

Routing

Application of Chernoff: analysis of load balancing.

- Already saw balls in bins example
- synchronous message passing
- bidirectional links, one message per step
- queues on links
- **permutation** routing
- **oblivious algorithms** only consider self packet.
- **Theorem** Any deterministic oblivious permutation routing requires $\Omega(\sqrt{N/d})$ steps on an N node degree d machine.
 - reason: some edge has lots of paths through it.
 - **homework:** special case

start here

- Hypercube.
 - N nodes, $n = \log_2 N$ dimensions
 - Nn directed edges
 - bit representation
 - natural routing: bit fixing (left to right)
 - paths of length n —lower bound on routing time
 - Nn edges for N length n paths suggest no congestion bound
 - but deterministic bound $\Omega(\sqrt{N/n})$

lecture 6 ended here

- Randomized routing algorithm:
 - $O(n) = O(\log N)$ randomized
 - how? load balance paths.
- First idea: random destination (not permutation!), bit correction
 - Average case, but a good start.
 - $T(e_i) =$ number of paths using e_i
 - by symmetry, all $E[T(e_i)]$ equal
 - expected path length $n/2$

- LOE: expected total path length $Nn/2$
- nN edges in hypercube
- Deduce $E[T(e_i)] = 1/2$
- Chernoff: every edge gets $\leq 3n$ (prob $1 - 1/N$)
- Naive usage:
 - n phases, one per bit
 - $3n$ time per phase
 - $O(n^2)$ total
- What if don't wait for next phase?
 - FIFO queuing
 - total time is length plus **delay**
 - Expected delay $\leq E[\sum T(e_l)] = n/2$.
 - Chernoff bound? no. dependence of $T(e_i)$.
- High prob. bound:
 - consider paths sharing i 's fixed route (e_0, \dots, e_k)
 - Suppose S packets intersect route (use at least one of e_r)
 - claim delay $\leq |S|$
 - Suppose true, and let $H_{ij} = 1$ if j hits i 's (fixed) route.
$$\begin{aligned}
 E[\text{delay}] &\leq E[\sum H_{ij}] \\
 &\leq E[\sum T(e_l)] \\
 &\leq n/2
 \end{aligned}$$
 - Now Chernoff **does** apply (H_{ij} independent conditioned on fixed i -route).
 - $|S| = O(n)$ w.p. $1 - 2^{-5n}$, so $O(n)$ delay for all 2^n paths.
- Lag argument
 - Exercise: once packets separate, don't rejoin
 - Route for i is $\rho_i = (e_1, \dots, e_k)$
 - charge each delay to a departure of a packet from ρ_i .
 - Packet waiting to follow e_j at time t has: **Lag** $t - j$
 - Delay of i is lag crossing e_k
 - When i delay rises to $l + 1$, some packet from S has lag l (since crosses e_j instead of i).

- Consider last time t' where a lag- l packet exists on path
 - * some lag- l packet w crosses $e_{j'}$ at t' (others increase to lag- $(l + 1)$)
 - * w leaves (next edge is off path) at this point (if not, then l at $e_{j'+1}$ next time)
 - * charge one delay to w .
- Worst case destinations
 - Idea [Valiant-Brebner] From intermediate random destination, route back!
 - routes **any** permutation in $O(n^2)$ expected time.
 - what's going in with $\sqrt{N/n}$ lower bound?
 - Adversary doesn't know our routing so cannot plan worst permutation
 - We are hedging: double typical delivery time, but eliminate worst case

Similar analysis for routing on butterfly.