VLSI Models

Mainly from and/or inspired by Chazelle and Monier,
"A Model of Computation for VLSI with Related Complexity Results"

VLSI Chips

Top-Down view

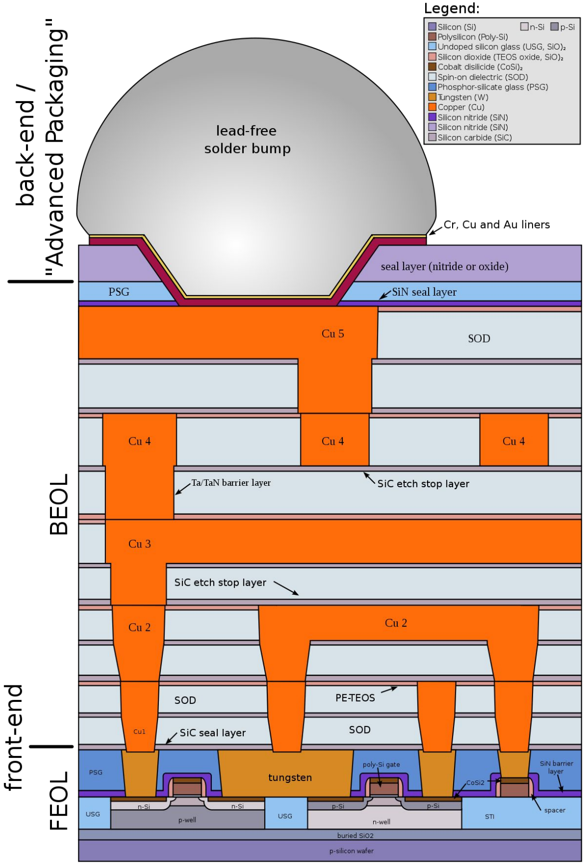
√A

Computation goes here

Area A

√A

Side view



Connection to other things

Constant number of layers for wires

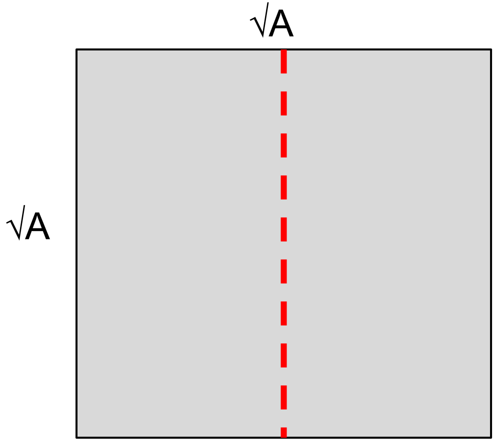
All computation and memory in 2d plane at bottom

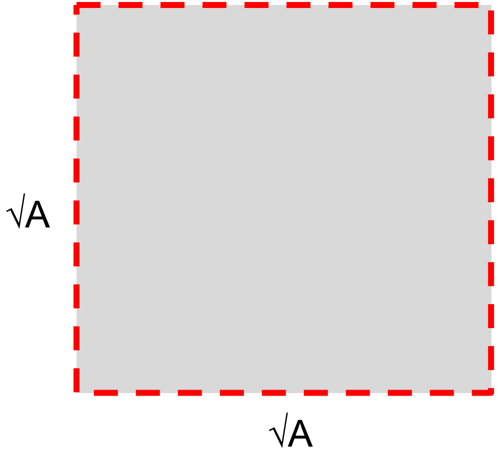
"Standard" assumptions

- Each bit of memory takes at least constant area; each individual bit of computation takes at least constant area and constant time
- A boundary of length L has at most O(L) wires across it (and thus at most O(L) bandwidth for communication)
- Communicating a distance of d takes Ω(d) time (this was controversial in the 80s)

And sometimes:

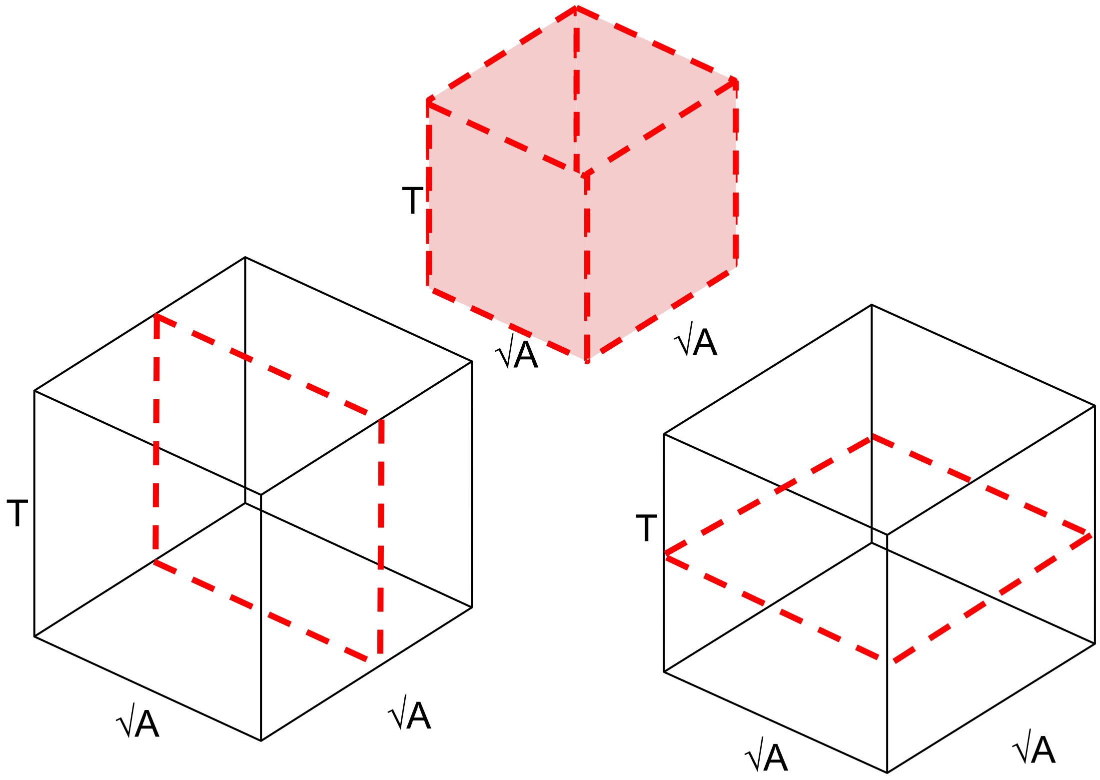
- All inputs and outputs must pass through the chip perimeter (4√A) (no longer realistic for a single chip, but may be relevant for a package?)





Adding time and getting bounds

Consider a computation (hyper)graph of N nodes, with n I/O values and minimum balanced cut of size k



1. All nodes must be mapped to some unique location and time: $AT \geq \Omega(N)$
2. Bisections of the volume induce balanced cuts: $A \geq \Omega(k)$ (*) and $\sqrt{(A)}T \geq \Omega(k)$ (therefore $AT^2 \geq \Omega(k^2)$)
3. (maybe) all I/Os must pass through the perimeter at some point: $\sqrt{(A)}T \geq \Omega(n)$
4. (in some cases) it must be possible to communicate across the chip: $T \geq \sqrt{(A)}$

Combining (1) and (4): $T^3 \geq \Omega(N)$

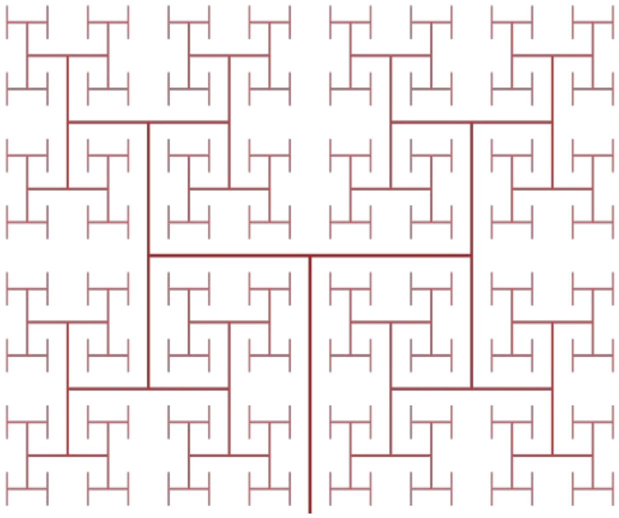
*: including memory-only area

Some example consequences

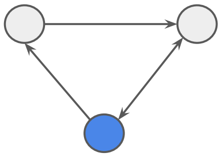
All one-output functions of n inputs take $\Omega(\sqrt[3]{n})$ time in parallel on a chip, since whenever there is one output, all inputs must have communication to the one place for the output; this includes, say, accessing memory

What about binary trees for reductions? Some wires will inevitably be length $\sqrt{(A^*)}$ (where A^* is the total area enclosing the computation), so we should not increase A^* beyond $\sqrt[3]{n^2}$; thankfully, this is achievable (do $\sqrt[3]{n^2}$ in-place accumulations of $\sqrt[3]{n}$ inputs each, then accumulate the results in $\sqrt[3]{n}$ time).

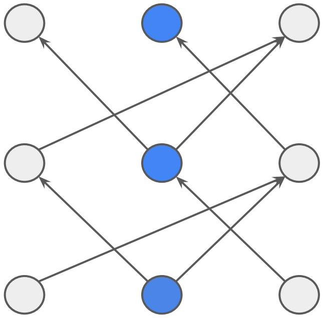
More interesting results for specific problems, given information about them (communication complexity, for instance).



Using algorithm structure: Bellman-Ford on an expander



G



BF computation graph on G

Bellman-Ford is a dynamic programming algorithm for single-source shortest paths in a weighted graph G=(V, E) where weights can be negative; it uses O(|V|^2) subproblems and takes O(E) time to compute each layer of |V| of them.

For sparse (constant-degree) graphs, this is O(|V|^2) sequential time.

How about parallel time?

Parallelization on a chip

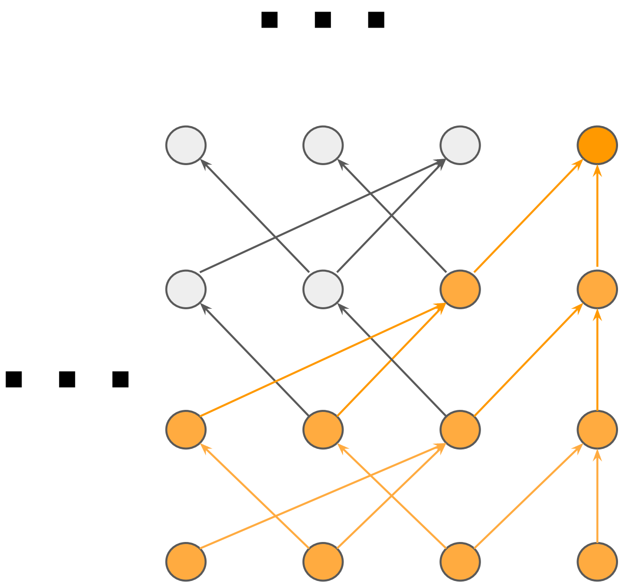
We can get 3 bounds on parallel runtime for Bellman-Ford on a sparse graph:

1. $T^3 \geq \Omega(N)$ gives $T \geq \Omega(|V|^{(2/3)})$
2. Dependencies must be respected even in parallel, and the depth of the graph is V; this gives us $T \geq \Omega(|V|)$
3. (I claim) $T \geq \Omega(|V|^{4/3}/(\log |V|))$

This will hold when the graph we are operating on has the following property:

For all subsets S of vertices, with $|S| \leq |V|/4$, the number of vertices with an edge into S is at least $2|S|$ (small-set vertex expander)

The computation graph for expanders



If the graph is a small-set vertex expander, each node in layer i of the computation graph depends on at least 2 nodes in the previous layer, 4 nodes in the one before that, etc…

At least V/4 nodes in layer (i-log_2(V/4))

Iterate the cube-root bound:

T_i ≥ Ω(V^(1/3)) + T_{i - O(log V)}

T_{V-1} ≥ Ω(V^(1/3))*Ω(V/log V)

T ≥ Ω(V^{4/3}/(log V))

Probably not achievable, but T=O(V^{3/2}) probably is (can this bound be strengthened?)

Final notes

- These bounds are stated for any 2d computation space, and therefore apply to all single chips, which is nice
  - All accelerator architectures
- Many of them apply to full-3d computation, with a difference (usually by 1) in exponent
- Using more information and assumptions from the architecture, or from the algorithm, or from the mapping (polyhedral…?) can produce better bounds
- I do not suspect that we will be able to produce anything interesting without assumptions about algorithms
- Other quantities are of interest (total communication distance for energy?)