ASPIRE: Algorithms and Specializers for Provably-Optimal Resiliency and Efficiency

Krste Asanovic, Jim Demmel, Armando Fox, Kurt Keutzer, Borivoje Nikolic, David Patterson, John Wawrzynek

Par Lab All Meeting

Feb 29, 2012

Not “The” Next Project

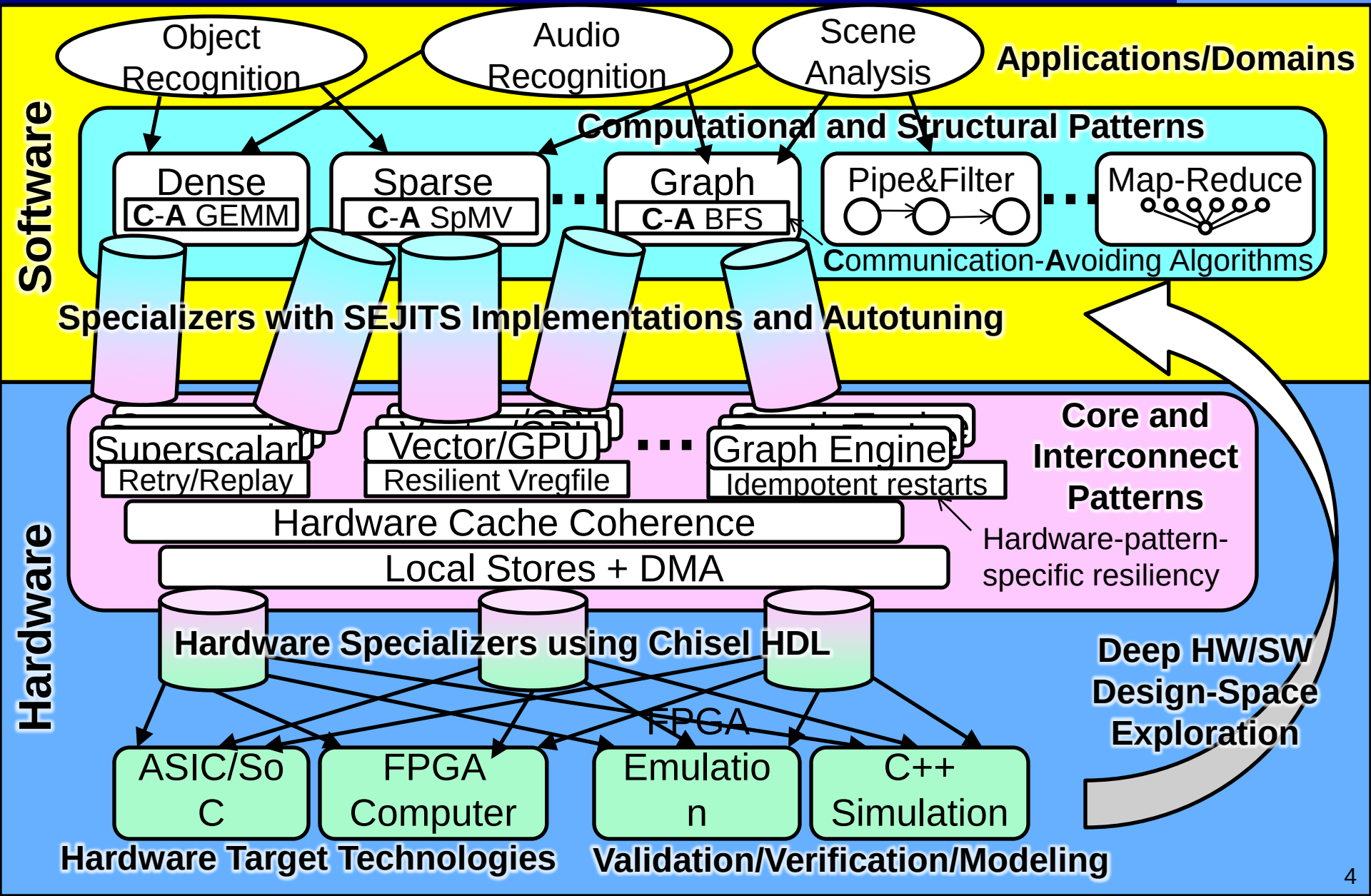
- A subset of what we’re doing in Par Lab
- Focused on the core computational stack

- Doesn’t include the user interface, agents, client+cloud, concurrent programming, pieces of the 2012 winter retreat “Next Project”

How to get more energy efficiency?

- Parallelism was one step, using more, lower-performance cores with better energy/op
  - Simpler general-purpose microarchitectures
    - Limited by smallest GP core
  - Lower Vdd/Frequency
    - Limited by Vdd/Vt scaling, errors
- Specialization is next step

UCB HW/SW Specializer Stack



Can we be provably optimal?

- Insight 1: Communication, both up and down memory hierarchy and across cores, dominates performance and energy consumption in many applications
- Insight 2: Jim's group keep getting best paper awards on communication-avoiding algorithms
  - Congrats on SIAM/LA Prize!

Strategy 1: Communication-Bound

- 1) Prove lower bounds on communication for a computation
- 2) Develop algorithm that achieves lower bound on a system
- 3) Find that communication energy cost is >90% of resulting algorithm
- 4) We know we're within 10% of optimal

- Supporting technique: Empircally optimize cores so that they get sufficiently low energy to ignore

Strategy 2: Core-Bound

- Can we develop new cores that reduce energy/task?
- Maybe push communication-avoiding ideas to finer grain?
  - Communication-avoiding microarchitectures?
- Example, what is optimal register file + scratchpad sizes for FFT engine?