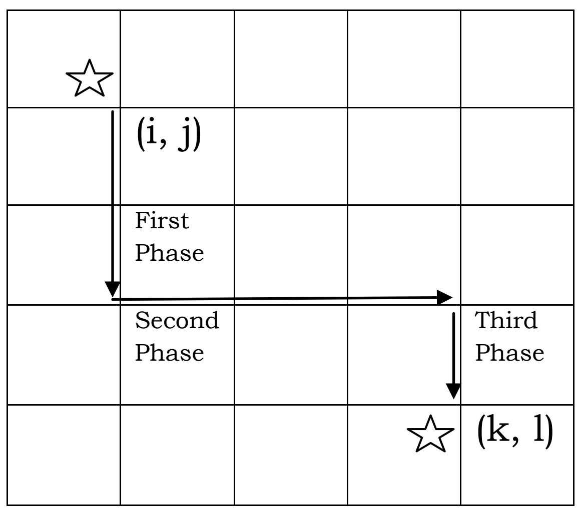
CSE6512 Class Lecture 26

Prepared By: *Subrata Saha*

Lemma: Packet routing in 2-dimentional mesh can be done in 3n + õ(n) steps.

The queue size is $\tilde{O}(\log n)$.

Proof: Consider a 3-phase algorithm as follows:



Consider a packet π whose origin is (i, j) and whose destination is (k, l).

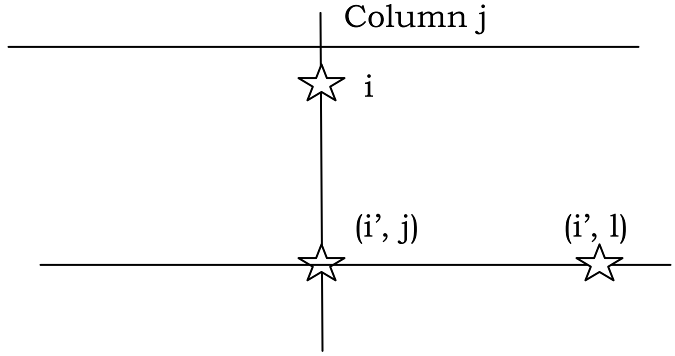
Phase 1: Packet π chooses a random node (i', j) in the column of its origin and goes there.

Phase 2: π traverses along row i' up to (i', l). Furthest origin first Q.D. is used.

Phase 3: π traverses along column l up to (k, l). Furthest destination first Q.D. is employed.

Analysis:

Phase 1: corresponds to Problem 1 (see prev. lecture). The time taken is ≤ n.



Phase 2: Number of packets ending up at (i', j) at the end of phase 1 is B(n, 1/n).

⇒ Total number of packets in the nodes (i', 1), (i', 2), …, (i', j) at the end of the Phase 1 is B(jn, 1/n)

⇒Using Chernoff bounds, this number is (j + õ(j))

⇒Phase 2 corresponds to Problem 3 (see previous lecture).

⇒Runtime of Phase 2 is (n + õ(n)).

Phase 3: In this phase every packet is in its correct column.

⇒Phase 3 corresponds to Problem 2 (see previous lecture).

⇒Phase 3 takes ≤ n steps.

In summary, the total runtime of the algorithm is 3n + õ(n).

<u>Queue Size:</u>

Observation: The queue size in any phase is max {queue size at the beginning, queue size at the end}.

Phase 1: Queue size at the beginning is 1 and at the end is B(n, 1/n)

⇒Queue size in Phase 1 is $\tilde{O}(\log n)$.

Phase 2: At the beginning the queue size is $\tilde{O}(\log n)$. At the end, the queue size is also $\tilde{O}(\log n)$.

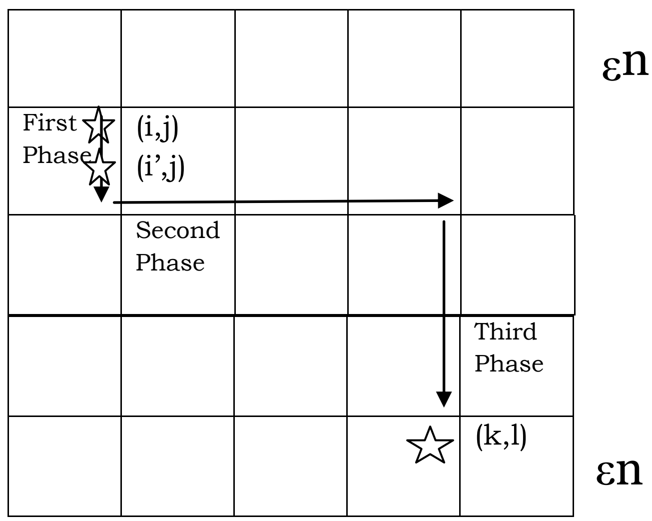
$\Rightarrow$ Queue size in Phase 2 is $\tilde{O}(\log n)$.

Phase 3: At the end, the queue size is 1.

Thus, the queue size for the entire algorithm is $\tilde{O}(\log n)$.

A (2 +ε)n + õ(n) time algorithm for any ε > 0

Description: Partition the mesh into slices of size εn rows each. Consider a packet π whose origin is (i, j) and whose destination is (k, l).



Phase 1: The packet π picks a random node (i', j) in the column of its origin and within its slice of origin.

Phase 2: π traverses along row i' up to column l.

Phase 3: π_traverses along column l up to row k.

Analysis: The number of packets at (i', j) at the end of Phase 1 is B(εn, 1/εn).

$\Rightarrow$ Phase 1 takes ≤ εn steps.

Phase 2: Phase 2 takes (n + õ(n)) steps.

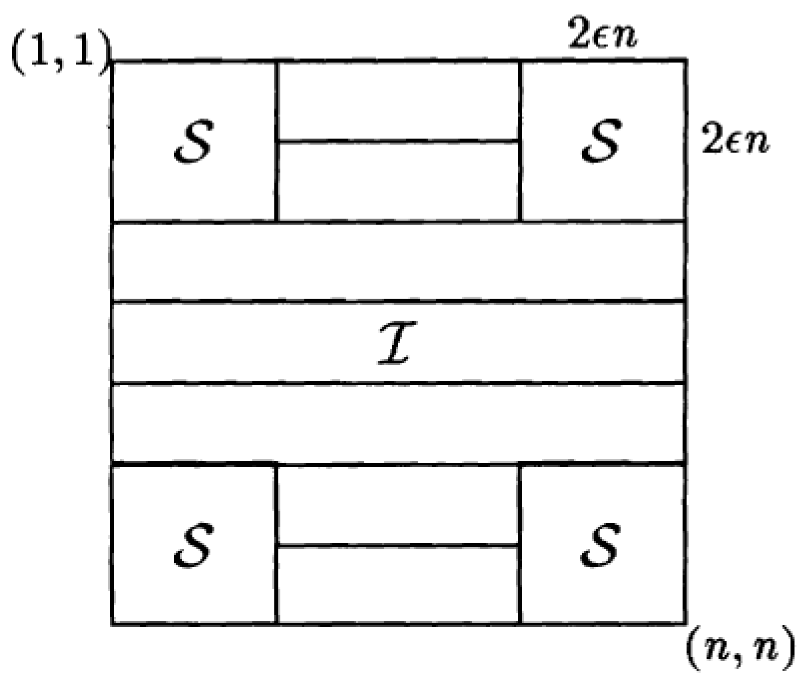
Phase 3: Phase 3 takes ≤ n steps.

Total Runtime = $(2+\varepsilon)n + \tilde{o}(n)$.

Queue Size: It is possible that all the packets destined for some column j could be in the same slice at the beginning. This means that the queue size could be $\geq (1/\varepsilon)$. Thus the queue size of the algorithm is $\tilde{O}(1/e + \log n)$. If a queue size of $\tilde{O}(\log n)$ is desired, then ε has to be $\Omega(1/\log n)$. This in turn will mean that the runtime of the algorithm = $2n + \tilde{O}(n/\log n)$. □

A better Algorithm

A $(2n + \tilde{O}(\log n))$ steps $\tilde{O}(1)$ queue size algorithm has been given by (*Rajasekaran and Tsantilas 1987*). We divide the mesh as follows. Packets originating from the S and I regions are called superior packets and inferior packets, respectively.

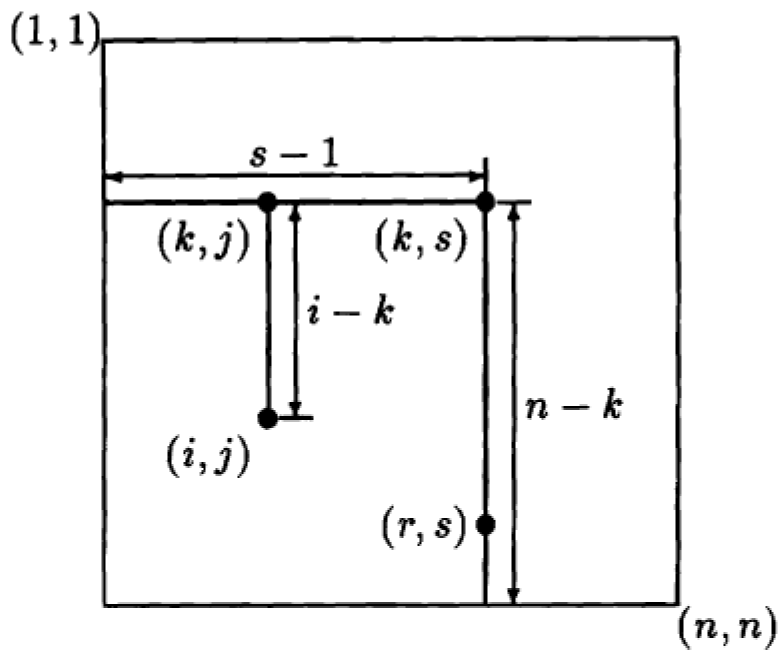


Algorithm for inferior packets:

This is the same as the $(2+\varepsilon)n + \tilde{o}(n)$ time algorithm described above.

Algorithm for superior packets:

Each node (i, j) in the upper two S squares chooses a random k in $(2\varepsilon n + 1, 2\varepsilon n + 2, \ldots, 2\varepsilon n + (1/4)n)$ and sends its packet, q, to (k, j) along column j. If (r, s) is the destination of q, q is then sent to (k, s) along row k and finally along column s to (r, s). Each node (i, j) in the lower two S squares chooses a random b in {n - 2εn - 1, n - 2εn - 2,. . . , n - 2εn - (1/4)n) and sends its packet to (k, j) along column j. The packet is then sent to (k, s) along row k and to (r, s), the packet's destination, along column s.



Queue Disciplines:

- In phases 1 and 3 no distinction is made between superior and inferior packets.
- If a packet doing its phase 1 and a packet doing its phase 3 contend for the same edge, the packet doing its phase 1 takes precedence.
- Furthest destination first priority scheme is used for phase 3.

Shear Sort

The Shear sort is for mesh architectures. A lower bound on the sorting time on a $\sqrt{n} \times \sqrt{n}$ mesh is $\Omega(\sqrt{n})$ since it takes $2(\sqrt{n}-1)$ steps to move a key from the top left corner to the bottom right corner.

The Shear sort uses (log n + 1) phases. In odd phases (1, 3, 5 ...) the following is done:

- Even rows - the row is sorted with the smallest number placed at the right.
- Odd rows - the row is sorted with the smallest number placed at the left.

In even phases (2, 4, 6, ...) the following is done:

- Each column is sorted independently with the smallest number placed at the top.

Sorting of rows and columns requires $\sqrt{n}$ time to complete using for example the odd-even transposition sort. The total time is then $\sqrt{n}$ (log n + 1). There are other mesh sorting algorithms whose asymptotic run times match the lower bound.