Stability of Networks and Protocols in the Adversarial Queueing Model for Packet Routing

Ashish Goel*
University of Southern California

December 1, 2000

Abstract

The adversarial queueing theory model for packet routing was suggested by Borodin et al. [2]. We give a complete and simple characterization of all networks that are universally stable in this model. We show that the same characterization also holds for networks which are stable given that the packet forwarding protocol is FIFO (First In First Out). We also show that a specific greedy protocol, SIS (Shortest In System), is stable against 0/1 stochastic adversaries.

1 The Adversarial Model for Packet Injection

In traditional queueing theory, the source which generates network traffic is typically assumed to be stochastic. Adversarial Queueing Theory developed out of a recent need for more robust models for these sources. The growing complexity of network traffic makes it increasingly unrealistic to model traffic as, say, a Poisson stream. It is therefore desirable to have a general robust framework which makes as few assumptions about the network traffic as possible. Such a framework was developed by Borodin et al. [2] in the context of packet routing and several very interesting results in this model were proved by Andrews et al. [1]. We refer the reader to [2, 1] for a more thorough motivation of the adversarial model. In this model, packets are injected into the network by an adversary. To keep things simple, it is assumed that the route of each packet is given along with the packet itself. The adversary is limited in the following way: over any w consecutive steps, the adversary can inject at most $\lceil w(1-\epsilon)\rceil$ path packings into the network, where a path packing is an edge disjoint collection of simple paths in a network. Each path can be looked upon as a route that a packet needs to follow. A single path packing may, therefore, correspond to many packets. The parameter w is called the burst size, and $1-\epsilon$ is the injection rate. Such adversaries are called (w, ϵ)-adversaries.

Once injected, packets follow their routes one edge at a time till they reach their destination. Only one packet may cross an edge during one time step, so a protocol is needed to decide which packet should cross an edge if more than one packets are waiting. A protocol is said to be greedy

*Department of Computer Science, University of Southern California, Los Angeles CA 90089-0781. Email: agoel@cs.usc.edu . Part of this research was conducted when the author was at Stanford University, supported by ARO Grant DAAH04-95-1-0121 and NSF Grant CCR9304971.

if it always forwards a packet on an edge if there are packets waiting to cross that edge. Any non-greedy protocol can be simulated by a (centralized) greedy protocol. Distributed greedy protocols are the simplest to implement and reason about, and we limit ourselves to such protocols.

Definition 1.1 *A protocol P is stable on network G (or equivalently, the pair (P,G) is stable) if for all (w,ϵ)-adversaries, the maximum number of packets in the system, as well as the maximum delay a packet experiences, is bounded.*

Definition 1.2 *A protocol P is universally stable if (P,G) is stable for all networks G. Similarly, a network G is universally stable if (P,G) is stable for all greedy protocols P.*

Some common greedy protocols are defined below. To define a greedy protocol, it is sufficient to specify how contention is resolved when more than one packets want to traverse the same edge.

FIFO: First In First Out – the packet which arrived at this node first wins.

SIS: Shortest In System – the packet which was injected last wins.

LIS: Longest In System – the packet which was injected first wins. Intuitivley, LIS is a global form of FIFO.

NTG: Nearest To Go – the packet nearest its final destination wins.

FTG: Furthest to Go – the packet furthest away from its final destination wins.

One of the most interesting results of Andrews et al. is that several natural greedy protocols for packet routing are not universally stable. Specifically, FIFO is not stable! But fortunately, several natural greedy protocols are stable on all networks – for example, LIS and SIS, which give priority to the oldest and latest packet, respectively. Andrews et al. show a similar dichotomy for networks. They show that all greedy protocols are stable on rings, trees, and DAGs. But they do not characterize all universally stable networks. We give a complete characterization of such networks in Section 2. Specifically, we show that a connected undirected graph is universally stable if and only if it has at most one cycle. For directed graphs, we give a simple decision procedure to determine universal stability. Interestingly, the same characterization holds for graphs that can be unstable given that the packet forwarding protocol is FIFO.

Borodin et al. [2] introduced the notion of stochastic adversaries. The number of packet packings injected by a stochastic adversary during time step i is X_i, where X_is are i.i.d. random variables with mean less than 1. However, the path packings chosen by the adversary need not come from a probability distribution. Hence stochastic adversaries are more powerful than traditional source models, where packet injection is probabilistic for each source-destination pair. Borodin et al. give some results for stability of protocols against stochastic adversaries on directed cycles. In Section 3, SIS is shown to be stable on all networks against all 0-1 adversaries (a 0-1 adversary injects either zero or one path packings into the network in a given time unit according to some probability distribution – see section 3 for a detailed definition). The two results in this paper (Section 2 and Section 3) are not related to each other, except in that they both pertain to stability of networks.

Subsequent to our work, Gamarnik [6] gave an equivalence between a fluid model and the adversarial model of stability. One of the motivations behind his work was to characterize universally stable networks. However, that work did not result in such a characterization. Recently, Gamarnik [7] extended the results of this paper to the case of an undirected network where an edge can be crossed only in one direction during one time step.

1.1 Related Models: Stability in Stochastic and Fluid networks

Analysis of stability of networks has also been studied from a more queueing theoretic viewpoint, which considers stochastic networks. In stochastic networks, the packets are injected and serviced according to a stochastic process. Instability in stochastic networks was first demonstrated by Rybko and Stolyar [11], building on the work of Lu and Kumar [9], and Kumar and Seidman [8]. The fluid model involves taking the fluid limit of a stochastic process. Dai [5] related stability in the fluid model to that in the stochastic model. Surprisingly, the queueing-theoretic and adversarial notions of stability (including the results of this paper) were developed independently and in complete oblivion of each other till Gamarnik [6] proved the analogue of Dai's result for adversarial networks.

2 Universal Stability of Networks

This section deals with the adversarial model (see Section 1 for a definition of the adversarial model). Andrews et al. [1] prove that not all graphs are universally stable, by giving a counter-example. They also attempt to characterize graphs that are universally stable. They claim that for undirected graphs, the property of universal stability of a graph is closed under minor inclusion. Using the results of Robertson and Seymour [10], it follows that universal stability can be decided in polynomial time. While this approach does offer a lot of insight, it is unsatisfactory for several reasons. First, the results of Robertson and Seymour are non-constructive. We are guaranteed that there are only finitely many minor-minimal graphs which are not universally stable, but there is no general technique for finding these forbidden minors. Moreover, the results of Andrews et al. hold only for undirected graphs. Andrews et al. do observe that large classes of commonly used networks are unstable.

A simple and constructive characterization of universally stable graphs is given in this section. An undirected connected graph G with n vertices is shown to be universally stable iff it has at most n edges (Theorem 2.11). For directed graphs, the set of forbidden minors is given (Theorem 2.8) along with a simple decision procedure (Theorem 2.10).

Lemma 2.1 *If digraphs G_1 and G_2 are universally stable, then so is any graph G formed by joining them with edges that go only from G_1 to G_2.*

Proof: Assume that the adversary we are working against has rate $1-\epsilon$ and burst size w. Since G_1 is stable, any packets originating in G_1 get out of G_1 within T_1 time steps, where T_1 is some function of w and ϵ. Some of these packets may then enter G_2. Now consider a time window T_2 units long. Any new packets that enter G_2 during this window must have been introduced during $T_1 + T_2$ contiguous units. The number of path packings introduced during this interval can be at most $(T_1+T_2+w)(1-\epsilon)$. Choose any ϵ' such that $0 < \epsilon' < \epsilon$. Now, set $T_2 = (T_1+w)(1-\epsilon)/(\epsilon-\epsilon')$. During any window of size $t \geq T_2$ at most $t(1-\epsilon') - (T_2(\epsilon-\epsilon') - (T_1+w)(1-\epsilon)) = t(1-\epsilon')$ path packings are introduced into G_2. This implies that these packets could have been introduced by an adversary of burst size T_2 and rate ϵ'. But by definition of universal stability, G_2 is stable against such an adversary. Therefore, the traversal time for a packet as well as the queue lengths are bounded in G. ■

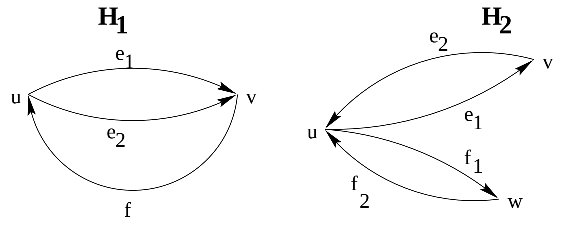

Figure 1: The graphs H_1 and H_2.

Lemma 2.1 implies that all acyclic digraphs are universally stable.

Corollary 2.2 *A digraph is universally stable iff all of its strongly connected components are universally stable.*

We now focus on graphs that are strongly connected. The following result by Andrews et al. [1] will be useful:

Lemma 2.3 *(Andrews et al.[1]) The directed cycle on any number of vertices is universally stable.*

The same result holds for the undirected cycle, as the undirected cycle can be modeled as two disconnected directed cycles. The next logical question to ask is the following: Does there exist a strongly connected digraph that is more complicated than a cycle and is still universally stable? The answer is no. Consider the following two simple digraphs, H_1 and H_2, also drawn in Figure 1.

$\underline{H_1}$: There are just two vertices u and v with two parallel edges e_1 and e_2 from u to v and an edge f from v to u.

$\underline{H_2}$: There are three vertices u, v and w with the following edges: $e_1 = (uv)$, $e_2 = (vu)$, $f_1 = (uw)$, $f_2 = (wu)$.

We will show that neither of these two simple graphs is universally stable. For now, we ignore the fact that the paths specified by the adversary have to be simple. We will come back to it later in the section.

Lemma 2.4 *H_1 is not universally stable.*

Proof: To prove instability, the adversary and the greedy protocol are allowed to work in collusion. Suppose there are s packets waiting to cross edge f, where s is larger than some large enough constant. The adversary introduces packets at a rate $r < 1$, and operates in four rounds. At the end of the fourth round, there will be more than s packets waiting to cross edge f. Our proof goes along the lines of a similar proof in [1] for a larger graph.

The first round lasts for s steps: the adversary introduces sr packets of the form (fe_2) and delays[1] all of these. At the beginning of the second round, there are sr packets of type (fe_2) waiting for edge f. During the second round (duration sr), sr^2 packets of type (e_2) and another sr^2 packets of type (fe_1) are introduced. Further, all these packets are delayed using packets of type (fe_2) left over from round 1. At the beginning of round 3, there are sr^2 packets each of type (e_2) and (fe_1). The adversary now introduces (in sr^2 steps) sr^3 packets each of types (e_2) and

[1]Recall that the adversary is also controlling the protocol and therefore can also choose which packets to delay as long as the greedy property is maintained.

$(e_1 f)$, delaying them using packets of type (e_2) and $(f e_1)$, respectively, from the previous round. At the beginning of the fourth round, there are sr^3 packets each of type (e_2) and $(e_1 f)$. During the fourth round (duration sr^3) the adversary introduces sr^4 packets each of types $(e_2 f)$ and (e_1). New packets of type $(e_2 f)$ are delayed using packets of type (e_2) left over from the previous round, whereas packets of type (e_1) are allowed to pass through, delaying sr^4 packets of type $(e_1 f)$ from the previous round (the remaining $sr^3 - sr^4$ packets of type $(e_1 f)$ are allowed to pass through). Notice that this is the only step when the greedy protocol differs from LIS. At the end of the fourth round, there are sr^4 packets each of types $(e_1 f)$ and $(e_2 f)$ waiting. All these packets need to cross edge f.

During the above procedure, the number of packets that need to cross edge f went from s to $2sr^4$. If $r > 0.5^{1/4} \approx 0.84$, then the number of packets goes up, and the adversary can use this procedure over and again to make the number of packets in the system unbounded.

In the above proof, the number of packets at the end of round 4 will actually be $2sr^4$ minus some constant. But since s was chosen to be larger than a large enough constant, this is not a problem. To initially generate a constant number of packets at edge f, we can attach a large acyclic graph at u, where the newly added acyclic portion is used to generate the initial packets. The new graph (ie. H_1 with an acyclic graph attached) is not universally stable. Corollary 2.2 now implies that H_1 is not universally stable. ■

Simple modifications to the above proof result in the following corollary.

Corollary 2.5 *Any graph obtained by replacing the edges e_1, e_2 and f in H_1 by disjoint directed paths is not universally stable.*

Lemma 2.6 *H_2 is not universally stable.*

Proof: The proof is similar to that for Lemma 2.4. The adversary again operates in four rounds. Suppose there are s packets waiting to cross edge e_1. In the first round (duration s), the adversary introduces sr packets of type $(e_1 e_2 f_1 f_2)$ and delays them. In the second round (duration sr), the adversary injects sr^2 packets each of types (e_2) and (f_2) and delays them. In the third round (duration sr^2), the adversary injects sr^3 packets each of types $(e_2 e_1)$ and (f_2) and again delays the newly introduced packets. In the fourth round (duration sr^3), the adversary introduces sr^4 packets each of types (e_2) and $(f_2 e_1)$. During this round, the adversary delays packets of type $(e_2 e_1)$ from the previous round and $(f_2 e_1)$ from the current round. Therefore, at the end of the fourth round, there are $2sr^4$ packets that need to cross edge e_1. ■

Corollary 2.7 *Any graph obtained by replacing the edges e_1, e_2, f_1 and f_2 in H_2 by disjoint directed paths is not universally stable.*

Any strongly connected digraph must either be a cycle, or it must consist of at least two cycles which either share an edge or a vertex. If these cycles share an edge, this digraph would be unstable by Lemma 2.4. If they share a vertex, this digraph would be unstable by Lemma 2.6. Imposing the restriction that each packet must follow a simple path, the forbidden minors for universally stable digraphs are given in Figure 2.

Theorem 2.8 *A digraph G is universally stable iff it does not contain any of the forbidden minors drawn in Figure 2.*

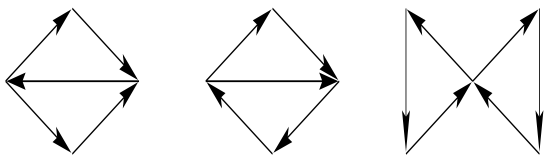

Figure 2: The forbidden minors for a universally stable digraph.

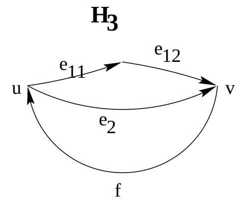

Figure 3: The graph H_3.

The above theorem does not indicate whether there exist any natural protocols for which the above characterization holds. We rectify that in the following theorem.

Theorem 2.9 *The pair (FIFO, G), where G is a digraph is stable iff G does not contain any of the forbidden minors drawn in Figure 2.*

The proof of the theorem is a simple modification of the proofs of Lemmas 2.4 and 2.6. The main differences are briefly sketched below for completeness.

Proof: We first prove that the graph H_3 drawn in Figure 3 is unstable for FIFO. We use essentially the same construction as in the proof of Lemma 2.4 with the following differences. Wherever the edge e_1 occurs in the construction, we replace it by the path $e_{11}e_{12}$. The duration of the third round is increased to $sr^2(1+r)$. and that of the fourth round to sr^4. During the fourth round, instead of introducing paths of the form e_1 we introduce paths of the form e_{12}. And there is an extra fifth round of duration sr^5 during which paths of the form f are introduced. It is straightforward to check that the graph is unstable if $r^6(2+r)/(1+r) > 1$ which holds for any $r \geq (2/3)^{1/6}$.

The above construction takes care of the first of the two minors in Figure 2. The third minor can be proved to be unstable for FIFO by modifying the proof of Lemma 2.6 using similar ideas to the above sketch. ∎

There is an even simpler procedure to check universal stability of a digraph G. Define a feasible line graph $L_F(G)$ of a digraph G as follows: the edge set of G forms the vertex set of $L_F(G)$ and there is an edge in $L_F(G)$ from e_1 to e_2 if, in the graph G, the head of e_1 coincides with the tail of e_2, and e_1 and e_2 do not form a directed cycle. The last condition captures the restriction that a packet cannot traverse the same edge in two directions consecutively. Note that the feasible line graph is itself a digraph.

The following theorem is now immediate.

Theorem 2.10 *A digraph G is universally stable iff in its feasible line graph $L_F(G)$, all strongly connected components are simple cycles.*

The above theorems completely characterize universally stable digraphs, and give an efficient $O(mn)$

time procedure. Computing the feasible line graph takes $O(mn)$ time[2] for deciding universal stability.

For undirected graphs, the above formulation translates into an even simpler characterization.

Theorem 2.11 *A connected undirected graph with n vertices is universally stable iff it has no more than n edges.*

Proof: If the graph has more than n edges than it must have two cycles. When we bi-direct each edge, this would result in an unstable minor. ∎

3 Against Stochastic Adversaries

This section deals with the stochastic-adversarial model (see Section 1 for a definition of this model). Borodin et al. [2] introduce the notion of stochastic adversaries. They show that several commonly used protocols such as LIS and FTG are stable on a directed cycle against stochastic adversaries with exponentially vanishing tails[3]. We show that SIS is stable on all networks against all 0-1 adversaries.

Definition 3.1 *A 0-1 stochastic adversary generates 0 path packings with probability ϵ and 1 path packing with probability $1-\epsilon$ during any single time step, for some constant $\epsilon > 0$.*

We first define a non-greedy protocol, SIS-NG, which takes at least as much time to route any packet as SIS. SIS-NG is then shown to be stable against a 0-1 stochastic adversary.

Definition 3.2 *The non greedy protocol SIS-NG does the following: a packet p is blocked at each edge in its path till the number of path packings introduced after p becomes less than the duration for which p has been waiting at this edge.*

Informally, SIS-NG assumes that each packet gets blocked once at each edge along its path by each of the path packings introduced after it.

Lemma 3.1 *Given the same sequence of requests, SIS takes at most as long to route any packet as SIS-NG.*

Theorem 3.2 *SIS-NG is stable against all 0-1 stochastic adversaries on arbitrary graphs.*

Proof: In this proof, the protocol will implicitly be assumed to be SIS-NG, and the mean number of path packings injected by the adversary during a single step will be $1-\epsilon, 0 < \epsilon < 1$.

Let $P_i(\alpha)$ represent the probability of α path packings being generated by the 0-1 adversary between the time when packet p was introduced and when it crosses the ith edge on its path.

[2] We follow the standard notation where the input graph has n vertices and m edges. Now, there are m vertices in the feasible line graph, and each vertex in this graph can have an edge to at most n other vertices. Testing for the presence of this edge is an $O(1)$ operation. The feasible line graph can have at most $O(mn)$ edges, and therefore all the strongly connected components can be genereated in $O(mn)$ time.

[3] These results were originally reported for adversaries with bounded variance, but the restriction that the input process have an exponential tail is essential.

Clearly, $P_0(0) = 1$, and $P_0(\alpha) = 0$ for all $\alpha > 0$. If there are β packets in the system when p crosses its ith edge, and there were α packets when p crossed its $(i-1)$th edge, then during the previous β steps, exactly $\beta - \alpha$ path packings must have been introduced, and no packet must have been introduced during the current step. This argument can be formalized to obtain the following recursive equation:

$$P_i(\beta) = \sum_{\alpha \le \beta} \binom{\beta}{\alpha} P_{i-1}(\alpha)\epsilon^{\alpha+1}(1-\epsilon)^{\beta-\alpha}.$$

We will now show inductively that the solution to the above equation is $P_i(\beta) = \epsilon^i(1-\epsilon^i)^\beta$. Clearly, this solution is valid for $i = 1$. Suppose the solution is valid for all $i \le j$. Now,

$$\begin{aligned}
P_{j+1}(\beta) &= \sum_{\alpha \le \beta} \binom{\beta}{\alpha} \epsilon^j (1-\epsilon^j)^\alpha \epsilon^{\alpha+1}(1-\epsilon)^{\beta-\alpha} \\
&= \epsilon^{j+1} \sum_{\alpha \le \beta} \binom{\beta}{\alpha} \left(\epsilon(1-\epsilon^j)\right)^\alpha (1-\epsilon)^{\beta-\alpha} \\
&= \epsilon^{j+1} \left(1 - \epsilon + \epsilon(1-\epsilon^j)\right)^\beta \\
&= \epsilon^{j+1}(1-\epsilon^{j+1})^\beta,
\end{aligned}$$

which completes the inductive proof. Let d be the maximum distance that a packet p needs to cover (clearly, $d \le m$, where m is the number of edges in the graph). Let β_d represent the number of path packings introduced after p but before p completes its journey. Expected delay for packet p is at most

$$\begin{aligned}
\mathbf{E}(d + d\beta_d) &= d\left(1 + \sum_{\beta \ge 0} \beta \epsilon^d (1-\epsilon^d)^\beta\right) \\
&= d\left(1 + \frac{1-\epsilon^d}{\epsilon^d}\right) \\
&= \frac{d}{\epsilon^d}.
\end{aligned}$$

This completes the proof of stability of SIS-NG against 0-1 adversaries. ■

Corollary 3.3 *If SIS-NG is used against 0-1 adversaries, the distributions of queue lengths and packet delays have exponentially vanishing tails.*

Proof: If $\beta_d \le k$ for a packet, then the delay faced by the packet can be at most $d + kd$ (each path packing can delay it at most once on each edge). Hence,

$$\mathbf{Prob}(\text{delay} > d + kd) \le \mathbf{Prob}(\beta_d > k) \le \epsilon^d \sum_{\beta > k} (1-\epsilon^d)^\beta = (1-\epsilon^d)^{k+1},$$

which proves that packet delays have exponentially vanishing tails.

Now,

$$\mathbf{Prob}(\text{number of packets} > dkm + dm) < \mathbf{Prob}(\text{oldest packet is at least } dk + d \text{ units old}).$$

There can be at most m packets per time step, and incrementing k by one corresponds to d time steps. Using the bound on delay obtained earlier in this proof, the above probability is at most $md \sum_{\beta > k} (1 - \epsilon^d)^\beta = (dm/\epsilon^d)(1 - \epsilon^d)^k$, which has an exponentially vanishing tail. ■

The expected delay is exponential in d. But for a large class of networks, d can be logarithmic in the network size, which would result in polynomial delays and queue sizes. Also, by Lemma 3.1, all the above results hold for SIS as well.

4 Open Problems

The obvious open problem posed by this paper is to characterizee the exact combinations of network, forwarding protocol, and injection rate that cause instability. More specifically, it would be interesting to determine whether FIFO can be unstable at arbitrarily low rates or whether there is a critical injection rate below which FIFO is stable. For stochastic networks, for the scenario when mean service times can vary for different visits by the same packet to the queue, Bramson [3] answered the previous question by demonstrating that FIFO can be unstable at arbitrarily low rates. Subsequently, Bramson [4] also showed that FIFO is stable in a class of stochastic networks called Kelly-type networks.

References

[1] 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.

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

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

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

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

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

[7] 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.

[8] P.R. 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.

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

[10] N. Robertson and P.D. Seymour. Recent results on graph minors. *Surveys in Combinatorics*, Cambridge University Press, 1985.

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