac{1}{C^3}] + \rho[2 - \frac{3}{C} - \frac{1}{C^2} + \frac{2}{C^3}] + [-\frac{1}{C} + \frac{2}{C^2} - \frac{1}{C^3}] > 1$. If we let $C = 100$ and $\rho = 0.4209$, the inequality holds. Thus, for $C \geq 100$ and $\rho = 0.4209$ the inequality holds, too.

When $C \to \infty$, it holds that $\frac{1}{C^k} \to 0$ for all $k \geq 1$. Then, the inequality $\rho^2[1 + \frac{2}{C} - \frac{1}{C^2} - \frac{1}{C^3}] + \rho[2 - \frac{3}{C} - \frac{1}{C^2} + \frac{2}{C^3}] + [-\frac{1}{C} + \frac{2}{C^2} - \frac{1}{C^3}] > 1$ becomes $\rho^2 + 2\rho - 1 > 0$ which holds for $\rho > \sqrt{2} - 1$. □

5. Instability bounds for forbidden subgraphs

In this section, we present instability bounds for forbidden subgraphs. First, we consider the network $\mathcal{S}_1$ in Fig. 1 that uses the NTG.LIS protocol. We show:

Theorem 5. *For the network $\mathcal{S}_1$ there is an adversary $\mathcal{A}_1$ of rate ρ such that the system $\langle \mathcal{S}_1, \mathcal{A}_1, \text{NTG.LIS} \rangle$ is unstable for $\rho \geq 0.82$ and $C \geq 1000$. When $C \to \infty$, the system $\langle \mathcal{S}_1, \mathcal{A}_1, \text{NTG.LIS} \rangle$ is unstable for $\rho > 0.8191$.*

Proof. We break the construction of the adversary $\mathcal{A}_1$ into phases.

Inductive hypothesis: At the beginning of phase j, there are s_j packets in the queues e_1, e_2 needing to traverse the edge f_1.

Induction step: We will prove that at the beginning of phase $j+1$, there will be $s_{j+1} > s_j$ packets in the queues e_1, e_2 needing to traverse the edge f_1.

We now construct an adversary $\mathcal{A}_1$ such that the induction step will hold. During phase j, the adversary uses four rounds of injections as follows:

- **Round 1:** It lasts $|T_1| = s_j/C$ time steps.

  During Round 1, all edges have capacity C. $\mathcal{A}_1$ injects in the queue f_1 a set X with $\rho C|T_1|$ packets needing to traverse the edges f_1, f_2. These injections satisfy the load condition because all edges in the assigned path have capacity C during Round 1.

  Note that the packets in S delay the packets in X in the queue f_1 because they are nearest to their destination (queue f_1) than the packets in X (queue f_2). Note also that the packets in S will traverse the edge f_1 and be absorbed.
- **Round 2:** It lasts $|T_2| = |X|/C$ time steps.

  During Round 2, all the edges have capacity C and the adversary injects a set Y with $\rho C|T_2|$ packets in the queue f_2 needing to traverse the edges f_2, e_1. These injections satisfy the load condition because all the edges in the assigned path have capacity C during Round 2.

  Note that the packets in X delay the packets of the set Y in the queue f_2 because they are nearest to their destination (queue f_2) than the packets in Y (queue e_1). Note also that the packets in X will traverse the edge f_2 and be absorbed.
- **Round 3:** It lasts $|T_3| = |Y|/C$ time steps.

  During Round 3, all the edges have capacity C. $\mathcal{A}_1$ injects a set Z with $\rho C|T_3|$ packets in the queue f_2 needing to traverse the edges f_2, e_2. Also, $\mathcal{A}_1$ injects a set Z_1 with $\rho C|T_3|$ packets in the queue e_1 needing to traverse the edges e_1, f_1. These injections satisfy the load condition because all the edges have capacity C, and the injection paths of the different packet sets are edge-disjoint.

  Note that the packets of Y delay the packets of the set Z in the queue f_2 because they are in the system longer, and observe that the packets of Y and Z have to traverse the same distance to their destination. Note also that the packets of Y delay the packets of Z_1 in the queue e_1 because they have nearest to go (queue e_1) than the packets of Z_1 (queue f_1). Finally, note that the packets of the set Y will traverse the edge e_1 and be absorbed.
- **Round 4:** It lasts $|T_4| = |Z|/C$ time steps.

  During Round 4 the edge e_1 has capacity 1, while all the other edges have capacity C. $\mathcal{A}_1$ injects a set Z_2 with $\rho C|T_4|$ packets in the queue e_2 needing to traverse the edges e_2, f_1. These injections satisfy the load condition because all the edges in the assigned path have capacity C during Round 4.

  Note that the packets of Z delay the packets of Z_2 in the queue e_2 because they have nearest to go (to queue e_2) than the packets of Z_2 (to queue f_1). Note also that the packets of Z will traverse the edge e_2 and be absorbed. Finally, note that the packets of Z_1 are delayed in the queue e_1 due to the unit capacity of the edge e_1 during Round 4. So, the remaining packets of Z_1 in the queue e_1 at the end of Round 4 form a set $Z_1^{'}$ with $|Z_1| - |T_4|$ packets, while the remaining packets of Z_1 will traverse their remaining path and be absorbed. So, the number of packets in the queues e_1, e_2 needing to traverse the edge f_1 at the end of Round 4 is $s_{j+1} = |Z_2| + |Z_1^{'}| = \rho^4 s_j + \rho^3 s_j - \frac{\rho^3}{C} s_j$.

  For instability, it must be that $s_{j+1} > s_j$, that is $\rho^4 s_j + \rho^3 s_j - \frac{\rho^3}{C} s_j > s_j$. Therefore, $\rho^4 C + \rho^3(C-1) > C$, or $\rho^4 + \rho^3(1 - \frac{1}{C}) > 1$. For $C = 1000$ and $\rho = 0.82$, the inequality holds and the claim follows.

  When $C \to \infty$, $\frac{1}{C^k} \to 0$ for all $k \geq 1$. Then, the inequality becomes $\rho^4 + \rho^3 - 1 > 0$ which holds for $\rho > 0.8191$ and the claim follows. □

Now, we consider the networks $\mathcal{S}_2, \mathcal{S}_3, \mathcal{S}_4$ in Fig. 1 that use the NTG.LIS protocol. Similarly to Theorem 5, we can prove Theorem 6 [10]. We summarize here some proof details. Each phase now consists of four time rounds.

- For the system $\langle \mathcal{S}_2, \mathcal{A}_2, \text{NTG.LIS} \rangle$ the *inductive argument* states that if at the beginning of a phase j, there are s_j packets in the queues e_2, e_4 needing to traverse the edge f, then at the beginning of phase $j+1$ there will be more than s_j packets in the same queues needing to traverse edge f.
- For the system $\langle \mathcal{S}_3, \mathcal{A}_3, \text{NTG.LIS} \rangle$ the *inductive argument* states that if at the beginning of a phase j, there are s_j packets in the queues f_1, f_3 needing to traverse the edges f_1, e_2 and f_3, e_1, e_2, then at the beginning of phase $j+1$ there will be more than s_j packets in the same queues needing to traverse the corresponding edges.

- For the system $\langle \mathcal{S}_4, \mathcal{A}_4, \mathsf{NTG.LIS}\rangle$ the *inductive argument* states that if at the beginning of a phase j, there are s_j packets in the queues f_1, f_3 needing to traverse the edges f_1, e_2 and f_3, e_1, g_2, e_2 respectively, then at the beginning of phase $j+1$ there will be more than s_j packets in the same queues needing to traverse the corresponding edges.

We obtain:

Theorem 6. *For the network $\mathcal{S}_i$ there is an adversary $\mathcal{A}_i$ of rate ρ such that the system $\langle \mathcal{S}_i, \mathcal{A}_i, \mathsf{NTG.LIS}\rangle$ is unstable where $i = \{2, 3, 4\}$ for $\rho = 0.82$ and $C \geq 1000$. When $C \to \infty$, the system $\langle \mathcal{S}_i, \mathcal{A}_i, \mathsf{NTG.LIS}\rangle$ is unstable for $\rho > 0.8191$.*

Now, we consider the network $\mathcal{U}_1$ in Fig. 1 that uses the NTG.LIS protocol.

Theorem 7. *For the network $\mathcal{U}_1$, there is an adversary $\mathcal{A}$ of rate ρ such that the system $\langle \mathcal{U}_1, \mathcal{A}, \mathsf{NTG.LIS}\rangle$ is unstable for $\rho = 0.8$ and $C \geq 1000$. When $C \to \infty$, the system $\langle \mathcal{U}_1, \mathcal{A}, \mathsf{NTG.LIS}\rangle$ is unstable for $\rho > \sqrt[3]{0.5}$.*

Proof. We break the construction of the adversary $\mathcal{A}$ into phases.

Inductive hypothesis: At the beginning of phase j, there are in total s_j packets in the queues e_1, e_2 needing to traverse the edge f.

Induction step: We will prove that at the beginning of phase $j+1$, there will be $s_{j+1} > s_j$ packets in the queues e_1, e_2 needing to traverse the edge f.

We now construct an adversary $\mathcal{A}$ such that the induction step will hold. During phase j, the adversary uses four rounds of injections as follows:

- **Round 1:** It lasts $|T_1| = s_j/C$ time steps.

  During Round 1 all the edges have capacity C and $\mathcal{A}$ injects in the queue f a set X with $\rho C|T_1|$ packets needing to traverse the edges f, e_1. These injections satisfy the load condition because all the edges in the assigned path have capacity C during Round 1.

  Note that the packets of S delay the packets of X in the queue f because they are nearest to their destination (to queue f) than the packets of (to queue e_1). Note also that the packets of S will traverse the edge f and be absorbed.
- **Round 2:** It lasts $|T_2| = |X|/C$ time steps.

  During Round 2, all the network edges have capacity C. $\mathcal{A}$ injects a set Y with $\rho C|T_2|$ packets in the queue f needing to traverse the edges f, e_2 and a set Z with $\rho C|T_2|$ packets in the queue e_1 needing to traverse the edge e_1. These injections satisfy the load condition because all the edges have capacity C in Round 2 and the injection paths of different packet sets are edge-disjoint.

  Note that the packets of X delay the packets of Y in the queue f because they are in the system longer, and recall that the packets of X and Y have to traverse the same distance to their destination. Note also that for the same reason the packets of X delay the packets of Z in the queue e_1. Finally, note that the packets of X will traverse the edge e_1 and be absorbed.
- **Round 3:** It lasts $|T_3| = |Y|/C$ time steps.

  During Round 3 the edge e_2 has capacity 1, while all other edges have capacity C. $\mathcal{A}$ injects a set Z_1 with $\rho C|T_3|$ packets in the queue e_1 needing to traverse the edges e_1, f. These injections satisfy the load condition because all edges in the assigned path have capacity C during Round 3.

  Note that the packets of Z delay the packets of Z_1 in the queue e_1, because they are nearest to go (to queue e_1) than the packets of Z_1 (to queue f). Note also that the packets of Y are delayed in e_2 due to the unit capacity of the edge e_2 during Round 3. So, the remaining packets of Y in the queue e_2 at the end of Round 3 form a set Y' with $|Y| - |T_3|$ packets, while the remaining packets from Y will traverse their remaining path and be absorbed.
- **Round 4:** It lasts $|T_4| = |Y'|/C$ time steps.

  During Round 4, the edge e_1 has capacity 1, while all the other edges have capacity C. $\mathcal{A}$ injects a set Z_2 with $\rho C|T_4|$ packets in the queue e_2 needing to traverse the edges e_2, f. These injections satisfy the load condition because all the edges in the assigned path have capacity C during Round 4.

  Note that the packets of Y' delay the packets of Z_2 in the queue e_2 because they are nearer to go (to queue e_2) than packets of Z_2 (to queue f). Note also that the packets of Y' will traverse the edge e_2 and be absorbed. Finally note that, the packets of Z_1 are delayed in the queue e_1 due to the unit capacity of the edge e_1 during Round 4.

Therefore, the packets of the set Z_1 in the queue e_1 at the end of Round 4 form a set Z_1' with $|Z_1| - |T_4|$ packets, while the remaining packets of Z_1 will traverse their path and be absorbed. So, the number of packets in the queues e_1, e_2 needing to traverse the edge f at the end of Round 4 is $s_{j+1} = |Z_2| + |Z_1'| = \rho^3 s_j - \frac{\rho^3}{C} s_j + \rho^3 s_j - \frac{\rho^2}{C} s_j + \frac{\rho^2}{C^2} s_j$.

For instability, it must be that $s_{j+1} > s_j$, that is $2\rho^3 s_j - \frac{\rho^3}{C} s_j - \frac{\rho^2}{C} s_j + \frac{\rho^2}{C^2} s_j > s_j$ or $2\rho^3 C^2 - \rho^3 C - \rho^2 C + \rho^2 > C^2$. Dividing by C^2 both sides yields that $\rho^3(2 - \frac{1}{C}) - \rho^2(\frac{1}{C} - \frac{1}{C^2}) > 1$, which holds for $\rho = 0.8$ and $C \geq 1000$. So, the claim follows for $\rho = 0.8$ and $C \geq 1000$.

When $C \to \infty$, $\frac{1}{C^k} \to 0$ for all $k \geq 1$. So, the claim follows for $C \to \infty$ and $\rho > \sqrt[3]{0.5}$. □

Now, we consider the simple-edge, but not the simple-node, network $\mathcal{U}_2$ in Fig. 1 that uses the NTG.LIS protocol. Similarly to Theorem 7, we can prove Theorem 8 [10]. Each phase now consists of three rounds.

- For the system $\langle \mathcal{U}_2, \mathcal{A}', \text{NTG.LIS} \rangle$ the *inductive argument* states that if at the beginning of a phase j, there are s_j packets in the queues e_1, f_1 needing to traverse the edge f_2, then at the beginning of phase $j+1$, there will be more than s_j packets in the same queues needing to traverse the corresponding edge.

Theorem 8. *For the network $\mathcal{U}_2$, there is an adversary $\mathcal{A}'$ of rate ρ such that the system $\langle \mathcal{U}_2, \mathcal{A}', \text{NTG.LIS} \rangle$ is unstable for $\rho = 0.76$ and $C \geq 1000$. When $C \to \infty$, the system $\langle \mathcal{U}_2, \mathcal{A}', \text{NTG.LIS} \rangle$ is unstable for $\rho > 0.754$.*

6. Epilogue

Our results suggest that for every unstable network, its instability bound in the AQSQT model may be lower than for the classical AQT. Proving (or disproving) this remains an open problem. Another direction for further research is to determine upper bounds on the injection rate that guarantees stability for forbidden subgraphs in the AQSQT model. Studying the impact of dynamically changing link capacities on other greedy protocols and networks remains another interesting problem.

References

[1] M. Andrews, B. Awerbuch, A. Fernandez, J. Kleinberg, T. Leighton, Z. Liu, Universal stability results for greedy contention–resolution protocols, Journal of the ACM 48 (1) (2001) 39–69.

[2] C. Álvarez, M. Blesa, M. Serna, A Characterization of universal stability in the Adversarial Queuing model, SIAM Journal on Computing 34 (1) (2004) 41–66.

[3] M. Andrews, A. Fernandez, A. Goel, L. Zhang, Source routing and scheduling in packet networks, Journal of the ACM 52 (4) (2005) 582–601.

[4] W. Aiello, E. Kushilevitz, R. Ostrovsky, A. Rosén, Adaptive packet routing for bursty adversarial traffic, Journal of Computer and System Sciences 60 (3) (2000) 482–509.

[5] R. Bhattacharjee, A. Goel, Z. Lotker, Instability of FIFO at arbitrarily low rates in the Adversarial Queuing model, SIAM Journal on Computing 34 (2) (2004) 318–332.

[6] A. Borodin, J. Kleinberg, P. Raghavan, M. Sudan, D. Williamson, Adversarial Queuing theory, Journal of the ACM 48 (1) (2001) 13–38.

[7] A. Borodin, R. Ostrovsky, Y. Rabani, Stability preserving transformations: Packet routing networks with edge capacities and speeds, Journal of Interconnection Networks 5 (1) (2004) 1–12.

[8] J. Diaz, D. Koukopoulos, S. Nikoletseas, M. Serna, P. Spirakis, D. Thilikos, Stability and non-stability of the FIFO protocol, in: Proceedings of the 13th Annual ACM Symposium on Parallel Algorithms and Architectures, Crete, Greece, 2001, pp. 48–52.

[9] A. Goel, Stability of networks and protocols in the Adversarial Queuing model for packet routing, Networks 37 (4) (2001) 219–224.

[10] D. Koukopoulos, Network stability under the Adversarial Queuing model, Ph.D. Thesis, Computer Engineering and Informatics Department, University of Patras, 2003.

[11] D. Koukopoulos, M. Mavronicolas, S. Nikoletseas, P. Spirakis, On the stability of compositions of universally stable, greedy, contention–resolution protocols, in: Proceedings of the 16th International Symposium on DIStributed Computing, in: Lecture Notes in Computer Science, vol. 2508, Springer-Verlag, Toulouse, France, 2002, pp. 88–102.

[12] D. Koukopoulos, M. Mavronicolas, S. Nikoletseas, P. Spirakis, The impact of network structure on the stability of greedy protocols, Journal Theory of Computing Systems 38 (4) (2005) 425–460.

[13] D. Koukopoulos, S. Nikoletseas, P. Spirakis, Stability issues in heterogeneous and FIFO networks under the Adversarial Queuing model, in: Proceedings of the 8th International Conference on High Performance Computing, in: Lecture Notes in Computer Science, vol. 2228, Springer-Verlag, Hyderabad, India, 2001, pp. 3–14.

[14] Z. Lotker, B. Patt-Shamir, A. Rosén, New stability results for Adversarial Queuing, SIAM Journal on Computing 33 (2) (2004) 286–303.

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