

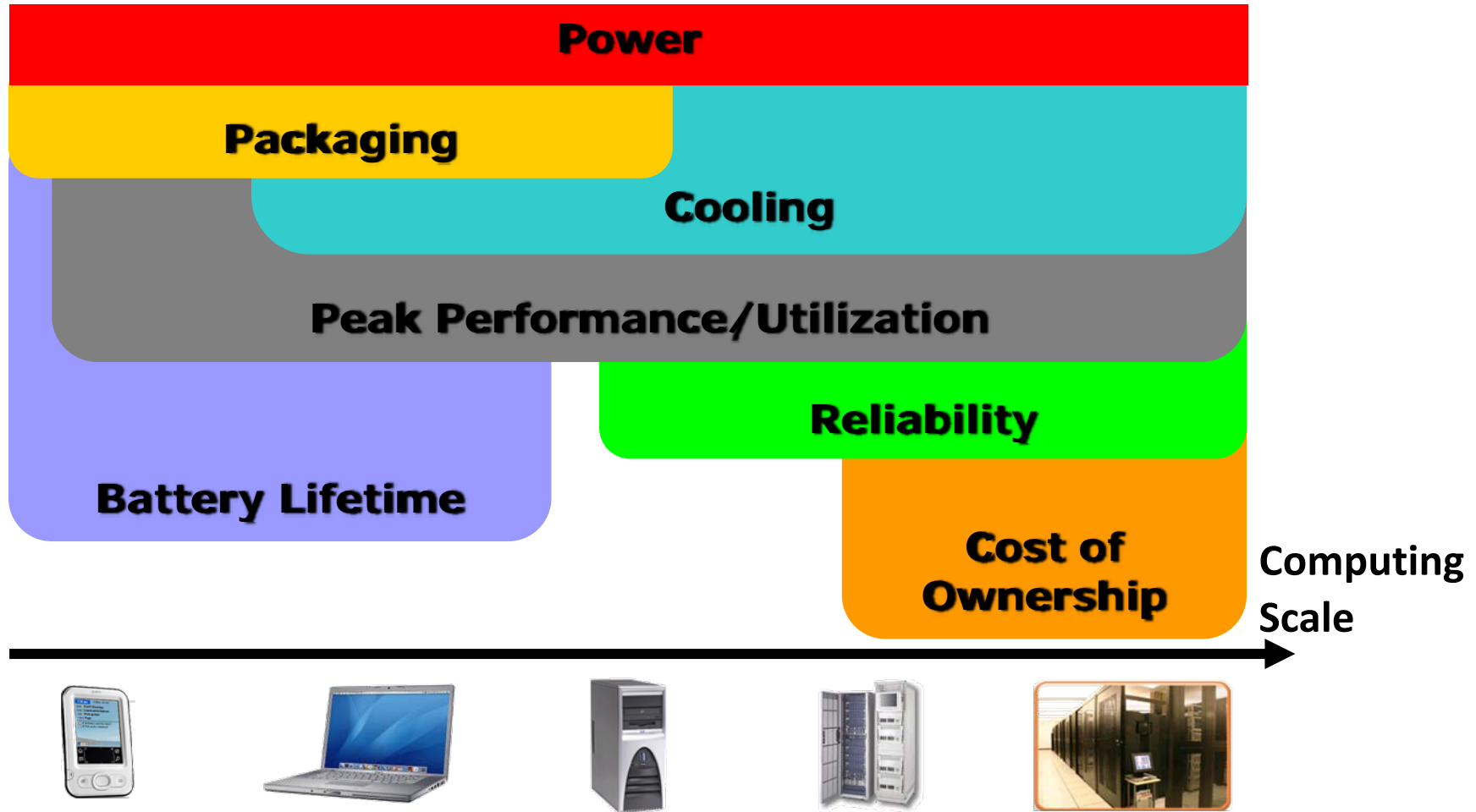
Emerging Issues for Next-Generation Parallelism

Prof. Margaret Martonosi
Princeton University



PRINCETON

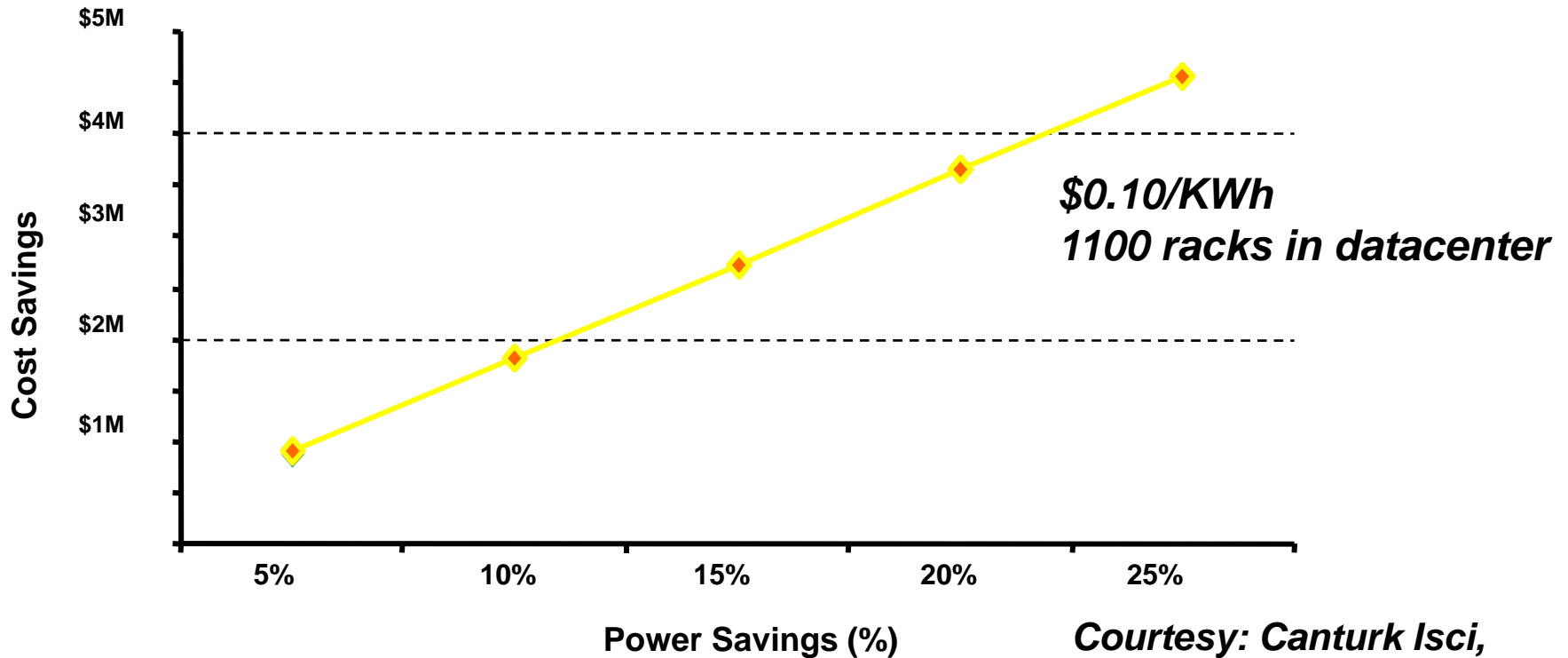
Power/Energy Critical Across Computing Spectrum



My work spans this whole range: CPUs -> data centers; mobile->enterprise
This talk: Start with CPUs and then broaden



The Economic Side: Cost of Power



*Courtesy: Canturk Isci,
Eugene Gorbatov, Intel '06*

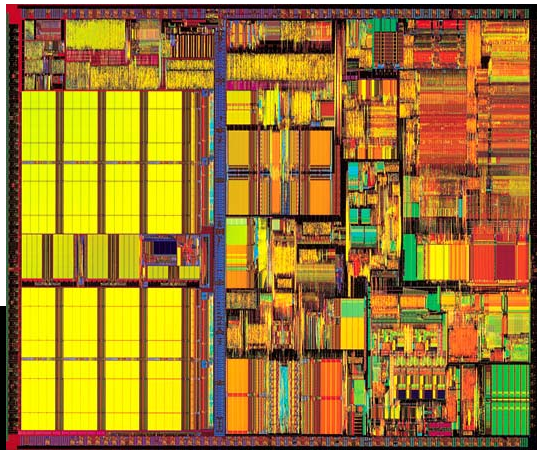
- Power Triple Play: Save 1W in processor power
 - ~1W power supply conversion
 - ~1W cooling power



Microprocessors: Then and now...

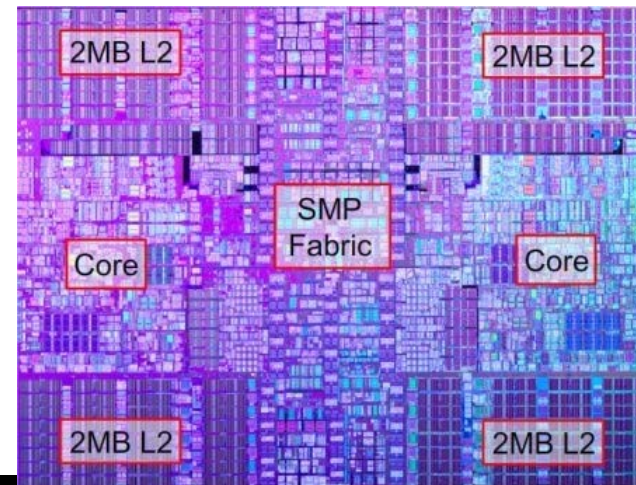
1999

- Unicore chips
- Millions of transistors
- Power not yet on radar screen
- Example: Intel Pentium III
 - 450MHz, 9.5M transistors, 0.25 μ



Now

- Multicore chips
- Billions of transistors
- Multidimensional design space: Performance, power, thermal, reliability...



IBM Power6



Then and now, part 2...

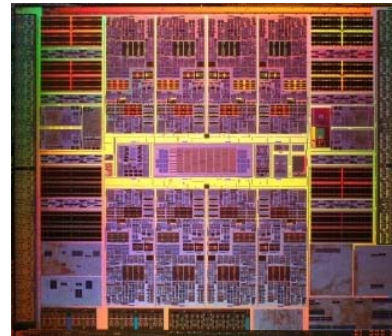
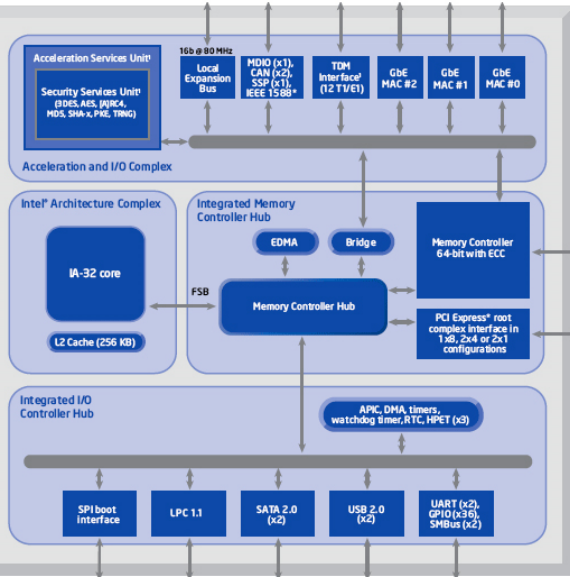
- 1990's: Embedded and High-Performance arenas quite distinct
 - Embedded processors: low-performance, low-power controllers
 - High-Performance: high-performance, high-clock rate, not much thought about power [yet].
- Now:
 - Embedded domain needs high performance
 - General-Purpose domain needs manageable power

=> Convergence!



CMP Design Issues

Block Diagram for the Intel® EP80579 Integrated Processor with Intel® QuickAssist Technology

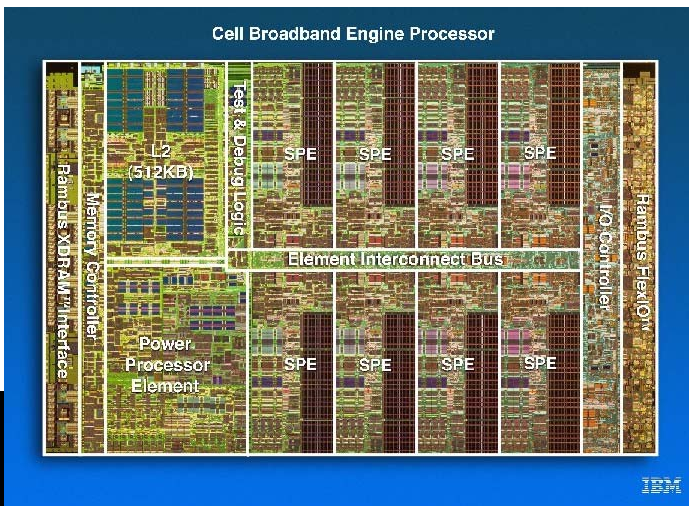


Goals:

- High performance and power-efficiency...
- Mitigating design and management complexity
- Performance portability across technology generations

Strategies:

- Multi-core
- Often with specialization and heterogeneity...
- Dynamic Resource and Parallelism Management



Our Opportunity

Create the CHIP-LEVEL or SYSTEM-LEVEL abstraction that is analogous to ISA's role at the core level



System-Level ISA Abstractions: Our Research Agenda

1. What are the new abstractions and APIs between system software and [heterogeneous] multi-core hardware?
 - How to efficiently program and manage these chips?
 - Balancing multiple metrics: performance, power, reliability
2. How to guarantee performance portability across technology generations?
 - while seamlessly tracking changes in core counts or types of heterogeneous cores available
3. How to design robust and flexible CMP architectures?
 - How to integrate many heterogeneous cores with 50X energy efficiency improvement? How to map the tasks on the cores?



System-Level ISA (SISA): Why? What is it good for?

Why?

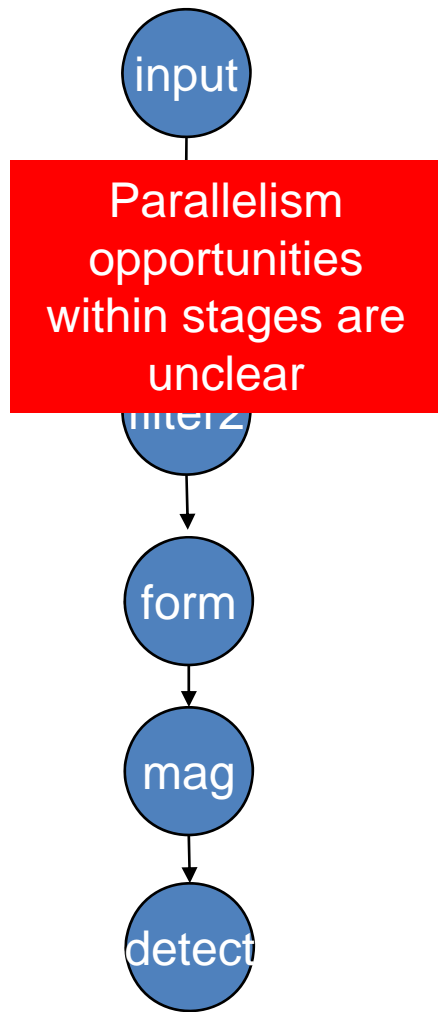
- Now: ISA is sole hw/sw interface
- ISA gives too little info on...
 - Parallelism available
 - Program structure
 - Criticality and work estimates
 - Mappings to Special functional units (SFUs) where available

What SISA Enables...

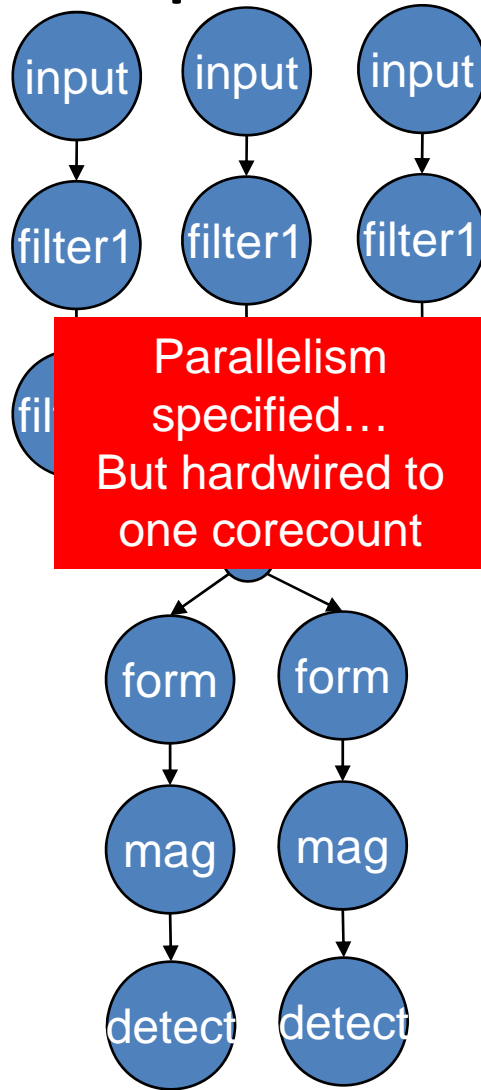
- Use or suppress parallelism depending on corecounts
- Rebalance work if core goes offline
- Estimate Criticality and use DVFS or other techniques to speed-balance work or maintain energy target
- Offload work to SFUs where available (Else, general-purpose CPUs)



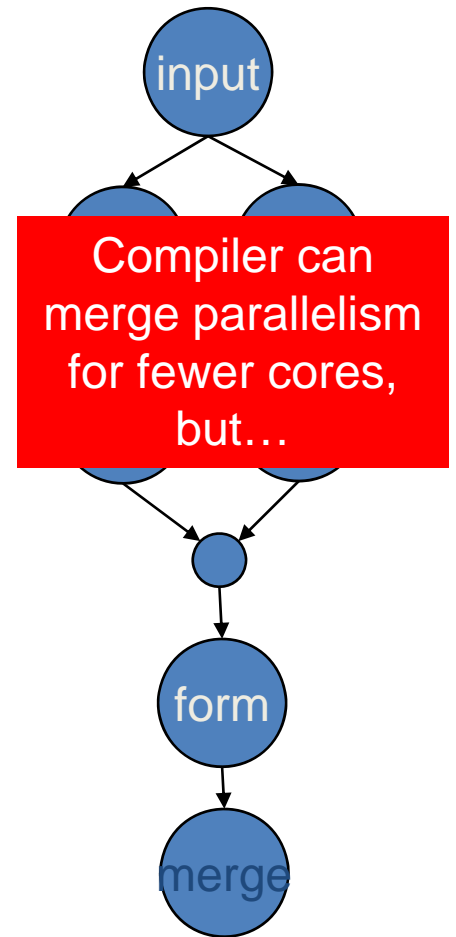
Example: Beam Forming



Algorithm Flow



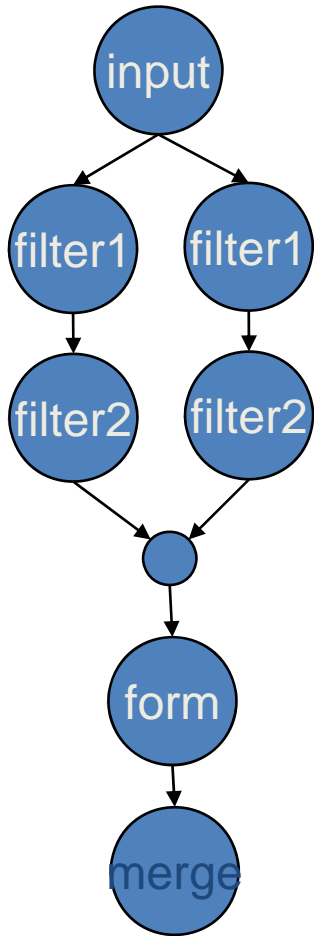
Hand-Parallelized



Compiler-Merged



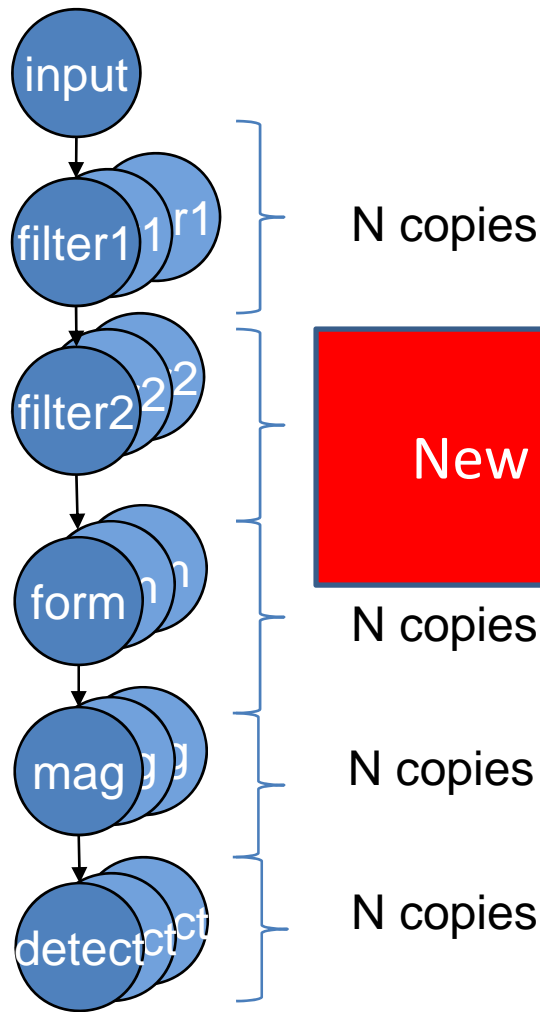
Currently, Parallelism Mapping limits Performance Portability



- Fundamentally limited by #cores programmer envisioned
 - Need to identify “all” parallelism to port to future cores...
- Currently almost all static
 - Need dynamic management due to Corecounts, Variations and reliability
- Once mapped, graph structure is difficult for hardware to reproduce
 - Need to retain graph structure to enhance opportunities for criticality optimization and resource management.



SISA



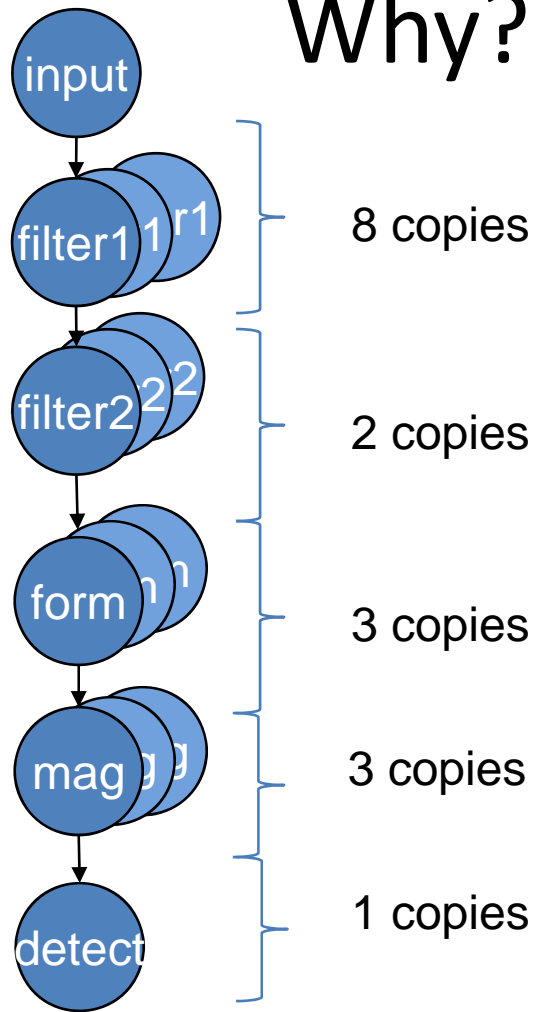
- Identify “maximal” parallelism
- Pass SISA graph to hardware as part of executable

**SISA Graph Structure:
New HW/SW Interface Abstraction to support
Dynamic Code Management**

- Compiles chunks to local ISA
- **Dynamic Management**
 - Adapts during run to changing resource and reliability landscape



System-Level ISA (SISA): Why? What is it good for?



What SISA Enables...

- Use or suppress parallelism depending on corecounts
- Rebalance work if core goes offline
- Estimate Criticality and use DVFS or other techniques to speed-balance work or maintain energy target
- Offload work to SFUs where available (Else, general-purpose CPUs)

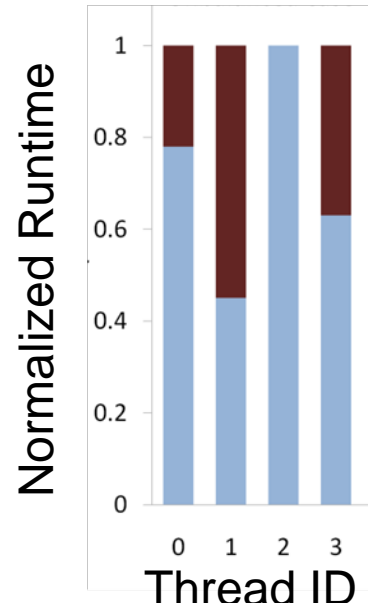


What SISA Enables: One example

Thread Criticality Prediction

Threads of a parallel program don't end at precisely the same time.

- Particularly if app is written in a general (untuned) manner to port across many systems
- Particularly if cores are heterogeneous in speed
- Longest running thread is "critical", but criticality is hard to deduce until runtime (caches, multiprocessing, etc.).



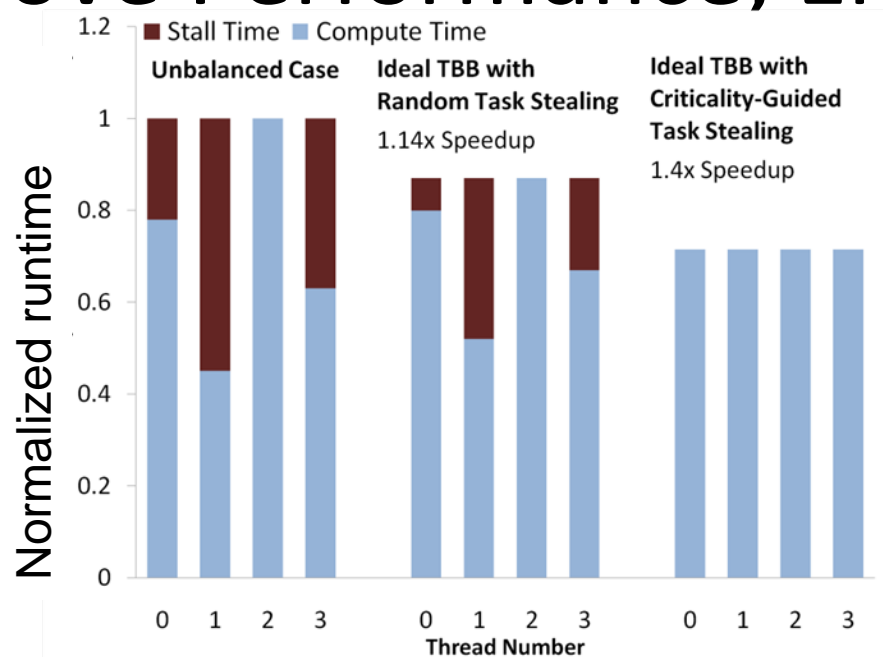
Blackscholes: A PARSEC app, rewritten for TBB.

Despite TBB's emphasis on dynamic task stealing, load imbalance still remains

If we can deduce that thread 2 is critical, and by how much, what could we do with this information?



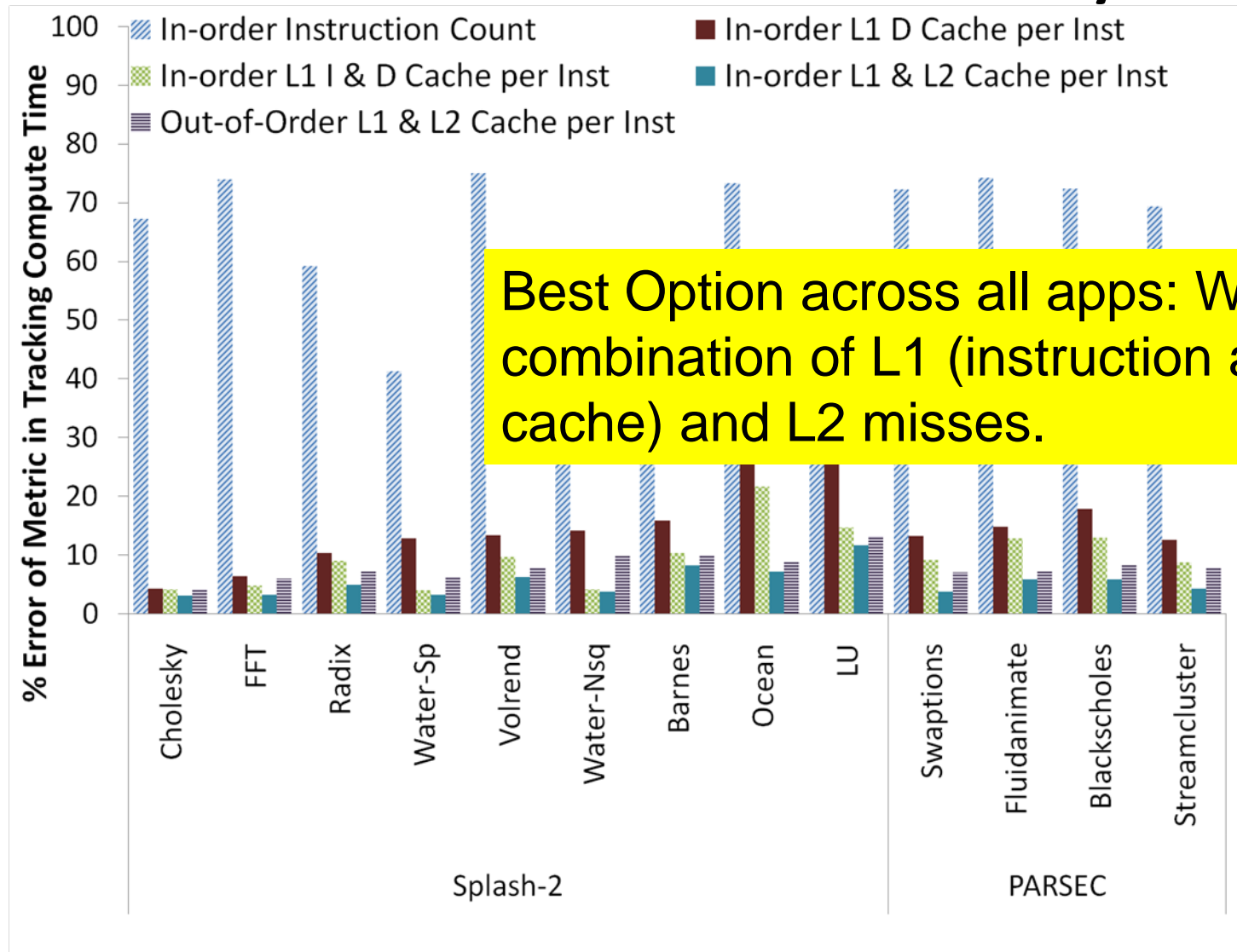
How much can Thread Criticality Prediction Improve Performance, Energy?



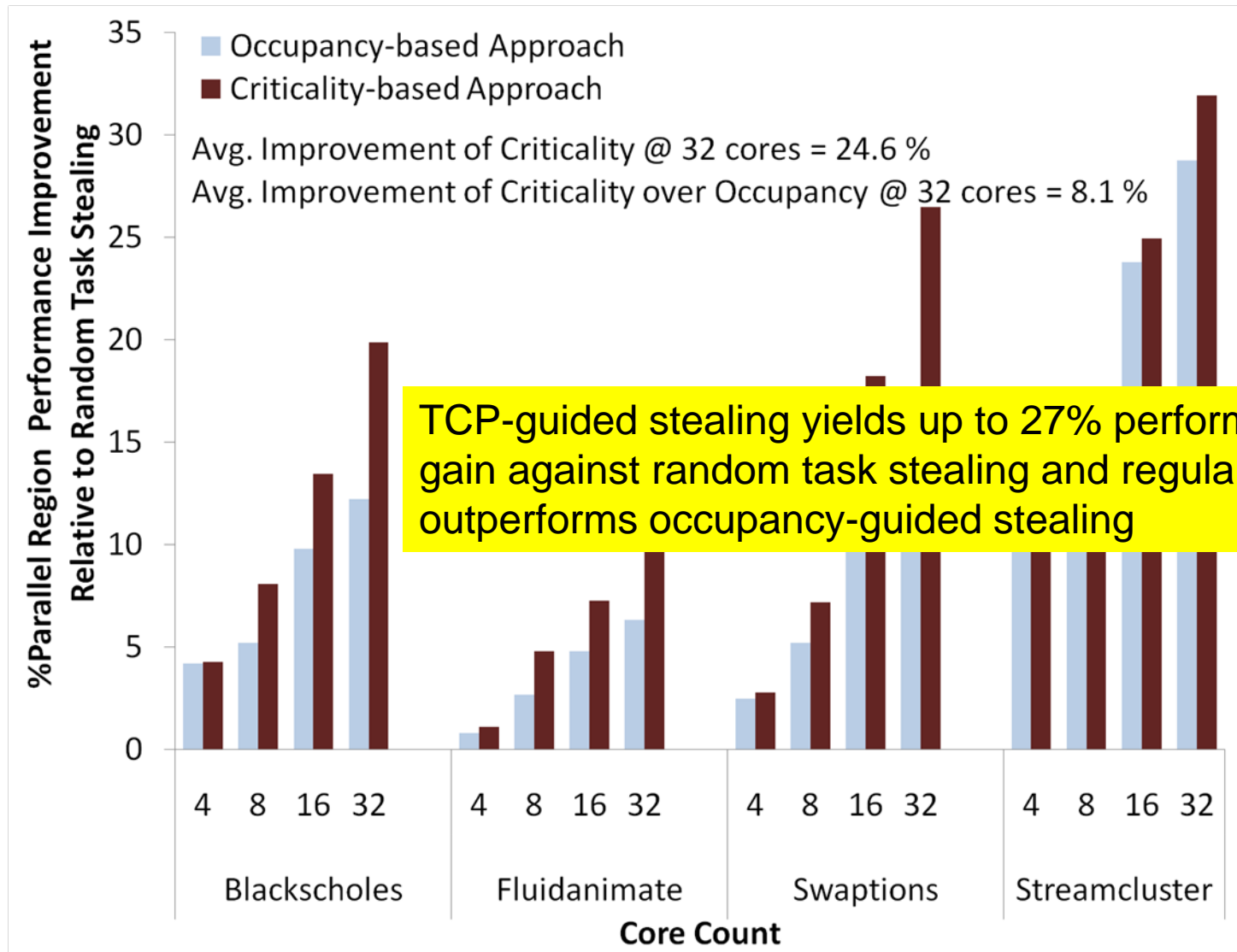
- TCP to save energy:
 - Slow down non-critical threads
- TCP to improve performance:
 - Use criticality to guide task rebalancing (“stealing”)



How to Predict Criticality?

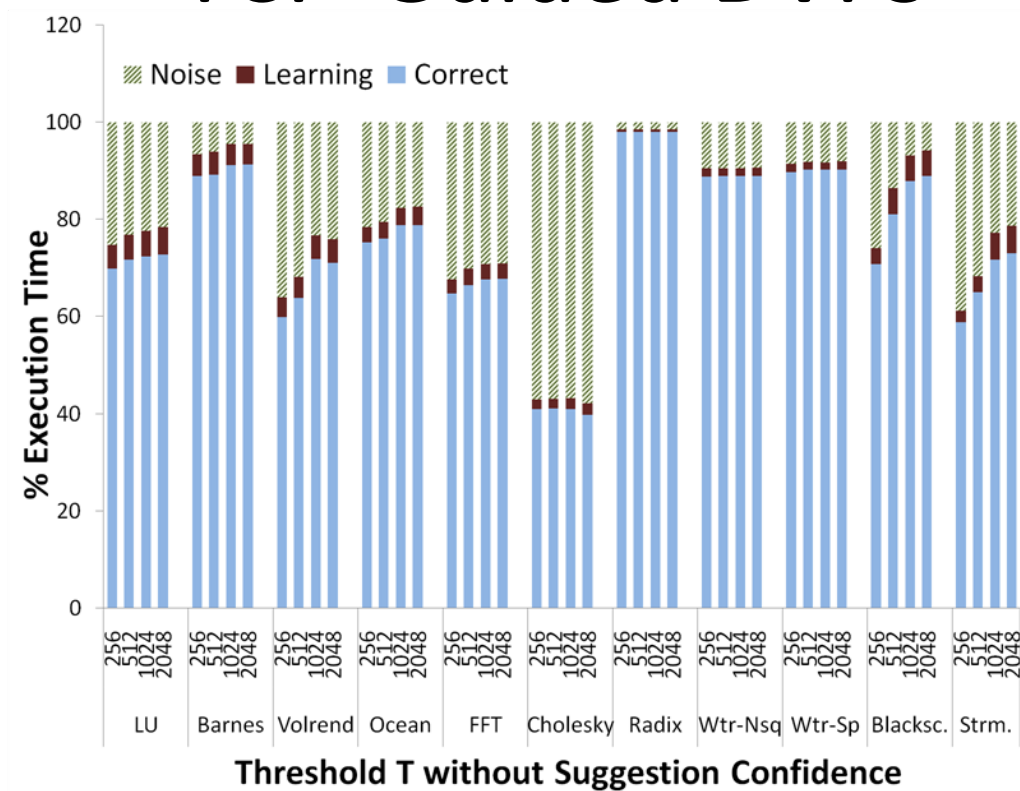


Results: TCP for Task Stealing



How about TCP for Energy?

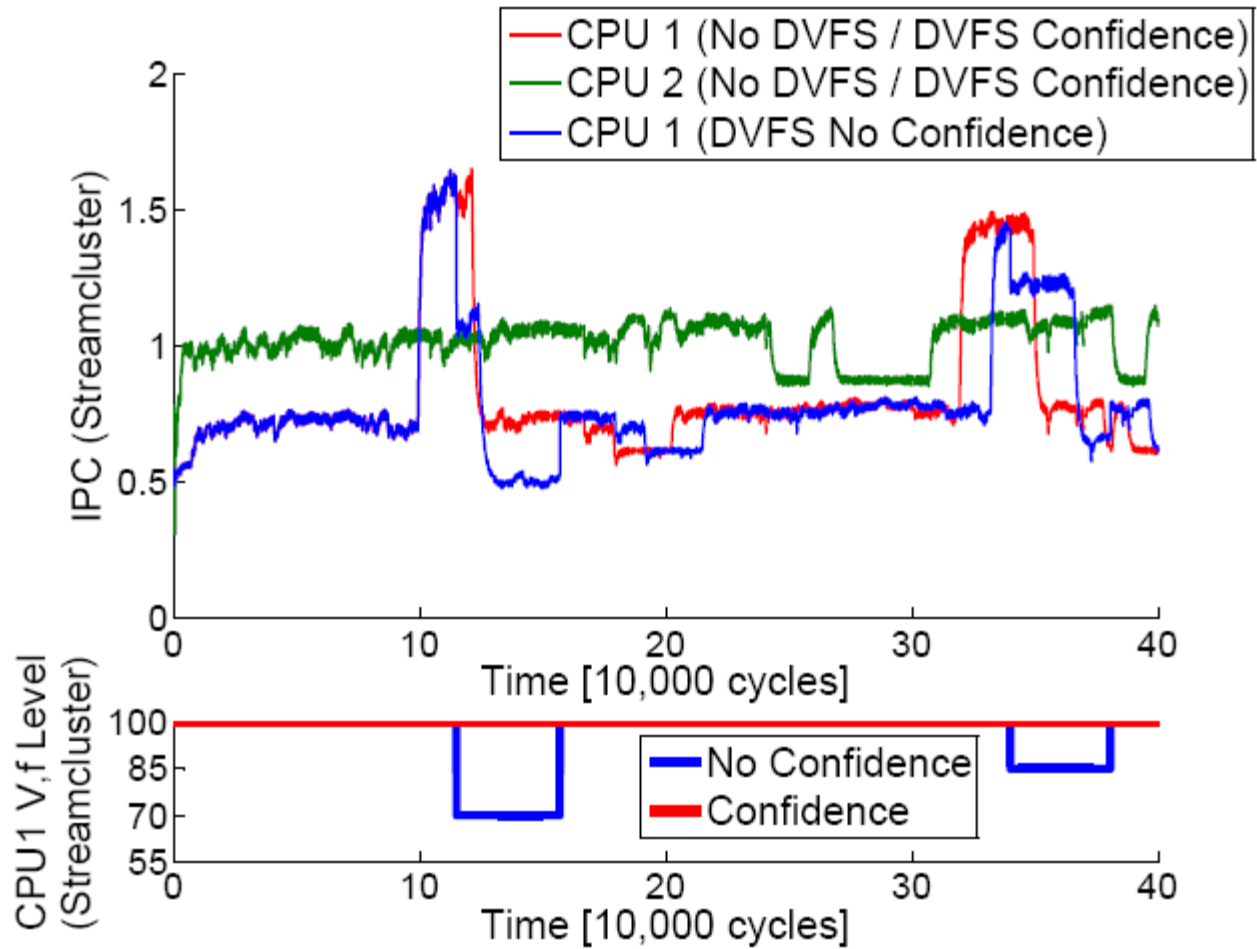
TCP-Guided DVFS



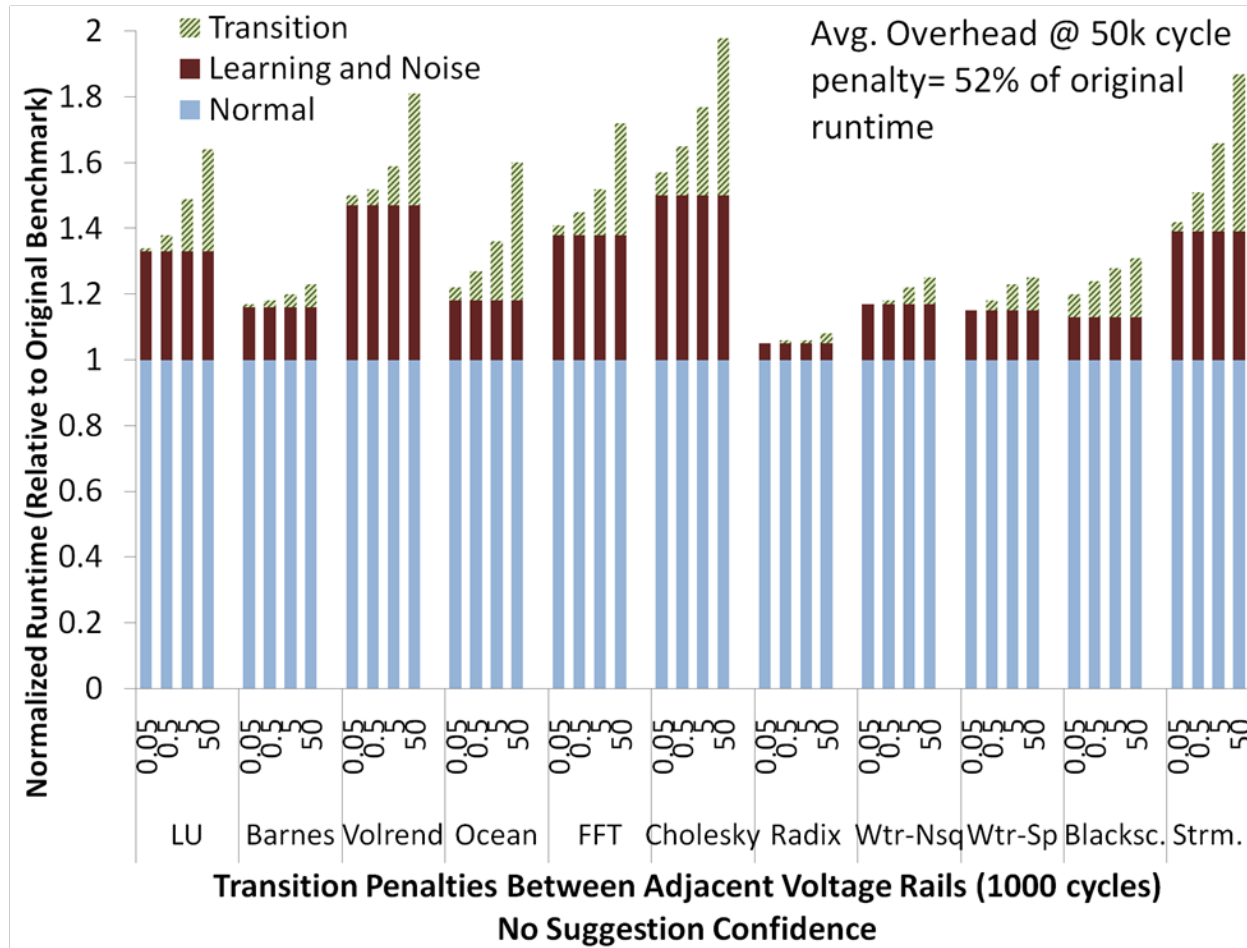
- About 40% Potential savings...
- But because DVFS has a time/energy cost, it must be only applied conservatively.
- Need a confidence estimator...



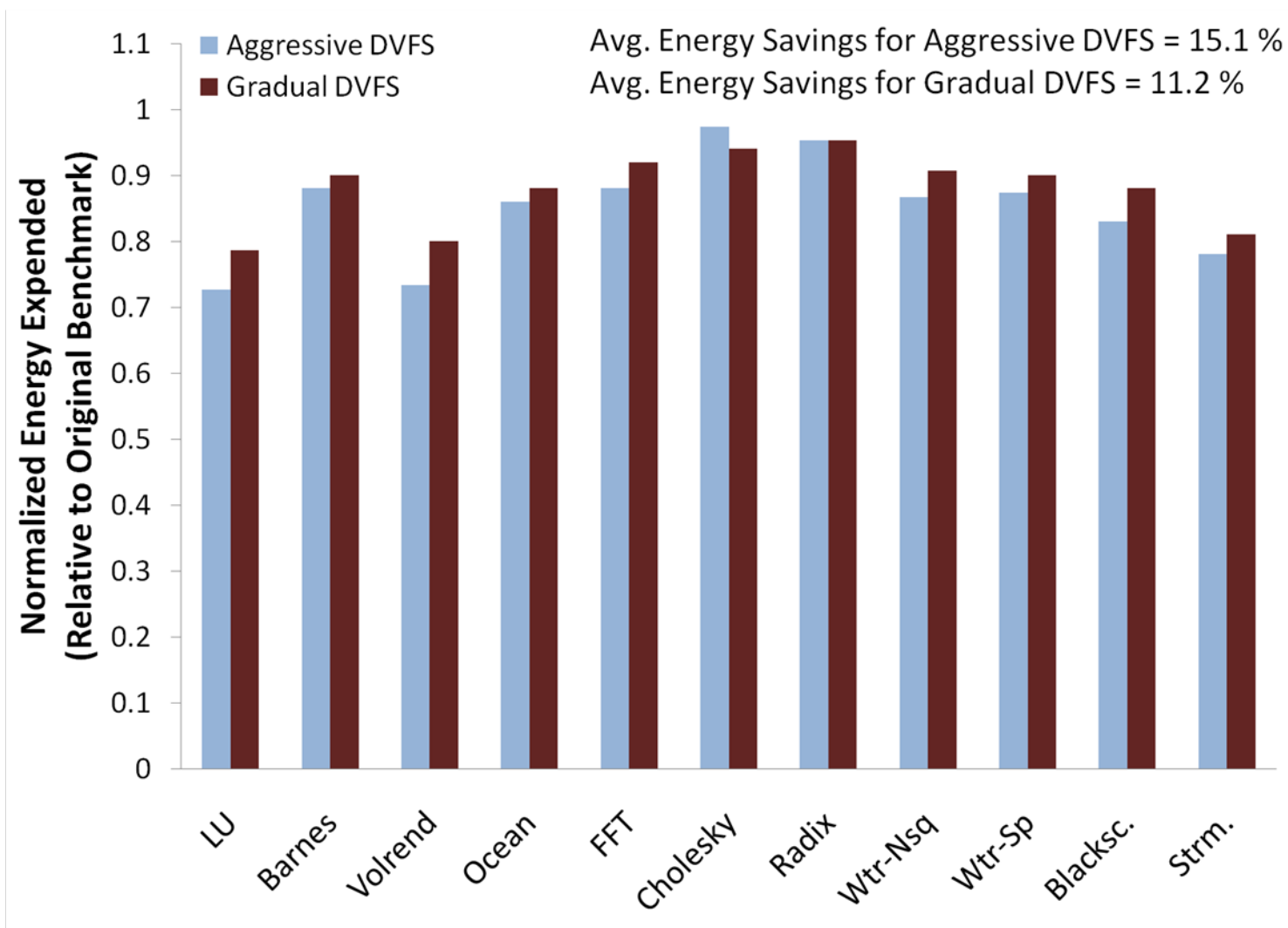
Confidence estimator only changes (V,f) when same one is repeatedly suggested



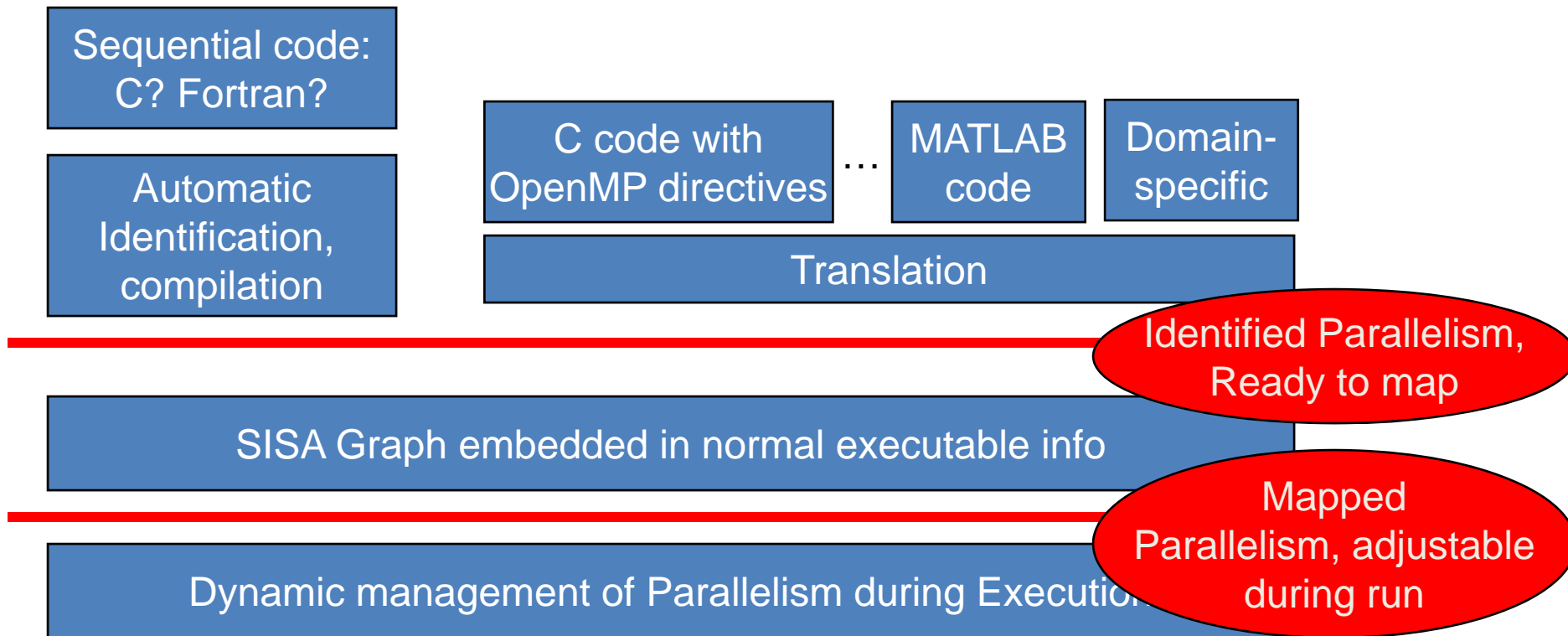
DVFS transition penalties: Another view



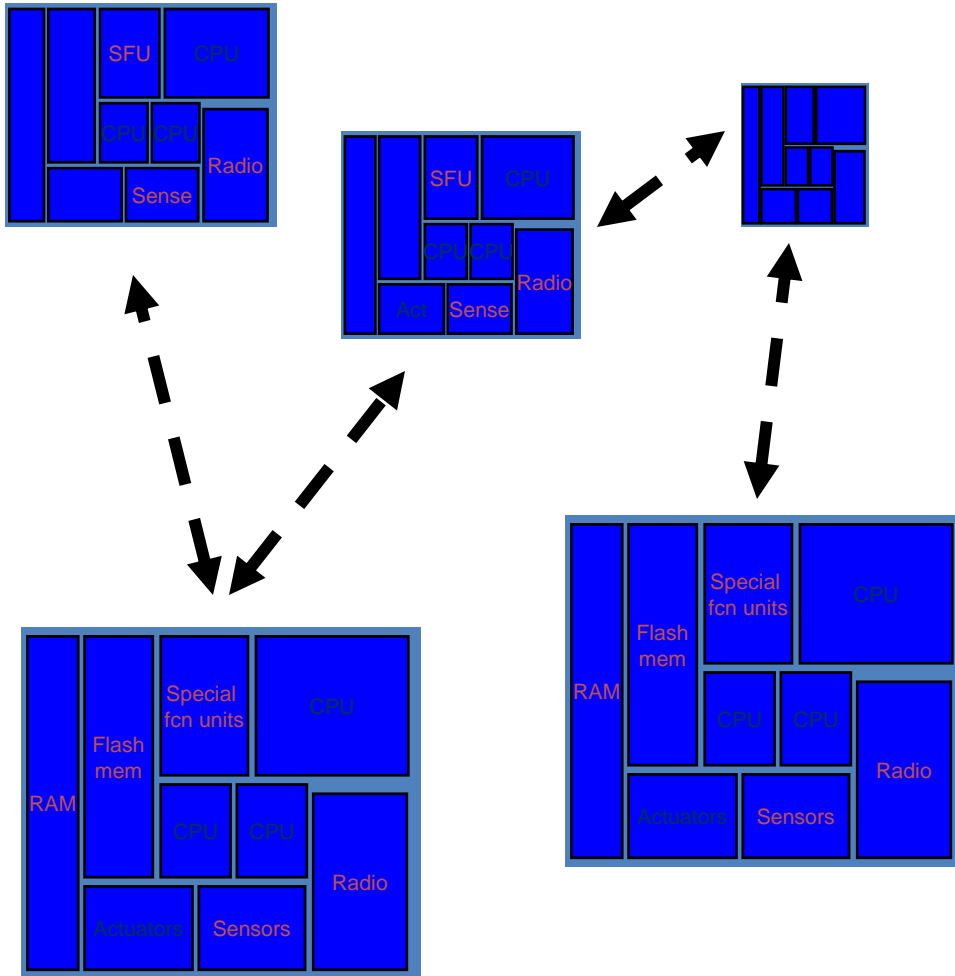
TCP Guided DVFS: Energy Savings



Where do SISA graphs come from?



Broader View: Across Chip Boundaries



System-level ISA

- Distributed resource management
- Managing interfaces, comm

Goal:

- Adaptive, Efficient resource brokers
- Multi-level ISA and control



Takeaway message

- Today's systems: CMPs, SoCs, and even building-scale need agile power/performance tradeoffs.
- Agile Power/performance tradeoffs require System-level Dynamic Management
 - This talk: Initial examples of criticality predictors and resource managers for power/perf/resource management
- To have effective, composable dynamic management, we need an “ISA-style” abstraction at system level
 - Our current/future work...

