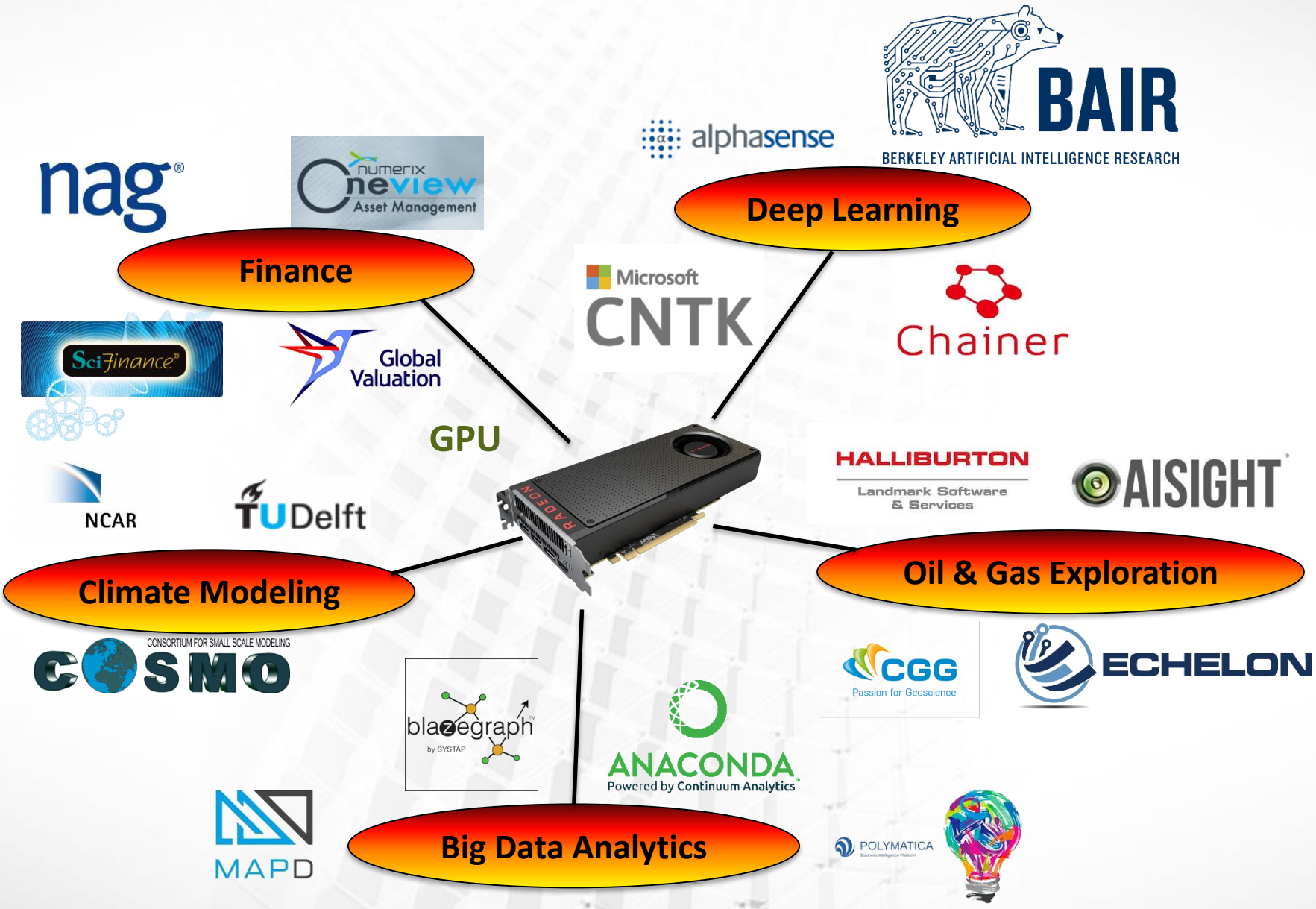
Neighborhood-aware address translation for irregular GPU applications

Seunghee Shin[1], Michael LeBeane[2], Yan Solihin[3], Arkaprava Basu[4]

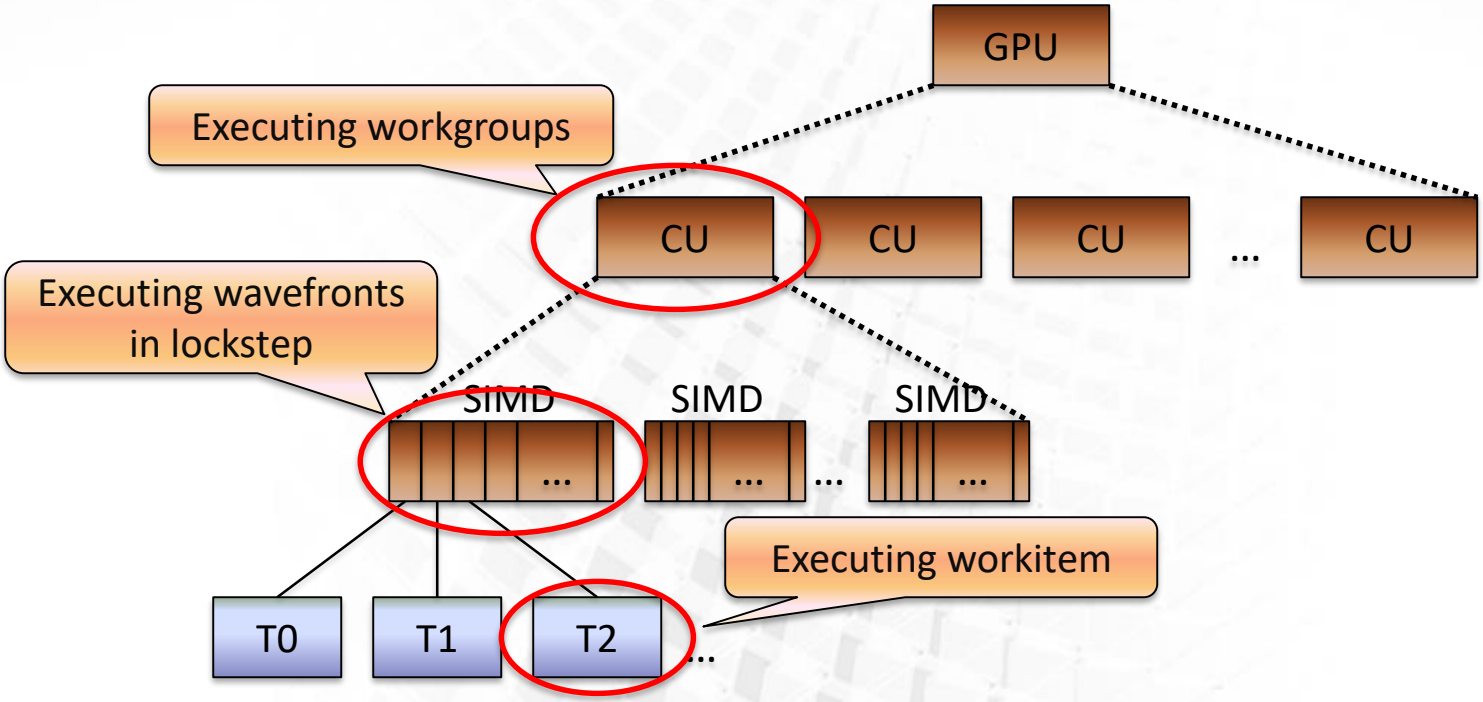
[1] Binghamton University, [2] AMD Research,
[3] University of Central Florida, [4] Indian Institute of Science

The 51st IEEE/ACM International Symposium on Microarchitecture (MICRO 51)

GPU APPLICATIONS

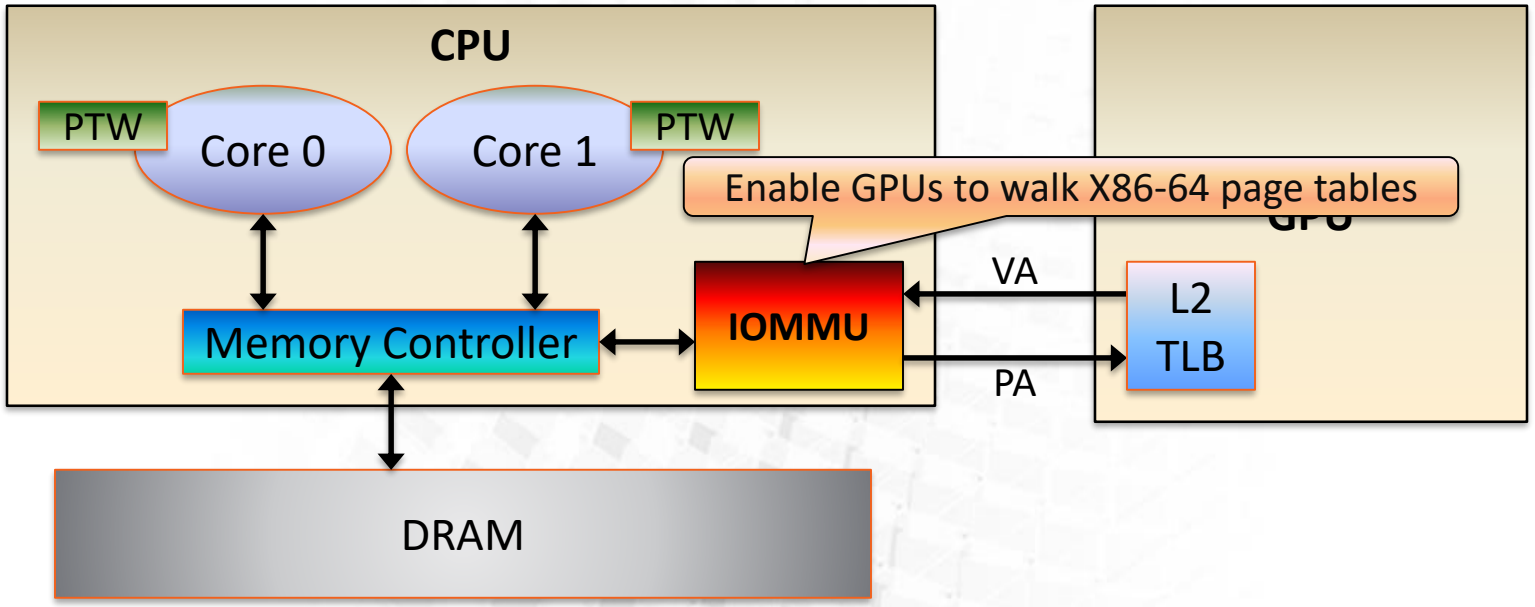


GPU ARCHITECTURE



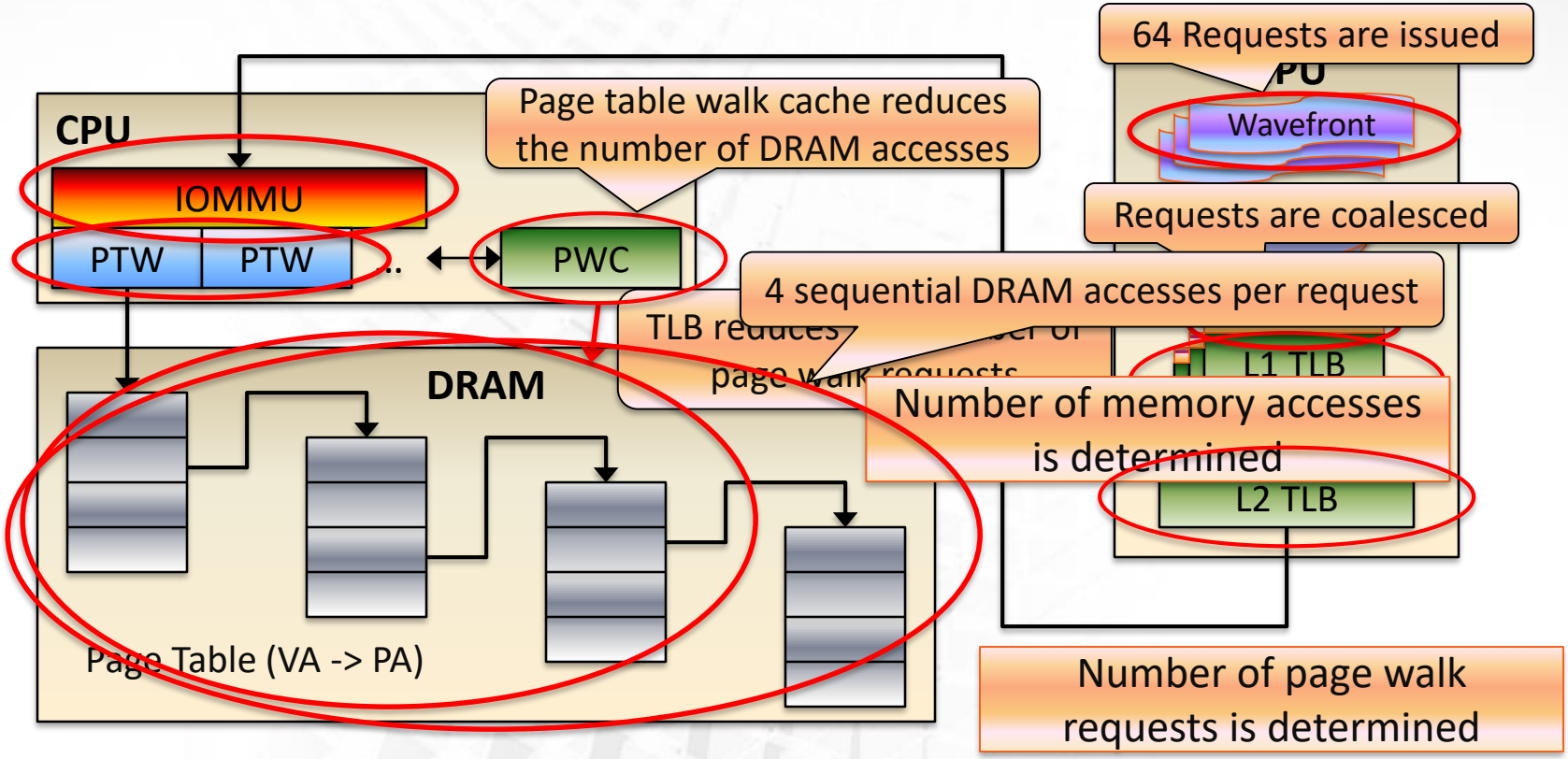
- A Compute Unit (CU) is the basic computational block (8-64 CUs in a GPU)
- Each CU has multiple Single-Instruction-Multiple-Data (SIMD) units
- A SIMD unit has multiple lanes of execution (32 or 64)
- Each lane executes one workitem (thread)

HETEROGENEOUS SYSTEM ARCHITECTURE (HSA)



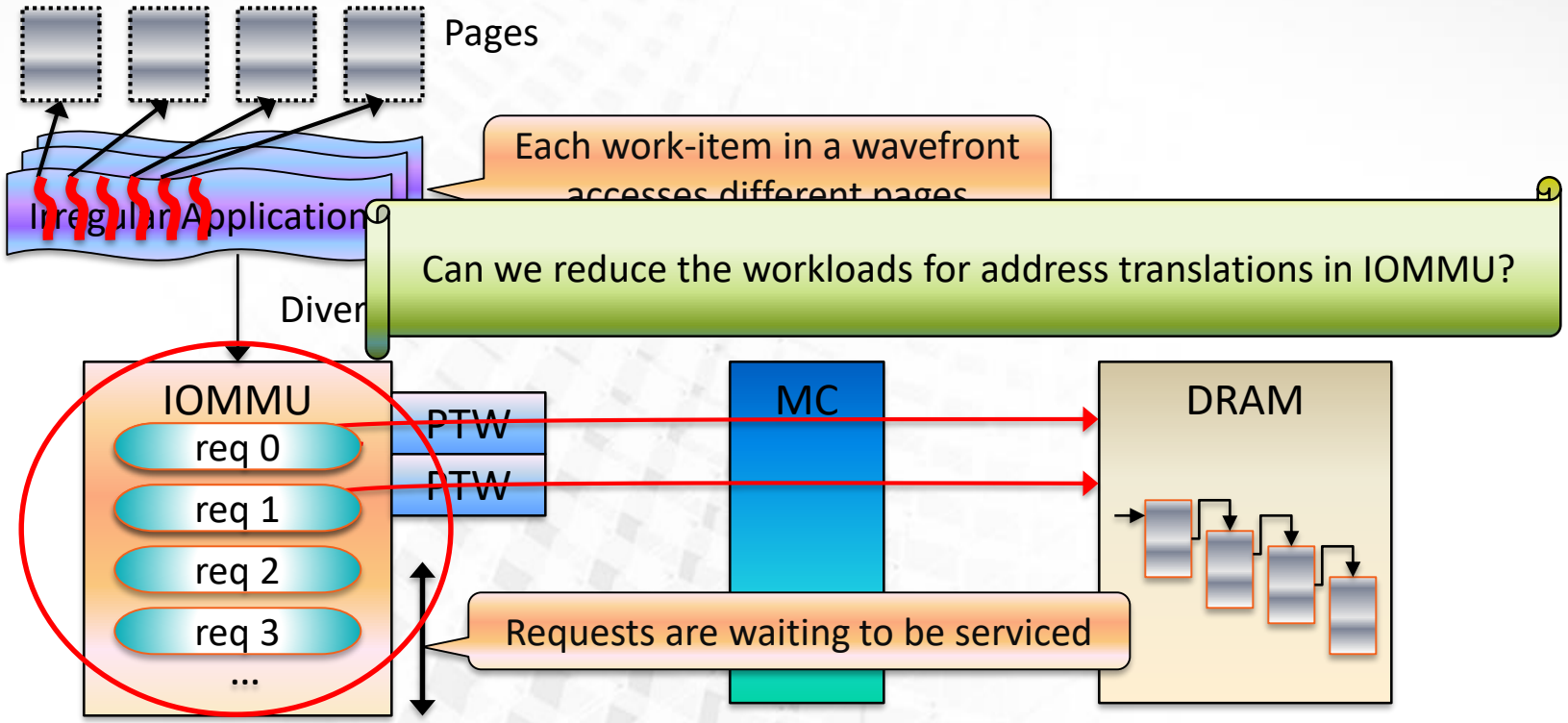
- Integration of CPUs and GPUs with shared memory
  - Provides unified virtual address space
  - Reduces CPU/GPU communication latency
  - Removes the CPU/GPU programmability barrier
  - Requires GPUs to walk the X86 page tables

VIRTUAL ADDRESS TRANSLATION WITH IOMMU



- Number of memory accesses for translations per wavefront
  - IOMMU has multiple page table walkers (e.g., 8–16)
  - Number of memory accesses per SIMD instruction varies
  - One SIMD instruction can create up to 256 memory accesses

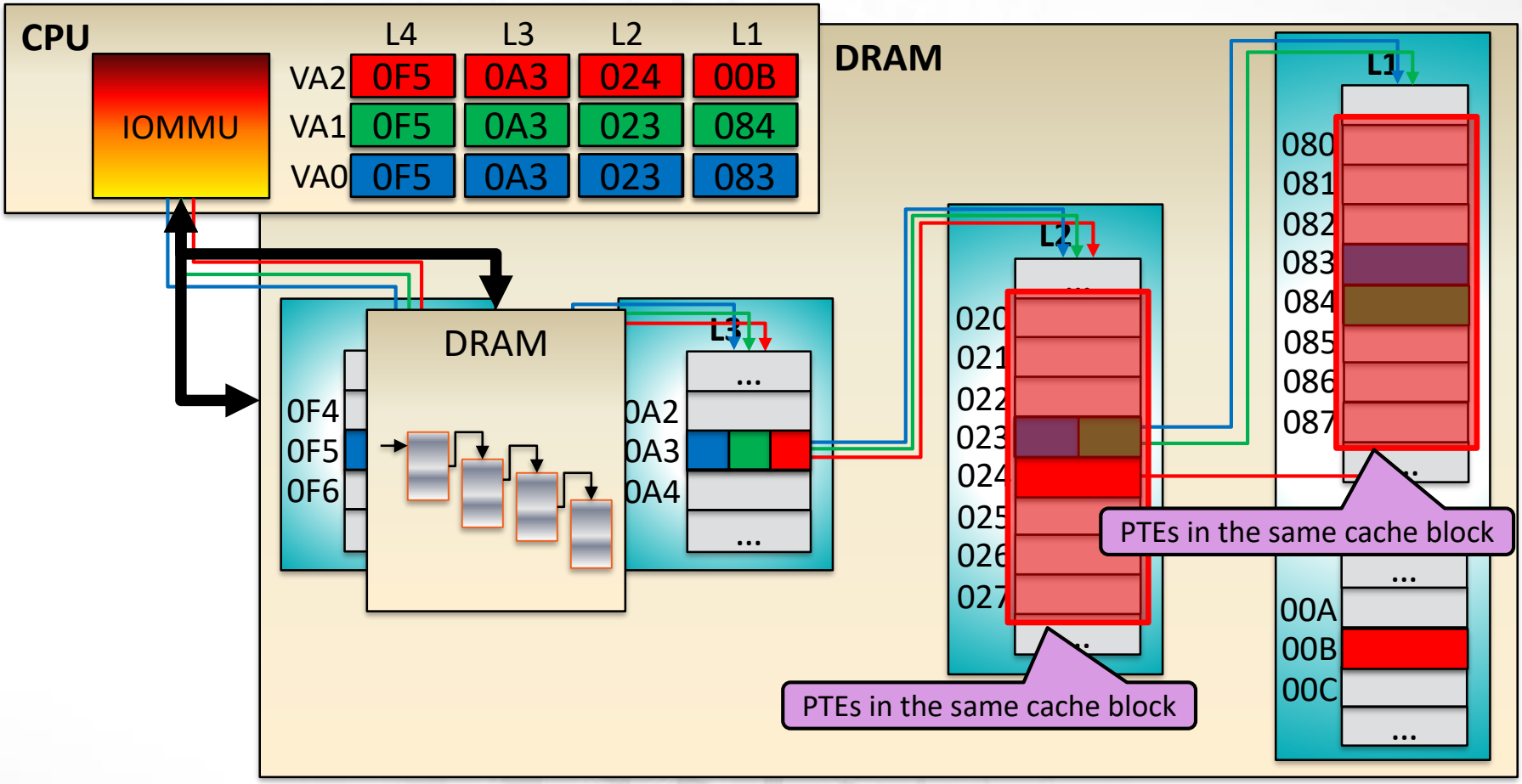
OVERHEADS BY IRREGULAR APPLICATIONS



- IOMMU bottleneck
  - Irregular application memory accesses have low spatial locality
  - Requests queue up in the IOMMU buffer to be serviced
  - Divergent accesses can slowdown irregular applications by up to 4x*

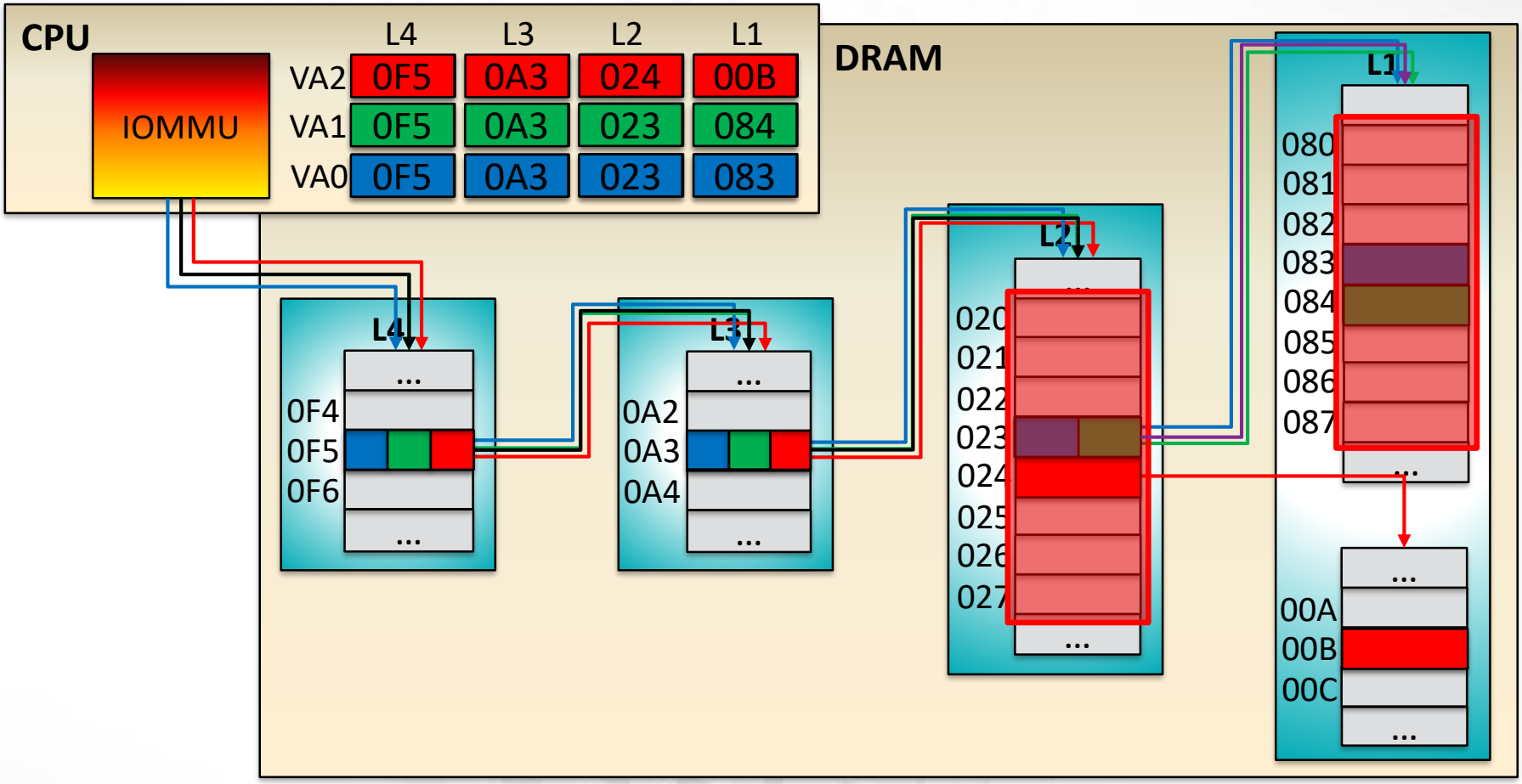
** Observations and Opportunities in Architecting Shared Virtual Memory for Heterogeneous Systems, ISPASS16, Jan Vesely*, Arkaprava Basu, Mark Oskin, Gabriel H. Loh, Abhishek Bhattacharjee

PAGE TABLE NEIGHBORHOOD



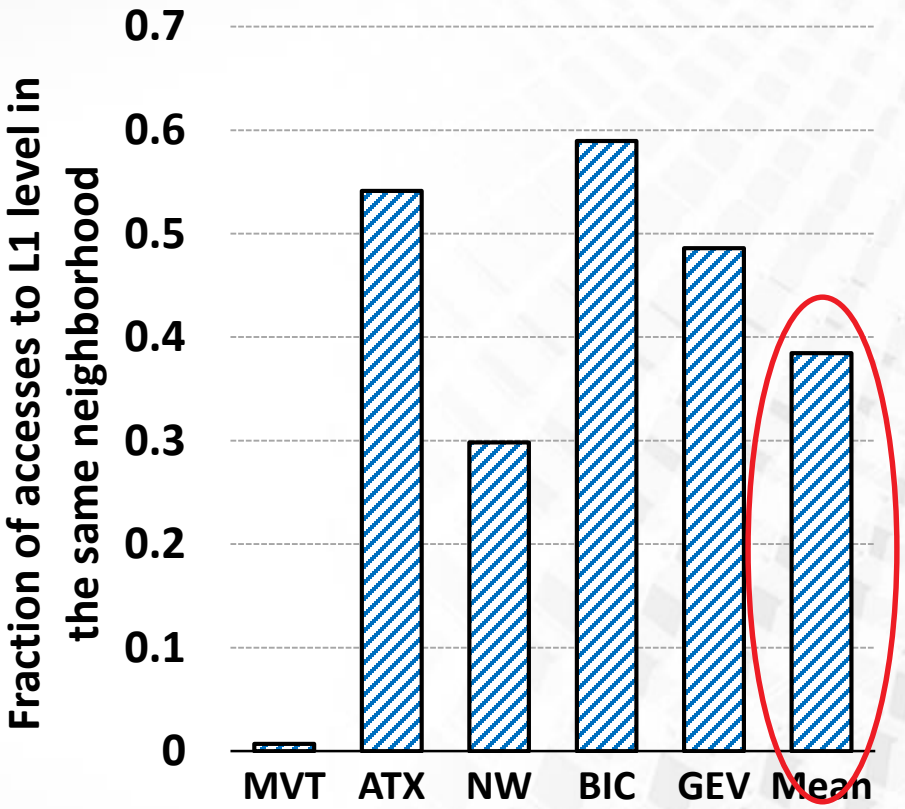
- Neighborhood: PTEs fall in the same cache line (8B per PTE)
- Page table walker access memory at the granularity of a cache line (64B: 8 PTEs)
- Neighborhood covers 32 KB in L1, 16MB in L2, 8GB in L3, 4TB in L4

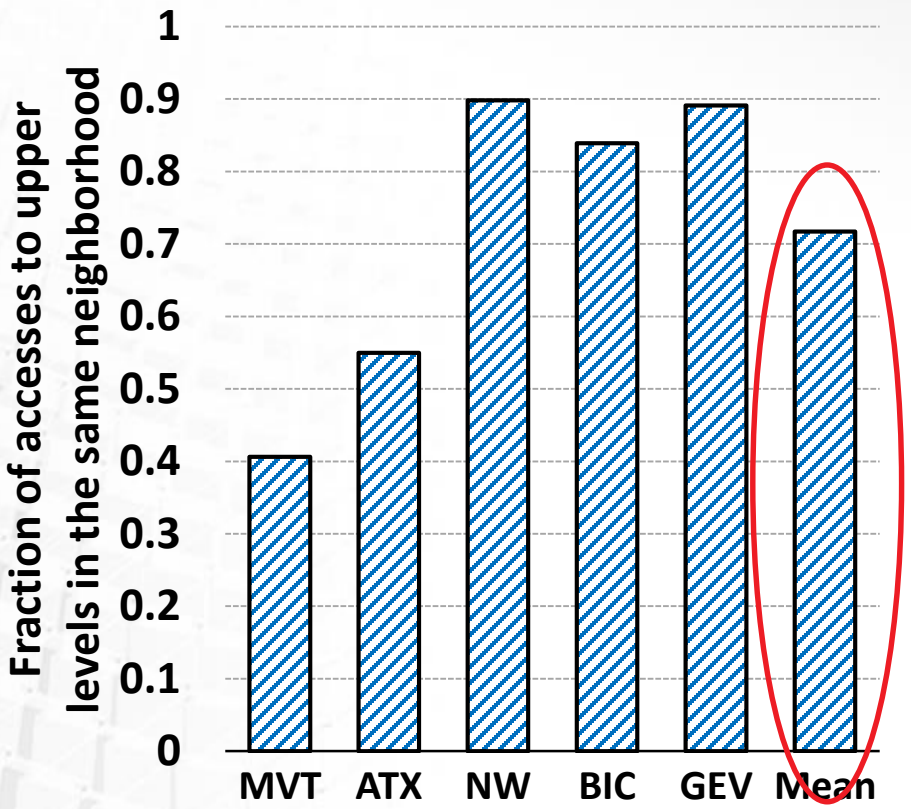
NEIGHBORHOOD-AWARE COALESCING



- VA0 and VA1 can fully coalesce and only need 4 in-memory page table accesses
- VA0 and VA2 can coalesce up to L2; an additional page table access after VA0 for VA2
- Three requests complete in 5 in-memory page table accesses with coalescing

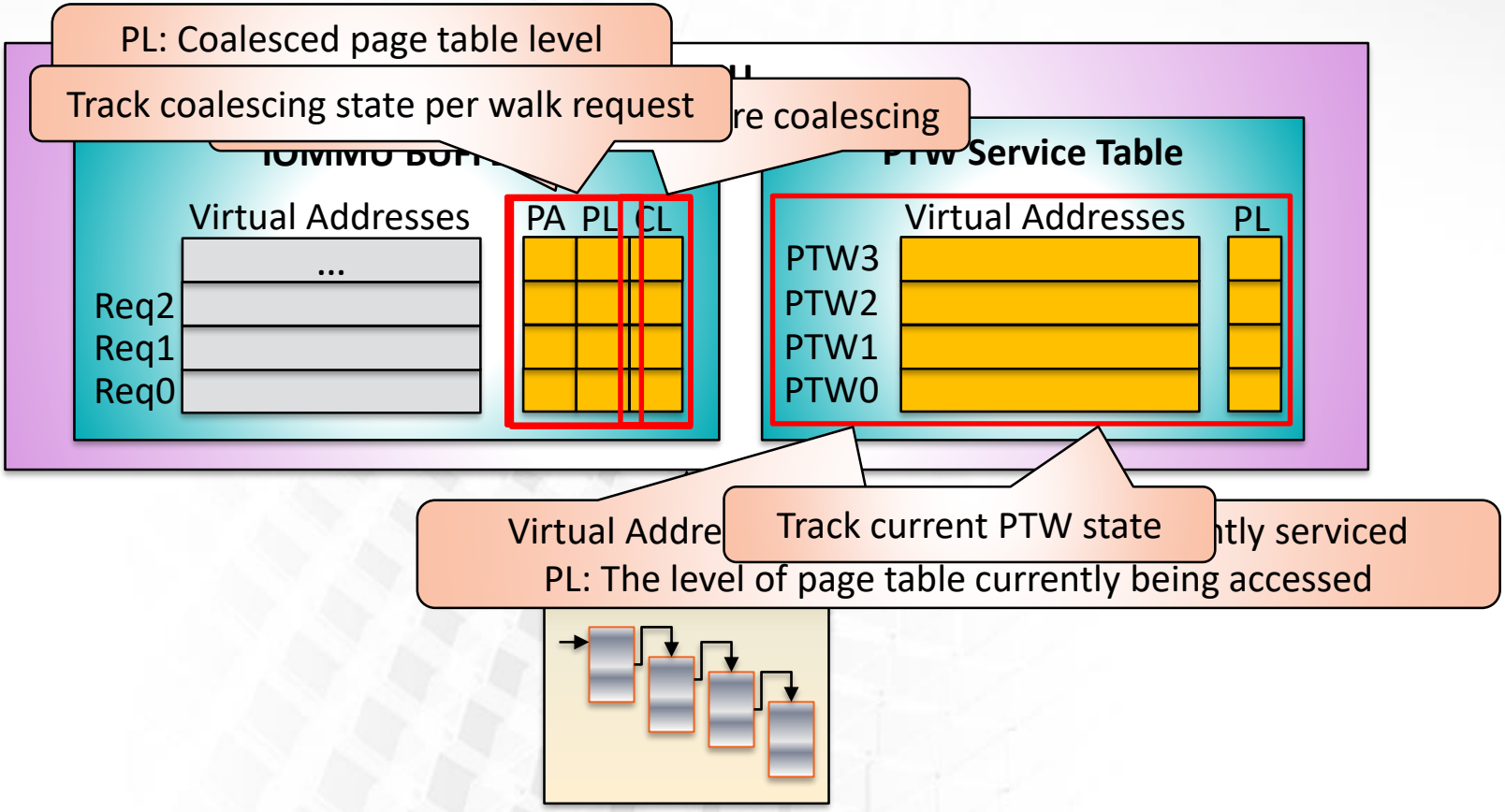
NEIGHBORHOOD-AWARE COALESCING OPPORTUNITY





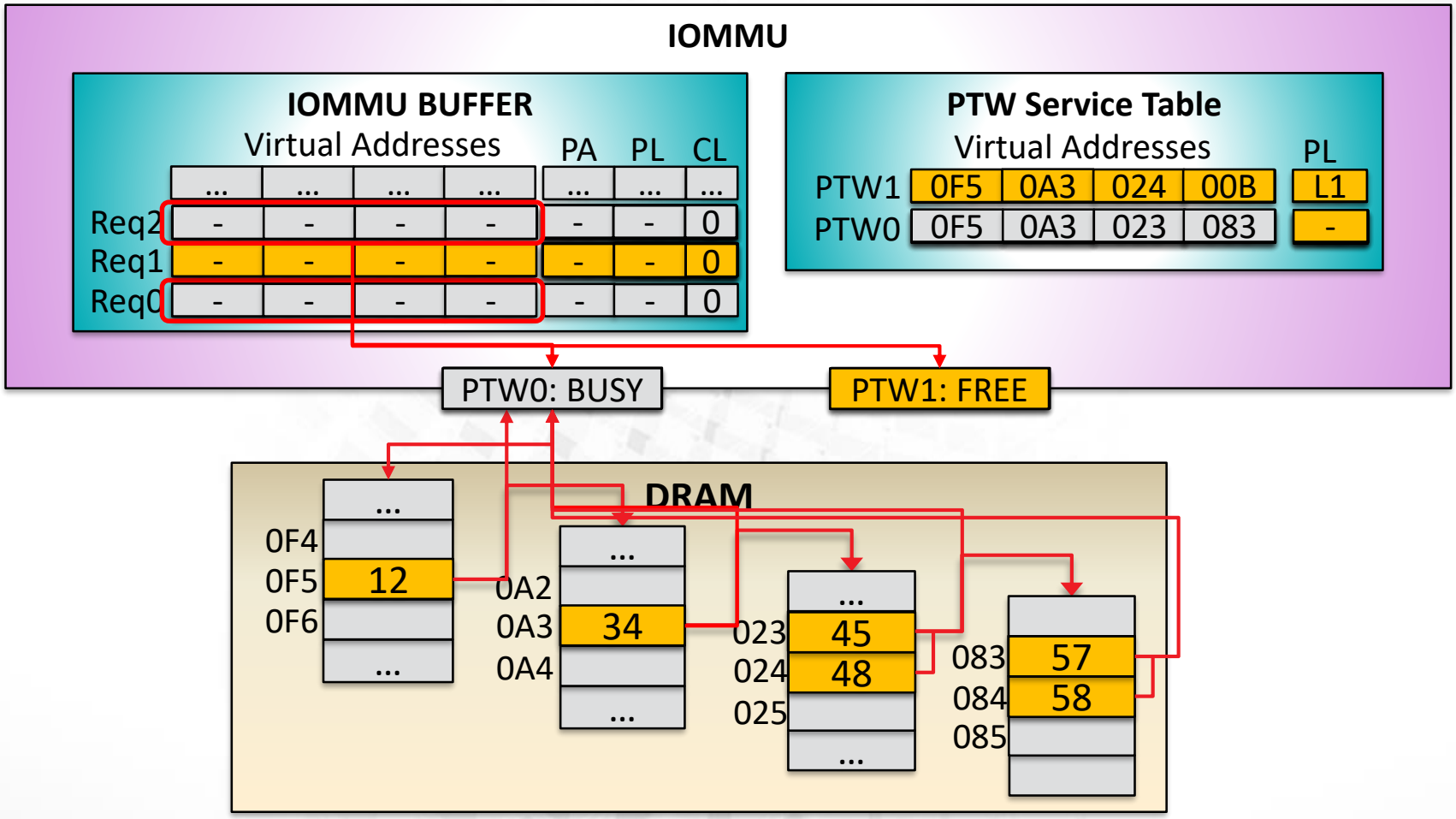
- 40% of coalescing opportunities in leaf level page table after PTW caches
- 70% of further coalescing opportunities in upper level page tables

ADDITIONAL IOMMU HARDWARE STATE FOR COALESCING



- Requests with coalescing opportunities are held in IOMMU buffer until the coalescing
- IOMMU needs to track the coalescing state per walk request
- IOMMU needs to track current PTW state
- Only around 1.5KB extra buffer space

NEIGHBORHOOD-AWARE COALESCING IMPLEMENTATION



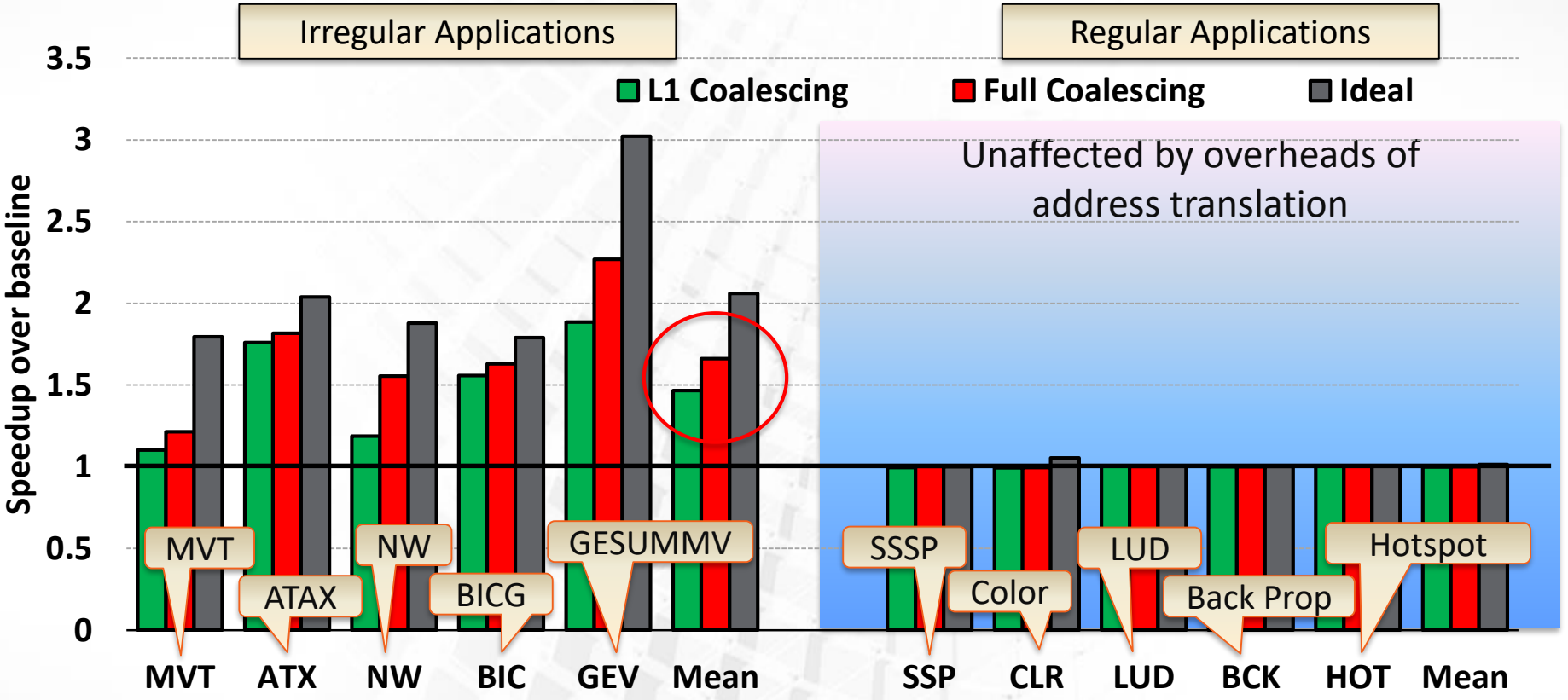
- Req0 and Req1 coalesce at L1: no further page walk needs
- Req0 and Req2 coalesce at L2: Req2 needs to walk from L1 after the coalescing

METHODOLOGY

System Configuration	
GPU	2GHz, 8 CUs, 4 SIMD per CU 16 SIMD width, 64 threads per wavefront
L1D Cache	32KB, 16-way, 64B block
L2 Cache	4MB, 16-way, 64B block
L1 TLB	32 entries, private, Fully-associative
L2 TLB	512 entries, shared, 16-way set associative
IOMMU	256 buffer entries, 8 page table walkers 32/256 entries for IOMMU L1/L2 TLB, FCFS scheduling of page walks
DRAM	DDR3-1600 (800MHz), 2 channel 16 banks per rank, 2 ranks per channel

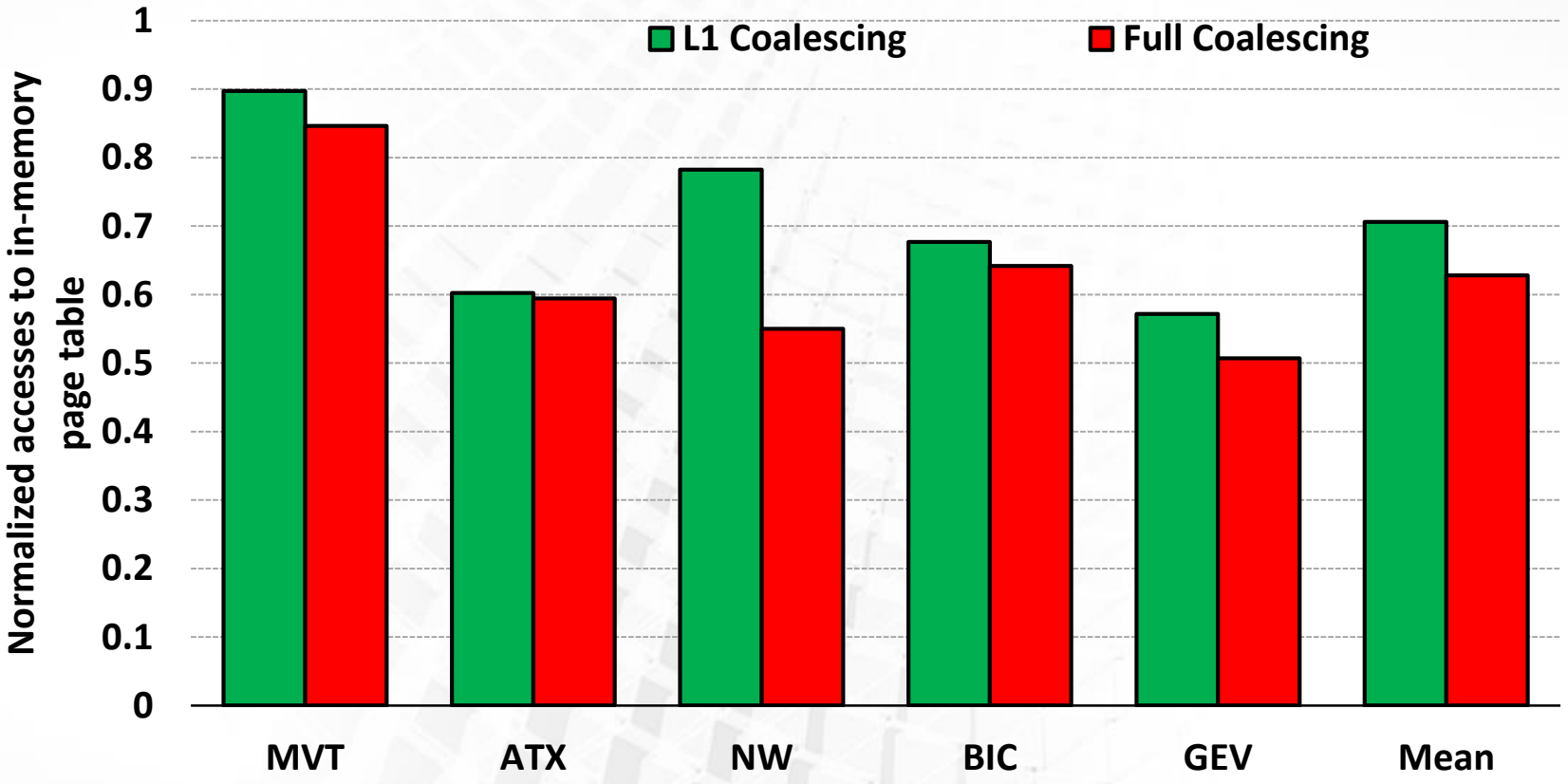
- GEM5 simulator modeling HSA (CPU + integrated GPU)
- Implemented detailed address translation model including request coalescer, TLB hierarchy, page table walk caches, and IOMMU

EVALUATION (1) - SPEEDUP



- L1 coalescing speeds up 1.5x and full coalescing speeds up 1.7x
- No performance impact on regular applications

EVALUATION (2) – NUMBER OF PAGE TABLE ACCESSES



- L1 coalescing reduces 29% accesses to the in-memory page table
- Full coalescing further reduces 37% accesses to the in-memory page table
- Reduced number of page table accesses lead to drop page walk latency by 47%

CONCLUSIONS

- Observations
  - Irregular GPU applications can bottleneck on address translations
  - Different SIMD instructions may incur vastly different numbers of memory accesses
- Neighborhood-aware page table walk coalescing
  - Utilize the observation that multiple PTEs are fetched per page table access
  - Coalesce page walk requests when they belong to the same neighborhood
  - Extend to apply the coalescing to the upper level page tables
- Results
  - Neighborhood-aware coalescing reduces the number of in-memory page table walks by 37%, leading to drop the page walk latency by 47%
  - Overall, neighborhood-aware coalescing shows 1.7x speedup for irregular GPU applications


AMD

DISCLAIMER & ATTRIBUTION

The information presented in this document is for informational purposes only and may contain technical inaccuracies, omissions and typographical errors.

The information contained herein is subject to change and may be rendered inaccurate for many reasons, including but not limited to product and roadmap changes, component and motherboard version changes, new model and/or product releases, product differences between differing manufacturers, software changes, BIOS flashes, firmware upgrades, or the like. AMD assumes no obligation to update or otherwise correct or revise this information. However, AMD reserves the right to revise this information and to make changes from time to time to the content hereof without obligation of AMD to notify any person of such revisions or changes.

AMD MAKES NO REPRESENTATIONS OR WARRANTIES WITH RESPECT TO THE CONTENTS HEREOF AND ASSUMES NO RESPONSIBILITY FOR ANY INACCURACIES, ERRORS OR OMISSIONS THAT MAY APPEAR IN THIS INFORMATION.

AMD SPECIFICALLY DISCLAIMS ANY IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR ANY PARTICULAR PURPOSE. IN NO EVENT WILL AMD BE LIABLE TO ANY PERSON FOR ANY DIRECT, INDIRECT, SPECIAL OR OTHER CONSEQUENTIAL DAMAGES ARISING FROM THE USE OF ANY INFORMATION CONTAINED HEREIN, EVEN IF AMD IS EXPRESSLY ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

ATTRIBUTION

EVALUATION (3) – PAGE WALK LATENCY