Lecture 3

1 Hypercube/Benes Routing

1.1 Offline Path Selection

The idea is to represent the network with a graph, $G = (V, E)$

Definition 1 Quality *can be thought of as two separate metrics:*

1. *Congestion* (C) = *max # paths that use any edge.*
2. *Dilation* (D) = *max length of any path.*

Definition 2 *Path Selection Problem:*
Given a graph $G = (V, E)$ *and* $(s_i, t_i) \ldots (s_k, t_k)$ *choose a set of paths connecting* s_i, t_i*, such that congestion and dilation are minimized.*

Example 1 : *Benes Routing on a Benes Network*
If s_i is an input and t_i is an output and all $t_i \neq t_j, s_i \neq s_j$, $\exists$ a congestion of 1 solution. Conceptually, the network has enough redundancy to route each packet "around" the others.

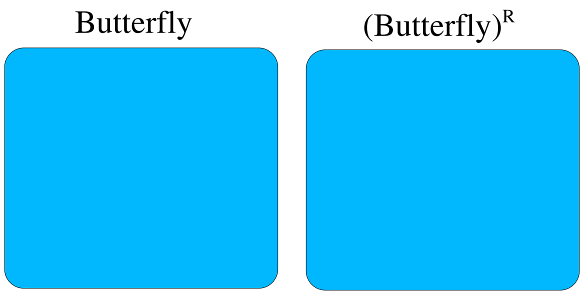

Figure 1: High-level diagram of a Benes network

Example 2 : *Bit Fixing in a Butterfly Network*
For a butterfly network, $\exists$ a permutation such that routing costs $\Omega(\sqrt{N})$. We refer to bit fixing as an *oblivious* routing scheme meaning that behavior of one packet doesn't depend on other packets (local decision).

Example 3 : *Single Destination, Source-Oblivious Routing Scheme*
Consider some "deterministic (as in non-randomized) and oblivious" routing scheme. For some network and a source-oblivious routing scheme, routes form a fixed tree leading to each of the destinations.

The routing cost is $\Omega(\frac{\sqrt{N}}{\sqrt{d}})$ where d is the degree of the network. This is not a good bound ($\Theta(\log N)$ is considered "good"). What to do? Randomize to improve performance.

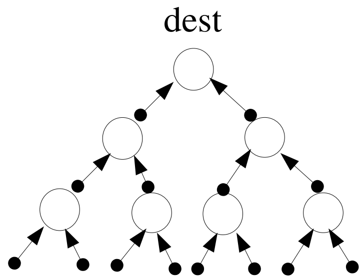

Figure 2: Source-oblivious routing scheme

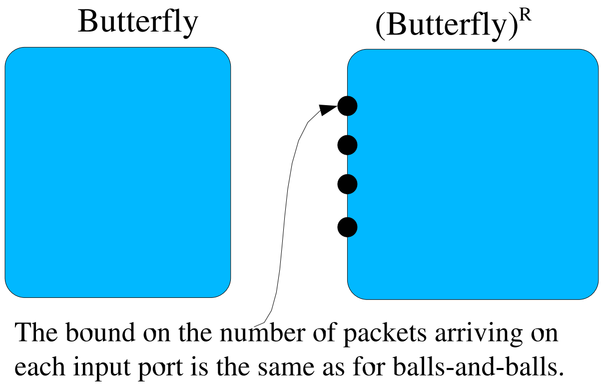

Figure 3: Load on intermediate inputs in a Benes network

1.2 Randomized Path Selection Strategy

For a Benes network, select a random intermediate destination. From the ball-and-bins result from one of the previous lectures, we know that the load on each one of the intermediate destinations will be $\Theta(\frac{\log N}{\log \log N})$ w.h.p. (see Figure 3). Assume for the time being that scheduling is greedy. This implies that $O(C * D)$ and $\Omega(C + D)$. For simplicity's sake, consider $\frac{1}{2}$ of the network. The second half is just the first half backwards.

Algorithm Outline

1. Randomize which intermediate destination we route to
2. Route greedily through the second half of the network

1.3 Process

Let $X_1, \ldots, X_p$ be random variables such that

$$X_i = \begin{cases} 1 & \text{packet } i \text{ hits edge } e, \\ 0 & \text{otherwise} \end{cases} \tag{1}$$

$$X = \Sigma X_i \tag{2}$$

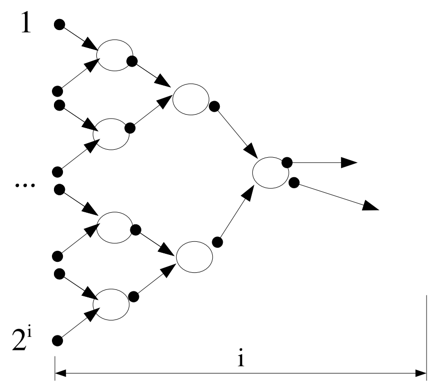

Figure 4: Expectation for edge congestion for a binary tree

by linearity of expectation

$$\mathbf{E}[X] = \frac{1}{2} \tag{3}$$
$$\textit{total path length} = N \log N \tag{4}$$
$$\textit{total edges} = 2N \log N \tag{5}$$

For 2^i packets:

$$\mathbf{E}[X_p] = \begin{cases} \frac{1}{2^{i+1}} & \text{for } 2^i \text{ packets that could go there,} \\ 0 & \text{otherwise} \end{cases} \tag{6}$$

Thus for all 2^i packets, $\mathbf{E}[X] = 2^i\left(\frac{1}{2^{i+1}}\right) = \frac{1}{2}$.

1.3.1 Chernoff Bound

$$\mathbf{Pr}[X \geq (1+\sigma)\mu] \leq \left(\frac{1}{2}\right)^{\sigma\mu} \tag{7}$$

Choose $\sigma = 4 \log N$

$$\mathbf{Pr}[X \geq 2\log N] \leq \left(\frac{1}{2}\right)^{2\log N} \leq \frac{1}{N^2} \tag{8}$$
$$\mathbf{Pr}[\textit{any edge has copy} \geq 2\log N] \leq \frac{N \log N}{N^2} \tag{9}$$

It follows that we can route a permutation in $O(\log^2 N)$ time.

2 Delay Sequences

Definition 3 : *Path Congestion*
For any path in the network, path congestion (PC) is the total # of packets that hit a path.

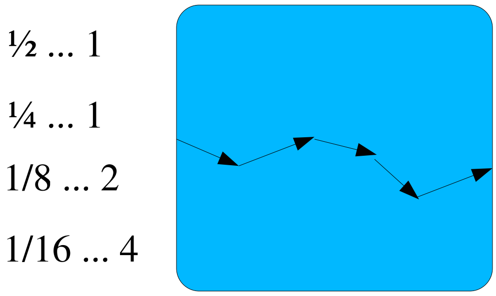

Figure 5: Expectation for a butterfly network

Consider a packet that arrives at time t. Construct a delay sequence for a sequence of edges from destination to source.

$$t \begin{cases} (p_{t-1}, e_{t-1}) & \text{crossed} \\ \dots & \\ (p_i, e_i) & \text{crossed} \\ (p_{i-1}, e_{i-1}) & \text{if I waited, } p_{i-1} \text{ to be packet I waited for } e_{i-1} = e_i \end{cases} \tag{10}$$

If length of the path is D, then at least $t - D$ packets use the path in the delay sequence.

LEMMA 1
$t \leq PC + D$ and $PC \geq t - D$

DEFINITION 4 *: Combinatorial Definition of Delay Sequence*
A delay sequence is a sequence of pairs (p_i, e_i), such that

1. *if $p_i = p_{i+1}, e_i \neq e_{i+1}$*
2. *if $p_i \neq p_{i+1}, e_i = e_{i+1}$*
3. *$e_1, \dots, e_i$ define a path in the network.*

If we have a butterfly network, the expectation for the number of packets hitting a particular edge is $\mathbf{E}[X] = \frac{\log N}{4}$ (see Figure 5. Assuming $\sigma \leq (2e - 1)$, $\mathbf{Pr}[X \geq (1 + \sigma)\mu] \leq e^{-\frac{\sigma^2 \mu}{3}} \leq \frac{1}{N^2}$.

Thus a permutation can be routed in $O(PC + D)$ and $\Theta(\log N)$.

$$\textit{total work} = N \log N \tag{11}$$
$$\textit{total resources} = N \log N * T = N \log^2 N \tag{12}$$

Route $\log N$ packets from every input instead of 1:

$$T \sim \Theta(\log N) \tag{13}$$
$$C = O(\log N) \tag{14}$$
$$D = \log N \tag{15}$$
$$T = \Omega(\log^2 N) \qquad (???) \tag{16}$$

3 Priority Based Routing

Choose random priorities for packets and use priority routing *i.e.*, if $p_i \neq p_j$ both are at node and want to use e and $R(p_i) \geq R(p_j)$ send p_i.

DEFINITION 5 *Combinatorial Definition of Prioritized Delay Sequence*
A delay sequence is a sequence of pairs (p_i, e_i), *such that*

1. *if* $p_i = p_{i+1}, e_i \neq e_{i+1}$
2. *if* $p_i \neq p_{i+1}, e_i = e_{i+1}$ *and* $R(p_i) \geq R(p_{i+1})$
3. $e_1, \ldots, e_i$ *define a path in the network.*

Arrival time t implies a delay sequence of length t. The probability that a particular sequence is sorted is:

$$\mathbf{Pr} = \frac{1}{(t-D)!} \tag{17}$$

The number of delay sequences is:

$$(\#\ outputs)(2^D)(C)^{t-D}\binom{t}{D} \tag{18}$$