STABILITY OF ADAPTIVE AND NON-ADAPTIVE PACKET ROUTING POLICIES IN ADVERSARIAL QUEUEING NETWORKS*

DAVID GAMARNIK†

Abstract. We investigate stability of packet routing policies in adversarial queueing networks. We provide a simple classification of networks which are stable under any greedy scheduling policy - network is stable if and only if the underlying undirected connected graph contains at most two edges. We also propose a simple and distributed policy which is stable in an arbitrary adversarial queueing network even for the critical value of the arrival rate $r = 1$. Finally, a simple and checkable network flow type load condition is formulated for adaptive adversarial queueing networks and a policy is proposed which achieves stability under this new load condition. This load condition is a relaxation of the integral network flow type condition considered previously in the literature.

Key words. congestion, scheduling, multicommodity flow.

AMS subject classifications. 68M20,60K25,90B10,90B25

1. Introduction.

The focus of this paper is stability of adversarial queueing systems. Such queueing models have attracted a lot of attention recently as a convenient tool for modeling packet injection and routing in a communication network. An adversarial assumption on the nature of the incoming traffic substitute more traditional stochastic arrivals assumption. Two types of queueing networks are usually considered: circuit switch and packet switch networks (also referred to as adaptive and non-adaptive packet routing networks). In the first model an adversary injects packets for processing, specifying the paths that the packets have to follow. The scheduler needs to decide which packets to process when several packets are competing for the same edge. Such models have been introduced by Borodin et al. in [6] and considered subsequently in several papers [3], [9], [11], [17].

In packet switch networks an adversary injects packets and only specifies their origin and destination. The scheduler is free to choose a path along which the packets are processed. This model has been considered only recently by Aiello et al. [1]. In both models the goal of the scheduler is to maintain the queue lengths of packets competing for the same edge as small as possible. While constructing schedules which guarantee minimal queue length is a computationally intractable problem (even static circuit switch and packet switch scheduling problems are NP-complete), researchers have focused on schedules which at least guarantee bounded queue lengths at all times - stability.

1.1. Stability of non-adaptive packet routing schedules.

A natural necessary condition for stability exists in circuit switch type networks. A positive integer w (called burstiness) exists, such that for any edge e and any time interval $[t_1, t_2)$, the total number of packets that are injected and contain edge e on their paths should not be bigger than $t_2 - t_1 + w$ (assuming each edge processes packets with unit speed). If the load condition is systematically violated, the queue lengths will build up no matter what schedule is used. The focus of the research has been understanding when is the load condition also sufficient for stability. It was proven that acyclic and unidirectional ring queueing networks are stable whenever the load condition is met and an

*A preliminary version of this paper appeared in Proceedings of 31st Ann. ACM Symposium on the Theory of Computing, 1999.

†T.J. Watson Research Center, IBM, Yorktown Heights, NY 10598 (gamarnik@watson.ibm.com).

arbitrary greedy schedule is implemented [3], [6], [16]. Meanwhile certain natural policies were shown to be unstable even if the load condition holds. Andrews [2], Andrews et al. [3] showed that First-In-First-Out (FIFO) and Nearest-To-Go (NTG) policies can be unstable. Instability of FIFO policies was also shown before by Bramson [7] for non-adversarial (stochastic) queueing networks. Borodin et al. [6] showed that NTG policy can be unstable in certain networks even if the arrival rate of the packets is bounded by an arbitrarily small constant r. On the other hand, Furthest-To-Go policy is stable in all networks (Andrews et al. [3]) whenever the maximal arrival rate r is strictly smaller than one. Goel [11] provided a complete algorithmic characterization of directed graphs which are stable for all greedy scheduling rules. Such characterization can be adapted to undirected graphs in which packets competing for an edge from opposite sides can simultaneously cross the edge. Our assumption throughout the paper will be that only one packet can cross any given edge at a time from either end and all the graphs are assumed to be undirected. Obtaining a complete algorithmic classification of stable networks for every policy seems to be an unachievable task. It is shown in Gamarnik [10] that checking stability for a class of *generalized priority* policies is an algorithmically undecidable problem. Whether undecidability holds for more common policies like FIFO or priority policies remains to be seen.

1.2. Stability of adaptive packet routing schedules. Stability of adaptive packet routing schedules in adversarial queueing networks has only recently been analyzed by Aiello et al. [1]. This model does not have a natural load condition for stability, as opposed to non-adaptive queueing models. There is no explicit load on edges implied by the incoming traffic, rather the load depends on the routing policy used. Aiello et al. thus introduced the following assumption. Suppose for some positive integer w and some positive real $r < 1$ an adversary can associate with each incoming packet a path in such a way that in every time interval $[t, t + w)$ every edge has been assigned to no more than rw paths. In other words the adversary should be able to reformulate the problem in non-adaptive sense, described above, without revealing the underlying assigned paths. It was shown in [1], that a stable distributed routing policy exists under the assumption above. Also the schedule does not assume the knowledge of the arrival rate r and interval w and the total number of packets in the network is bounded by $O(m^{5/2}n^{5/2}w/(1 - r))$, where m and n is the number of edges and nodes in the graph respectively.

1.3. Results. A number of questions remain outstanding, some of which are listed in Borodin et al. [6]. It is not clear whether ring type queueing network allowing traffic in both directions is stable under any greedy scheduling rule. More generally which networks are stable under all greedy scheduling rules (universally stable)? This question was resolved by Goel [11] for directed graph queueing networks, but remains outstanding for undirected graphs in which each edge can be crossed by only one packet at a time from either end. We provide in this paper a very simple answer to this problem. A connected undirected graph is universally stable if and only if it contains at most two edges. We establish this result by proving universal stability for a simple graph with two edges: G_1 and constructing unstable greedy schedules for graphs G_2, G_3, G_4 (see Figure 1). We will show specifically that such unstable greedy policy exists when the maximal arrival rate r satisfies $r^3 + r^4 > 1$ for the graph G_2 and satisfies $r^4 + r^5 > 1$ for the graphs G_3, G_4.

These schedules are very similar to the ones constructed by Goel [11] and Andrews et al. [3]. Clearly any connected undirected graph with more than two edges contains

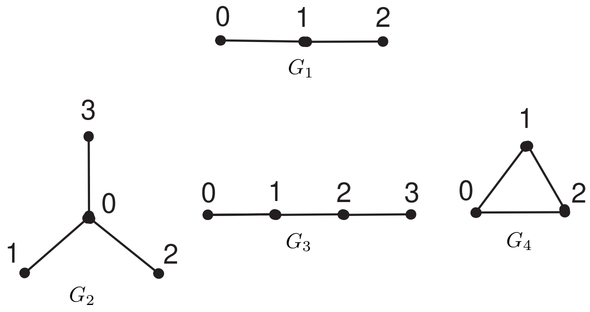

FIG. 1.1. *Graphs* G_1, G_2, G_3, G_4.

one of the graphs G_2, G_3, G_4 as a subgraph and as a result is unstable. In particular, ring type queueing network is either graph G_4, or contains G_3 as a subgraph, and, as a result, is not universally stable. We then propose a very simple distributed Nearest-To-Origin (NTO) priority policy, and prove that this policy is stable in all adversarial queueing networks even for the critical arrival rate $r = 1$. This answers positively the question posed in [6] on existence of a stable scheduling policy under a critical arrival rate $r = 1$.

For the case of adaptive packet routing models, we consider a more relaxed load condition than the one used in [1]. We assume that for the packets that arrive during any time interval of the length w, the corresponding static multicommodity flow problem has a feasible *fractional* solution with maximal congestion (to be defined) not bigger than rw, where $r < 1$. That is we relax the integrality requirement in the multicommodity flow type constraint on the arriving traffic, considered in [1]. We construct a simple discrete review type policy based on static packet routing problem and prove that this policy is stable under this relaxed load condition. The algorithm is based on an algorithm proposed in [5] for the static packet routing problem, which achieves asymptotic optimality as the network load diverges to infinity.

The advantage of the relaxed assumption above is clear - the load condition can be checked efficiently by solving a corresponding fractional multicommodity flow problem, whereas integral multicommodity flow problem is NP-complete. Specifically, if the maximal arrival rate for any pair of origin-destination nodes (i, j) is r_{ij}, then there exists a stable packet routing protocol if the fractional multicommodity flow problem with parameters r_{ij} has a feasible solution with maximal congestion smaller than one. Our scheduling rule, however, would not assume the knowledge of the rates r_{ij}. We will also show that if for any feasible solution the maximal congestion is bigger than one then no stable policy exists. We do not know whether a stable policy exists when the smallest maximal congestion is equal to one.

The disadvantage of our schedule compared to the one of Aiello et al. is that it occasionally needs information about the queue lengths in the entire network, and thus is not distributed. Our bounds on maximal queue lengths are also inferior to the ones in [1]. We conclude with some open questions and directions for further research.

2. Definitions and assumptions. A non-adaptive adversarial queueing network is given as a graph (V, E). An adversary injects packets for processing. Each arriving packet has a pre-specified path it has to follow. Once the end of the path is reached, the packet leaves the network. Each packet takes a unit time to cross a single edge, and only one packet in either direction can cross any given edge at a time. Packet processing occurs at integer time epochs $t = 0, 1, 2, \ldots$, although packet arrival can occur at an arbitrary real time. Packets that wait to cross some edge e accumulate into a queue at the vertex of e, until chosen to cross. We introduce some additional notations in order to formally describe the dynamics. Let $\mathcal{P}$ be the set of all simple paths in the network, (set of paths that can be requested by packets).

For each path $P \in \mathcal{P}$, let $\{e_0^P, e_1^P, \ldots, e_{k(P)}^P\}$ be the set of consecutive edges in P. Let $A_P(t_1, t_2)$ denote the total number of packets injected during the time interval $[t_1, t_2)$ that request path P. For each $P \in \mathcal{P}$ and $e \in P$, let $D_{(e,P)}(t_1, t_2)$ be the total number of packets, following path P, that crossed edge e during the time interval $[t_1, t_2)$. In particular, $D_{(e,P)}(t, t+1)$ takes value 0 or 1, for each $t = 0, 1, 2, \ldots$. The values of $D_{(e,P)}(t, t+1)$ depend on the rule by which packets competing for the same edge are prioritized - scheduling rule. Some examples of scheduling rules include First-In-First-Out, in which packets are prioritized according to their arrival time into edge e, Longest-In-System, in which packets are prioritized according to their arrival time into the network, Shortest-In-System, Furthest-To-Go and many other. Note, that whichever policy is used the following restriction applies. For any edge e and time t

$$\sum_{P: e \in P} D_{(e,P)}(t, t+1) \le 1.$$

In other words, at most one packet can cross an edge e during the time interval $[t, t+1)$. Finally, let $A_P(t) = A_P(0, t), D_{(e_i^P,P)}(t) = D_{(e_i^P,P)}(0, t)$, and let $Q_{(e,P)}(t)$ be the total number of packets following path P that are waiting to cross edge e at time t. The dynamics of the network is described as follows. For each $t = 0, 1, 2, \ldots$ and each path $P \in \mathcal{P}$

$$Q_{(e_0^P,P)}(t) = Q_{(e_0^P,P)}(0) + A_P(t) - D_{(e_0^P,P)}(t), \tag{2.1}$$

and

$$Q_{(e_i^P,P)}(t) = Q_{(e_i^P,P)}(0) + D_{(e_{i-1}^P,P)}(t) - D_{(e_i^P,P)}(t), \tag{2.2}$$

for all $i = 1, 2, \ldots, k(P)$. We let $Q(t) = \sum_{e,P} Q_{(e,P)}(t)$ denote the total number of packets in the network at time t.

The packets are injected into the system by an adversary in a restricted manner. There exists a positive real number r, called the arrival rate, and a positive integer w with the following property. For each edge e, the total number of packets, injected during any interval $[t_1, t_2)$, whose assigned paths contain e, is at most $r(t_2 - t_1) + w$. Formally, for each $e \in E$ and $t_1 < t_2$

$$\sum_{P: e \in P} A_P(t_1, t_2) \le r(t_2 - t_1) + w. \tag{2.3}$$

This is the load assumption considered in [3], [6], [9], [11] and is a generalization of the path-specific arrival rate assumption considered earlier by Cruz [8].

The goal of the stability analysis is to understand the conditions under which the total number of packets in the network stays bounded - network is stable. Specifically, we are interested, when is a particular scheduling policy stable and which networks are stable under an arbitrary greedy scheduling policy.

DEFINITION 1. *A scheduling policy in an adversarial queueing network* (V, E, r, w) *is defined to be greedy if whenever there is a positive amount of packets waiting to cross any given edge* e *at time* t*, at least one of these packets will cross* e *during interval* $[t, t+1)$*. Formally, for each* $e \in E$ *and* $t = 0, 1, 2, \ldots,$

$$\sum_{P: e \in P} Q_{(e,P)}(t) > 0$$

implies

$$\sum_{P: e \in P} D_{(e,P)}(t, t+1) = 1. \tag{2.4}$$

DEFINITION 2. *A scheduling policy in an adversarial queueing network* (V, E, r, w) *is defined to be stable if, under this policy, the total number of packets in the network stays bounded for all times. Namely,*

$$\sup_{t \in \Re_+} Q(t) < \infty.$$

A scheduling policy is defined to be universally stable, if it is stable in all graphs. A (directed or undirected) graph (V, E) *is defined to be universally stable if every greedy policy in it is stable for all* $r < 1$ *and all non-negative* w*.*

The necessary condition for stability is

$$r \leq 1. \tag{2.5}$$

If this condition is violated then an adversary can inject packets so that no scheduling rule will be able to keep the number of packets bounded.

An adaptive packet routing model is similar to the model above. An undirected graph (V, E) is given. An adversary injects packets, but now specifies their origin-destination pair $(i, j) \in V^2$ only. The goal of the scheduler is to select the paths for packets as well as to prioritize packets competing for the same edge. The total number of packets in the network again needs to be bounded. Immediately the question arises what is the analog of the condition (2.5). Aiello et al. [1] considered the following condition. A certain integer w and a real value $r < 1$ are fixed. It is assumed that the packets that arrived during any time interval $[t, t+w)$ can be associated with paths in the graph (V, E) in such a way that any edge e belongs to no more than rw paths. This condition can be reformulated using integral multicommodity flow problem as follows. For a given graph (V, E) and a set of positive integers $n_{ij}, i, j \in V$ consider the following integer programming problem (we represent edges as pairs of nodes $(k, l) \in E$)

$$\text{Minimize } C_{\max}$$

$$\text{Subject to}$$

$$\sum_{k:(i,k) \in E} x_{ik}^{ij} = n_{ij}, \qquad (i, j) \in V^2, \tag{2.6}$$

$$\sum_{k:(k,j) \in E} x_{kj}^{ij} = n_{ij}, \qquad (i, j) \in V^2, \tag{2.7}$$

$$\sum_{l:(l,k)\in E} x^{ij}_{lk} = \sum_{l:(k,l)\in E} x^{ij}_{kl}, (i,j)\in V^2,\ k\neq i,j, \tag{2.8}$$

$$C_{kl} = \sum_{(i,j)\in V^2} x^{ij}_{kl}, \qquad (k,l)\in E, \tag{2.9}$$

$$C_{kl} \le C_{\max}, \qquad (k,l)\in E, \tag{2.10}$$

$$x^{ij}_{kl},\ C_{kl}\ge 0, \tag{2.11}$$

$$x^{kl}_{ij}\in Z_+. \tag{2.12}$$

Here x^{ij}_{kl} represents the number of packets going from node i to node j that pass through the edge (k,l). Equations (2.6)-(2.8) represent the conservation of flow. C_{kl} represents the total amount of integral flow assigned to any edge $(k,l)\in E$, $C_{\max}$ represents the maximal amount of flow assigned to any edge $e\in E$.

It is not hard to prove that the load condition considered by Aiello et al. is equivalent to the following condition. Let $A_{ij}(t,t+w)$ denote the number of packets that arrived during the time interval $[t,t+w)$ and have an origin-destination pair (i,j). The condition is that for any time t the integral multicommodity flow problem above with input $n_{ij}=A_{ij}(t,t+w)$ has a solution $C_{\max}$ satisfying

$$C_{\max}\le rw. \tag{2.13}$$

DEFINITION 3. *An adversarial queueing network is said to be of the type* (r,w,IMF) *(IMF stands for integral multicommodity flow) if the condition (2.13) is satisfied for any time* t*, where* $C_{\max}$ *is the optimal value of the integral multicommodity flow problem (2.6)-(2.12) on the input* $n_{ij}=A_{ij}(t,t+w)$.

An algorithm was constructed in [1] which achieves stability under the load condition (r,w,IMF) for networks of type (r,w,IMF) with $r<1$. In this paper we consider queueing networks of the type (r,w,FMF) (FMF - fractional multicommodity flow) where the load condition above is still assumed to be satisfied, but the solution to the multicommodity flow problem above need not be integral (constraint (2.12) is removed).

DEFINITION 4. *An adversarial queueing network is said to be of the type* (r,w,FMF) *if the condition (2.13) is satisfied for any time* t*, where* $C_{\max}$ *is the optimal value to the fractional multicommodity flow problem (2.6)-(2.11) on the input* $n_{ij}=A_{ij}(t,t+w)$.

Clearly our load condition is weaker. Since it uses a linear programming formulation, the condition is also efficiently checkable. We construct in Section 4 a stable scheduling policy under this relaxed load condition whenever $r<1$.

3. Universally stable graphs and universally stable policies. In the first part of this section we focus on universal stability of undirected graphs. An exact characterization of directed stable graphs is given in [11]. Two directed graphs were constructed in which are not universally stable. It is then proven that a directed graph is universally stable if and only if it does not contain one of these two graphs as a minor (for a definition of a graph minor see [14]). This leads to an efficient algorithm for checking whether a given graph is stable.

In this section we show that for undirected graphs the classification is even simpler. A graph with one edge or a graph G_1 with two edges are the only connected undirected universally stable graphs. Note that the stability of unconnected graphs can be resolved by considering its connected components.

THEOREM 3.1. *A connected undirected graph is universally stable if and only if it has at most two edges.*

Remark : We conjecture that the graph with two edges is stable also for the critical arrival rate $r = 1$ but we do not have a proof.

Proof: We omit a trivial case of a graph with only one edge. We prove the stability of the graph G_1 by a simple reduction to a unidirectional ring network with two nodes. An adversarial queueing network (V, E, r, w) is called a unidirectional ring network, if it is of the form $V = \{v_1, v_2, \ldots, v_n\}, E = \{e_1, e_2, \ldots, e_n\}$, $e_k = (v_k, v_{k+1})$, $k = 1, 2, \ldots, n-1, e_n = (v_n, v_1)\}$, and if every path P contains edges in increasing order modulo n. Namely $P = \{e_j, e_{j+1}, \ldots, e_{j+k(P)}\}$ for some e_j, where the convention is to identify edge e_{n+i} with e_i. Thus, unidirectional ring is a circular form graph with all the packets moving in one direction. It was shown in [3] that the unidirectional ring is universally stable for all $r < 1$ and the total number of packets in the network at any moment is not bigger than $n^2 w/(1-r)$ (assuming initially there are no packets in the network). We now show that from the stability point of view our graph G_1 is equivalent to the unidirectional ring with two nodes $V = \{v_1, v_2\}$ and two edges $E = \{e_{\text{lower}}, e_{\text{upper}}$ connecting nodes v_1 and v_2. Any packet in G_1 going from 0 to 1 or from 1 to 0 we associate with a packet going along the edge e_{lower} in the directions $v_1 \to v_2$. Any packet in G_1 going from 1 to 2 or from 2 to 1 we associate with a packet going along the edge e_{upper} in the directions $v_2 \to v_1$. We associate packets going along nodes $0, 1, 2$ or $2, 1, 0$ similarly. It is easy to see that this correspondence makes two systems equivalent. In particular graph G_1 is stable and, if the initial number of packets is zero, then the maximal number of packets at any time is not more than $4w/(1-r)$.

We now prove the second part of the theorem. We will show that in any connected graph with more than two edges there exists an unstable greedy scheduling policy whenever $r > .86$. Clearly it suffices to prove the existence of such policies only for graphs G_2,G_3, G_4 on Figure 1.

Consider the graph G_2 first. The arrival pattern and the scheduling policy are described in several stages. Suppose initially there are c packets waiting to cross the edge $(1, 0)$, where c is a sufficiently large number. During the time interval $[0, c)$ we process these c packets and generate rc packets requesting the path $1, 0, 2$. These packets do not move until time c. During the time interval $[c, c+rc)$ we process these rc packets and generate r^2c packets requesting path $2, 0, 3$ and r^2c packets requesting $0, 1$. These packets also do not move until the time $c + rc$. During the next time interval of the length r^2c we generate r^3c packets requesting $1, 0, 3$ and r^3c packets requesting $2, 0$. The latter packets are processed before the previously generated $2, 0, 3$ packets. As a result at time $c + rc + r^2c$ we obtain r^3c packets requesting $1, 0, 3$ and r^3c packets requesting $2, 0, 3$ (the latter generated in the previous round). During the next r^3c time units we process entirely packets on the path $1, 0, 3$, process packets on the path $2, 0, 3$ along the edge $(2, 0)$ and generate r^4c packets requesting path $0, 3$. In the end we obtain $r^3c + r^4c$ packets requesting edge $(0, 3)$. If $r^3 + r^4 > 1$ (which holds for $r > .82$), we end up with more than c packets requesting the path $0, 3$. Repeating the schedule for $c' = (r^3 + r^4)c$ packets starting from the edge $(0, 3)$ we obtain an unstable schedule. This completes the proof for the graph G_2.

The proof for the graph G_3 is very similar. Suppose initially we have c packets requesting path $2, 1$. Process these packets and generate rc packets requesting $2, 0$ during the time interval $[0, c)$. Process the new packets and generate r^2c packets requesting paths $1, 2, 3$ and $0, 1$. Process these packets and generate r^3c packets requesting $3, 2, 1$ and $0, 1$. During the next r^3c time units process all $0, 1$ packets and generate r^4c packets requesting $0, 1, 2$. Also during this time interval process generate

r^4c packets requesting $3,2$, give them priority over previously generated $3,2,1$ packets, and the remaining time process these $3,2,1$ packets. We obtain in the end r^4c packets requesting $0,1,2$ and r^4c packets requesting $3,2,1$. Note that all these packets require edge $(1,2)$. Process all the $0,1,2$ packets, process $3,2,1$ packets through their first edge $(3,2)$ and generate r^5c packets requesting $2,1$. As a result we obtain r^4c+r^5c packets requesting edge $2,1$. If $r^4+r^5>1$, that is $r>.86$ then we end up with more than c packets requesting edge $2,1$. It follows that the scheduling rule is unstable. The construction of unstable policy in the graph G_4 is identical to the one of G_3 where we identify node v_3 of G_3 with v_0 of G_4. This completes the proof of the theorem . □

In the remainder of this section we address the question of universal stability of specific policies. We propose a simple Nearest-To-Origin policy and prove that it is stable in all graphs even for the critical arrival rate $r=1$. The Nearest-To-Origin policy gives priority to packets which have crossed the smallest amount of edges. Namely if two packets following paths $P,P'\in\mathcal{P}$ compete for the same edge $e=e_i^P=e_j^{P'}$ and $i<j$ then packet following P should be processed first. If $i=j$ then the packets are prioritized arbitrarily.

THEOREM 3.2. *NTO policy is stable in any network for $r=1$.*

Proof. Let $Q_e(0)$ denote the total initial number of packets waiting to cross an edge e. Let also $Q_k(t)$ denote the total number of packets at time t which are within exactly k steps from the origin. That is

$$Q_k(t)=\sum_{P\in\mathcal{P}}Q_{(e_k^P,P)}(t).$$

Let us call these packets layer k packets. We will show by induction by k that $Q_k(t)$ is bounded by a constant for all t.

Base step $k=0$. Fix an edge e and time t. Let $t_0\in[0,t]$ be the largest time at which no packets of layer 0 (packets that have not crossed any edge yet) were waiting at e. If no such time exist, set $t_0=0$. The total number of layer 0 packets in e at time t_0 is then at most $Q_e(0)$. During the time interval $[t_0,t)$ at most $r(t-t_0)+w=t-t_0+w$ layer 0 (external) packets have arrived, that want to cross e. Also, by the choice of t_0, edge e was processing packets constantly during the time interval $[t_0,t)$. Since NTO policy is used, layer 0 packets have priority over all other packets. It follows that total number of layer 0 packets at the edge e at time t satisfies

$$\sum_{P:e_0^P=e}Q_{(e_0^P,P)}(t)\le Q_e(0)+t-t_0+w-(t-t_0)=Q_e(0)+w.$$

As a result $Q_0(t)\le\sum_e Q_e(0)+w|E|$, for all t. We denote $\sum_e Q_e(0)+w|E|$ by B_0.

Induction step. Suppose, for some constants $B_j, j=0,1,\ldots,k-1$, $Q_j(t)\le B_j$, for all $j\le k-1$ and for all times t. We will show that for some constant B_k, $Q_k(t)\le B_k$ for all t. Again fix an edge e and an arbitrary time t. Let again $t_0\le t$ denote the largest time that there were no layer k packets waiting to cross e. Then edge e was always processing packets during the time interval $[t_0,t)$ (packets in layer k or lower). The layer k packets that wait to cross e at time t then are composed only of packets which were in layers $j\le k-1$ at time t_0 or packets that arrived externally during the time interval $[t_0,t)$ and have edge e as their k-th edge on the requested path. The first group of packets has a size bounded by $\sum_{j=0}^{k-1}B_j$, by the induction assumption. The second is bounded by $\sum_{P:e_k^P=e}A_P(t_0,t)$. We now estimate the number of layer k packets that crossed e during the time interval $[t_0,t)$. Since NTO policy is used these packets were not processed only when there were packets in layers up to $k-1$ that wanted to cross e. The number of such packets is bounded by $\sum_{j=0}^{k-1} B_j$ - total possible number of packets in layers up to $k-1$ at time t_0, plus $\sum_{j=0}^{k-1}\sum_{P:e_j^P=e} A_P(t_0,t)$, which is number of new packets that arrived in $[t_0,t)$ and cross e within $k-1$ steps. We conclude that at least

$$\max\{0, t-t_0-\sum_{j=0}^{k-1} B_j - \sum_{j=0}^{k-1}\sum_{P:e_j^P=e} A_P(t_0,t)\}$$

packets of layer k crossed e during the time interval $[t_0,t)$. We obtain

$$\sum_{P:e_k^P=e} Q_k(t) \le \sum_{j=0}^{k-1} B_j + \sum_{P:e_k^P=e} A_P(t_0,t) - (t-t_0-\sum_{j=0}^{k-1} B_j - \sum_{j=0}^{k-1}\sum_{P:e_j^P=e} A_P(t_0,t)) \le$$

$$2\sum_{j=0}^{k-1} B_j + w,$$

where the last inequality follows from (2.3) and $r=1$. Thus the total number of layer k packets is bounded by $B_k = 2|E|(\sum_{j=0}^{k-1} B_j) + w|E|$. This completes the induction step. □

After this paper was written it was pointed to the author by Kleinberg [12] that a similar analysis shows stability of FTG policy (which gives priority to packets closest to their destination) when $r=1$. During the course of the proof we obtained the following bound on the total number of packets in the network at time t

$$|Q(t)| \le (2|E|)^{p_{\max}+1}(B_0+w)$$

where B_0 is the initial number of packets in the network and $p_{\max}$ is the maximal length $|P|$ of paths $P \in \mathcal{P}$. Unfortunately, the bound is exponential in the network parameters. It is shown in [4] that both NTO and FTG lead to exponentially large queue sizes in certain networks. It is also shown that no bound better than $2^{\sqrt{\max|V|,|E|}}$ is possible for a whole class of distributed deterministic policies including NTG, FTG.

Note that, unlike $r<1$ case, stability under $r=1$ condition does not necessarily imply that all the packets are delivered within a finite time. Indeed, consider the graph G_1 operating under NTG policy. Assume that initially there are several packets requesting path $0,1,2$. Assume that at each integer moment $t=1,2,\dots$ an adversary injects a packet following $1,2$. These packets have priority over the initial packets and block them from processing forever. Thus the delivery time for the initial packets is infinity. Whether there exists a policy with bounded delivery time for every packet when $r=1$, is an open question.

4. Stable scheduling policies in adaptive adversarial queueing networks. In this section we focus on adaptive packet routing policies in adversarial queueing networks. We have an undirected graph (V,E) and parameters $r,w>0$. An adversary generates packets and specifies their origin-destination only. The network flow load assumption (r,w,FMF), described in Section 2, is assumed to be satisfied by an adversary traffic. The goal is to construct a policy which is stable under the

(r, w, FMF) assumption, when $r < 1$ and show that no stable policy exist against any adversary traffic when $r > 1$. For the case $r < 1$ we consider a corresponding static packet routing problem on a fixed input n_{ij} which is formulated as follows. Suppose for each $i, j \in V$ we are given n_{ij} packets which are required to go from node i to node j via some path selected by the scheduler. Each edge can process only one packet at one time unit. The objective is to find a routing schedule which would minimize the time until all the packets reach their destination (makespan time). This static version of the packet routing problem with the makespan objective has been considered before by Srinivasan and Teo [15] and Bertsimas and Gamarnik [5]. We use here an asymptotically optimal scheduling algorithm developed in [5] (Packet Routing Synchronization Algorithm or PRSA) for this static packet routing problem. The following result was proven in [5].

THEOREM 4.1. *Let n_{ij} denote the number of packets that are present in the network at time 0 and have nodes $i, j \in V$ as their origin-destination pair. Let $C_{\max}$ be the optimal solution to the (fractional) multicommodity problem with input $n_{ij}, i, j \in V$. Then there exists a packet routing scheduling algorithm which brings all the packets to their destination (has makespan time) in not more than*

$$C_{\max} + O(|V|^3|E|\sqrt{C_{\max}}) \tag{4.1}$$

time units. Moreover the algorithm is such that after time $T = C_{\max} + |V|\sqrt{C_{\max}}$ not more than $2|V||E|\sqrt{C_{\max}} + |V|^2|E|$ packets are still present in the network.

Since $C_{\max}$ is a lower bound on any feasible makespan time, the schedule is asymptotically optimal when the total initial number of packets $\sum_{ij} n_{ij}$ diverges to infinity. We now use this scheduling algorithm to construct a stable policy in a network (V, E) which satisfies the load condition (r, w, FMF). The routing policy is *Discrete Review* type and is described as follows. We assume that $T_0 = 0$. The number of packets of type (i, j) at time t is denoted by $Q_{ij}(t)$.

Discrete Review Algorithm: for $k = 0, 1, 2, \ldots,$ let $C^k_{\max}$ denote the optimal value to the multicommodity flow problem with the input $n_{ij} = Q_{ij}(T_k)$. Set $T_{k+1} = T_k + C^k_{\max} + |V|\sqrt{C^k_{\max}}$. Implement PRSA at time T_k to the input $n_{ij} = Q_{ij}(T_k)$ for the first $C^k_{\max} + |V|\sqrt{C^k_{\max}}$ time units, ignoring packets arriving after time T_k.

In words, Discrete Review algorithm looks at times $T_k, k = 0, 1, 2, \ldots$ at the entire network and solves the corresponding static packet routing problem, ignoring packets that arrive after T_k. Intuitively, since the PRSA algorithm has makespan time close to the maximal congestion when the loads are high, then by (r, w, FMF) assumption, $C^{k+1}_{\max}/C^k_{\max} \approx r < 1$, and the *Discrete Review* policy is stable. The following result is obtained.

THEOREM 4.2. *Suppose an adversarial queueing network (V, E) satisfies (r, w, FMF) load condition. If $r < 1$ then Discrete Review routing schedule is stable. Moreover*

$$\limsup_{t\to\infty} \sum_{ij} Q_{ij}(t) = O\Big(\frac{|V|^4|E|^3 + w^2|E|}{(1-r)^2}\Big) \tag{4.2}$$

If $r > 1$ then no stable routing policy exists.

Proof: First we prove the stability of the Discrete Review routing schedule. We show that for any k

$$C^{k+1}_{\max} \le r(C^k_{\max} + |V|\sqrt{C^k_{\max}}) + w + |V|^2|E|\sqrt{C^k_{\max}}. \tag{4.3}$$

In fact the total number of type (i,j) packets in the network at time T_{k+1} consists of two types of packets. We have $A_{ij}(T_k,T_{k+1})$ packets that arrived during the time interval $[T_k,T_{k+1})$, and packets that arrived before time T_k and still have not been processed. Denote the latter packets by $\hat{Q}_{ij}(T_k)$. From Theorem 4.1, since $T_{k+1}-T_k = C^k_{\max}+|V|\sqrt{C^k_{\max}}$, we have

$$\sum_{ij}\hat{Q}_{ij}(T_k)\le 2|V||E|\sqrt{C^k_{\max}}+|V|^2|E|\le |V|^2|E|\sqrt{C^k_{\max}} \tag{4.4}$$

(as long as $|V|,C_{\max}\ge 4$, which we assume to hold). Since our network is (r,w,FMF) type, the optimal value of the multicommodity flow problem on the input $n_{ij} = A_{ij}(T_k,T_{k+1})$ is at most

$$r(\lceil\frac{T_{k+1}-T_k}{w}\rceil w)\le r(T_{k+1}-T_k)+w.$$

The optimal value of the multicommodity flow problem on the input $n_{ij}=\hat{Q}_{ij}(T_k)$ is upper bounded by $\sum_{ij}\hat{Q}_{ij}(T_k)$. Combining, the optimal value $C^{k+1}_{\max}$ of the multicommodity flow problem on the input $n_{ij}=Q_{ij}(T_{k+1})=A_{ij}(T_k,T_{k+1})+\hat{Q}_{ij}(T_k)$ satisfies

$$C^{k+1}_{\max}\le r(T_{k+1}-T_k)+w+|V|^2|E|\sqrt{C^k_{\max}}=r(C^k_{\max}+|V|\sqrt{C^k_{\max}})+w+|V|^2|E|\sqrt{C^k_{\max}}.$$

This proves (4.3). From (4.3) we obtain

$$\limsup_{k\to\infty}C^k_{\max}\le r\limsup_{k\to\infty}C^k_{\max}+(r|V|+|V|^2|E|)\limsup_{k\to\infty}\sqrt{C^k_{\max}}+w\le$$

$$r\limsup_{k\to\infty}C^k_{\max}+(r+|V|^2|E|+w)\sqrt{\limsup_{k\to\infty}C^k_{\max}}$$

or

$$\limsup_{k\to\infty}C^k_{\max}\le\frac{(r+|V|^2|E|+w)^2}{(1-r)^2}. \tag{4.5}$$

We finally argue that for any $t=0,1,2,\ldots$ the total number of packets in the network at time t, for large t, is at most $2|E|C^{k_t}_{\max}$, where k_t satisfies $T_{k_t}\le t<T_{k_t+1}$. We split all the packets in the network at time t into two groups: those that arrived before time T_{k_t} and those that arrived after time T_{k_t}. The first group has a size at most $\sum_{ij}Q_{ij}(T_{k_t})$ which is at most $|E|C^{k_t}_{\max}$, since $C^{k_t}_{\max}$ is a feasible solution to the multicommodity problem on the input $n_{ij}=Q_{ij}(T_{k_t})$. The second group has a size at most $\sum_{ij}Q_{ij}(T_{k_t+1})$ since none of these packets are processed before time T_{k_t+1}. Therefore, the second group has a size at most $|E|C^{k_t+1}_{\max}$. We apply (4.5) and conclude the proof of the theorem.

We now prove that if $r>1$ then no stable routing policy exist. For that we need to exhibit a certain adversarial arrival pattern with arrival rate r for which stability cannot be achieved. Select $\delta>0$ a small positive value and construct a set of rates r_{ij} such that the optimal value r_0 of the multicommodity flow problem on the input $n_{ij}=r_{ij}$ satisfies $r_0=r/1+\delta)$. This can be achieved as follows: select values r'_{ij} very small arbitrarily and increase them proportionally by t, $r_{ij}=r'_{ij}t$ until the objective value becomes r_0. We now construct an arrival pattern. For each pair (i, j) inject a type (i, j) packet every $1/r_{ij}$ time units. Select an integer $w > 0$ so that $r_{ij}w > 1/\delta$ for every r_{ij}. Then for every time t,

$$A_{ij}(t, t+w) \leq \lceil \frac{w}{1/r_{ij}} \rceil \leq r_{ij}w + 1 = r_{ij}w + \frac{1}{r_{ij}w} r_{ij}w \leq r_{ij}(1+\delta)w.$$

Since the optimal value of the multicommodity flow problem on the input $r_{ij}(1+\delta)$ is $r_0(1+\delta) = r$, then the network is of the (r, w, FMF) type. In the proof of the Corollary 1 below we show that for this arrival pattern if the objective value $r_0 = r/(1+\delta) > 1$ then no stable policy can exist. By choosing δ sufficiently small, we obtain the result. □

The Discrete Review algorithm is a one way of turning a schedule for a static packet routing problem into a schedule for a dynamic packet routing problem. There are schedules other then PRSA, which achieve certain degree of closeness to the optimal makespan time. For example, a routing schedule was constructed in [15] with makespan time $cC_{\max}$, where $C_{\max}$ is the optimal maximal congestion (solution to the multicommodity flow problem), and c is some (large) constant independent of the data of the problem. Note that this schedule, if implemented in *discrete review* manner does not necessarily lead to a stable policy if $r > 1/c$. The asymptotic optimality of the routing schedule (which is achieved by PRSA) is essential for stability.

The result above can be used to decide what are the necessary and sufficient conditions for stability in adversarial queueing networks, when the arrival rate for each pair of origin-destination is bounded by a constant.

COROLLARY 1. *Given an adversarial queueing network (V, E), suppose there exist constants $r_{ij}, i, j \in V$, and a positive integer $w > 0$, such that the total number of packets injected during any interval $[t_1, t_2)$ with origin-destination pair (i, j) is not bigger than $r_{ij}(t_2 - t_1) + w$. If the multicommodity flow problem on the input $r_{ij}, i, j \in V$ has an optimal solution satisfying $C_{\max} < 1$ then there exists a stable packet routing schedule. If the optimal solution satisfies $C_{\max} > 1$, then a stable schedule cannot exist.*

Proof: Suppose $C_{\max} < 1$ for the multicommodity flow problem on the input $n_{ij} = r_{ij}$. Select a positive integer

$$W > \frac{2|V|^2 w}{1 - C_{\max}}.$$

By assumption, for any time t

$$A_{ij}(t, t+W) \leq r_{ij}W + w$$

Also by assumption, the optimal value of the multicommodity flow problem on the input $n_{ij} = r_{ij}W$ is at most $C_{\max}W$. The optimal value of the multicommodity flow problem on the input $n_{ij} = w$ is trivially at most $|V|^2 w$. Therefore, the optimal value of the multicommodity flow problem on the input $n_{ij} = r_{ij}W + w$ is at most

$$C_{\max}W + |V|^2 w = C_{\max}W + \frac{|V|^2 w}{W} W \leq C_{\max}W + \frac{1 - C_{\max}}{2} W = \frac{1 + C_{\max}}{2} W.$$

We set $r = (1 + C_{\max})/2 < 1$. Then our queueing network is of the type (r, W, FMF). We apply Theorem 4.2 and complete the proof of the first part.

Suppose now the optimal solution to the multicommodity flow problem on the input $n_{ij} = r_{ij}$ satisfies $C_{\max} > 1$. Let an adversary inject type (i,j) packet every $1/r_{ij}$ time unit. Then for every type (i,j)

$$A_{ij}(t_1,t_2) \leq \lceil \frac{t_2 - t_1}{1/r_{ij}} \rceil \leq r_{ij}(t_2 - t_1) + 1 \leq r_{ij}(t_2 - t_1) + w.$$

So this arrival pattern satisfies the conditions of the theorem. Suppose, for the purposes of contradiction, there exists a stable routing policy. For every type (i,j) and every edge $(k,l) \in E$ let $D^{ij}_{kl}(t_1,t_2)$ denote the number of type (i,j) packets that crossed the edge (k,l) during the time interval $[t_1,t_2)$, when this stable policy is implemented. Let also $Q^{ij}_k(t)$ denote the number of type (i,j) packets that are queued at the node k at time t. Then

$$Q^{ij}_i(t) = Q^{ij}_i(0) + A_{ij}(0,t) - \sum_{k:(i,k)\in E} D^{ij}_{ik}(0,t). \tag{4.6}$$

Also for each $k \neq i,j$

$$Q^{ij}_k(t) = \sum_{l:(l,k)\in E} D^{ij}_{lk}(0,t) - \sum_{l:(k,l)\in E} D^{ij}_{kl}(0,t). \tag{4.7}$$

By assumption, $Q^{ij}_j(t) = 0$ and $D^{ij}_{jl}(0,t) = 0$ for all l such that $(j,l) \in E$. Since the policy implemented is stable, then there exists $B > 0$ such that for all k and all times $t, Q^{ij}_k(t) \leq B$. In particular,

$$Q^{ij}_k(t)/t \to 0 \tag{4.8}$$

when $t \to \infty$. Note that for each $(k,l) \in E$

$$\sum_{i,j} D^{ij}_{kl}(0,t)/t \leq 1 \tag{4.9}$$

since each edge processes at most one packet at a time. Therefore there exists a sequence $t_1 < t_2 < \ldots < t_s < \ldots$ along which all the limits

$$x^{ij}_{kl} \equiv \lim_{s\to\infty} \frac{D^{ij}_{kl}(0,t_s)}{t_s}$$

exist. Note also that

$$\lim_{t\to\infty} \frac{A_{ij}(0,t)}{t} = r_{ij}.$$

From (4.6)-(4.8) it follows that $x^{ij}_{kl}, i,j \in V, (k,l) \in E$ is a feasible solution to the multicommodity flow problem on the input $n_{ij} = r_{ij}$. But from (4.9) it follows that for each edge $(k,l) \in E$

$$\sum_{i,j} x^{ij}_{kl} \leq 1.$$

In particular, the maximal congestion $C_{\max}$ corresponding to this feasible solution satisfies $C_{\max} \leq 1$. This contradicts the assumption. □

It is not clear whether a stable packet routing schedule exists for the critical case $C_{\max} = 1$, analogous to the NTO policy proposed in Section 3. The existence of such policy might depend on the integrality of a feasible solution to the multicommodity flow problem (2.6)-(2.12).

Note that the bound given by Theorem 4.2 is weaker than the bound

$$O\Big(\frac{|V|^{5/2}|E|^{5/2}w}{1-r}\Big) \tag{4.10}$$

obtained in [1] for networks of type (r, w, IMF). Also, the schedule constructed in [1] is distributed whereas our schedule needs information about the entire network at times T_k. This raises the question whether the schedule in [1] is applicable for the network of type (r, w, FMF). This turns out to be possible but at the cost of a somewhat inferior performance. Consider[1] an optimal fractional solution (maximal congestion) $C_{\max}$ to the multicommodity problem on the input $n_{ij} = A_{ij}(t, t+w)$. Using Raghavan and Thomson randomized algorithm (see [13]) one can construct an integral solution to the multicommodity flow problem with expected maximal congestion satisfying $C^{\text{int}}_{\max} \le C_{\max} + C_{\max} \cdot O(\sqrt{\frac{\log|E|}{C_{\max}}})$. Suppose now

$$w > O\Big(\frac{\log|E|r}{(1-r)^2}\Big). \tag{4.11}$$

Then $C_{\max} \le rw$ implies $C^{\text{int}}_{\max} \le rw + \sqrt{rw}O(\sqrt{\log|E|}) < w$. Thus, if (4.11) holds, then the network is of the type (r', w, IMF) for $r' \equiv C^{\text{int}}_{\max}/w$. Unfortunately, the bound (4.11) combined with (4.10) implies the bound

$$O\Big(\frac{|V|^{5/2}|E|^{5/2}\log|E|r}{(1-r)^2(1-r')}\Big)$$

which is at best (using $r \le r'$) $O(\frac{1}{(1-r)^3})$, when $r \approx 1$. This is inferior to our bound $O(\frac{1}{(1-r)^2})$ in Theorem 4.2 when $r \approx 1$. (Recall that our bound does not require (4.11)).

Observe, that we used the second part of Theorem 4.1 in the construction of the *Discrete Review* algorithm. We could also use the first part (route entirely the packets present in the system at time T_k) at the price of somewhat larger upper bounds, although this scheme has the following advantage. In this form of implementation the path of each packet is predetermined at time T_k for packets present in their origination node at time T_k. Thus, the path information can be recorded in the header of the packet and no path recalculation is needed while the packet is on its route.

5. Conclusion. We have provided in this paper a simple classification of undirected graphs which are universally stable - a connected undirected graph is universally stable if and only if it has at most two edges. We have also proved that a simple distributed policy NTO achieves stability in all the graphs under the critical arrival rate $r = 1$. A multicommodity flow type load condition was formulated for adaptive adversarial queueing networks and a stable policy was constructed whenever this load condition is met.

A number of interesting questions remain outstanding. It is not clear which graphs are universally stable for a given value of $r < 1$. Given that a network could be

[1]The author wishes to thank Ashish Goel for pointing out this argument.

unstable for an arbitrarily small r (see [6]) such classification could be quite nontrivial. Deciding stability of specific policies is not well understood in general and might become an impossible problem in light of undecidability results in [10].

Acknowledgments. The author wishes to thank Ashish Goel, Matthew Andrews and several anonymous referees for many fruitful suggestions and corrections.

REFERENCES

[1] W. Aiello, E. Kushilevitz, R. Ostrovsky, and A. Rosen. Adaptive packet routing for bursty adversarial traffic. *Proc. 30th Ann. ACM Symposium on the Theory of Computing*, 1998.
[2] M. Andrews. Instability of FIFO in session-oriented networks. *Proc. 11th ACM-SIAM Symposium on Discrete Algorithms*, 2000.
[3] M. Andrews, B. Awerbuch, A. Fernandez, Jon Kleinberg, T. Leighton, and Z. Liu. Universal stability results for greedy contention-resolution protocols. *Proc. 27th IEEE Conf. on Foundations of Computer Science*, pages 380–389, 1996.
[4] M. Andrews and L. Zhang. The effects of temporary sessions on network performance. *Proc. 11th ACM-SIAM Symposium on Discrete Algorithms*, 2000.
[5] D. Bertsimas and D. Gamarnik. Asymptotically optimal algorithm for job shop scheduling and packet routing. *Journal of Algorithms*, 33(2):296–318, 1999.
[6] A. Borodin, J. Kleinberg, P. Raghavan, M. Sudan, and D. Williamson. Adversarial queueing theory. *Journal of ACM*, 48:13–38, 2001.
[7] M. Bramson. Instability of FIFO queueing networks. *Ann. Appl. Probab.*, 2:414–431, 1994.
[8] R. Cruz. A calculus for network delay, part II: network analysis. *IEEE Trans. Information Theory*, 37:132–141, 1991.
[9] D. Gamarnik. Using fluid models to prove stability of adversarial queueing networks. *IEEE Transactions on Automatic Control*, 4:741–747, 2000.
[10] D. Gamarnik. On deciding stability of constrained homogeneous random walks and queueing systems. *Mathematics of Operations Research*, 27(2):272–293, 2002.
[11] A. Goel. Stability of networks and protocols in the adversarial queueing model for packet routing. *Proc. 10th ACM-SIAM Symposium on Discrete Algorithms*, 1999.
[12] J. Kleinberg. Personal communication. 1998.
[13] R. Motwani and P. Raghavan. *Randomized Algorithms*. Cambridge University Press, 1995.
[14] N. Robertson and P.D. Seymour. *Recent results on graph minors*. Cambridge University Press, 1985.
[15] A. Srinivasan and C.P. Teo. A constant-factor approximation algorithm for packet routing, and balancing local vs. global criteria. *Proc. 29th Ann. ACM Symposium on the Theory of Computing*, 1997.
[16] L. Tassiulas and L. Georgiadis. Any work-conserving policy stabilizes the ring with spatial reuse. *IEEE/ACM Trans. on Networking*, 4(2):205–208, 1996.
[17] P. Tsaparas. *Stability in Adversarial Queueing Theory*. M.Sc. Thesis, University of Toronto, 1997.