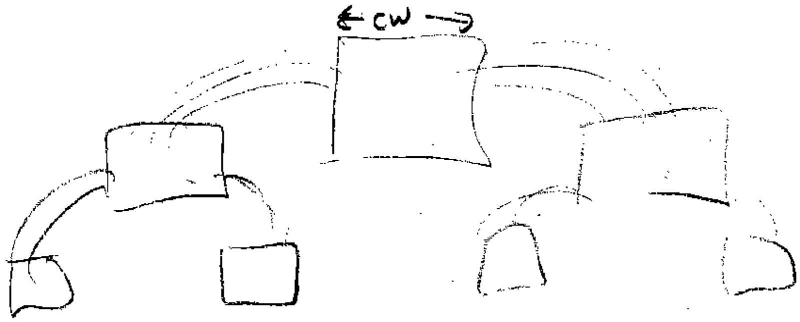
Recall from last time:

- Truncated TOM(n, k) has area $O(n^2k^2)$.
- TOM(n) has area $O(n^2 \lg^2 n)$
- If G has a (w, α) decomp tree, it has a $(O(w), \alpha)$ balanced decomp tree.

Theorem Every N-node graph with a $(w, \sqrt{2})$ decomp tree can be laid out in $O(w^2 \lg^2(N/w))$ area.

Pf. Get an $(O(w), \sqrt{2})$ balanced decomp tree for G,

Embed G in TOM$(cw, 2\lg(N/w))$ for some const c:

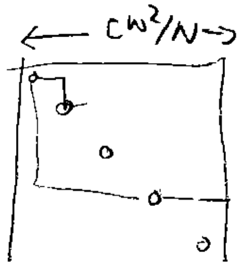


Leaf meshes: $\#\text{nodes} = \dfrac{(cw)^2}{2^{2\lg(N/w)}} = \dfrac{c^2 w^2}{N^2/w^2} = \dfrac{c^2 w^4}{N^2}$

side length $= \dfrac{cw^2}{N}$.

$\#$vertices in leaf meshes $= \dfrac{N}{2^{2\lg(N/w)}} = \dfrac{N}{N^2/w^2} = \dfrac{w^2}{N}$

$\#$ edges leaving leaf mesh $= \dfrac{O(w)}{(\sqrt{2})^{2\lg(N/w)}} = \dfrac{O(w)}{N/w} = O\left(\dfrac{w^2}{N}\right)$



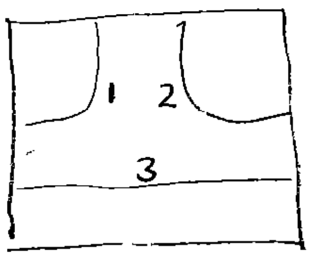
By adjusting c, can route edges within mesh + room on perimeter for wires to escape.

At depth p, side length of mesh is $\geq \frac{cw}{2^{\lceil p/2 \rceil}}$.

#edges leaving $= O(w)/(\sqrt{2})^{p-1} = O(w)/2^{p/2}$.

Adjust c for adequate capacity.

Routing internal-node meshes:



Type 1 & 2: 2 layers each.
Type 3: 3 layers

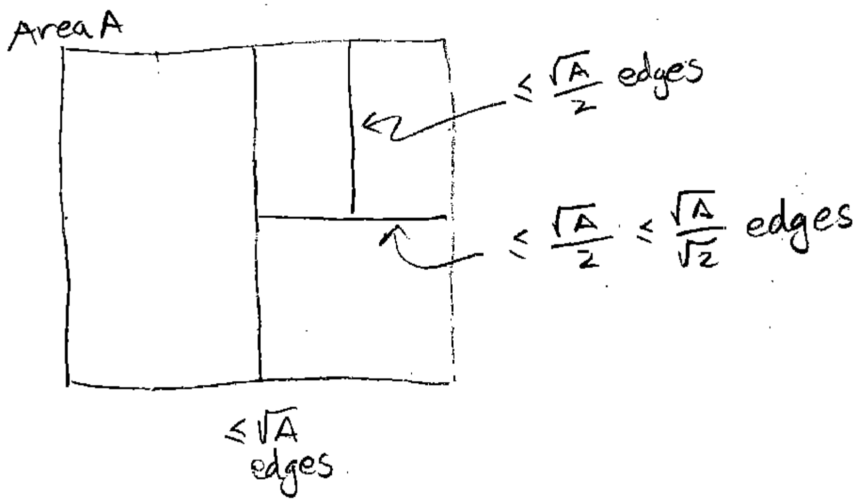
$\therefore$ 7 layers (squash to 2 if desired).

Area of TOM$(cw, 2\lg(N/w))$ is

$$O((cw)^2(2\lg(N/w))^2) = O(w^2 \lg^2(N/w)) \quad \boxtimes$$

<u>Corollary</u>. Let w be smallest value for a $(w, \sqrt{2})$ decomp tree for N-node graph G. Let A be min area. Then, $w^2 \leq A \leq O(w^2 \lg^2(N/w))$.

Pf.



#edges leaving subgraph at depth p
$\leq \frac{\sqrt{A}}{\sqrt{2}^{p-1}}$

$\therefore G$ has $(\sqrt{A}, \sqrt{2})$ decomp tree $\Rightarrow w \leq \sqrt{A}$. $\boxtimes$

Which network is best?

Network	Area	Routing Time	AT^2
Linear array	N	N	N^3
2D array	N	$\sqrt{N}$	N^2
3D array	$N^{4/3}$	$N^{1/3}$	N^2
CBT	N	N	N^3
hypercube	N^2	$\lg N$	$N^2 \lg^2 N$
butterfly	$N^2/\lg^2 N$	$\lg N$	N^2
2D MOT	$N \lg^2 N$	$\sqrt{N}$	$N^2 \lg^2 N$

Universality: An N-node butterfly can simulate any other N-node bounded-degree network with $O(\lg N)$ slowdown, just by routing messages.

Universal = expensive?

VLSI perspective: normalize to area, not # procs.

Area A network. Route A packets
- 2D array: $N = A$
  Route A packets in $\sqrt{A}$ time
- Butterfly: $N = \sqrt{A}\lg A$ $(\lg A \sim \lg N)$
  Route $\underset{\sqrt{A}\lg A}{N}$ packets in $\underset{\lg A}{\lg N}$ time

$\sqrt{A}/\lg A$ batches of $\sqrt{A}\lg A$ packets, each taking $\lg A$ time. Total time $= \sqrt{A}/\lg A \times \lg A = \sqrt{A}$.

Same! (Reason: basically since $AT^2 = N^2$ for both)

How can we compare? Ans. Simulate
- Can an area-A butterfly simulate any other area-A network efficiently? ($O(\lg A)$ slowdown)

$\rightarrow$ Can't even do linear array.

procs in butterfly $= \sqrt{A}\lg A$
procs in lin. array $= A$
Slowdown $= A/\sqrt{A}\lg A = \sqrt{A}/\lg A$

6.896
5/3/04
L21.4

1D array can't sim 2D array (diam)
2D array can't sim CBT (diam)
CBT can't sim 2D array (bis. width)
2D MOT can sim others with $\lg^2 A$ slowdown.

Next time: "Area-universal" networks.
Idea: physical structure is TOM, but low diam.
"Fat-trees"

«Reminder: catch up on reading for final»