Graph Reduction Hardware

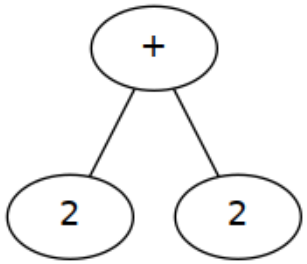
Shae M Erisson

2021-05-11

Outline

What is graph reduction?

- A graph is nodes connected by edges.



- A graph describing a computation is reduced to give a result.
- The computation 2 + 2 can be seen as the following graph.

- This boring graph unsurprisingly reduces to "4".

You want your laptop to go longer on less battery

- Past a certain point small increases in CPU speed require large increases in power and cooling.
- for details see power is supralinear in frequency.

We have to go parallel

- Parallel is spreading the work across multiple CPUs
- More not-so-fast cores increases total CPU power without unreasonable increases in power and cooling requirements.
- We can't get better performance-per-watt CPUs without multicore processors.
- You could pay many thousands of dollars to get a really fast CPU, or buy a bunch of slower CPUs for a much smaller total price.

Languages are not good at parallel

- C and most of its descendants were designed to work on one core. Those languages cannot easily use multiple cores.
- How do you make your Python code faster? (call out to something else, often another Python process)

Graph reduction is Haskell, graph reduction is good at parallel

- Almost all our society's programs have relied on one single CPU getting faster over time. That's unlikely to return.
- graph reduction is good at parallel, usually multiple parts of the graph can be reduced at the same time
- Haskell is graph reduction, because lambda calculus!

Existing hardware is not so good at parallel

- Can programming be liberated from the von Neumann style? 1978 John Backus
- We can write code that uses multiple cores, but it's not easy
- Even then, all these cores still pretend to be one fast CPU! They have the same view of main memory! More cores means more overhead synchronizing the shared view of memory.

Design your own CPU with reconfigurable hardware!

- You can design your own computer chips with an FPGA!

FPGAs can do parallel

- TIGRE (1992) - At that time, sequential CPU speed was still increasing, decided special purpose chips couldn't keep up
- Reduceron (2010) - CPU designed to do graph reduction
- PilGRIM (2010) - builds on Reduceron, includes pipelining, closer to existing CPUs

Reduceron

- The Reduceron (2008-2010) is an open source design that runs on an FPGA.
- designed for a much wider data bus than x86
- Each instruction does more work, 6 reductions per clock cycle!
- Programs running on an FPGA at 90MHz could outperform a 2GHz x86 system
- Proof of concept, not production quality, programs can't have input!
- small group updating (2020) Reduceron on GitHub
  https://github.com/tommythorn/Reduceron

PilGRIM - Pipelined Graph Reduction Instruction Machine

- The PilGRIM (2010) built on the success of the Reduceron
- deep pipeline with the goal of pushing Reduceron's ~90MHz towards 1GHz

Will this ever reach mass produced hardware?

- Might already exist with the Graphcore IPU products!
- starter kit is $20,000 USD
- 1472 tiny CPUs with small amounts of local memory
- 47 TB/s memory bandwidth per CPU
- available from Microsoft Azure, but I haven't tried it.

Questions? Thoughts? Bonus content?

- Graph reduction isn't the only approach to implicit parallelism
- Graph reduction doesn't create parallelism in a problem that isn't parallel
- Cycle count from CPU to main memory continues to increase
- graph reduction decreases the need for cache coherency? lock free mechanism for evaluating shared thunks
- What about doing graph reduction on GPUs? I don't know, maybe you know?
- Does custom/offbeat/niche hardware ever succeed? GPUs and DSPs yes, also many failures

FPGA floorplans look really cool

- FPGA floorplan viewer! https://knielsen.github.io/ice40_viewer/ice40_viewer.html Wow!