CS294-1 On-line Computation & Network Algorithms	**Spring 1997**

Lecture 11: February 25, 97

Lecturer: Professor Jon Kleinberg *Scribe: Tina Wong*

In this guest lecture John Kleinberg talks about Adverserial Queuing Theory. The talk is based on the papers [BKRSW95] and [AAFKLL96].

11.1 Introduction

We first go over some introductory material on queuing theory.

11.1.1 Routing in Networks

Circuit-switched and packet-switched are the two basic routing technology for networks. In this lecture, we concentrate on the "dynamics of traffic" of the former.

The model generally used consists of an injection process, which inject packets into a link one at a time, and a link-traversed process, which let packet crosses the link one at a time. Contention can result if there are more than one packet waiting to traverse a link; this can be resolved by using a contention resolution protocol. Examples of such include First-In-First-Out (FIFO), Farthest-To-Go (FTG), Nearest-To-Go (NTG), and mechanisms that based on timestamps like Longest-In-System (LIS) and Shortest-In-System (SIS).

11.1.2 Packet Routing

The research of packet routing is applicable in the areas of parallel architectures and communication networks. There are research on static routing, and dynamic routing in which you want to allow the algorithm to run continuously and study its long term behavior.

11.1.3 Queuing Theory

An example of queuing theory in practice is ATM: packets arrive at the ATM switch with certain arrival rate, and the switch has to service the packets efficiently with a limited service rate.

There are many well solved problems in queuing theory but they all make assumptions about the network configurations. For example, when service rate or arrival rate is exponentially distributed, we can infer several things like which is the best contention resolution protocol to use in this situation. However, our goal here is to find out what can be proved with as few of assumptions as possible.

11.2 Adverserial Queuing Theory

In this model, packets are generated by an adversary, and we do not make any assumptions about them. We study the evolution of traffic in the network, which is a game played between the adversary and the on-line algorithm.

ADV: injects packets with specified paths. For example, a stochastic adversary can inject packets based on a stochastic process. There are conditions the ADV need to follow:

- *rate restrictions* (w, r): In every window of w steps, at most $r \cdot w$ packets can be injected into any single edge. w is a natural number, and r is between 0 and 1.

ALG: advances packets. The conditions are:

- at most one packet can be inject into each edge at a time
- *greedy*: which means whenever there are packets waiting to be injected, inject them immediately.
- takes one time step to advance a packet.

Definition 11.1 (Stability) *Given a graph G, an adversary A, and a path P, System (G, A, P) is stable if $\exists$ C such that at all times, the number of packets in system $\leq C$.*

Conjecture: $\forall G, \forall P, \forall A : (G, A, P)$ is stable. This is false.

- Here is an example of an unstable network: (The main idea behind this is to create a swamping effect to make the network unstable.)

  phase 1 set $A =$ "train" of s packets passing through $v \Rightarrow w$.

  phase 2 queue B of size $r \cdot s$, injecting while A is passing through. Now, A and B are both queued up.

  phase 3

  a. next $r \cdot s$ steps: create queue C of size $r \cdot r \cdot s = r^2 s$

  b. block B for $r \cdot s$ steps, thereby shrinking B to $r^2 s$.

  phase 4 release B and C. —$B \cup C$— is a train of size $r^2 \cdot s + r^2 \cdot s = 2r^2 s$

  So, if $r^2 > \frac{1}{2}$, $|B \cup C| > |A|$.

  The contention resolution protocol is Nearest-To-Go. NTG is instable for any rate. (though you have to grow the network as the rate decreases). On an $n \cdot n$ torus, NTG is instable against an ADV of rate $\Theta(\frac{1}{n})$.

Definition 11.2 (Universal Stability (G, A, P)**)** *P is universally stable if $\forall G \forall A (G, A, P)$ is stable. G is universally stable if $\forall A \forall P (G, A, P)$ is stable.*

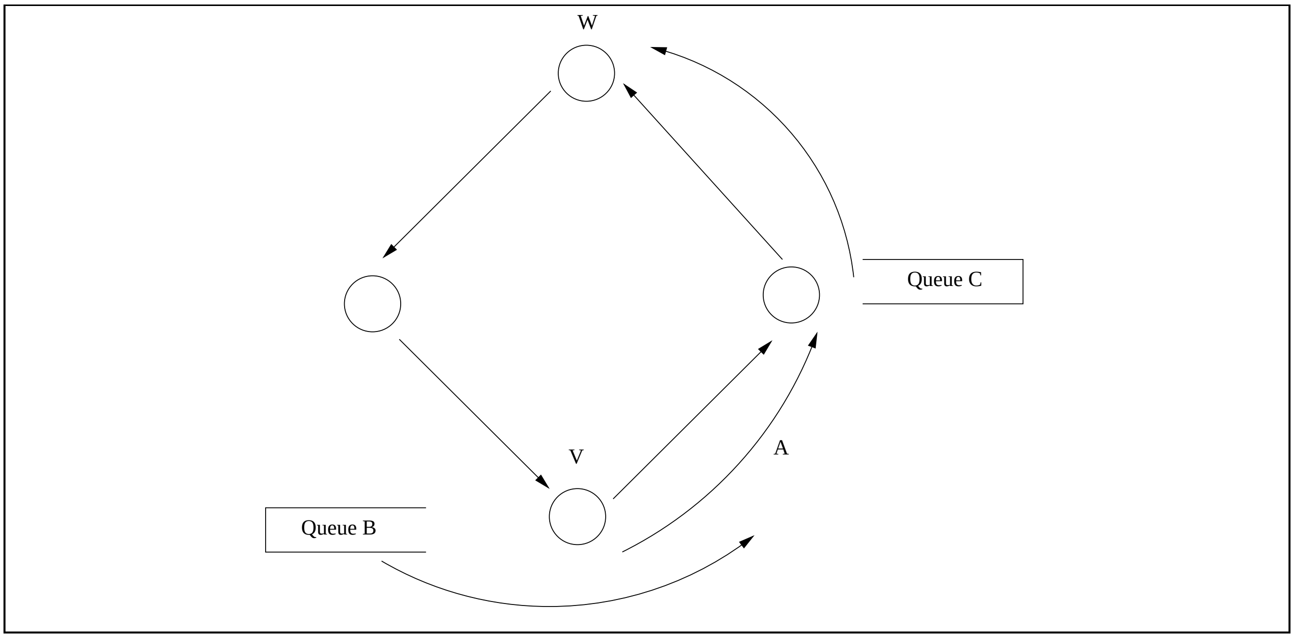

Figure 11.1: Example of an unstable network that satisfy rate restrictions and conditions.

For example, a ring is universally stable.

Universally stable contention resolution protocols are FTG ($\Theta(exp(n))$), LIS ($O(exp(n))$), and SIS ($\Theta(exp(n))$). NTG, FIFO, LIFO are not universally stable.

Useful Facts:

- $\forall(G, A, P)$, each queue empties infinitely often
- if at most C packets is in the system, then delay $\leq O(\frac{m}{s-r} \cdot C)$
- if delay $\leq d$, then number of packets is $O(m \cdot d)$

Claim 11.3 *Shortest-In-System (SIS) is stable.*

Proof: Assume some graph G, some adversary of rate $(w, 1-\epsilon)$, link e and packet p. Suppose k packets in system currently have priority over p. Let $k' > k$ and a multiple of w.

Claim: p will cross e within the next $\frac{k'}{\epsilon}$ steps.

Consider the next t steps. At most $t \cdot (1-\epsilon)$ packets arrive wanting e. So if p doesn't move in these t steps, then someone else moves:

- k packets already there with priority
- $t(1-\epsilon)$ already injected

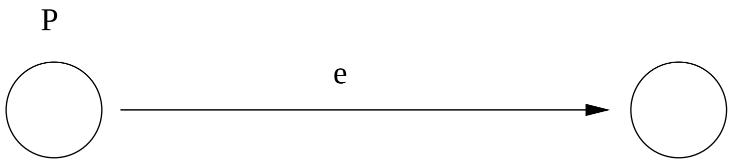

Figure 11.2: Packet p and link e.

If $t > \frac{k}{\epsilon}$ then $k + t(1-\epsilon) = k + t - \epsilon \cdot t < t$. ■

Claim 11.4 *SIS: can show the delay is bounded, so the number of packets is bounded.*

Proof:

step 1 Packet p is injected. Path of p has at most m edges (no loops). But at most w packets can have priority on first edge. Therefore, p crosses in $\leq (1-\epsilon) \cdot w$ steps.

step 2 At most $(1-\epsilon) \cdot w$ packets injected. p crosses 2nd edge in $\frac{(1-\epsilon) \cdot w}{\epsilon}$ steps.

step 3 At most $\frac{(1-\epsilon) \cdot w}{\epsilon} \cdot (1-\epsilon)$ can arrive with priority over 3rd edge. p crosses 3rd edge in $\frac{(1-\epsilon)^2 \cdot w}{\epsilon^2}$ steps.

By induction: Only bounded amount of packets show up with priority, crosses r^{th} edge within $\frac{(1-\epsilon)^r \cdot w}{\epsilon^r}$ steps.

■

11.3 Randomized Protocol

X_t = number of packets in the system at time t. (random variables)

$Seq[X_i]$ is bounded by C if $\Pr[X_i > C + k] < e^{-\alpha k}$

[Hujek 1982] says it is enough to show that x_i has the following:

- $\Delta x = x_{i+1} - x_i = O(1)$
- $\mathrm{E}[\Delta x] < -\epsilon$ once $x_i > C$

[Leighton-Maggs-Rao] works on static packet routing. Some definitions:

- c: congestion = max edge usage
- d: dilation = max path length

- packet schedule: for each packet p, give the time when it should cross each edge

How fast a schedule can we find? The obvious lower bounds are d, c.

Theorem 11.5 *$\forall$ packet routing problems, $\exists$ a schedule of time $O(c+d) = O(\max(c,d))$*

Simple Leighton-Maggs-Rao: $O(c + d\log N) \rightarrow (1+\epsilon)c + O(d \log mcd)$, with m = number of edges.

Idea: break time into intervals of length $\frac{(1+\epsilon)}{\delta} \cdot \log(mcd)$, with small $\delta > 0$.

Packet p: pick random initial delay from $1 \ldots \frac{\alpha c}{\log(mcd)}$. The goal here is to desynchronize packets. Go one edge per interval.

Claim 11.6 *with high priority, no interval is overfull.*

Proof:

$N_{e,i}$ = number of packets arriving for e in interval i. The time bound is:

$$[\frac{\alpha c}{\log mcd} + d]\frac{1+\epsilon}{\alpha} = (1+\epsilon)c + \frac{1+\epsilon}{\alpha} d \log mcd$$
$$\mathrm{E}[N_{e,i}] = c \cdot \frac{\log(mcd)}{\alpha c} = \frac{\log(mcd)}{\alpha}$$

Chernoff bound: let $X_1 \ldots X_n$ be independent $0-1$ random variables, $Y = \sum X_i$;

$$\mu = \mathrm{E}[Y] = \sum \mathrm{E}[X_i] = \sum \Pr[X_i = 1]$$
$$\Pr[Y > (1+\epsilon)\mu] < e^{-\frac{1}{4}\epsilon\mu}$$

So how many possible $N_{e,i}$ are there? There are m choices for e, and $\frac{\alpha c}{\log mcd} + d$ choices for i, which is less than mcd choices.

Probability of each being bad is $< \frac{1}{(mcd)^2} \Rightarrow \Pr[anybad] < \frac{1}{mcd}$

What bound do we need in the Chernoff bound?

$$\Pr[Y > (1+\epsilon)\mu] < e^{-\frac{1}{4}\epsilon^2\mu} < e^{-\frac{1}{4}\epsilon^2 \frac{\log mcd}{\alpha}} < e^{-2\log mcd} = \frac{1}{mcd^2}$$

■

Protocol:

- Parameters T, T', μ, $(T' > T + \mu)$

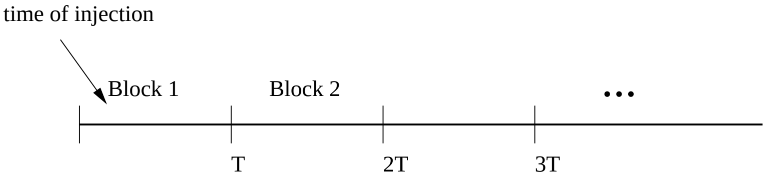

Figure 11.3: Illustration of Randomized Protocol.

- Packet arrives at time t, label $= T^{'}\lceil\frac{t}{T}\rceil + rand[1\ldots\mu]$
- p increments label each time it crosses an edge

Leighton-Maggs-Rao says with high probability block 1 ends by $2T$, with high probability block 2 ends by $3T$, and so on.

Block 1: easy, it has priority

Block 2: show that it doesn't hurt that some of the packets of block 2 have already been traversing edges.

Suffix: schedule $\sigma : P\times$ Edge $\rightarrow$ Time $\sigma^{'}$ is a suffix of σ if:

- each packet started at some arbitrary point on its path
- at edge e, greedily advance packet p with smallest $\sigma(p, e)$.

Claim 11.7 *$\sigma^{'}$ is at least as fast as σ*

Proof: Induction on time.

At $t = 0$, no packet is further behind in $\sigma^{'}$ than in σ.

Consider first time t when a packet p "falls behind" in $\sigma^{'}$. In step $t-1$, p is at tail of e:

- some other q must have crossed e
- $\sigma(q, e) < \sigma(p, e) \leq t - 1$.

The reason is if p can't cross e, someone else crossed, then it must have already been behind, so t wasn't the first time it fell behind. ∎

11.4 Open Problems

A list of open problems in this area are presented in class:

1. Is Poisson arrival or constant edge traversal or FIFO universally stable?
2. Is there $0 - 1$ law for protocol? $\exists \epsilon$ such that FIFO is stable whenever $r < \epsilon$?
3. Adaptive routing
4. Queue size
   - does LIS has polynomial bound?
   - does there $\exists$ a deterministic local control algorithm with polynomial bound?
5. Find simple/constructive/local-control proof of LMR.

References

[AAFKLL96] D.M. Andrews, B. Awerbuch, A. Fernandez, J. Kleinberg, F.T. Leighton, Z. Liu. Universal stability results for greedy contention-resolution protocols. In *PrProc. 37th IEEE Symposium on Foundations of Computer Science*, 1996.

[BKRSW95] A. Borodin, J. Kleinberg, P. Raghavan, M. Sudan, D. Williamson. Adversarial queueing theory. In *Proc. 28th ACM Symposium on Theory of Computing '96*, 1996.