Cornell University
Computer Systems Laboratory

PIMCloud: QoS-Aware Resource Management of Latency-Critical Applications in Clouds with Processing-in-Memory

Shuang Chen,* Yi Jiang, Christina Delimitrou, José F. Martínez

Cornell University
*Currently with Shuhai Lab at Huawei Cloud

Cloud Applications

Cloud Applications

Best-effort

hadoop

GraphLab

Cloud Applications

Best-effort

- Throughput-oriented
- No latency constraint

hadoop

GraphLab

CSL

Best-effort

- Throughput-oriented
- No latency constraint

Latency-critical

hadoop

GraphLab

Google Maps

Bing

Google Translate

Cloud Applications

Best-effort

- Throughput-oriented
- No latency constraint

hadoop

GraphLab

Latency-critical

- Tail latency
- Strict QoS constraint

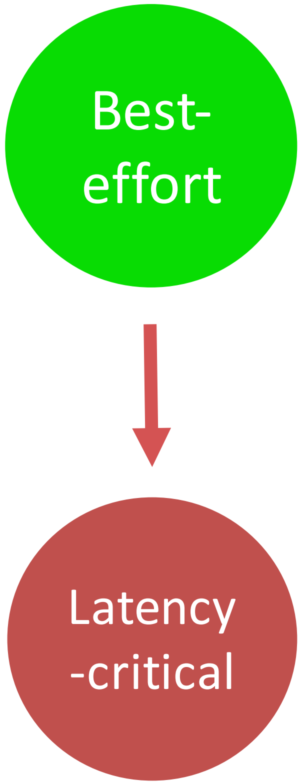
Google Maps

Bing

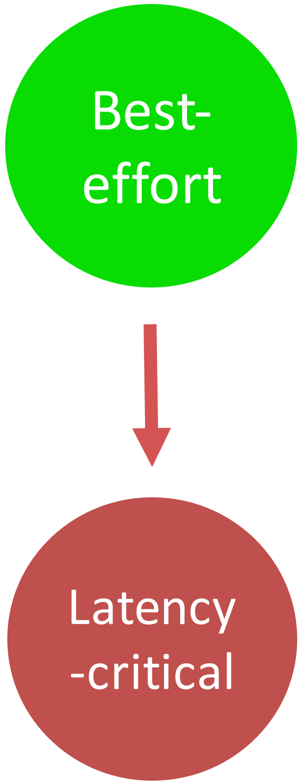
Google Translate

Software Trends in Clouds

Software Trends in Clouds



Software Trends in Clouds



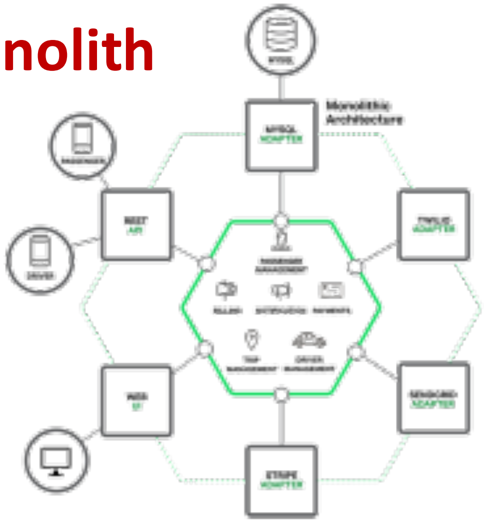
- More LC applications

Best-effort

↓

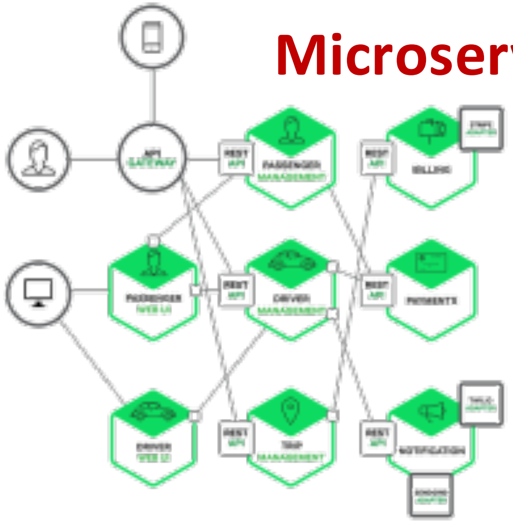
Latency-critical

Monolith



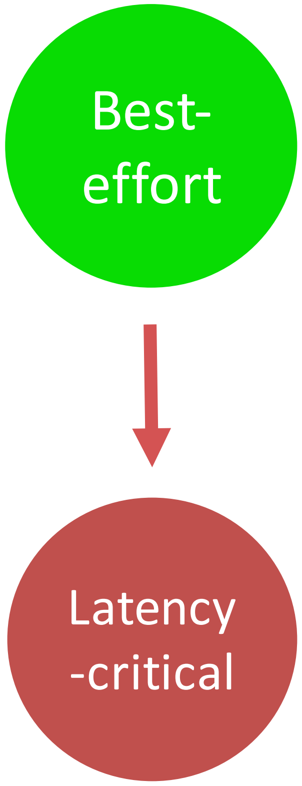
→

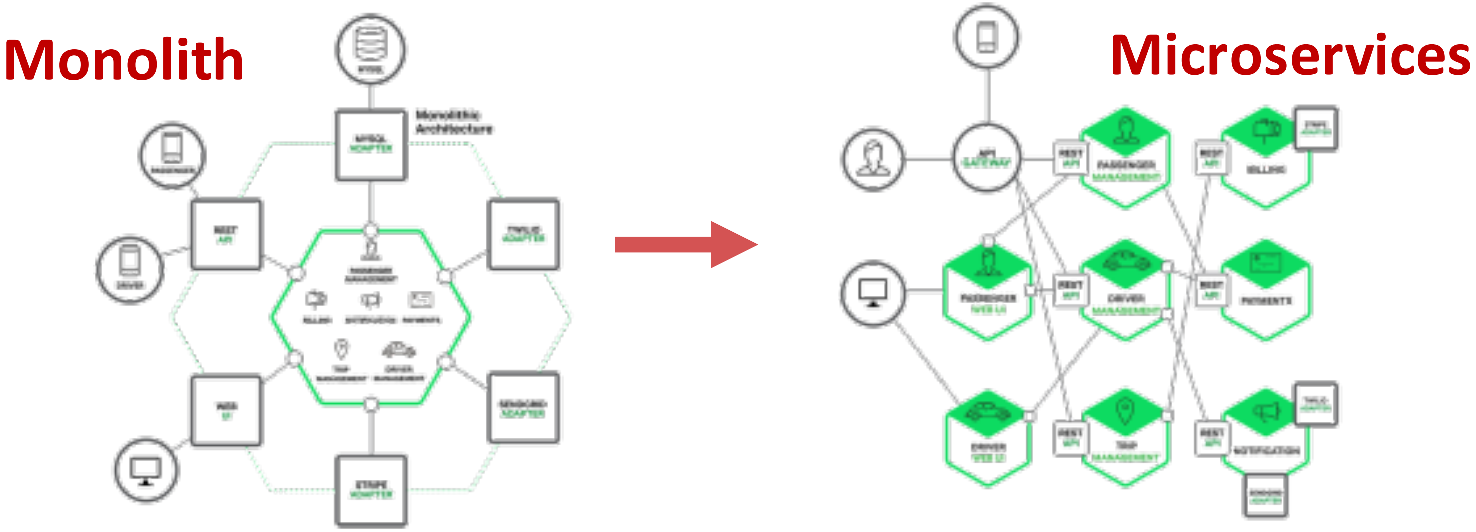
Microservices



- More LC applications

Software Trends in Clouds

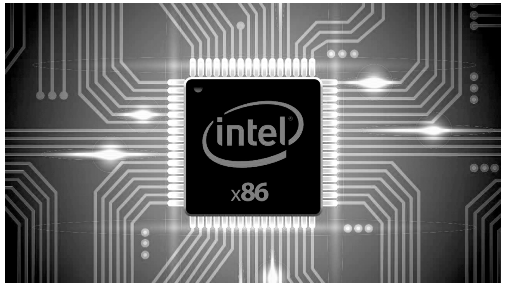




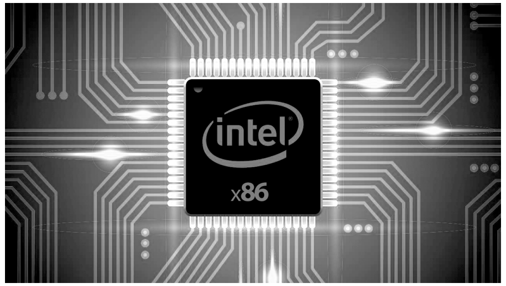
- More LC applications

- LC microservices
- Colocation of LC applications on the same node

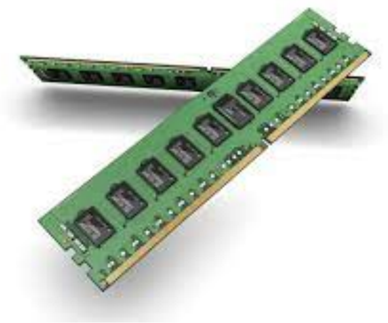
HARDWARE TRENDS IN CLOUDS

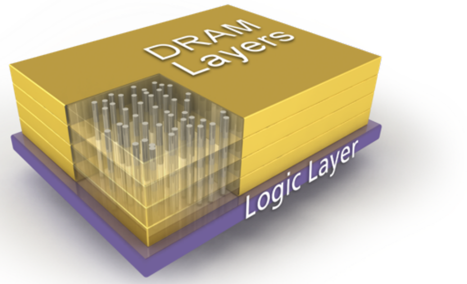


Heterogeneous computation

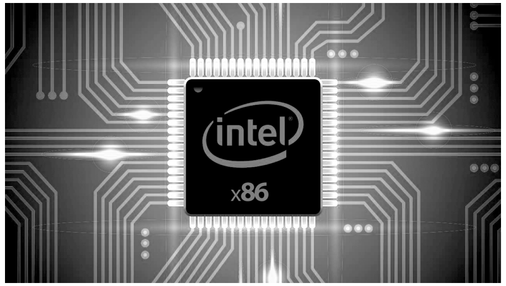


Heterogeneous computation

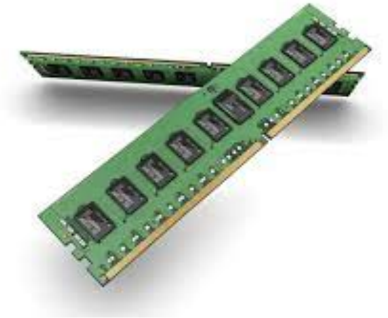


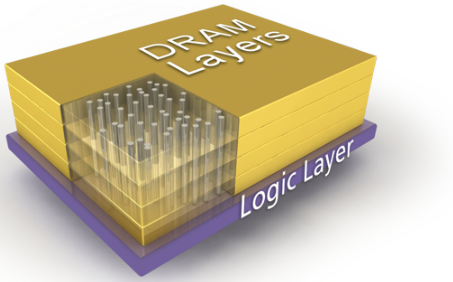


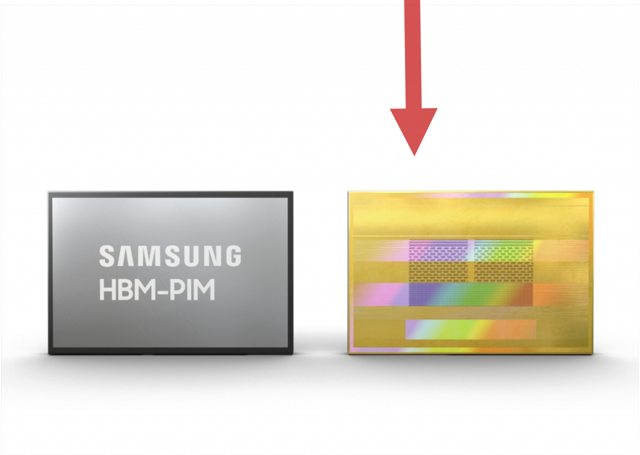
Hardware Trends in Clouds



Heterogeneous computation

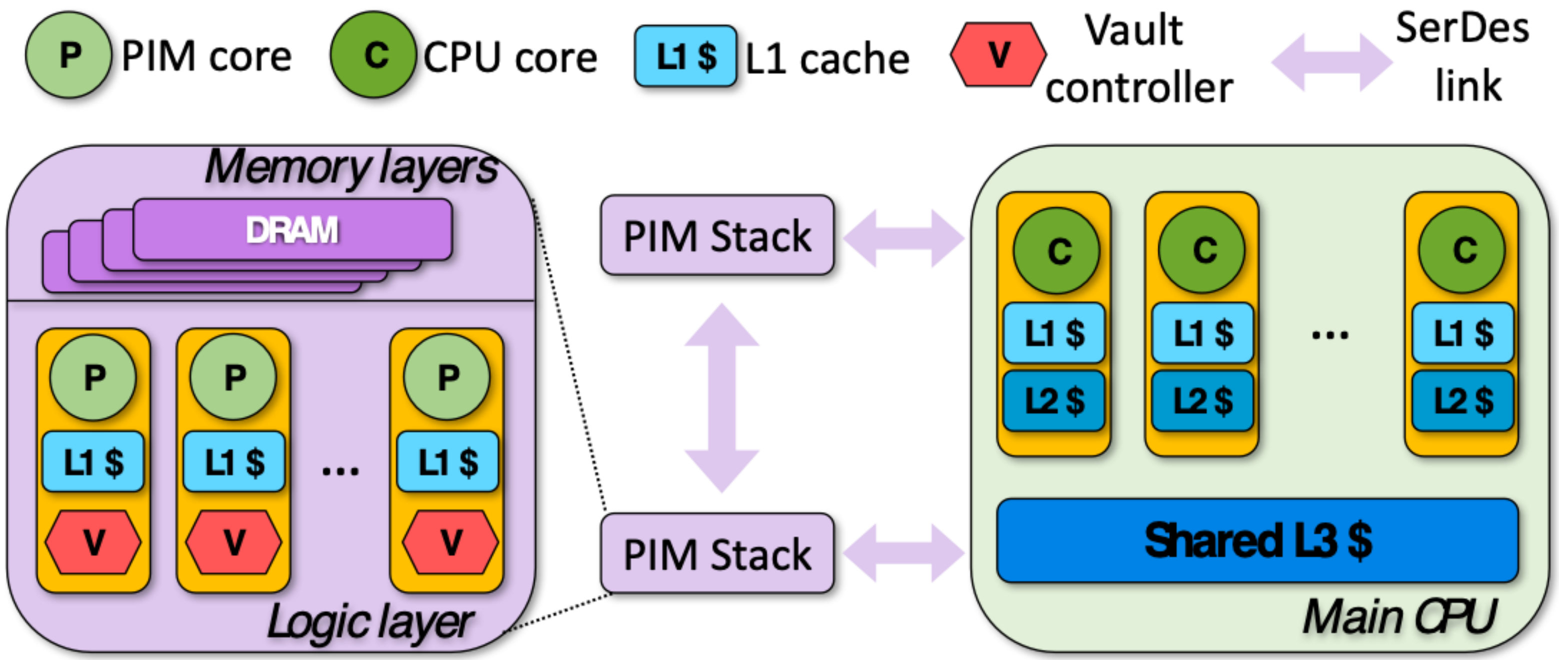




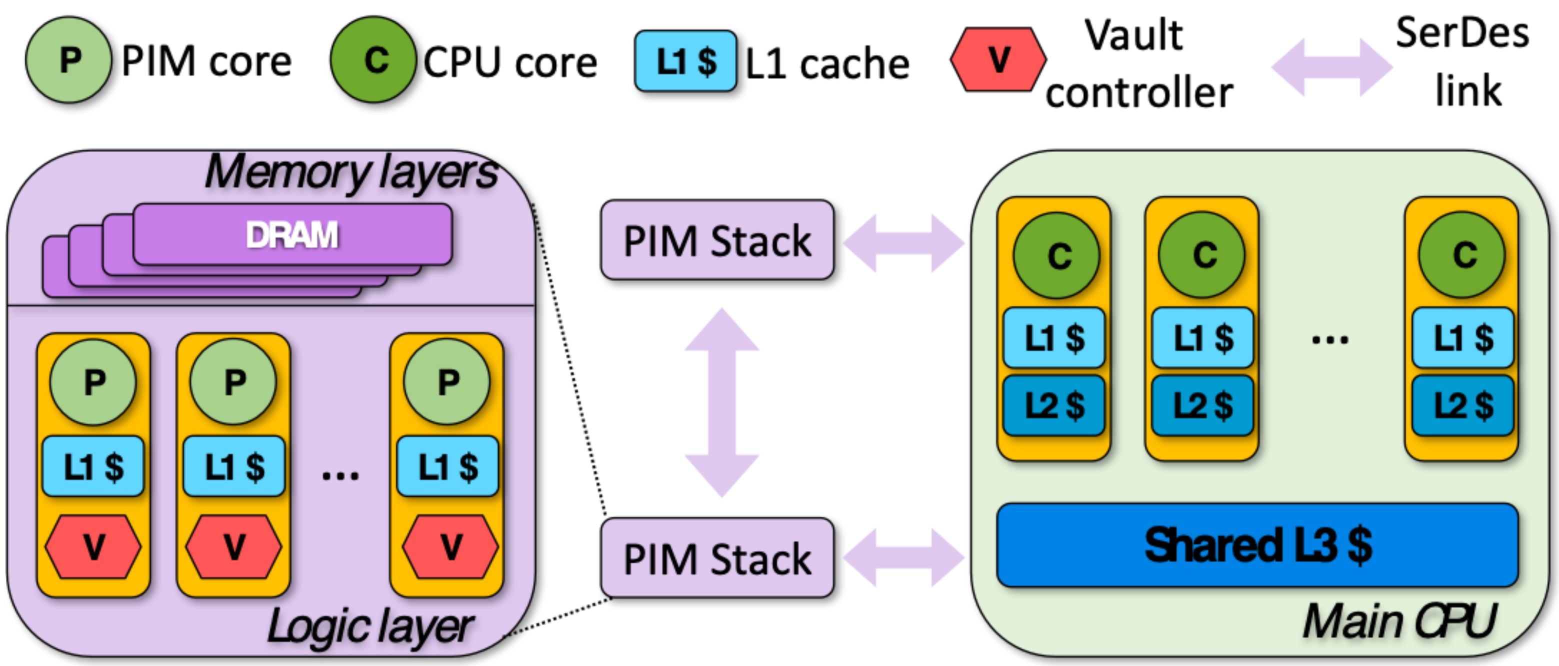


Heterogenous **memory accesses**

PIM-Enabled Cloud Server



PIM-Enabled Cloud Server



- Low memory latency
- Shallow memory hierarchy
- Wimpy core type
- Varying core count

Hardware Trend + **Software Trend**

Emerging PIM Platforms

Latency-Critical(LC) Cloud Applications

Hardware Trend + **Software Trend**

Emerging PIM Platforms

Latency-Critical(LC) Cloud Applications

How can LC applications leverage PIM?

Hardware Trend + **Software Trend**

Emerging PIM Platforms | Latency-Critical(LC) Cloud Applications

How can LC applications leverage PIM?

- **First study to explore PIM for latency-critical (LC) cloud applications**
- **Characterization**
  - To understand the implications of the PIM architecture to LC applications
- **PIMCloud: A QoS-aware resource manager for multiple LC applications in PIM-enabled systems**
  - Manages PIM-introduced resources

Hardware Trend + **Software Trend**

Emerging PIM Platforms

Latency-Critical(LC) Cloud Applications

How will LC applications perform on PIM?

- **The first to explore PIM for latency-critical (LC) cloud applications**
- ***Characterization***
  - *To understand the implications of the PIM architecture to LC applications*
- **PIMCloud: A QoS-aware resource manager for multiple LC applications in PIM-enabled systems**
  - Manages PIM-introduced resources

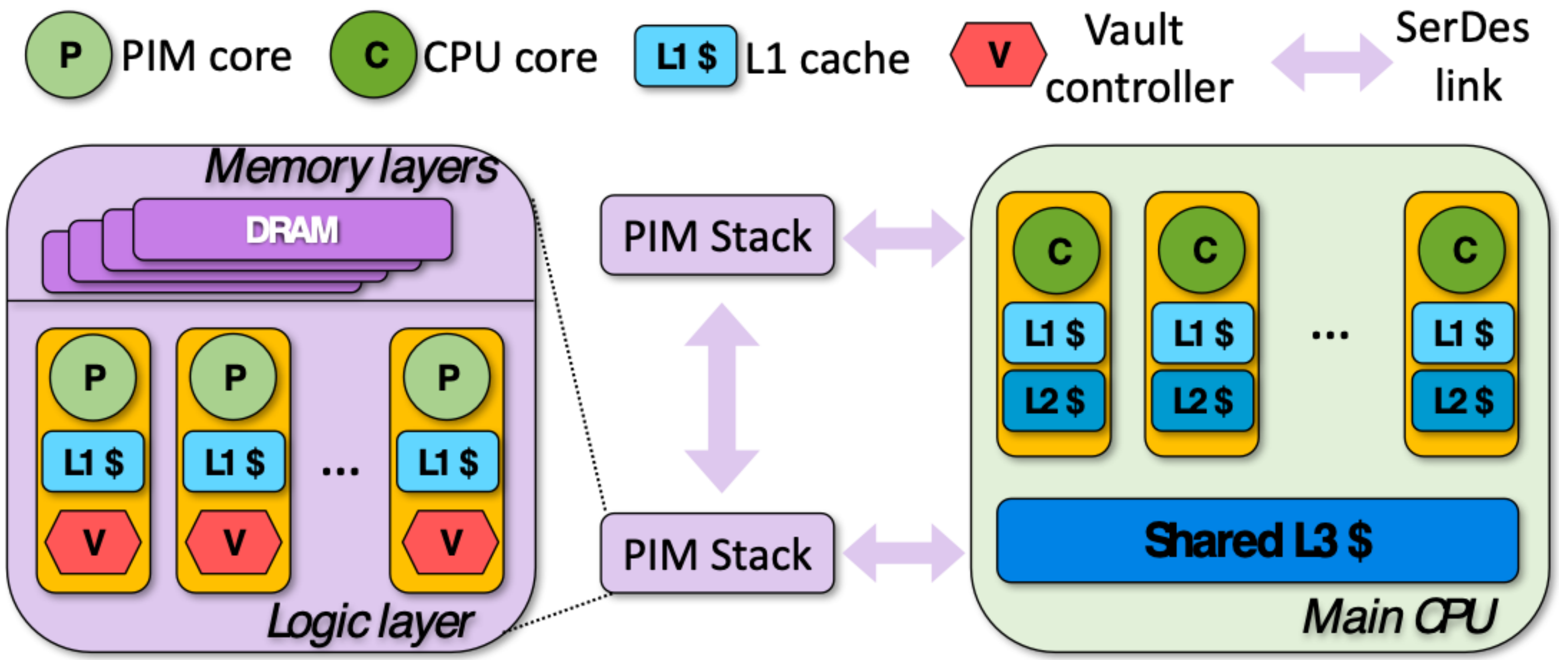
Application	Silo	Masstree	ImgDNN	Xapian	Moses	Sphinx
Domain	In-memory Database	Key-value store	Image recognition	Web search	Real-time translation	Speech recognition
Target QoS	1 ms	1 ms	7 ms	10 ms	10 ms	6 s
Per-core IPC	1.18	1.09	1.07	1.38	0.99	0.55
LLC MPKI	1.50	6.02	16.78	3.66	23.17	10.40
LLC Miss Rate	2%	12%	45%	37%	77%	47%
Memory Bandwidth (GB/s)	0.32	3.40	7.83	2.58	10.29	2.57
Memory Capacity (GB)	1.8	9.3	0.3	5.6	2.5	1.4

Six diverse LC applications from Tailbench [IISWC'16]

Application	Silo	Masstree	ImgDNN	Xapian	Moses	Sphinx
Domain	In-memory Database	Key-value store	Image recognition	Web search	Real-time translation	Speech recognition
Target QoS	1 ms	1 ms	7 ms	10 ms	10 ms	6 s
Per-core IPC	1.18	1.09	1.07	1.38	0.99	0.55
LLC MPKI	1.50	6.02	16.78	3.66	23.17	10.40
LLC Miss Rate	2%	12%	45%	37%	77%	47%
Memory Bandwidth (GB/s)	0.32	3.40	7.83	2.58	10.29	2.57
Memory Capacity (GB)	1.8	9.3	0.3	5.6	2.5	1.4

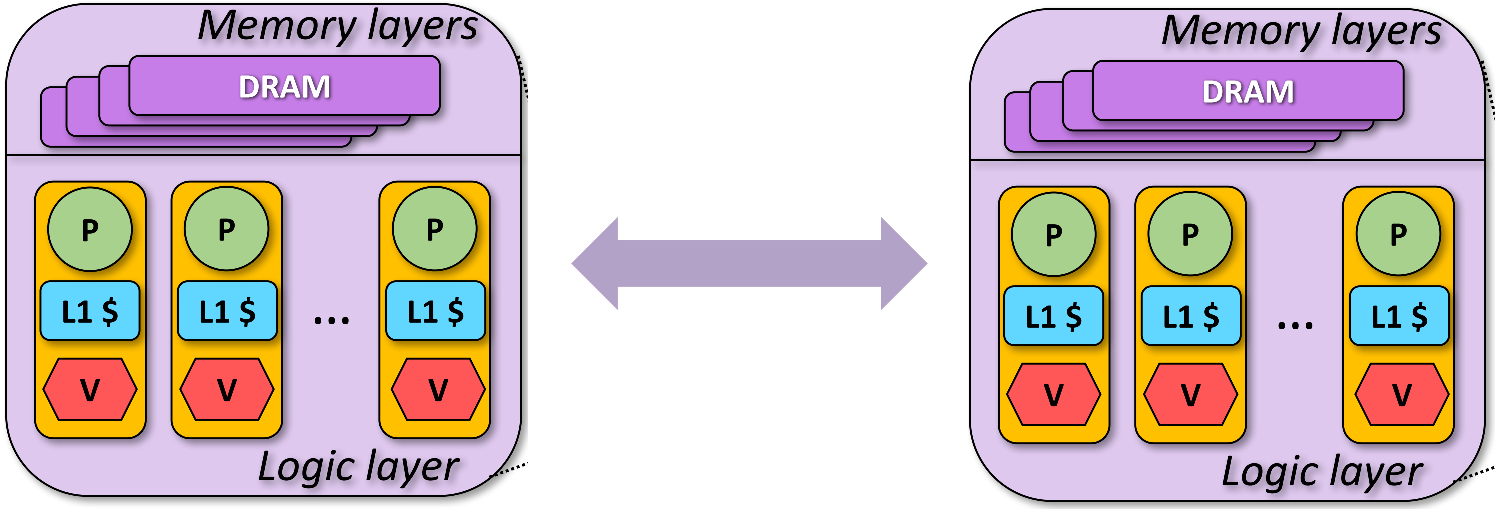
Six diverse LC applications from Tailbench [IISWC'16]

PIM-ENABLED CLOUD SERVER

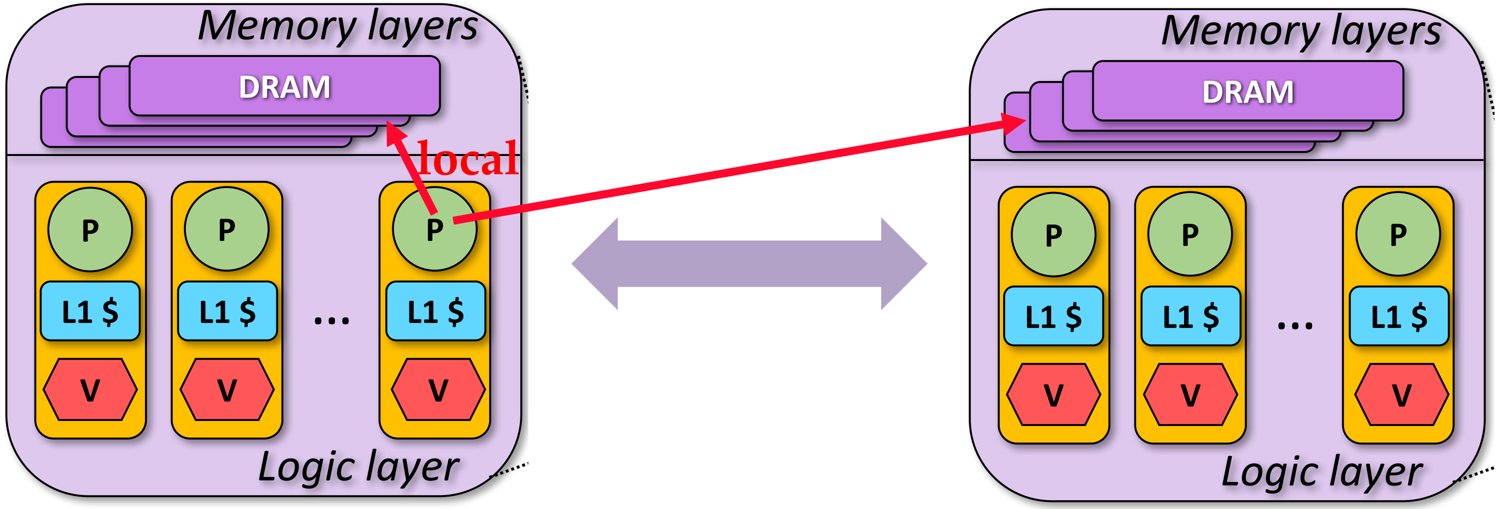


- Low memory latency
- Shallow memory hierarchy
- Wimpy core type
- Varying core count

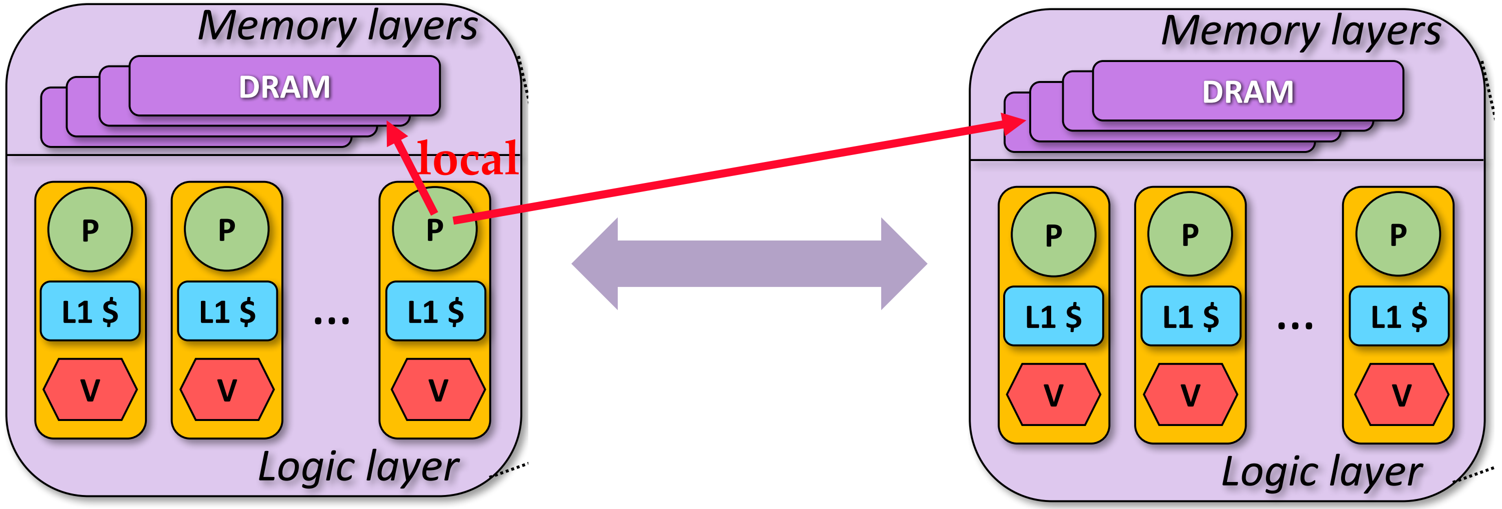
PIM-Enabled Cloud Server



PIM-Enabled Cloud Server



PIM-ENABLED CLOUD SERVER



- Which PIM stack to place each memory page?
- Local VS remote memory access for PIM cores
  - 20ns VS 35ns (VS 62ns from a CPU core)

IMPLICATIONS OF PIM – CORE & MEMORY

	Characterized Architecture			
ID	MemLat	Core	MemHie	#Cores
1	High	Brawny	Deep	4

	Characterized Architecture				
	ID	**MemLat**	**Core**	**MemHie**	**#Cores**
CPU-centric	1	High	Brawny	Deep	4
	2	High	Brawny	Shallow	
	3	High	Wimpy	Deep	
	4	High	Wimpy	Shallow	
Un-realistic	5	Low	Brawny	Deep	
	6	Low	Brawny	Shallow	
PIM-centric	7	Low	Wimpy	Deep	
	8	Low	Wimpy	Shallow	

IMPLICATIONS OF PIM – CORE & MEMORY

	Characterized Architecture				
	ID	MemLat	Core	MemHie	#Cores
CPU-centric	1	High	Brawny	Deep	4
	2	High	Brawny	Shallow	6
	3	High	Wimpy	Deep	10
	4	High	Wimpy	Shallow	16
Un-realistic	5	Low	Brawny	Deep	4
	6	Low	Brawny	Shallow	6
PIM-centric	7	Low	Wimpy	Deep	10
	8	Low	Wimpy	Shallow	16

Implications of PIM – Core & Memory

	Characterized Architecture				
	ID	**MemLat**	**Core**	**MemHie**	**#Cores**
CPU-centric	1	High	Brawny	Deep	4
	2	High	Brawny	Shallow	6
	3	High	Wimpy	Deep	10
	4	High	Wimpy	Shallow	16
Un-realistic	5	Low	Brawny	Deep	4
	6	Low	Brawny	Shallow	6
PIM-centric	7	Low	Wimpy	Deep	10
	8	Low	Wimpy	Shallow	16
PIM	9	Low*	Wimpy	Shallow	8+8

IMPLICATIONS OF PIM – CORE & MEMORY

	Characterized Architecture					Tail Latency/QoS Target						
	ID	MemLat	Core	MemHie	#Cores	Silo	Masstree	ImgDNN	Xapian	Moses	Sphinx	AVG
CPU-centric	1	High	Brawny	Deep	4	0.22	0.21	0.33	0.37	0.26	0.26	0.28
	2	High	Brawny	Shallow	6	0.36	0.24	0.28	0.40	0.35	0.29	0.32
	3	High	Wimpy	Deep	10	0.71	0.40	1.29	0.62	0.84	0.56	0.74
	4	High	Wimpy	Shallow	16	0.74	0.53	1.09	0.63	0.89	0.55	0.74
Un-realistic	5	Low	Brawny	Deep	4	0.21	0.18	0.25	0.36	0.22	0.23	0.24
	6	Low	Brawny	Shallow	6	0.28	0.20	0.20	0.37	0.27	0.27	0.26
PIM-centric	7	Low	Wimpy	Deep	10	0.59	0.34	0.97	0.59	0.67	0.54	0.62
	8	Low	Wimpy	Shallow	16	0.58	0.40	0.75	0.60	0.66	0.47	0.58
PIM	9	Low*	Wimpy	Shallow	8+8	0.61	0.47	0.95	0.61	0.77	0.49	0.65

Implications of PIM – Core & Memory

	Characterized Architecture					Tail Latency/QoS Target						
	ID	MemLat	Core	MemHie	#Cores	Silo	Masstree	ImgDNN	Xapian	Moses	Sphinx	AVG
CPU-centric	1	High	Brawny	Deep	4	0.22	0.21	0.33	0.37	0.26	0.26	0.28
	2	High	Brawny	Shallow	6	0.36	0.24	0.28	0.40	0.35	0.29	0.32
	3	High	Wimpy	Deep	10	0.71	0.40	1.29	0.62	0.84	0.56	0.74
	4	High	Wimpy	Shallow	16	0.74	0.53	1.09	0.63	0.89	0.55	0.74
Un-realistic	5	Low	Brawny	Deep	4	0.21	0.18	0.25	0.36	0.22	0.23	0.24
	6	Low	Brawny	Shallow	6	0.28	0.20	0.20	0.37	0.27	0.27	0.26
PIM-centric	7	Low	Wimpy	Deep	10	0.59	0.34	0.97	0.59	0.67	0.54	0.62
	8	Low	Wimpy	Shallow	16	0.58	0.40	0.75	0.60	0.66	0.47	0.58
PIM	9	Low*	Wimpy	Shallow	8+8	0.61	0.47	0.95	0.61	0.77	0.49	0.65

- QoS violations only occur in the combinations of wimpy cores and high memory latency

	Characterized Architecture					Tail Latency/QoS Target						
	ID	MemLat	Core	MemHie	#Cores	Silo	Masstree	ImgDNN	Xapian	Moses	Sphinx	AVG
CPU-centric	1	High	Brawny	Deep	4	0.22	0.21	0.33	0.37	0.26	0.26	0.28
	2	High	Brawny	Shallow	6	0.36	0.24	0.28	0.40	0.35	0.29	0.32
	3	High	Wimpy	Deep	10	0.71	0.40	1.29	0.62	0.84	0.56	0.74
	4	High	Wimpy	Shallow	16	0.74	0.53	1.09	0.63	0.89	0.55	0.74
Un-realistic	5	Low	Brawny	Deep	4	0.21	0.18	0.25	0.36	0.22	0.23	0.24
	6	Low	Brawny	Shallow	6	0.28	0.20	0.20	0.37	0.27	0.27	0.26
PIM-centric	7	Low	Wimpy	Deep	10	0.59	0.34	0.97	0.59	0.67	0.54	0.62
	8	Low	Wimpy	Shallow	16	0.58	0.40	0.75	0.60	0.66	0.47	0.58
PIM	9	Low*	Wimpy	Shallow	8+8	0.61	0.47	0.95	0.61	0.77	0.49	0.65

- QoS violations only occur in the combinations of wimpy cores and high memory latency

CSL

	Characterized Architecture					Tail Latency/QoS Target						
	ID	MemLat	Core	MemHie	#Cores	Silo	Masstree	ImgDNN	Xapian	Moses	Sphinx	AVG
CPU-centric	1	High	Brawny	Deep	4	0.22	0.21	0.33	0.37	0.26	0.26	0.28
	2	High	Brawny	Shallow	6	0.36	0.24	0.28	0.40	0.35	0.29	0.32
	3	High	Wimpy	Deep	10	0.71	0.40	1.29	0.62	0.84	0.56	0.74
	4	High	Wimpy	Shallow	16	0.74	0.53	1.09	0.63	0.89	0.55	0.74
Un-realistic	5	Low	Brawny	Deep	4	0.21	0.18	0.25	0.36	0.22	0.23	0.24
	6	Low	Brawny	Shallow	6	0.28	0.20	0.20	0.37	0.27	0.27	0.26
PIM-centric	7	Low	Wimpy	Deep	10	0.59	0.34	0.97	0.59	0.67	0.54	0.62
	8	Low	Wimpy	Shallow	16	0.58	0.40	0.75	0.60	0.66	0.47	0.58
PIM	9	Low*	Wimpy	Shallow	8+8	0.61	0.47	0.95	0.61	0.77	0.49	0.65

- QoS violations only occur in the combinations of wimpy cores and high memory latency
- PIM-centric architectures are able to meet QoS at low load

CSL

	Characterized Architecture					Tail Latency/QoS Target						
	ID	MemLat	Core	MemHie	#Cores	Silo	Masstree	ImgDNN	Xapian	Moses	Sphinx	AVG
CPU-centric	1	High	Brawny	Deep	4	0.22	0.21	0.33	0.37	0.26	0.26	0.28
	2	High	Brawny	Shallow	6	0.36	0.24	0.28	0.40	0.35	0.29	0.32
	3	High	Wimpy	Deep	10	0.71	0.40	1.29	0.62	0.84	0.56	0.74
	4	High	Wimpy	Shallow	16	0.74	0.53	1.09	0.63	0.89	0.55	0.74
Un-realistic	5	Low	Brawny	Deep	4	0.21	0.18	0.25	0.36	0.22	0.23	0.24
	6	Low	Brawny	Shallow	6	0.28	0.20	0.20	0.37	0.27	0.27	0.26
PIM-centric	7	Low	Wimpy	Deep	10	0.59	0.34	0.97	0.59	0.67	0.54	0.62
	8	Low	Wimpy	Shallow	16	0.58	0.40	0.75	0.60	0.66	0.47	0.58
PIM	9	Low*	Wimpy	Shallow	8+8	0.61	0.47	0.95	0.61	0.77	0.49	0.65

- QoS violations only occur in the combinations of wimpy cores and high memory latency
- PIM-centric architectures are able to meet QoS at low load

	Characterized Architecture					Tail Latency/QoS Target						
	ID	MemLat	Core	MemHie	#Cores	Silo	Masstree	ImgDNN	Xapian	Moses	Sphinx	AVG
CPU-centric	1	High	Brawny	Deep	4	0.22	0.21	0.33	0.37	0.26	0.26	0.28
	2	High	Brawny	Shallow	6	0.36	0.24	0.28	0.40	0.35	0.29	0.32
	3	High	Wimpy	Deep	10	0.71	0.40	1.29	0.62	0.84	0.56	0.74
	4	High	Wimpy	Shallow	16	0.74	0.53	1.09	0.63	0.89	0.55	0.74
Un-realistic	5	Low	Brawny	Deep	4	0.21	0.18	0.25	0.36	0.22	0.23	0.24
	6	Low	Brawny	Shallow	6	0.28	0.20	0.20	0.37	0.27	0.27	0.26
PIM-centric	7	Low	Wimpy	Deep	10	0.59	0.34	0.97	0.59	0.67	0.54	0.62
	8	Low	Wimpy	Shallow	16	0.58	0.40	0.75	0.60	0.66	0.47	0.58
PIM	9	Low*	Wimpy	Shallow	8+8	0.61	0.47	0.95	0.61	0.77	0.49	0.65

- QoS violations only occur in the combinations of wimpy cores and high memory latency
- PIM-centric architectures are able to meet QoS at low load

Implications of PIM – Core & Memory

	Characterized Architecture					Normalized Max Load (Max RPS under QoS)						
	ID	MemLat	Core	MemHie	#Cores	Silo	Masstree	ImgDNN	Xapian	Moses	Sphinx	AVG
CPU-centric	1	High	Brawny	Deep	4	1.00	1.00	1.00	1.00	1.00	1.00	1.00
	2	High	Brawny	Shallow	6	0.89	0.85	1.17	1.33	1.41	1.25	1.15
	3	High	Wimpy	Deep	10	0.68	0.72	0.00	0.87	0.55	1.50	0.72
	4	High	Wimpy	Shallow	16	0.53	0.93	0.00	1.12	1.09	2.25	0.99
Un-realistic	5	Low	Brawny	Deep	4	1.08	1.09	1.20	1.05	1.23	1.20	1.14
	6	Low	Brawny	Shallow	6	1.05	1.00	1.66	1.50	1.82	1.50	1.42
PIM-centric	7	Low	Wimpy	Deep	10	0.89	0.85	0.71	1.13	0.91	1.75	1.04
	8	Low	Wimpy	Shallow	16	0.79	1.15	0.93	1.65	1.82	2.65	1.50
PIM	9	Low*	Wimpy	Shallow	8+8	0.68	1.09	0.63	1.58	1.50	2.50	1.33

Implications of PIM – Core & Memory

	Characterized Architecture					Normalized Max Load (Max RPS under QoS)						
	ID	MemLat	Core	MemHie	#Cores	Silo	Masstree	ImgDNN	Xapian	Moses	Sphinx	AVG
CPU-centric	1	High	Brawny	Deep	4	1.00	1.00	1.00	1.00	1.00	1.00	1.00
	2	High	Brawny	Shallow	6	0.89	0.85	1.17	1.33	1.41	1.25	1.15
	3	High	Wimpy	Deep	10	0.68	0.72	0.00	0.87	0.55	1.50	0.72
	4	High	Wimpy	Shallow	16	0.53	0.93	0.00	1.12	1.09	2.25	0.99
Un-realistic	5	Low	Brawny	Deep	4	1.08	1.09	1.20	1.05	1.23	1.20	1.14
	6	Low	Brawny	Shallow	6	1.05	1.00	1.66	1.50	1.82	1.50	1.42
PIM-centric	7	Low	Wimpy	Deep	10	0.89	0.85	0.71	1.13	0.91	1.75	1.04
	8	Low	Wimpy	Shallow	16	0.79	1.15	0.93	1.65	1.82	2.65	1.50
PIM	9	Low*	Wimpy	Shallow	8+8	0.68	1.09	0.63	1.58	1.50	2.50	1.33

Implications of PIM – Core & Memory

	Characterized Architecture					Normalized Max Load (Max RPS under QoS)						
	ID	MemLat	Core	MemHie	#Cores	Silo	Masstree	ImgDNN	Xapian	Moses	Sphinx	AVG
CPU-centric	1	High	Brawny	Deep	4	1.00	1.00	1.00	1.00	1.00	1.00	1.00
	2	High	Brawny	Shallow	6	0.89	0.85	1.17	1.33	1.41	1.25	1.15
	3	High	Wimpy	Deep	10	0.68	0.72	0.00	0.87	0.55	1.50	0.72
	4	High	Wimpy	Shallow	16	0.53	0.93	0.00	1.12	1.09	2.25	0.99
Un-realistic	5	Low	Brawny	Deep	4	1.08	1.09	1.20	1.05	1.23	1.20	1.14
	6	Low	Brawny	Shallow	6	1.05	1.00	1.66	1.50	1.82	1.50	1.42
PIM-centric	7	Low	Wimpy	Deep	10	0.89	0.85	0.71	1.13	0.91	1.75	1.04
	8	Low	Wimpy	Shallow	16	0.79	1.15	0.93	1.65	1.82	2.65	1.50
PIM	9	Low*	Wimpy	Shallow	8+8	0.68	1.09	0.63	1.58	1.50	2.50	1.33

- On average, PIM-centric architectures outperform CPU-centric ones

Implications of PIM – Core & Memory

	Characterized Architecture					Normalized Max Load (Max RPS under QoS)						
	ID	MemLat	Core	MemHie	#Cores	Silo	Masstree	ImgDNN	Xapian	Moses	Sphinx	AVG
CPU-centric	1	High	Brawny	Deep	4	1.00	1.00	1.00	1.00	1.00	1.00	1.00
	2	High	Brawny	Shallow	6	0.89	0.85	1.17	1.33	1.41	1.25	1.15
	3	High	Wimpy	Deep	10	0.68	0.72	0.00	0.87	0.55	1.50	0.72
	4	High	Wimpy	Shallow	16	0.53	0.93	0.00	1.12	1.09	2.25	0.99
Un-realistic	5	Low	Brawny	Deep	4	1.08	1.09	1.20	1.05	1.23	1.20	1.14
	6	Low	Brawny	Shallow	6	1.05	1.00	1.66	1.50	1.82	1.50	1.42
PIM-centric	7	Low	Wimpy	Deep	10	0.89	0.85	0.71	1.13	0.91	1.75	1.04
	8	Low	Wimpy	Shallow	16	0.79	1.15	0.93	1.65	1.82	2.65	1.50
PIM	9	Low*	Wimpy	Shallow	8+8	0.68	1.09	0.63	1.58	1.50	2.50	1.33

- **On average, PIM-centric architectures outperform CPU-centric ones**
  - Up to 52% gain from low memory latency
  - Up to 44% gain from shallow memory hierarchy
  - Up to 5% gain from many wimpy cores

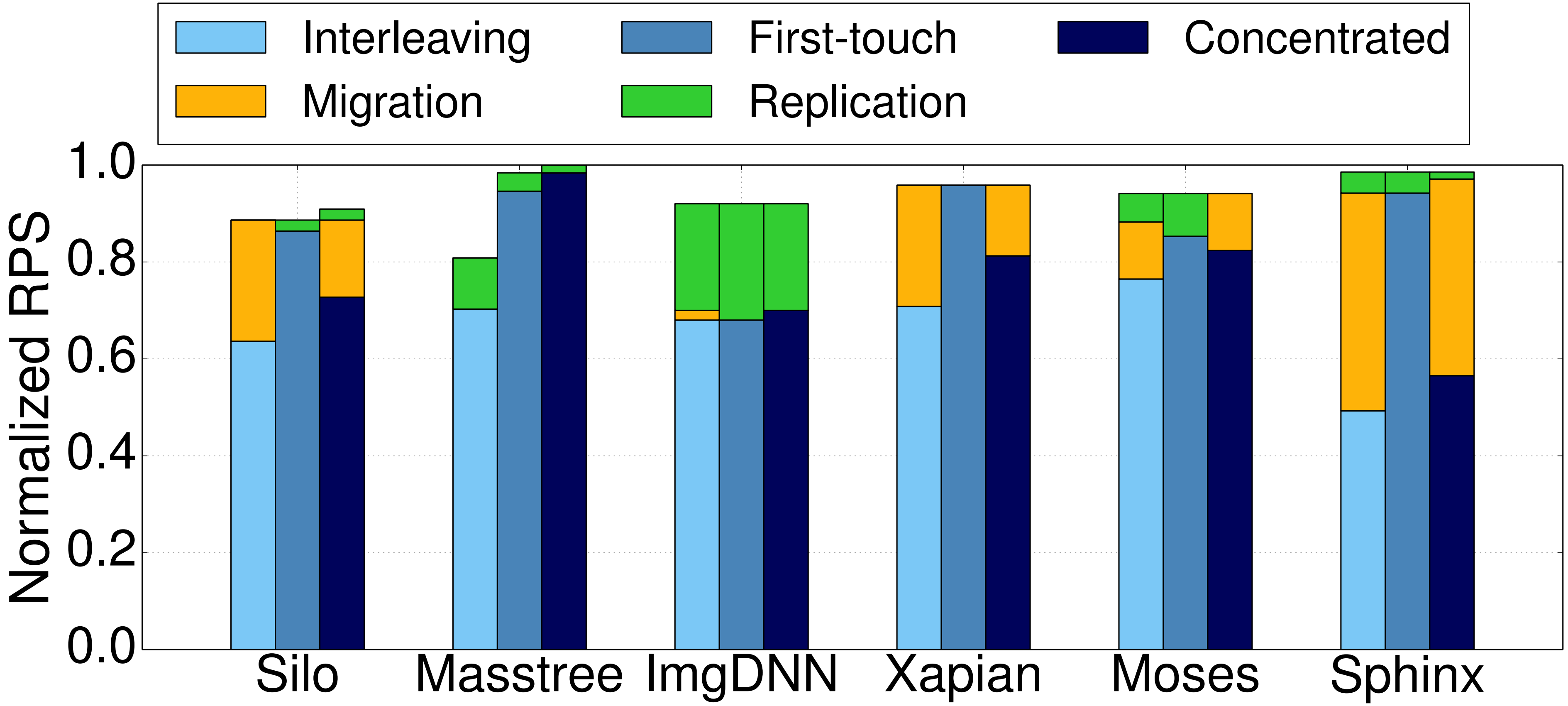
Implications of PIM – Core & Memory

CSL

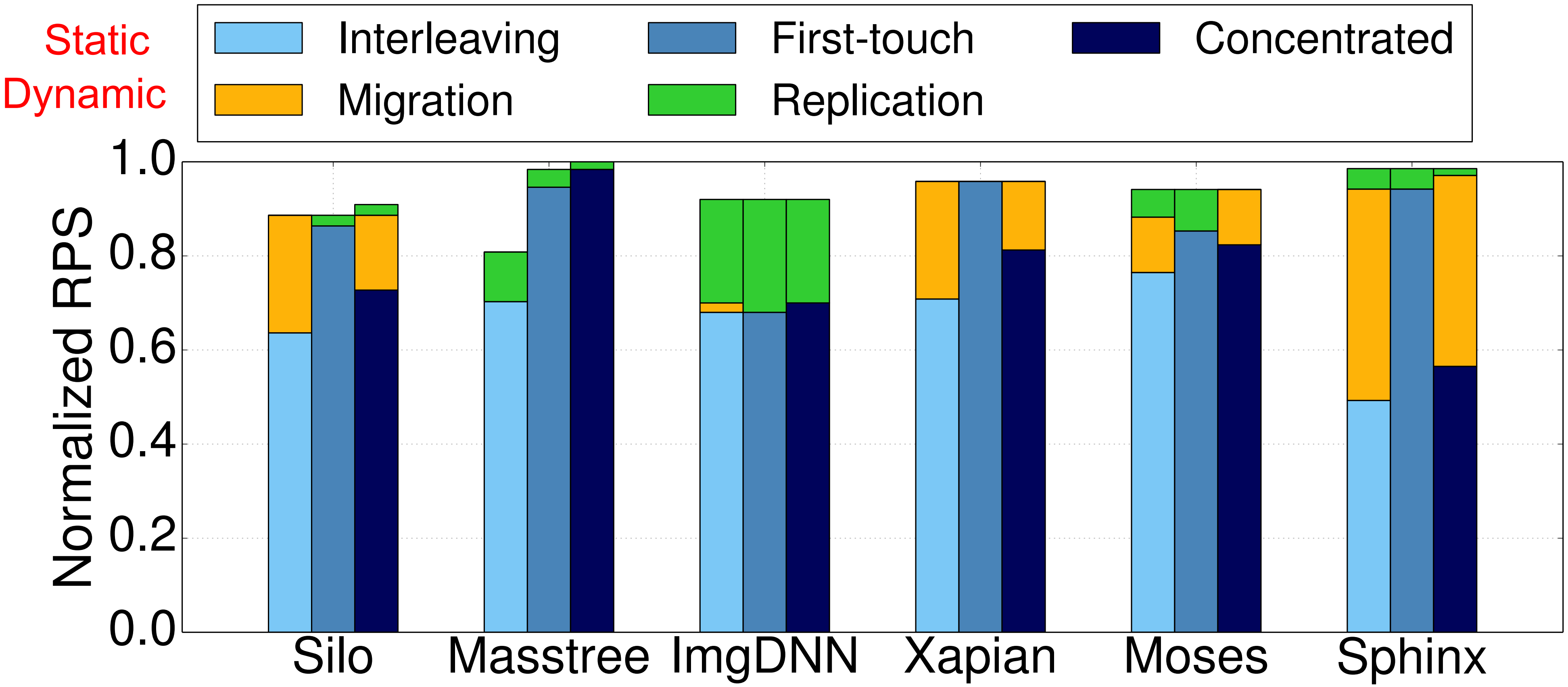
	Characterized Architecture					Normalized Max Load (Max RPS under QoS)						
	ID	MemLat	Core	MemHie	#Cores	Silo	Masstree	ImgDNN	Xapian	Moses	Sphinx	AVG
CPU-centric	1	High	Brawny	Deep	4	1.00	1.00	1.00	1.00	1.00	1.00	1.00
	2	High	Brawny	Shallow	6	0.89	0.85	1.17	1.33	1.41	1.25	1.15
	3	High	Wimpy	Deep	10	0.68	0.72	0.00	0.87	0.55	1.50	0.72
	4	High	Wimpy	Shallow	16	0.53	0.93	0.00	1.12	1.09	2.25	0.99
Un-realistic	5	Low	Brawny	Deep	4	1.08	1.09	1.20	1.05	1.23	1.20	1.14
	6	Low	Brawny	Shallow	6	1.05	1.00	1.66	1.50	1.82	1.50	1.42
PIM-centric	7	Low	Wimpy	Deep	10	0.89	0.85	0.71	1.13	0.91	1.75	1.04
	8	Low	Wimpy	Shallow	16	0.79	1.15	0.93	1.65	1.82	2.65	1.50
PIM	9	Low*	Wimpy	Shallow	8+8	0.68	1.09	0.63	1.58	1.50	2.50	1.33

- **On average, PIM-centric architectures outperform CPU-centric ones**
  - Up to 52% gain from low memory latency
  - Up to 44% gain from shallow memory hierarchy
  - Up to 5% gain from many wimpy cores
- **Individual applications have different preferences over CPU and PIM**

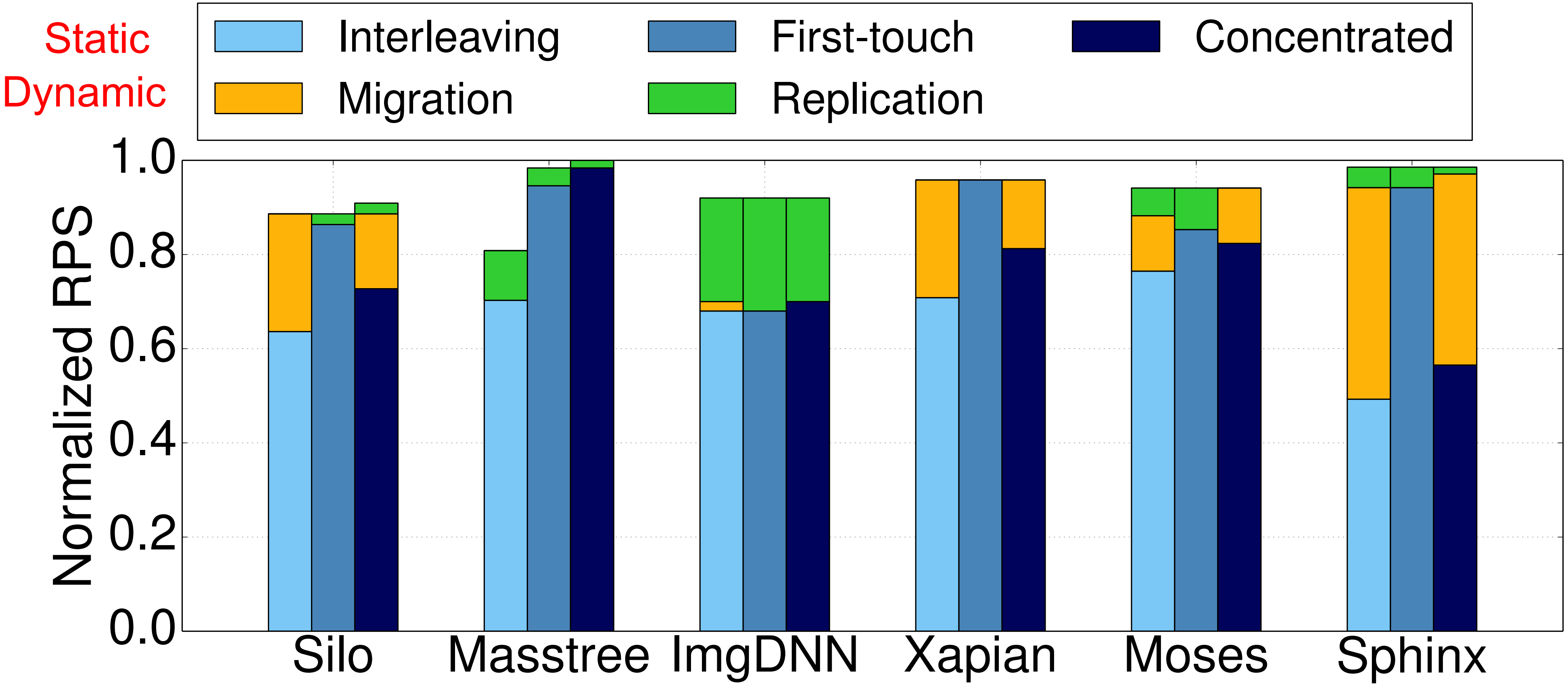
Implications of PIM – Data Placement



Implications of PIM – Data Placement



Implications of PIM – Data Placement



- **Dynamic page manipulation (page migration+replication) is essential to achieve the best performance.**

Takeaways from Characterization

- **LC applications have varying preference to PIM**
  - At runtime, it is critical to be aware of the heterogeneity, and allocate the right type of resources to each application
- **Dynamic data placement is critical to achieve the best performance**

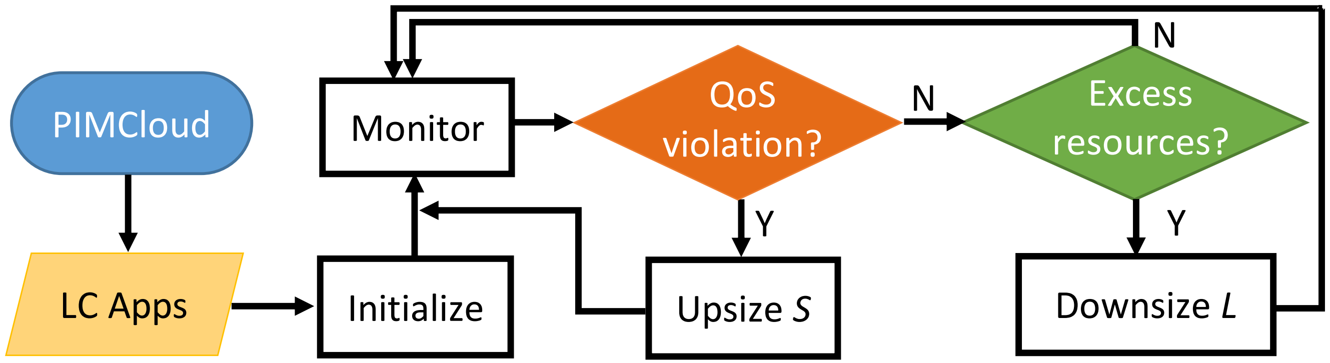
Hardware Trend + **Software Trend**

Emerging PIM Platforms | Latency-Critical(LC) Cloud Applications

How will LC applications perform on PIM?

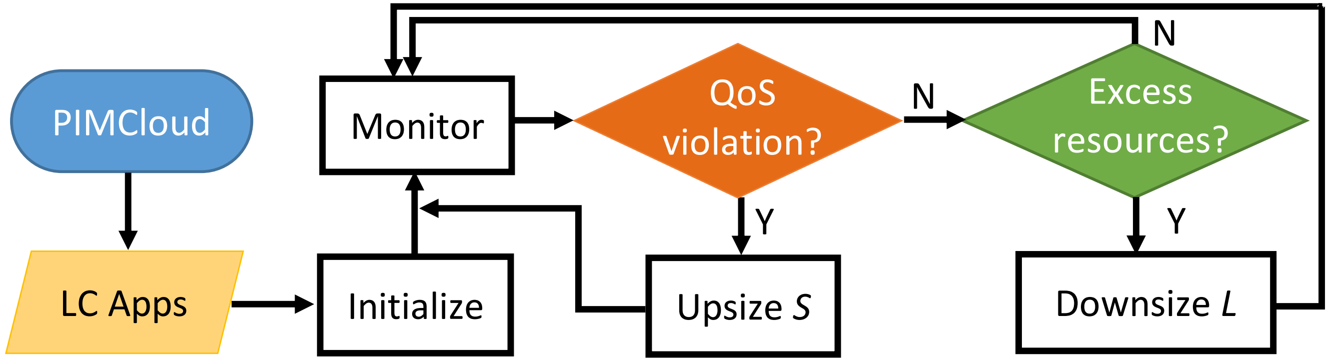
- **The first to explore PIM for latency-critical (LC) cloud applications**
- **Characterization**
  - To understand the implications of the PIM architecture to LC applications
- ***PIMCloud: A QoS-aware resource manager for multiple LC applications in PIM-enabled systems***
  - *Manages PIM-introduced resources*

PIMCLoud – Resource Management



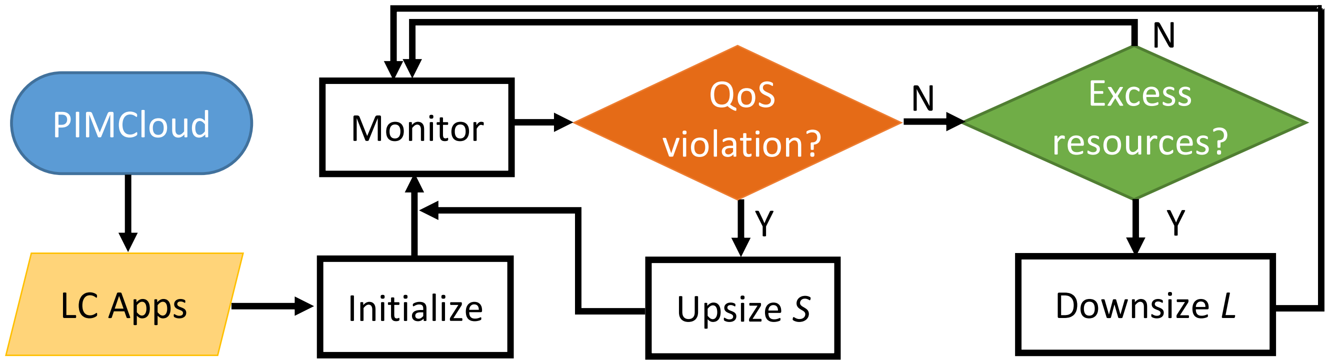
PIMCLoud – Resource Management

- The same feedback-control loops as PARTIES [ASPLOS'19]

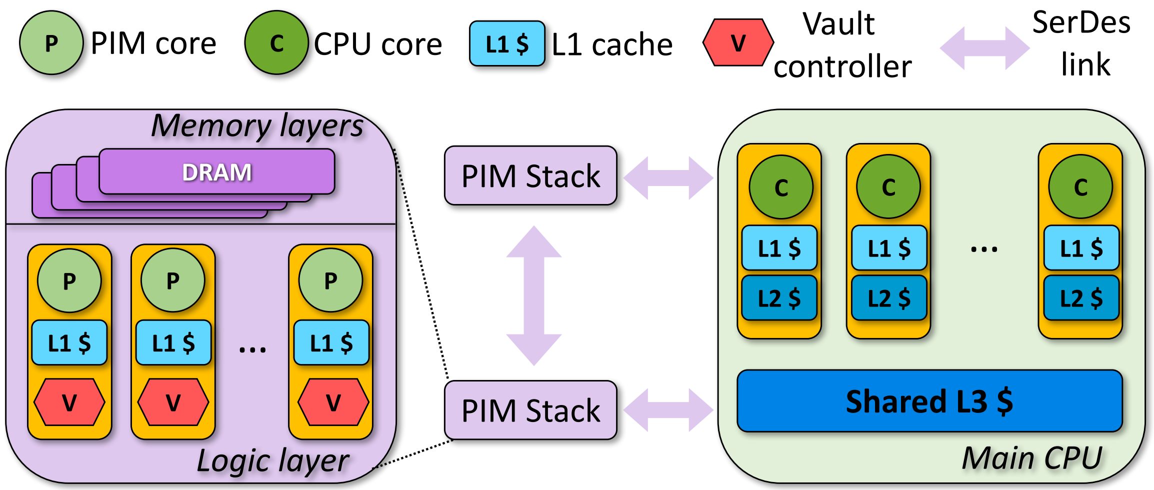


PIMCLoud – Resource Management

- **The same feedback-control loops as PARTIES [ASPLOS'19]**



- **Upsize/Downsize handles resource adjustment**
  - Core allocation
    - Core type
    - Core count
  - Data placement
    - Pages to migrate/replicate at runtime



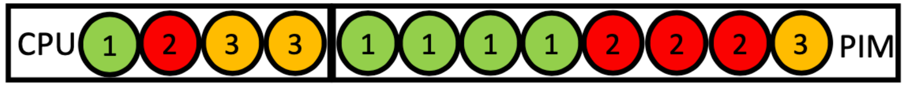
PIMCloud – Core Allocation

PIMCloud – Core Allocation

- **Main challenge: reduce the allocation space**

PIMCloud – Core Allocation

- Main challenge: reduce the allocation space

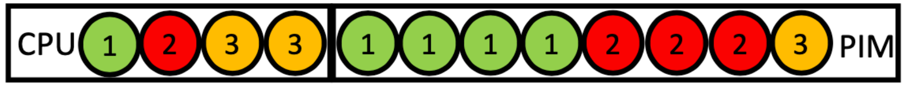


(a) Preference-oblivious managers: mixed cores for each app.

PIMCloud – Core Allocation

- **Main challenge: reduce the allocation space**

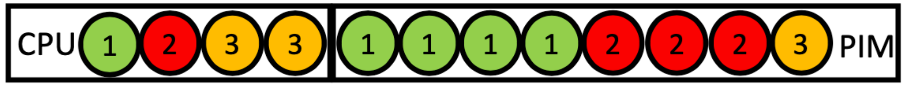
- **Pre-sorting offline**
  - A quick offline profiling to obtain each application's preference
  - Sort applications in decreasing preference to PIM



(a) Preference-oblivious managers: mixed cores for each app.

PIMCloud – Core Allocation

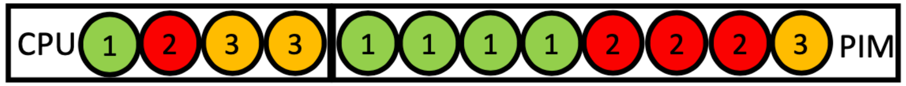
- **Main challenge: reduce the allocation space**
- **Pre-sorting offline**
  - A quick offline profiling to obtain each application's preference
  - Sort applications in decreasing preference to PIM
- **At runtime**
  - Applications are allocated in order
  - The allocation space is the same as a homogeneous setting



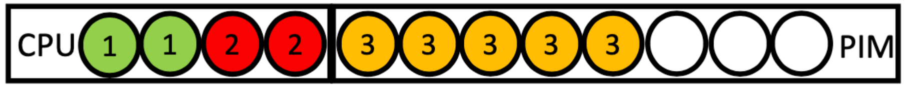
(a) Preference-oblivious managers: mixed cores for each app.

PIMCloud – Core Allocation

- **Main challenge: reduce the allocation space**
- **Pre-sorting offline**
  - A quick offline profiling to obtain each application's preference
  - Sort applications in decreasing preference to PIM
- **At runtime**
  - Applications are allocated in order
  - The allocation space is the same as a homogeneous setting



(a) Preference-oblivious managers: mixed cores for each app.

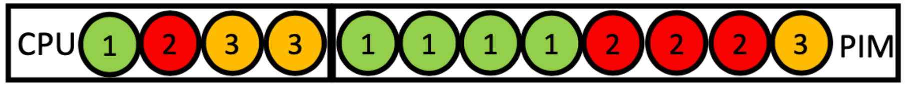


(b) Preference-aware PIMCloud: reduced allocation space, and more saving in cores under the same performance.

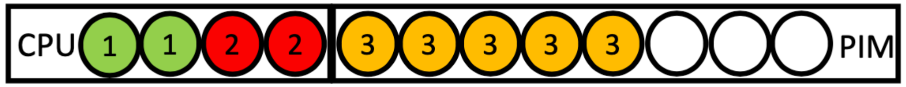
PIMCloud – Core Allocation

- **Main challenge: reduce the allocation space**

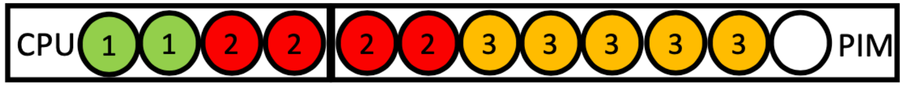
- **Pre-sorting offline**
  - A quick offline profiling to obtain each application’s preference
  - Sort applications in decreasing preference to PIM
- **At runtime**
  - Applications are allocated in order
  - The allocation space is the same as a homogeneous setting



(a) Preference-oblivious managers: mixed cores for each app.



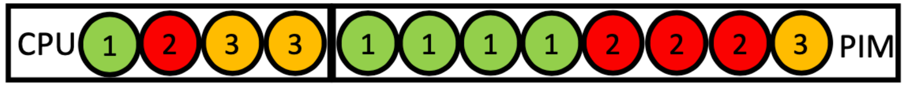
(b) Preference-aware PIMCloud: reduced allocation space, and more saving in cores under the same performance.



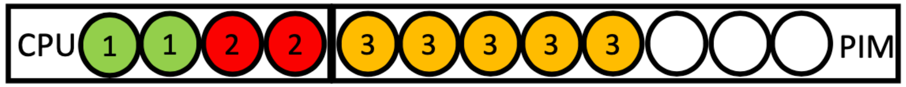
(c) PIMCloud at runtime when App2’s input load increases: the preference order is maintained.

PIMCloud – Core Allocation

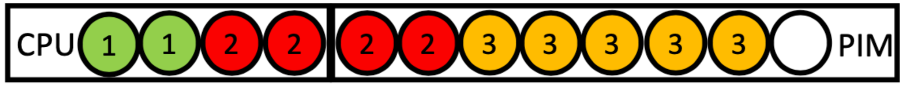
- **Main challenge: reduce the allocation space**
- **Pre-sorting offline**
  - A quick offline profiling to obtain each application's preference
  - Sort applications in decreasing preference to PIM
- **At runtime**
  - Applications are allocated in order
  - The allocation space is the same as a homogeneous setting
- **Theoretical analysis in the paper**



(a) Preference-oblivious managers: mixed cores for each app.



(b) Preference-aware PIMCloud: reduced allocation space, and more saving in cores under the same performance.



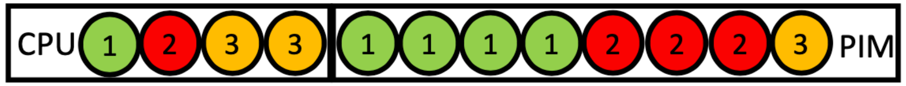
(c) PIMCloud at runtime when App2's input load increases: the preference order is maintained.

PIMCloud – Core Allocation

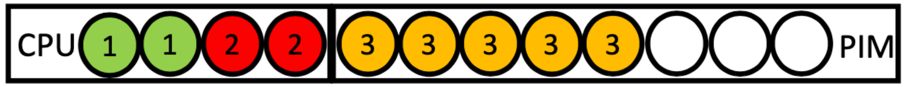
- **Main challenge: reduce the allocation space**

- **Pre-sorting offline**
  - A quick offline profiling to obtain each application's preference
  - Sort applications in decreasing preference to PIM
- **At runtime**
  - Applications are allocated in order
  - The allocation space is the same as a homogeneous setting

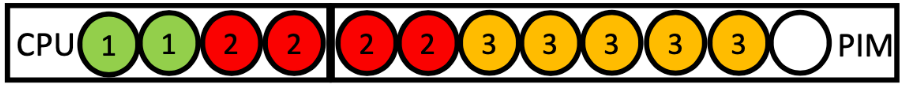
- **Theoretical analysis in the paper**



(a) Preference-oblivious managers: mixed cores for each app.



(b) Preference-aware PIMCloud: reduced allocation space, and more saving in cores under the same performance.



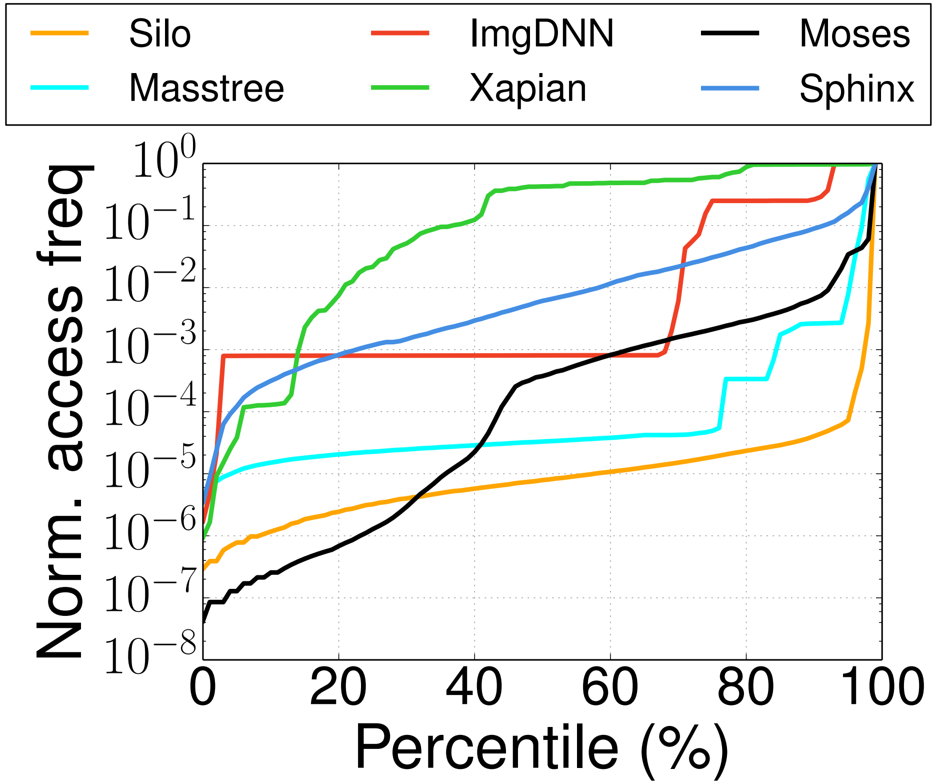
(c) PIMCloud at runtime when App2's input load increases: the preference order is maintained.

PIMCloud – Data Placement

- Main challenge: reduce the number of migrated/replicated pages
- Only the hottest pages are manipulated at runtime

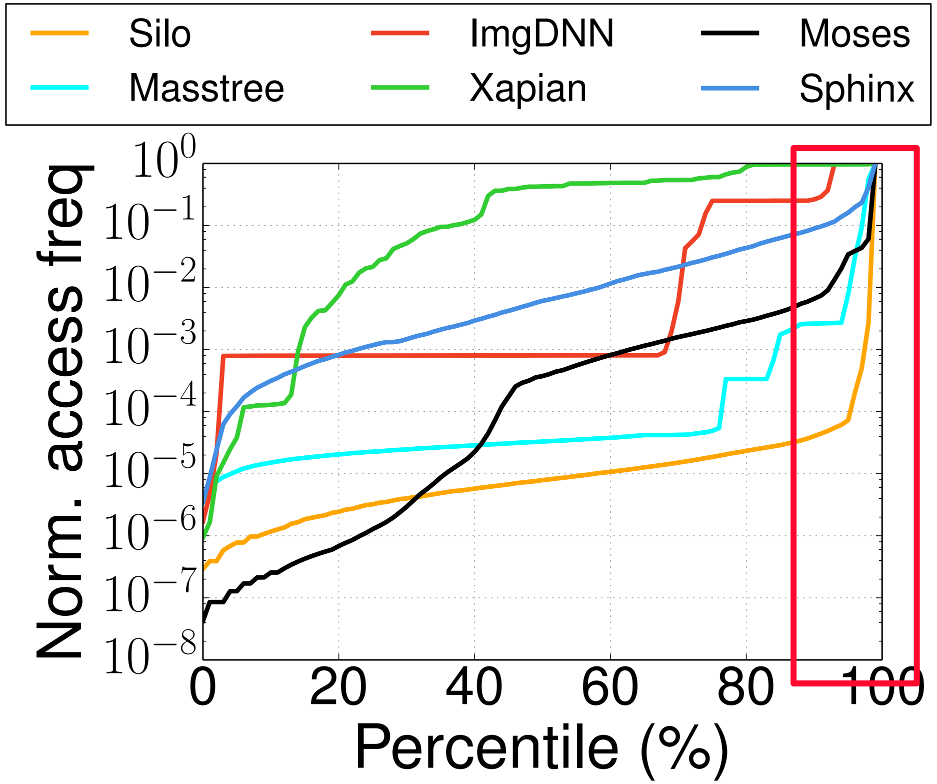
PIMCloud – Data Placement

- **Main challenge: reduce the number of migrated/replicated pages**
- **Only the hottest pages are manipulated at runtime**



PIMCloud – Data Placement

- **Main challenge: reduce the number of migrated/replicated pages**
- **Only the hottest pages are manipulated at runtime**



Evaluation Methodology

Simulator: ZSim

- CPU: 4 Haswell-like cores, 2.4 GHz, 32KB L1, 256KB L2, 8MB L3
- PIM: 8 ARM Cortex-A57-like cores per memory stack, 2 GHz, 32KB L1
- Extend the memory model to HBM
  - »16 vaults, 160GB/s peak memory bandwidth

Applications: Tailbench

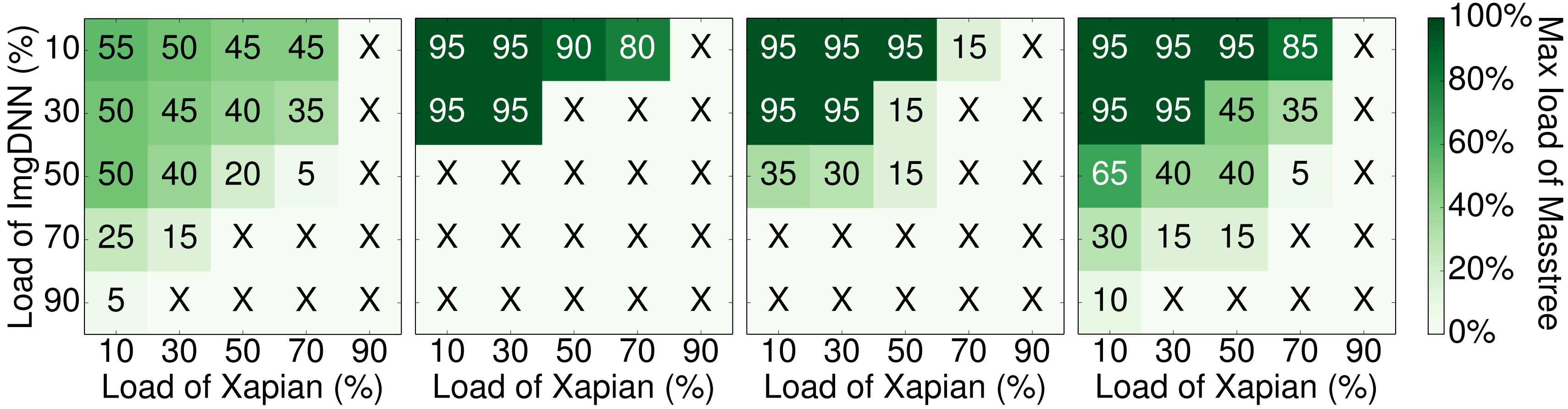
- 20 threads
- 20s warmup, 10s execution (about 72 billion cycles)
- Run on 8 Haswell-like cores by default

Baselines:

- **Default**: relying on the OS to manage resources
- **AMS [MICRO'18]**: a scheduler for batch jobs in PIM systems
- **Octopus-Man [HPCA'15]:** a scheduler for LC apps in systems with heterogeneous cores

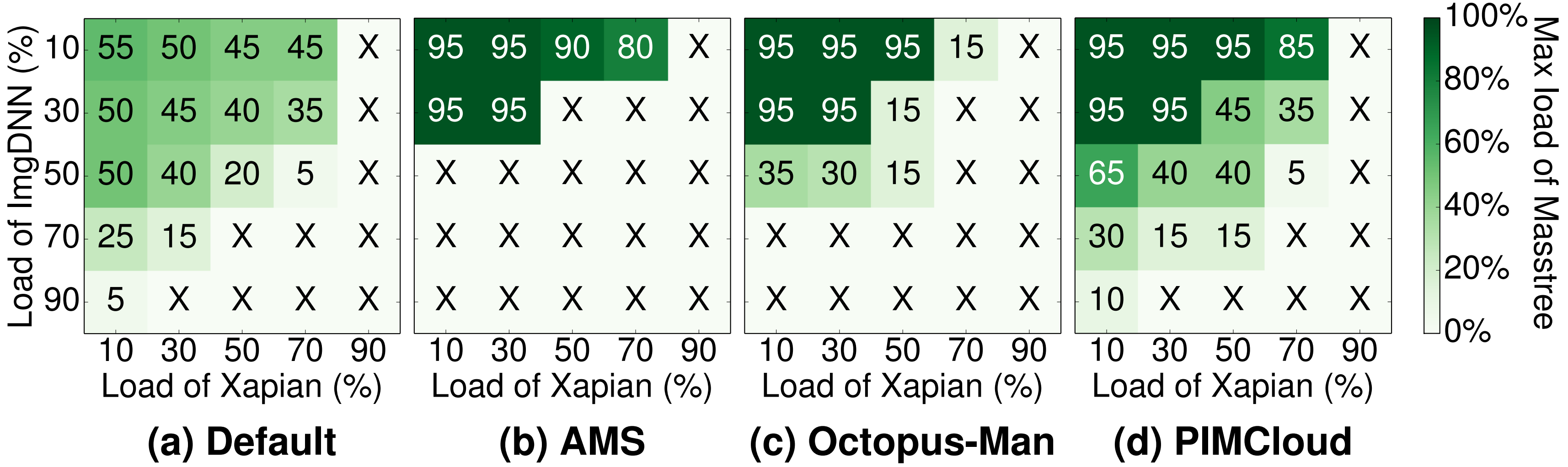
Evaluation - Constant Loads

Colocation of Xapian, ImgDNN and Masstree at various input loads.



(a) Default (b) AMS (c) Octopus-Man (d) PIMCloud

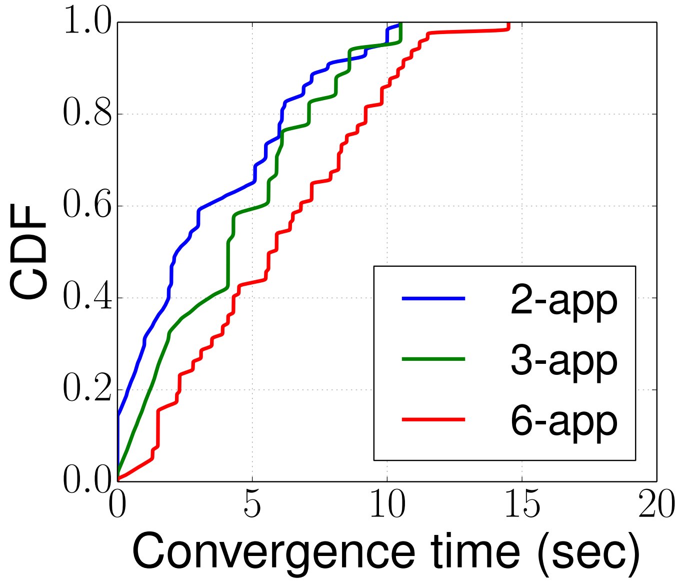
Colocation of Xapian, ImgDNN and Masstree at various input loads.

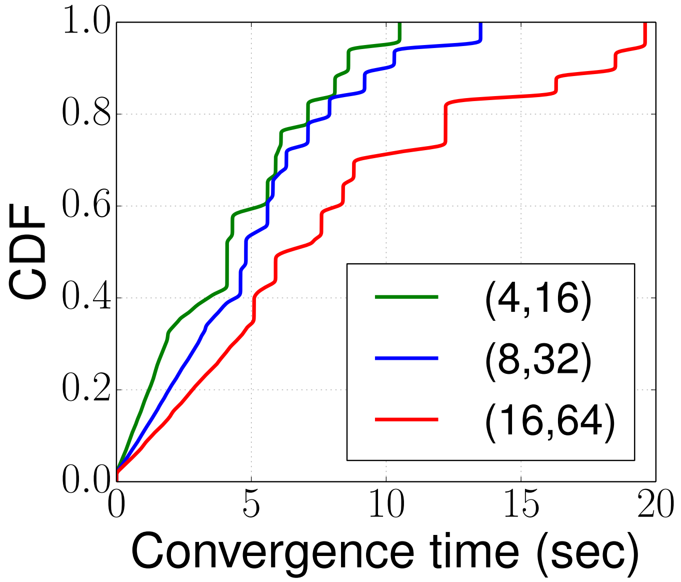


PIMCloud outperforms all the baselines

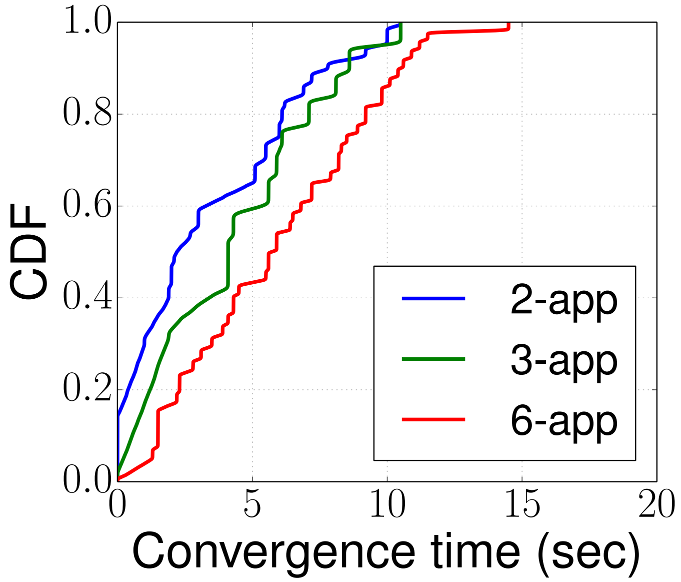
- **Core isolation**: better than *Default*
- **Adjust core count based on load**: better than AMS
- **Preference-aware**: better than Octopus-Man
- **Manage data placement:** better than all the baselines

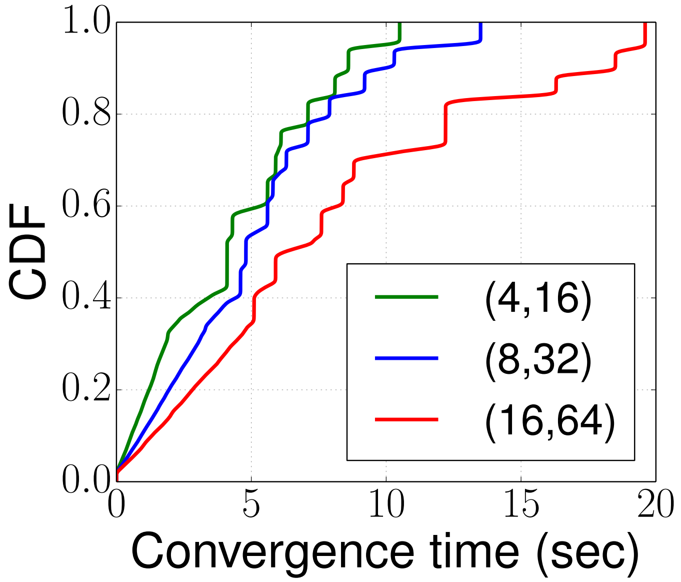
Evaluation - Scalability





Evaluation - Scalability





- **Convergence time doesn't increase exponentially with more apps / larger systems**
- **Worst case convergence time is 20s**
  - Convergence time is less than 10s more than 70% of the time.

More Evaluation Results in the Paper

- Colocation of 2 LC apps at various input load
- Colocation of 2 LC apps and a BE job at various input load
- Decomposition of PIMCloud
- Dynamic load

Motivation

- Increasingly important LC applications that will be colocated on the same node
- Increasingly heterogeneous cloud platforms
  - Especially PIM that brings heterogeneity to computation and memory at the same time

PIMCloud

- Characterization of LC applications on PIM
  - More than half of the LC applications perform better on PIM than on CPU
- A QoS-aware and PIM-aware resource manager for LC applications in PIM-enabled systems
  - Leverages preference to reduce the allocation space down to a homogeneous setting
  - Manipulates only hot pages at runtime

Cornell University
Computer Systems Laboratory

PIMCloud:
QoS-Aware Resource Management of Latency-Critical Applications in Clouds with Processing-in-Memory

Thanks! Q & A

Offline discussion: chenshuang0804@gmail.com