Stability Preserving Transformations: Packet Routing Networks with Edge Capacities and Speeds

Allan Borodin* Rafail Ostrovsky† Yuval Rabani‡

Abstract

In the context of an adversarial input model, we consider the effect on stability results when edges in packet routing networks can have capacities and speeds/slowdowns. In traditional packet routing networks, every edge is considered to have the same unit capacity and unit speed. We consider both static modifications (i.e. where the capacity or speed of an edge is fixed) and dynamic modifications where either the capacity or the speed of an edge can be dynamically changing over time. Amongst our results, we show that the universal stability of LIS is not preserved when either the capacity or the speed is changing dynamically whereas many other common scheduling protocols do maintain their universal stability. In terms of universal stability of networks, stability is preserved for dynamically changing capacities and speeds. The situation for static modifications is not as clear but we are able to show that (in contrast to the dynamic case) that any "well defined" universally stable scheduling rule maintains its universality under static capacities, and common scheduling rules also maintain their universal stability under static speeds.

*Department of Computer Science, University of Toronto, Toronto, Canada M5S 3G4. Part of this work was performed while visiting Telcordia Technologies and the Technion Computer Science Department. Email: bor@cs.toronto.edu

†Telcordia Technologies, MCC-1C357B, 445 South Street, Morristown, New Jersey 07960-6438, USA. Email: rafail@research.telcordia.com; URL: http://www.argreenhouse.com/bios/rafail/index.shtml

‡Computer Science Department, Technion — IIT, Haifa 32000, Israel. Part of this work was done while visiting Telcordia Research. Work at the Technion supported by BSF grant 96-00402, by Ministry of Science contract number 9480198, and by a grant from the Fund for the Promotion of Research at the Technion. Email: rabani@cs.technion.ac.il

1 Introduction

We continue the study of adversarial packet routing networks as initiated in Borodin *et al.* [6], and significantly advanced in Andrews *et al.* [2], Aiello *et al.* [1], Gamarnik [8], Andrews and Zhang [5], and Andrews [4]. Briefly stated these papers analyze stability and queue sizes for various networks and greedy (work preserving) scheduling rules when (uniform size) input packets are being generated by an adversary. A greedy scheduling rule insures that some packet crosses a given edge (link) if the queue for that edge is non-empty. For a given input process, a network is stable with respect to a scheduling rule if all edge queues are bounded. (Here the bound may depend on the network but does not depend on the duration of the process.) A network is universally stable (for for the adversarial input processes being considered) if any greedy scheduling rule can be used and stability is guaranteed. Similarly, a scheduling rule

is universally stable if stability is guaranteed for any network.

Most studies of packet routing networks assume that one packet can cross an edge in a single time step. This assumption is well motivated when we assume that all edges (= communication links) are identical. However, it is also reasonable to assume that a packet routing network could contain different types of links, in which case we would need to assign a capacity and/or a speed to each edge. Note that we are still assuming uniform packet sizes so that the capacity or speed at which packets pass through a particular edge does not depend on the packet. Once we allow the service time of an edge to depend on the packets we are effectively assuming a general multiclass queuing network. An adversarial analysis of general multiclass queuing networks was begun in Tsaparas [12].

In this paper, we consider the impact on stability results when edge capacities and speeds are introduced.[1] These edge capacities or speeds may either be static or dynamic (i.e. changing over time). In the case of dynamic edge capacities or slowdowns, we assume that an adversary is setting these capacities (or slowdowns) as well as determining the packets (i.e. paths) being injected at each time step. As a special case of dynamic speeds or capacities, we are in effect approximating the fault tolerance of a network where edges can temporarily fail (i.e. have infinite slowdown or zero capacity). We shall show (under a very natural assumption on the class of scheduling rules being considered) that the property of a rule being universally stable is preserved in the context of static edge capacities. However, the universal stability of a scheduling rule is not necessarily preserved in the context of dynamic capacities with LIS (i.e. the scheduling rule that gives priority to the *longest in system* packet) being a notable example of a rule that does not remain universally stable.

We have not yet been able to show that universal stability of a rule is preserved in the context of static edge speeds but all known rules previously studied enjoy this property. Indeed many rules also remain universally stable in the context of dynamic edge slowdowns. However, as in the case of dynamic edge capacities, LIS is again a notable example where universal stability is not preserved. Finally, with regard to the universal stability of networks, we can show that universal stability is preserved under dynamic edge capacities and speeds.

2 Definitions

We assume the reader is familiar with the basic definitions of a greedy scheduling rule, stability of a network system $(G, \Upsilon, \mathcal{S})$, and universal stability of a rule or network (see, for example, Borodin *et al.* [6]). These definitions were given in the context of "standard" oblivious packet routing where all edges have a uniform capacity and uniform speed of one packet/step and each packet has a fixed simple path (independent of other packets) it must traverse. We now want to consider networks in which edges can have different integer capacities or speeds which may or may not vary over time. We will let $c_e(t)$ denote the capacity of edge e at time step t. That is, we assume edge e is capable of simultaneously transmitting up to $c_e(t)$ packets at time t. When this capacity does not depend on time we simply write c_e.

The definition for the (time varying) speed of an edge is a little more problematic. For the purpose of this paper we will restrict ourselves to a synchronous framework. We will assume we know the maximum speed possible (normalized to 1 packet/step). We let the positive integer $s_e(t)$ denote the *slowdown* of edge e at time t. To make the semantics of this slowdown precise, we

[1] In this paper, we only study the effect of introducing either capacities or speeds (but not both).

chose the following interpretation which allows us to maintain a synchronous view of packet routing. If a packet P is scheduled to traverse link e at time t and at time t the slowdown of this link is $s_e(t)$, then packet P completes the traversal of e at time $t + s_e(t)$ and during this interval of time, no other packet can be scheduled on e. And again we simply write s_e for the case of static slowdown.

Let w be an arbitrary positive integer, e any edge in the network and τ any sequence of w consecutive time steps. We define $N(\tau, e)$ to be the number of packets injected by the adversary during time interval τ that traverse edge e.

Definition. Consider the case of edges with capacities. For any $\rho, 0 < \rho \leq 1$, we define a (w, ρ) *adversary* as an adversary which injects packets (= paths) subject to the following *load condition*: for every sequence τ of w consecutive time steps and for every edge e, $N(\tau, e) \leq \rho \sum_{t \in \tau} c_e(t)$.

Definition. Now consider the case of edges with slowdowns. Again, let $0 < \rho \leq 1$. We first consider the case of static slowdowns and define a (w, ρ) *adversary* as one which injects paths subject to the following load condition: for every sequence τ of w consecutive time steps and for every edge e, $N(\tau, e) \leq \rho \sum_{t \in \tau} \frac{1}{s_e} = \rho w \frac{1}{s_e}$.

The definition for an adversary in the context of the dynamic slowdown model requires some care. The most obvious extension of the static model condition is : $N(\tau, e) \leq \rho \sum_{t \in \tau} \frac{1}{s_e}(t)$. However, our definition of edge slowdown , $s_e(t)$ at a given time t, impacts the effective speed of the edge e for the next $s_e(t)$ time steps. Indeed because we are only considering greedy scheduling rules, an adversary can start a packet on edge e at time t and then this edge is unavailable for the next $s_e(t)$ time steps no matter what values are given for the slowdowns $s_e(t')$ for $t < t' < t + s_e(t)$. Hence for any time t', we define the effective slowdown $\tilde{s}_e(t)$ at time t' to be the $max_t[s_e(t) | t \leq t' < t + s_e(t)]$. Now we can define a (w, ρ) *adversary* as one which injects paths subject to the load condition: for every sequence τ of w consecutive time steps and for every edge e, $N(\tau, e) \leq \rho \sum_{t \in \tau} \frac{1}{\tilde{s}_e(t)}$ Note that this definition coincides with the definition for the static slowdown model since in this case, $\tilde{s}_e(t) = s_e(t) = s_e$.

For either (static or dynamic) capacities or slowdowns, we say that a (w, ρ) adversary injects packets at *rate* ρ with *window size* w. A rate ρ adversary is a (w, ρ) adversary for some w.

Definition. Let G be a network (with or without edge capacities or slowdowns) and $\mathcal{S}$ a scheduling rule. A network environment $(G, \mathcal{S})$ is ρ-stable if for every initial configuration [2] $C_0(G)$, and every w, there is a bound $B = B(C_0(G), w)$ such that for any rate ρ adversary (with window w), the size of every queue is bounded by B at any time. A scheduling rule $\mathcal{S}$ is universally ρ-stable if (G, S) is ρ-stable for all G. Similarly, a network G is universally ρ-stable if (G, S) is ρ-stable for all greedy [3] scheduling rules $\mathcal{S}$.

Examples. For uniform capacities and speeds, any ring is a universally ρ-stable network for any $\rho < 1$ ([2]) and any DAG is 1-stable ([6]). Andrews *et al.* [1] show that a number of scheduling rules, namely "longest in system" (LIS), "farthest to go" (FTG) and "shortest in system" (SIS) are universally ρ-stable scheduling rules for any $\rho < 1$. Gamarnik [9] shows that "nearest to origin" (NTO) is universally 1-stable . However, $FIFO$ is not universally ρ-stable for $\rho \geq .85$ (Andrews *et al.* [1]) and NTG is not

[2] The initial configuration is not really significant for the purposes of this paper and henceforth it will be ignored.

[3] In this context, a greedy scheduling rule is one that always sends as many packets as available across an edge, up to its capacity.

universally ρ-stable for any $\rho > 0$ (Borodin *et al.* [6]).

Definition. A greedy scheduling rule $\mathcal{S}$ is *capacity (respectively, speed) invariant* iff for every set A of packets in an edge queue, $\mathcal{S}$ induces a total order on the packets in A that is independent of the capacities (respectively, speeds) of the network edges, and furthermore, for all $A' \subseteq A$, the total order induced by $\mathcal{S}$ on A' is consistent with the total order induced by $\mathcal{S}$ on A. Equivalently, $\mathcal{S}$ is capacity invariant if the relative order (imposed by the rule) of any pair of packets in a queue is independent of the capacity of the edges and independent of other packets. We claim that this is a very natural assumption as it insures that the definition of the rule when applied at an edge (having an arbitrary capacity) is unambiguous.[4] Since the context will always be clear, we will simply say "invariant (scheduling) rule".

Examples. $FIFO$, $LIFO$, FTG, NTG, SIS, LIS, NTO are all invariant rules. In fact, all natural scheduling rules are invariant rules. As an example of a non-invariant rule, consider the rule of using LIS if the number of packets in the queue is less than 10, and $LIFO$ otherwise.

3 Networks with Edge Capacities

Theorem 3.1. There is a standard packet routing network G (namely the 4 node baseball graph introduced by Andrews *et al.* [2] such that when G is allowed to have dynamic capacities, (G, LIS) is not stable. (Recall that LIS is universally stable with regard to standard packet routing networks against any rate $\rho < 1$ adversary.) More specifically, if G' denotes G when all dynamic edge capacities are either 1 or C, then (G', LIS) can be unstable for rate $\rho > \frac{C}{2C-1}$.

Proof. The proof is motivated by the construction in Andrews *et al.* [2] showing that for the baseball graph, neither NTG nor FIFO are ρ-stable for a sufficiently large ρ. See Figure 1

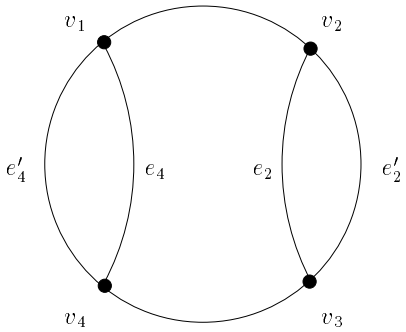

Figure 1: Baseball graph

As in the Andrews *et al.* proof for NTG, we assume that the packets are injected in stages and each stage consists of two substages. At the start of a stage we assume that there is a set X of at least t packets queued at nodes v_4 and v_1 (with at least one packet at node v_1) that need to traverse edges e_4, e_1, e_2. It suffices to show that at the end of the stage, more than t packets will be queued at nodes v_2 and v_3 (with at least one packet at node v_3)waiting to traverse edges e_3, e_4. To simplify the proofs we avoid the use of floors and ceilings. Let ρ be the rate of an adversary which sets the capacities of the edges and injects packets as follows:

1. For the first $\frac{1}{\rho C+1} \cdot t$ steps, all edges have capacity C except for edge e_2 which has capacity 1. (Note that when C is large, we are relatively shutting down edge e_2.) During these steps, the adversary injects a

[4] We would certainly not want to allow the rule "LIS if all edges have capacity one, else FIFO". On the other hand the "speed invariant" restriction prohibits a rule such as "most time to go" which is a natural generalization of FTG. It is not clear what is the most appropriate restriction for studying networks with edge speeds.

set Y of $\frac{\rho C}{\rho C+1} \cdot t$ packets that need to traverse the path e_4, e_1, e_2', e_3. By the nature of the LIS rule, all of these Y packets are blocked by the X packets as they enter node v_1. At the end of this substage, there are $\frac{\rho C}{\rho C+1} \cdot t$ packets in X (respectively, in Y) that still need to traverse edge e_2 (respectively, the path e_4, e_1, e_2', e_3).

2. For the next $\frac{\rho C}{(\rho C+1)^2} \cdot t$ steps. all edges have capacity C except for edge e_2' which has capacity 1. The adversary now injects a set Z of $[\frac{\rho C}{(\rho C+1)}]^2 \cdot t$ packets which need to traverse the path e_2, e_3, e_4. All the packets in Z will be blocked by the X packets. Hence at the end of this substage, we have $|Y|+|Z| \geq 2[\frac{\rho C}{\rho C+1}]^2 \cdot t$. For $\frac{\rho C}{\rho C+1} > \sqrt{(1/2)}$, there are then more than t packets queued at nodes v_2 and v_3 (with at least one packet at node v_3) that need to traverse edges e_3 and e_4. Note that for $C \geq 3$, we can set $\rho < 1$.

■

However, we now show that the property of an invariant scheduling rule being universally ρ-stable is preserved when adding *static* capacities to edges.

Theorem 3.2. Let $\mathcal{S}$ be an invariant scheduling rule, which is universally ρ-stable for standard unit capacity networks. Then, for any $\epsilon > 0$, $\mathcal{S}$ is universally $(\rho - \epsilon)$-stable for integer capacity networks.

Proof. Intuitively, a directed graph $G = (V, E)$ having edge capacities c_e for $e \in E$, can be thought of as a non-simple, unit capacity graph $\tilde{G}$ with c_e parallel edges [5] replacing each capacitated edge e. Assume (for contradiction) that there exists $\epsilon > 0$ such that (G, S) is not $(\rho - \epsilon)$-stable. Therefore, for all w and for all B, there exists a $(w, \rho - \epsilon)$ adversary that produces a finite sequence of packet injections which causes some queue to exceed size B. We would like to simulate $\mathcal{S}$ on G by using $\mathcal{S}$ on $\tilde{G}$. The simulation will be such that we can argue that the queue size in $\tilde{G}$ exceeds B/c where $c = \max_e c_e$. This will contradict the ρ-stability of $(\tilde{G}, S)$. The issue is how to assign packets to the parallel edges so that the rate condition is not violated. Resolving this issue requires some care.

Conceptually, for any edge e and time t, we consider the set $A_{e,t}$ of the first c_e packets queued at e (in the order induced by $\mathcal{S}$), or all the packets queued at e, if there are fewer than c_e packets there. Map the packets in $A_{e,t}$ to the c_e parallel edges in $\tilde{G}$ by a uniformly at random matching. This induces a probability distribution over mappings of packets to the edges of $\tilde{G}$. Note that because the adversary (producing the large queue size in G) is finite, every packet eventually reaches every edge in its path (in G) and thus the mapping of packets to edges in $\tilde{G}$ is well defined. We shall argue that there exists a point in this distribution such that the induced injection rate on each edge of $\tilde{G}$ preserves the rate less than ρ constraint. The theorem follows as the packet traversals scheduled by $\mathcal{S}$ in $\tilde{G}$ are exactly the same as the traversals scheduled by $\mathcal{S}$ in G (this follows from the invariant property of $\mathcal{S}$). It is worth noting that we do not have to construct the matching of packets to parallel edges; we only need to know it exists.

Let us now consider a $(\rho - \epsilon, w)$ adversary for some sufficiently large window w. Consider

[5] However, this intuition cannot always be applied since the baseball graph can also be thought of as a 4 node ring with two edges having capacity 2. Since the ring is a universally stable network, *FIFO* is stable on any ring but *FIFO* is not stable on the baseball graph. Although it is possible that the theorem holds for any network system $(G, \mathcal{S})$, our proof relies on the assumption that $\mathcal{S}$ is universally stable.

windows beginning at time j for $j = 0, 1, \ldots$. Let W_j denote the j^{th} such window. The random matchings discussed above are now viewed as randomly choosing a selected edge (let's call it a slot) in $\tilde{G}$ for each edge (in G) in the path of a packet injected in this window. We say that a window W_j is *good* if for every slot in $\tilde{G}$ there are at most $(\rho - \epsilon/2) \cdot w$ packets injected during W_j that are assigned to this slot by the random mapping. We want to show that there is a positive probability that *every* window will be good and hence there is a mapping which induces an injection by a $(\rho - \epsilon/2, w)$ adversary in $\tilde{G}$.

Of course, if W_j and $W_{j'}$ overlap then the events "W_j is good" and "$W_{j'}$ is good" are not independent. Moreover, if packets p (injected during W_j) and p' (injected during $W_{j'}$) are involved in the same matching, then again the events "W_j is good" and "$W_{j'}$ is good" are not independent, even if these windows do not overlap. However, for a given window W, the event "W is good" is independent of any combination of other such events excluding the above mentioned events, the number of which is bounded by $2w + [(\rho - \epsilon) \cdot w \cdot \sum_e c_e]^2 = \alpha(G)w$ (where $\alpha(G)$ is a constant that only depends on the graph G) The fact that dependency is limited will allow us to invoke the Lovász Local Lemma (see [11]).

We now bound the probability that a window W is not good. Consider an edge e in G and the $t \leq (\rho - \epsilon)wc_e$ packets injected during W that may need to traverse e. If it were not for the fact that the random matchings impose a condition on which slots can be assigned to a packet, we could view the random process of assigning slots to packets as a traditional balls and bins experiment. Namely, we are throwing $(\rho - \epsilon)wc_e$ balls (=packets) at c_e bins (=slots). The expected number of packets assigned to a slot is obviously $(\rho - \epsilon)w$ and using the Chernoff bound, the probability that the number of packets assigned to a slot exceeds $(\rho - \epsilon/2) \cdot w$ is at most $e^{-\Omega(w)}$. However, this analysis is flawed because the balls participating in any single matching are not being thrown independently. Intuitively, the fact that the balls have to satisfy a matching constraint should only help to reduce the maximum congestion on a slot. We can make this intuition precise by the following argument.

We consider the process of sequentially throwing balls into a given slot. Define a sequence of random variables $Y_0, Y_1, \ldots Y_t$ where Y_0 is the expected number of balls that will end up in this slot and Y_i is the same expectation after i balls have been thrown. Note that $Y_{(\rho-\epsilon)w}$ is the final congestion on this slot. Clearly by linearity of expectations, $Y_0 = (\rho - \epsilon)w$. By definition $Y_i = E[Y_{i+1}|Y_i]$. Moreover, we claim $|Y_{i+1} - Y_i| \leq 1$. This follows because any ball is correlated by the matching with at most c_e other balls in the sequence and the contribution of all these balls to the congestion is at most 1. Hence the sequence is a martingale. We now can calculate an upper bound on the probability that the final congestion on a slot is more than $((r - \epsilon/2)w = Y_0 + w\epsilon/2 = Y_0 + \lambda\sqrt{t}$ for $\lambda = w\epsilon/2\sqrt{t} \geq \frac{\epsilon\sqrt{w}}{2\sqrt{(\rho-\epsilon)c_e}}$. By Azuma's inequality [11], this probability is less than $e^{-\lambda^2/2} \leq e^{-\frac{\epsilon^2}{8(r-\epsilon)c_e}w}$.

By the union bound, the probability that any slot is overcongested (that is, that the window W is not good) is at most $\sum_e c_e \cdot e^{-\frac{\epsilon^2}{8(\rho-\epsilon)c_e}w}$. To apply the local lemma we need this probability to be at most $1/e(d+1)$ where $d = \alpha(G)w$ is the bound on the dependency. This clearly holds for sufficiently large w.

■

We can now apply Theorem 3.2 to show that (in contrast to LIS), some common scheduling rules do remain universally stable with respect to dynamically changing capacities.

Theorem 3.3. The SIS and NTO scheduling rules are universally stable for dynamically changing integer capacity networks.

Proof. (Sketch) For simplicity assume that there is a known bound c on the largest capacity allowed. (This assumption can be removed by noting as in the proof of Theorem 3.2 that if there was a counterexample to universal stability then this counterexample would only use a finite set of capacities.) Next we observe that for the SIS and NTO rules, newly injected packets will take priority over packets in the system. Now at any point of time t, suppose we have an edge e with capacity $c_e(t) < c$. Then we can inject $c - c_e(t)$ "dummy" packets which only need to traverse this edge. This can be done without violating the load condition for a network in which every edge has (static) capacity c. Now we appeal to Theorem 3.2 to obtain the desired result. ■

We now turn our attention to universally stable networks. From the results of Goel [10] we know that there is a nice characterization of the class of directed graphs which are universally stable (in the context of unit edge capacities). Goel's characterization is based on the following facts:

- The unidirectional cycle and all DAGS are universally stable.
- A digraph is universally stable if and only if all of its strongly connected components are universally stable.
- The unidirectional cycle is the only strongly connected digraph that is universally stable.[6]

[6] Here for simplicity we are assuming simple paths. We can then superimpose two unidirectional cycles on the same set of nodes and have a universally stable network. These two unidirectional cycles clearly do not interfere with each other and one can argue about each cycle separately.

One can then establish the same facts for the arbitrary capacity model and obtain:

Theorem 3.4. Let G be a universally stable network in the unit capacity model. Then G remains stable in the context of dynamic edge capacities.

Proof Sketch. Using the idea of dummy packets, it is easy to see that by modifying a given scheduling rule so that it gives priority to these dummy packets, we can assume that every edge has a fixed (static) capacity c. We then must show that the universal stability proofs for DAGs and for the cycle can be generalized to the case of a fixed static capacity c. The proof is then completed following the characterization of Goel [10]. We briefly indicate how to modify the DAG and cycle proofs:[7]

- For DAGs, we modify the ψ function in Theorem 1 of [7]. Let edge e have edges $f_1, \ldots, f_k$ entering the tail of e. For our case, the inductive definition is then:

  $$\psi(e) = \max\{2c \cdot w, Q_0(e)\} + \sum_{i=1}^{k} \psi(f_i).$$

  The goal is to show that for all $t = l \cdot w \geq 0$ and all $e \in G$, we have

  $$A_t(e) \leq \psi(e) \tag{3.1}$$

  where $A_t(e)$ denotes the number of packets (not already absorbed) that have arrived by time t and are eventually destined to cross edge e. One then argues by induction on l and by cases, according to whether or not $A_{t-w}(e) \leq c \cdot w + \sum_{i=1}^{k} \psi(f_i)$.

[7] There are some minor differences in notation and in the definitions of a rate $1 - \epsilon$ adversary as they appear in [7] and [3]. The former paper incorporates the initial queues $Q_0(e)$ and the latter paper dispenses with the notion of a window in favor of an additive constant. For simplicity we will just indicate how to modify the proofs as they appear in these papers.

- For the unidirectional cycle, we modify the definition of the f function in the proof of Theorem 3.7 of [3]. For our case, we need

$$f(j, T_0) = Q + c(b+1)(j - i_0)$$

$$f(j,t) = Q - c\cdot\epsilon(t-T_0) + c\cdot(b+1)(1+j-i_0)$$
$$\text{for } t > T_0.$$

The goal is to show that for all applicable pairs (j,t), $P_{j,t} \leq f(j,t)$ where $P_{j,t}$ denotes the number of packets (not already absorbed) that have arrived by time t and are eventually destined to cross edge j. Essentially the c in the term $c\cdot(b+1)(1+j-i_0)$ is sufficient to modify the proof of Lemma 3.6 where one argues by cases depending on whether or not c packets crossed edge j in the past $(t-T_0)$ consecutive steps. The c in the term $c\cdot\epsilon(t-T_0)$ is needed for the rest of the proof in Theorem 3.7.

■

4 Networks with Edge Slowdowns

We will now see that every universally stable network remains universally stable when edges can have slowdowns and this holds even for dynamic edge slowdowns.

Theorem 4.1. Let G be a universally stable network at every rate $\rho < 1$ in the standard packet routing context. Consider any scheduling rule $\mathcal{S}$ and any execution of the derived network system $(G', \mathcal{S})$ in which the inputs are being generated by an adversary in the context of dynamic edge slowdowns (see Definition 2). Then the system $(G', \mathcal{S})$ remains stable for all rates $\rho < 1$.

Proof. The idea is similar to Theorem 3. Essentially we want to simulate the behavior of the network system $(G', \mathcal{S})$ by a standard packet routing system $(G, \mathcal{S}')$. We do so by delaying (real) packets at a slow edge by using newly injected dummy packets traversing that edge alone. We give the dummy packets higher priority in order to force the delay of the real packets. This remains a greedy rule (call it $\mathcal{S}'$) and hence we are assured stability for $(G, \mathcal{S}')$ since G is universally stable.

More specifically, consider a rate ρ adversary with window w for the network G' with slowdowns. To simplify the discussion let's first assume that the edge slowdowns are static with integer $s_e \geq 1$ being the slowdown of edge e. Then in any window of w steps there are at most $w \cdot \frac{\rho}{s_e}$ packets injected that need to traverse edge e. Consider an edge with slowdown $s_e \geq 2$. Then during these w time steps there are at least $w \cdot \frac{s_e - \rho}{s_e}$ steps in which no packets are injected that need to traverse edge e. When a packet P is ready to traverse edge e we first inject $s_e - 1$ dummy packets that only need to traverse edge e and which are given priority over packet P. Then packet P completes its traversal of e in s_e steps. The injection rate of the derived adversary is still less than 1 (with the same window w) and hence we are indeed assured stability. The same proof applies to dynamically changing slowdowns. When there is a packet P ready to traverse edge e at time t, the adversary injects $\tilde{s}_e(t) - 1$ dummy packets.

■

The idea in the above proof can be used to show that certain scheduling rules remain universally stable in the context of (dynamically changing) edge slowdowns. For example, consider farthest to go (FTG) and any packet routing network $G = (V, E)$. We can modify G so that from every node $v \in V$, there is a path Π_e of length $|V| + 1$ directed away from V. Now to simulate a slowdown of $\tilde{s}_e(t)$ on edge e at time t, if a packet P wants traverse e, the adversary injects $\frac{\tilde{s}_e(t)-1}{\tilde{s}_e(t)}$ dummy packets that need to traverse e and then Π_e. Since these dummy packets

have the farthest distance to go they have priority over P and hence delay P for $\tilde{s}_e(t)$ steps.

For SIS and NTO the same idea (of inserting dummy packets) can used and in these cases we do not have to add extra edges since priority of the dummy packets is ensured by the definition of the rule (assuming that dummy packets have priority over real packets originating at the same node. We thus have:

Theorem 4.2. SIS, NTO and FTG remain universally stable in the context of dynamically changing edge slowdowns.

This simple dummy packet idea does not directly extend to the LIS rule. For static edge slowdowns, we know that LIS remains universally stable by the result of Tsaparas [12] who shows that LIS remains universally stable even in the context that for each edge and each packet P, there is a given speed or alternatively (since we are assuming a maximum speed) a given slowdown $s_e(P)$ defining the speed at which P traverses e. However, the situation is different for LIS and dynamically changing edge slowdowns as we now see.

Theorem 4.3. LIS is not ρ stable for dynamically changing edge slowdowns for any $\rho > \frac{1}{2(\frac{D-1}{D})}$ where D is the maximum slowdown allowed.

Proof. The proof is quite similar to the proof of Theorem 3.1. We again assume that the packets are injected in stages and each stage consists of two substages. At the start of a stage we now assume that there is a set X of at least t packets queued at node v_4 that need to traverse edges e_4 or e'_4, followed by edges e_1, e_2. It suffices to show that at the end of the stage, more than t packets will be queued at node v_2 waiting to traverse edges e_2 or e'_2 followed by edges e_3 and e_4. Again, to simplify the proofs we avoid the use of floors and ceilings. Let ρ be the rate of an adversary which sets the slowdowns of the edges and injects packets as follows: Again to simplify the proofs, we abuse the defintion of a rate ρ adversary and ignore the fact that the adversary must observe the "effective slowdown" load condition. The justification for this abuse is that when t is sufficiently large (compared to D), the adversary can utilize the full speed edges (i.e. slowdown = 1) for almost all steps in the phase.

1. For the first t steps, no edge has a slowdown except edge e_2 which has slowdown D. During these t steps, the adversary injects a set Y of $\rho \cdot t$ packets that need to traverse the path e_1, e'_2, e_3. By the nature of the LIS rule, all of these Y packets are blocked by the X packets. At the end of this substage, there are (approximately) $\frac{D-1}{D} \cdot t$ packets in X that still need to traverse edge e_2 and ρt packets in Y that need to traverse the path e_1, e'_2, e_3.

2. For the next $\frac{D-1}{D} \cdot t$ steps no edge has a slowdown except edge e'_2 which has slowdown D. During these steps, the adversary injects a set Z of $\rho\frac{D-1}{D} \cdot t$ packets that need to traverse the path e_2, e_3, e_4. All the packets in Z will be blocked by the X packets and because of the slowdown in edge e'_2 there will still be $\rho\frac{D-1}{D} \cdot t$ packets in Y. Hence at the end of this substage, we have $|Y| + |Z| \geq 2\rho\frac{D-1}{D} \cdot t$. For $\rho > \frac{1}{2(\frac{D-1}{D})}$, this number exceeds t. Note that for $D \geq 3$, we can set $\rho < 1$.

■

5 Conclusions and Open Problems

We have shown that that LIS does not have the dynamic stability properties of other universally stable scheduling rules. We have also shown that (for capacity invariant rules) universal stability

is preserved under static capacities, and (for known rules) under static slowdowns. There are many open problems that remain including the following:

- Is there a natural definition of a "speed invariant" rule such that universal stability of a speed invariant scheduling rule is preserved under static slowdowns?
- For a given network system $(G, \mathcal{S})$, is stability preserved for either static capacities or slowdowns?
- Can these stability preserving results be extended to an asynchronous framework (i.e. for arbitrary real valued speeds)?

Acknowledgements

We thank Eyal Kushilevitz and Yishay Mansour for helpfull discussions in the initial stages of this research. We also wish to thank Adi Rosén for pointing out an inaccurate definition in an earlier version of the paper.

References

[1] W. Aiello, E. Kushilevitz, R. Ostrovsky, A. Rosén Adaptive Packet Routing for Bursty Adversarial Traffic. In *Proc. of the 30th Ann. ACM Symposium on the Theory of Computing* (STOC), 359-368, 1998.

[2] M. Andrews, B. Awerbuch, A. Fernández, J. Kleinberg, F.T. Leighton, Z. Liu. Universal Stability Results for Greedy Contention-Resolution Protocols. *Proceedings of the Thirty-Seventh Annual IEEE Symposium on Foundations of Computer Science*, 380–389, 1996.

[3] M. Andrews, B. Awerbuch, A. Fernández, J. Kleinberg, F.T. Leighton, Z. Liu. Universal Stability Results and Performance Bounds for Greedy Contention-Resolution Protocols. *Journal version of [2]; to appear in JACM.*

[4] M. Andrews Instability of FIFO in Session Oriented Networks. *Proceedings of the Eleventh Annual ACM-SIAM Symposium on Discrete Algorithms*, Jan, 2000.

[5] M. Andrews and L. Zhang The Effects of Temporary Sessions on Network Performance. *Proceedings of the Eleventh Annual ACM-SIAM Symposium on Discrete Algorithms*, Jan, 2000.

[6] A. Borodin, J. Kleinberg, P. Raghavan, M. Sudan, and D. Williamson. Adversarial Queueing Theory. *Proceedings of the Twenty–Eighth Annual ACM Symposium on Theory of Computing*, 376-385, 1996.

[7] A. Borodin, J. Kleinberg, P. Raghavan, M. Sudan, and D. Williamson. Adversarial Queueing Theory. *Journal version of [6]; to appear in JACM.*

[8] D. Gamarnik. Stability of Adversarial Queues via Fluid Models. *Proceedings of the 39th Symposium on the Foundations of Computer Science*, 60-70, 1998.

[9] D. Gamarnik. Stability of Adaptive and Non-Adaptive packet Rounting Policies in Adversarial Queuing Networks. *Proceedings of the Thirty–First Annual ACM Symposium on Theory of Computing*, 206-214, 1999.

[10] A. Goel. Stability of Networks and Protocols in the Adversarial Queueing Model for Packet Routing. Stanford University Technical Note STAN-CS-TN-97-59, June 1997.

[11] R. Motwani and P. Raghavan *Randomized Algorithms.* Cambridge University Press, 1995.

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