Systolic computation

E.g., linear array \( I/0 - 0 - 0 - 0 - 0 \ldots - 0 \)

"Fixed-connection" network
1. Underlying graph fixed
2. Local communication only
3. I/O location restricted

At each step of a globally synchronous clock,
each processor:
1. receives inputs from neighbors (or I/O)
2. inspects local memory
3. performs local computation
4. updates local memory
5. generates outputs for neighbors

Example: Sorting

- Accept left input
- Compare input to stored value
- Store smaller value
- Output bigger \# to right.
Correctness: induction

N inputs. How many steps? 2N = \Theta(N)

"Discuss outputting of values."

Total time = 3N

Sorting in the bit model (vs. word model)

- One processor per bit.

N k-bit #’s

\[
\begin{array}{cccccc}
3 & 5 & 1 & 2 & 3 \\
0 & 0 & 0 & 0 & 0 \\
1 & 0 & 0 & 1 & 1 \\
1 & 1 & 1 & 0 & 1 \\
\end{array}
\]
Comparing two \( k \)-bit words

\[
\begin{align*}
1/1 &= \Theta(k) \text{ steps to compute} \\
0/0 &= \text{Sort in } \Theta(Nk) \text{ time}
\end{align*}
\]

Faster comparison - binary tree

\[
\begin{align*}
1/1 &= \Theta(\lg k) \text{ compare} \\
0/0 &= \text{Sort in } \Theta(N\lg k) \\
0/1 &< \quad \text{lg} = \log_2
\end{align*}
\]
Pipelining

- compare while sorting
- stagger bits of input

Each processor:

\[
\begin{align*}
0/1 & \rightarrow 0/1 \\
<,>,= & \\
0/1 & \rightarrow 0/1
\end{align*}
\]

Time = \(\Theta(N+k)\) bit steps.
Can we do better on \(N \times k\) array?

Lower bounds

1. I/O bandwidth
   \[\text{Nk bits to input at k places} \Rightarrow \Omega(N)\text{ steps}\]

2. Network diameter
   \[\Omega(N+k)\]

3. Communication bandwidth (bisection width)
   \[T \geq \frac{\#\text{bits crossing cut}}{\text{size of cut}}\]
   \[T \geq \frac{\Theta(Nk)}{\Theta(k)} = \Theta(N)\]
Problem: \( N \) 1-bit #'s input at \( N \) leaves of complete binary tree. Time to sort?

I/O: \( T \geq N \) / \( N = 1 \)

Diam: \( T \geq 2 \lg N \)

BW: \( T \geq \Theta(N)/1 = \Theta(N) \)

\[ \therefore \text{Sorting } N \text{ 1-bit #'s takes } \Omega(N) \text{ time on CBT.} \]

Wrong! Can sort in \( O(\lg N) \) time!

Idea: Only need to count # 0's.

Input: 1 0 1 1 0 0 1 1

Output: 0 0 0 1 1 1 1

Sum 0's upward, select downward.

1-bit summer

1. LSB\((x+y+z)\) = parity
2. MSB\((x+y+z)\) = majority

Q: Why doesn't BW lower bound hold?
A: Didn't really need to ship \( \Theta(N) \) bits across bisection. Could encode into more compactly. Careful!