Alcom-FT Technical Report Series
ALCOMFT-TR-03-136

Performance and Stability Bounds for Dynamic Networks*

Dimitrios Koukopoulos† *Marios Mavronicolas*‡ *Paul Spirakis*§

Abstract

In this work, we study the impact of dynamically changing *link capacities* on the performance bounds of LIS (*Longest-In-System*) and SIS (*Shortest-In-System*) protocols on specific networks (that can be modelled as Directed Acyclic Graphs-DAGs) and stability bounds of greedy contention-resolution protocols running on arbitrary networks under the *Adversarial Queueing Theory*. Especially, we consider the model of *dynamic capacities*, where each link capacity may take on integer values from $[1, C]$ with $C > 1$, under a (w, ρ)-adversary. The *stability* of a greedy protocol guarantees that the number of packets in the network remains bounded at all times. However, it does not guarantee the existence of small bounds on the packet delay. We therefore show:

- The packet delays on DAGs for LIS is upper bounded by $O(iw\rho C)$ where i is the length of the longest path leading to a node when nodes are ordered by the topological order induced by the DAG.
- Complementarily, the performance of LIS on DAGs is lower bounded by $\Omega(iw\rho C)$ through an involved combinatorial construction. Therefore, we show tight performance bounds for LIS on DAGs. In a similar way, we show that the performance of SIS on DAGs is lower bounded by $\Omega(iw\rho C)$.
- Any arbitrary network $\mathcal{G}$ (possibly cyclic) running a *greedy* protocol is stable for a rate not exceeding a particular stability threshold, depending on C and the length $d(\mathcal{G})$ of the longest path in the network.

1 Introduction

Motivation-Framework. We are interested in the behavior of *packet-switched networks* in which 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 stochastic ones); We work within a model of *worst-case* continuous packet arrivals, originally proposed by Borodin *et al.* [3] and termed *Adversarial Queueing Theory* to reflect the assumption of an *adversarial* way 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? Besides the existence or not of upper bounds on the packet delays, a complementary question concerning stability is whether stability guarantees that there are small bounds on packet delays. The answer to these questions may depend on the *rate* of injecting packets into the network, the *capacity* of the links, which is the rate 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. We study these questions considering that packets are injected by an adversary (rather than by an oblivious randomized process) and capacities are chosen by the same

*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), and by funds from the Joint Program of Scientific and Technological Collaboration between Greece and Cyprus.

†**Contact Author.** Department of Computer Engineering and Informatics, University of Patras, Rion, 265 00 Patras, Greece, & RACTI, P. O. Box 1122, 261 10 Patras, Greece. Email: `Dimitrios.Koukopoulos@cti.gr`

‡Departemnt of Computer Science, University of Cyprus, 1678 Nicosia, Cyprus. Email: `mavronic@ucy.ac.cy`

§Department of Computer Engineering and Informatics, University of Patras, Rion, 265 00 Patras, Greece, & RACTI, P. O. Box 1122, 261 10 Patras, Greece. Email: `spirakis@cti.gr`

adversary in a dynamic way.

Most studies of packet-switched networks assume that one packet can cross a network link in a single time step. This assumption is well motivated when we assume that all network links are identical. However, a packet-switched network can contain different types of links, which is common especially in large-scale networks like Internet. Then, it is well motivated to assign a capacity to each link that takes on values from the integer set $[1, C]$ with $C > 1$. If C is a large integer, we can consider approximately as a link failure the assigning of unit capacity to a link and the assigning of capacity C as the proper service rate. Therefore, the study of protocol stability under this model of dynamically changing capacities can be considered as an approximation of the fault-tolerance of a network where links can temporarily fail (zero capacity).

In this work we consider the impact on performance bounds and stability properties if the adversary besides the packet injections in paths which it determines, it also can set the capacities of network edges in each time step. This subfield of study was initiated by Borodin *et al.* in [4]. Note that we continue to assume uniform packet sizes. Furthermore, we consider greedy contention-resolution protocols (all of which enjoy simple implementations)- 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, such as LIS (*Longest-in-System*) that gives priority to the packets that have been for the longest amount of time in the network and SIS (*Shortest-in-System*) that gives priority to the packets that have been for the shortest amount of time in the network.

Roughly speaking a protocol P is *stable* [3] on a network $\mathcal{G}$ against an adversary $\mathcal{A}$ of rate r if there is an integer 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. Moreover, a *a network $\mathcal{G}$ is universally stable* [3] if every greedy protocol is stable against every adversary of rate less than 1 on G. Directed Acyclic Graphs are *universally stable* in the more powerful model of general dynamic capacities– each is stable against every adversary of rate less than 1 for every protocol [4, Theorem 3].

Contribution. We use here the model of dynamic capacities that has been initiated in [4] where link capacities may take on integer values in the interval $[1, C]$ with $C > 1$[1] under a (w, ρ)-adversary that injects packets at rate ρ with *window size* w. In this framework, we show:

- The delay of packets on DAGs (where LIS is running on top of them) is upper bounded by $O(iwC\rho)$ where i is the level of a node in a DAG (the length of the longest path leading to node v when nodes are ordered by the topological order induced by the graph). We use double-induction on time and the node level to prove it.
- We make this performance bound tight showing a lower bound of $\Omega(iwC\rho)$ on the packet delay that is suffered by a packet targeted with a node u of level i. The maximum queue size in this case is $(w+1)C$. The proof of this result is based on an involved adversarial construction. In a similar way, we prove that SIS has a lower bound of $\Omega(iwC\rho)$ on the packet delay when it is running on top of DAGs.
- We show two results which provide some upper bounds on the allowable rate of injections that still guarantees stability. Specifically, we prove that any arbitrary network (possibly cyclic) running a *greedy contention-resolution* protocol is stable as long as the injection rate does not exceed $\frac{1}{C(d(\mathcal{G})+1)}$, where $d(\mathcal{G})$ is the length of the longest path in the network that can be followed by any packet. A slightly improved result is proved for the stability threshold in the case of a special class of greedy protocols, the *time priority* protocols. Such a protocol is stable on all networks as long as the injection rate of the adversary does not exceed $\frac{1}{Cd(\mathcal{G})}$.

Roughly speaking, the results that have been obtained in this paper show that the linear performance bounds that have been proved in the classical setting of Adversarial Queueing Theory remain linear when link capacities vary dynamically although the adversary in such environments is more powerful. The only difference is that performance bounds

[1]The classical Adversarial Queueing Theory corresponds to the case where only one capacity value is available to the adversary.

in the dynamic setting have as expense a multiplicative factor of C. The same holds for the stability thresholds that are obtained in this paper for dynamically varying link capacities.

Related Work. *Adversarial Queueing Theory* was developed by Borodin *et al.* [3] as a more realistic model that replaces traditional stochastic assumptions in Queueing Theory by more robust, worst-case ones. Adversarial queueing theory received a lot of interest in the study of stability and instability issues (see, e.g., [1, 6, 7, 10, 11]). The universal stability of various natural greedy protocols (LIS, SIS, NTS (Nearest-to-Source), FTG (Furthest-to-Go)) has been established by Andrews *et al.* [1]. Furthermore, the set of universally stable networks has been well-characterized [1, 3] (DAGs, trees, ring).

Stability Bounds for Greedy Protocols. The subfield of proving stability thresholds for greedy protocols on every network was first initiated by Diaz *et al.* [6] showing an upper bound on injection rate for the stability of FIFO in networks with a finite number of queues that is based on network parameters. This result was significantly improved for all networks by Koukopoulos *et al.* [8] demonstrating a method for estimating an upper bound for FIFO stability which is, also, based on network parameters. In an alternative interesting work, Lotker *et al.* [11] proved that any greedy protocol can be stable in any network if the injection rate of the adversary is upper bounded by $1/(d+1)$, where d is the maximum path length that can be followed by any packet. Also, they proved that for a specific class of greedy protocols, time-priority protocols the stability threshold becomes $1/d$.

Performance Bounds for Greedy Protocols. Several universally stable protocols have exponential lower bounds on queue sizes and packet delays (SIS, NTS, FTG) [1]. In an interesting work, Adler and Rosén [2] proved tight polynomial bounds for the stability of LIS on DAGs. Especially, they have shown that LIS on DAGs has linear (in the longest path in the network that can be followed by any packet) queue sizes and packet delays.

Stability Issues in Dynamic Networks. Borodin *et al.* in [4] studied for the first time the impact on stability when the edges in a network can have capacities. They proved that many well-known universally stable protocols (SIS, NTS, FTG) do maintain their universal stability when the link capacity is changing dynamically, whereas the universal stability of LIS is not preserved. Also, the universal stability of networks is preserved under this varying context. The study of stability when link capacities change dynamically has been further extended in [9] presenting involved combinatorial constructions of the adversary that lead LIS and certain compositions of universally stable protocols to instability for a threshold of $\sqrt{2}-1$.

Road Map. The rest of this paper is organized as follows. Section 2 presents our model definitions. Bounds for the LIS protocol on DAGs are presented in Section 3. A bound for the SIS protocol on DAGs is presented in Section 4. Section 5 includes our upper bounds on the stability threshold of any network. We conclude, in Section 6, with a discussion of our results and some open problems.

2 Model

The model definitions are patterned after those in [3, Section 3], adjusted to reflect the fact that link capacities may vary arbitrarily as in [4, Section 2]. Link capacities may take on integer values in the interval $[1, C]$ with $C > 1$. A *routing network* is modelled as a directed graph $\mathcal{G} = (V, E)$ with $|V| = n$ *nodes* and $|E| = m$ *edges.* Each node $v \in V$ represents a communication switch and each directed edge $e \in E$ represents a link between two switches that delivers packets only in the direction it is oriented. Every switch has a buffer (queue) at the tail of each out-going link and stores there the packets to be sent on the corresponding link.

Time proceeds in discrete steps. New packets, requiring to traverse predetermined paths, can be injected into the network at any time step. A *packet* is an atomic entity that resides at a node at the end of any step. It must travel along paths in the network from its *source* to its *destination,* both of which are nodes in the network. When it reaches its destination, we say that 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 edge e at time step t. That is, we assume that edge e is capable of simultaneously transmitting up to $\mathsf{C}_e(t)$ packets at time t.

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 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 followed by a packet.[2] We say that the adversary generates a set of packets when it generates a set of requested paths. We restrict our study to the case of *non-adaptive* routing, where the path traversed by each packet is fixed at the time of injection, so that we are able to focus on queueing rather than routing aspects of the problem. There are no computational restrictions on how the adversary chooses its requests in any given time step.

Fix any arbitrary positive integer $w \geq 1$. Throughout, for any sequence of time steps $\mathcal{T} = [t_1, t_2]$ and for any integer $w \geq 1$, denote $\mathcal{T} \pm w$ the sequence of time steps $[t_1 - w, t_2 + w]$. For any edge e of the network and any sequence $\mathcal{T}_w$ of w consecutive time steps, define $N(\mathcal{T}, e)$ to be the number of paths injected by the adversary during the time interval $\mathcal{T}$ that traverse edge e. For any constant ρ, $0 < \rho \leq 1$, a (w, ρ)-*adversary* $\mathcal{A}_{w,\rho}$ is an adversary that injects packets subject to the following *load condition*: For every edge e and for every sequence $\mathcal{T}_w$ of w consecutive time steps, $N(\mathcal{T}_w, e) \leq \rho \sum_{\tau \in \mathcal{T}_w} \mathsf{C}(\tau)$. We say that a (w, ρ)-adversary $\mathcal{A}_{w,\rho}$ injects packets at rate ρ with *window size* w.

Lemma 2.1 *Fix any (w, ρ)-adversary $\mathcal{A}_{w,\rho}$. Then, for any time step $t \geq 1$ and for any edge e,*

$$N([t], e) \quad \leq \quad \rho \max_{\max\{1, t-w+1\} \leq \delta \leq t} \sum_{\delta \leq \tau \leq \delta + w - 1} \mathsf{C}(\tau) \,.$$

Lemma 2.1 immediately implies:

Corollary 2.2 *Fix any (w, ρ)-adversary $\mathcal{A}_{w,\rho}$. Then, for any edge e,*
$N([1], e) \leq \rho \sum_{1 \leq \tau \leq w} \mathsf{C}(\tau)$.

Lemma 2.3 *Fix any (w, ρ)-adversary $\mathcal{A}_{w,\rho}$. Then, for any time interval $\mathcal{T}$ and for any edge e, $N(\mathcal{T}, e) \leq \rho \sum_{\tau \in \mathcal{T} \pm w} \mathsf{C}(\tau)$.*

[2] In this work, it is assumed, as it is common in packet routing, that all such paths are simple paths with no overlapping edges.

The assumption that $\rho \leq 1$ ensures that it is not necessary a priori that some edge of the network is congested (which would surely happen when $\rho > 1$).

A *contention-resolution* protocol specifies, for each pair of an edge e and a time step, which packet among those waiting at the tail of edge e will be moved along edge e. A *greedy* contention-resolution protocol always specifies some packet to move along edge e if there are packets waiting to use edge e. All these contention-resolution protocols require some tie-breaking rule in order to be unambiguously defined. In this work, whenever we are proving a positive result, we assume that the adversary can break the tie arbitrarily; for proving a negative result, we can assume any well-determined tie breaking rule for the adversary. For simplicity, and in a way similar to that in [1] and in works following it, we omit floors and ceilings from our analysis, and we sometimes count time steps and packets only roughly. This may only result to loosing small additive constants, while it implies a gain in clarity.

3 LIS on Directed Acyclic Graphs

Upper and lower bounds on the performance of LIS on DAGs appear in Sections 3.1 and 3.2, respectively. Throughout this section, we will consider Directed Acyclic Graphs, in which nodes are ordered by the topological order induced by the graph. This order assigns to each node v a level $i \geq 0$, denoted *level*(v) such that the longest path leading to node v has length i.

3.1 Upper Bound

In this section, we present our upper bound on the performance of LIS on directed acyclic graphs. We start by proving:

Proposition 3.1 *Consider any packet p injected at time t, and let v be any node of level $i \geq 0$ on its path other than its destination node. Then, packet p clears node v by time $t + \delta(t)$, where $\delta(t) > 0$ is the least integer such that $\sum_{t \leq \tau \leq t+\delta(t)} \mathsf{C}(\tau) \geq \rho(i+1)$ $\sum_{\max\{1, t-\rho(2w-1)C(i+1)-(i+1)+1-w+1\} \leq \tau \leq t+w-1} \mathsf{C}(\tau)$.*

Proof: By double induction on i and on t; the outer induction is on i. The basis case and the induction step of the outer induction are shown by an inner induction on t. Throughout, denote $e = \langle v, u\rangle$ the edge over which p has to leave v.

For the basis case of the outer induction, assume that $i = 0$. Note that in this case any packet that has node v on its path must be injected into node v, since node v has level 0. In particular, packet p is injected into node v. By definition of the LIS protocol, it follows that packet p can be delayed at node v only by packets injected into v at times $t' \leq t$.

We prove the claim for the case $i = 0$ by an inner induction on t.

For the basis case of the inner induction, assume that $t = 1$. Since $w \geq 1$ while, in this case, $\max\{1, t - w + 1\} = \max\{1, 2 - w\} = 1$, we need to show that packet p clears node v by time $1 + \delta(1)$, where $\delta(1) > 0$ is the least integer such that

$$\sum_{1 \leq \tau \leq 1+\delta(1)} \mathsf{C}(\tau) \quad \geq \quad \rho \sum_{1 \leq \tau \leq w} \mathsf{C}(\tau)\,.$$

Note that since $t = 1$, packet p can be delayed only by packets injected into v at time step 1 that have edge e on their path. By Corollary 2.2, there are at most $\rho \sum_{1 \leq \tau \leq w} \mathsf{C}(\tau)$ such packets (including p itself). It follows that p will clear v by time $1+\delta(1)$ where $\delta(1)$ is the least integer such that

$$\sum_{1 \leq \tau \leq 1+\delta(1)} \mathsf{C}(\tau) \quad \geq \quad \rho \sum_{1 \leq \tau \leq w} \mathsf{C}(\tau)\,.$$

For the induction step, consider any time step $t > 1$. Consider any packet p' injected into v at time $t' \leq t - (2w - 1)C\rho$. Since $t' < t$, induction hypothesis implies that p' clears node v by time $t' + \delta(t')$, where $\delta(t')$ is the least integer such that

$$\sum_{t' \leq \tau \leq t'+\delta(t')} \mathsf{C}(\tau) \quad \geq \quad \rho \sum_{\max\{1, t'-w\} \leq \tau \leq t'+w-1} \mathsf{C}(\tau)\,.$$

We continue to prove:

Lemma 3.2 $\delta(t') \leq (2w - 1)C\rho - 1$

Proof: By definition of $\delta(t^{'})$, it suffices to show that $\sum_{t' \leq \tau \leq t'+(2w-1)C\rho-1} \mathsf{C}(\tau) \geq \rho \sum_{\max\{1,t'-w+1\} \leq \tau \leq t'+w-1} \mathsf{C}(\tau)$.

Since $\mathsf{C}(t) \geq 1$ for all time steps t, it follows that $\sum_{t' \leq \tau \leq t'+(2w-1)\mathsf{C}\rho-1} \mathsf{C}(\tau) \geq (2w - 1)\mathsf{C}\rho - 1 + 1 = (2w - 1)\mathsf{C}\rho$.

Since $\mathsf{C}(t) \leq C$ for all time steps t, it follows that $\rho \sum_{\max\{1,t'-w+1\} \leq \tau \leq t'+w-1} \mathsf{C}(\tau) \leq \rho(2(w - 1) + 1)\, C = \rho(2w - 1)\, C$.

Hence, $\sum_{t' \leq \tau \leq t'+(2w-1)C\rho-1} \mathsf{C}(\tau) \geq \rho \sum_{\max\{1,t'-w+1\} \leq \tau \leq t'+w-1} \mathsf{C}(\tau)$. ■

It follows that $t' + \delta(t') \leq t' + \rho(2w - 1)C - 1$ by Lemma 3.2. But, $t' + \rho(2w - 1)C - 1 \leq t - 1$ since $t' \leq t - \rho(2w - 1)C$ and $t - 1 < t$. Thus, packet p can be delayed in node v only by packets injected in the time interval $[t - (2w - 1)C\rho + 1, t]$. By Lemma 2.3, there are at most $\rho \sum_{\max\{1,t-(2w-1)C\rho+1-w+1\} \leq \tau \leq t+w-1} \mathsf{C}(\tau)$ such packets. It follows that packet p clears node v by time $t + \delta(t)$, where $\delta(t) > 0$ is the least integer such that $\sum_{t \leq \tau \leq t+\delta(t)} \mathsf{C}(\tau) \geq \rho \sum_{\max\{1,t-(2w-1)C\rho+1-w+1\} \leq \tau \leq t+w-1} \mathsf{C}(\tau)$. This completes the proof of the inner induction (on t).

The basis case of the outer induction (on i) is now complete. We now proceed to the induction step of the outer induction. We prove the claim for the case $i > 0$ by an inner induction on $t \geq 1$. By definition of the LIS protocol, it follows that packet p can be delayed at node v only by packets injected either into v or into other nodes that are predecessors of v in the network G at times $t' \leq t$.

For the basis case of the inner induction, assume that $t = 1$. Note that since $t = 1$, packet p can be delayed at node v only by packets injected at time $t = 1$ that have edge e on their path. We distinguish between two cases:

- Assume first that packet p is injected into node v. Note that since $t = 1$, packet p can be delayed only by packets injected at time step 1 that have e on their path. By Corollary 2.2, there are at most $\rho \sum_{1 \leq \tau \leq w} \mathsf{C}(\tau)$ such packets (including p itself). It follows that p will clear v by time $1 + \delta(1)$ where $\delta(1)$ is the least integer, such that $\sum_{1 \leq \tau \leq 1+\delta(1)} \mathsf{C}(\tau) \geq \rho \sum_{1 \leq \tau \leq w} \mathsf{C}(\tau)$ as needed.

- Assume now that packet p is injected into some node different than v. Thus, p arrives at node v over an edge (w, v) for some node w such that $\mathit{level}(w) < \mathit{level}(v) = i$. Denote $k = \mathit{level}(w)$. By the induction hypothesis (on i), packet p clears node w by time $1 + \widetilde{\delta}(1)$, where $\widetilde{\delta}(1)$ is the least integer, such that $\sum_{1\leq\tau\leq 1+\widetilde{\delta}(1)} \mathsf{C}(\tau) \geq \rho(k+1)\sum_{\max\{1,1-w+1\}\leq\tau\leq 1+w-1} \mathsf{C}(\tau) = \rho(k+1)\sum_{1\leq\tau\leq w} \mathsf{C}(\tau)$.

Thus, packet p arrives at node v by time $1+\widetilde{\delta}(1)$. Note that packet p can be delayed (at node v) only by packets that are injected at time $t=1$ that have e on their path. By Corollary 2.2, there are at most $\rho\sum_{1\leq\tau\leq w} \mathsf{C}(\tau)$ such packets. It follows that p will clear v by time $1+\widetilde{\delta}(1)+\widetilde{\delta}(1+\widetilde{\delta}(1))$, where $\widetilde{\delta}(1+\widetilde{\delta}(1))$ is the least integer, such that $\sum_{1+\widetilde{\delta}(1)\leq\tau\leq 1+\widetilde{\delta}(1)+\widetilde{\delta}(1+\widetilde{\delta}(1))} \mathsf{C}(\tau) \geq \rho\sum_{1\leq\tau\leq w} \mathsf{C}(\tau)$.

Define now $\delta(1)$ to be the least integer, such that $\sum_{1\leq\tau\leq 1+\delta(1)} \mathsf{C}(\tau) \geq \rho(i+1)\sum_{1\leq\tau\leq w} \mathsf{C}(\tau)$.

It remains to prove that packet p clears node v by time $\delta(1)$. But, $\rho(i+1)\sum_{1\leq\tau\leq w} \mathsf{C}(\tau) \geq \rho(k+2)\sum_{1\leq\tau\leq w} \mathsf{C}(\tau)$. However, $\sum_{1\leq\tau\leq 1+\widetilde{\delta}(1)+\widetilde{\delta}(1+\widetilde{\delta}(1))} \mathsf{C}(\tau)$
$= \sum_{1\leq\tau\leq 1+\widetilde{\delta}(1)} \mathsf{C}(\tau)$
$+ \sum_{1+\widetilde{\delta}(1)\leq\tau\leq 1+\widetilde{\delta}(1)+\widetilde{\delta}(1+\widetilde{\delta}(1))} \mathsf{C}(\tau)$
$\geq \rho\sum_{1\leq\tau\leq w} \mathsf{C}(\tau) + \rho(k+1)\sum_{1\leq\tau\leq w} \mathsf{C}(\tau)$
$= \rho(k+2)\sum_{1\leq\tau\leq w} \mathsf{C}(\tau)$

and packet p clears node v by time $1+\widetilde{\delta}(1)+\widetilde{\delta}(1+\widetilde{\delta}(1))$. It follows that packet p clears node v by time $\delta(1)$.

This completes the proof of the basis case of the inner induction (on t).

We proceed now to the induction step of the inner induction, where we assume that $t>1$. We again distinguish between two cases regarding the node where packet p is injected.

- Assume first that packet p is injected into node v. Then, clearly, packet p resides at node v by the end of time step t.

- Else, assume that p arrives at node v over an edge $\langle w,v\rangle$ for some node w such that $\mathit{level}(w) < \mathit{level}(v) = i$. By the induction hypothesis (on i), packet p clears node w by time $t+\widetilde{\delta}(t)$ where $\widetilde{\delta}(t)$ is the least integer such that $\sum_{t\leq\tau\leq t+\widetilde{\delta}(t)} \mathsf{C}(\tau)$
$\geq \rho(k+1)\sum_{\max\{1,t-w+1\}\leq\tau\leq t+w-1} \mathsf{C}(\tau)$.

Thus, in any case, packet p clears node w by time $t+\widetilde{\delta}(t)$, where $\widetilde{\delta}(t)$ is the least integer such that $\sum_{t\leq\tau\leq t+\widetilde{\delta}(t)} \mathsf{C}(\tau)$
$\geq \rho(k+1)\sum_{\max\{1,t-w+1\}\leq\tau\leq t+w-1} \mathsf{C}(\tau)$.

Consider now any packet p' injected at time step $t' \leq t-\rho(2w-1)C(i+1)-(i+1)$. Since $t' < t$, the induction hypothesis (on t) implies that packet p' clears node v by time step $t'+\delta(t')$, where $\delta(t')$ is the least integer such that $\sum_{t'\leq\tau\leq t'+\widetilde{\delta}(t')} \mathsf{C}(\tau)$
$\geq \rho(k+1)\sum_{\max\{1,t'-w+1\}\leq\tau\leq t'+w-1} \mathsf{C}(\tau)$.

We continue to prove:

Lemma 3.3 $\delta(t') \leq ((2w-1)C\rho-1)(i+1)+i$

Proof: By definition of $\delta(t')$, it suffices to show that $\sum_{t'\leq\tau\leq t'+((2w-1)C\rho-1)(i+1)+i} \mathsf{C}(\tau)$
$\geq \rho\ \sum_{\max\{1,t'-w+1\}\leq\tau\leq t'+w-1} \mathsf{C}(\tau)$.

Since $\mathsf{C}(t)\geq 1$ for all time steps t, it follows that $\sum_{t'\leq\tau\leq t'+((2w-1)C\rho-1)(i+1)+i} \mathsf{C}(\tau)$
$\geq ((2w-1)C\rho-1)(i+1)+i+1 = (2w-1)C\rho(i+1)$.

Since $\mathsf{C}(t)\leq C$ for all time steps t, it follows that $\rho(i+1)\sum_{\max\{1,t'-w+1\}\leq\tau\leq t'+w-1} \mathsf{C}(\tau)$
$\leq \rho(2(w-1)+1)C(i+1) = \rho(2w-1)\,C(i+1)$. Hence, $\sum_{t'\leq\tau\leq t'+((2w-1)C\rho-1)(i+1)+i} \mathsf{C}(\tau)$
$\geq \rho\ \sum_{\max\{1,t'-w+1\}\leq\tau\leq t'+w-1} \mathsf{C}(\tau)$ as needed. ∎

It follows that $t'+\delta(t') \leq t'+((2w-1)C\rho-1)(i+1)+i-1$ by Lemma 3.3. But, $t'+((2w-1)C\rho-1)(i+1)+i-1 \leq t-1$ since $t' \leq t-((2w-1)C\rho-1)(i+1)-(i+1)$ and $t-1<t$. Thus, packet p can be delayed in node v only by packets injected in the time interval $[t-\rho(2w-1)C(i+1)-(i+1)+1, t]$.

By Lemma 2.3, there are at most $\rho\sum_{\max\{1,t-\rho(2w-1)C(i+1)-(i+1)+1-w+1\}\leq\tau\leq t+w-1} \mathsf{C}(\tau)$ such packets. It follows that packet p clears node v by time $t+\delta(t)$, where $\delta(t)>0$ is the least integer such that $\sum_{t\leq\tau\leq t+\delta(t)} \mathsf{C}(\tau) \geq \rho\sum_{\max\{1,t-\rho(2w-1)C(i+1)-(i+1)+1-w+1\}\leq\tau\leq t+w-1} \mathsf{C}(\tau)$ as needed. This completes the proof of the inner induction (on t). The proof is now complete. ∎

From Proposition 3.1 it immediately follows:

Theorem 3.4 *For any DAG $\mathcal{G}$, consider the system $\langle\mathcal{G},\mathcal{A}_{w,\rho},\mathsf{LIS}\rangle$. Then, the delay of any packet is at most $(\rho(2w-1)C-1)l(\mathcal{G})+l(\mathcal{G})$, where $l(\mathcal{G})$ is the length of the longest path in the network.*

Proof: Consider any packet p injected at time t with destination v of $level(v) = i$. Such a packet has to arrive to v over an edge $\langle w, v\rangle$ for some node w such that $level(w) < level(v) = i$. By Proposition 3.1 packet p clears node w (and arrives at v) by time $t+\widetilde{\delta}(t)$ where $\widetilde{\delta}(t)$ is the least integer such that $\sum_{t\leq\tau\leq t+\widetilde{\delta}(t)} \mathsf{C}(\tau) \geq \rho(k+1)$ $\sum_{\max\{1,t-\rho(2w-1)C(k+1)-(k+1)+1-w+1\}\leq\tau\leq t+w-1} \mathsf{C}(\tau)$.

Consider now any packet $p^{'}$ injected at time step $t^{'} \leq t-(2w-1)C\rho i - i$. Since $t^{'} < t$, packet $p^{'}$ clears node w by time step $t^{'}+\delta(t^{'})$, where $\delta(t^{'})$ is the least integer such that $\sum_{t^{'}\leq\tau\leq t^{'}+\widetilde{\delta}(t^{'})} \mathsf{C}(\tau)$ $\geq \rho(k+1)\sum_{\max\{1,t^{'}-w+1\},\leq\tau\leq t^{'}+w-1} \mathsf{C}(\tau)$.

We continue to prove:

Lemma 3.5 $\delta(t') \leq (\rho(2w-1)C-1)i+i$

Proof: By definition of $\delta(t')$, it suffices to show that $\sum_{t'\leq\tau\leq t'+((2w-1)C\rho-1)i+i-1} \mathsf{C}(\tau) \geq \rho\sum_{\max\{1,t'-w+1\}\leq\tau\leq t'+w-1} \mathsf{C}(\tau)$.

Since $\mathsf{C}(t) \geq 1$ for all time steps t, it follows that $\sum_{t'\leq\tau\leq t'+((2w-1)C\rho-1)i+i-1} \mathsf{C}(\tau) \geq$ $(\rho(2w-1)C-1)i+i-1+1 = \rho(2w-1)Ci$.

Since $\mathsf{C}(t) \leq C$ for all time steps t, it follows that $\rho i\sum_{\max\{1,t'-w+1\}\leq\tau\leq t'+w-1} \mathsf{C}(\tau) \leq$ $\rho(2(w-1)+1)Ci = \rho(2w-1)Ci$.

Hence, $\sum_{t'\leq\tau\leq t'+((2w-1)C\rho-1)i+i-1} \mathsf{C}(\tau) \geq$ $\rho\ \sum_{\max\{1,t'-w+1\}\leq\tau\leq t'+w-1} \mathsf{C}(\tau)$ as needed. ■

It follows that $t'+\delta(t') \leq t'+((2w-1)C\rho-1)i+i-2$ by Lemma 3.3. But, $t'+((2w-1)C\rho-1)i+i-2 \leq t-1$ since $t' \leq t-((2w-1)C\rho-1)i-i$, and $t-1 < t$. Thus, packet p can be delayed in node w only by packets injected in the time interval $[t-(2w-1)C\rho i-i+1, t]$.

By Lemma 2.3, there are at most $\rho\sum_{\max\{1,t-\rho(2w-1)Ci-i+1-w+1\}\leq\tau\leq t+w-1} \mathsf{C}(\tau)$ such packets. It follows that packet p clears node w by time $t+\delta(t)$, where $\delta(t) > 0$ is the least integer such that $\sum_{t\leq\tau\leq t+\delta(t)} \mathsf{C}(\tau) \geq$ $\rho\sum_{\max\{1,t-\rho(2w-1)Ci-i+1-w+1\}\leq\tau\leq t+w-1} \mathsf{C}(\tau)$.

Because $\delta(t) \leq (\rho(2w-1)C-1)i+i$, it holds that packet p clears node w by time $t+\delta(t) \leq t+(\rho(2w-1)C-1)i+i$. Similarly, we can prove that packet p clears node w arriving at a node v of $level(v) = l(\mathcal{G})$ by time $t+\delta(t) \leq t+(\rho(2w-1)C-1)l(\mathcal{G})+l(\mathcal{G})$. ■

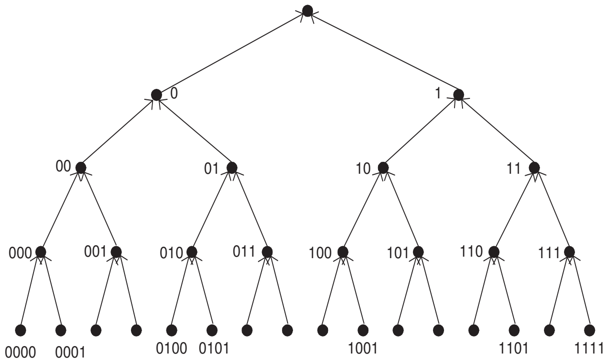

Figure 1: The network $\mathcal{G}_0(4)$

3.2 Lower Bound

In this section, we present our lower bound on the performance of LIS on DAGs. We show:

Theorem 3.6 *There is a DAG $\mathcal{G}$ and an adversary $\mathcal{A}_{w,\rho}$ such that in the system $\langle\mathcal{G}, \mathcal{A}, \mathsf{LIS}\rangle$ any packet with destination a node v of $\mathcal{G}$ at level i suffers a delay of $\Omega(iwC\rho)$ time steps, while the largest queue required is $(w+1)C$.*

The Construction of $\mathcal{G}$. For any integer $k \geq 2$, consider the complete binary tree $\mathcal{G}_0(k)$ of height k, with all edges directed towards the root v_0. Furthermore, assign to every node v of $\mathcal{G}_0(k)$ a binary string $s(v)$, called the label of node v, as follows: (i) $s(v_0) = \epsilon$ (the empty string), (ii) for each level l of $\mathcal{G}_0(k)$ $(1 \leq l \leq k)$ order the nodes at level l from left to right and number them with the integers $1, \ldots, 2^l$. For each node v of order m $(1 \leq m \leq 2^l)$, $s(v)$ will be the binary representation of $m-1$ that uses l bits. From the definition of the label function s, it immediately follows:

Lemma 3.7 *For each internal node v of $\mathcal{G}_0(k)$, $s(v)$ is the longest common prefix of all $s(u)$, where u runs over the two children of v.*

Proof: Assume that v has order m, so that $s(v) = b(m-1)$, and consider any child u of v. Clearly, the order of u is either $2m-1$ or $2m$. Hence, $s(u)$ is the binary representation b of either $2m-2$ or $2m-1$. But, $b(2m-2) = b(2(m-1)) = b(m-1)0$ and $b(2m-1) = b(2m-2+1) = b(2m-2)+b(1) = b(m-1)0+b(1) = b(m-1)0$. Thus, $s(v)$ is the

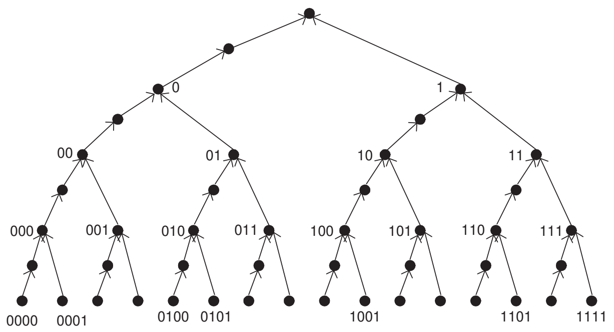

Figure 2: The network $\mathcal{G}(4)$

longest common prefix ($b(m-1)$) of the two $s(u)$, where u is a child of v, as needed. ■

Modify now the network $\mathcal{G}_0(k)$ to obtain the network $\mathcal{G}(k)$. For each internal node v: (i) remove the edge (v_l, v) pointing to v from its left child v_l, and (ii) add a node $v_l^{'}$ and the edges $(v_l, v_{l'})$ and $(v_{l'}, v)$. Note that the resulting network $\mathcal{G}(k)$ is a subgraph of the complete binary tree of height $2k$. Figures 1 and 2 illustrate the networks $\mathcal{G}_0(4)$ and $\mathcal{G}(4)$ providing the labels of many nodes.

Preliminaries. For each node v in the network $\mathcal{G}_0(k)$, denote $Z(v)$ the number of 0s in the string $s(v)$. Note that the distance of node v from the root is different is $|s(v)|$ in $\mathcal{G}_0(k)$, while it is $|s(v)| + Z(v)$ in $\mathcal{G}(k)$. The additional term $Z(v)$ accounts for the number of nodes added in the path from v to the root, which is, by construction, the number of 0s in $Z(v)$ (since a node is added for each node in the path that is a left child).

For each leaf v in $\mathcal{G}_0(k)$, define $D(v)$ to be the predecessor u of v in $\mathcal{G}_0(k)$ that is closest to v and v is a descendant of, or equal to, the left child of a child of $D(v)$, or the root, if no such predecessor u exists. The definition of $D(v)$ immediately implies: For each leaf v in $\mathcal{G}_0(k)$, $D(v)$ is the node on the path from v to the root, such that $s(D(v))$ is obtained by removing the least significant bits from $s(v)$ till either one bit after the first 0 has been removed, or the empty string has been obtained.

The Adversary. We assume that, initially ($t = 0$), there is a number of packets that are queued at each leaf requiring to traverse only the out-going edge of the leaf where they are queued. The number of leaves in $\mathcal{G}(k)$ is 2^k. In the queue at the tail of the out-going edge of leaf v, there is initially queued a number of $wC + k - Z(v)$ packets.

The out-going edge of each leaf v has unit capacity at the time interval $[0, k - Z(v))$. Therefore, at each one of these time steps one packet from the initial ones leaves the system at each out-going edge. Till the time $k - Z(v) + w$ all the packets that are queued initially in a leaf v have been absorbed. At time step $k - Z(v)$, the adversary injects $\frac{w\rho C}{2}$ packets to each leaf node v targeted with node $D(v)$. The path that is assigned to these paths has capacity C at the interval $[k - Z(v), k - Z(v) + w - 1]$, while after this time the path changes capacity to unit (only the edges that do not overlap with paths that are used for packet injections at other time steps).

In order to guarantee that this adversary is a valid one, we should show that the packets that want to traverse any common edge with capacity C at the time of their injection cannot be more than $w\rho C$ packets. Because the adversary injects in any leaf $\frac{w\rho C}{2}$ packets, it suffices to show that any edge is used at most by the packet flows injected in two leaves. Note that if a packet that is injected to a leaf v wants to reach the nodes u_1 and u_2 of the original tree there are two cases depending on whether these nodes are neighbors or not. If these nodes are neighbors, the packet should traverse the edge (u_1, u_2), otherwise it should traverse the edges $(u_1, u_1^{'})$ and $(u_1^{'}, u_2)$. For this purpose, the $k - |s(u_1)| - 1$ least significant bits of $s(v)$ must all be 1's, and the $|s(u_1)|$ most significant bits of $|s(v)|$ must all match the corresponding bits of $s(u_1)$. This can happen for at most two leaves. Thus, at most two leaves send packets that traverse any edge, and therefore any edge is used by at most $w\rho C$ packets.

Evolution of the System Configuration. We show here the evolution of the system configuration at any internal node of the network $\mathcal{G}_k$.

Claim 3.8 *Consider any node v of $\mathcal{G}_0(k)$ such that $|s(v)| \leq k-2$. Then, all packets that traverse v arrive at v between time $(k - |s(v)| - 1)\frac{wC\rho}{2} + 2k - Z(v) - |s(v)| + w - 1$ and time $(k - |s(v)|)\frac{wC\rho}{2} + 2k - Z(v) - |s(v)| + w - 2$.*

Proof: We prove this by induction on $k - |s(v)|$, i.e., on the height of the node v.

Basis case: Let v be any node such that $|s(v)| = k-2$, let v_ρ be the right child of v, and let v_l be the left child of v in the original tree. Note that $s(v_l) = s(v)0$. Let $v_l^{''}$ be the right child of v_l, and let $v_l^{'}$ be the left child of v_l in the original tree.

At time $k - Z(v_l^{'})$, the adversary injects in $v_l^{'}$ $\frac{wC\rho}{2}$ packets requiring a path with capacity C at time $k - Z(v_l^{'})$. After time $k - Z(v_l^{'}) + w - 1$ the path they follow to reach v_l changes capacity to 1. Therefore, these packets arrive, one per time step, to v_l starting at time $k - Z(v_l^{'}) + 1 + w$. Since $s(v_l^{'}) = s(v_l)0$, then $Z(v_l^{'}) = Z(v_l) + 1$. Thus, $k - Z(v_l^{'} + 2 = k - Z(v_l) + 1$. The packets that v_l receives from $v_l^{''}$ are injected by the adversary to $v_l^{''}$ at time $k - Z(v_l^{''}) = k - Z(v_l)$. So, the packets from $v_l^{''}$ and $v_l^{'}$ arrive to v_l at exactly the same time steps.

But, the packets coming to v_l from $v_l^{'}$ have been in the system longer than those coming from $v_l^{''}$. So, they will have priority over the packets from $v_l^{''}$. But, packets from $v_l^{'}$ have v as final destination. Thus, the first of the packets that v receives from v_l, which will be forwarded on (packets coming from $v_l^{''}$), arrives at v at time $[k - Z(v_l) + 1] + \frac{w\rho C}{2} + 1 + w$, and the remainder arrive one per time step for the next $\frac{wC\rho}{2} - 1$ time steps as the path they follow till there has unit capacity. Since $Z(v_l) = Z(v) + 1$ and $|s(v)| = k - 2$, we take $[k - Z(v_l) + 1] + \frac{w\rho C}{2} + 1 + w = (k - |s(v)| - 1)\frac{w\rho C}{2} + 2k - Z(v) - |s(v)| + w - 1$. This means that exactly one of those packets arrives at every time step between time $(k - |s(v)| - 1)\frac{wC\rho}{2} + 2k - Z(v) - |s(v)| + w - 1$, and time $(k - |s(v)|)\frac{wC\rho}{2} + 2k - Z(v) - |s(v)| + w - 2$. An analogous fact holds for the packets arriving to v from v_ρ that are forward onward by v.

Inductive hypothesis: For any node v of $\mathcal{G}_0(k)$, the packets that reach v on their way to another node, arrive in v between time $(j - 1)\frac{wC\rho}{2} + 2k - Z(v) - |s(v)| + w - 1$ and time $j\frac{wC\rho}{2} + 2k - Z(v) - |s(v)| + w - 2$ for $k - |s(v)| = j$.

Induction step: We assume the claim for all $v^{'}$ such that $k - |s(v^{'})| \leq j$. Choose any v such that $k - |s(v)| = j + 1$, let v_ρ be the right child of v, and let v_l be the left child of v in $\mathcal{G}_0(k)$. By induction, the packets that v_l forwards to v arrive at v_l between time $(j - 1)\frac{wC\rho}{2} + 2k - Z(v_l) - |s(v_l)| + w - 1$ and time $j\frac{wC\rho}{2} + 2k - Z(v_l) - |s(v_l)| + w - 2$. Of these packets, the ones that v_l receives from its left child have been originated at a node $v_l^{'}$ such that $s(v_l^{'}) = s(v_l)01^{j-1}$, and the ones that v_l receives from its right child have been originated at a node $v_l^{''}$ such that $s(v_l^{'}) = s(v_l)1^j$. Since $Z(v_l^{'}) = Z(v_l^{''}) + 1$, all the packets that v_l receives from its left child have been in the system longer than the packets that v_l receives from its right child.

Thus, all the packets that v_l receives from its left child and passes on to v (that also have v as their destination) have priority over any packet that v_l receives from its right child and passes on to v. Since v_l receives one packet to forward per time step from each of its children, all the packets, which have v as their destination, are forwarded before any packet that has a further destination. There are $\frac{wC\rho}{2}$ packets that v_l forwards to v having v as a final destination. Thus, the packets that v receives from v_l that need to be forwarded arrive between time $j\frac{wC\rho}{2} + 2k - Z(v_l) - |s(v_l)| + 1 + w$, and $(j+1)\frac{wC\rho}{2} + 2k - Z(v_l) - |s(v_l)| + w$. Since $|s(v_l)| = |s(v)| + 1$ and $Z(v_l) = Z(v) + 1$, these times steps are between time $(k - |s(v)| - 1)\frac{wC\rho}{2} + 2k - Z(v) - |s(v)| + w - 1$ and $(k - |s(v)|)\frac{wC\rho}{2} + 2k - Z(v) - |s(v)| + w - 2$. Note that exactly one of those packets arrives at v at each time step. A similar argument shows that at each time step where v receives a packet from v_l to forward it onward, it also receives a packet from v_ρ for forwarding. ∎

Proof of the Theorem 3.6. The theorem follows from the fact that the longest path that leads to any node v is $i = 2(k - |s(v)|)$. By Claim 3.8, packets destined for v reach v_ρ, the right child of v, no earlier than time $(k - |s(v_\rho)| - 1)\frac{wC\rho}{2} + 2k - Z(v_\rho) - |s(v_\rho)| + w - 1$. The last of these packets will reach v at time $(k - |s(v_\rho)|)\frac{wC\rho}{2} + 2k - Z(v_\rho) - |s(v_\rho)| + w$. These packets are inserted at a leaf $v^{'}$ such that $s(v^{'}) = s(v_\rho)01^{k-|s(v_\rho)|-1}$ at time step $k - Z(v^{'}) = k - Z(v_\rho) - 1$ and the path that is assigned to them has capacity C. Thus, the total time spent in the system by these packets is $(k - |s(v_\rho)|)\frac{wC\rho}{2} + k - |s(v_\rho)| + w + 1$. Substituting $|s(v)| + 1$ for $|s(v_\rho)|$, this is $(k - |s(v)| - 1)(\frac{wC\rho}{2} + 1) + w + 1 = \Omega(iwC\rho)$. The largest queue required for the adversary construction is $wC + k$. For $k = C$, we take $wC + +C = (w+1)C$.

4 SIS on DAGs

In this section, we present lower bounds on the performance of SIS on DAGs. Recall that we consider

DAGs, in which nodes are ordered by the topological order induced by the graph. This order assigns to each node v a level $i \geq 0$, denoted *level*(v) such that the longest path leading to node v has length i.We show:

Theorem 4.1 *There is a DAG $\mathcal{G}$ and an adversary $\mathcal{A}_{w,\rho}$ such that in the system $\langle \mathcal{G}, \mathcal{A}, \mathsf{LIS} \rangle$ any packet with destination a node v of $\mathcal{G}$ at level i suffers a delay of $\Omega(iwC\rho)$ time steps, while the largest queue required is $(w+1)C$.*

Sketch of proof: This proof is similar to the proof of Theorem 3.6. Again for any integer $k \geq 2$ we consider the complete binary tree of height k, $\mathcal{G}_0(k)$ where a binary string $s(v)$, called the label of node v, is assigned to every node v as in Theorem 3.6 (Figure 1 illustrates the network $\mathcal{G}_0(4)$, providing the labels of many nodes.) We construct similarly to Theorem 3.6 the network $\mathcal{G}(k)$ that is a subgraph of the complete binary tree of height $2k$ (Figure 2 illustrates the network $\mathcal{G}(4)$.)

The only difference here to the proof of Theorem 3.6 is the construction of the adversary. More specifically, now we assume that, initially ($t = 0$) at each leaf v with order m from left to right, where m takes on even values from the integer interval $[1, 2^k]$, there are queued $wC + k - (k - Z(v))$ packets requiring to traverse only the queue where they have been injected. Also, we assume that, initially at each leaf v with order m from left to right, where m takes on odd values from the integer interval $[1, 2^k]$, there are queued $wC + k - (k - Z(v)) + 1$ packets.

The out-going edge of each leaf v with order m ($1 \leq m \leq 2^k$) has unit capacity at the time interval $[0, Z(v))$. Therefore, at each one of these time steps one packet from the initial ones leaves the system at each out-going edge. Till the time $Z(v) + w$ all the packets that are queued initially in a leaf v have been absorbed. At time step $Z(v)$, the adversary injects $\frac{w\rho C}{2}$ packets to each leaf node v targeted with node $D(v)$. The path that is assigned to these paths has capacity C till the time $Z(v) + w - 1$, while after this time the path changes capacity to unit (only the edges that do not overlap with paths that are used for packet injections at other time steps).

The rest of the proof is similar to the corresponding part of the proof of Theorem 3.6. The only differences are that the injection time of packets at each node v is $Z(v)$ instead of $k - Z(v)$ and the initial packets at the leaves of $\mathcal{G}_k$ leave till time $Z(v) + w$ instead of $Z(v) + w - 1$. Thus, the total packet delay in the system is $\Omega(iwC\rho)$ and the largest queue required for the adversary construction is $wC + k$. For $k = C$, we take $wC + +C = (w+1)C$. An open question that remains is if there is a linear upper bound on the packet delay of SIS on DAGs. ∎

5 Stability Thresholds

In this section, we present upper bounds on stability thresholds. For any directed network $\mathcal{G}$ (possibly cyclic) denote $d(\mathcal{G})$ the length of the longest directed path that may be followed by any packet. We still consider the model of dynamic capacities [4] where each link capacity may take on integer values from $[1, C]$. We first show:

Theorem 5.1 *Let $\rho \leq \frac{1}{C(d(\mathcal{G})+1)}$. For any network $\mathcal{G}$, any adversary $\mathcal{A}_{w,\rho}$ and any greedy protocol P, the system $\langle \mathcal{G}, \mathcal{A}, \mathsf{P} \rangle$ is stable.*

Proof: It suffices to show that any packet that arrives at a queue at time t, leaves this queue by time $t + \lfloor w\rho \rfloor$. Our proof is by induction on time t. *Basis case:* Consider any $t \leq d(\mathcal{G})w\rho + 1$. Let p be a packet that arrives to the queue e at time $t \leq d(\mathcal{G})w\rho + 1$. The edge e has unit capacity in the time interval $[t, t + \lfloor w\rho \rfloor]$ in the worst-case (unit capacity permits the preservation of the biggest number of packets in the queue). Assume towards a contradiction that p is at the same queue at the end of time $t + \lfloor w\rho \rfloor$. This means that for each of the $\lfloor w\rho \rfloor$ time steps in $[t+1, t + \lfloor w\rho \rfloor]$ some other packet was sent over edge e due to the use of a greedy protocol.

From the definition of the adversary, the largest number of packets that can be injected into the system by the end of time $t + \lfloor w\rho \rfloor - 1$ requiring edge e is $\lfloor w\rho C \rfloor + 1$ packets because C is the capacity value of edge e at injection time that maximizes the number of packets that can be injected into the system requiring e (these are the packet p itself, and the $\lfloor w\rho C \rfloor$ packets that were sent over e). Since $t \leq d(\mathcal{G})w\rho + 1$, we have $t + \lfloor w\rho \rfloor - 1 \leq (d(\mathcal{G}) + 1)w\rho$. By the definition of the adversary, the number of packets that require e and they are injected by the end of any time step $t' \leq (d(\mathcal{G}) + 1)w\rho$ is at most

$\lceil (d(\mathcal{G})+1)\rho \rceil \lfloor w\rho C \rfloor$ because C is the capacity value of edge e at injection time that maximizes the number of packets that can be injected into the system requiring e. Since we assume $\rho \leq \frac{1}{C(d(\mathcal{G})+1)}$ this is at most $\lfloor w\rho \rfloor$. A contradiction to the fact that we identified $\lfloor w\rho \rfloor + 1$ packets.

Inductive hypothesis: Any packet that arrives at some queue at time $t' \leq t$, leaves the queue by time step $t' + \lfloor w\rho \rfloor$.

Induction step: We now prove the claim for any $t > d(\mathcal{G})w\rho + 1$. Let p be a packet that arrives to the queue e at some time t. Consider any packet that requires edge e and it was injected by time $t - d(\mathcal{G})\lfloor w\rho \rfloor$. By inductive hypothesis, we know that such a packet left the queue where it was injected by time $t - d(\mathcal{G})\lfloor w\rho \rfloor + \lfloor w\rho \rfloor$, left the next queue by time step $t - d(\mathcal{G})\lfloor w\rho \rfloor + 2\lfloor w\rho \rfloor$, etc. I.e., it arrived to its destination by time $t - d\lfloor w\rho \rfloor + d(\mathcal{G})\lfloor w\rho \rfloor = t$ (since the length of its path is at most $d(\mathcal{G})$, and all its "arrival times" are earlier than t, so the induction hypothesis holds). It follows that any packet that can delay packet p from going over edge e must be injected at time $t - w\rho d(\mathcal{G}) + 1$ or later.

Now assume towards a contradiction that packet p is still at queue e at the end of time $t + \lfloor w\rho \rfloor$. So, there are some other packets that crossed edge e in $[t+1, t+\lfloor w\rho \rfloor]$. These packets are present in the network at the end of time t or later, and they are injected by time $t + \lfloor w\rho \rfloor - 1$. However, we know that any packet injected by time $t - d(\mathcal{G})\lfloor w\rho \rfloor$ already left the network by the end of time t. Thus, those packets must have been injected in the time interval $[t - d(\mathcal{G})\lfloor w\rho \rfloor + 1, t + \lfloor w\rho \rfloor - 1]$. There are $\lfloor w\rho \rfloor (d(\mathcal{G}) + 1) - 1$ time steps in this interval. So, the number of packets that require e that can be injected during this interval is bounded by $\lceil (d(\mathcal{G})+1)\rho \rceil \lfloor w\rho C \rfloor$ because C is the capacity value of edge e at injection time that maximizes the number of packets that can be injected into the system requiring e. Since $\rho \leq \frac{1}{C(d(\mathcal{G})+1)}$ this is at most $\lfloor w\rho \rfloor$, a contradiction. ■

Now we will show how we can relax the obtained stability condition for *time-priority* protocols, such as FIFO and LIS. A *time priority* protocol is any greedy protocol that forwards a packet arriving at a queue at time t against any other packet that is injected into the system after time t. We show:

Theorem 5.2 *Let $\rho \leq \frac{1}{Cd(\mathcal{G})}$. For any network $\mathcal{G}$, any adversary $\mathcal{A}_{w,\rho}$ and any time-priority protocol* P*, the system $\langle \mathcal{G}, \mathcal{A}, \mathsf{P} \rangle$ is stable.*

Proof: It suffices to show that any packet that arrives at a queue at time step t, leaves this queue by time $t + \lfloor w\rho \rfloor$. Our proof is by induction on time t.

Basis case: Consider any $t \leq d(\mathcal{G})w\rho + 1$. We will use contradiction in order to prove it. Let p be a packet that arrives at queue e at time $t \leq d(\mathcal{G})w\rho$. Assume towards a contradiction that p is at the same queue at the end of time step $t + \lfloor w\rho \rfloor$. Since the protocol is greedy and the edge e has unit capacity in the time interval $[t, t + \lfloor w\rho \rfloor]$ in the worst-case (unit capacity permits the preservation of the biggest number of packets in the queue), there are at least $\lfloor w\rho \rfloor + 1$ packets that traverse edge e in the time interval $[t, t + \lfloor w\rho \rfloor]$ (these are p, and the packets that were sent over e in the interval $[t, t + \lfloor w\rho \rfloor]$). However, since the protocol is timepriority, these packets must have been injected into the system by the end of time step t (otherwise they cannot delay p). By definition of the adversary, the number of packets that may use e and they are injected by the end of time t is at most $\lceil t/w \rceil \lfloor w\rho C \rfloor \leq \lceil d(\mathcal{G})\rho \rceil \lfloor w\rho C \rfloor \leq \lfloor w\rho \rfloor$, since $\rho \leq 1/Cd(\mathcal{G})$, and C is the capacity value of edge e at injection time that maximizes the number of packets that can be injected into the system requiring e, a contradiction.

Inductive hypothesis: Any packet that arrives at some queue at time $t' \leq t$, leaves the queue by time step $t' + \lfloor w\rho \rfloor$.

Induction step: Let $t > d(\mathcal{G})w\rho$, and assume by induction that any packet that arrives at some queue at time $t' \leq t$, leaves this queue by time $t' + \lfloor w\rho \rfloor$. Let p be a packet that arrives at the queue of edge e at t. Assume towards a contradiction that packet p is still at e at the end of time $t + \lfloor w\rho \rfloor$. The edge e has unit capacity in the interval $[t, t + \lfloor w\rho \rfloor]$ in the worst-case. Then, there is a set of $\lfloor w\rho \rfloor + 1$ distinct packets that use edge e in the time interval $[t, t + \lfloor w\rho \rfloor]$. Since we have a timepriority protocol, all these packets were injected by the end of time t.

Moreover, we now prove that all these packets were injected at time $t - w\rho d(\mathcal{G}) + 1$ or later. Consider any packet q that was injected by time

$t - d(\mathcal{G})\lfloor w\rho \rfloor$. By induction, q left the first queue on its path by time $t - d(\mathcal{G})\lfloor w\rho \rfloor + \lfloor w\rho \rfloor$, left the next queue by time $t - d(\mathcal{G})\lfloor w\rho \rfloor + 2\lfloor w\rho \rfloor$, and so on. Hence, q arrived at its destination by time $t - d(\mathcal{G})\lfloor w\rho \rfloor + d(\mathcal{G})\lfloor w\rho \rfloor = t$: since the length of its path is at most $d(\mathcal{G})$, all its "arrival times" are earlier than t, so we may apply the inductive hypothesis. So, all the $\lfloor w\rho \rfloor + 1$ packets that use e in the interval $[t, t + \lfloor w\rho \rfloor]$ must have been injected in the interval $[t - d(\mathcal{G})\lfloor w\rho \rfloor + 1, t]$. There are $\lfloor w\rho \rfloor d(\mathcal{G})$ steps in this interval, and therefore the number of packets that require e and they can be injected during this interval is bounded by $\lceil d(\mathcal{G})\rho \rceil \lfloor w\rho C \rfloor$ since C is the capacity value of edge e at injection time that maximizes the number of packets that can be injected into the system requiring e. Since $\rho \leq 1/Cd(\mathcal{G})$ the number of packets that are injected during the interval $[t - d(\mathcal{G})\lfloor w\rho \rfloor + 1, t]$ is at most $\lfloor w\rho \rfloor$, a contradiction. ■

6 Conclusion

In this work, we studied the impact of dynamically changing *link capacities* (link capacities can take on integer values from $[1, C]$ with $C > 1$) on the performance bounds of LIS and SIS protocols on DAGs and we obtained stability bounds for greedy protocols running on arbitrary networks. We proved that LIS and SIS protocols have linear performance bounds on DAGs and that there are polynomial stability thresholds for greedy protocols on arbitrary networks depending on the maximum link capacity C and the length of the longest network path.

A careful inspection of the proofs of the lower bound for LIS obtained in this work reveals that they also hold in the more restrictive model, where link capacities stay fixed for time at least a constant of the number of packets in the system at the time when the capacity was last set. This model is called Quasi-Static Link Capacities Model [9]. Therefore, the tight bound we prove here for LIS implies some kind of collapse of a powerful model (Model of Dynamic Capacities) to a weaker one (Quasi-Static Link Capacities Model) that is, rather surprising.

An interesting avenue for further research would be to determine performance bounds for other stable protocols on DAGs such as FTG and NTS in the model of dynamic capacities.

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. Adler and A. Rosén, "Tight Bounds for the Performance of Longest-in-System on DAGs," *Proceedings of the 19th Annual Symposium on Theoretical Aspects of Computer Science,* LNCS 2285, pp. 88–99, March 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] A. Borodin, R. Ostrovsky and Y. Rabani, "Stability Preserving Transformations: Packet Routing Networks with Edge Capacities and Speeds," *Proceedings of the 12th Annual ACM-SIAM Symposium on Discrete Algorithms,* pp. 601–610, January 2001.

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

[6] 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.

[7] D. Koukopoulos, M. Mavronicolas, S. Nikoletseas and P. Spirakis, "On the Stability of Compositions of Universally Stable, Greedy, Contention-Resolution Protocols," *Proceedings of the 16th International Symposium on DIStributed Computing,* LNCS 2508, pp. 88–102, October 2002.

[8] D. Koukopoulos, M. Mavronicolas, S. Nikoletseas and P. Spirakis, "The Impact of Network Structure on the Stability of Greedy Protocols," *5th Italian Conference on Algorithms and Complexity,* accepted.

[9] D. Koukopoulos, M. Mavronicolas and P. Spirakis, "Instability of Networks with Quasi-Static Link Capacities," *10th International Colloquium on Structural Information and Communication Complexity,* accepted.

[10] D. Koukopoulos, S. Nikoletseas, and P. Spirakis, "Stability Issues in Heterogeneous and FIFO Networks under the Adversarial Queueing Model," Invited Keynote Address, *Proceedings of the 8th International Conference on High Performance Computing*, pp. 3–14, December 2001.

[11] Z. Lotker, B. Patt-Shamir and A. Rosén, "New Stability Results for Adversarial Queueing," *Proceedings of the 15th Annual ACM Symposium on Parallel Algorithms and Architectures,* pp. 192–199, August 2002.