ACM Symposium on Cloud Computing

13th Symposium on Cloud Computing 2022 (SoCC'22)

Demeter: QoS-Aware CPU Scheduling to Reduce Power Consumption of Multiple Black-Box Workloads

Wenda Tang[1,2], Yutao Ke[2], Senbo Fu[2], Hongliang Jiang[3], Junjie Wu[3],
Qian Peng[2], Feng Gao[1]

[1]School of Automation Science and Engineering, Xi'an Jiaotong University
[2]Huawei Cloud Computing Technologies Co., Ltd.
[3]Huawei Technologies Co., Ltd.

Presentation Outline

1. Background
2. Workload Characterization & Analysis
3. *Demeter Design*
4. Experimental Results
5. Conclusions

Challenges for reducing power consumption of CPU in public clouds

- Challenge 1: Black-box/Opaque workloads [1]

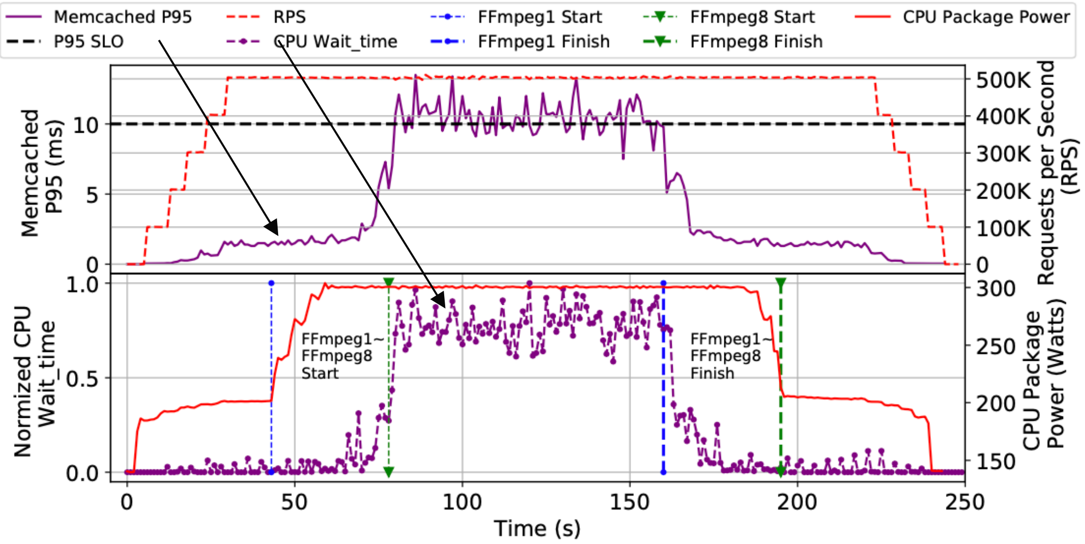
LC: Latency-critical workloads, e.g., MySQL, Redis, …
Sensitive to frequency scaling
BE: Best-effort workloads, e.g., Hadoop, ML training,…
Tolerate frequency fluctuation

How to identify the LC from BE?

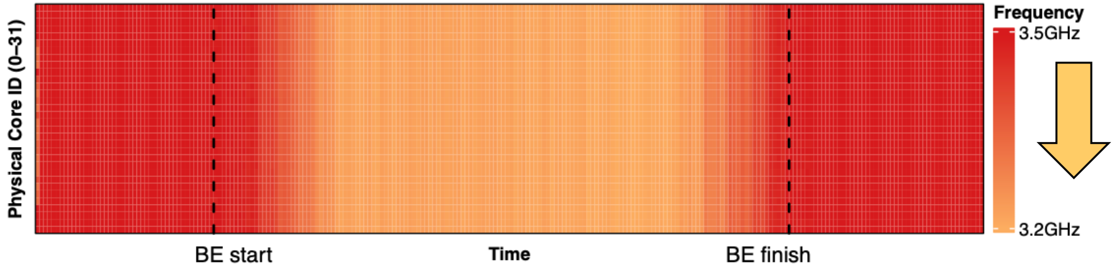
- Challenge 2: All Cores Power-throttling

CPU contention between LC and BE.
Severe performance degradation induced by high power.

How to guarantee high performance while reducing power consumption?



(a) Memcached's SLO violation



(b) All cores power-throttling

[1] Kostis Kaffes, Dragos Sbirlea, Yiyan Lin, David Lo, and Christos Kozyrakis. 2020. Leveraging application classes to save power in highly-utilized data centers. In Proceedings of the 11th ACM Symposium on Cloud Computing (SoCC '20). ACM, New York, NY, USA, 134–149.

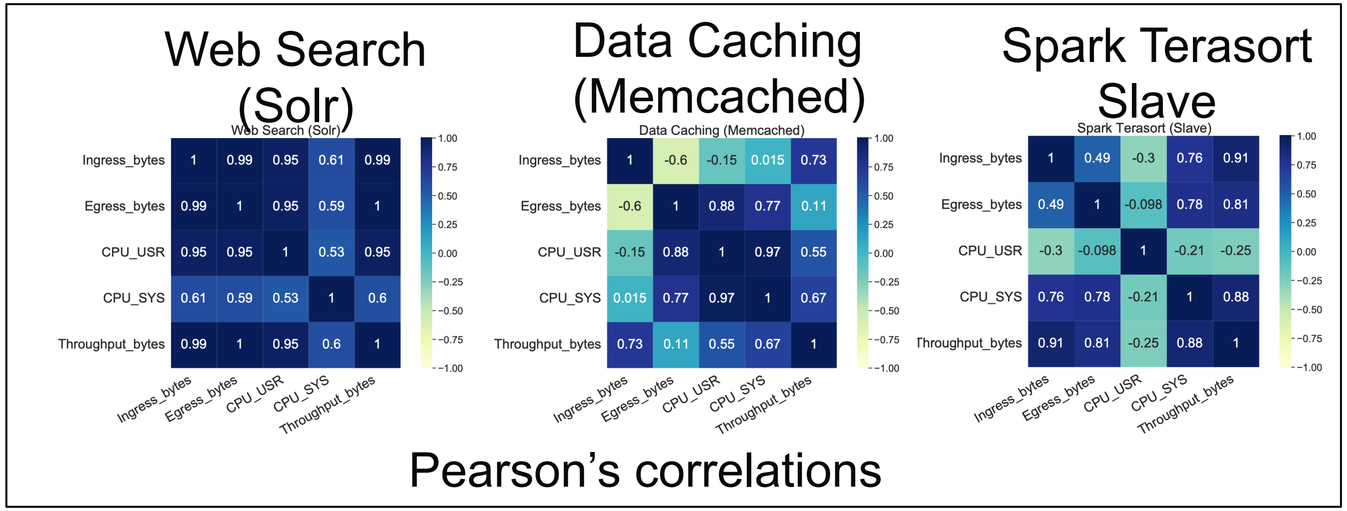
Observations

1. LC – **Stable** network ingress/egress traffic ratio
2. LC – **Stable** network throughput

VS. BE – **Unstable** network throughput (if any)

3. LC – **No Network, No CPU usage**

VS. BE – **No Network, High CPU usage**



Web Search (Solr)

	Ingress_bytes	Egress_bytes	CPU_USR	CPU_SYS	Throughput_bytes
Ingress_bytes	1	0.99	0.95	0.61	0.99
Egress_bytes	0.99	1	0.95	0.59	1
CPU_USR	0.95	0.95	1	0.53	0.95
CPU_SYS	0.61	0.59	0.53	1	0.6
Throughput_bytes	0.99	1	0.95	0.6	1

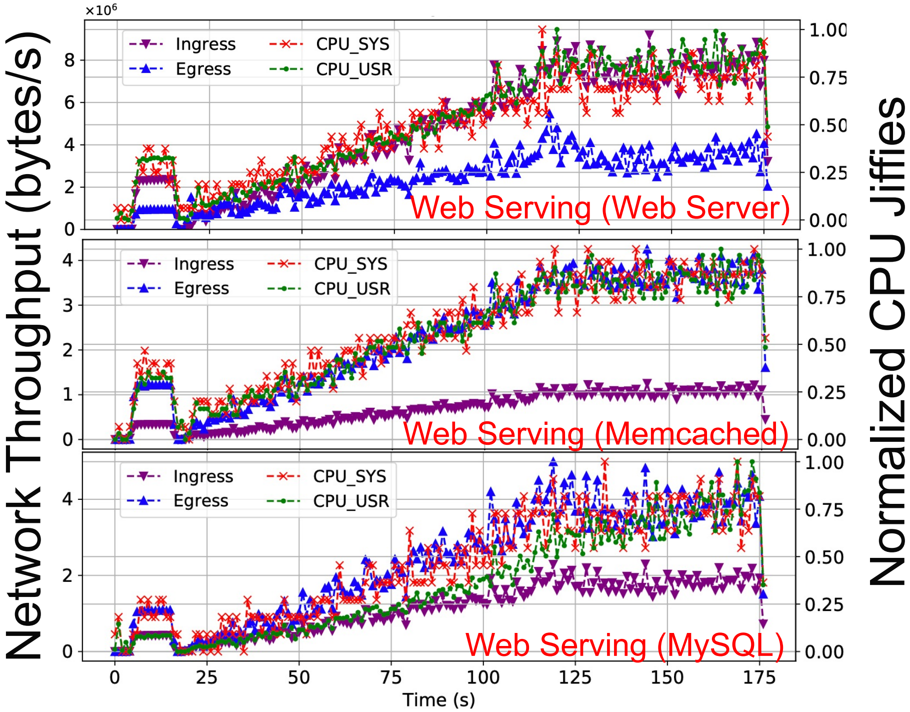
Data Caching (Memcached)

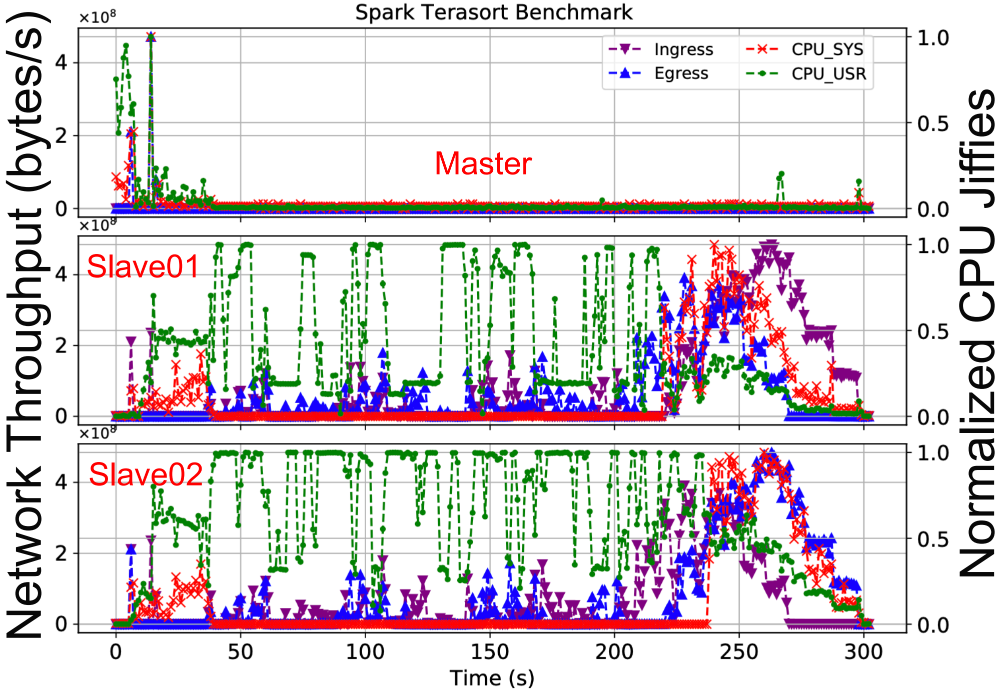
	Ingress_bytes	Egress_bytes	CPU_USR	CPU_SYS	Throughput_bytes
Ingress_bytes	1	-0.6	-0.15	0.015	0.73
Egress_bytes	-0.6	1	0.88	0.77	0.11
CPU_USR	-0.15	0.88	1	0.97	0.55
CPU_SYS	0.015	0.77	0.97	1	0.67
Throughput_bytes	0.73	0.11	0.55	0.67	1

Spark Terasort (Slave)

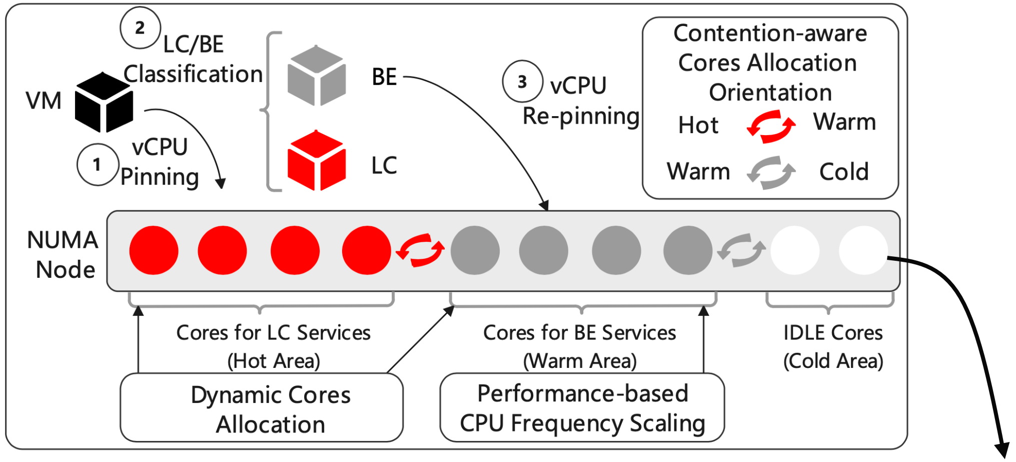
	Ingress_bytes	Egress_bytes	CPU_USR	CPU_SYS	Throughput_bytes
Ingress_bytes	1	0.49	-0.3	0.76	0.91
Egress_bytes	0.49	1	-0.098	0.78	0.81
CPU_USR	-0.3	-0.098	1	-0.21	-0.25
CPU_SYS	0.76	0.78	-0.21	1	0.88
Throughput_bytes	0.91	0.81	-0.25	0.88	1

Pearson's correlations





Overview of *Demeter*

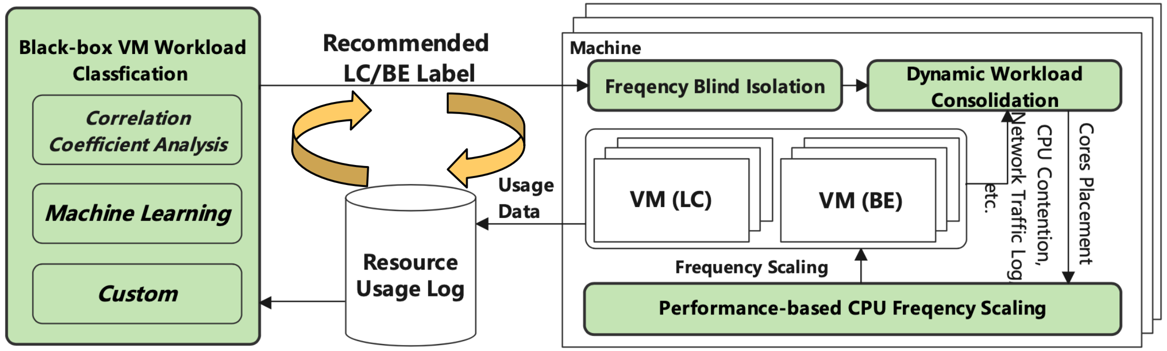

2 LC/BE Classification
VM
1 vCPU Pinning
BE
LC
3 vCPU Re-pinning
Contention-aware Cores Allocation Orientation
Hot
Warm
Warm
Cold
NUMA Node
Cores for LC Services (Hot Area)
Cores for BE Services (Warm Area)
IDLE Cores (Cold Area)
Dynamic Cores Allocation
Performance-based CPU Frequency Scaling

Stable Maximum Frequency

Dynamic Frequency Scaling

Minimum Frequency

Dataflow of *Demeter*


Black-box VM Workload Classfication
Correlation Coefficient Analysis
Machine Learning
Custom
Recommended LC/BE Label
Usage Data
Resource Usage Log
Machine
Freqency Blind Isolation
Dynamic Workload Consolidation
VM (LC)
VM (BE)
CPU Contention, Network Traffic Log etc.
Cores Placement
Frequency Scaling
Performance-based CPU Freqency Scaling

Periodical Classification & QoS-aware CPU scheduling to reduce power consumption of black-box workloads

Features	Summary
1. Black-box Workloads Classification	No prior knowledge and no offline profiling
2. Frequency Blind Isolation	Performance guarantee (Hard resource isolation)
3. IDLE CPU Cycles Harvesting	Soft resource isolation to improve resource utilization
4. Dynamic Workload Consolidation	QoS-aware resource allocation
5. Performance-based CPU Frequency Scaling	Novel CPU frequency scaling governor

① Black-box workloads classification

Main points:

1) All workloads are default LC

2) Periodical classification for all workloads

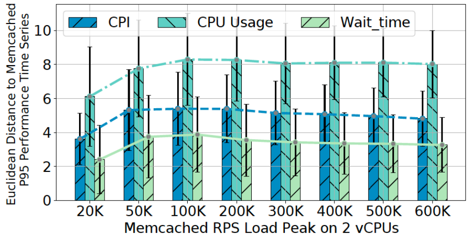
Check if there is BE behavior

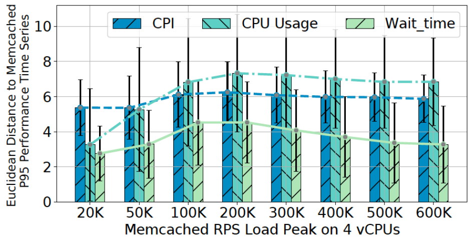
a. little network traffic with high CPU usage

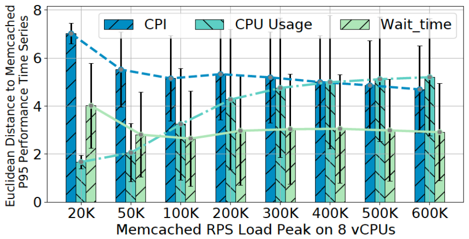
b. correlation analysis

(e.g., r(Throughput_bytes & CPU_USR) < 0.3)

④ Dynamic Workload Consolidation





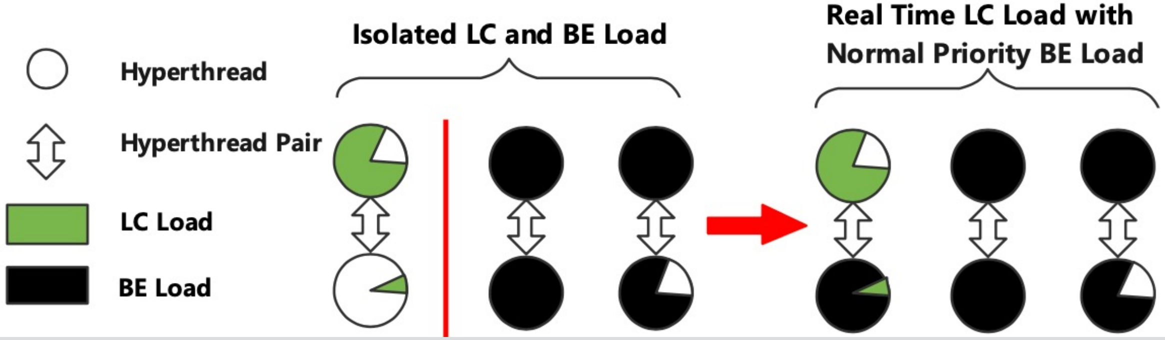


CPU Wait_time Metric as a scale out/in signal

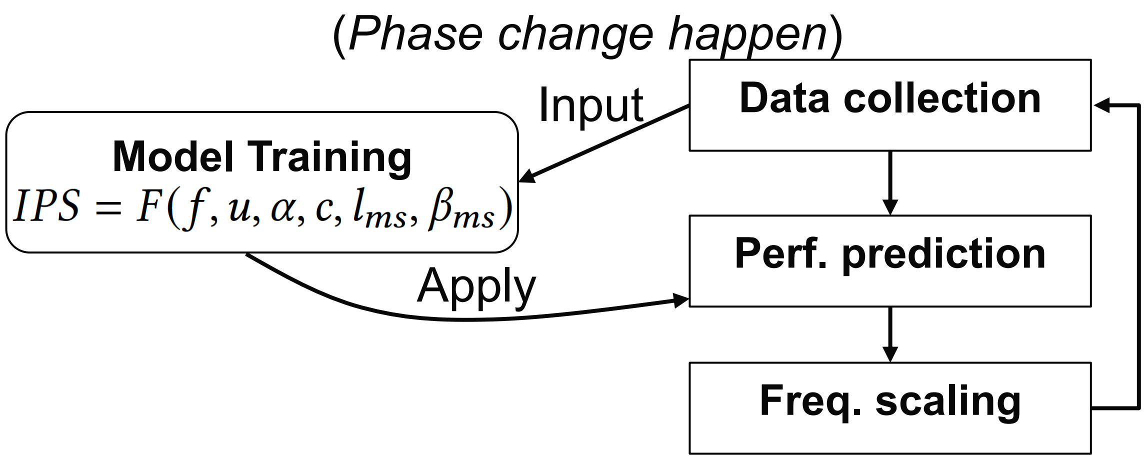
② Frequency Blind Isolation

- Hot area → Stable High Freq. & C0 state
- Warm area → Dynamic Freq.
- Dynamic → Stable Low Freq & Deepest C-state

③ IDLE CPU Cycles Harvesting



⑤ Performance-based CPU Frequency Scaling



(Phase change happen)

Input

Data collection

Model Training

$$IPS = F(f, u, \alpha, c, l_{ms}, \beta_{ms})$$

Apply

Perf. prediction

Freq. scaling

Evaluation Settings and Metrics

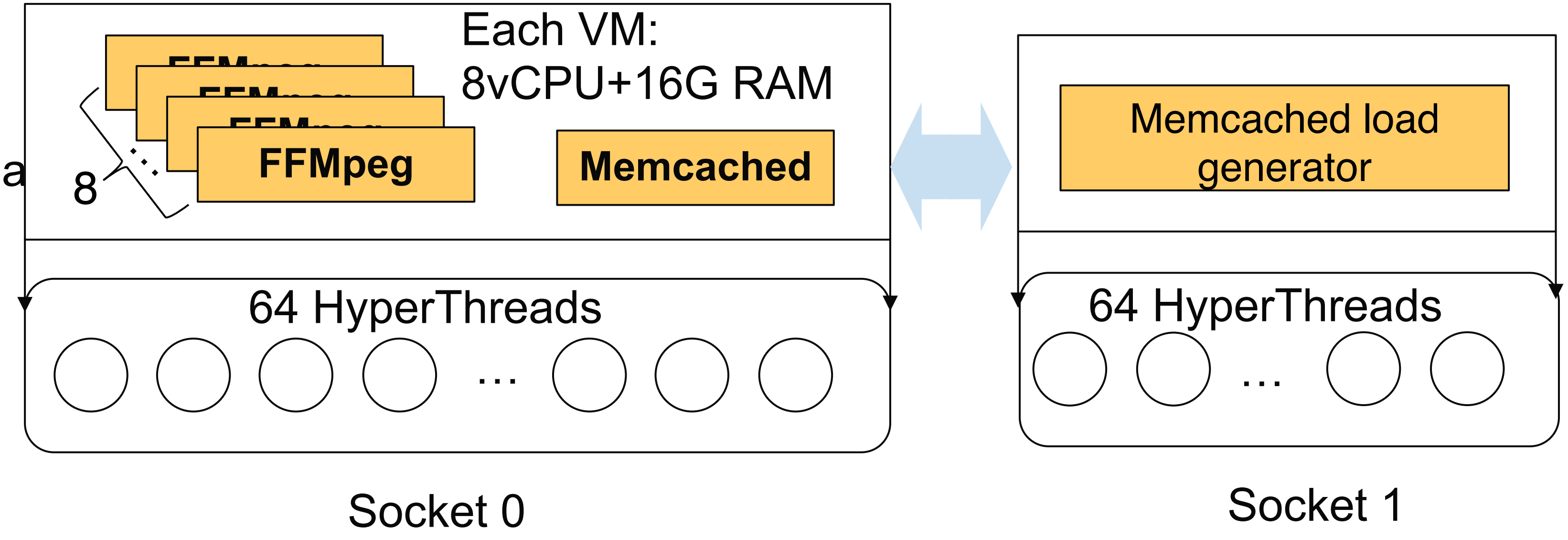
Model	Intel(R) Xeon(R) Platinum 8378A
CPU frequency	3.0GHz Base
OS	CentOS 7.9 with kernel 3.10
Sockets	2
Cores per socket	32
Threads per core	2
NUMA nodes	2
TDP (Watts)	300

1. QoS:
- LC : Memcached P95 < 10ms
- BE : Job finish time, the shorter the better

2. Energy efficiency:
- Energy consumed by all workloads from start to finish, the less the better

1) Turn off c-states for cores in hot area

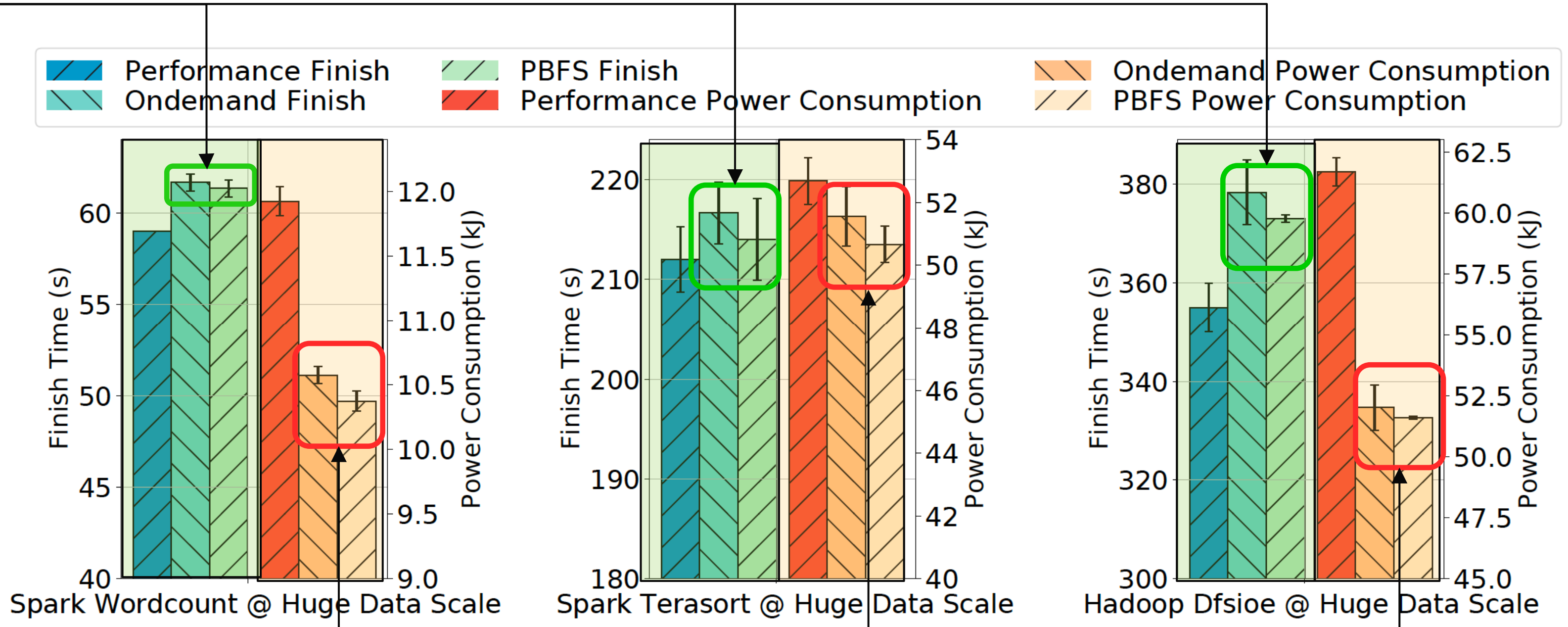
2) Measure power consumption of Socket 0



Each VM:
8vCPU+16G RAM

3) Keep Max Freq. and Turn off c-states for cores in socket 1

PBFS Effectiveness in *Demeter*



Reduce up to 1.4% finish time

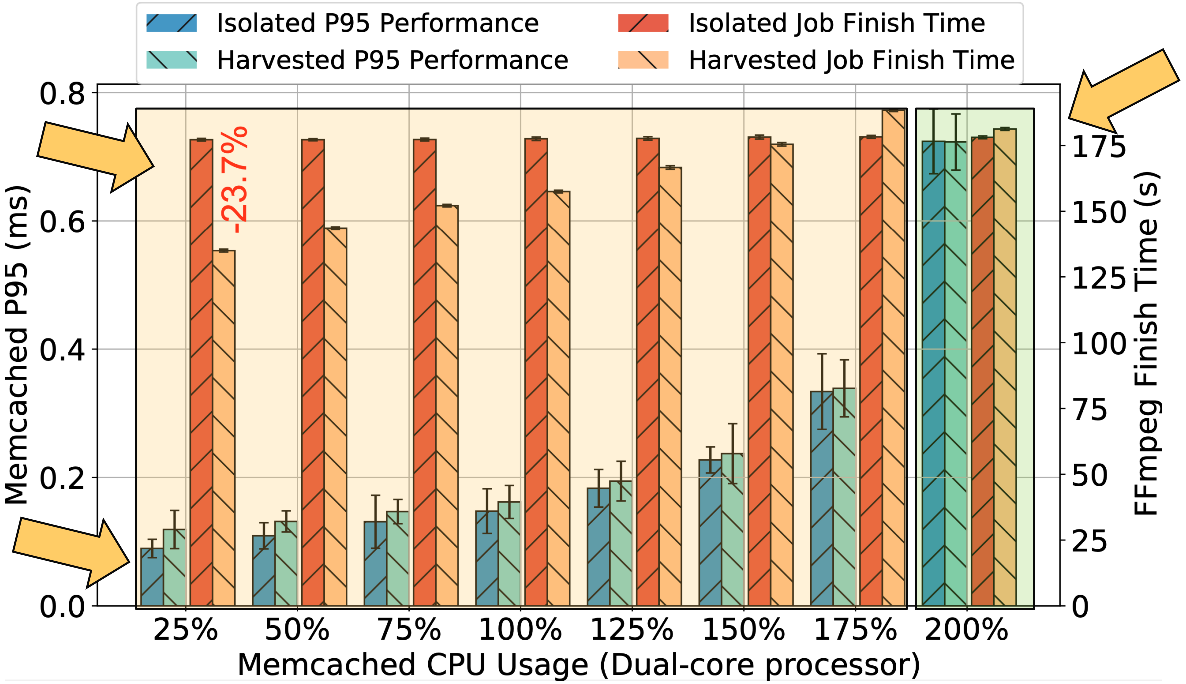
Reduce up to 1.9% energy consumption

Comparison among *PBFS*, *Performance* and *Ondemand* governor.

IDLE Cycles Harvesting Effectiveness in *Demeter*

Obvious BE improvement

Almost same LC performance



Should stop harvesting

Performance comparison between w/ and w/o Harvesting.

Comparison with Other Controllers

Settings:

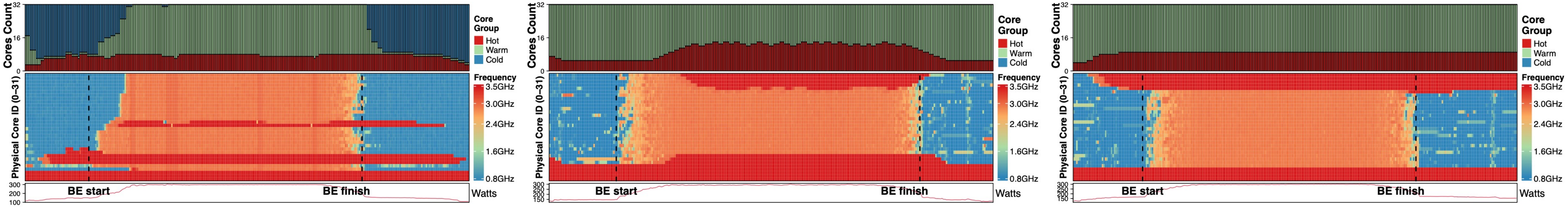
1. Controllers:

- BASE : No QoS guarantee and all cores run at the maximum frequency
- PerfISO [ATC’18] : LC with IDLE cores based autoscaling
- PACT [SoCC’20] : LC with CPU usage based autoscaling
- Demeter: Solution with full features
- $\text{Demeter}_{\text{-IDLE-BE's wait\_time}}$: Turn off IDLE Harvesting feature and BE’s cores autoscaling feature
- $\text{Demeter}_{\text{-IDLE}}$: Turn off IDLE Harvesting feature

2. Workloads

- Scenario I (LC high + BE high): Memcached’s RPS 100K→500K(200s)→100K + 8 FFMpeg workloads
  Scenario II (LC high + BE low) : Memcached’s RPS 100K→500K(200s)→100K + 1 FFMpeg workloads
- Scenario III (LC low + BE high): Memcached’s RPS 5K→50K(200s)→5K + 8 FFMpeg workloads
- Scenario IV (LC low + BE low) : Memcached’s RPS 5K→50K(200s)→5K + 1 FFMpeg workloads

Scenario I Experimental Results



(a) *Demeter* (b) *PerfIso* (c) *PACT*

Demeter use as fewer CPU cores as possible

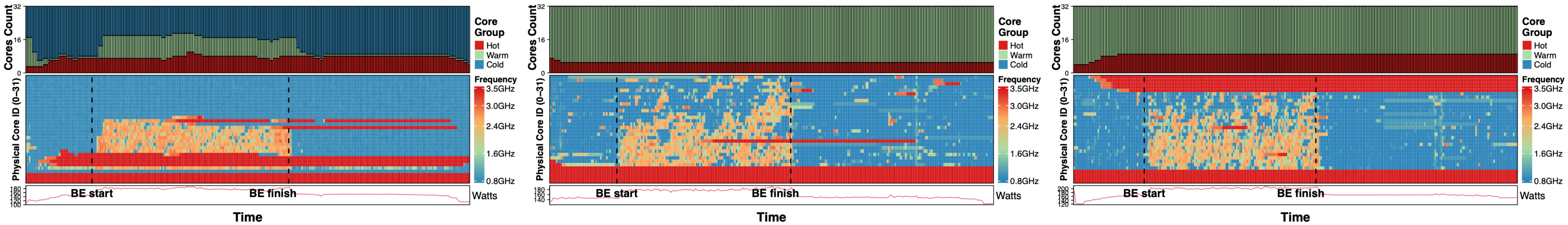
Scenario	Mechanism	Average power consumption ± SD(kJ)	BE finish time ± SD(s)	LC P95 max (ms)
I (LC high + BE high)	Base	81.99±0.17	127.57±1.49	22.7
	PerfIso	72.17±0.14	167.91±1.68	6.5
	PACT	72.21±0.13	150.53±1.29	6.9
	Demeter$_{\text{-IDLE-BE's wait\_time}}$	69.34±0.19	143.28±1.31	2.5
	Demeter$_{\text{-IDLE}}$	67.91±0.26	145.25±2.41	2.6
	Demeter	67.59±0.17	146.57±3.36	2.0

in Top 3 in Top 3 in Top 3

performance interference

Demeter has the best energy efficiency and provides enough QoS guarantee for workloads.

Scenario II Experimental Results



(a) *Demeter* (b) *PerfIso* (c) *PACT*

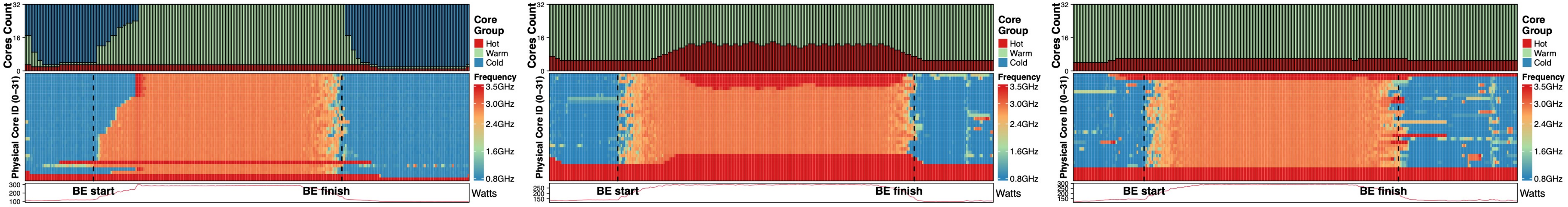
Demeter use as fewer CPU cores as possible

Scenario	Mechanism	Average power consumption ± SD(kJ)	BE finish time ± SD(s)	LC P95 max (ms)
II (LC high + BE low)	Base	75.83±0.08	137.75±17.37	1.4
	PerfIso	48.94±0.12	140.62±9.86	6.8
	PACT	54.82±0.16	140.59±9.95	1.7
	Demeter$_{\text{-IDLE-BE's wait\_time}}$	51.42±0.22	140.70±9.67	1.7
	Demeter$_{\text{-IDLE}}$	48.91±0.38	141.34±8.05	1.4
	Demeter	48.13±0.34	141.55±7.57	2.1

in Top 3 in Top 3

1）*Demeter* VS. *PACT*
Both have similar QoS guarantee perf., but *Demeter* has better energy efficiency.
2) *Demeter* VS. *PerfIso*
PerfIso has shorter finish time of BE but three times higher P95.

Scenario III Experimental Results



(a) *Demeter* (b) *PerfIso* (c) *PACT*

Demeter use as fewer CPU cores as possible

Scenario	Mechanism	Average power consumption ± SD(kJ)	BE finish time ± SD(s)	LC P95 max (ms)
III (LC low + BE high)	Base	78.34±0.09	120.98±1.41	7.0
	PerfIso	67.76±0.20	165.27±1.54	0.1
	PACT	64.92±0.12	138.21±1.21	0.12
	Demeter$_{\text{-IDLE-BE's wait\_time}}$	61.30±0.45	133.02±0.99	0.28
	Demeter$_{\text{-IDLE}}$	59.18±0.31	134.73±1.95	0.28
	Demeter	58.60±0.16	134.34±1.65	0.28

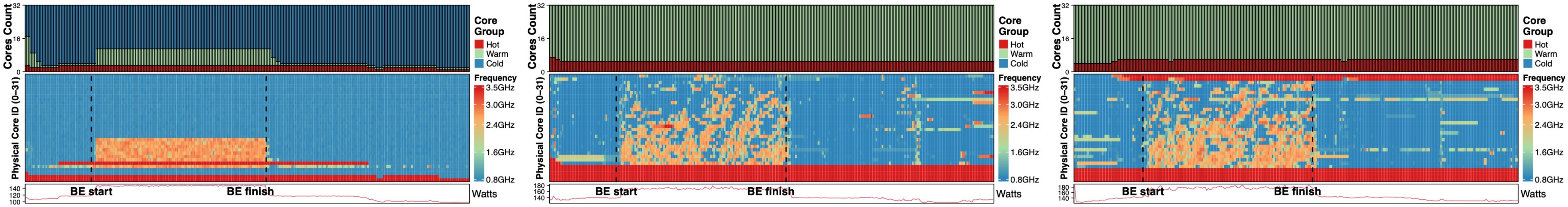
in Top 3 in Top 3 in Top 3

performance interference

1） *Demeter* VS. *PACT*
Both have similar QoS guarantee perf., but *Demeter* has better energy efficiency.

2) *Demeter* VS. *PerfIso*
PerfIso has excellent P95 perf but much longer finish time of BE.

Scenario IV Experimental Results



(a) *Demeter* (b) *PerfIso* (c) *PACT*

Demeter use as fewer CPU cores as possible

Scenario	Mechanism	Average power consumption ± SD(kJ)	BE finish time ± SD(s)	LC P95 max (ms)
IV (LC low + BE low)	Base	70.21±0.16	127.93±13.77	0.06
	PerfIso	46.27±0.47	130.71±6.49	0.09
	PACT	46.97±0.21	130.75±6.39	0.12
	Demeter$_{\text{-IDLE-BE's wait\_time}}$	42.38±0.42	130.80±6.25	0.28
	Demeter$_{\text{-IDLE}}$	39.03±0.47	131.62±4.19	0.28
	Demeter	38.47±0.47	131.29±5.02	0.28

Demeter improves the energy efficiency by more than 10% comparing to others.

in Top 3

Conclusions

- Black-box workloads in public clouds calls for new QoS-aware techniques for reducing power consumption.

- *Demeter* adopts a robust and online technique for classifying black-box workloads as BE or LC.

- *Demeter* provides differentiated CPU management strategies (including dynamic core allocation and frequency scaling) to both LC and BE workloads without any application level metrics.

- *Compared with SOTA mechanisms, Demeter* achieves considerable power savings (around average -10%) together with minimum impact on the performance of all workloads.

Thank you ！

Any questions?

Email: tangwenda@xjtu.edu.cn