Instability of FIFO at Arbitrarily Low Rates in the Adversarial Queueing Model

Rajat Bhattacharjee *
University of Southern California

Ashish Goel †
Stanford University

Abstract

We study the stability of the commonly used packet forwarding protocol, FIFO (First In First Out), in the adversarial queueing model. We prove that FIFO can become unstable, i.e., lead to unbounded buffer-occupancies and queueing delays, at arbitrarily low injection rates. In order to demonstrate instability at rate r, we use a network of size polynomial in $1/r$.

1 Introduction

In traditional queueing theory, the source which generates network traffic is typically assumed to be stochastic. However, the growing complexity of network traffic makes it increasingly unrealistic to model traffic as, say, a Poisson stream. Adversarial Queueing Theory is a robust and elegant framework developed by Borodin *et al.* [8] to address this problem. In this model, packets are injected into the network by an adversary rather than by a stochastic process. The route of each packet is given along with the packet itself. Each edge in the network can forward at most one packet in one time step. If there are multiple packets waiting to cross the same edge, then we need a contention resolution protocol to decide which packet goes across and which packets wait in the queue. The adversary is limited in the following way: over any window of T consecutive time steps, the adversary can inject at most $w + rT$ packets that need to traverse any edge in the network. The parameter r is called the injection rate and must be less than 1. The parameter w is called the burst-size. Such an adversary is called a (w, r)-adversary. Once injected, packets follow their routes one edge at a time till they reach their destination.

Intuitively, the adversary is not allowed to introduce more traffic on an average than r times the capacity of any edge. Thus, there can be no identifiable "hot-spots" in the system. This model finds the fine middle ground between stochastic arrivals on the one hand (as in traditional queueing theory) where packet arrival is too predictable, and completely unconstrained adversaries on the other (as in competitive analysis), where the adversary is allowed to overload the system.

A packet forwarding protocol is said to be *stable* against a given adversary and for a given network if the maximum queue size, as well as the maximum delay experienced by a packet, remain bounded. A packet forwarding protocol is said to be stable at rate R (or, R-stable) if it is stable against all (w, R) adversaries, and for all networks. It is said to be *universally stable* if it is r-stable for all $r < 1$. Studying the stability of protocols was the main motivation behind the adversarial queueing theory model. In a seminal paper, Andrews *et al.* [5] showed that several natural protocols are universally stable, but surprisingly, FIFO is not. This is an important observation since FIFO is by far the most widely used scheduling protocol. It also leaves the following question open:

> Is FIFO R-stable at some rate $R > 0$, or is FIFO unstable at arbitrarily low rates?

This is an important question, given the prominence of FIFO as the packet forwarding protocol for the Internet and other networks, and has been the subject of much study over the last few years [11, 9, 10, 5, 4, 21, 23, 26]. For several natural protocols other than FIFO, the corresponding question about the stability threshold in the adversarial model has already been answered [29]. The problem for FIFO is particularly intriguing since there is some intuitive evidence for each direction, due to Bramson [10, 11], arising from two somewhat unrelated models (more details are given in the related work section). The original proof of Andrews *et al.* [5] showed instability of FIFO at rate 0.85. Diaz *et al.* [21] improved the threshold of instability for FIFO to 0.83. This was further improved by Koukopoulos *et al.* [23] to 0.749 and recently, Lotker *et al.* [26] further improved it to 0.5.

*Department of Computer Science, University of Southern California, Los Angeles CA 90089-0781. Email: rajatbha@cs.usc.edu.

†Department of Management Science and Engineering and (by courtesy) Computer Science, Stanford University, Terman 311, Stanford CA 94305. Email: ashishg@stanford.edu. Research supported by NSF CAREER Award 0133968.

In this paper, we will assume that packet paths are required to be simple.

1.1 Our result

We prove that FIFO can become unstable at arbitrarily low rates in the adversarial model. This was one of the major remaining open problems in the field of adversarial queueing theory. In particular, for an arbitrarily low injection rate r, we construct a network and determine adversarial injections so that FIFO becomes unstable. The size of our network is polynomial in $1/r$. This is quite strong since it even excludes the possibility that FIFO might be stable at rate $O(1/\log^c m)$ where m is the size of the network and c is a constant[1].

The main idea is the construction of a gadget which, assuming certain initial conditions, allows only a small fraction of packets to pass through it for a long duration. In particular, the fraction of packets which escape is bounded by $k/(1+r)^k$ where k is a parameter of the gadget and can be increased arbitrarily. The network is constructed using this gadget. The adversary works in phases. At the beginning of a phase, we assume that there are some packets waiting to pass through a column of gadgets. Using each gadget in the column, more packets are generated which want to ultimately traverse through a second column. Additional copies of the gadget mentioned above are used to delay and synchronize these new packets so that, at the end of the phase, there are more packets waiting to traverse the second column than were waiting at the first column at the beginning of the phase. Applying this inductively leads to instability.

While the idea of concatenating parameterized gadgets to prove instability has been used before (most recently by Lotker *et al.* [26], for example), our gadget and network are quite different from those used in earlier works [10, 5, 29, 26] to prove instability results. A review of related work is presented below.

1.2 Related Work

Andrews *et al.* [5] proved that rings and DAGs are universally stable networks. They showed that Longest In System (LIS) and Shortest In System (SIS) are universally stable protocols and that FIFO is not universally stable. In fact, they showed that FIFO can become unstable at rates greater than 0.85. Several natural protocols such as NTG (Nearest To Go) and LIFO (Last In First Out) have been shown to be unstable at arbitrarily low rates [29]. Goel [20], Gamarnik [18], and Alvarez *et al.* [3] gave a simple and complete characterization of universally stable networks. Diaz *et al.* [21] improved the threshold of instability for FIFO to 0.83, and Koukopoulos *et al.* [23] improved it to 0.749. Lotker *et al.* [26] further improved the instability threshold to all rates above 0.5. They also proved that a network with diameter d is stable at all rates below $1/d$. Subsequent to our work, Lotker [25] has tightened our construction to almost match the above bound on the diameter. Also subsequent to our work, some partial progress towards an alternate proof of our result has been made by Koukopoulos *et al.* [22].

For the case when routes are not given by the adversary, Aiello *et al.* [1] and Andrews *et al.* [6] studied routing algorithms which ensure that no edge gets overloaded, assuming that the adversary injects packets for which this is feasible. Aiello *et al.* [2] have recently initiated the study of stability in the presence of fixed size buffers. Gamarnik [19] showed that it is undecidable to determine whether a given protocol is universally stable, for an interesting class of protocols. Feige [16] demonstrated non-monotonic phenomena in packet routing. Andrews [4] demonstrated the instability of FIFO in session-oriented networks [12, 13][2].

Stability of networks has also been studied from a more queueing theoretic viewpoint in stochastic networks, where the packets are injected and serviced according to a stochastic process. Instability in stochastic networks was first demonstrated by Rybko and Stolyar [28], building on the work of Lu and Kumar [27], and Kumar and Seidman [24]. The fluid model involves taking the fluid limit of a stochastic process. Dai [14] related stability in the fluid model to that in the stochastic model. Gamarnik [17] proved an analog of Dai's result for adversarial networks. Dai and Prabhakar [15] studied the stability of scheduling protocol for data switches with speedup in the fluid model. Bennett *et al.* [7] study the bounds on the worst case delay in a network implementing aggregate scheduling, assuming stochastic arrivals.

Bramson studied the stability of FIFO in two different stochastic models. He showed that FIFO is stable at all rates $R < 1$ if packets are injected by a Poisson process, and the time for a packet to traverse an edge is an i.i.d. exponential random variable (i.e. the network is Kelly-type) [11]. Quite interestingly, Bramson [10] also showed that FIFO can become unstable at arbitrarily low rates in a job-shop scheduling model. Superficially, it would seem that a minor modification of Bramson's techniques could imply our result. However, in Bramson's construction, the same job can visit the same machine multiple times, and have a *different mean processing time* on each visit. In trying to adapt his

[1] It is trivial to see that FIFO (in fact, any greedy protocol) is stable at rate $1/m$.

[2] Whether FIFO is unstable at arbitrarily low rates in the session-oriented model remains an interesting open problem.

result to the networking case, we run into the problem that different packets queued up at the same link can have different traversal times. Thus the same link appears to be of different length to different packets in a queue. To implement this construction directly requires injecting extra packets to make the link appear "long" to some packets; these extra packets result in a violation of the rate threshold. Hence, a gadget that can delay packets for arbitrarily long durations (like ours) seems inevitable. Even given our gadget, it is not obvious to us how to use a network similar to Bramson's.

Section 2 describes and analyzes our basic gadget. Section 3 gives the construction of a network which is unstable at arbitrarily low injection rates. The adversarial injection patterns are described in section 4. Finally, we prove that FIFO is unstable against this adversary in section 5.

2 The Basic Gadget

Section 2.1 describes the topology of the gadget. Section 2.2 talks about a special kind of flow. Finally, in section 2.3 an upper bound on the number of packets that escape the gadget for this flow is proved.

2.1 The topology of the gadget

The gadget has a parameter k. A k-gadget has $2k$ vertices: $v_1, \ldots, v_k, w_1, \ldots, w_k$. There are four groups of edges:

- Input edges, $g_1, \ldots, g_k$, pointing into $v_1, \ldots, v_k$ respectively.
- Output edges, $h_1, \ldots, h_k$, pointing out from $w_1, \ldots, w_k$ respectively.
- Load edges, $e_1, \ldots, e_k$, pointing from v_i to w_i.
- Helper edges, $f_1, \ldots, f_k$, pointing from w_i to v_{i+1}. The i's wrap around k.

Note that the load and helper edges form a ring with the input and output edges pointing in and out from alternate nodes. Figure 1 shows an example gadget.

2.2 A special flow

There are two kind of packets in this flow:

- For each $i \in 1, \ldots, k$, packets enter the gadget through g_i at rate 1. The route of a particular packet is:

$$g_i, e_i, f_i, e_{i+1}, f_{i+1}, e_{i+2}, \ldots, f_{i+k-2}, e_{i+k-1}, h_{i+k-1}$$

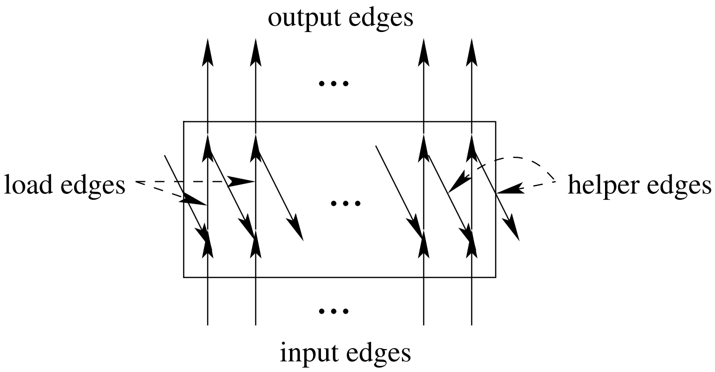

Figure 1. A gadget

Hence, each packet traverses all the load edges in the gadget. These packets are referred to as the *gadget-traversing* packets.

- For each load edge, e_i, single edge packets are introduced at a rate r. These packets are referred to as the *internal-gadget* packets.

2.3 Upper bound on the leak

Rate of leak from a gadget, R, is the sum of the rates at which the gadget-traversing packets arrive at the source node of an output edge. In the following, we prove an upper bound for R assuming the flows mentioned in section 2.2. For proving the upper bound we use the following property of FIFO:

Remark 2.1 *Let e be an edge in the network and $X_1, X_2, \ldots, X_n$ be n different types of packets which arrive at the source node of e at rates $R_1, R_2, \ldots, R_n$ respectively. Then the rate at which packets of type X_i traverse e is given by the expression,*

$$\min\{R_i, \frac{R_i}{\sum_{j=1}^{n} R_j}\}$$

Lemma 2.2 *During the time when the special flow is maintained, $R \leq \frac{k}{(1+r)^k}$.*

Proof: By symmetry, the total rate of arrival of packets at the source node of the load edges is the same. Let that rate be T.

$$T = 1 + r + r_1 + \ldots + r_{k-1}$$

where r_i is the rate of arrival of packets which have traversed i of the k load edges. Using remark 2.1, we obtain $r_1 = 1/T$. Similarly, for all $2 \leq i \leq k$,

$$r_i = r_{i-1}/T = 1/T^i$$

Since $T \geq 1 + r$ at all times, it follows that $r_k \leq 1/(1+r)^k$. Therefore at all times,

$$R = kr_k \leq k/(1+r)^k$$

■

Although the above lemma assumes the fluid model, the same bounds can be achieved for the discrete model using slightly higher rates. We omit the details from this version.

3 The Network Topology

The network contains three components, the columns, the connectors and the shortcuts. We describe each of these in turn, after first describing the concept of concatenation of gadgets, which is required for each of these components. In this section and all others, all gadgets would have the same parameter, k.

3.1 Concatenating gadgets

A gadget G_2 is said to be concatenated to a gadget G_1 if

1. The output edges of G_1 act as the load edges of G_2, and
2. The load edges of G_1 act as the input edges of G_2.

Note that more than one gadgets can be concatenated to a gadget, and also, a single gadget can be concatenated to more than one gadgets. A *Chain* $C = \langle H_1, H_2, \ldots, H_n \rangle$, of length n, is produced by the concatenation of the gadget H_{i+1} to the gadget H_i, for $1 \leq i < n$. A *Bridge* B of length l is said to exist between gadgets G_1 and G_2 if there exists a chain $\langle G_1, H_1, H_2, \ldots, H_l, G_2 \rangle$.

3.2 Columns

The network has two separate columns, C_1, C_2. A column is a chain of length α, where α is a parameter which will be specified later.

$$C_1 = \langle C_{1,1}, C_{1,2}, \ldots, C_{1,\alpha} \rangle$$

$$C_2 = \langle C_{2,1}, C_{2,2}, \ldots, C_{2,\alpha} \rangle$$

3.3 Connectors

There are two sets of connectors, one from C_1 to C_2 and the other from C_2 to C_1. Connectors are bridges of length β between each gadget of a column and the first gadget of the other column, where β is a parameter which will be specified later. So, for each $C_{1,i}$, we have the following bridge:

$$\langle C_{1,i}, D_{1,i,1}, D_{1,i,2}, \ldots, D_{1,i,\beta}, C_{2,1} \rangle$$

Similarly, for each $C_{2,i}$, we have the following bridge:

$$\langle C_{2,i}, D_{2,i,1}, D_{2,i,2}, \ldots, D_{2,i,\beta}, C_{1,1} \rangle$$

3.4 Shortcuts

Shortcuts are bridges of length 1 from each connector gadget, $D_{1,i,j}$ to $C_{1,1}$. We refer to the respective shortcut gadgets as $E_{1,i,j}$. Similar shortcuts exist between the other set of connector gadgets and $C_{2,1}$.

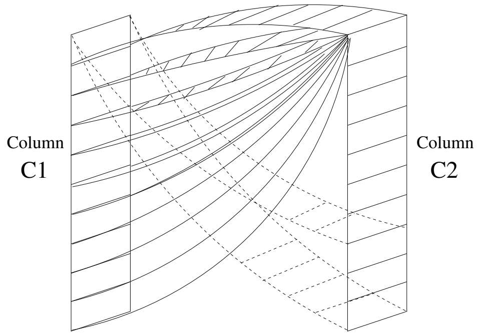

Figure 2. Topology of the network: Shortcuts are not shown

Figure 2 shows the schematic of the network topology.

4 The Adversarial Injection Pattern

The adversary introduces two kinds of flows of packets (in addition to the internal gadget packets). Flow through the columns and flow through the connectors. The description of both assumes the concept of activation of a gadget and sequential activation of gadgets in a chain, which are described next. We will assume that there is a chain $\langle G_1, G_2, \ldots, G_p \rangle$. These gadgets will be activated one after the other.

A packet is considered to be a *chain-traversing* packet for a chain $\langle G_1, G_2, \ldots, G_p \rangle$ if the route of the packet is gadget-traversing for each G_i in the chain.

4.1 Activation of a gadget

Precondition: There are t gadget-traversing packets in the queue of each input edge of the gadget G_i. Also,

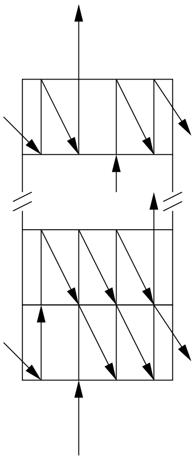

Figure 3. Route of a chain-traversing packet

there is no other packet in any other gadget in the chain. Moreover, the packets are chain-traversing for the chain $\langle G_i, G_{i+1}, \ldots, G_p \rangle$ (we will slightly modify this condition later).

Activation: During the activation phase, the adversary introduces internal-gadget packets at rate r in the gadget G_i. The activation phase lasts for t time steps. Note that during this phase the internal-gadget packets and the gadget-traversing packets compete with each other to traverse the load edges of the gadget.

Postcondition: There are $t' < t$ packets in the queue of each load edge of the gadget G_i and there are no other packets in any other gadget. Each packet is chain-traversing for the chain $\langle G_{i+1}, G_{i+2}, \ldots, G_p \rangle$. Later a lower bound for the quantity t' will be presented. In order to ensure that these conditions are met, we modify the injection in the following way (we essentially use the fact that the routes of all the packets are predetermined): (1) In order to get rid of internal-gadget packets which might be interleaved with the chain-traversing packets in the queue (after t time steps), we make sure that these particular internal-gadget packets are never introduced. (2) To avoid the case when packets in the queue of the load edges have other edges, within the gadget G_i, to traverse, we modify the routes of the packets such that after t time steps, they are simply queued in one of the load edges of G_i and the next edge on their route is a load edge of gadget G_{i+1}. This can be done since it amounts to merely taking away some of the edges from the path of the packets and can only reduce the injection rate. (3) We will assume that the routes of all the packets which escaped the gadget during the t time steps end at gadget G_i itself. This ensures that there are no packets in the other gadgets.

Sequential activation of gadgets: In a chain of gadgets, postcondition of the activation of a gadget acts as the precondition for the activation of the concatenated gadget. Sequential activation of gadgets is the cascading activation of gadgets starting from some initial chain-traversing packets waiting in the queue of the input edges of the first gadget.

4.2 Flow through the columns

Time steps are grouped into phases. It is assumed that at the beginning of an even phase there are s chain-traversing (for column C_1) packets in the queue of each of the input edges of the gadget $C_{1,1}$. We show that at the end of the phase there are more than s chain-traversing (for column C_2) packets waiting on each of the input edges of the gadget, $C_{2,1}$. Similarly, at the beginning of an odd phase, there are s' chain-traversing packets in the queue of each of the input edges of the gadget, $C_{2,1}$ and at the end of the phase there are more than s' chain-traversing packets in the queue of each of the input edges of the gadget, $C_{1,1}$. We only show packet injections for the even phases, packet injections for the odd phases are similar. Applying the above repeatedly leads to instability.

Subphases: Each phase is divided into α subphases. During the i^{th} subphase the gadget $C_{1,i}$ is activated.

4.3 Flow through the connectors

At the end of subphase i, let there be s_i packets in the queue of each of the input edges of $C_{1,i+1}$. During the next s_i time steps (i.e. the subphase $i + 1$), rs_i/k packets each are introduced in the queue of each input edge of $C_{1,i+1}$. The route of these packets are chosen such that they are chain-traversing for the chain $\langle D_{1,i,1}, D_{1,i,2}, \ldots, D_{1,i,j}, E_{1,i,j}, C_{2,1}, C_{2,2}, \ldots, C_{2,\alpha} \rangle$, for some $j < \beta$. After s_i time steps, these packets form the precondition for the sequential activation of the gadgets mentioned above. This is because there are s_i packets present in the queue of each input edge of $C_{1,i}$ ahead of the newly injected packets. The routes of the packets are such that at the end of the phase i.e. at time $\sum_{i=1}^{\alpha} s_i$, they are all queued at the input edges of the the gadget, $C_{2,1}$. This can be achieved using the shortcut gadgets and the fact that the route of each packet is determined in advance ($j < \beta$ can be chosen arbitrarily). Observe that the routes of the chain-traversing packets and the internal-gadget packets introduced during a subphase are mutually exclusive.

Moreover, the injection rate of each type of packet is less than r for any edge.

5 Proof of Instability

This section shows that at the end of a phase, there are more packets queued at the input edges of $C_{2,1}$, than were queued at the input edges of $C_{1,1}$. To initially generate constant number of packets to start phase 0, we can attach large acyclic graphs to the input edges of $C_{1,1}$, where the acyclic portion is used to generate the initial packets.

Analogous to our description of the flow, the analysis can be broken down into two parts: flow through the columns and flow through the connectors.

5.1 Flow through the columns

First we set the values of the various parameters. Let k be such that $(1+r)^k > 64k^3/r^2$, $\alpha = 4k/r$, $\beta = 16k^2/r^2$.

Lemma 5.1 *Let s_i be the duration for which the gadget $C_{1,i}$ remains activated during the i^{th} subphase. For $1 \le i \le \alpha$, the number of packets in the queue of each of the input edges of the gadget $C_{1,i}$ at time $\sum_{j=1}^{i-1} s_j$, i.e. the beginning of the i^{th} subphase, is lower bounded by $s/2$.*

Proof: Recall that the duration for which a gadget remains activated is the same as the number of packets waiting at each input edge when the gadget gets activated. is also the number of packets waiting at the Observe that $s_1 = s$. Lemma 2.2 implies that for a gadget activated for s_i time steps, the total number of packets which leak through in the s_i steps is at most

$$\left\lceil \frac{s_i k}{(1+r)^k} \right\rceil \le 2 s_i \frac{k}{(1+r)^k}$$

By our choice of α and k, it follows that $\alpha \frac{k}{(1+r)^k} < 1/2$. We now have

$$\begin{aligned} s_i &\ge s_{i-1}\left(1 - \frac{2k}{(1+r)^k}\right) \\ &\ge s\left(1 - \frac{2k}{(1+r)^k}\right)^{\alpha} \qquad [\text{since } i \le \alpha] \\ &\ge s\left(1 - \alpha\frac{2k}{(1+r)^k}\right) \\ &\ge s(1-1/2) \\ &= s/2 \end{aligned}$$

Hence, the lemma follows. ■

5.2 Flow through the connectors

Recall that for each $1 \le i < \alpha$, a set of rs_i/k (for each edge) chain-traversing packets for the chain: $\langle C_{1,i}, D_{1,i,1}, D_{1,i,2}, \ldots, C_{2,1}, \ldots, C_{2,\alpha}\rangle$ is introduced during the subphase $i+1$.

Lemma 5.2 *After the packets have activated $j-1$ gadgets in a connector, the number of packets queued at each input edge of the connector gadget $D_{1,i,j}$ is lower bounded by $rs/4k$.*

Proof: Using arguments similar to the one used in Lemma 5.1, we can conclude that the number of packets queued at each input edge of the gadget $D_{1,i,j}$ (after $j-1$ gadgets have been activated) is at least:

$$\frac{rs_i}{k}\left(1 - \frac{2k}{(1+r)^k}\right)^{\beta}$$

By the choice of β and k, $\beta\frac{2k}{(1+r)^k} < 1/2$ and since $s_i > s/2$, the number of packets is at least:

$$\frac{rs}{2k}\left(1 - \beta\frac{2k}{(1+r)^k}\right) \ge \frac{rs}{2k}(1-1/2) = \frac{rs}{4k}$$

Hence, the lemma follows. ■

Lemma 5.3 *At time $\sum_{i=1}^{\alpha} s_i$, i.e. at the end of all subphases, the number of packets queued at the input edges of $C_{2,1}$ is greater than s.*

Proof: We first show that β is large enough to allow the different sets of chain-traversing packets injected during each of the α subphases to simultaneously arrive at the input edges of $C_{2,1}$. This can be accomplished if the time step at which $D_{1,i,\beta}$ (for all $1 \le i \le \alpha$) is activated is greater than $\sum s_i$. By Lemma 5.2, the time step at which $D_{1,i,\beta}$ is activated is at least:

$$\frac{rs}{4k}\beta = \frac{rs}{4k}\alpha^2 = \alpha s \ge \sum s_i$$

Using Lemma 5.2, the number of packets on the queue of each input edge is at least:

$$\alpha rs/4k = s$$

Hence the lemma. ■

We now state the main theorem of our paper.

Theorem 5.4 *FIFO is unstable for arbitrarily low injection rates.*

Proof: In light of Lemma 5.3, we only need to show that the rate of injection is always less than r. The latter is true because the gadget-traversing packets and the internal-gadget packets are always injected for mutually exclusive set of edges and these packets themselves respect the injection rate r (See section 4 for details). ■

5.3 The size of the network

We now show that the size of the network is polynomial in $1/r$. Let k be $c\frac{1}{r}\log(\frac{1}{r})$ for a large enough c.

$$(1+r)^k > 2^{c\log(\frac{1}{r})} \geq 64k^3/r^2$$

Therefore, $k = O(\frac{1}{r}\log(\frac{1}{r}))$. Now, the size of the network is $O(\alpha\beta k) = O(k^4/r^3)$, which is polynomial in $1/r$. This is quite strong since it even excludes the possibility that FIFO might be stable at rate $O(1/\log^c m)$ where m is the size of the network and c is a constant. Subsequent to our work, Lotker [25] has tightened our construction to reduce the diameter of the network to $\tilde{O}(1/r)$.

References

[1] W. Aiello, E. Kushilevitz, R. Ostrovsky, and A. Rosen. Adaptive packet routing for bursty adversarial traffic. *Proceedings of the 30th Annual ACM Symposium on Theory of Computing*, pages 359–368, 1998.

[2] W. Aiello, R. Ostrovsky, E. Kushilevitz, and A. Rosen. Dynamic routing on networks with fixed-size buffers. *ACM-SIAM Symposium on Discrete Algorithms*, pages 771–780, 2003.

[3] C. Alvarez, M. Blesa, and M. Serna. Universal stability of undirected graphs in the adversarial queueing model. *Proceedings of the ACM Symposium on Parallel Algorithms and Architectures (SPAA)*, pages 183–197, 2002.

[4] M. Andrews. Instability of FIFO in session-oriented networks. *Proceedings of the eleventh annual ACM-SIAM symposium on Discrete algorithms*, pages 440–447, 2000.

[5] M. Andrews, B. Awerbuch, A. Fernandez, J. Kleinberg, T. Leighton, and Z. Liu. Universal stability results for greedy contention-resolution protocols. *37th IEEE symposium on Foundations of Computer Science*, pages 380–389, 1996.

[6] M. Andrews, A. Fernandez, A. Goel, and L. Zhang. Source routing and scheduling in packet networks. *42nd IEEE Symposium on Foundations of Computer Science*, 2001.

[7] J. Bennett, K. Benson, A. Charny, W. Courtney, and J. Le Boudec. Delay jitter bounds and packet scale rate guarantee for expedited forwarding. *ACM/IEEE Transactions on Networking*, June 2002.

[8] A. Borodin, J. Kleinberg, P. Raghavan, M. Sudan, and D. Williamson. Adversarial queueing theory. *28th ACM Symposium on Theory of Computing (Also, to appear in JACM)*, pages 376–385, 1996.

[9] M. Bramson. Instability of FIFO queueing networks. *Annals of Applied Probability*, 4(2):414–31, 1994.

[10] M. Bramson. Instability of FIFO queueing networks with quick service times. *Annals of Applied Probability*, 4(3):693–718, 1994.

[11] M. Bramson. Convergence to equilibria for fluid models of FIFO queueing networks. *Queueing Systems: Theory and Applications*, 22:5–45, 1996.

[12] R. L. Cruz. A calculus for network delay, part i: Network elements in isolation. *IEEE Transactions on Information Theory*, pages 37:114–131, 1991.

[13] R. L. Cruz. A calculus for network delay, part ii: Network analysis. *IEEE Transactions on Information Theory*, pages 37:132–141, 1991.

[14] J. Dai. Stability of open multiclass queueing networks via fluid models. In F. Kelly and R. Williams, editors, *Stochastic Networks*, volume 71 of *The IMA volumes in mathematics and its applications*, pages 71–90. Springer-Verlag, 1995.

[15] J. Dai and B. Prabhakar. The throughput of data switches with and without speedup. *Proceedings of the IEEE INFOCOM*, pages 556–564, 2000.

[16] U. Feige. Nonmonotonic phenomena in packet routing. *STOC*, pages 583–591, 1999.

[17] D. Gamarnik. Stability of adversarial queues via fluid models. *39th IEEE Symposium on Foundations of Computer Science*, 1998.

[18] D. Gamarnik. Stability of adaptive and non-adaptive packet routing policies in adversarial queueing networks. *Proceedings of the 31st ACM Symposium on Theory of Computing*, 1999.

[19] D. Gamarnik. On deciding stability of some scheduling policies in queueing systems. *SODA*, 2000.

[20] A. Goel. Stability of networks and protocols in the adversarial queueing model for packet routing. *Networks*, 37(4):219–224, June 2001.

[21] D. Josep, D. Koukopoulos, S. Nikoletseas, M. Serna, P. Spirakis, and D. Thilikos. Stability and non-stability of the fifo protocol. *Proceedings of the Thirteenth Annual ACM Symposium on Parallel Algorithms and Architectures*, pages 48–52, 2001.

[22] D. Koukopoulos, M. Mavronicolas, and P. Spirakis. *Personal Communication*, 2003.

[23] D. Koukopoulos, S. E. Nikoletseas, and P. E. Spirakis. The range of stability for heterogeneous and FIFO queueing networks. *Electronic Colloquim on Computational Complexity (099)*, 2001.

[24] P. Kumar and T. Seidman. Dynamic instabilities and stabilization methods in distributed real-time scheduling of manufacturing systems. *IEEE Transactions on Automatic Control*, AC-35:289–298, 1990.

[25] Z. Lotker. Note on instability of FIFO in the adversarial queueing model. *Personal Communication*, 2003.

[26] Z. Lotker, B. Patt-Shamir, and A. Rosen. New stability results for adversarial queuing. *Proceedings of the Fourteenth Annual ACM Symposium on Parallel Algorithms and Architectures*, pages 192–199, 2002.

[27] S. Lu and P. Kumar. Distributed scheduling based on due dates and buffer policies. *IEEE Transactions on Automatic Control*, 36:1406–1416, 1991.

[28] A. Rybko and A. Stolyar. Ergodicity of stochastic processes describing the operation of open queueing networks. *Problems of Information Transmission*, 28:199–220, 1992.

[29] T. Tsaparas. Stability in adversarial queuing theory. *M.Sc Thesis, Department of Computer Science, University of Toronto*, 1997.