Lecture 6

1 Delay Sequences

Definition 1 (Delay Sequence - FIFO) *A Delay Sequence is a sequence of pairs* (p_i, e_i) *satisfying one of two properties:*

> *Let* p_i*,* p_{i+1} *be packets, and* e_i*,* e_{i+1} *be edges sharing the same node on a path in a network such that* p_i *travels across edge* e_i *at time* i*.*
>
> 1. $p_i = p_{i+1}$ *and* $e_i \neq e_{i+1}$
> 2. $p_i \neq p_{i+1}$ *and* $e_i = e_{i+1}$

The above conditions for the definition of a delay sequence deserves some explanation. The first property is the condition necessary for packet p_i to cross edge e_i at time t. The second property refers to the possibility that the packet must wait because another packet is being sent across the edge at time i.

It is important to note that if we know a packet p_t arrives at a particular node at time t, we can always construct a delay sequence as defined above. Furthermore, this definition is valid for First In, First Out scheduling (ie. if there are two packets waiting to cross a particular edge, then the packet that arrived first is sent across the edge). In addition to this, the unsent packets that are not immediately sent across the edge is stored in an unbounded queue. We will later see that changing this constraint to a bounded queue significantly changes the performance of the routing algorithm.

Before jumping into the analysis of FIFO and Greedy scheduling, it should be emphasized that delay sequences are just combinatorial objects. We use this notion of a delay sequence to abstract the events over a particular path making it somewhat simpler to analyze.

2 Results From Last Time

Recall that

$$\begin{aligned} C &= \text{maximum congestion over any path} \\ D &= \text{dilation (maximum length of the path)} \end{aligned}$$

that specify the quality of the path

First, recall that if we use a greedy scheduling scheme, we have that routing takes $O(CD)$ when using unbounded queues.
In addition, we have

LEMMA 1
If a packet arrives at time t, then $\exists$ a delay sequence of length t.

COROLLARY 2
Given a delay sequence of length $t \Rightarrow t \leq C + D^*$ where $D^* =$ maximum length of any path.

These two statements provide very useful results. The above lemma tells us that we can always construct a delay sequence whenever a packet arrives at at time t. The corollary tells us that we cannot construct a delay sequence that is larger than $C + D^*$, which allows us to make an upper bound in our analysis.

APPLICATION 1 *$\theta(\log N)$ time routing permutation on butterfly network. This result follows from the path selection giving path congestion $C = O(\log n)$.*

We will now look at a different scheduling algorithm that is based on random priorities:

APPLICATION 2 *$\theta(\log N)$ time routing of $\theta(\log N)$ relation where path selection gives $C = \theta(\log N)$ and $D^* = \log N$.*

(The strict definition of relation is each node starts with h packets and ends with h packets).

3 Random Priority Routing

Before the algorithm is run, each packet is assigned a random priority. If there is any contention between packets for use of an edge, the edge will send the packet that has the highest priority. To simplify calculations, we will assume that each packet has a unique priority. This problem can be circumvented by choosing priorities from the reals or from a large enough sample space that the probability of collision would be small.

The purpose of considering Random Priority with $\theta(\log N)$ relation is to show that routing $\log N$ packets from each node takes about the same amount of time as routing *one* packet from each node. Intuitively, this means that the first application leaves some resources idle, while the second application makes more efficient use of the resources completing more work in the same amount of time.

We analyze Random Priorities by considering sorted delay sequences with respect to a random function, $r(\cdot)$.

DEFINITION 2 (DELAY SEQUENCE - RANDOM PRIORITY) *A Delay Sequence is a sequence of pairs (p_i, e_i) satisfying one of two properties:*

> *Let p_i, p_{i+1} be packets, and e_i, e_{i+1} be edges sharing the same node on a path in a network such that p_i travels across edge e_i at time i.*
>
> 1. *$p_i = p_{i+1}$ and $e_i \neq e_{i+1}$*
> 2. *$p_i \neq p_{i+1}$ and $e_i = e_{i+1}$ <u>and</u> $r(p_i) > r(p_{i+1})$.*

LEMMA 3 (PRIORITY ALGORITHM)
If a packet arrives at time t, then $\exists$ a sorted delay sequence w.r.t. $r(\cdot)$ of length t.

THEOREM 4
Priority Algorithm delivers all packets in $O(C + D^* + \log N)$ time w.h.p.

PROOF:

$$\mathbf{Pr}[\text{delay sequence of length } t \text{ is sorted}] \leq \frac{1}{\#\text{ outputs}} \tag{1}$$
$$\leq \frac{1}{(t - D^*)!} \tag{2}$$

$$\#\text{ of delay sequences} \leq (\#\text{outputs})\,(\text{in-degree of nodes})^{D^*} \binom{t}{D^*} C^{t-D^*} \tag{3}$$

Let d_i = in-degree, and N = # outputs

$$\mathbf{Pr}[\text{any delay sequence is sorted}] \leq (N)\,(d_i)^{D^*} \binom{t}{D^*} C^{t-D^*} \frac{1}{(t - D^*)!} \tag{4}$$
$$\leq N\,(d_i)^{D^*} \left(\frac{te}{D^*}\right)^{D^*} C^t \left(\frac{e}{t - D}\right)^{t-D^*} \tag{5}$$

Note that $d_i = 2$ in this case. If we pick $t = D + D\log\left(\frac{t}{D^*}\right) + 2eC + c\log N$, the above probability becomes smaller than $\frac{1}{N^c}$. With closer analysis and tighter approximations, it can be shown that $t = O(C + D + \log N)$ w.h.p. □

4 Bounded Queue Routing

Bounded queues for pending packets introduces a new possibility in routing events. In the unbounded case, a packet either goes across an edge or waits for another packet to cross the edge. In both cases, the edges always send a packet across if it has waiting packets. In the bounded case, an edge may sit idle not sending *any* packet if the parent has a full queue. Therefore, the only thing that we can deduce from a waiting packet is that *some* packet crosses an edge in the chain of parents. To deal with this new complication, we modify the definition of a delay sequence.

DEFINITION 3 (DELAY SEQUENCE - UNBOUNDED QUEUE) *A Delay Sequence is a sequence of pairs (p_i, e_i) or triples (p_i, e_i, s_i) satisfying one of following properties:*

> *Let p_i, p_{i+1} be packets, and e_i, e_{i+1} be edges sharing the same node on a path in a network such that p_i travels across edge e_i at time i.*

1. $p_i = p_{i+1}$ *and* $e_i \neq e_{i+1}$
2. $p_i \neq p_{i+1}$ *and* $e_i = e_{i+1}$
3. *If* (p_i, e_i, s_i), *then* e_{i+1} *points to* e_i *and* $s_i \subseteq \{$*packets that use* $e_i\}$ *where* $|s_i|$ = *queue size*

The addition of part (3) explains that if a packet is sent from a full queue by edge e_i (thus becoming less than full), then a packet will be sent out by child edge e_{i+1} which points to edge e_i.

It is worth noting that this new constraint of having a bounded queue for every edge adds an exponentially larger penalty on performance. Before, we saw that unbounded queues had a performance of $O(CD)$, while the bounded queue model has a lower bound of $\Omega(C^D)$. You can informally convince yourself of this fact by considering a binary tree that contains all queues full. Since all queues in the path are filled, all children edges must wait until room is made in a parent edge queue. Therefore, only one packet is sent at a time in the network. Therefore the time to send all packets is the size of the tree, which is C^D.

LEMMA 5
If a packet arrives at time t, then $\exists$ a delay sequence (bounded queue) of length t.

DEFINITION 4 (LEVELLED NETWORKS) *A Levelled Network is a network where each node can be categorized into a level such that there exists only edges connecting nodes of level i to nodes of level $i+1$ for all i.*

We now give an overview of the Ghost Message Algorithm. The main strategy of this algorithm is to guarantee that packets are sent sorted. This is achieved by an inductive definition for the algorithm. A node will only send a packet if it knows that it will never send a packet with higher priority. Therefore, if all nodes follow this rule, then when a node sends a parent a packet p_i, the parent knows it will never see a packet with greater priority than p_i's priority from that child node.

The only problem that arises so far is a parent node will never know the priority of the packets its children nodes send if the children nodes don't send the packets to that parent. This is remedied by sending a Ghost Message. A ghost message is basically a packet g_i that is sent to all parents when a node sends a packet p_i to any parent. The priority of g_i is the same as the priority of p_i. This ensures that all parents know the current priority of a packet any child node is sending. With ghost packets, all nodes have adequate information to guarantee that packets are only sent in sorted order (highest to lowest). Therefore, a node will only send a packet when all its children nodes only have lower priority packets to send. The ghost algorithm will be clearly defined in the next lecture. Like in the previous analyses using delay sequences, we will make use of the following lemma next time.

LEMMA 6 (GHOST MESSAGE ALGORITHM)
If a packet arrives at time t, then $\exists$ a sorted (bounded queue) delay sequence of length t.