NOMAD: Enabling Non-blocking OS-managed DRAM Cache via Tag-Data Decoupling

Josh Rong, Siyuan Li

Y. Kim, H. Kim and W. J. Song, "NOMAD: Enabling Non-blocking OS-managed DRAM Cache via Tag-Data Decoupling," 2023 IEEE International Symposium on High-Performance Computer Architecture (HPCA), Montreal, QC, Canada, 2023, pp. 193-205, doi: 10.1109/HPCA56546.2023.10071016.

Authors

Youngin Kim

PhD Candidate @ Yonsei University

System Architect @ MangoBoost

Hyeonjin Kim

PhD student @ Yonsei University

William J. Song

Associate professor @ Yonsei University

PhD @ Georgia Institute of Technology

Background

Problem:

Demands for larger memory bandwidth and capacity.

Solution:

Heterogeneous memory systems combining high-bandwidth on-package DRAM (**DRAM Cache**) and large-capacity off-package memory.

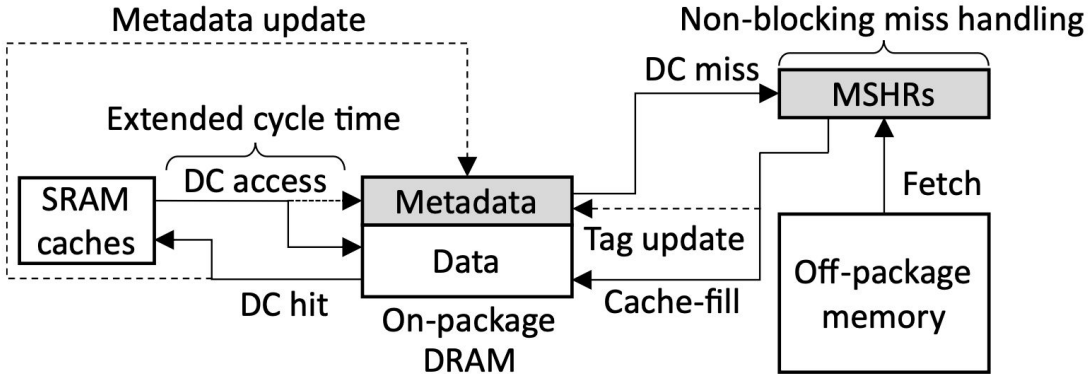
Existing DRAM Cache Designs (HW-based)

1. Tag lookup in On-package DRAM.

If Tag Hit, return data.

If Tag Miss,

2. OS resumes. MSHR service the request.
3. Cache Fill. (Eviction, Data placement, Tag)



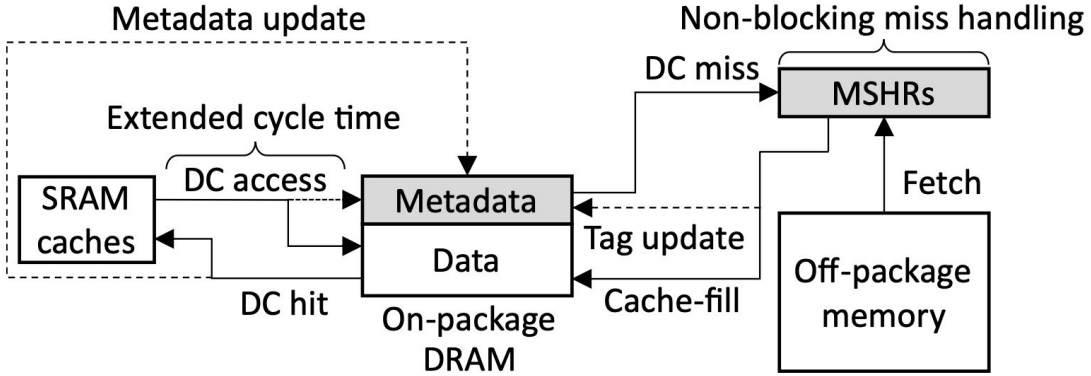
Existing DRAM Cache Designs (HW-based)

Pro:

Non-blocking Miss Handling: Uses Miss Status Holding Registers (MSHRs) to handle multiple misses simultaneously.

Con:

High Metadata Management Overhead: Metadata (tags, dirty bits, etc.) is stored in on-package DRAM.



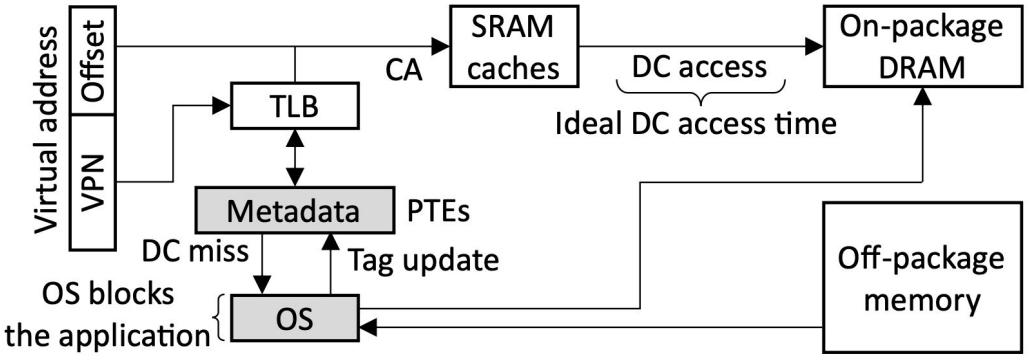
Existing DRAM Cache Designs (OS-managed)

1. Check TLB for cached page.

If TLB Hit, access DRAM Cache and return data.

If TLB Miss,

2. Miss handler allocates a new cache frame.
3. OS wait for Tag management and Data management.



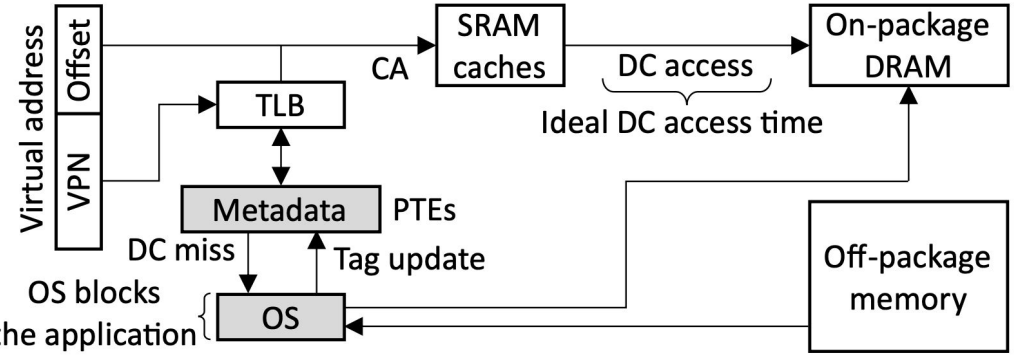
Existing DRAM Cache Designs (OS-managed)

Pro:

Ideal DRAM Cache Access Time: Utilizes PTE stored in TLB, avoiding the need to transfer metadata between DRAM and the cache.

Con:

Blocking Miss Handling: When a DRAM cache miss occurs, the OS halts the application thread until the cache fill (thousands of cycles) is completed.



Benchmark Workload

Excess:

RMHB > Bandwidth

Tight:

RMHB = Bandwidth

Loose:

RMHB = ½ Bandwidth

Few:

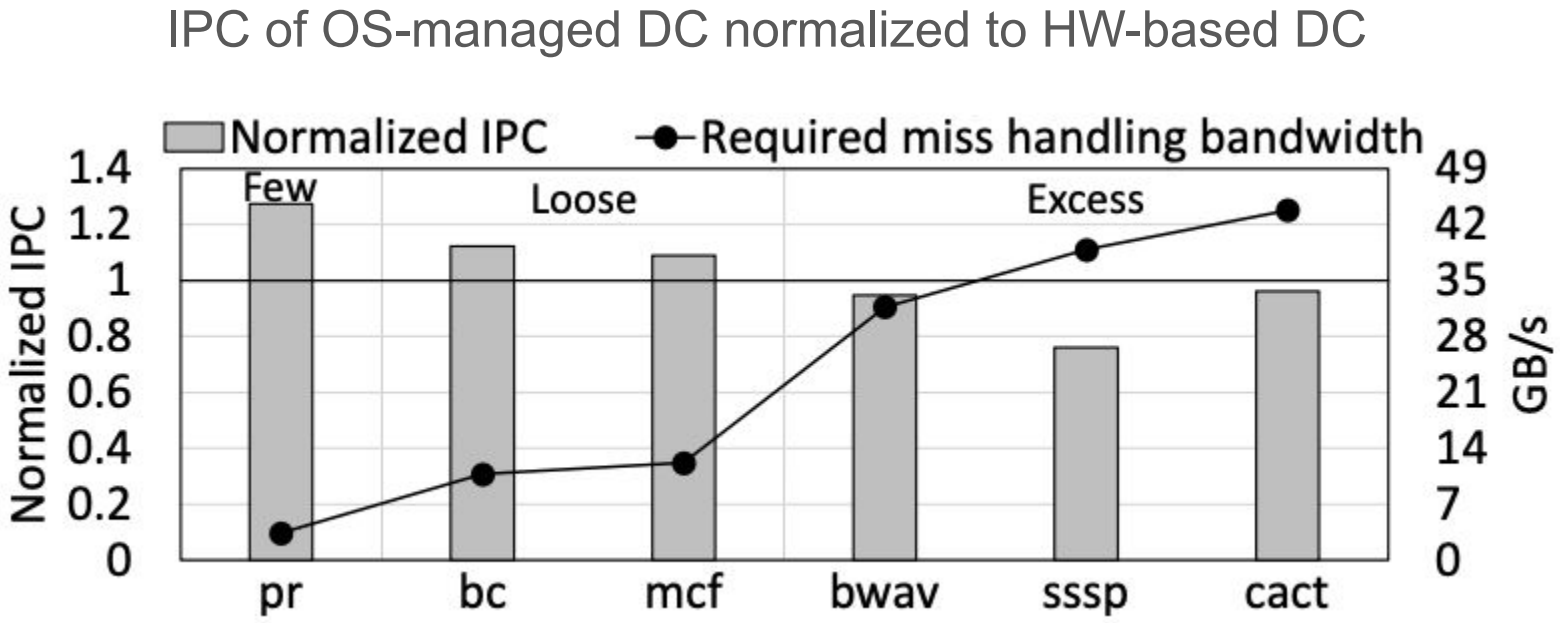
RMHB << Bandwidth

Required Miss Handling Bandwidth

LLC Misses Per Microsecond

Class	Abbr.	Workloads	RMHB (GB/s)	LLC MPMS	Memory footprint (GB)
Excess	cact	cactusADM [3]	43.8	486.6	11.9
	sssp	sssp [3]	38.8	511.1	2.3
	bwav	bwaves [16]	31.7	588.1	4.5
Tight	les	leslie3d [16]	26.5	532.8	7.5
	libq	libquantum [16]	25.1	210.6	4.0
	gems	gemsFDTD [16]	24.8	269.2	6.3
	bfs	bfs [3]	23.1	298.5	2.4
Loose	cc	cc [3]	13.5	183.1	2.3
	lbm	lbm [16]	12.4	270.5	3.2
	mcf	mcf [16]	12.2	472.0	2.8
	bc	bc [3]	10.8	533.7	1.3
Few	ast	astar [16]	6.9	72.1	1.0
	pr	pr [3]	3.4	691.9	4.8
	sop	soplex [16]	1.7	310.2	1.2
	tc	tc [3]	1.66	226.3	2.3

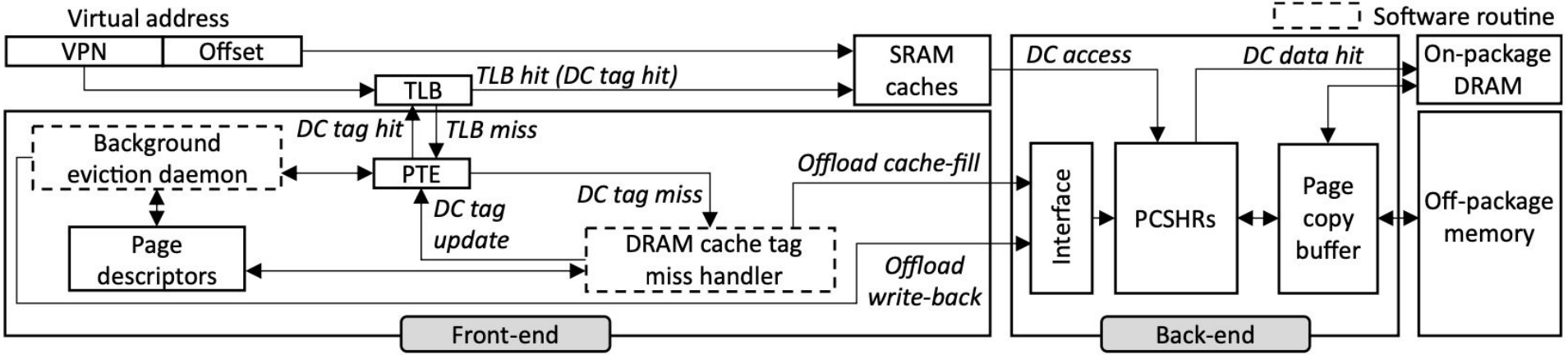
Motivation


IPC of OS-managed DC normalized to HW-based DC
Normalized IPC
Required miss handling bandwidth
Few
Loose
Excess
Normalized IPC
1.4
1.2
1
0.8
0.6
0.4
0.2
0
GB/s
49
42
35
28
21
14
7
0
pr
bc
mcf
bwav
sssp
cact

NOMAD Overall Structure

Decoupled Tag-Data Management

- Front-end: update tag
- Back-end: cache-fill



Front-end OS Routine (Page Descriptor & PTE)

On a cache miss:

1. OS checks PTE , and find that the page is cacheable but not cached.
2. Miss handler allocate a cache frame, the CFN replaces the PFN in the PTE.
3. CPD is updated to store the original PFN.
4. The application resumes. Cache-fill in back-end.

C: cached

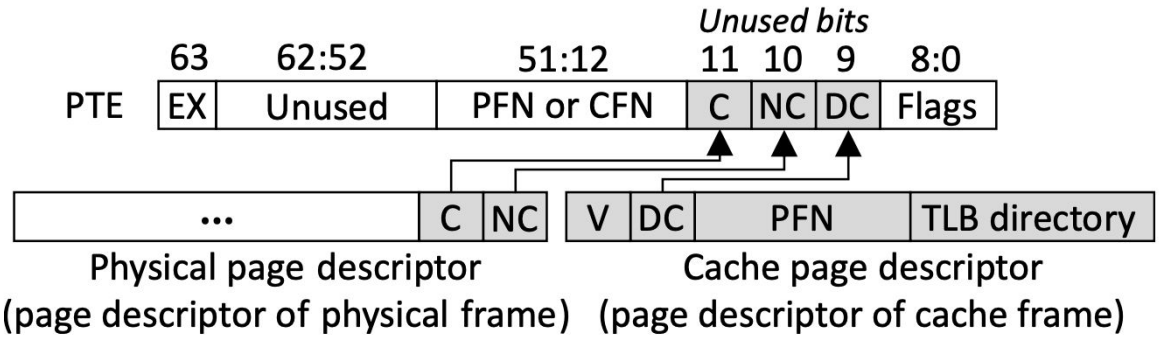
NC: non-cacheable

V: valid

DC: dirty-cache

On a cache eviction (in batch):

1. Skip frame referenced by TLB.
2. If dirty, schedule writeback.
3. Restore PTE to point to the original PFN in CPD.
4. Invalidate evicted frames.



Front-end OS Routine (Cache Frame Management)

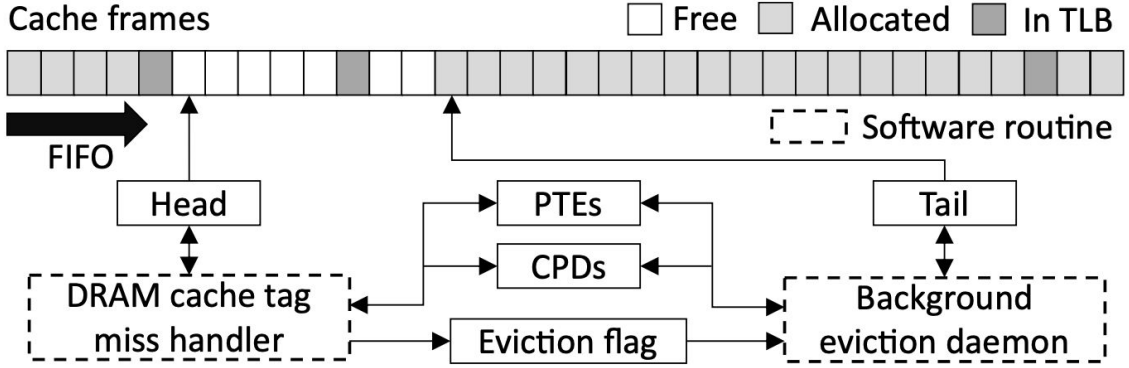
FIFO Replacement Policy

Simplicity.

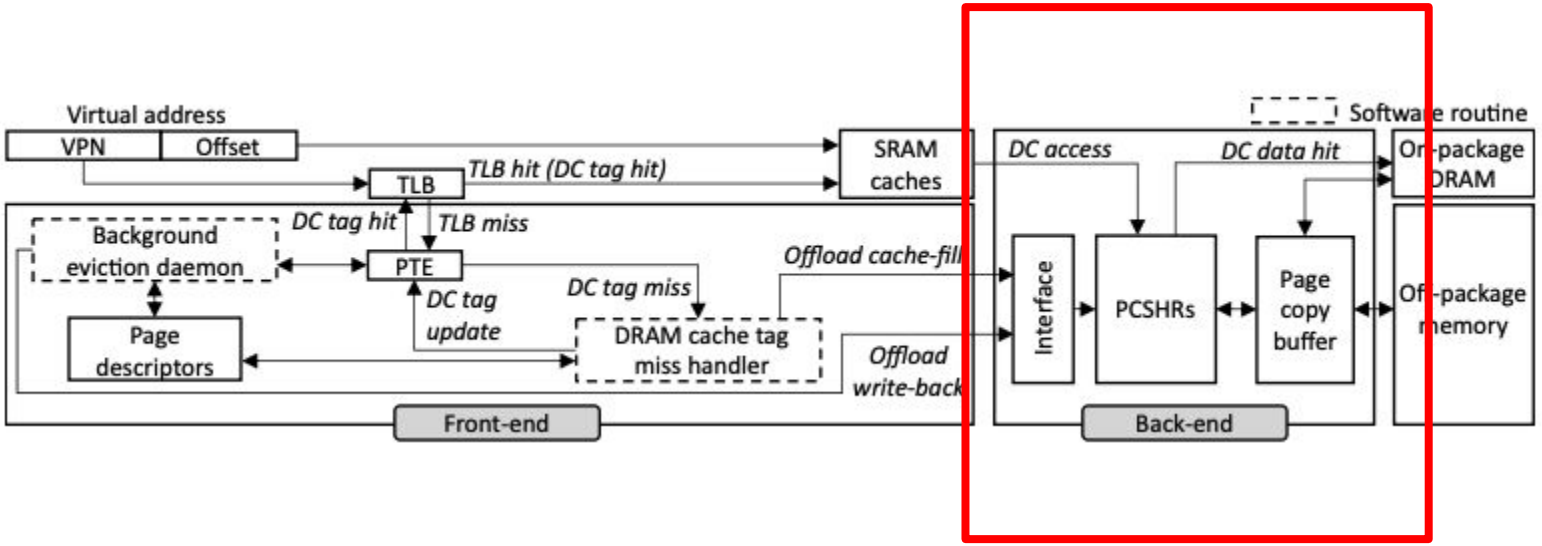
Fully-associative nature of OS-managed DRAM Cache.

23% less DC misses than 16-way set-associative HW-based DC with LRU.

Prior work of HW-based DC can only scale up to 4-way set-associative.



Back-end Hardware



Back-end Hardware

Interface:

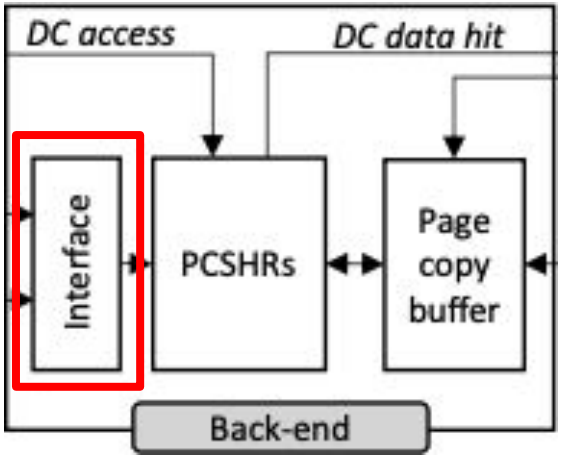
Memory-mapped device register

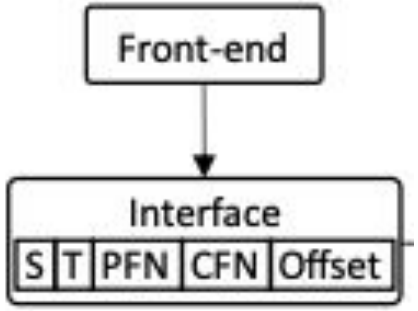
S: whether the interface is busy or not

T: Specifies whether the command is cache-fill or write back execution

PFN, CFN, Offset: address information

Front-end request -> allocate PCSHR -> idle





Back-end Hardware

PCSHR: Page Copy Status/Information Holding Register

Handles multiple page copy commands

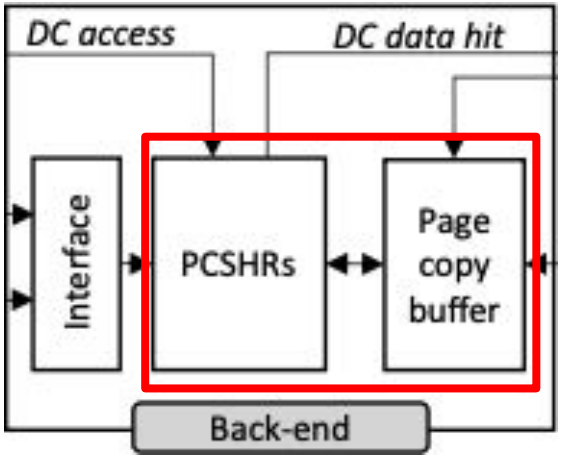
V: Valid command or not

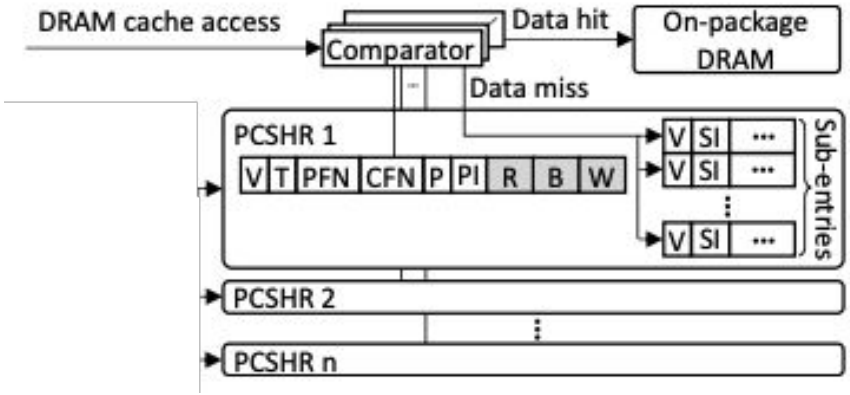
T: Command type (obtained upon allocation)

R: Read-issued

B: In-buffer

W: Partial-write





Back-end Hardware

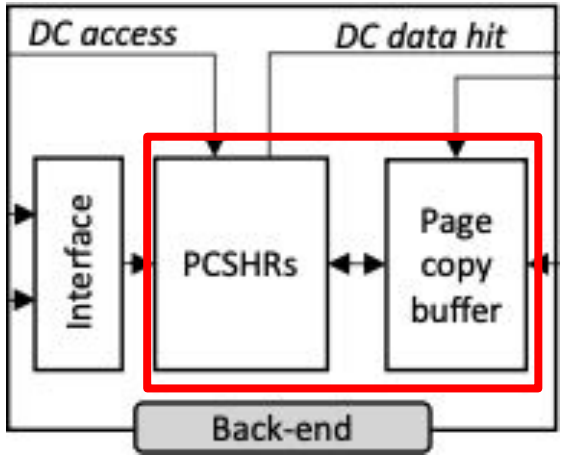
Sub-blocks of a page is fetched sequentially

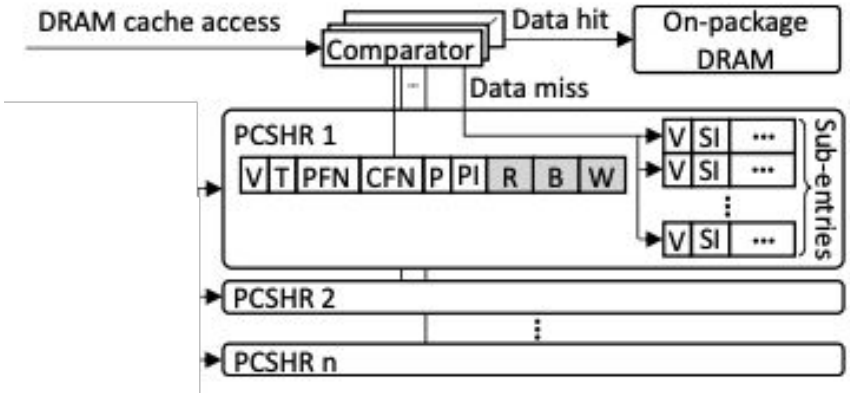
P: Prioritize sub-block

PI: Index of prioritized sub-block

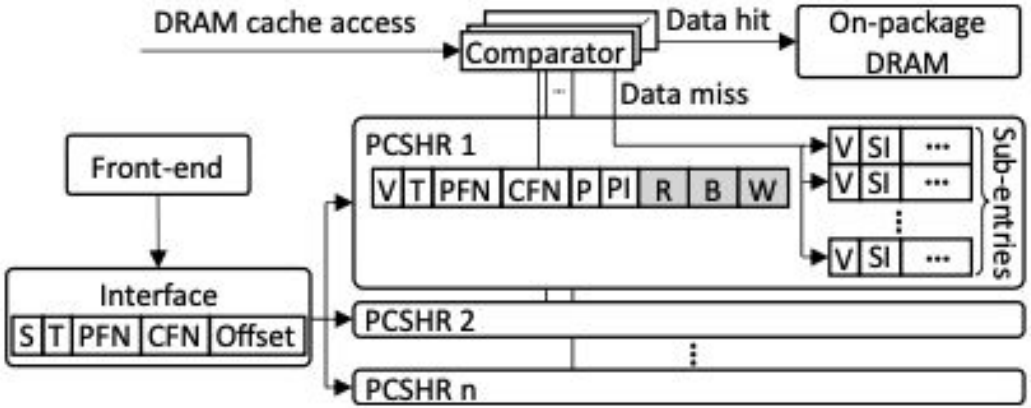
V (sub-entries): valid bit

SI (sub-entries): sub-block index





DRAM Cache Access

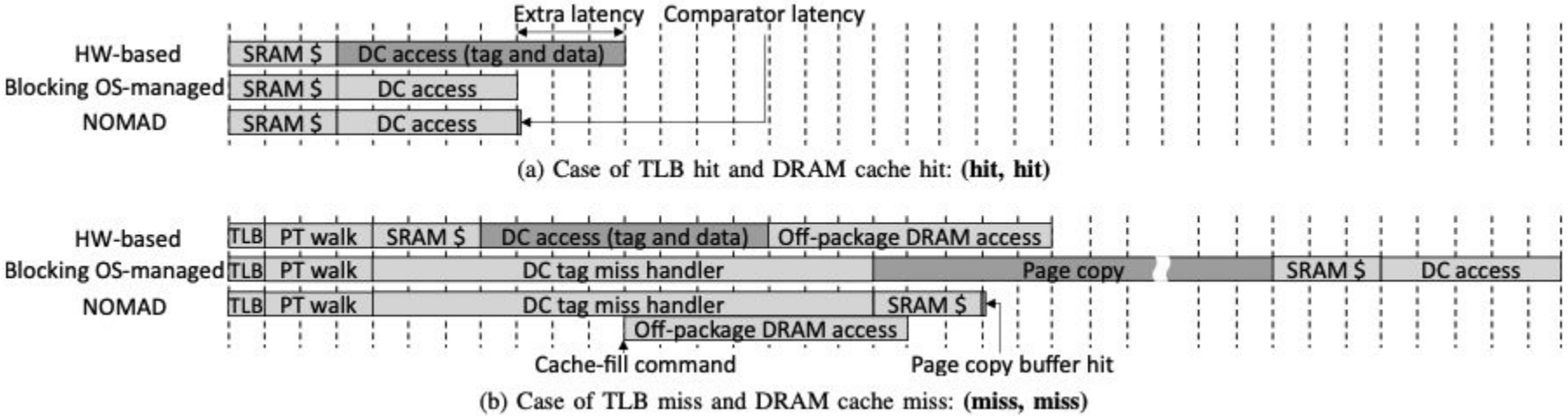


TLB hit - > CFNs of PCSHR compared

No match: Page is available in DRAM cache (data hit)

Match: pending request from DRAM (data miss)

DRAM Cache Access



Evaluation of NOMAD

Baseline: no on-package DRAM

TiD: HW-based Cache with tag management

TDC: State-of-the-art OS-managed design

Ideal: OS-managed DRAM cache with no latency penalties for miss handling

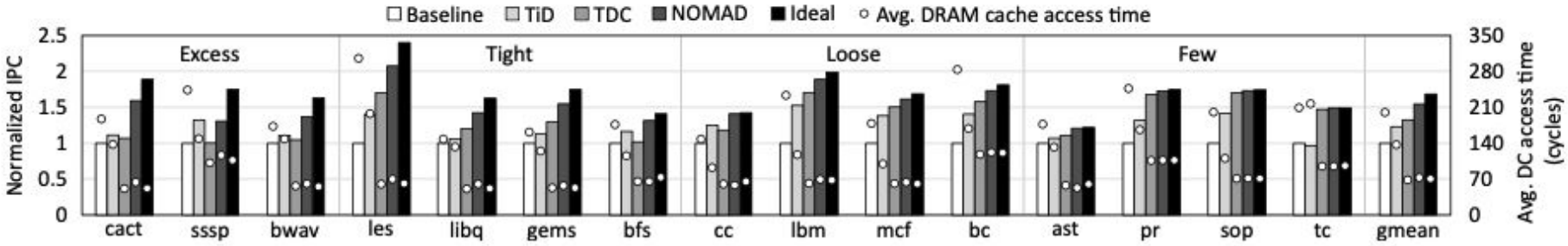


Fig. 9: IPC normalized to the baseline and average DRAM cache access time in CPU cycles.

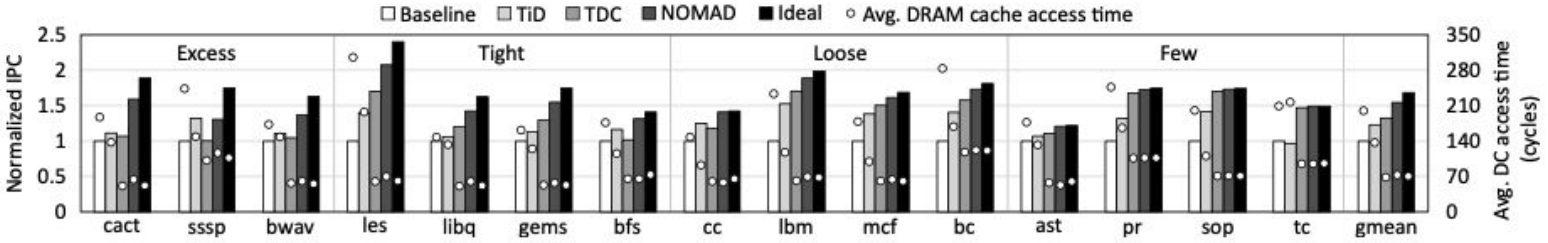


Fig. 9: IPC normalized to the baseline and average DRAM cache access time in CPU cycles.

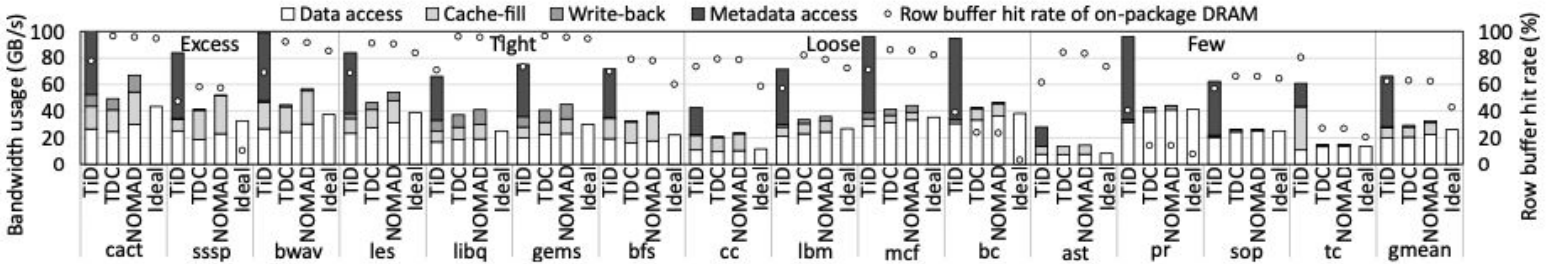


Fig. 10: Breakdown of on-package DRAM bandwidth usage and row buffer hit rates of the on-package DRAM.

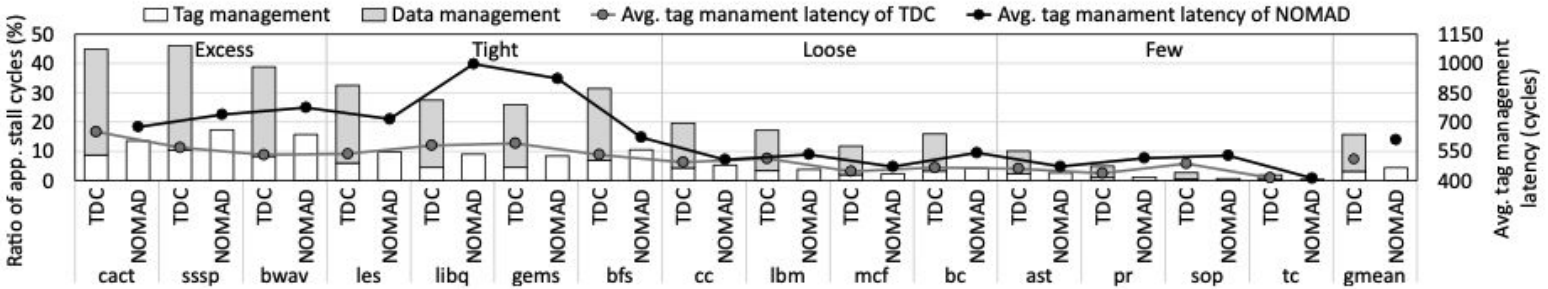


Fig. 11: Breakdown of application stall cycle ratios and the average tag management latency of OS-managed schemes.

Evaluation of NOMAD

	TDC (OS-managed)	TiD (HW-managed)	NOMAD
Excess	Stall 43% of runtime	Substantial DRAM bandwidth	- Reduce stall via non-block cache - Reduced metadata transfer
Tight	Stall 29%. No performance gain for workload with less spatial locality	Suffer from non-ideal memory access time	Best. - Tolerate tag misses - Near-ideal access time
Loose	Stall level of 15%	Suffer from high LLC MPMS benchmark	Less than 5% stall Near ideal performance (few tag misses and less contention)
Few	Negligible stall time	Large bandwidth consumption bottlenecks performance	Near ideal performance

Further Optimization: Area

- Number of PCSHRs

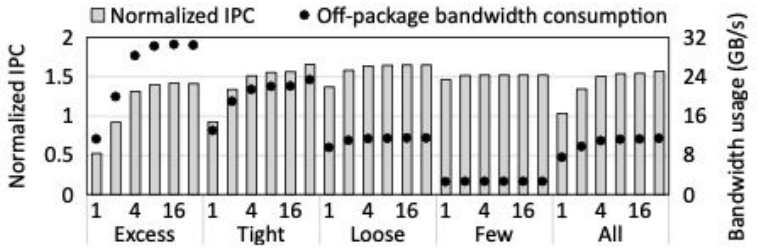


Fig. 12: Per-class average IPC relative to the baseline and the average off-package memory bandwidth consumption of NOMAD with respect to the number of PCSHRs.

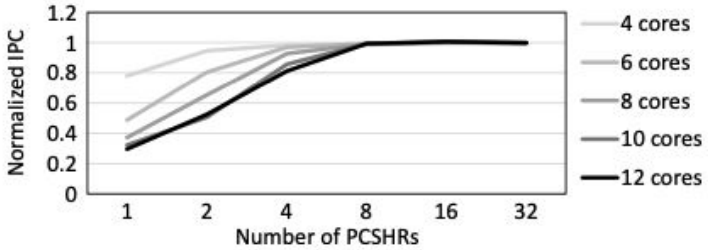


Fig. 13: Average IPC of Excess-class benchmarks with different number of PCSHRs for increasing CPU core count.

Further Optimization: Area

- Size of page copy buffers

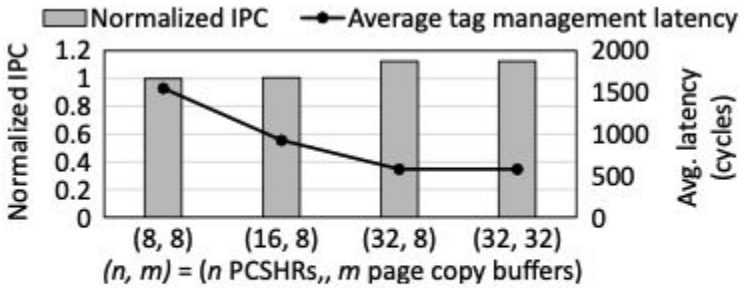
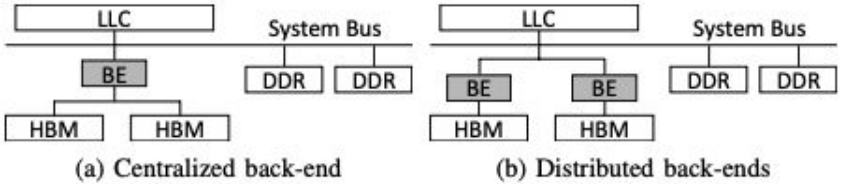


Fig. 15: Normalized IPC and the tag management latency of burst-RMHB workloads (i.e., libq, gems) with different configurations of (n PCSHRs, m page copy buffers).

Further Optimization: Centralized vs Distributed



(a) Centralized back-end (b) Distributed back-ends

Fig. 8: Centralized and distributed back-end (BE) structures that keep the generality of HBMs (on-package DRAM) and DDRs (off-package memory).

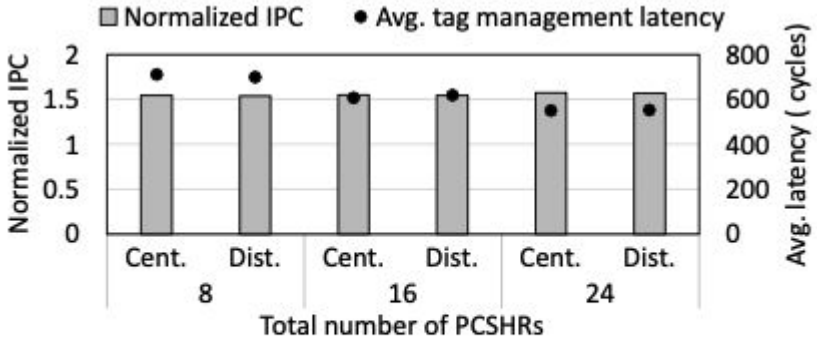


Fig. 16: Average IPC normalized to the baseline and the average tag management latency of centralized and distributed back-end designs with different numbers of PCSHRs.

Previous observation: Imbalanced accesses would cause frequent lockups and require greater hardware resources.

NOMAD: Sequential cache frame allocation guarantees uniformly distributed commands

Thanks