Mathematics for Computer Science - notes

Angjoo Kanazawa

July 19, 2011

8 Chapter 8 Communication Networks

8.1 Complete Binary Tree

Definitions/formulas:

- **Switches** direct packets through the network (circle nodes)
- **Latency** is the time required for a packet to travel from an input to an output.
- The **diameter** of a network is the number of switches on the shortest path between the input and output that are farthest apart. Approxmiates the worst-case latency.
- The diameter of a complete binary tree with N in/outputs is $2logN + 1$. Not bad, even with 2^{10} terminals, latency is $2log(2^{10}) + 1 = 21$
- Total number of switches in a complete binary tree is $2N - 1$.
- A **permutation** is a bijective function $\pi : 0, 1, \ldots, N-1 \to 0, 1, \ldots, N-1$
- $\forall\pi\exists$ **permutation routing problem**, where the challenge is to direct a packet starting at input i to output $\pi(i)$
  - Solution: specification fo the path taken for all N packets. Where the path of packet i to output $\pi(i)$ is denoted $P_{i,\pi(i)}$.
  - The **congestion** of a set of paths $P_{0,\pi(0)}, \ldots, P_{N-1,\pi(N-1)}$, is the largest number of paths that pass through a single switch. i.e for $\pi(i) = i$, the congestion is 1, for $\pi(i) = (N-1) - i$, the congestion is 4. Lower the congestion, the better the set of paths.
  - The **Max congestion** of a network is "maximum over all permutations π of the minimum over all paths' congestion... $\arg\max_\pi \arg\min_{\text{all possible paths}} Cong(P_{i,\pi(i)})$ where $Cong$ is the congestion of a given path?
  - The max congestion of a complete binary tree is N with $\pi(i) = (N-1) - i$, because with that π, every packet needs to go through the root. You can't do better and its horrible.
  - over all permutation.. $= N!$ permutations for N in/out networks. For each of those permutations, there are paths that takes i to $\pi(i)$ = a lot of paths, but chose the BEST congestion. And the max of those out of $N!$ perm is the max congestion..

Goals:

- larger the switches $\to$ smaller diameter. Most nodes in a binary tree takes 3 in/out edges =¿ 3x3 switches
- Want: how to get NxN monster switch using 3x3 simple switches

8.2 2-D Array/Grid/Crossbar

- Diameter of an array with N in/outputs is $2N - 1$.
- Each switch takes two in/out edges, so switch size is $2x2$.
- # of switches is the number of elem in NxN array $= N^2$

Theorem 8.2.1. *The (max) congestion of an N-input array is 2*

Proof. It's at most 2: Let π be any permutation. Let $P_{i,\pi(i)}$ to be the path fro input i rightward to col j, downward to output $\pi(i)$ (can go other ways but this is the best path), so the (i, j)th switch transmits at most 2 packets. The one coming from left and the one coming from the top..?
It's at least 2: With $\pi(0) = 0$ and $\pi(N - 1) = N - 1$, the packet in the lower left corner must pass two packets. (It's at least so just show it exists). □

8.3 Butterfly

- All terminals and switches are in N rows, where inputs, ordered, are the first col, output, ordered, are the last col. Now label the rows in binary, so row i has binary number $b_1 b_2 \cdots b_{logN}$ that represents i.
- $\exists\ log(N) + 1$ levels of switches (nodes), numbered from 0 to $logN$. Each level is a column of N switches. So every switch has a unique sequence $(b_1, b_2, \ldots, b_{logN}, l)$, where $b_1 b_2 \cdots b_{logN}$ is the switch's row label and l is the level of the switch $(0 \to logN)$.
- So... $Nby1 \to Nbylog(N) + 1 \to Nby1$
- Connection: there are directed edges from $(b_1, b_2, \ldots, b_{logN}, l)$ to two switches in the next level, one in the same row, one in the row with label $l + 1$ (inverting bit $l + 1$).
- Recursive structure. A butterfly of size $2N$ is made up of two N butterflies plus one more level of switches.

There is only one path from an input to an output, where the path is by correcting each successive bit with the bit of the output.

Corollary 8.3.1. *The congestion of the butterfly network is exactly $\sqrt{N}$ when N is an even power of 2:*

Proof. Let B_n denote the butterfly network with $N = 2^n$ inputs and N outputs.

For B_n, there is a unique path from each input to each output, so congestion is the max number of transmission for a vertex. (The number of total vertices is $N(log(N) + 1)$, or $2^n(n + 1)$... useless)

For every vertex v at level i, there's a path from exactly 2^i input vertices to v and exactly 2^{n-i} output vertices (where n is the power to 2).

Since there is a unique path from each input to output, the number of messages that passes through a vertex is at most the minimum of number of input or output vertices it has a path from/to.

So congestion must be worst at the center level of the network. i.e for $n = 3$, level 0 vertices have a path from 1 input vertex and have a path to 8 output vertices, level 1 has 2-from and 4-to, level 2 has 4-from and 2-to, and the last level has 8-from and 1-to. Now there are $n + 1$ levels (or $log(N) + 1$). So with even n, there exists a center level with vertices that have a path from exactly $2^{(n/2)}$ input vertices, and a path to exactly $2^{n/2}$ output vertices.

Which means that congestion of vertices at the middle level is at most $\sqrt{N} = 2^{n/2}$.

... not clear which one is at least and at most.. It's like.. at most and at least because in the middle its $\sqrt{N}$ from and $\sqrt{N}$..

□