CSE6512 Lecture 25 Notes

Sudipta Pathak

December 06, 2011

Problem 2:

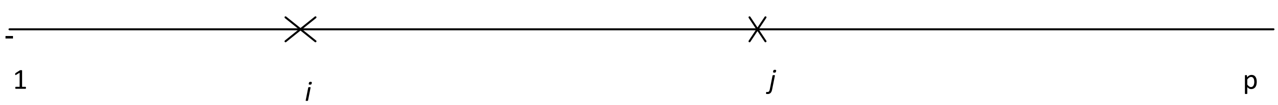
IN A LINEAR ARRAY, EACH NODE IS THE DESTINATION OF ATMOST ONE PACKET. ROUTE THE PACKETS.

Lemma :

We can route these in ≤ (p-1) steps.

Proof :

Use the **FARTHEST DESTINATION FIRST** Queue Discipline. Consider a packet π whose origin is *i* and whose destination is *j.*

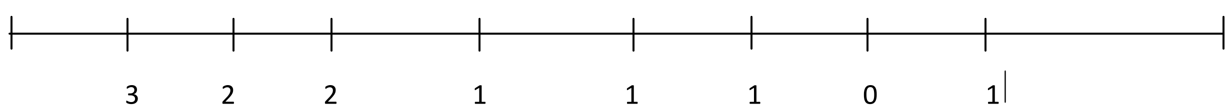


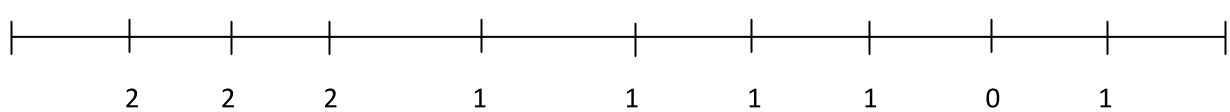
Let m_q be the number of packets that can delay π and that are in node q ($1 \le q \le p$).

Let r be such that $m_r, m_{r+1}, \ldots, m_p \le 1$ and $m_{r-1} > 1$

Call the sequence $m_r, m_{r+1}, \ldots, m_j$ as the free sequence.

Note: In every time step, at least one new packet will join the free sequence. Also, if a packet is in the free sequence it will neither delay, nor be delayed by any other packet. See the following example.





Note that $\,^{p}_{q=1} m_q \leq$ (p-j).

- ⇨ After (p-j) steps all of the packets that can delay π (including π) will be in the free sequence.
- ⇨ π needs an additional ≤ (j-i) steps.
- ⇨ Total time needed for π is ≤ (p-i). □

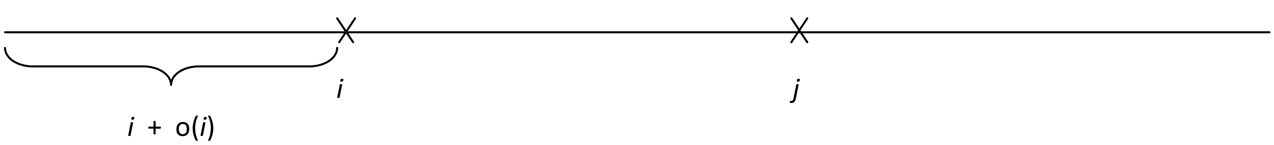
<u>Problem 3</u>

In a **LINEAR ARRAY** the number of packets originating from the first *i* nodes is *i* + o(*i*) for 1≤ *i* ≤p. Route the packets.

<u>Lemma :</u> We can route the packets in ≤ p + o(p) steps.

<u>Proof :</u> Use the **FARTHEST ORIGIN FIRST Q.D.**

CONSIDER A PACKET π whose origin is *i* and whose destination is *j*.



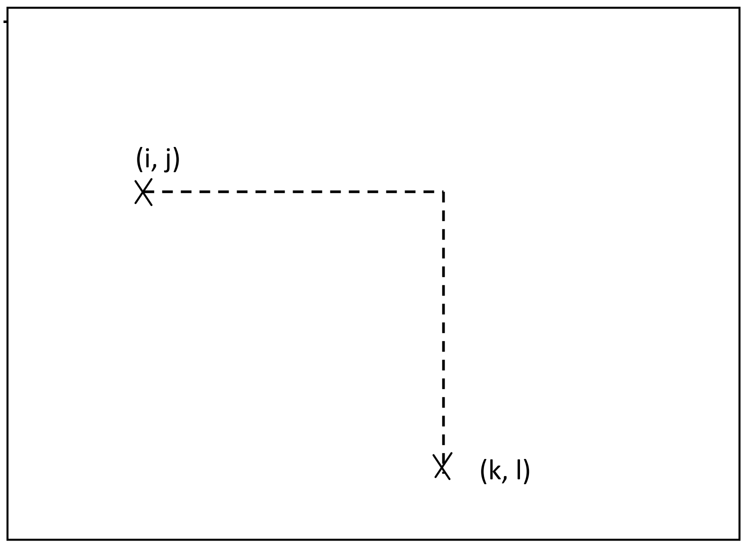
The packets π can only be delayed by *i* + o(*i*) packets.

If π is delayed more than once by some packet Q, it means that Q has been delayed by another packet R with a higher priority. R will not get to delay π.

- ⇨ The total delay π suffers is ≤ *i* + o(*i*)
  It has to travel a distance of (*j*-*i*)
- ⇨ The time taken by π is *j* + o(*i*) ≤ p + o(p). □

Routing in a n × n Array:

Consider any partial permutation. We can use a greedy algorithm to route the packets in time that is no more than the diameter of the mesh. This will be optimal in the worst case.



Consider a Packet

π whose origin is

(i, j) and destination

is (k, l).

π uses the shortest path along the row of origin first (Phase 1) and then

along the correct column (Phase 2).

There is no need for a Q.D. in Phase 1.

Use the **FARTHEST DESTINATION FIRST** Q.D. in Phase 2.

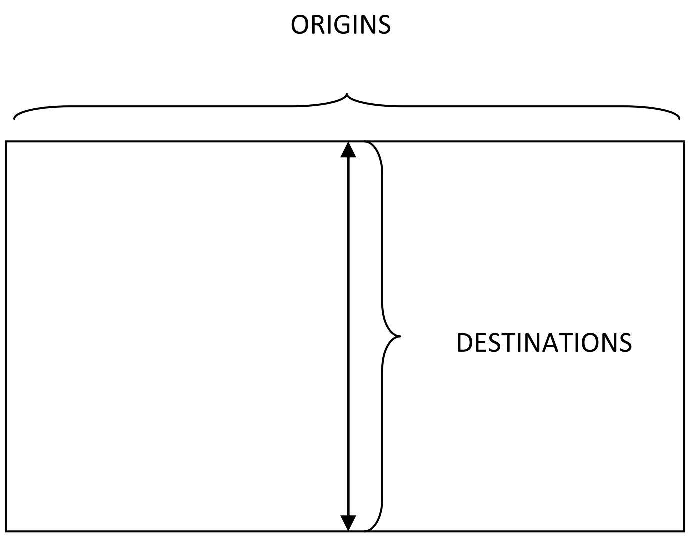
Phase 1 takes ≤ (n-1) steps.

Phase 2 takes ≤ (n-1) steps,

since Phase 2 is nothing but Problem 2.

- ⇨ Total time is ≤ 2(n-1). This indeed great!

What is the queue size needed? Consider an input where all the packets destined for column n/2 have their origins in row 1:



In this case, it is easy to see that the QUEUE SIZE needed is (n/2). This is because, at each time step two packets arrive at node (1, n/2) and only one of these can be sent along column n/2 and the other packet has to be queued. This happens for the first n/2 steps.

We can use randomization to reduce queue sizes. This will be discussed in the next lecture.