Programming the Energy-Efficiency of High-Performance Computing Systems

Professor Dimitrios S. Nikolopoulos

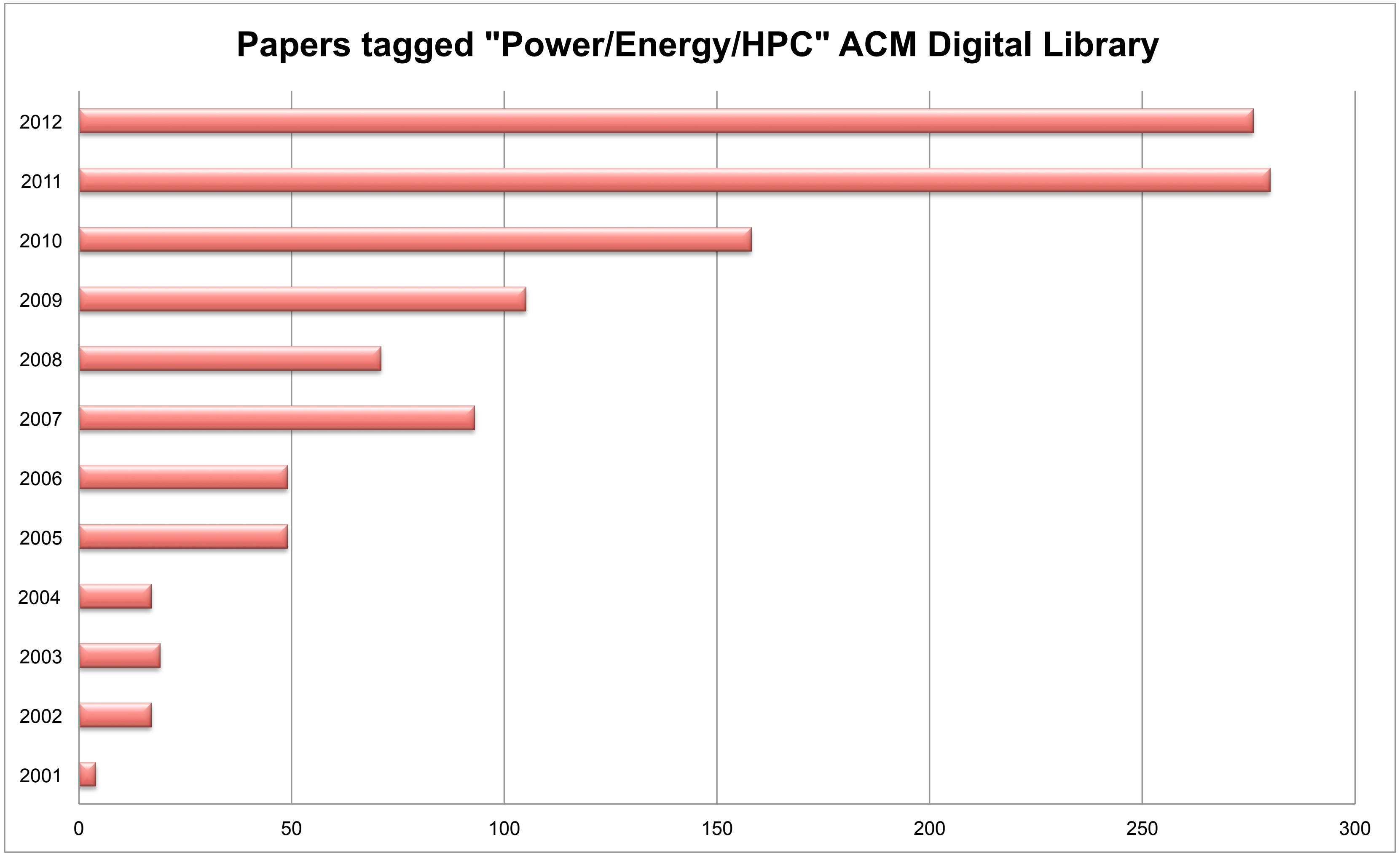
HPDC Research Cluster, Queen's University of Belfast

Points to get across

- Waste-free HPC software is instrumental in the battle against power
- Scale-freedom in parallel programming is a path to energy-efficiency
- Energy challenges will remind us of the *Hydra Lernaia*

School of EEECS
HPDC Cluster

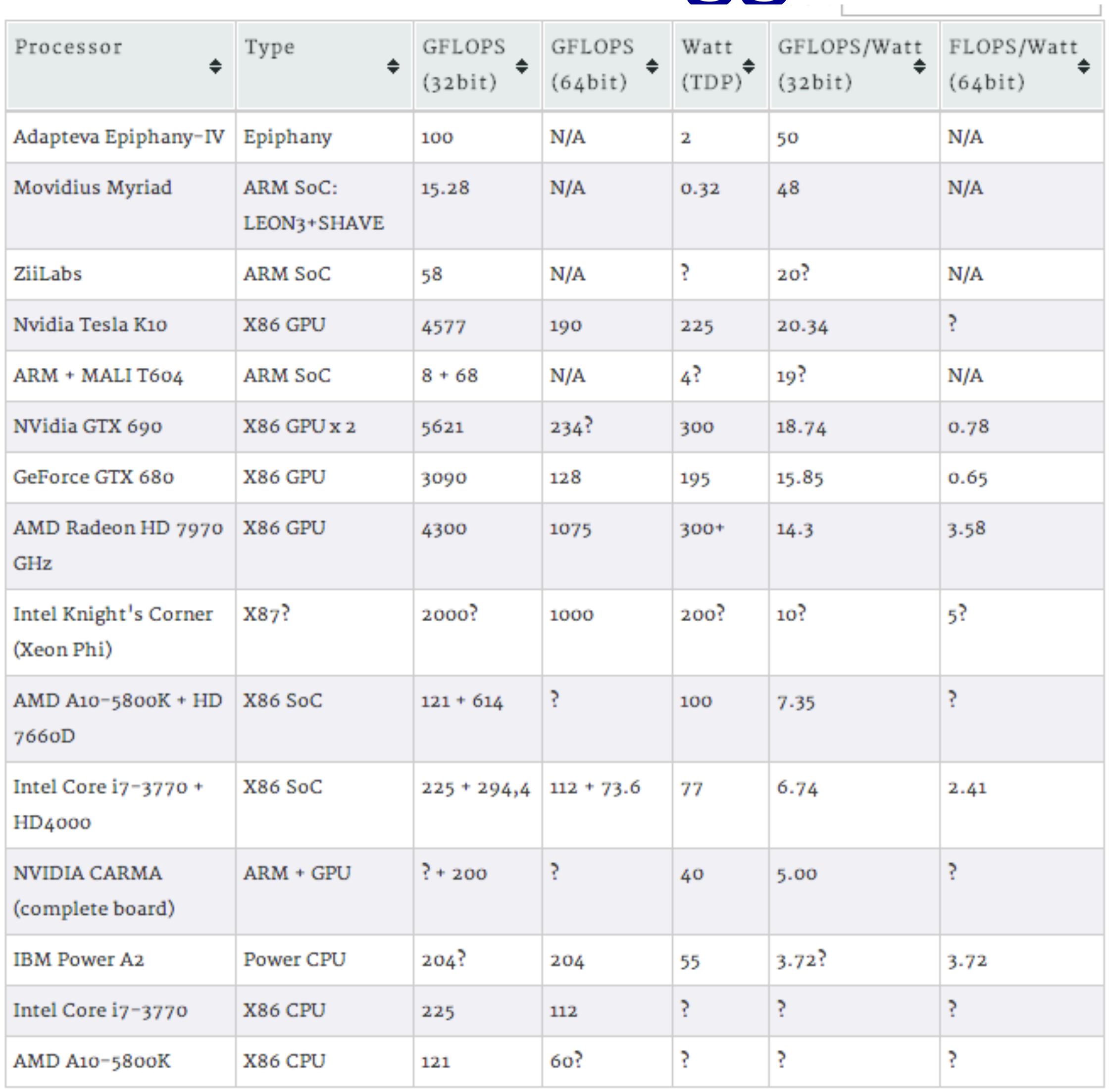
Energy in HPC



HPC has lead the way (or not?)

- The history of BlueGene
  - Based on processors for the embedded systems market (PowerPC)
  - Pioneered "scale-out" idea, now common in datacentres
    - Many nodes with simple cores, fast interconnect
  - Dominated Top-500, Green-500 list
- Embedded processors are now commodity components
  - Able to power competing supercomputers (e.g. BSC MontBlanc)

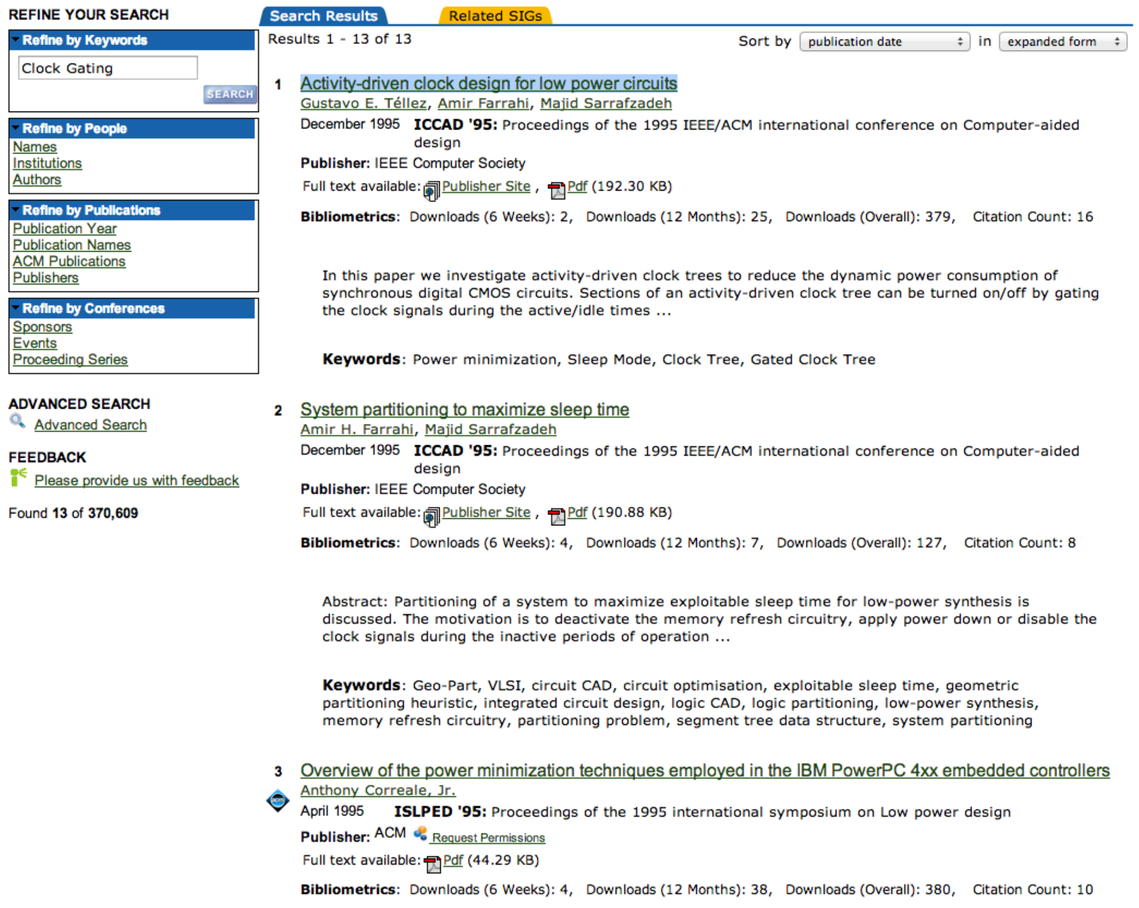
Leader or laggard?



Processor	Type	GFLOPS (32bit)	GFLOPS (64bit)	Watt (TDP)	GFLOPS/Watt (32bit)	FLOPS/Watt (64bit)
Adapteva Epiphany-IV	Epiphany	100	N/A	2	50	N/A
Movidius Myriad	ARM SoC: LEON3+SHAVE	15.28	N/A	0.32	48	N/A
ZiiLabs	ARM SoC	58	N/A	?	20?	N/A
Nvidia Tesla K10	X86 GPU	4577	190	225	20.34	?
ARM + MALI T604	ARM SoC	8 + 68	N/A	4?	19?	N/A
NVidia GTX 690	X86 GPU x 2	5621	234?	300	18.74	0.78
GeForce GTX 680	X86 GPU	3090	128	195	15.85	0.65
AMD Radeon HD 7970 GHz	X86 GPU	4300	1075	300+	14.3	3.58
Intel Knight's Corner (Xeon Phi)	X87?	2000?	1000	200?	10?	5?
AMD A10-5800K + HD 7660D	X86 SoC	121 + 614	?	100	7.35	?
Intel Core i7-3770 + HD4000	X86 SoC	225 + 294,4	112 + 73.6	77	6.74	2.41
NVIDIA CARMA (complete board)	ARM + GPU	? + 200	?	40	5.00	?
IBM Power A2	Power CPU	204?	204	55	3.72?	3.72
Intel Core i7-3770	X86 CPU	225	112	?	?	?
AMD A10-5800K	X86 CPU	121	60?	?	?	?

Leader or laggard?

- Is HPC reusing or discovering?
  - Processors originally designed for the mobile phones market
  - Clock gating, DVFS, device sleep states well known for 20 years



REFINE YOUR SEARCH

Refine by Keywords
Clock Gating
SEARCH

Refine by People
Names
Institutions
Authors

Refine by Publications
Publication Year
Publication Names
ACM Publications
Publishers

Refine by Conferences
Sponsors
Events
Proceeding Series

ADVANCED SEARCH
Advanced Search

FEEDBACK
Please provide us with feedback

Found 13 of 370,609

Search Results | Related SIGs

Results 1 - 13 of 13 | Sort by publication date in expanded form

1 Activity-driven clock design for low power circuits
Gustavo E. Téllez, Amir Farrahi, Majid Sarrafzadeh
December 1995 **ICCAD '95:** Proceedings of the 1995 IEEE/ACM international conference on Computer-aided design
Publisher: IEEE Computer Society
Full text available: Publisher Site , Pdf (192.30 KB)
Bibliometrics: Downloads (6 Weeks): 2, Downloads (12 Months): 25, Downloads (Overall): 379, Citation Count: 16

In this paper we investigate activity-driven clock trees to reduce the dynamic power consumption of synchronous digital CMOS circuits. Sections of an activity-driven clock tree can be turned on/off by gating the clock signals during the active/idle times ...

Keywords: Power minimization, Sleep Mode, Clock Tree, Gated Clock Tree

2 System partitioning to maximize sleep time
Amir H. Farrahi, Majid Sarrafzadeh
December 1995 **ICCAD '95:** Proceedings of the 1995 IEEE/ACM international conference on Computer-aided design
Publisher: IEEE Computer Society
Full text available: Publisher Site , Pdf (190.88 KB)
Bibliometrics: Downloads (6 Weeks): 4, Downloads (12 Months): 7, Downloads (Overall): 127, Citation Count: 8

Abstract: Partitioning of a system to maximize exploitable sleep time for low-power synthesis is discussed. The motivation is to deactivate the memory refresh circuitry, apply power down or disable the clock signals during the inactive periods of operation ...

Keywords: Geo-Part, VLSI, circuit CAD, circuit optimisation, exploitable sleep time, geometric partitioning heuristic, integrated circuit design, logic CAD, logic partitioning, low-power synthesis, memory refresh circuitry, partitioning problem, segment tree data structure, system partitioning

3 Overview of the power minimization techniques employed in the IBM PowerPC 4xx embedded controllers
Anthony Correale, Jr.
April 1995 **ISLPED '95:** Proceedings of the 1995 international symposium on Low power design
Publisher: ACM Request Permissions
Full text available: Pdf (44.29 KB)
Bibliometrics: Downloads (6 Weeks): 4, Downloads (12 Months): 38, Downloads (Overall): 380, Citation Count: 10

What can HPC contribute towards zero-power computing?

The challenge and the opportunity

- Assume that currently most energy-efficient supercomputer sustains improvement towards an Exaflop
  - Will need 2384× in performance, 202.7 MW
- Assume target power cap of 25 MW
  - Need two orders of magnitude improvement in FLOPS/W
  - Can hardware achieve this improvement without compromising the power target?
  - If systems have any hope to achieve this they must eliminate waste
  - Actual power cap may be lower than nominal power consumption
  - Opportunities for software!

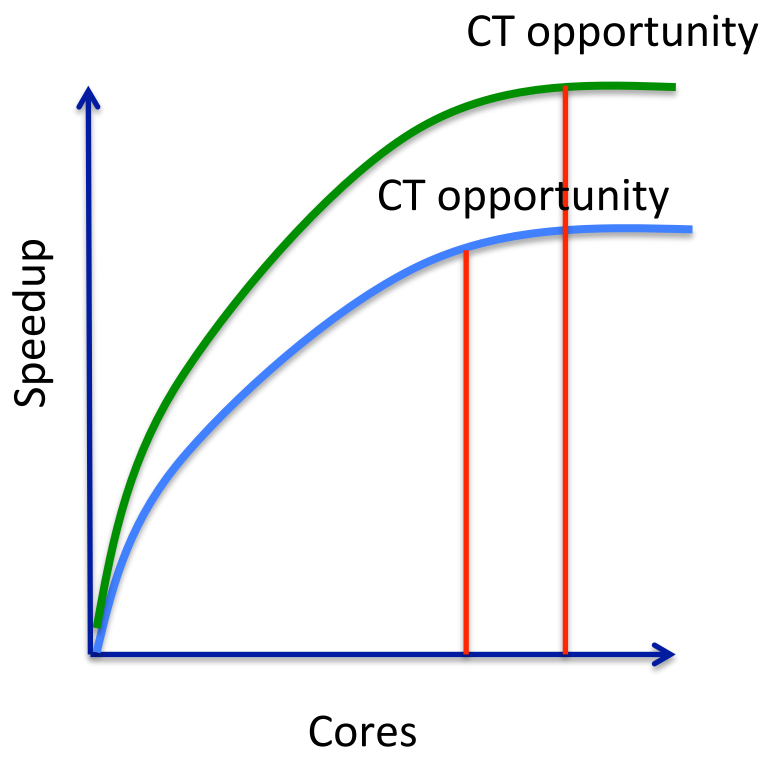
Where can HPC make the difference?

- HPC has been leading the way in parallel programming technology
  - Parallel languages, compilers, runtime systems
- HPC prioritizes efficiency in programming
  - Minimise communication
  - Balance the load
  - Utilise available cores
  - Reduce cache and memory footprint
- *Waste-free parallel programming is energy-efficient*
  - *Opportunity to reduce power consumption*
  - *Opportunity to do more within a power budget*

What can parallel languages and runtimes do to reduce waste?

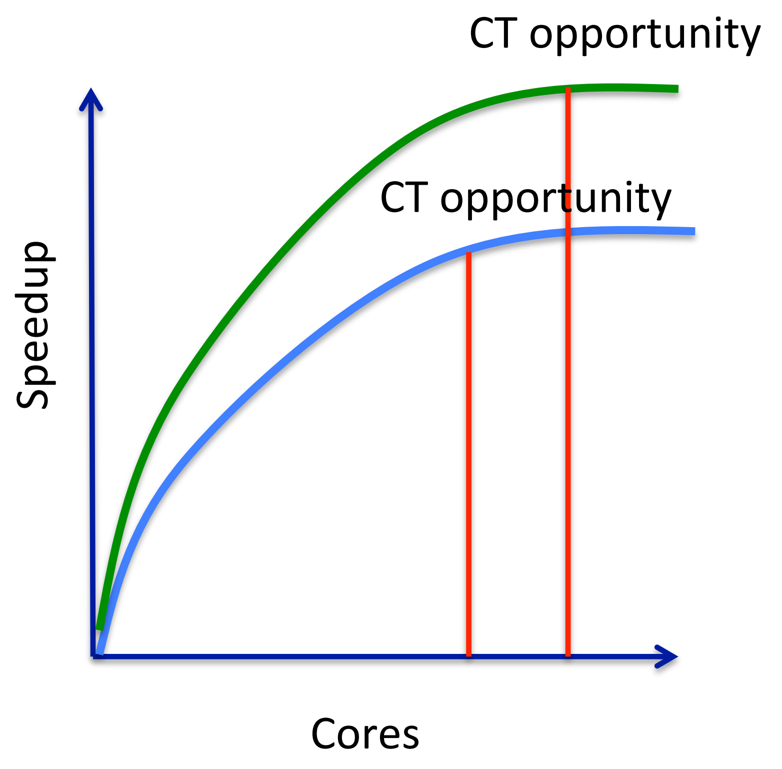
If parallel programs were scale-free

- Power increasing linearly with active cores
  - Previously dynamic, but now also static power
- Program speedup lines have knees
  - Synchronisation, contention for resources or the algorithm itself
  - Energy-efficient programs would execute at the beginning of the knee
  - How do we locate the knee?



Exploring the concurrency-power trade-off

- Programs execute distinct phases
  - Programmer annotated or auto-mined from time series of HPMs
  - Compute-, memory- or communication-bound
- Dynamic scalability predictors
  - Concurrency sweet spot detection with empirical modeling [ICS06]
  - Rigorous statistical modeling [TPDS08]
  - Machine learning approaches [EuroPar10]
  - MPI task aggregation [IPDPS10]

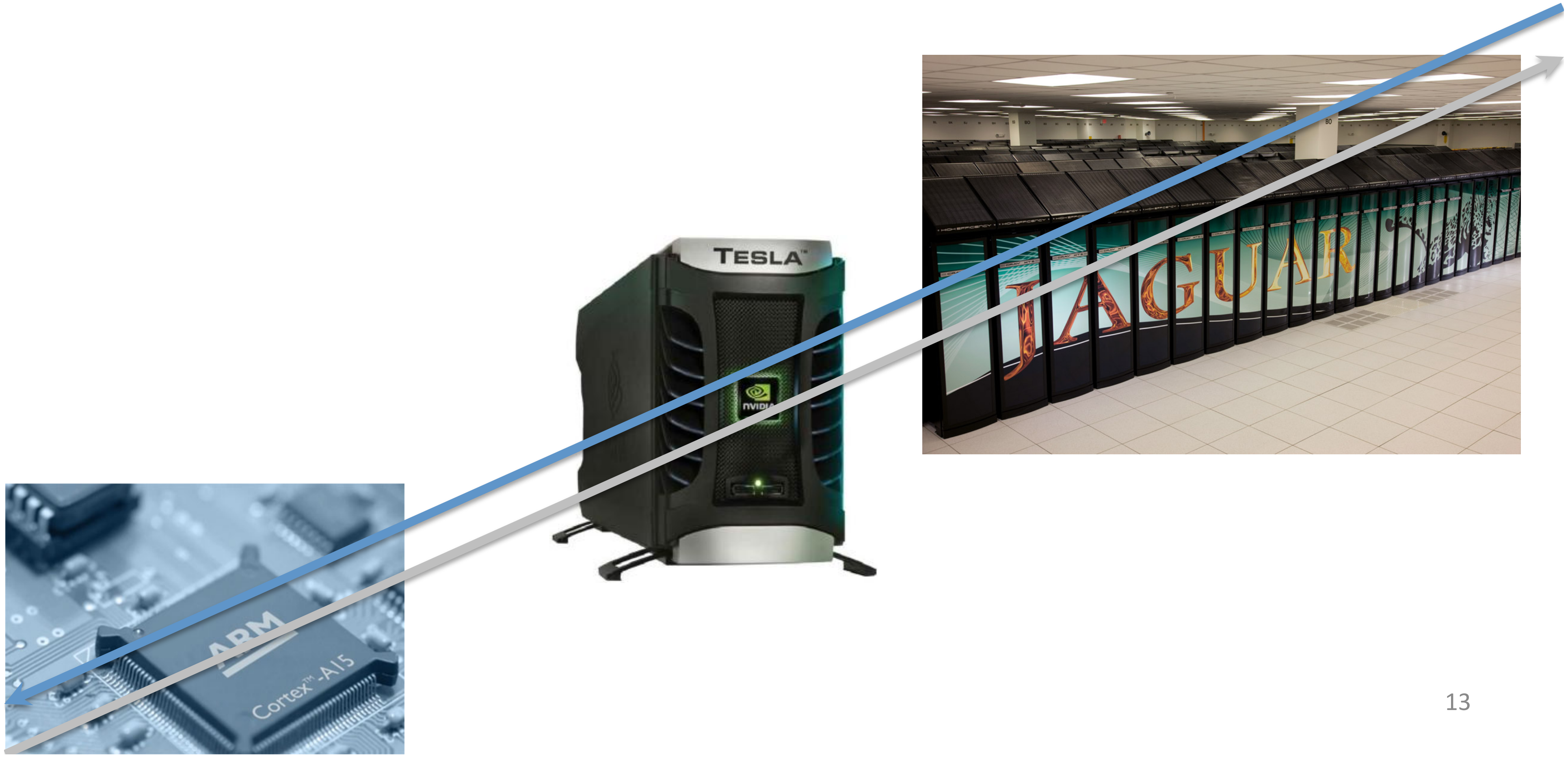


Queen's University Belfast

Scale-freedom improves energy-efficiency

Scale-free parallel programs can reduce their power budget

Scale-free parallel programs adapt to power caps

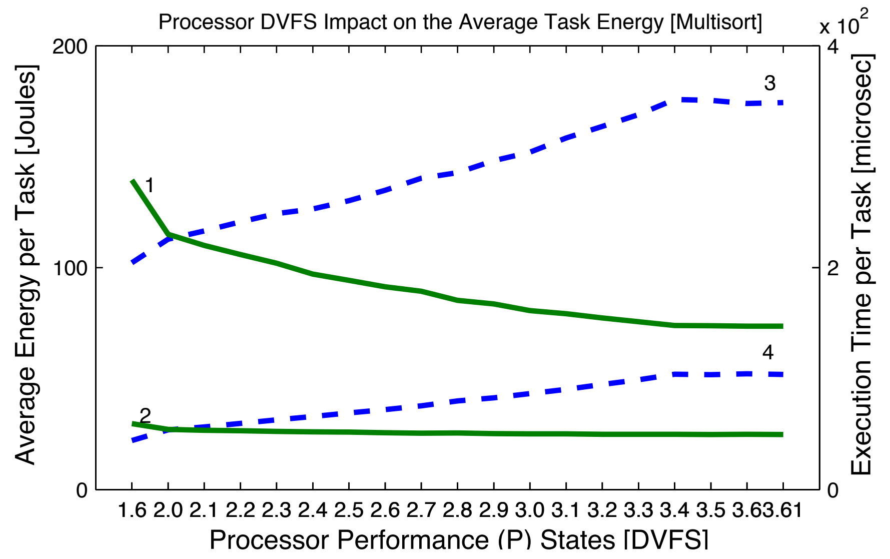


Is controlling concurrency enough?

Beyond scale freedom

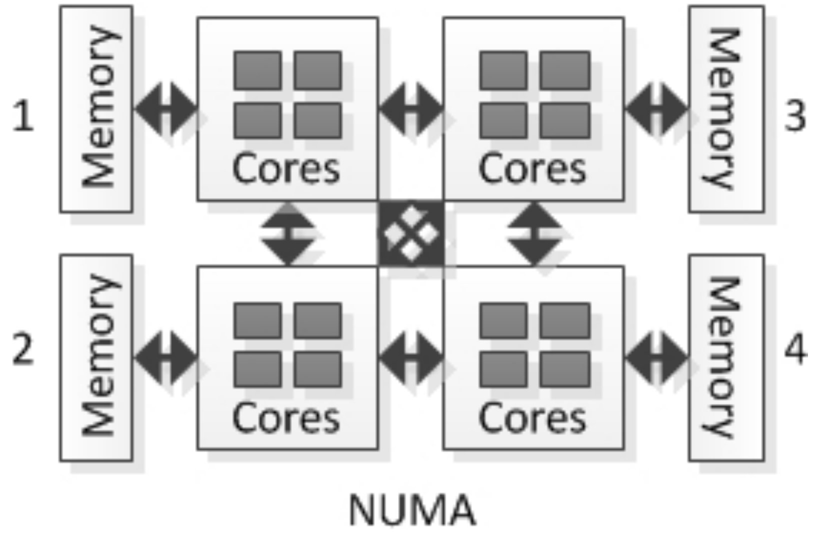
Multi-objective optimization [PACT08]:

- Control multiple power knobs at a fine granularity (per task, in microseconds)
- Applied to DCT, DVFS



Taming locality issues

Original DCT work failed to capture implications of thread migration



Example: Up to 45% execution time variation across 85 mappings

4, 4-core nodes: 43,680 mappings.

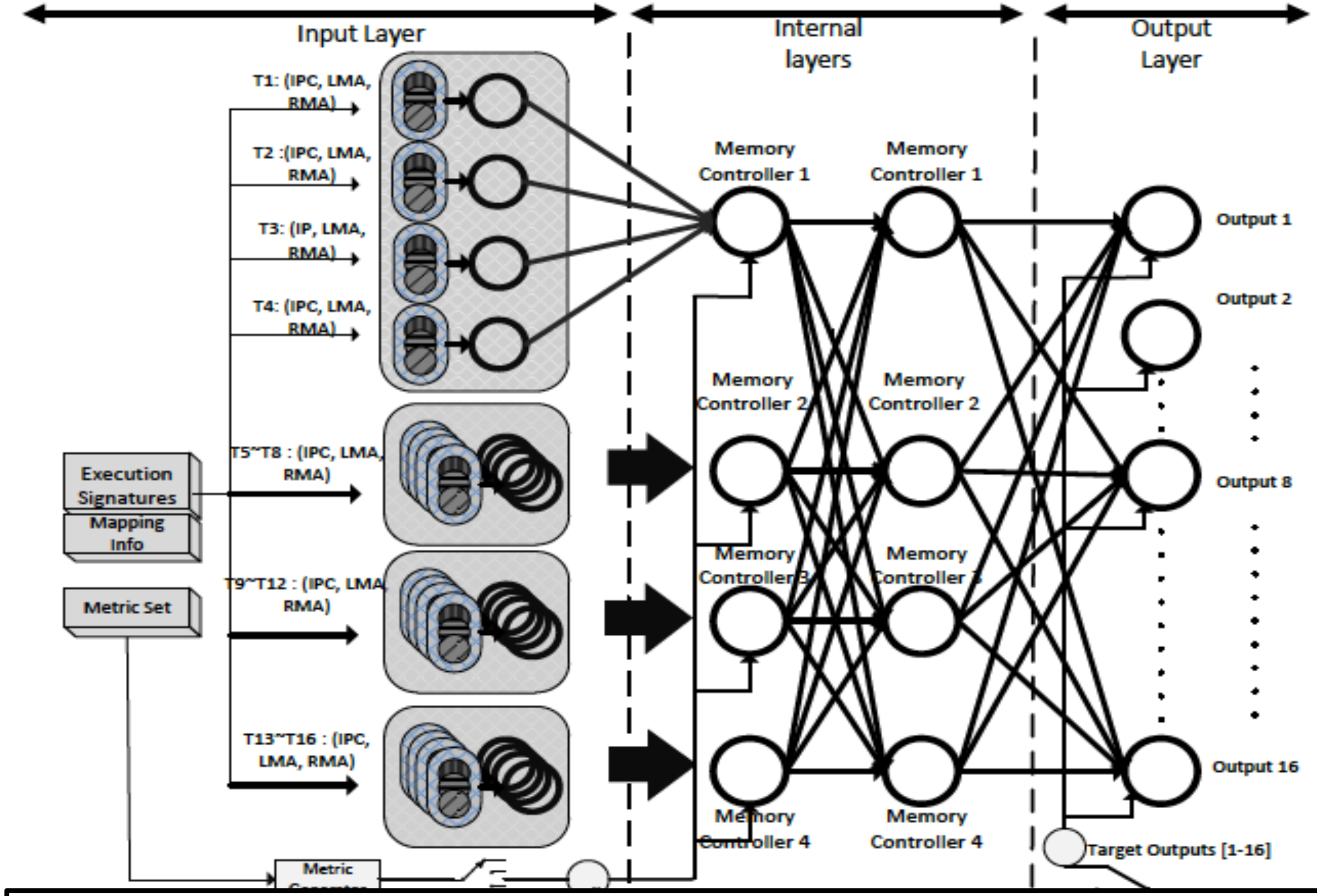
16, 4-core nodes: 63 million mappings.

1000, 4-core nodes: 10^{43} mappings.

Queen's University Belfast

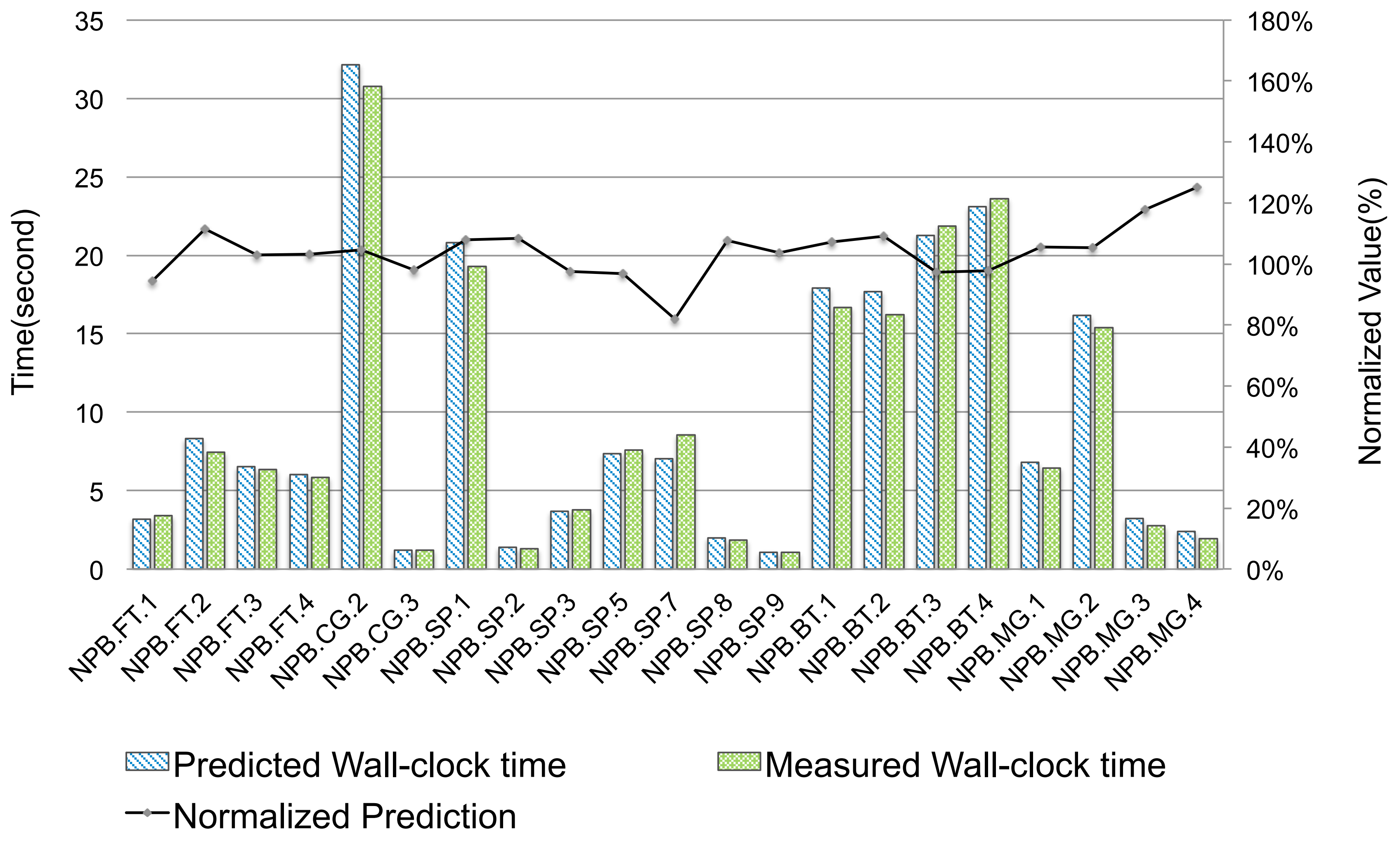
School of EEECS
HPDC Cluster

DyNUMA training using ANN

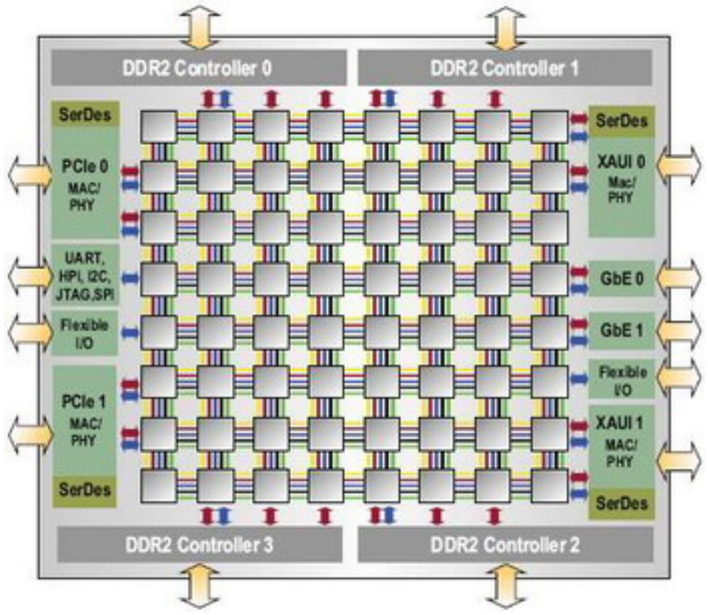


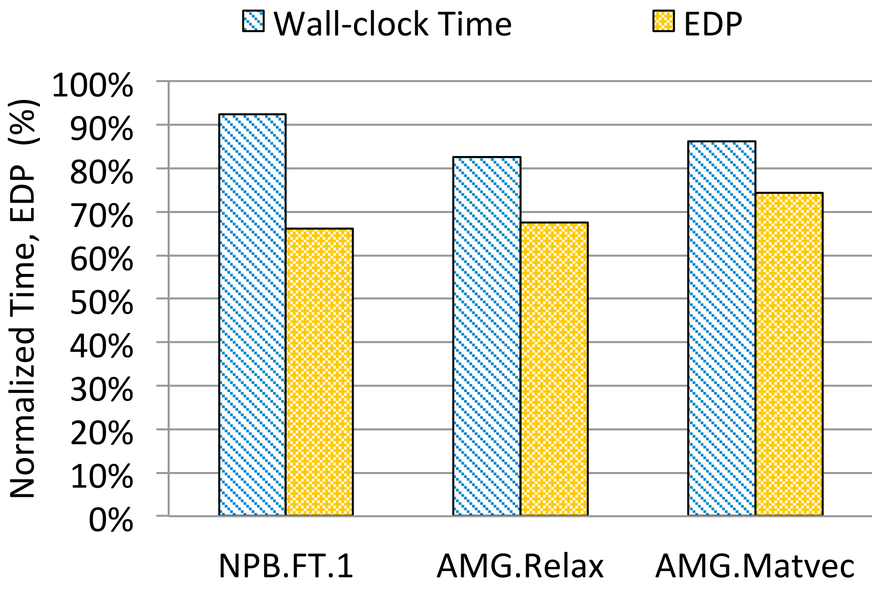
Optimize for concurrency, vertical and horizontal locality
[IISWC12, SIGMETRICS PER]

Modeling accuracy



Performance on TilePro64

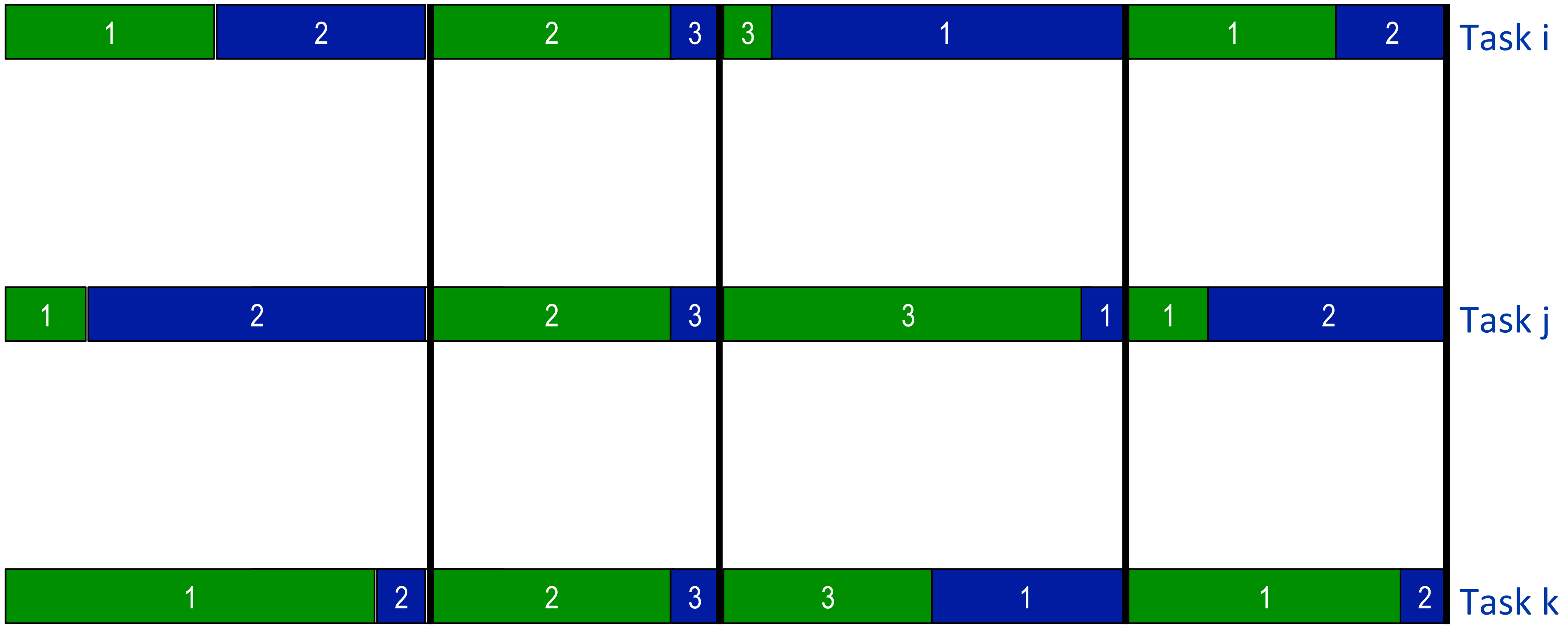




- Tile64Pro OS default Linux mapping is inefficient
- More concurrency does not necessary improve performance
- Counter-intuitive mappings optimize energy-efficiency

Is controlling concurrency, mapping and waste at one program level enough?

Energy-Aware Hybrid Programming



Slack dispersion algorithms [IPDPS10,TPDS13]

Critical path based modeling

Predicting time vs. predicting scaling function

$$t_i = \sum_{j=1}^{M} \min_{1 \le |thr| \le X \cdot Y} t_{i,j,thr}$$

$$t_c = \max_{1 \le i \le N} \sum_{j=1}^{M} \min_{1 \le |thr| \le X \cdot Y} t_{i,j,thr}$$

Time modeling enables slack dispersion

Slack dispersing DCT&DVFS [IPDPS10,TPDS13]

Use critical path time to determine slack (essentially imbalance)

$$\Delta t_i^{slack} = t_c - t_i - t_i^{comm} - t_{dvfs}$$

Time constraint:

$$\sum_{1 \le j \le M} \Delta t_{ijk} \le \Delta t_i^{slack}$$

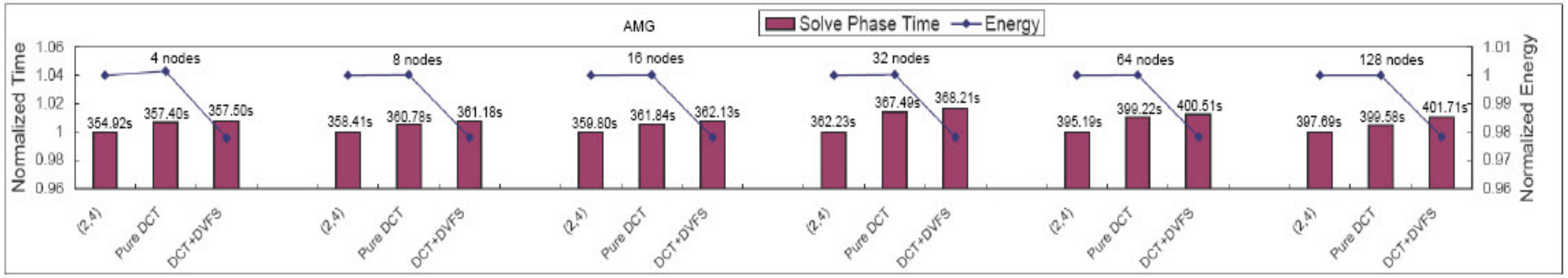
Energy constraint:

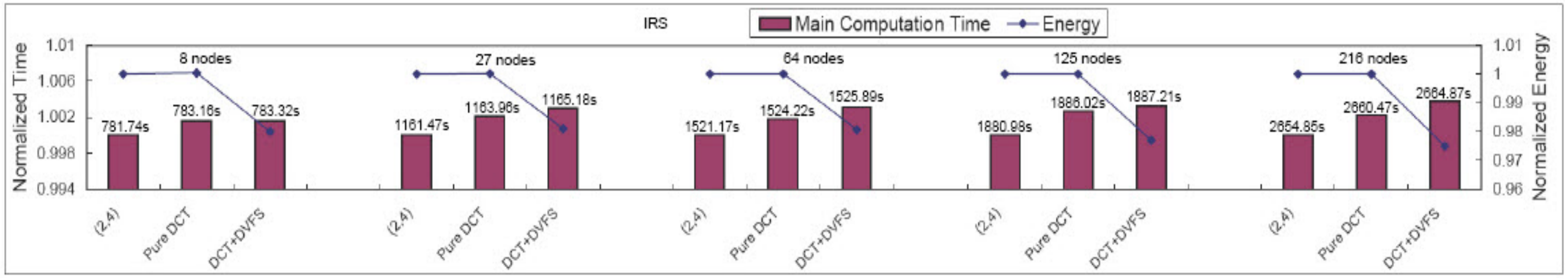
$$\sum_{1 \le j \le M} t_{ijk} f_k \le t_i \, f_0$$

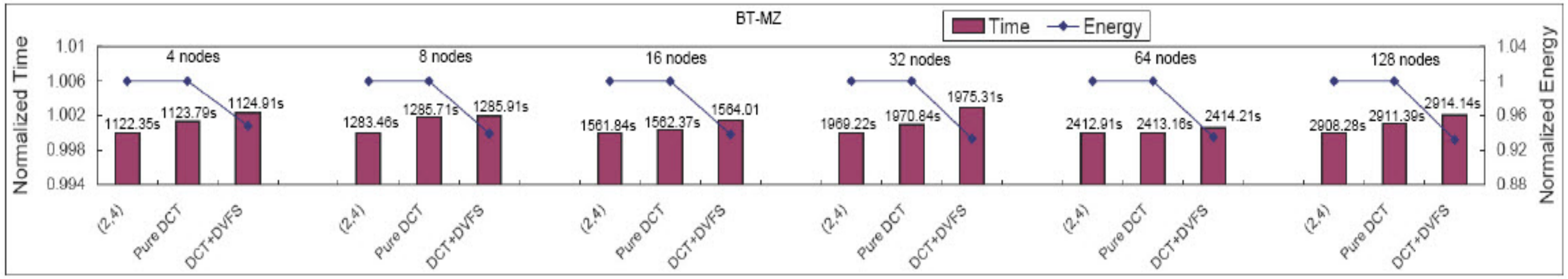
Queen's University Belfast

School of EEECS
HPDC Cluster

Performance Evaluation



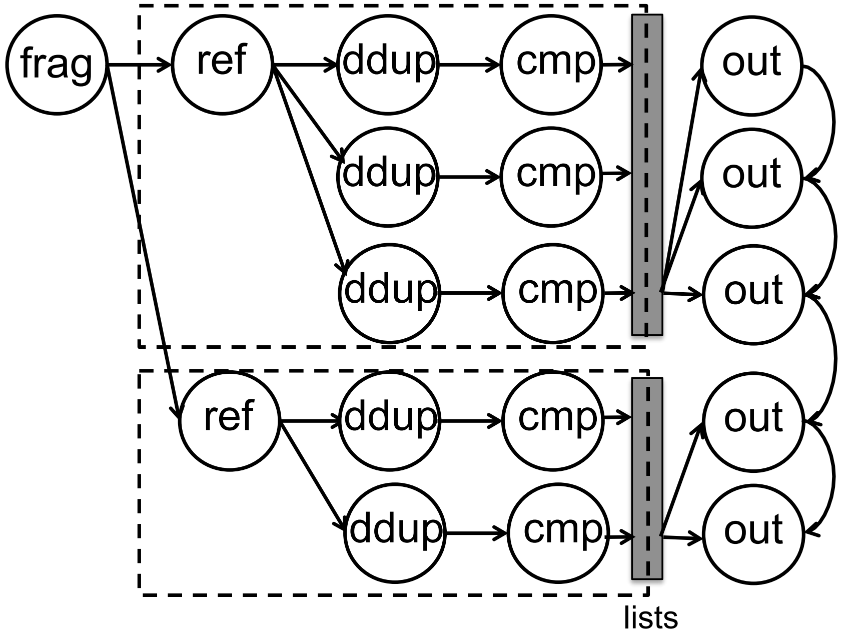




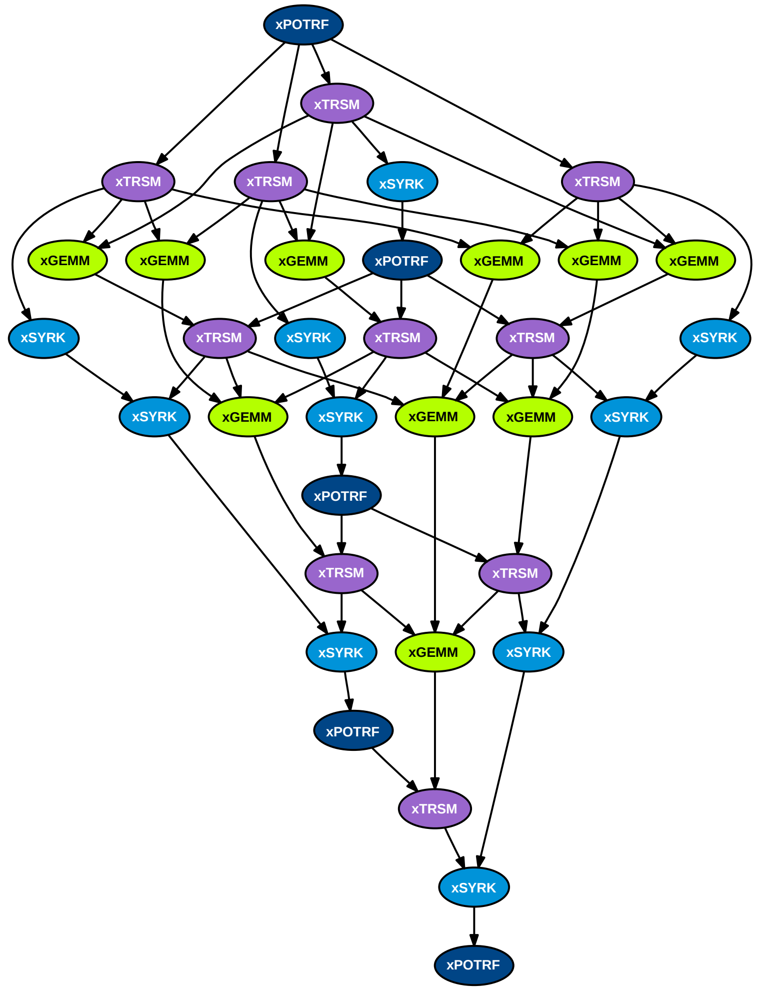
Consistently significant energy savings with weak and strong scaling

Have we solved our problems?

How much parallelism is (really) there ?



(a) Nested pipelines



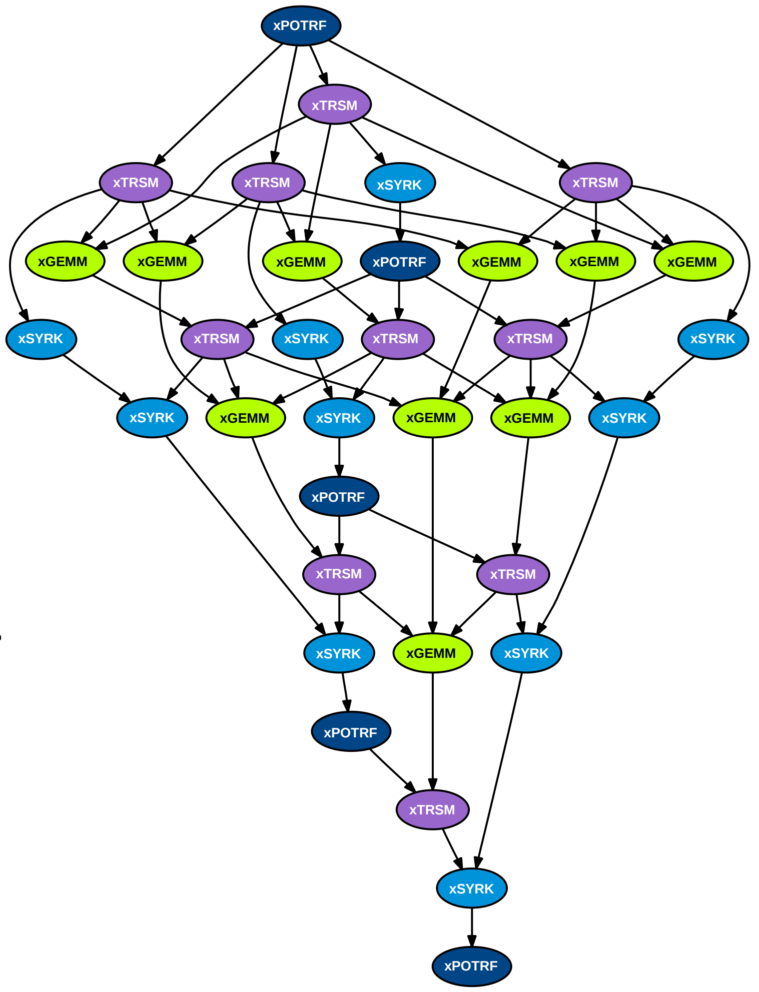
© Ltaief et al., LAPACK Note #223

Emerging scale-free programming models

- Annotate task memory footprint and side-effect

  `input (rd-only), inout (rw), output (wr-only)`

- Discover task dependences at run-time
  - dynamically extract task parallelism
  - schedule tasks out-of-order
  - E.g. "depend" clause in OpenMP 4.0 RC2 (March 2013)

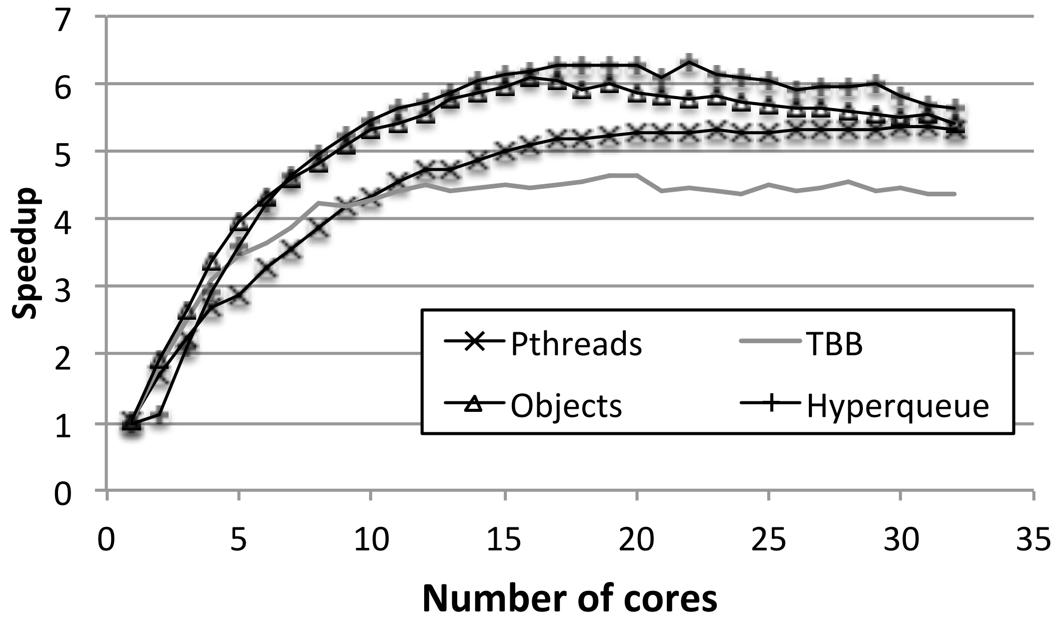


© Ltaief et al., LAPACK Note #223

Better concurrency control saves energy

```
struct data { ... };
void consumer(popdep<data> queue) {
    while( !queue.empty() ) {
        data d = queue.pop();
        // ... operate on data ...
    }
}
void producer(pushdep<data> queue, int start, int end) {
    if( end-start <= 10 ) {
        for( int n=start; n < end; ++n ) {
            data d = f(n);
            queue.push(d);
        }
    } else {
        spawn producer(queue, start, (start+end)/2);
        spawn producer(queue, (start+end)/2, end);
        sync;
    }
}
void pipeline(int total) {
    hyperqueue<data> queue;
    spawn producer((pushdep<data>)queue, 0, total);
    spawn consumer((popdep<data>)queue);
    sync;
}
```

Swan [PACT11] Hyperqueues express and control data-dependent parallelism in variable-rate pipelines [SC13]



Task dataflow and locality

- Rich semantic information available to the compiler and runtime system
  - DAG, program order for correctness and determinism, task memory footprints for locality
- Opportunity to make memory system aware of working sets
  - Runtime explicitly manages the memory hierarchy by placing task footprints in caches

Overlooking the memory hierarchy

[SBAC-PAD'12]

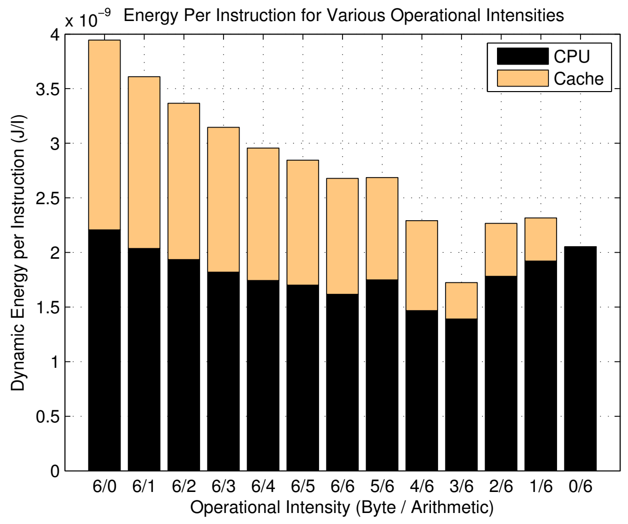


Figure: EPI while traversing OI of a L3 Cache sensitive workload.

Observations

1. OI Counts L3 accesses instead of memory ones.
2. L3 accesses also degrade energy efficiency for high OI.
3. Cache Hierarcy consumes up to **50% of the total energy.**

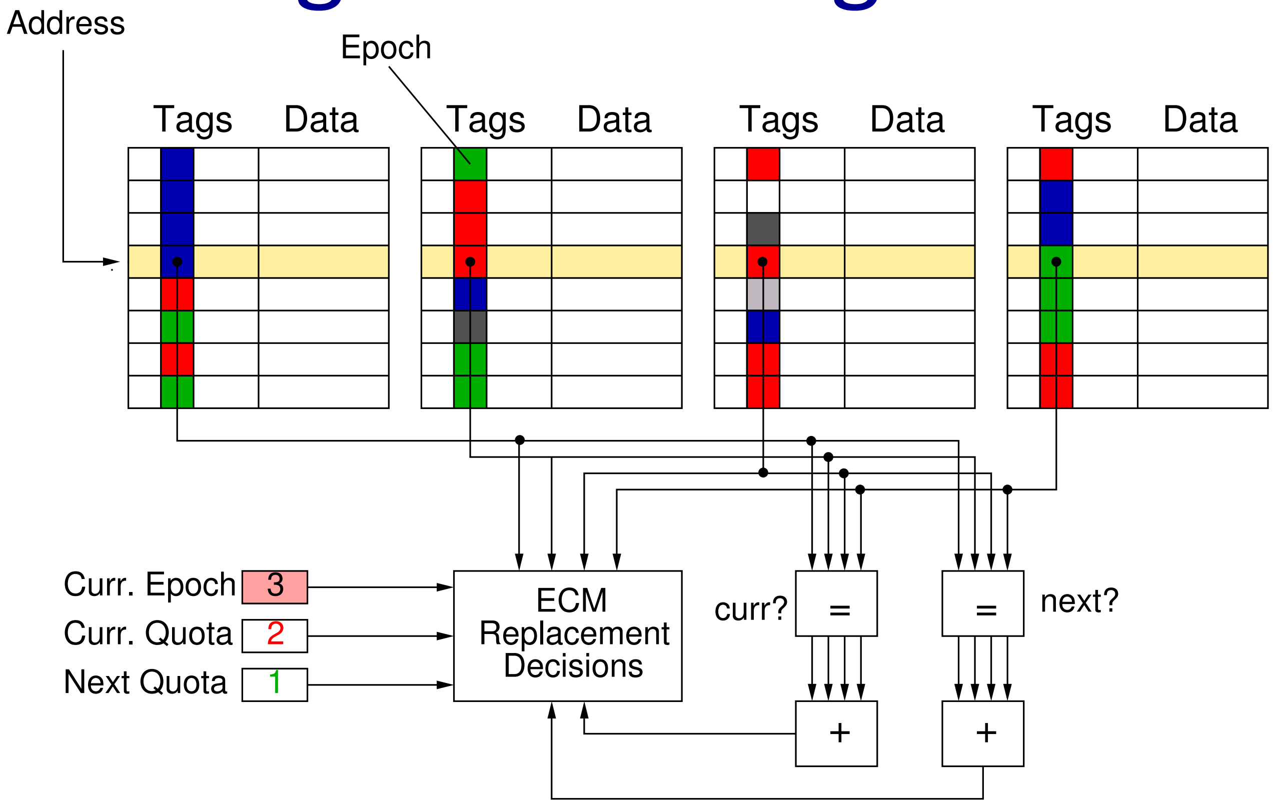
Overlooking the memory hierarchy

Workload	**OI**	**EPI**	**Against L3**
L3	High	$3.9\times^{-9}$	1
Throughput	High	$1.18\times^{-8}$	3.02
Latency	High	$5.8\times^{-8}$	14.9
L3	Low	$2.4\times^{-9}$	1
Throughput	Low	$4.0\times^{-9}$	1.6
Latency	Low	$3.6\times^{-8}$	15

Table: EPI comparison of throughput, latency and L3 sensitive workloads.

Cache management using task lifetimes

[ICS13]

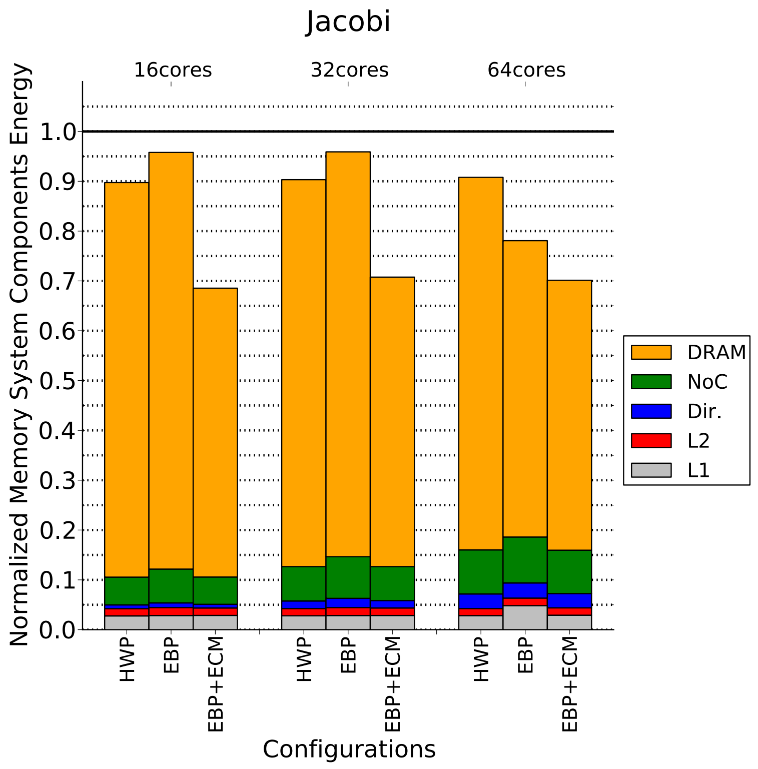


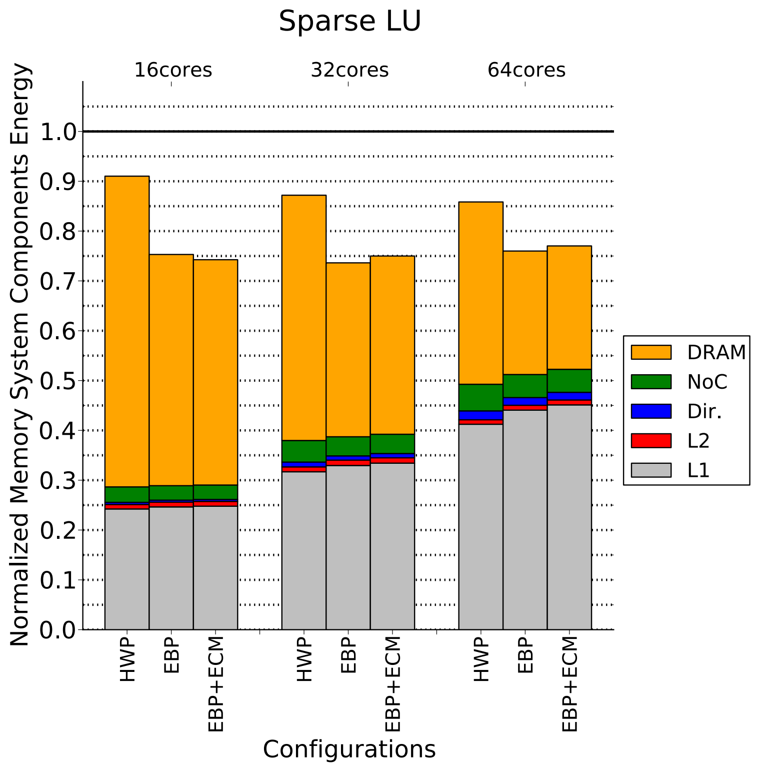
- Epoch quotas: cache space allocation per task (best-effort)
  - SW declares quota from task footprint size (ECM converts to ways)
  - when current and next compete ➔ guarantee minimum allocation
- Replacement: computes current & next occupancy (per-set)
  - replace from requesting epoch when set is full (e.g. use LRU bits)
  - throttle EBP (prefetching) when set is full and epoch exceeded quota.

Queen's University Belfast

School of EEECS
HPDC Cluster

Better locality cuts down energy consumption





Jacobi, Sparse-LU: memory-intensive codes, medium or low OI

Energy savings of 20%-30%

Joe the ~~Plumber~~
Green Programmer

Should the programmer care about energy-efficiency?

Energy and the programmer

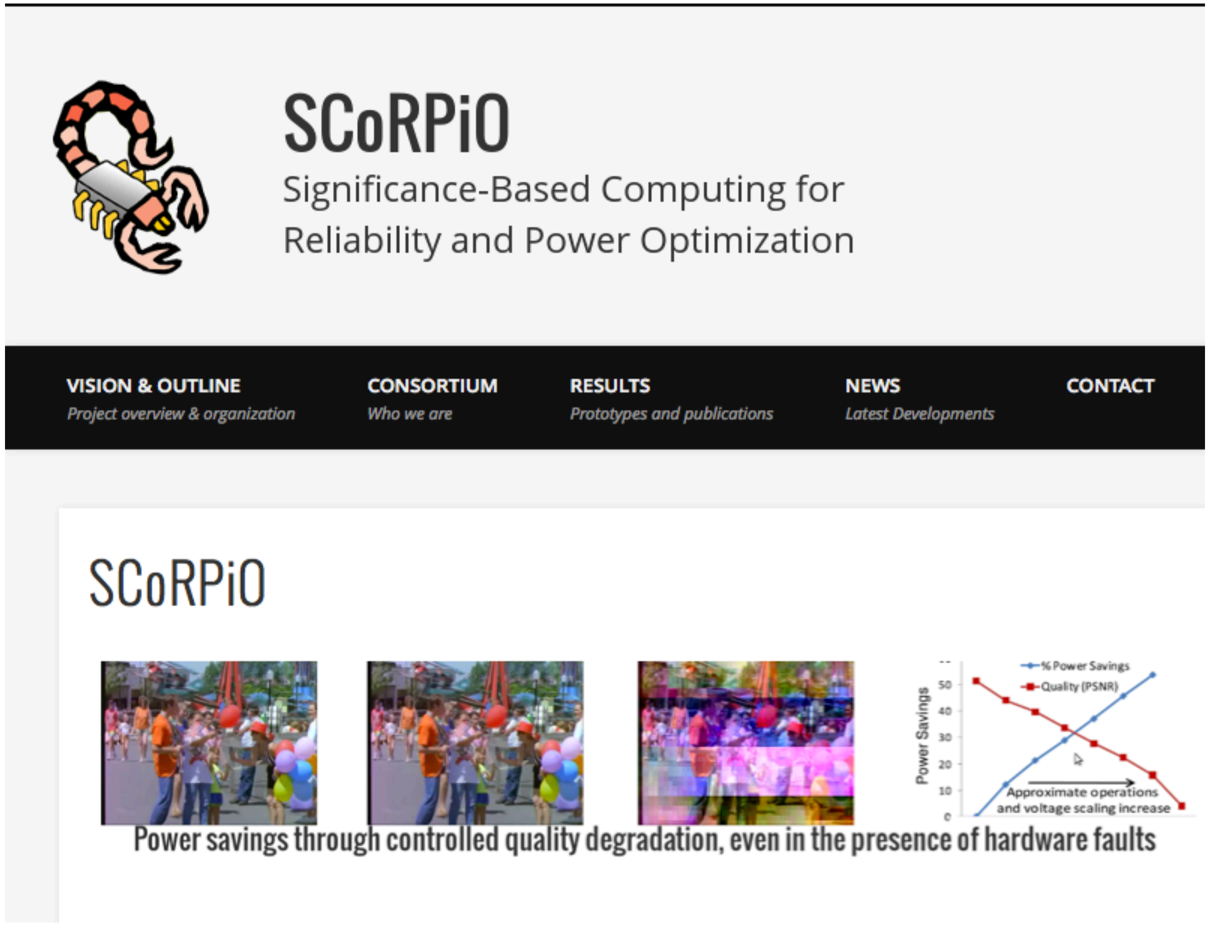
- Should programmers go back to half a century--old principles?
  - Eliminate waste
  - Much of the programming we do already does this
    - Load balancing
    - Communication or synchronization removal
- Scale-free programming models can help programmers achieve this
  - Programmer expresses exact parallelism and locality patterns
  - Runtime system maps to cores, memories and interconnect so as to avoid waste
  - Component-level power management further minimizes waste

The "Lernaia Hydra"

- Power instrumentation is inaccurate, intrusive, coarse-grain,…
  - Software is at the mercy of hardware (PMCs, sensors, voltage regulators, everything machine-specific,…)
- No software standards for power measurement and management
  - How would power knobs make it into MPI, OpenMP, Cilk, PGAS, or even mainstream languages?
- What if a power cap is imposed?
  - And violated?
- Riding the technology curve is dangerous
  - Low voltage may become sub-threshold voltage
  - Subthreshold voltage will increase soft error rate
  - Soft errors will cause failures

Looking forward: SCoRPiO project

- Computing at the limits of energy and reliability
  - Components with sub-threshold voltage
- Embrace uncertainty!
  - Not all bits in memory and registers are equally critical
  - Application-specific quality control
- Minimize power by scaling gracefully under hardware errors
  - Scale-free parallel programming



SCoRPiO
Significance-Based Computing for Reliability and Power Optimization

SCoRPiO

Power savings through controlled quality degradation, even in the presence of hardware faults

Acknowledgments

Our resources

EPSRC
Pioneering research and skills

DEPARTMENT OF ENERGY · UNITED STATES OF AMERICA ·

NSF

PEOPLE

IBM®

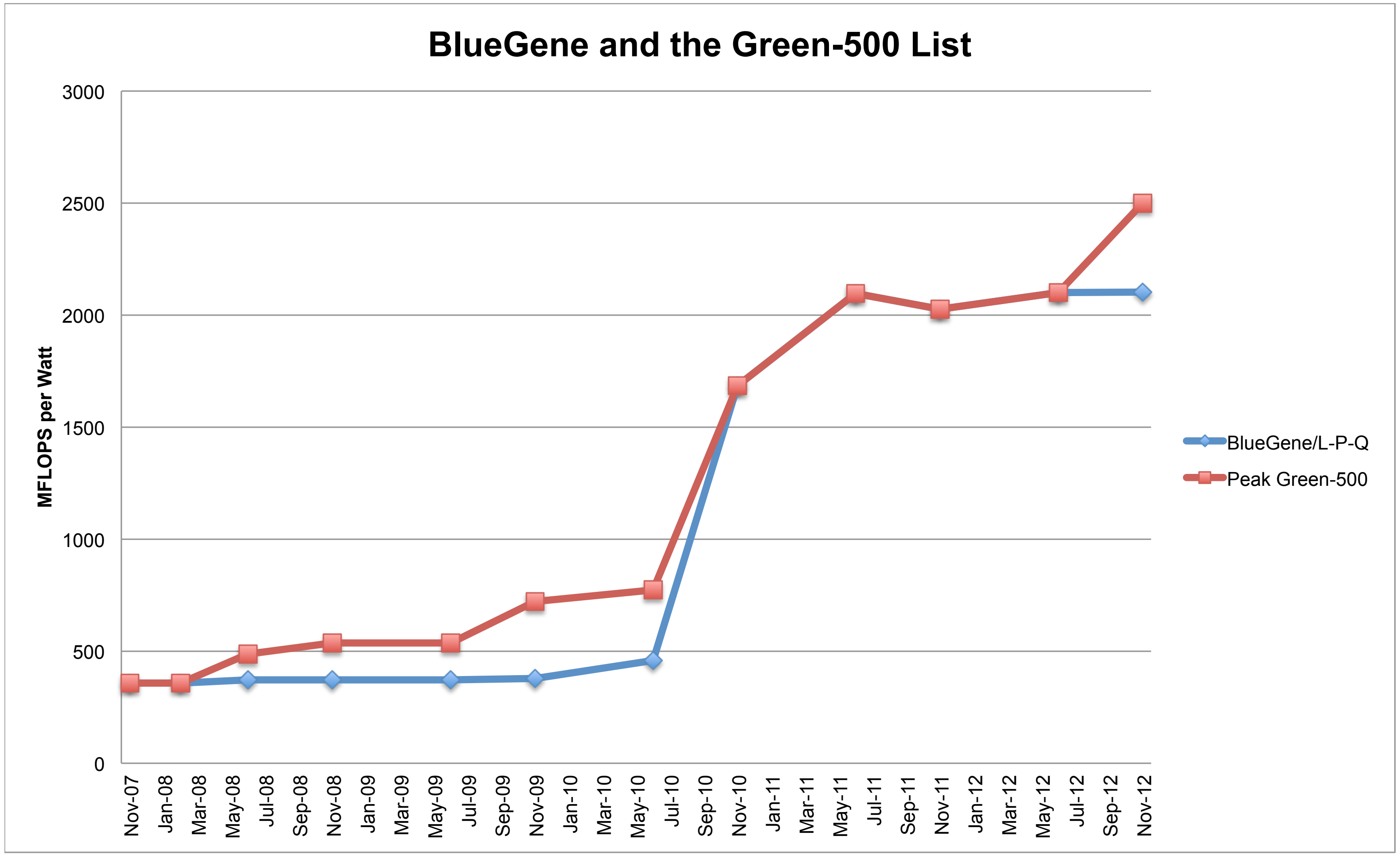
COOPERATION

Lawrence Livermore National Laboratory

More information

http://www.qub.ac.uk/research-centres/HPDC/

BlueGene on the Green500


BlueGene and the Green-500 List
MFLOPS per Watt
Nov-07
Jan-08
Mar-08
May-08
Jul-08
Sep-08
Nov-08
Jan-09
Mar-09
May-09
Jul-09
Sep-09
Nov-09
Jan-10
Mar-10
May-10
Jul-10
Sep-10
Nov-10
Jan-11
Mar-11
May-11
Jul-11
Sep-11
Nov-11
Jan-12
Mar-12
May-12
Jul-12
Sep-12
Nov-12
BlueGene/L-P-Q
Peak Green-500