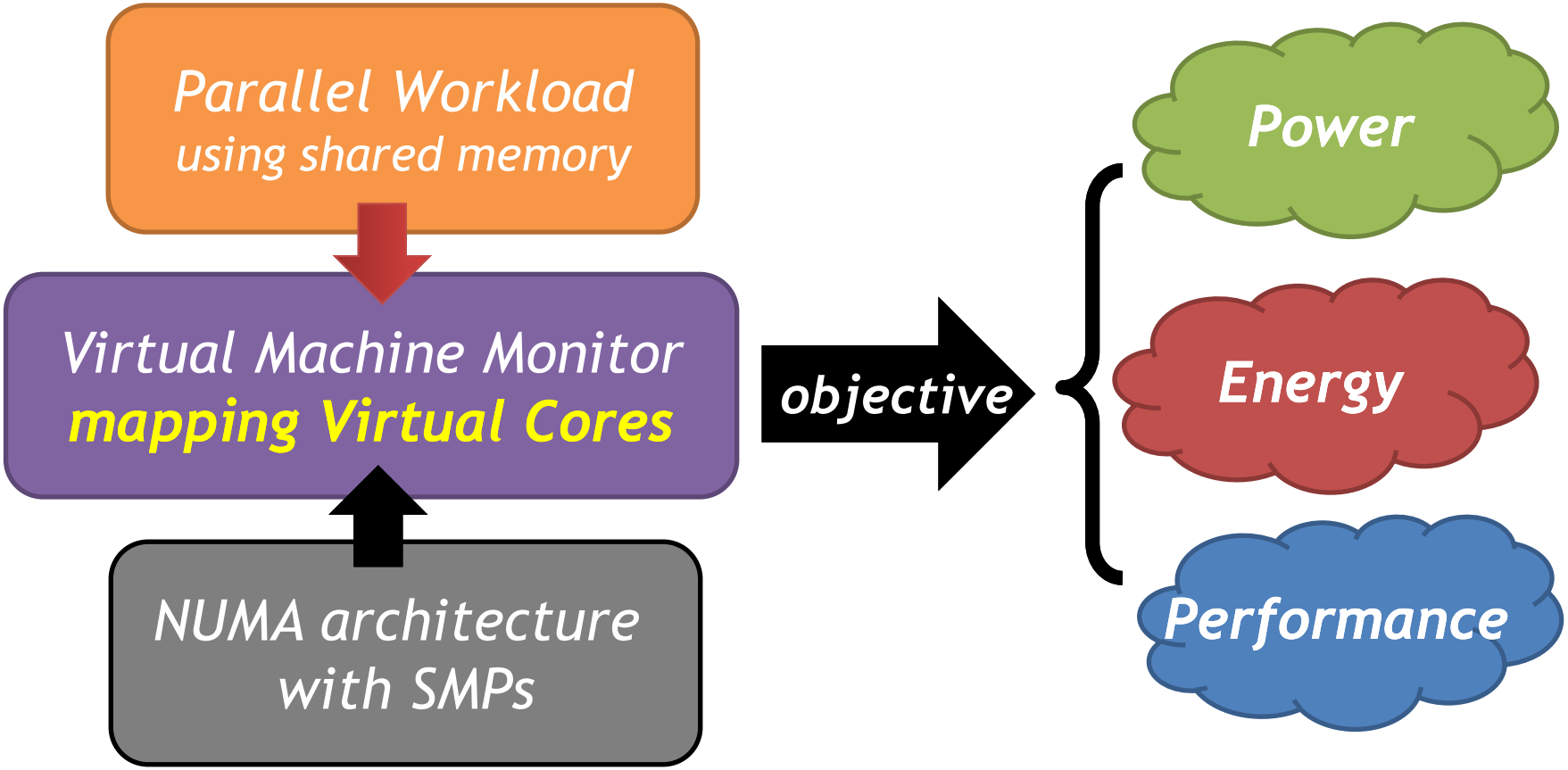
Dynamic Adaptive Virtual Core Mapping to Improve Power, Energy, and Performance in Multi-socket Multicores

Chang Bae, Lei Xia, Peter Dinda, John Lange

Prescience Lab, Dept. of EECS, Northwestern Univ.

Dept. of CS, Univ. of Pittsburgh

Virtual cores mapping problem in NUMA architecture



Contribution of the work

- **Identify virtual core optimization opportunity**
  - With two virtual core (vcore) mappings
  - Trade-offs in power, energy, and performance

Contribution of the work

- Identify virtual core optimization opportunity

- **A new HW assisted SW detection mechanism**
  - Detects a new set of metrics
  - Observes shared memory reference behaviors

Contribution of the work

- Identify virtual core optimization opportunity
- A new HW assisted SW detection mechanism
- **Design, implementation, and evaluation of adaptive system**
  - Incorporates the proposed control algorithm
  - Results in
    - Boosting performance ***up to 66%***
    - Minimizing energy ***up to 31%***
    - Minimizing average power ***up to 17%***
    - With ***<0.05% overhead***

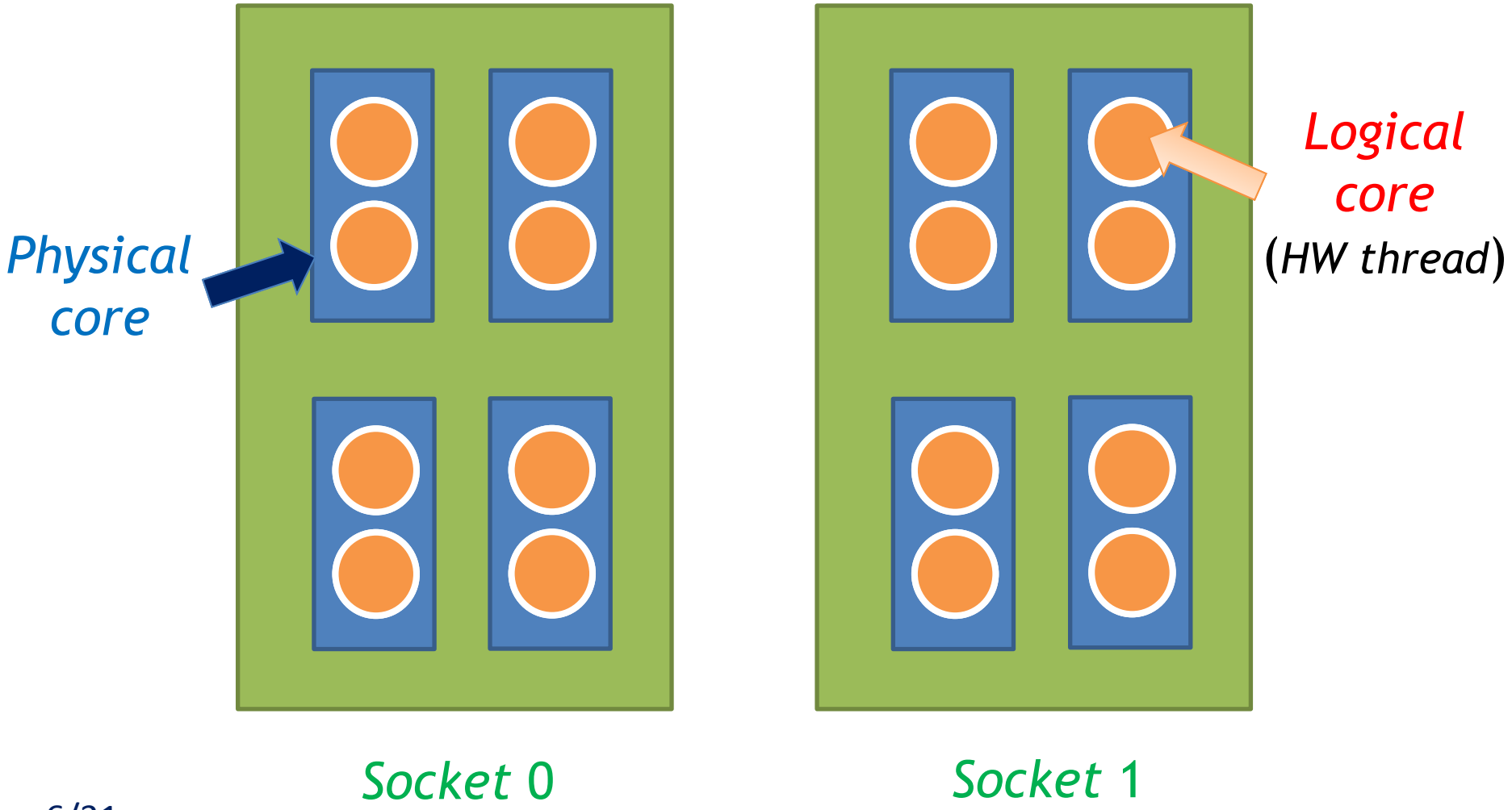
Outline

- Opportunities in virtual core (vcore) mapping
- Metrics and measurement
- System
- Results
- Conclusion

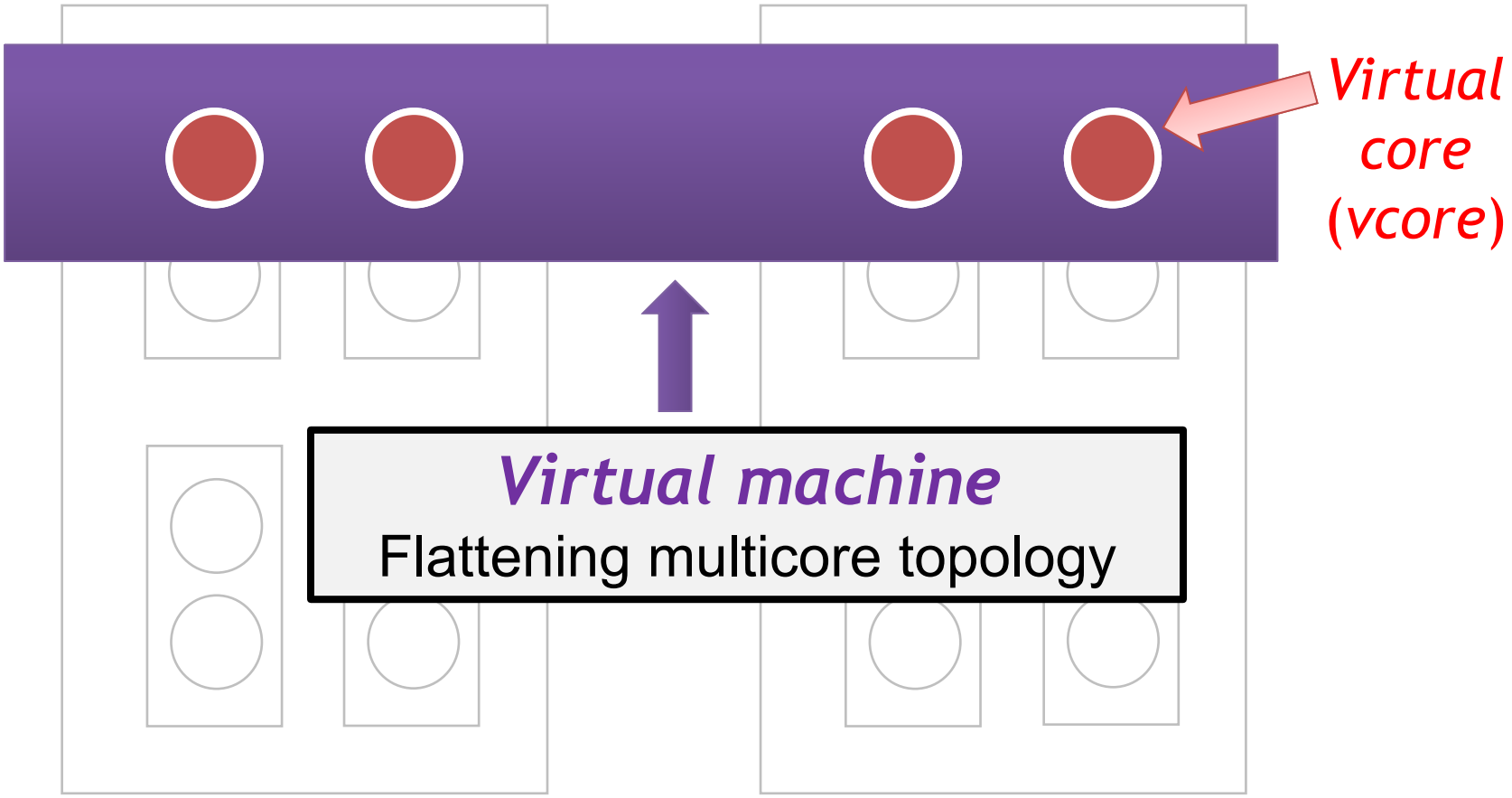
Outline

- **Opportunities in vcore mapping**
  - Virtualized multi-core processors
  - Trade-offs in energy, power, and performance
- Metrics and measurement
- System
- Results
- Conclusion

NUMA architecture with SMPs

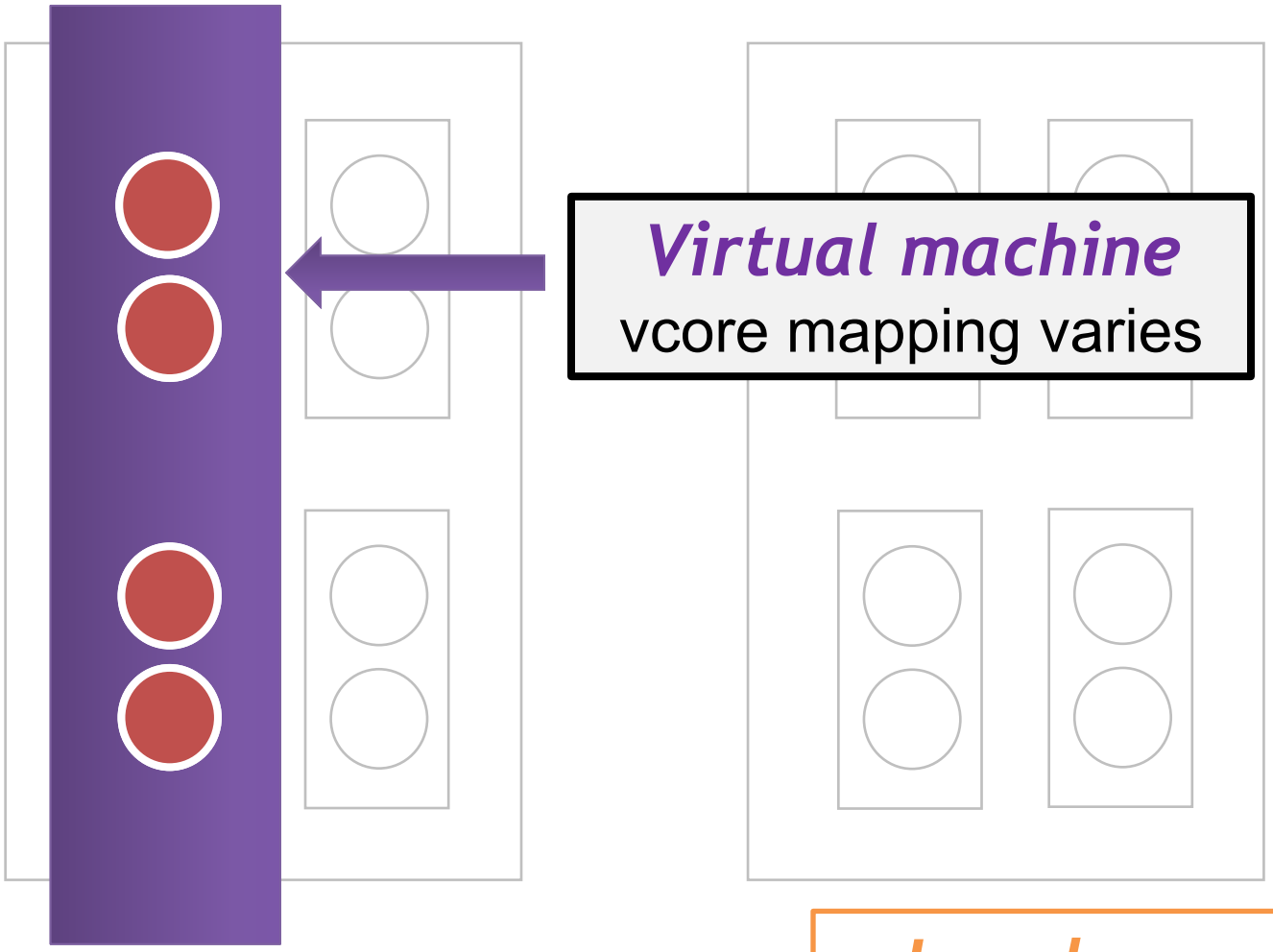


Virtualized multi-core processors



Interleaved mapping

Virtualized multi-core processors



vcore mapping strategy

- *Local* mapping
  - Aggregating vcores locations within a socket
- *Interleaved* mapping
  - Spreading vcores across multiple sockets

vcore mapping trade-offs

	Local	*Interleaved*
Cache contention	Worse	Better
Cache coherency cost	Better	Worse
DRAM access time	Better	Worse
Power	Better	Worse

Palacios VMM

- OS-independent embeddable virtual machine monitor

- Open source and freely available
- Virtualization layer for multiple OSs (Linux and Kitten)

- Successfully used on supercomputers, clusters (Infiniband and Ethernet), and servers

Palacios

An OS Independent Embeddable VMM

http://www.v3vee.org/palacios

Application benchmarks

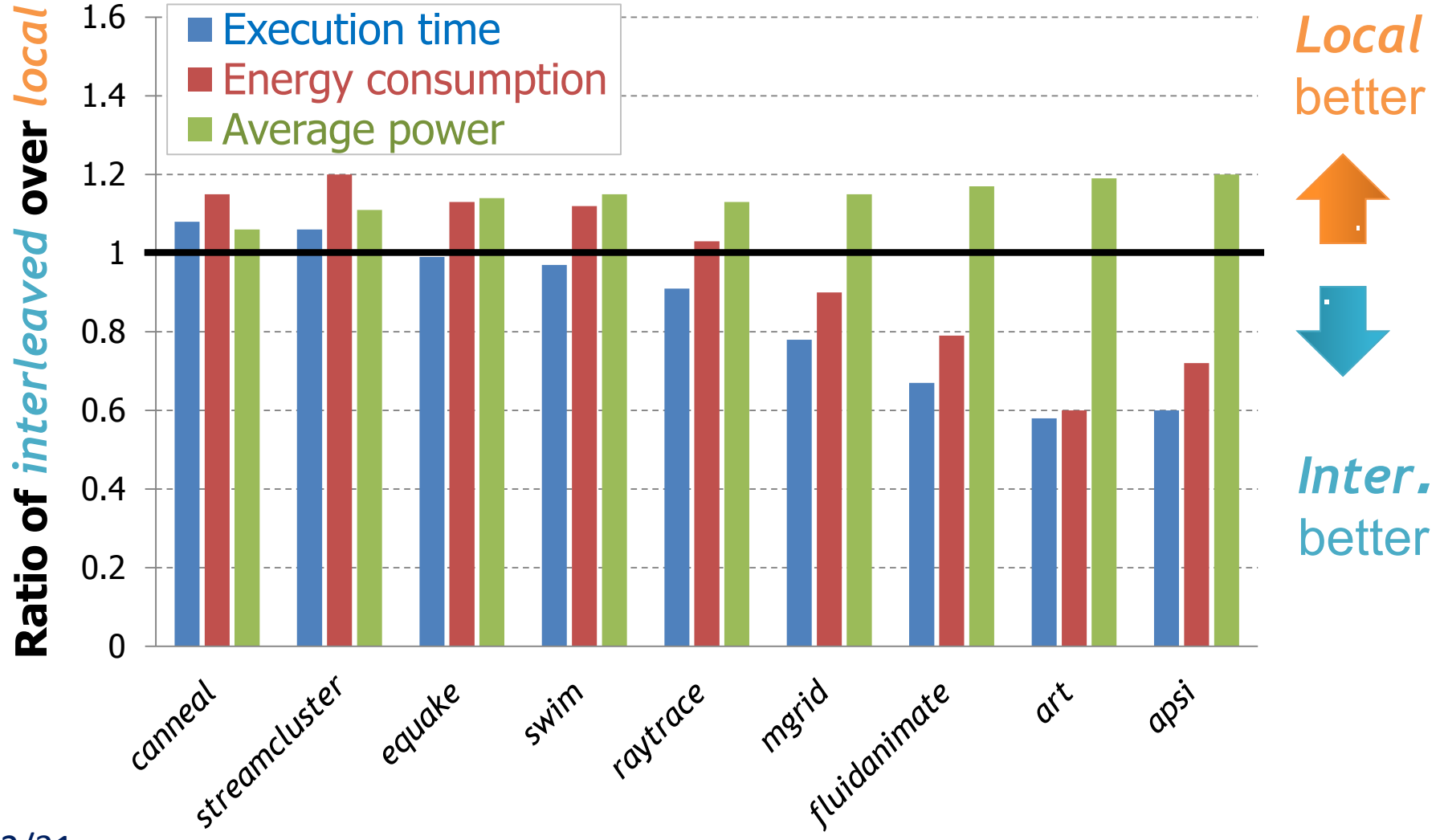
- SPEC OMP 2001[1]
- PARSEC 2.1[2]

- Widely used and *representative* workloads
- This talk focuses on benchmarks with *the greatest variations* in results

[1] SPEC CPU Benchmark Suites
www.spec.org/omp

[2] PARSEC Benchmark Suite
parsec.cs.princeton.edu

Optimization opportunities in vcore mapping for various objectives



Outline

- Opportunities in vcore mapping
- **Metrics and measurement**
  - Selection of metrics; about the metrics
  - Measurement mechanism
- System
- Results
- Conclusion

Considerations in selecting new metrics

- Architectural analysis
  - Captures shared memory traffic
- Measurable in a VMM
  - Page granularity
- Minimally correlated set
  - Correlation of each pair of metrics
  - Drop metrics with high correlation

Technique works on all current processors; Future chips will provide PMU measurement of off-chip traffic which this work can also leverage

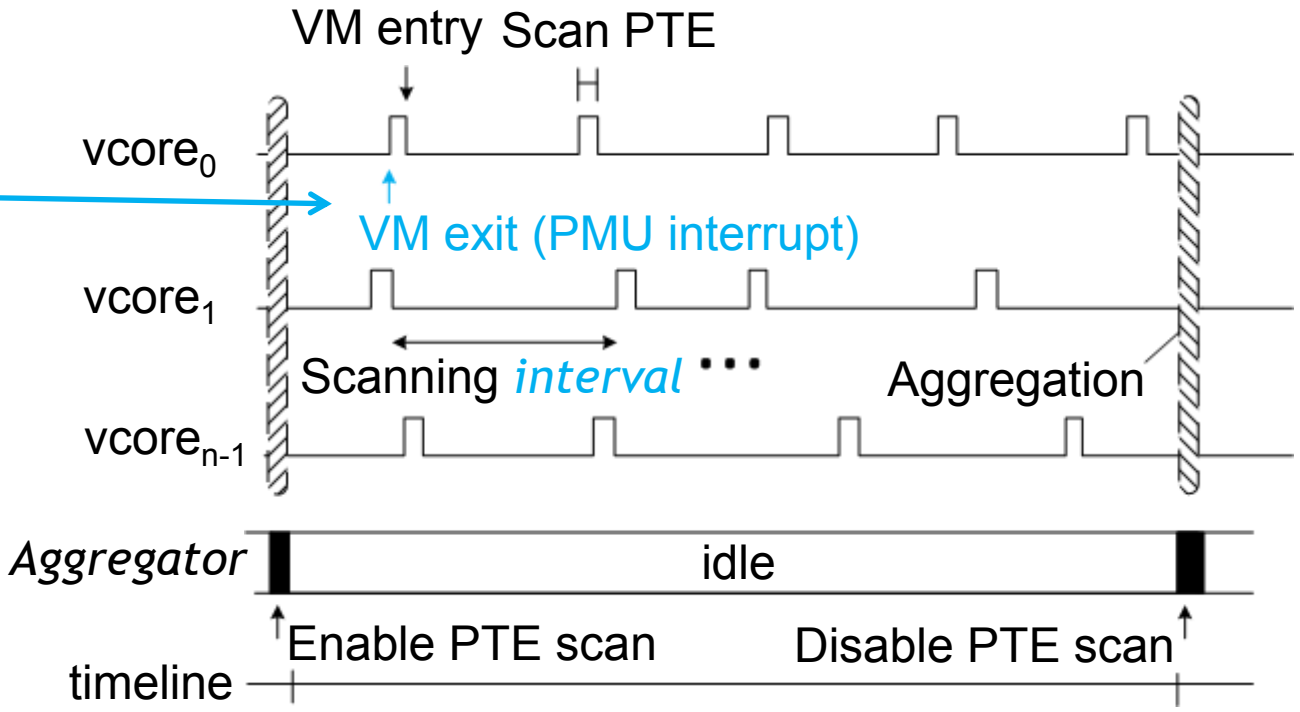
Selected metrics

- Pages with memory load from all vcores
  - Average page access rate *per memory* or *write op*
- Pages with memory store from all vcores
  - Average page write rate *per memory* or *write op*
- Degree of read or write sharing
  - Shared page access ratio *per memory* or *write op*
- Degree of write sharing
  - Shared page write ratio *per memory* or *write op*

Flow of measurement mechanism

In each vcore,

(1) Hardware triggers the ***interval*** after a specified number of memory ops or write ops

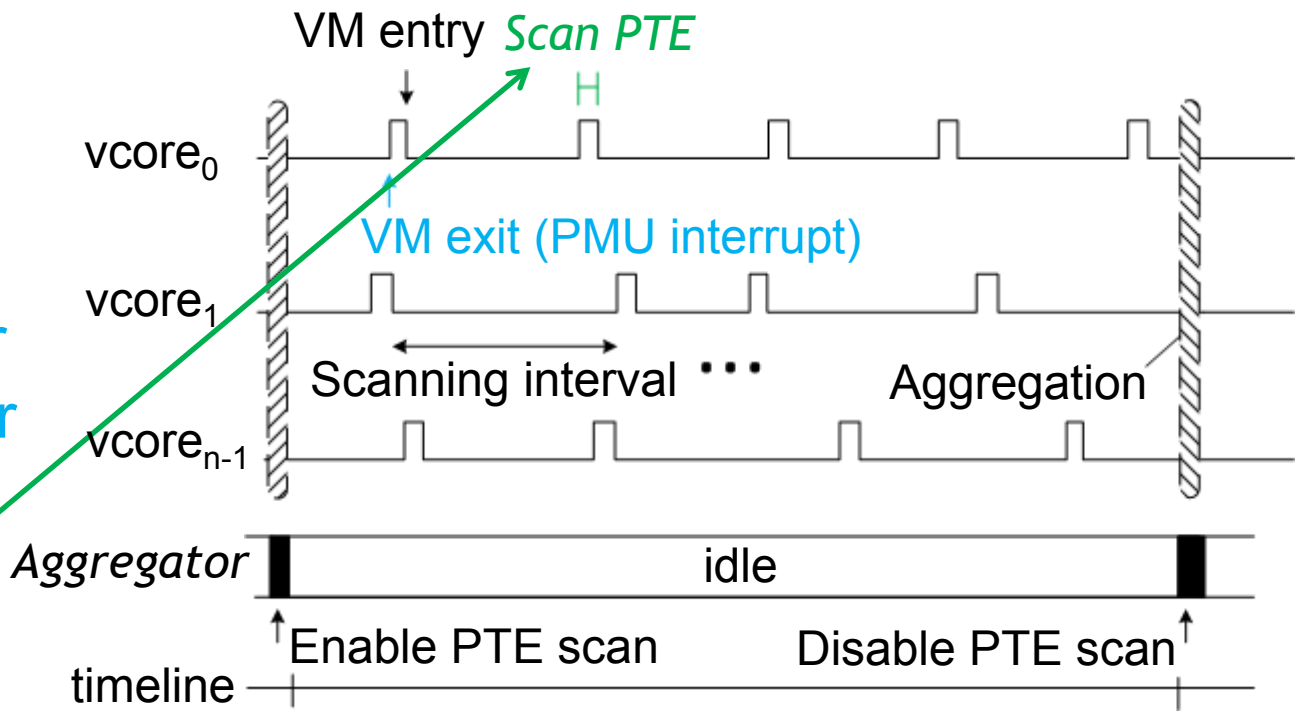


Flow of measurement mechanism

In each vcore,

(1) Hardware triggers the ***interval*** after a specified number of memory ops or write ops

(2) VMM ***scans page table*** to find page access or page writes generating bitmap(s)

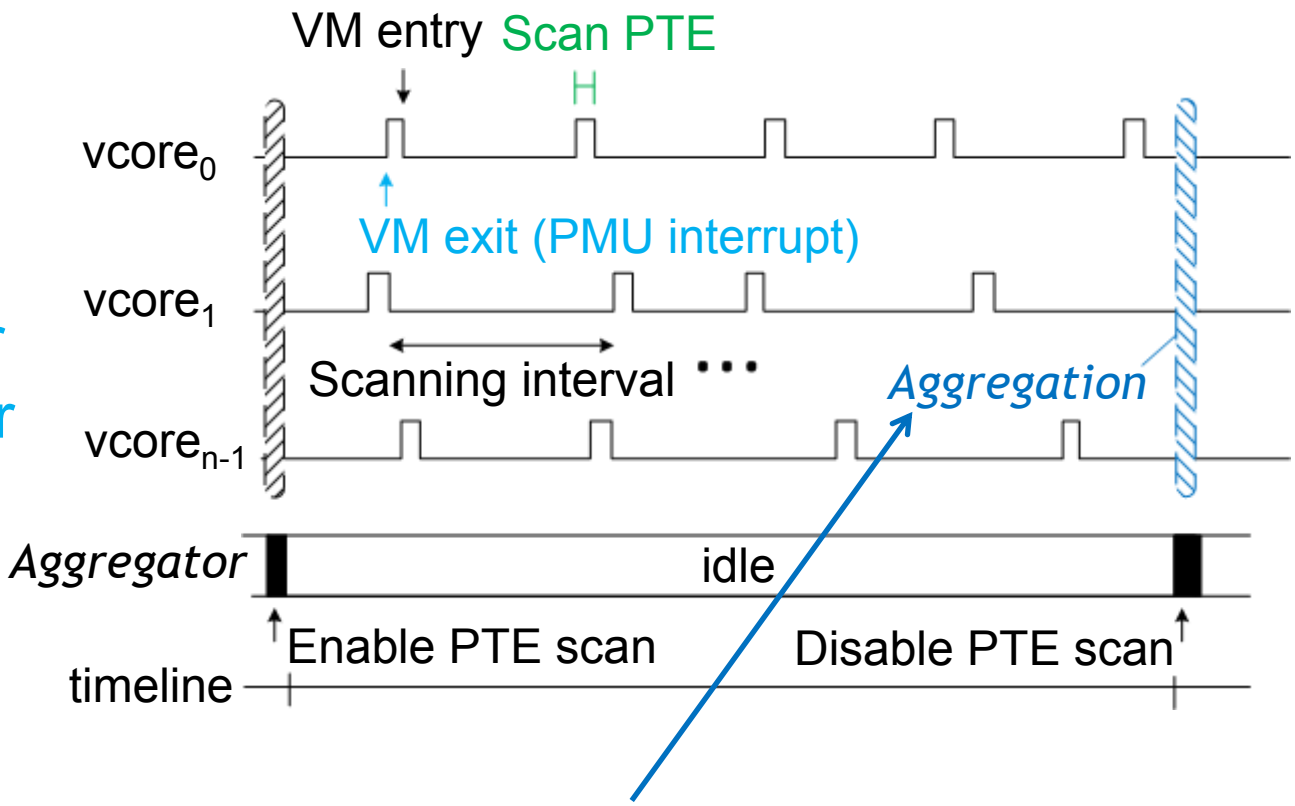


Flow of measurement mechanism

In each vcore,

(1) Hardware triggers the ***interval*** after a specified number of memory ops or write ops

(2) VMM ***scans page table*** to find page access or page writes generating bitmap(s)

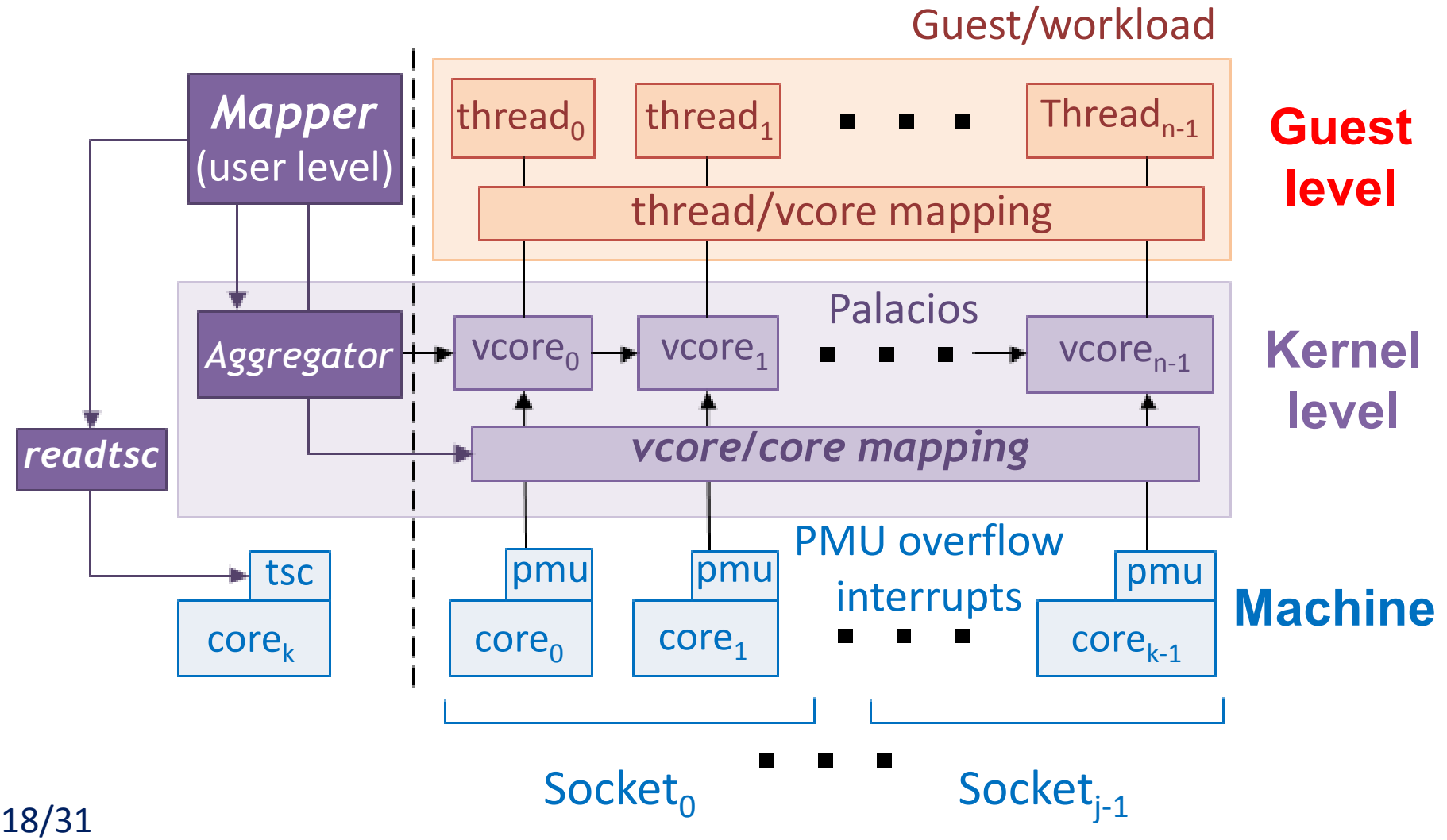


(3) Aggregator collects bitmaps across vcores, and ***computes the metrics***

Outline

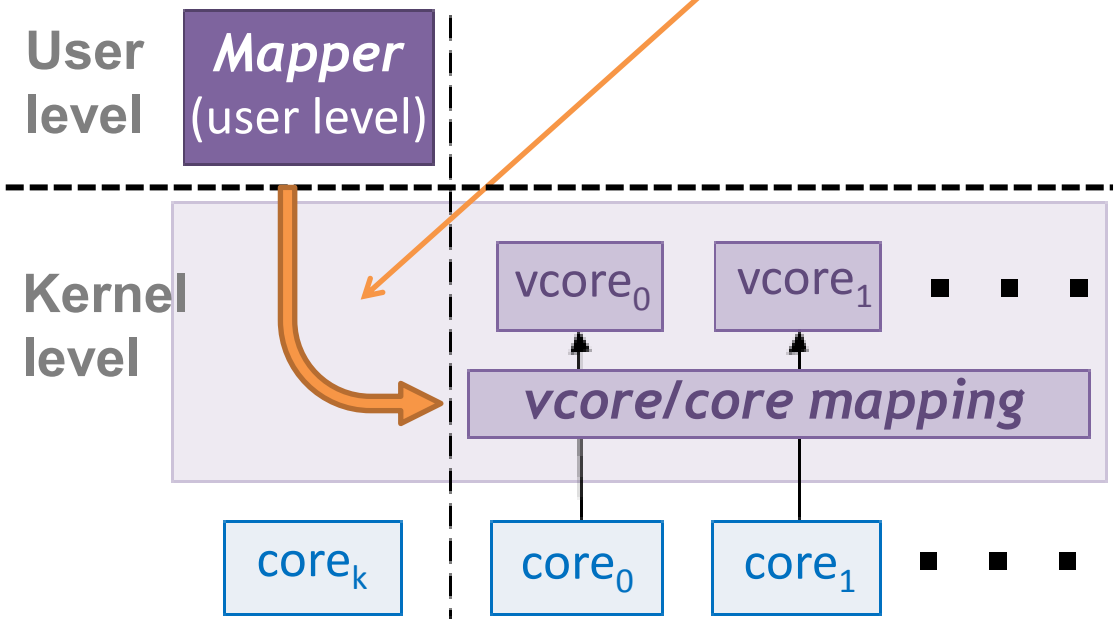
- Opportunities in vcore mapping
- Metrics and measurement
- **System**
  - Overview
  - vcore migration mechanism
  - vcore mapping policy
- Results
- Conclusion

System overview



Virtual core migration mechanism

vcore mapping in Palacios is changed only on ***explicit request(s)*** from ***Mapper***



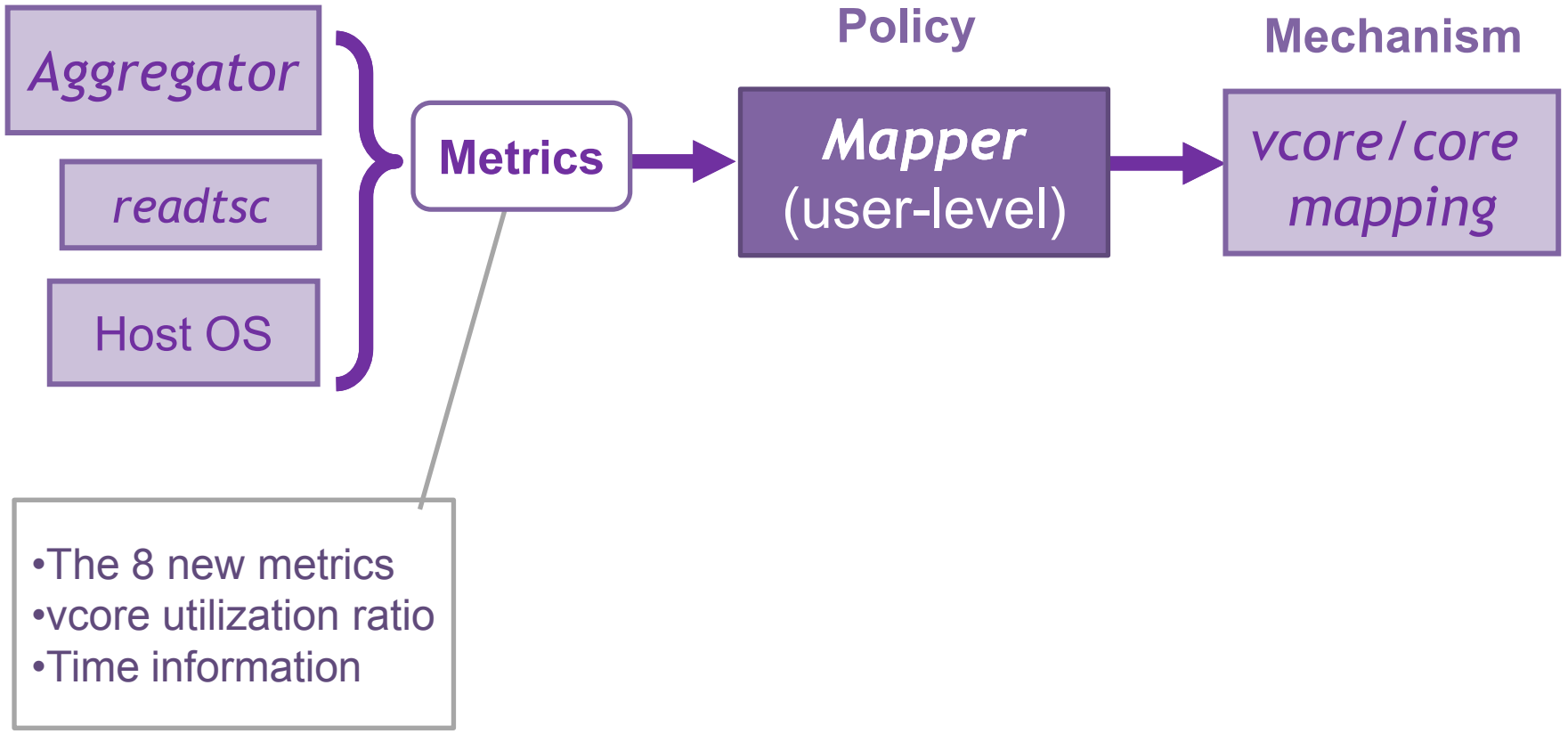
Steps in the request:

1) Forces all vcores to exit

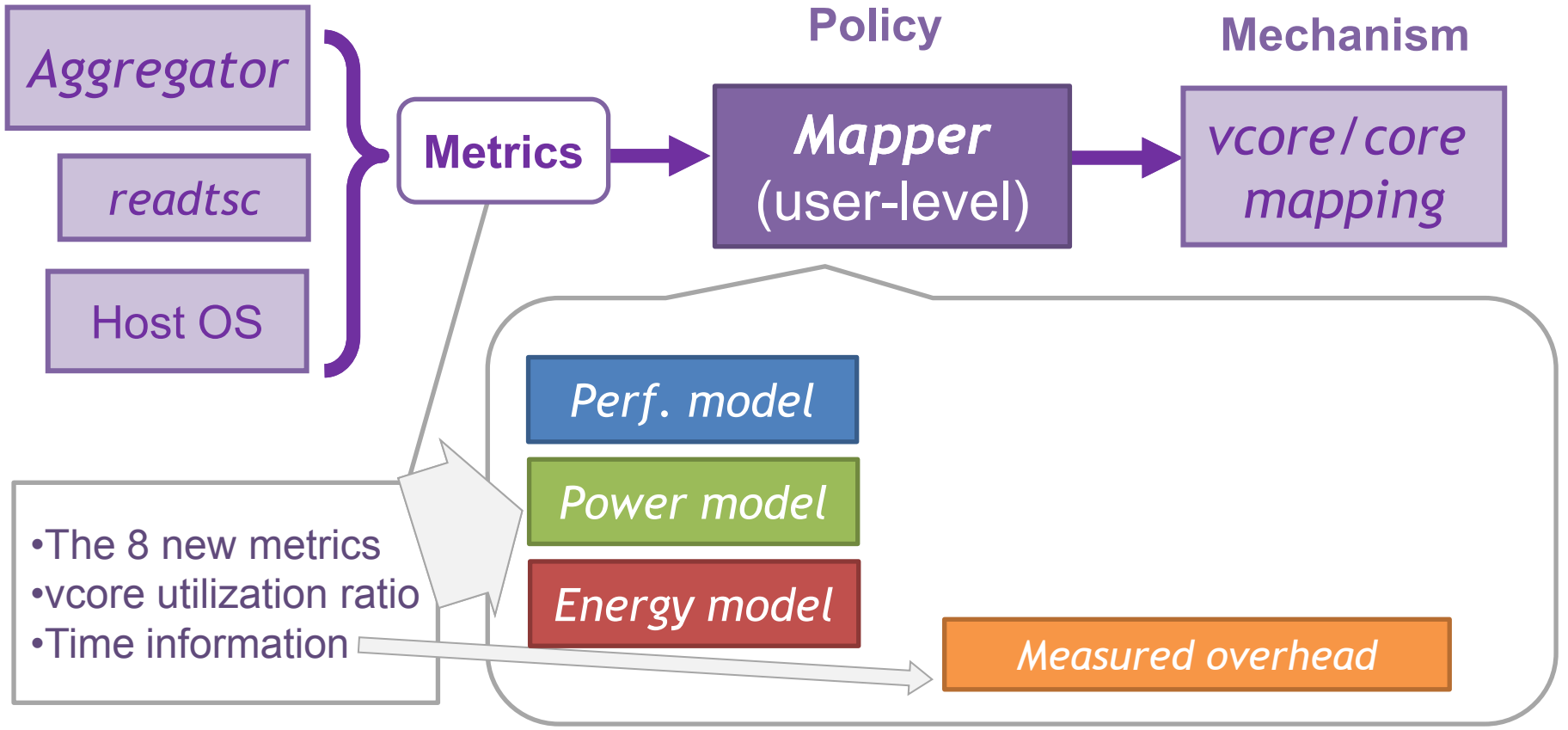
2) Rebinds host kernel threads, with virtualization states to the new locations

3) Synchronizes threads and reenters the guest

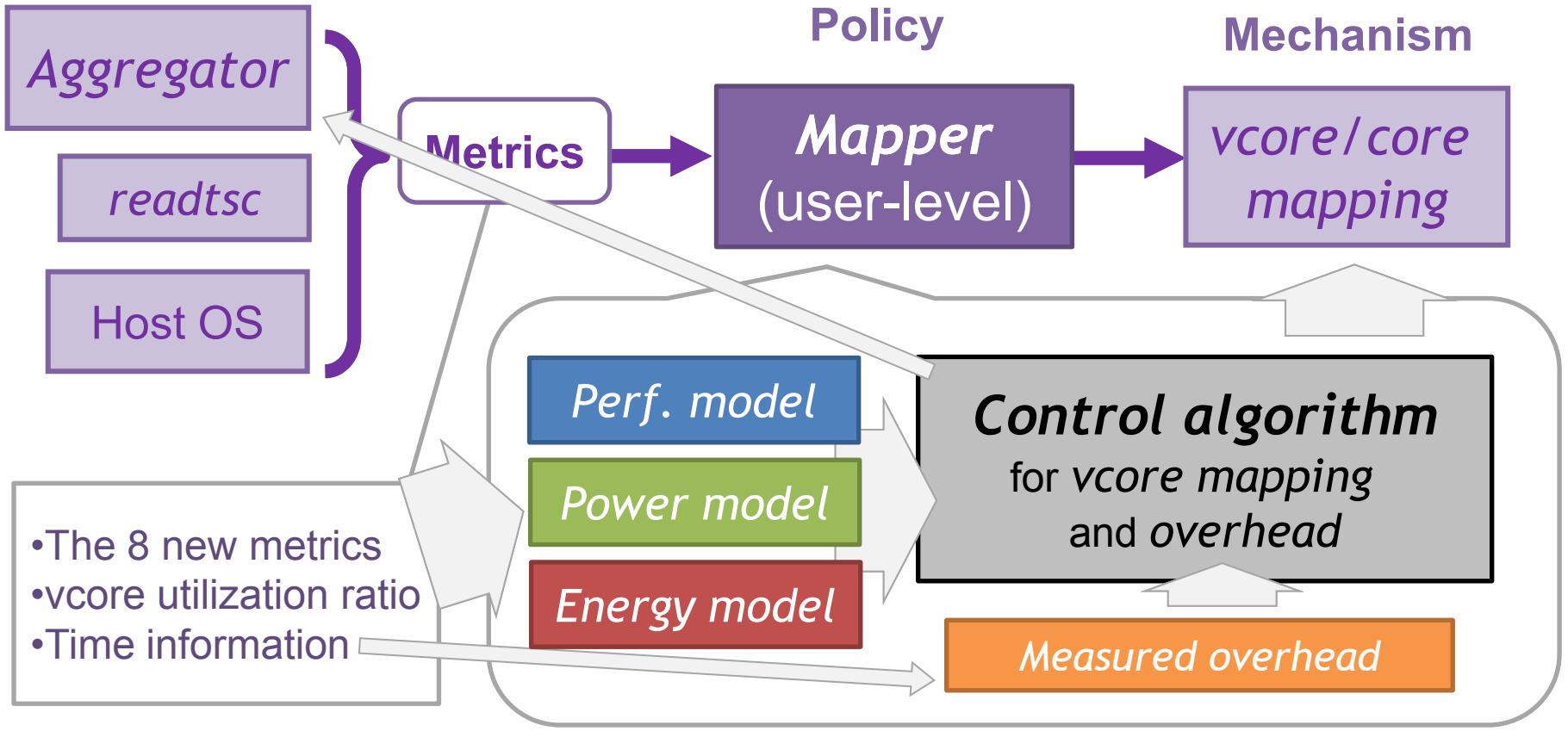
Mapper finds vcore mapping with controlling overheads



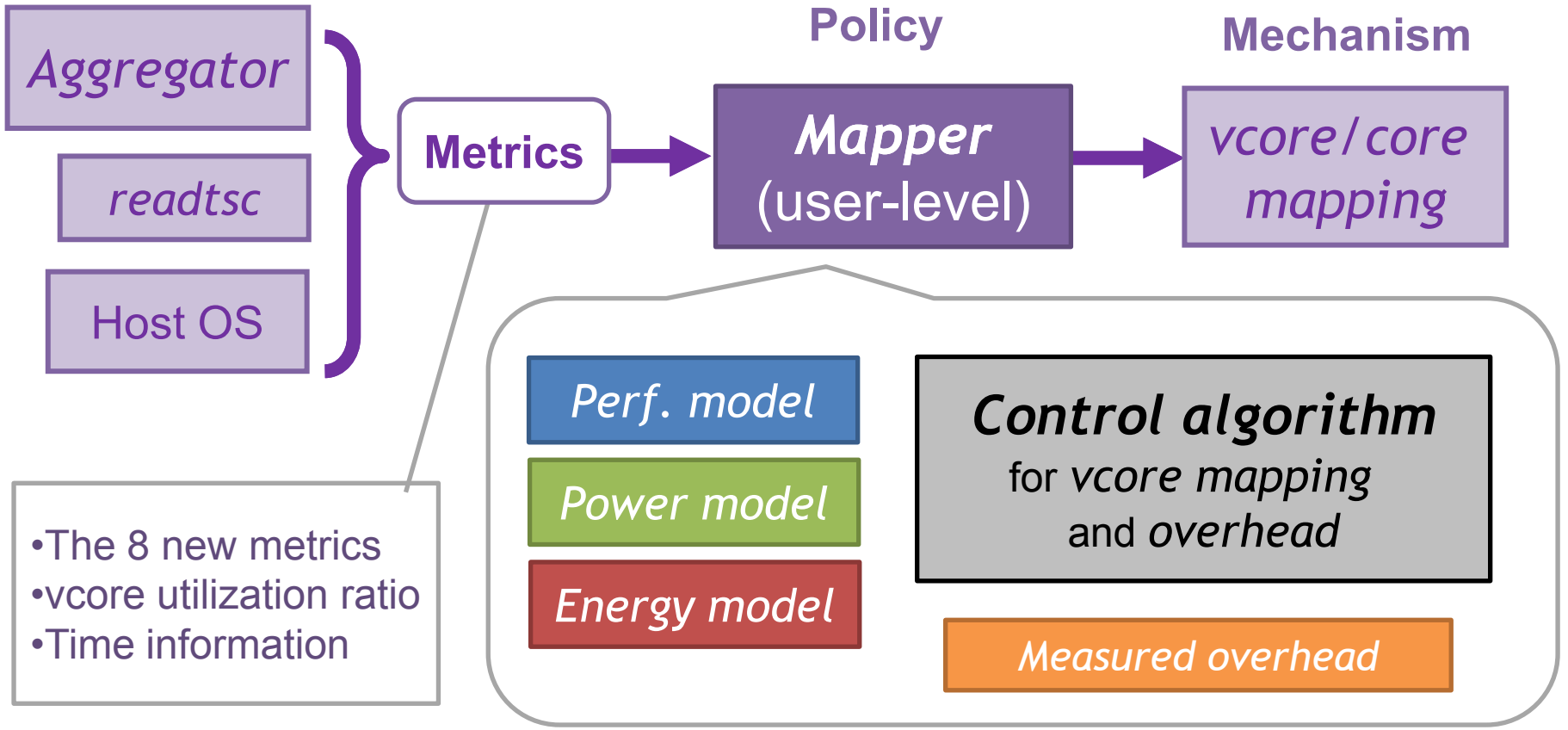
Mapper finds vcore mapping with controlling overheads



Mapper finds vcore mapping with controlling overheads



Mapper finds vcore mapping with controlling overheads



Details on *the **models*** incorporated in *adaptive **control algorithm*** can be found in the paper

Outline

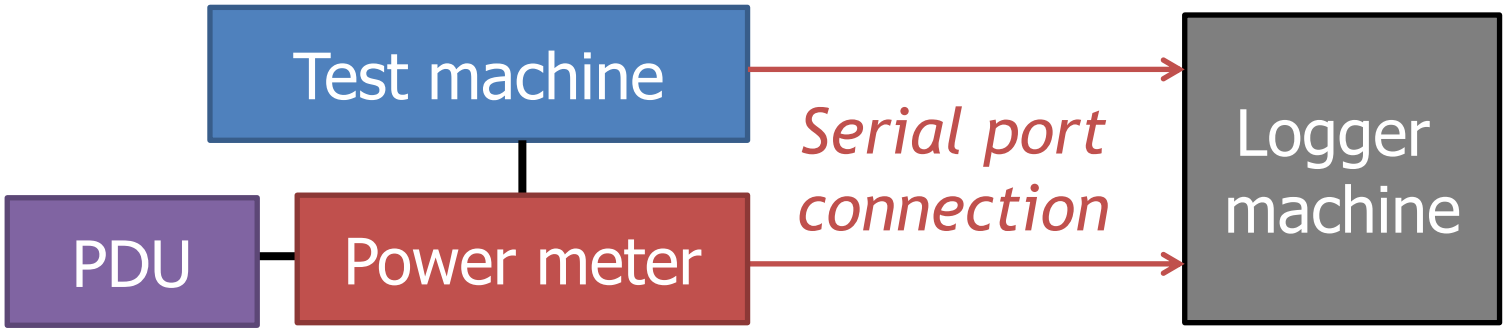
- Opportunities in vcore mapping
- Metrics and measurement
- System
- **Results**
  - Setup
  - Experimental results
- Conclusion

Experimental setup

- Workload – SPEC OMP 2001, PARSEC 2.1
- Software
  - Guest OS – Linux ver2.6.30
  - VMM – Palacios ver1.3
  - Host OS – Linux ver2.6.38
- Hardware
  - 2 Processor sockets (NUMA)
  - CPU – Intel® Xeon™ E5620,

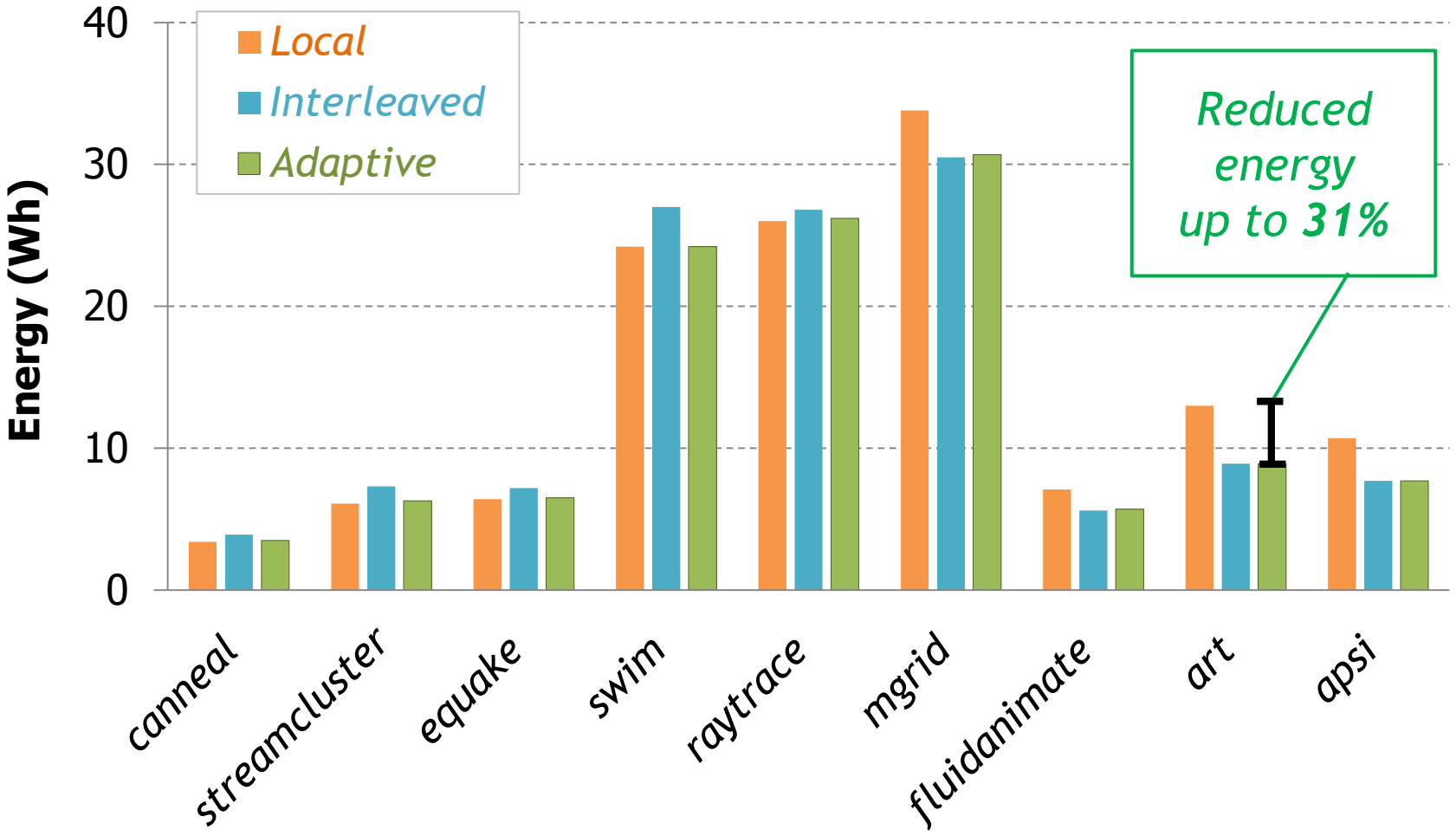
    with 4 cores (8 HW threads) x 2
  - Memory – 4GB with 1066 MHz (DDR3) x 2

Measurement for the results

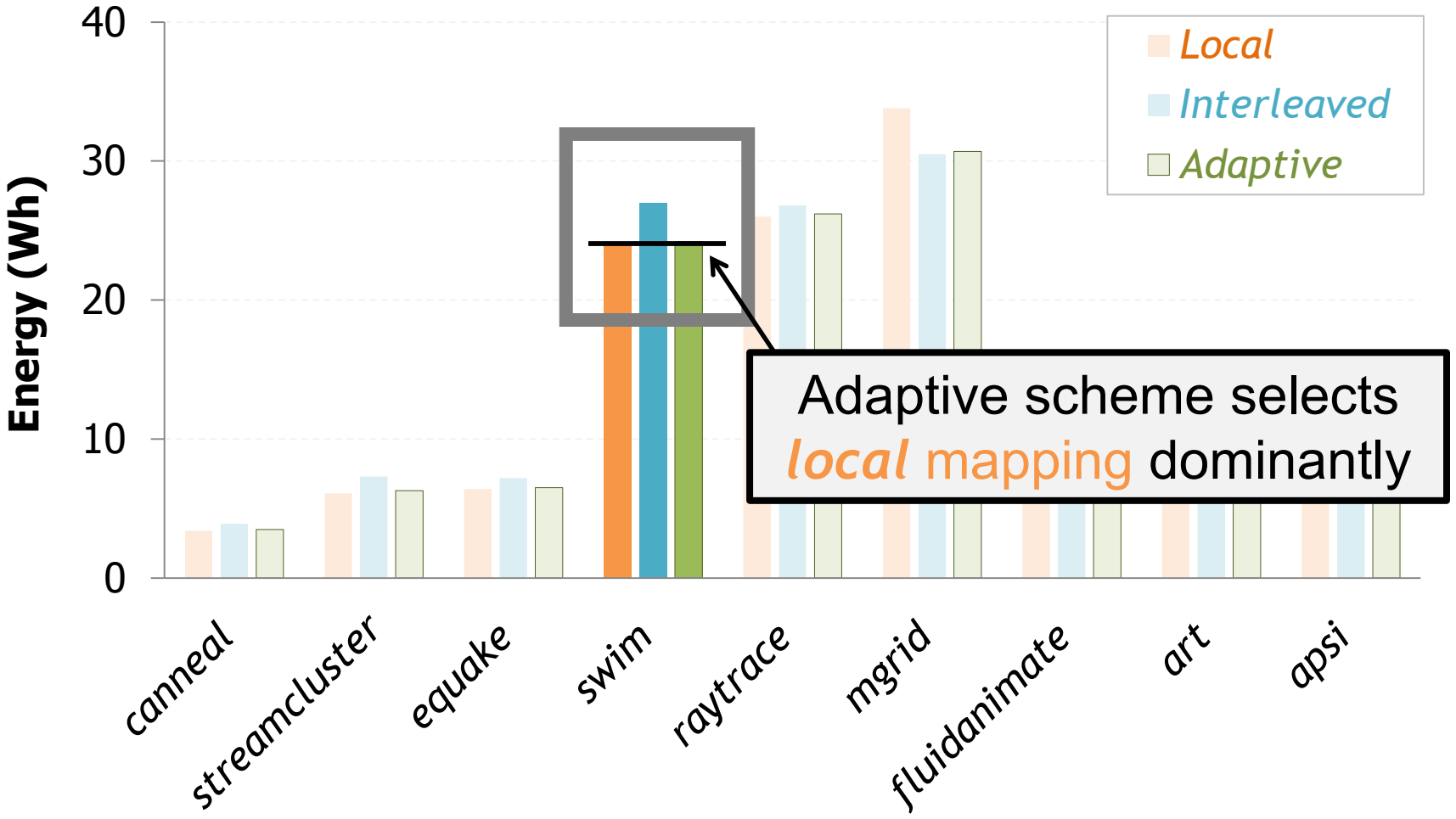


- Execution time – time stamps at the start/end of execution
- Energy – power meter outputs energy information
- Average power – from energy by execution time.

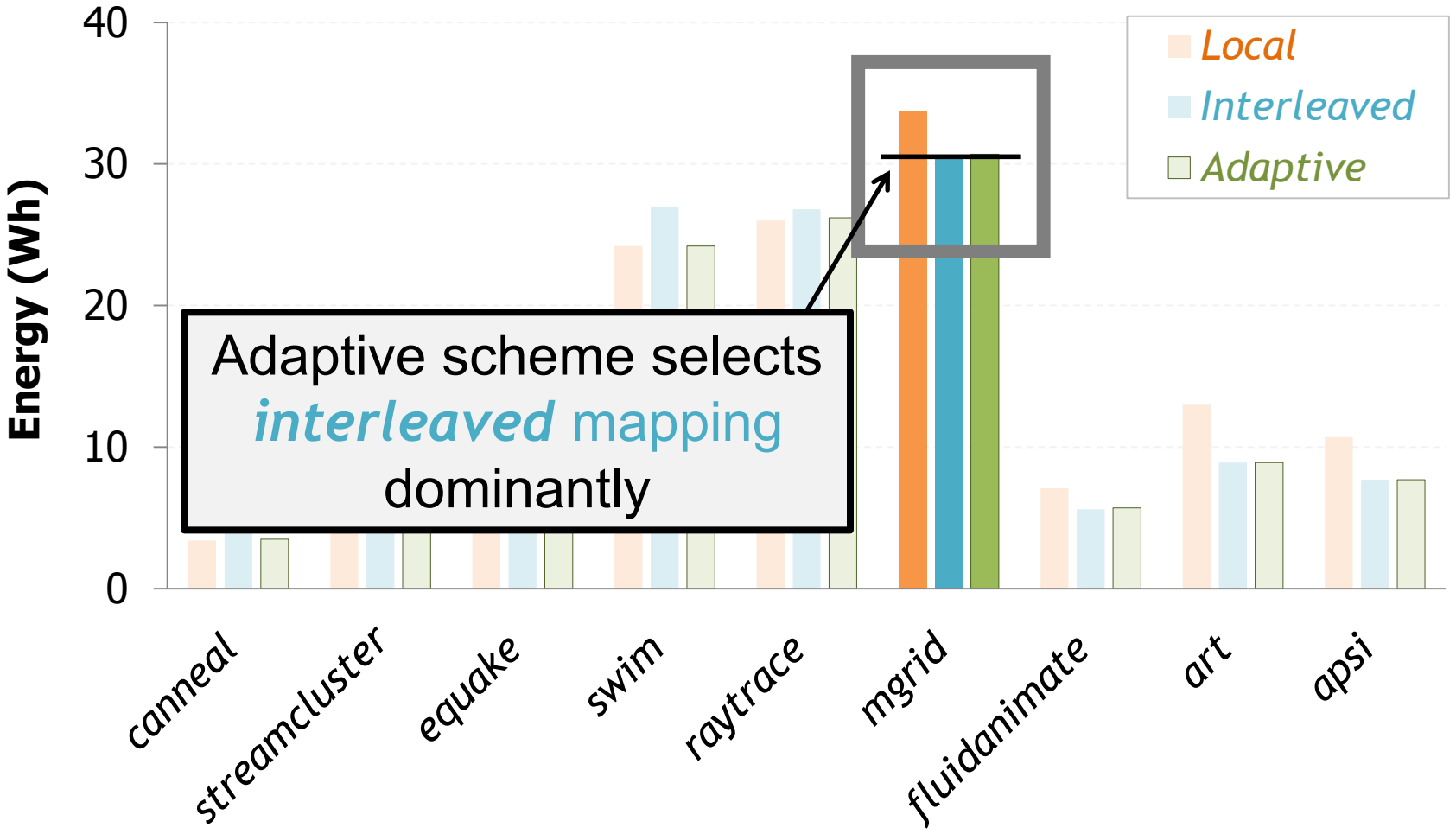
Adaptive always chooses best mapping for *energy*



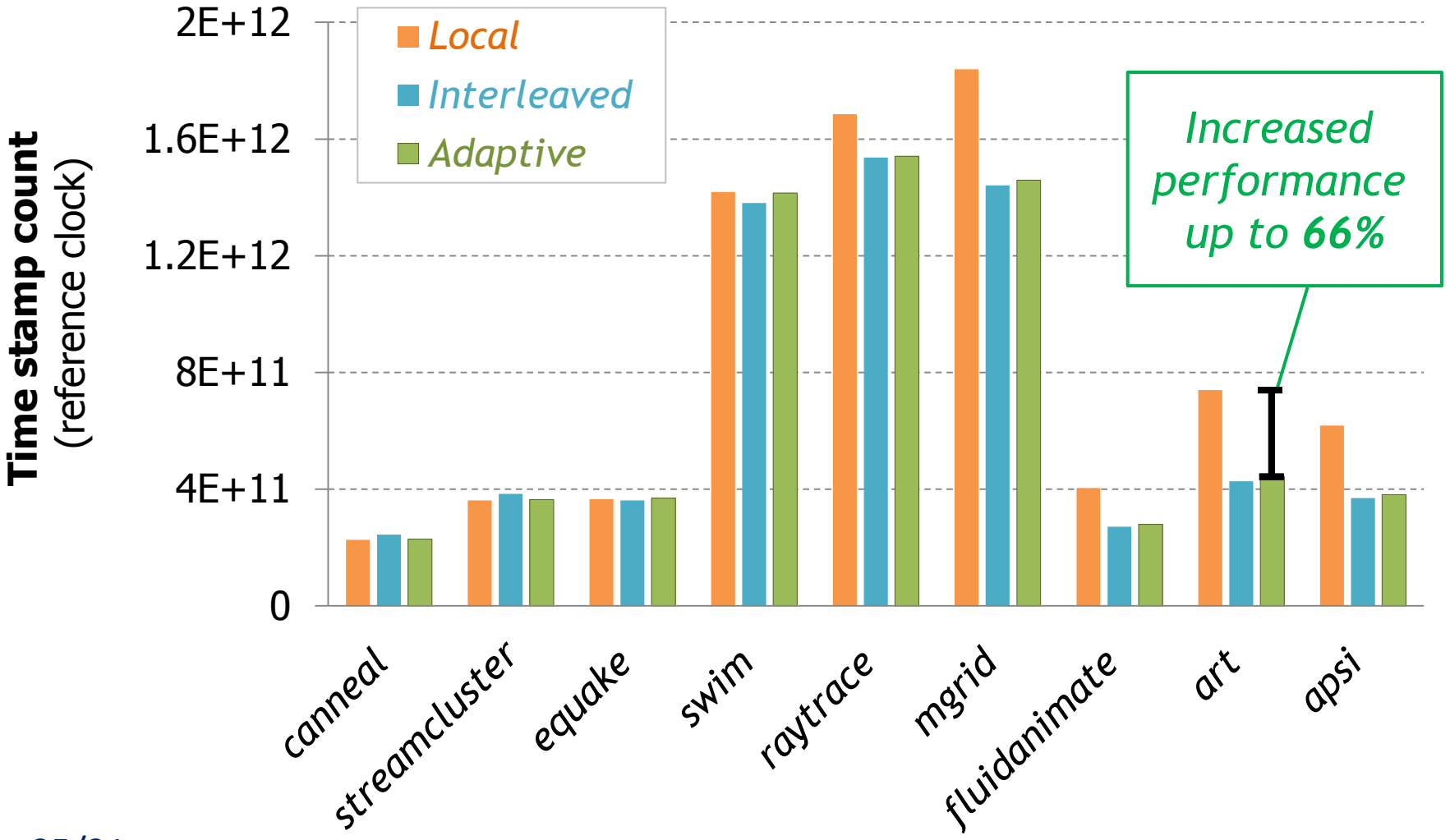
Adaptive always chooses best mapping for *energy*



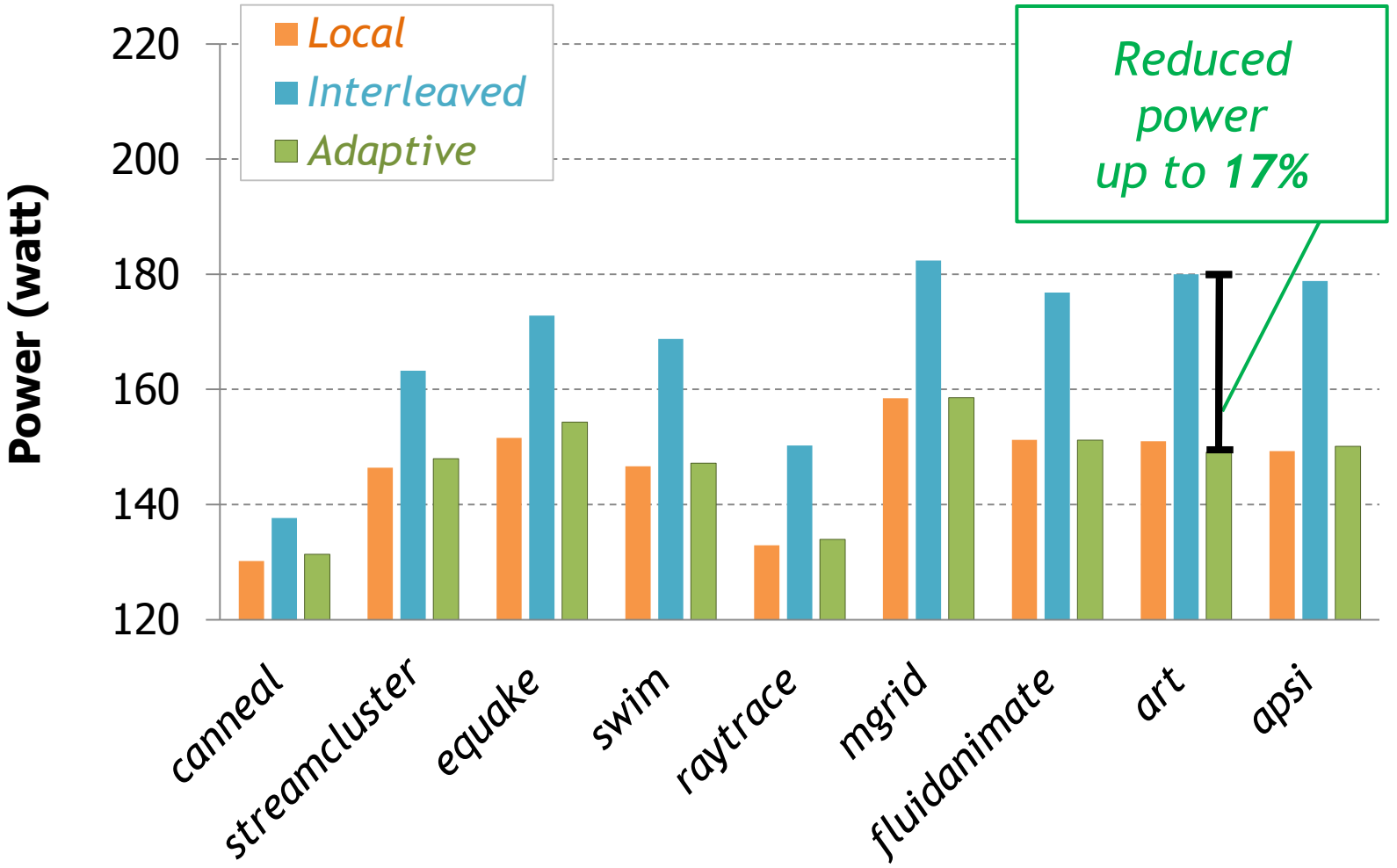
Adaptive always chooses best mapping for *energy*



Adapt. always chooses the best mapping for *performance*



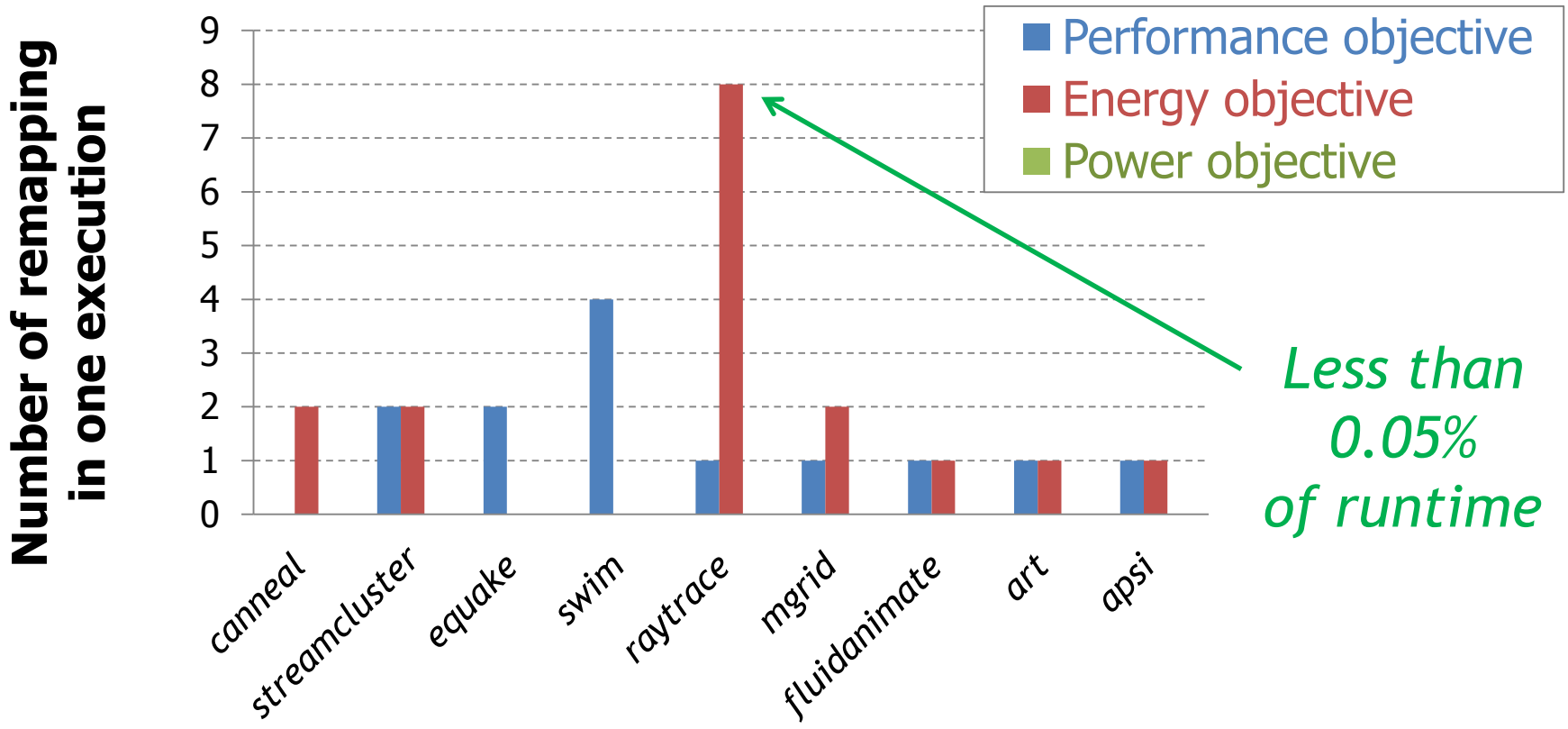
Adaptive always chooses best mapping for *power*



Overheads in measurement and remapping are small

Benchmark	PT scanning overhead (ms)	vcores mapping cost (ms)
canneal	1.51	5.24
streamcluster	0.78	5.27
equake	0.82	5.25
swim	2.34	**5.08**
raytrace	**0.39**	5.24
mgrid	0.61	5.27
fluidanimate	0.58	5.25
art	1.30	**5.30**
apsi	**4.61**	5.27

Remapping cost is controlled



raytrace, *swim*, and *mgrid* run for >10 mins
Others run for 3—5 mins

Conclusion

- *Opportunity* for optimizing the selected objective by selecting one of two vcore mappings
- *Detection framework* for capturing shared memory reference behavior with a set of new metrics
- *Dynamic adaptive system* for selecting the best mapping

Future work

- Developing formulations for generic vcore mapping, scheduling, and page mapping
- Extending HW assisted SW monitor to capture other sets of new metrics
- Working on design, implementation, and evaluation of adaptive system incorporating NUMA optimization in a VMM

Questions?

- Questions and Answers

- Contact information

  chang.bae@eecs.northwestern.edu

  http://www.changbae.org

- Project website

  http://v3vee.org