**VLSI lower bounds**

*Lemma.* A network with bisection width $B$ has area $\Omega(B^2)$.  

\[
Pf. \quad B \leq W + 1 \implies A = LW \geq W^2 \geq (B - 1)^2 = \Omega(B^2)
\]

Good recursive bisection $\implies$ small area

**Bisection width lower bounds on computation**

**Shifting.**

- **Input:** $x_0, x_1, \ldots, x_n$ data
- $s \leq 0, \ldots, n-1$ control

- **Output:** $y_0, y_1, \ldots, y_n-1$ & $y_n = (x_0 - s) \mod n$

<table>
<thead>
<tr>
<th>Network</th>
<th>$S(n)$</th>
<th>$B(n)$</th>
<th>$A(n)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Linear array</td>
<td>$O(1)$</td>
<td>$O(n)$</td>
<td>$\Theta(n)$</td>
</tr>
<tr>
<td>Tree</td>
<td>$O(n)$</td>
<td>$\Theta(1)$</td>
<td>$\Theta(n)$</td>
</tr>
<tr>
<td>$\sqrt{n}$ mesh</td>
<td>$O(\sqrt{n})$</td>
<td>$\Theta(n)$</td>
<td>$\Theta(n)$</td>
</tr>
<tr>
<td>$n$-input butterfly</td>
<td>$O(\log n)$</td>
<td>$G(n)$</td>
<td>$\Theta(n^2)$</td>
</tr>
</tbody>
</table>

**Let $B(n) =$ # edges cut to bisection outputs**

If $T(n) =$ worst-case #bits to cross this bisection, then $B(n)T(n) \geq I(n)$.  

**Ex:**  

- $S = 0 \implies 1$ bit crosses bisection ($L \to R$)
- $S = 3 \implies 3$ bits cross - worst case $\implies I = 3, T \geq B/I = 3/2$.  

**MIT LCS**
How do we know 3 bits must cross? Might there be a clever encoding?

"Fooling argument"

Ex. Sup. we have following bisection:

\[
\begin{array}{c|c|c|c}
X_0 & Y_0 & X_{n/2} & Y_{n/2} \\
X_1 & Y_1 & \quad & \quad \\
\vdots & \vdots & \quad & \quad \\
X_{n/2-1} & Y_{n/2-1} & \quad & \quad \\
X_n & Y_n & \quad & \quad
\end{array}
\]

\[S = n/2 \Rightarrow \text{intuitively } n/2 \text{ bits must cross (L to R)}\]

Claim: \( B(n) \cdot T(n) \geq n/2 \)

PF. (fooling arg.) Sup. \( B(n) \cdot T(n) < n/2 \)

\# communication patterns on \( B(n) \) wires (L to R) over time \( T(n) = 2^{B(n)T(n)} < 2^{n/2} \)

\# values for \( x_0, \ldots, x_{n/2-1} = 2^{n/2} \)

\[\exists 2 \text{ distinct } x'_0, \ldots, x'_{n/2-1} \text{ and } x''_0, \ldots, x''_{n/2-1} \text{ that produce identical comm. patterns.}\]

RHS of circuit can't distinguish \( \Rightarrow \) produces same values for \( x_{n/2}, \ldots, x_n \) for both \( \Rightarrow \) must operate wrong for one, contradiction. \( \Box \)
Thm. For any bisection of outputs, $B(n) T(n) \geq n/2$.

Pf. Consider an arbitrary bisection.

Ex. $Y_0 \ x_0 \ | \ Y_1 \ x_3$
    $Y_2 \ x_1 \ | \ Y_3$
    $Y_4 \ x_2 \ | \ Y_5$
    $x_4

Make an $n \times n$ table:

|   | $x_0$ | $x_1$ | $x_2$ | $x_3$ | $x_4$ | $x_5$
|---|---|---|---|---|---|---
| 0 | 0   | 1   | 1   | 1   | 0   | 0   |
| 1 | 1   | 1   | 1   | 1   | 1   | 1   |
| 2 | 0   | 0   | 1   | 1   | 0   | 0   |
| 3 | 1   | 1   | 1   | 1   | 1   | 1   |
| 4 | 0   | 0   | 1   | 1   | 0   | 0   |
| 5 | 1   | 1   | 1   | 1   | 1   | 1   |

X if shift of s causes $x_i$ to cross bisection.

Every column contains $n/2$ X's.

\[ \text{Average } \# \text{ X's per row } \geq n/2, \]

\[ \Rightarrow \text{some row contains } 2n/2 \text{ X's.} \]

(some shift causes $n/2$ bits to cross). $\blacksquare$

<table>
<thead>
<tr>
<th>Network</th>
<th>$B(n)$</th>
<th>$T(n)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Linear array</td>
<td>$\Theta(1)$</td>
<td>$n(n)$</td>
</tr>
<tr>
<td>Tree</td>
<td>$\Theta(1)$</td>
<td>$\Omega(n)$</td>
</tr>
<tr>
<td>$\sqrt{n} \times \sqrt{n}$ mesh</td>
<td>$\Theta(1)$</td>
<td>$\Omega(n)$</td>
</tr>
<tr>
<td>$n$-input butterfly</td>
<td>$\Theta(n)$</td>
<td>$\Omega(1)$</td>
</tr>
</tbody>
</table>
**Theorem** Any circuit for shifting \( n \) 6-bit inputs has \( A T^2 = \Omega(n^2) \).

**Proof:** \( A T^2 = \Omega(n^2) \).

**Theorem** Any circuit for multiplying two \( n \)-bit numbers has \( A T^2 = \Omega(n^2) \).

**Proof:** Let \( b \) take on powers of 2. Essentially correspond to shift problem.

\[ C \leftarrow \overset{2n}{\ldots} \overset{\text{outputs}}{\text{c}} \overset{1}{\ldots} \]

\[ A \leftarrow \overset{n}{\text{inputs}} \overset{1}{\ldots} \]

Consider bisection of outputs. Build shift/input matrix as before: \( n \times n/2 \)

\[
\begin{array}{cccc}
0 & a_1 & a_2 & \ldots & a_n \\
1 & x & x & x & \ldots \\
s & x & x & x & \ldots \\
n-1 & x & x & x & \ldots \\
\end{array}
\]

\( n/4 \) X's per column

\( n^2/8 \) X's in matrix

\( \Rightarrow \) some row has \( n/8 \) X's.

\[ \therefore B \geq \Omega(n) \]

Also, FFT, convolution, sorting, routing, etc.

General Layout Strategy

Tree of meshes (not mesh of trees)

\[ \text{TOM}(n) : \]

\[ N = n^2 \lg n \]

\[ n^2 \text{ leaves} \]

Area:

\[ S(n) = 2 S(n/2) + n \]

\[ = \Theta(n \lg^2 n) \]

\[ A(n) = \Theta(n^2 \lg^2 n) \]
Fold and squash:

\[ n^2 \times \log n \text{ layers} \Rightarrow \Theta(n^2 \log^2 n) \text{ area.} \]

\text{Truncated TOM: } \text{TOM}(n,k) - \text{top k levels}
\[ \text{Area} = \Theta(n^2 k^2) \]

\text{Decomposition trees}

\( T \) is a \((w_1, w_2, \ldots, w_r)\) decomposition tree for \( G = (V, E) \):

1. Vertices in \( V \) mapped to leaves of \( T \).
2. Edges in \( E \) run through links of \( T \).
3. # edges leaving subtree rooted at depth \( i \) is \( \leq w_i \)

For \( 1 \leq \alpha \leq 2 \), \( G \) has a \((w, \alpha)\) decomp tree if it has a \((w, w/\alpha, w/\alpha^2, \ldots, 0!0)\) decomp tree.

A decomp tree is balanced if all subgraphs at the same depth have same # vertices up to within 1.
**Layout strategy**

1. Start with \((w, \sqrt{2})\) decomp tree.
2. Balance the decomp tree.
3. Embed the balanced tree in trunc TCM.
4. Use trunc TCM layout to yield \(O(w^2 \log^2 n)\) area layout.

**Balancing decomp trees**

Warm-up: Necklace with black and white pearls.
How many cuts to divide into 2 sets, each with half the pearls of each color?

2 cuts suffice.
Continuity argument.

**Lemma:** Consider any 2 strings composed of an even # of black pearls and an even # of white pearls. By making at most 2 cuts, the pearls can be partitioned into 2 sets, each containing 2 strings such that each set has 1/2 the pearls of each color.

**Pt. (Continuity arg.)**
Lemma. Let $T$ be a CBT drawn with $n$ leaves on a straight line, and consider any set $S$ of $k$ consecutive leaves of $T$. Then, if a forest $F$ of complete binary subtrees of $T$ is

1. $S = \{\text{leaves of } F \}$
2. at most 2 trees of $F$ have any given height.
3. depth of largest tree in $F$ is $\leq \log k$

\[ \text{Pf. } F \text{ be forest of maximal CBT's whose leaves lie only in } S. \text{ (1) and (3) follow. Use induction to prove (2).} \]

Theorem. Let $G$ be a graph on $n$ vertices that has a $(w_1, w_2, \ldots, w_r)$ decomposable tree $T$. Then, $G$ has a $(w'_1, w'_2, \ldots, w'_{r+1})$ balanced decomposable tree $T'$, where

\[ w'_i = 4 \sum_{k=i}^{r} w_k. \]

\[ \text{Pf. Color leaves of } T: 1 \text{ = node of } G, 0 \text{ = empty.} \]

Recursively split $B \& W$ leaves evenly. Each stage has $\leq 2$ strings of consecutive leaves from $T$, each of which has $\leq 2$ CBT's of a given height.
Total # wires leaving a string ≤ sum of wires leaving each of its cbts.

\[ w_i \leq \frac{4}{\alpha} \sum_{k=0}^{r} w_k. \]

**Corollary** A graph with a \((w, \alpha)\) decomp tree, \(\alpha\) const, has an \(O(w, \alpha)\) balanced decomp tree.

**Proof** Sum is geometric:

\[ w'_i = \frac{4}{\alpha} \sum_{k=0}^{r} w_k \]

\[ \leq \frac{4}{\alpha} \sum_{k=0}^{r} \frac{w}{\alpha^{k-1}} \]

\[ \leq \frac{4w}{\alpha^{\alpha-1} (\alpha-1)} \]

Graph has \((4w\alpha/(\alpha-1), \alpha)\) decomp tree.  □


**Exam issues**.