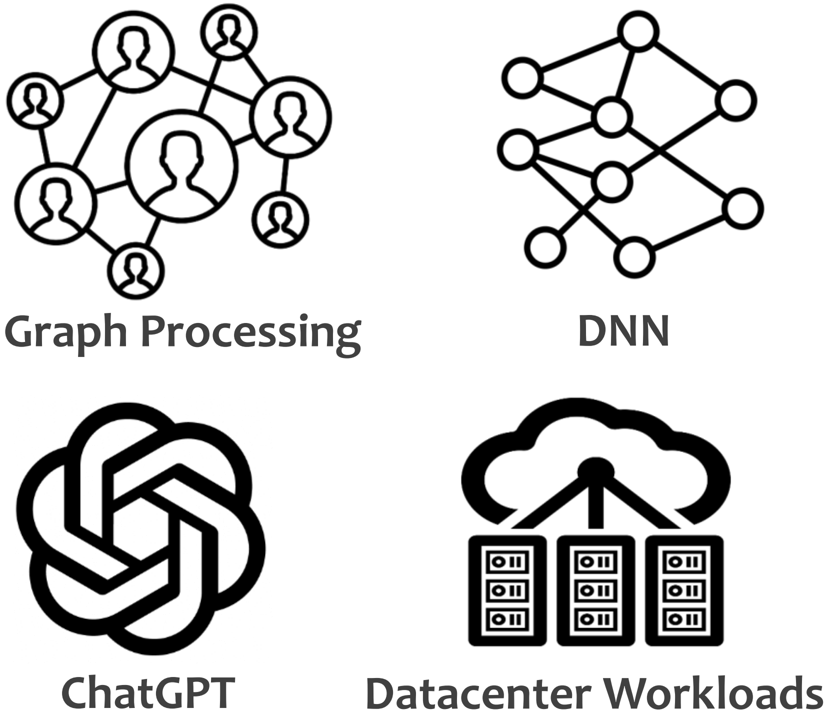
GRIT: Enhancing Multi-GPU Performance with Fine-Grained Dynamic Page Placement

Yueqi Wang*[1], ***Bingyao Li*[1]***, Aamer Jaleel[2], Jun Yang[1], Xulong Tang[1]

[1]University of Pittsburgh, [2]NVIDIA

Multi-GPU is Popular

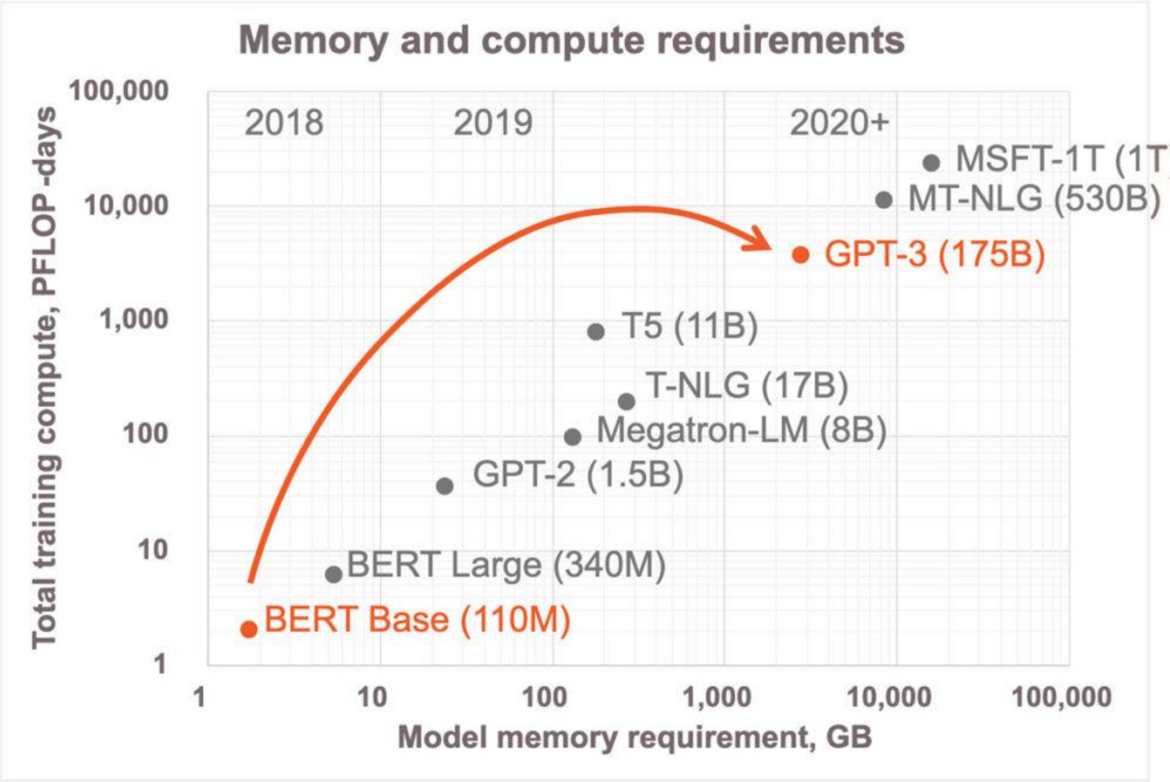


Graph Processing

DNN

ChatGPT

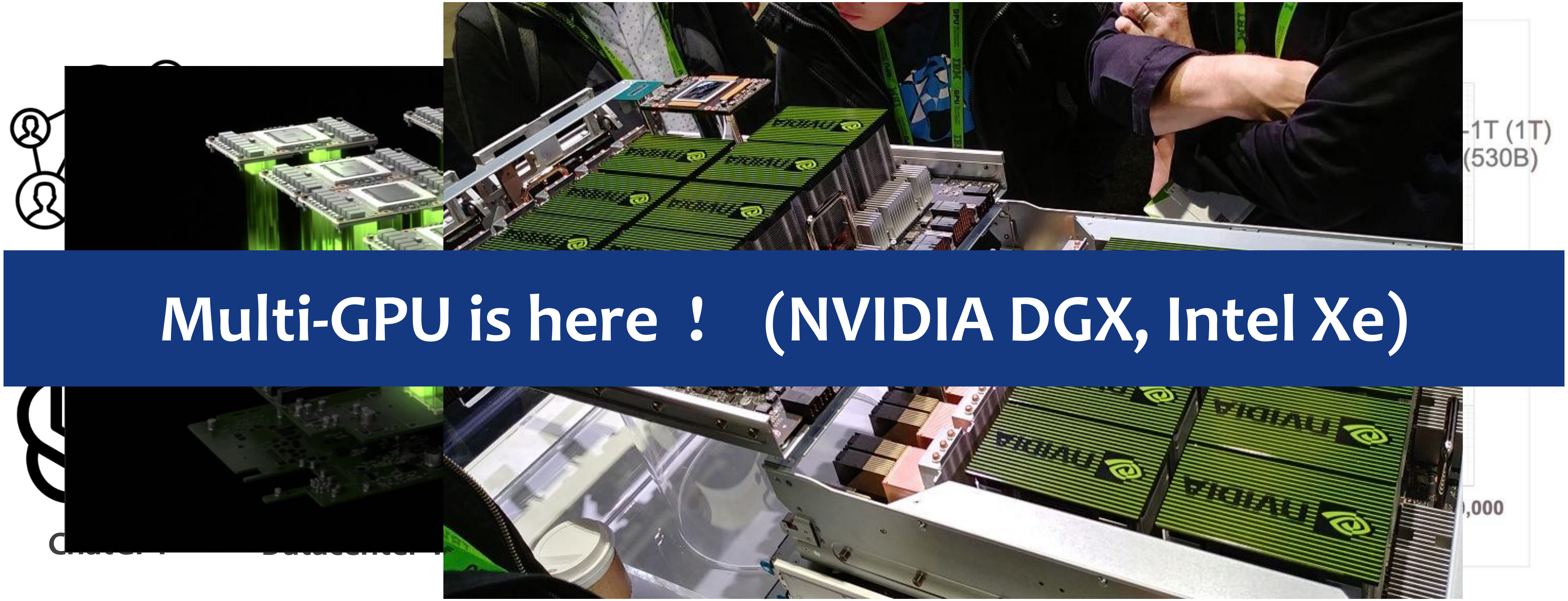
Datacenter Workloads



Ever-growing application complexity and input dataset sizes.

[1] https://www.cerebras.net/blog/harnessing-the-power-of-sparsity-for-large-gpt-ai-models

Multi-GPU is Popular



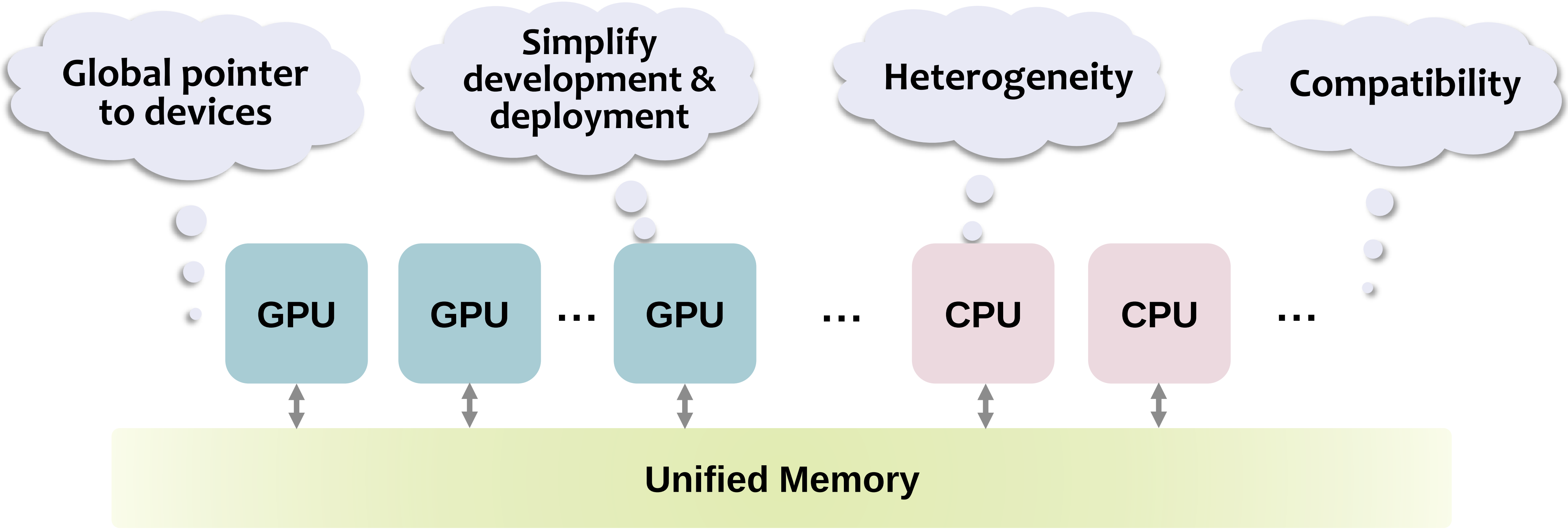
Multi-GPU is here ！ (NVIDIA DGX, Intel Xe)

Ever-growing application complexity and input dataset sizes.

[1] https://www.cerebras.net/blog/harnessing-the-power-of-sparsity-for-large-gpt-ai-models

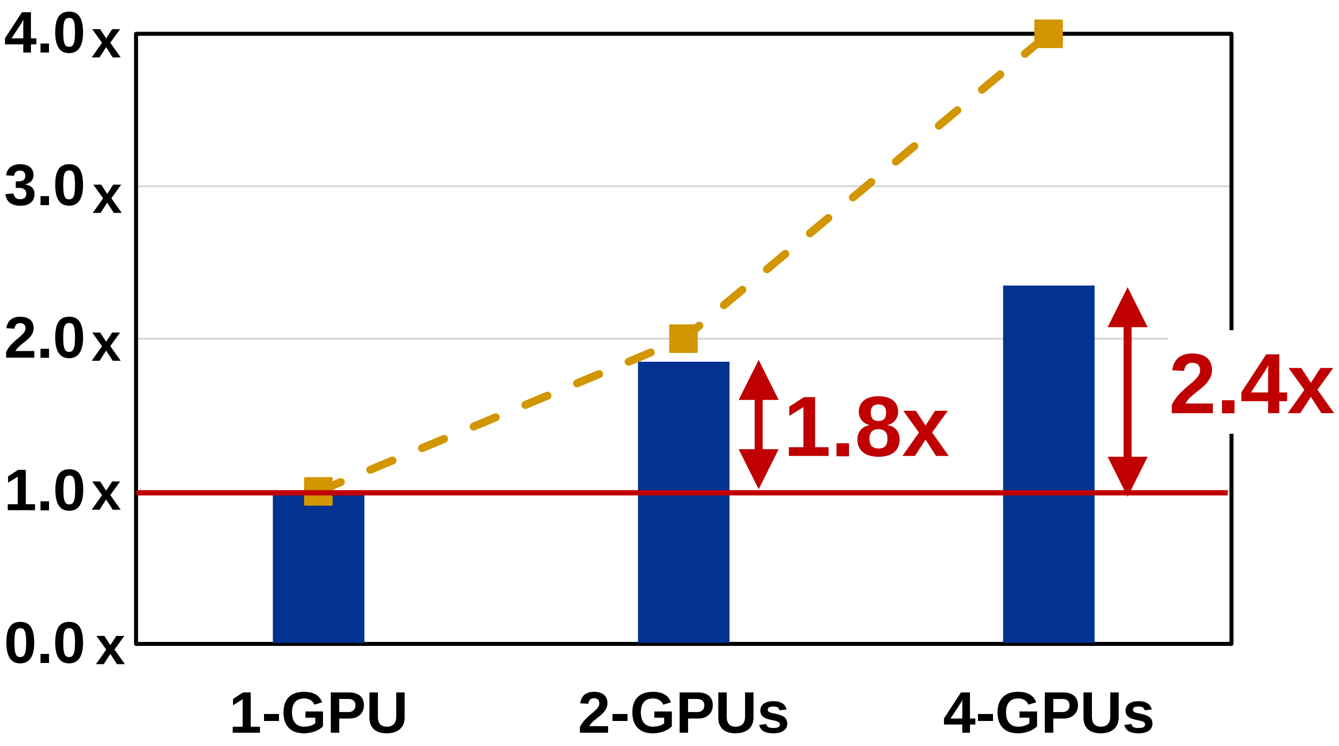
UVM for Multi-GPU

- Growing trend of multi-GPUs leveraging Unified Virtual Memory (UVM)



University of Pittsburgh

Multi-GPU Scalability



Why performance gap:

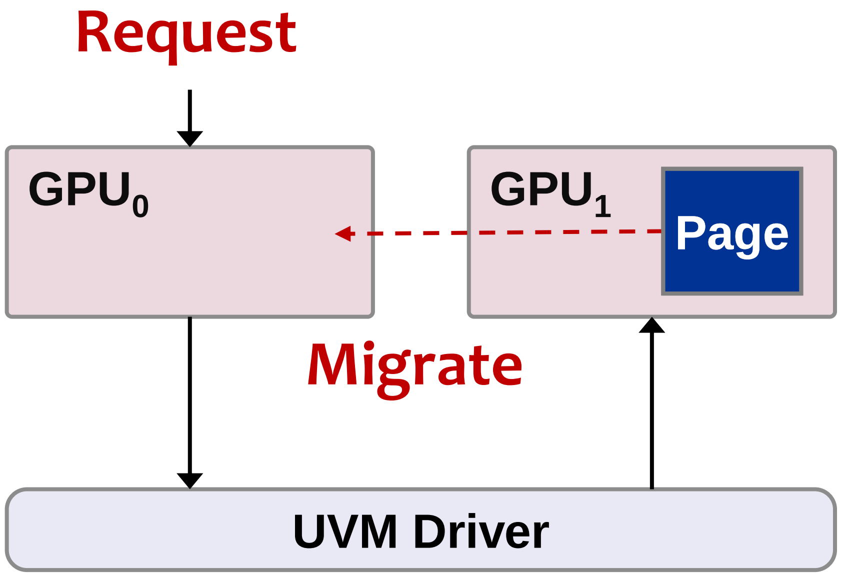
- NUMA data access
- Data transfer
- Address translation

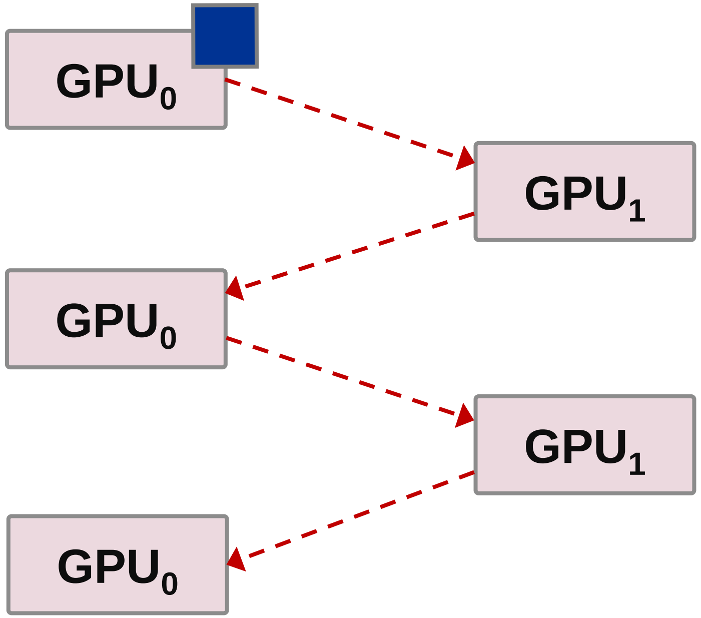
……

[2] https://developer.nvidia.com/blog/easy-multi-gpu-deep-learning-digits-2/

Multi-GPU Page Placement Schemes

1. On-Touch Migration: Request, Migrate.



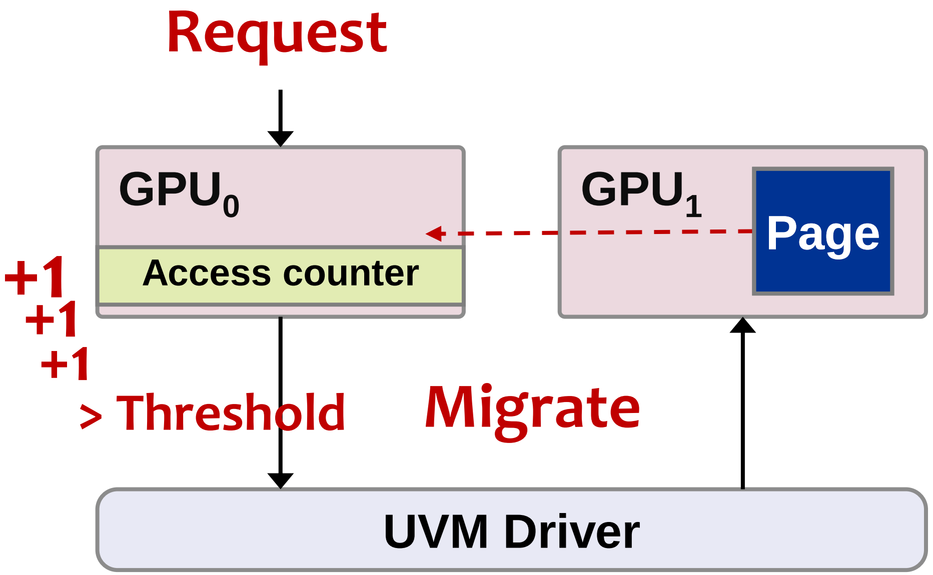


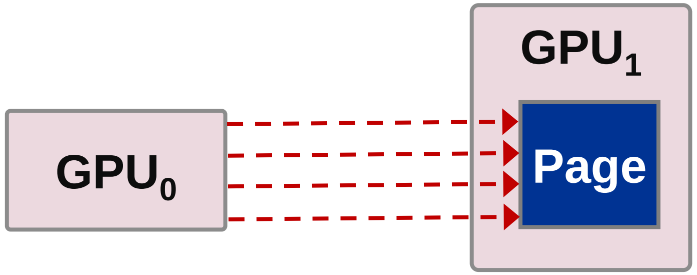
😇 Every memory access is local

😠 Ping-Pong effect

Multi-GPU Page Placement Schemes

2. Access Counter-based Migration: Reaching threshold, Migrate.



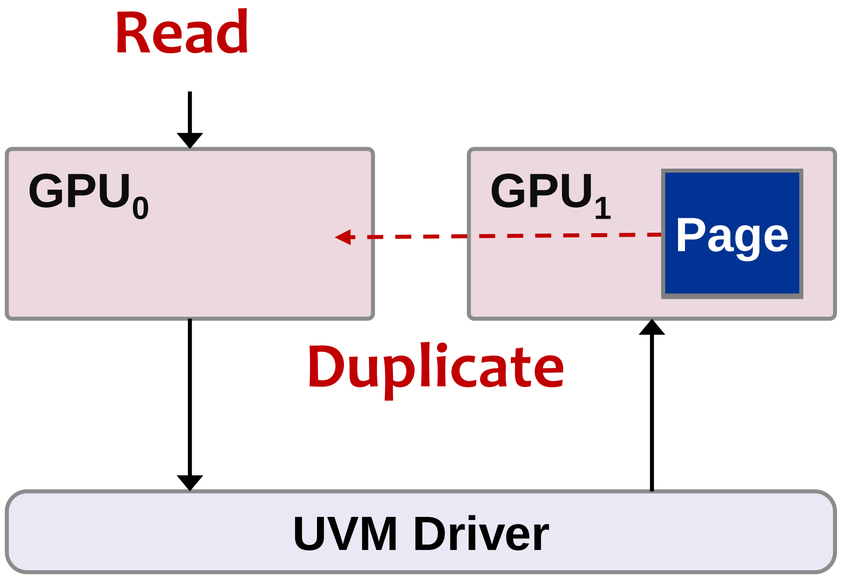


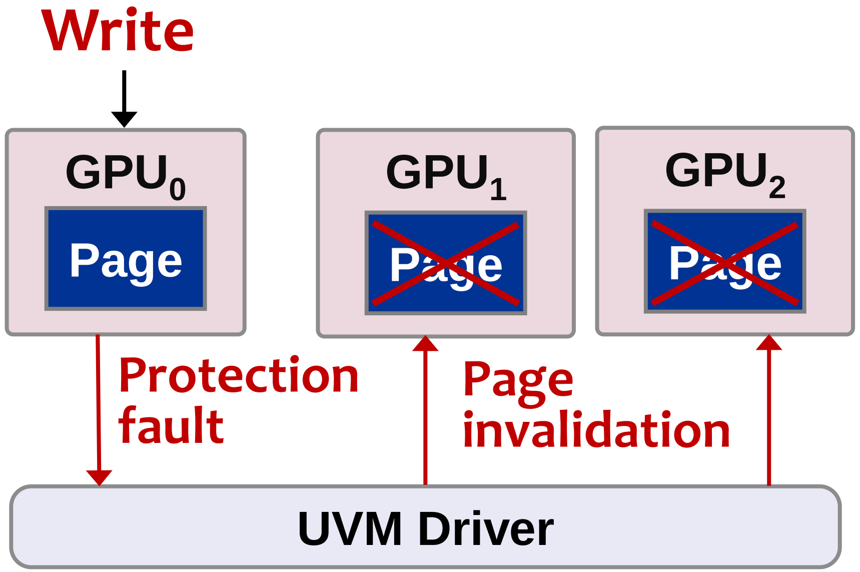
😇 Reduce Ping-Pong migration

😠 Substantial remote access

Multi-GPU Page Placement Schemes

3. Page Duplication: Read, Duplicate.

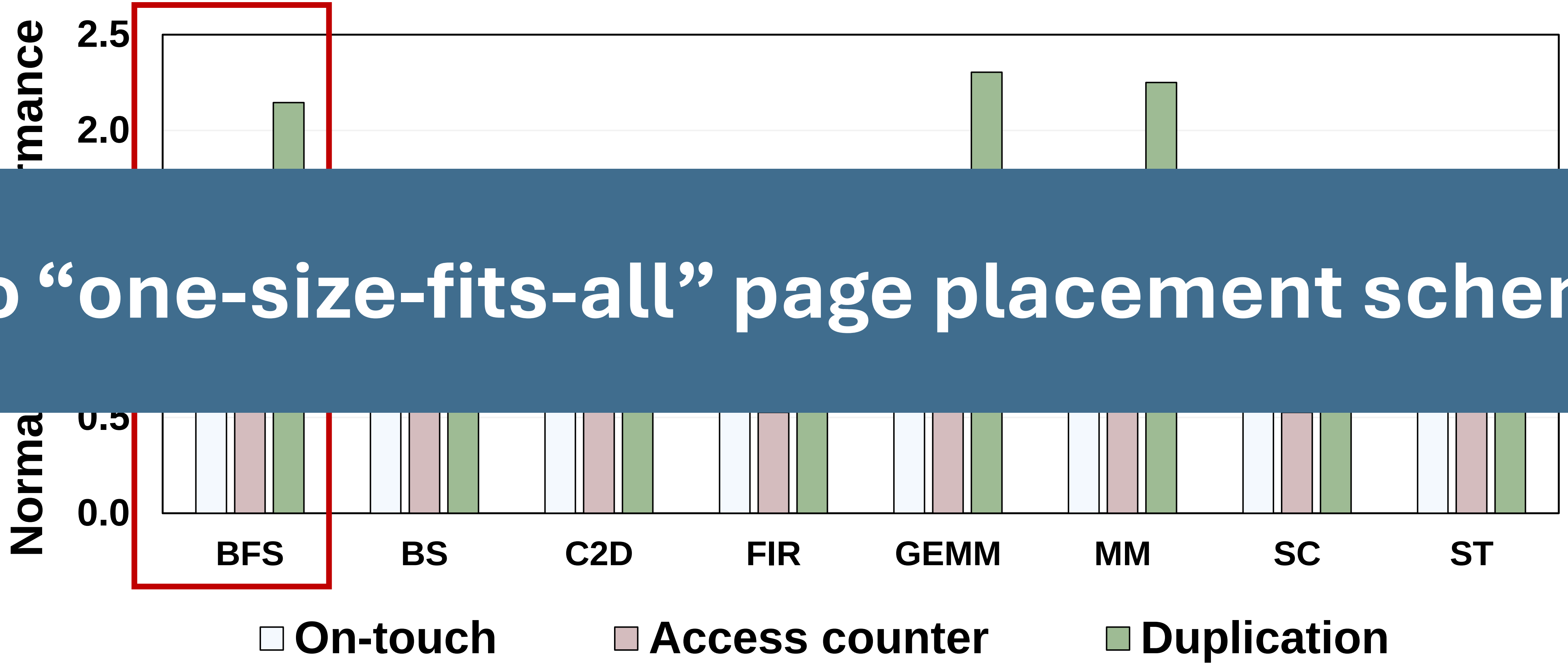




😇 Reduce migration & remote access

😣 Significant collapsing overhead

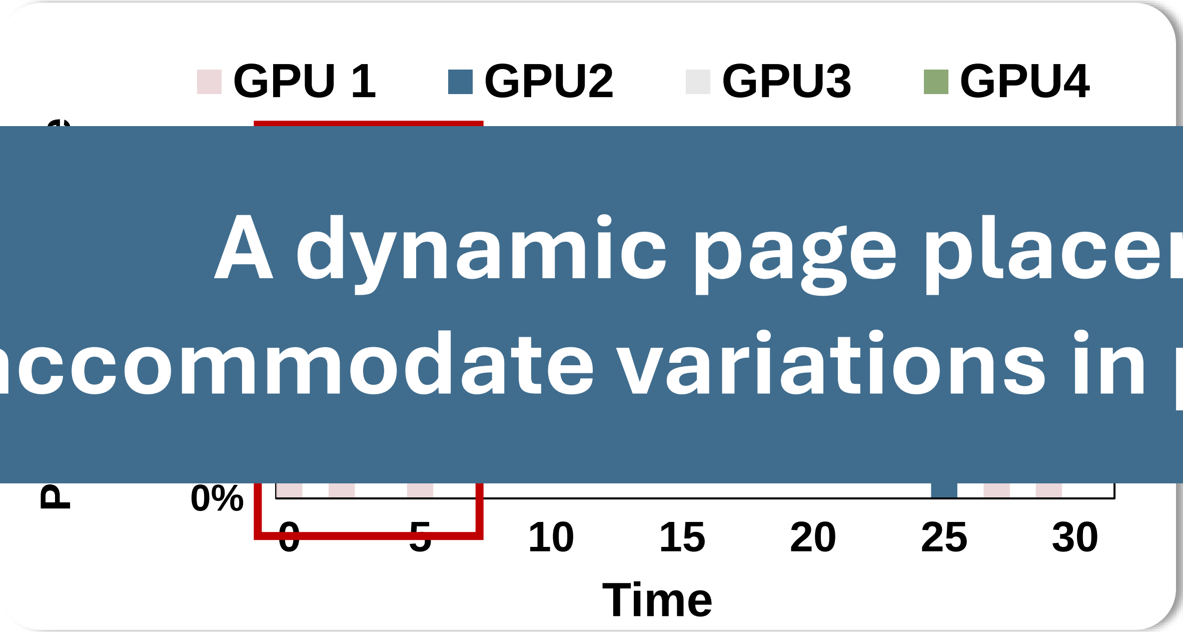
Performance of Page Placement Schemes



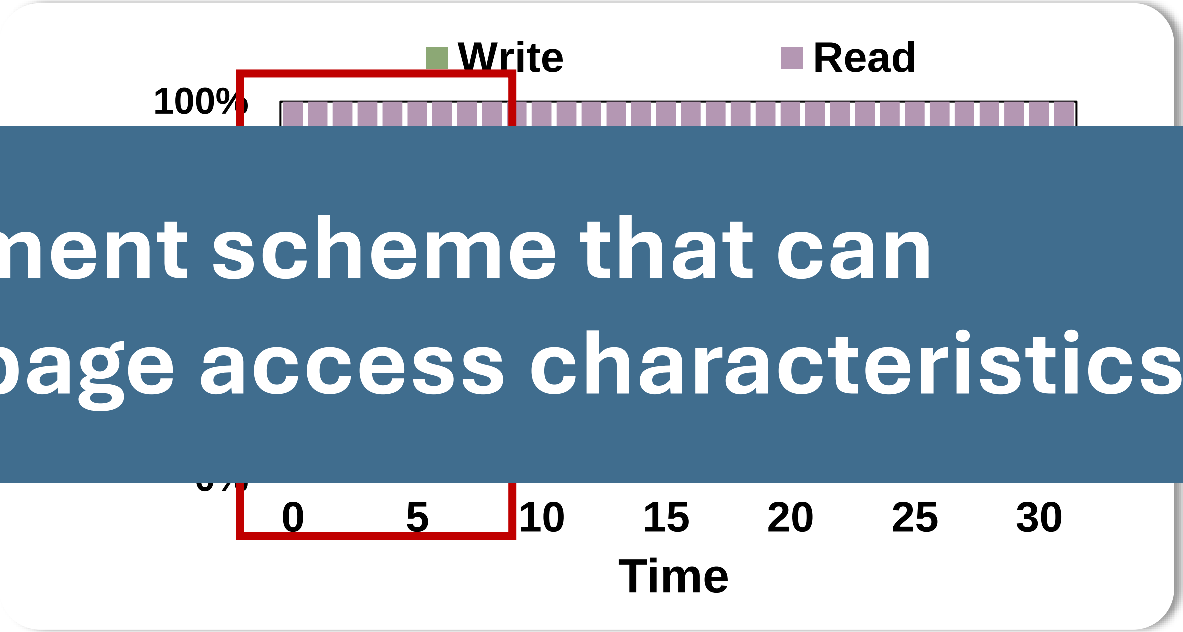
No “one-size-fits-all” page placement scheme

Page Access Characteristics

- ➢ The page-sharing / read-write patterns vary **within the same application**



Private / Share



Read / Write

A dynamic page placement scheme that can accommodate variations in page access characteristics

Problem Summary

Problem:

Delivered performance is constrained by NUMA overhead

No "one-size-fits-all" page placement scheme

Goal:

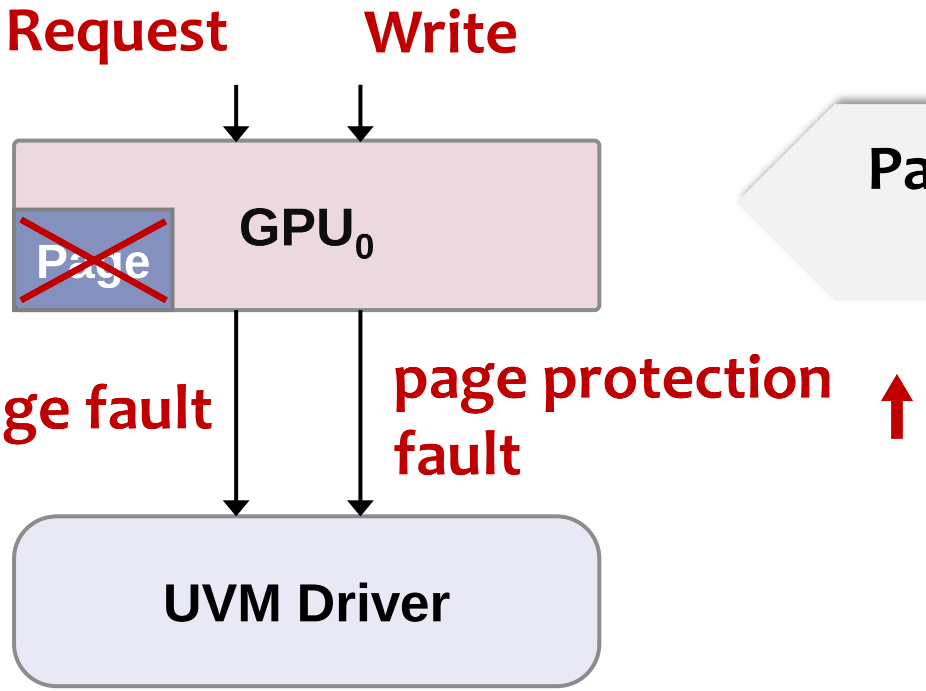
Effectively reduce NUMA overhead in multi-GPU by determining page placement scheme at runtime

GRIT (Fine-GRained dynamIc page placemenT)

Scheme change metric

Dynamically determine page placement scheme at runtime

GRIT – Scheme Change Metric



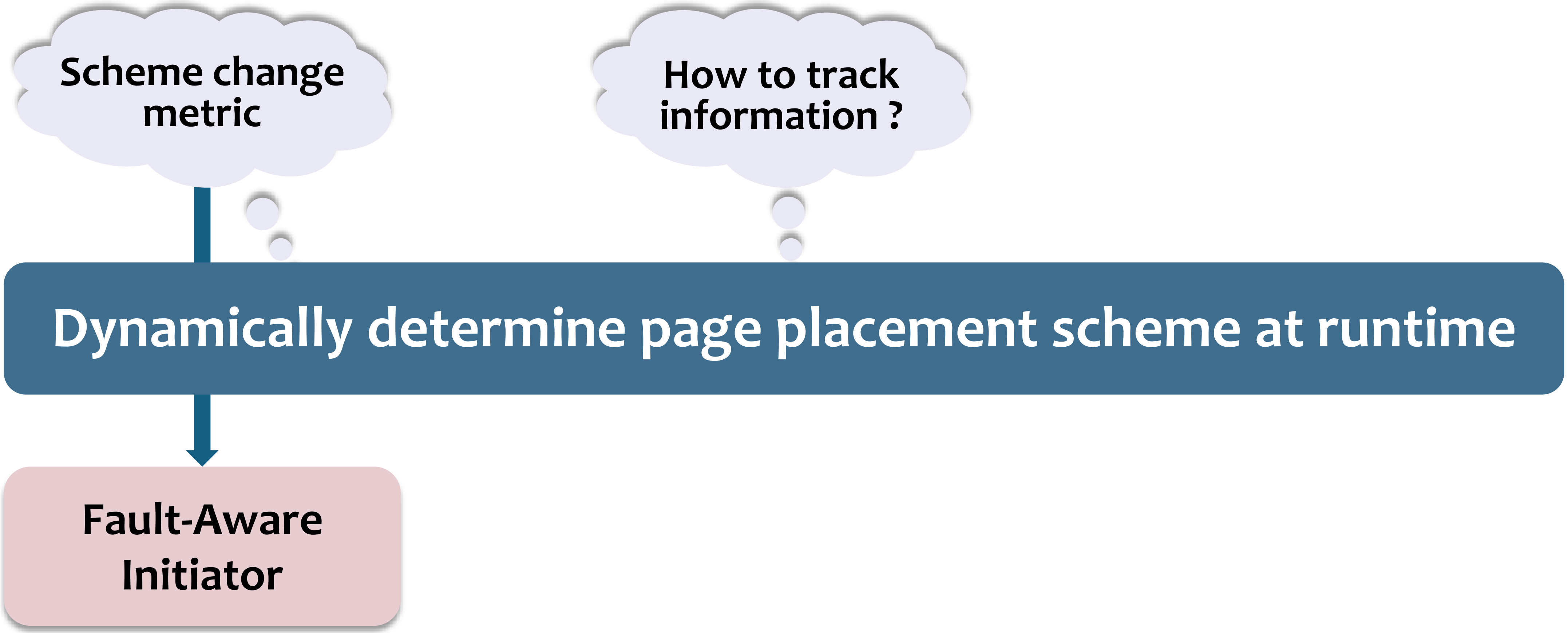
Current scheme is unsuitable

Page duplication should be avoided

Indicator: Number of page faults (local page fault & page protection fault)

University of Pittsburgh

GRIT: Dynamic Page Placement Scheme



Scheme change metric

How to track information ?

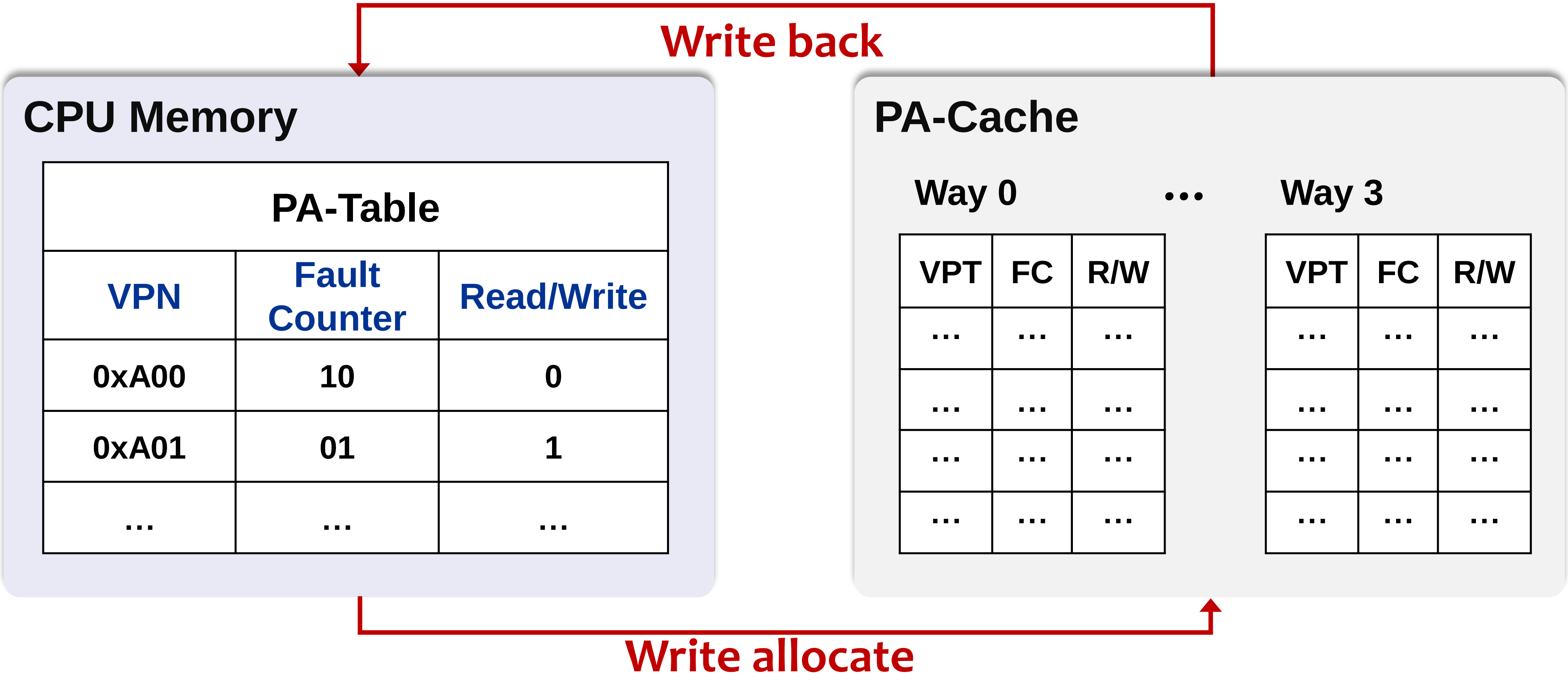
Dynamically determine page placement scheme at runtime

Fault-Aware Initiator

GRIT – Page Attribute Table and Cache (PA-Table & Cache)

😟 Additional memory access

😇 Facilitate lookup



Write back

CPU Memory

PA-Table		
VPN	Fault Counter	Read/Write
0xA00	10	0
0xA01	01	1
…	…	…

PA-Cache

Way 0

VPT	FC	R/W
…	…	…
…	…	…
…	…	…
…	…	…

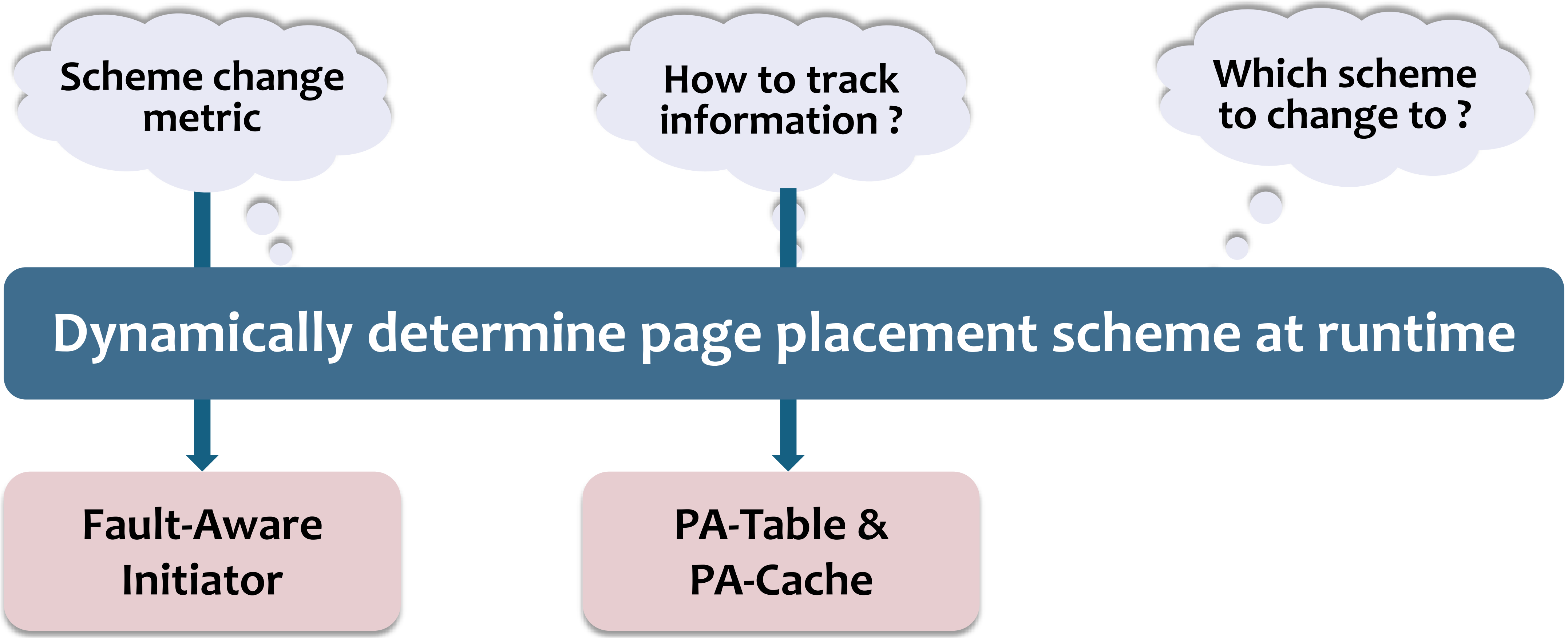
•••

Way 3

VPT	FC	R/W
…	…	…
…	…	…
…	…	…
…	…	…

Write allocate

GRIT: Dynamic Page Placement Scheme



Scheme change metric

How to track information ?

Which scheme to change to ?

Dynamically determine page placement scheme at runtime

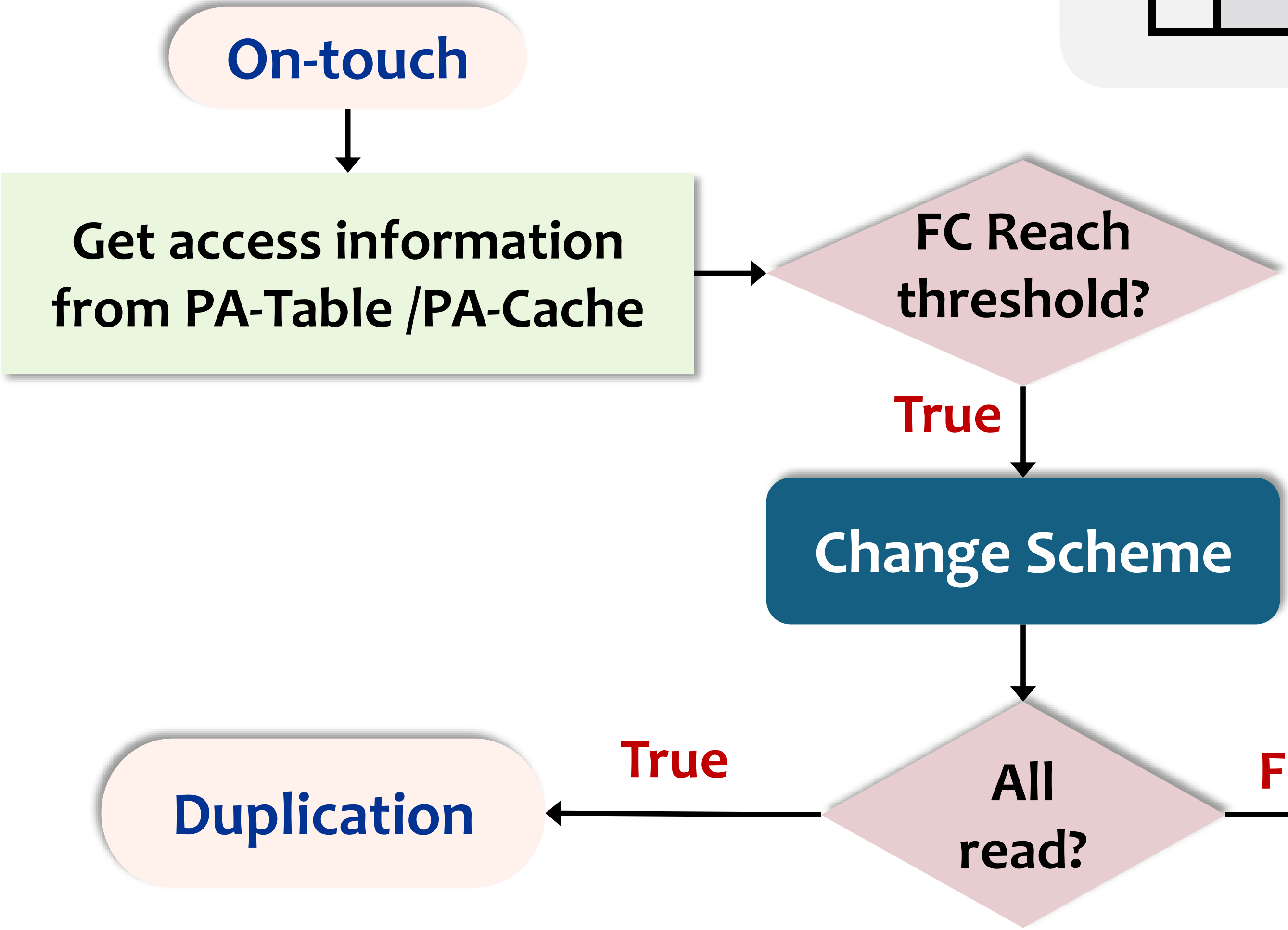
Fault-Aware Initiator

PA-Table & PA-Cache

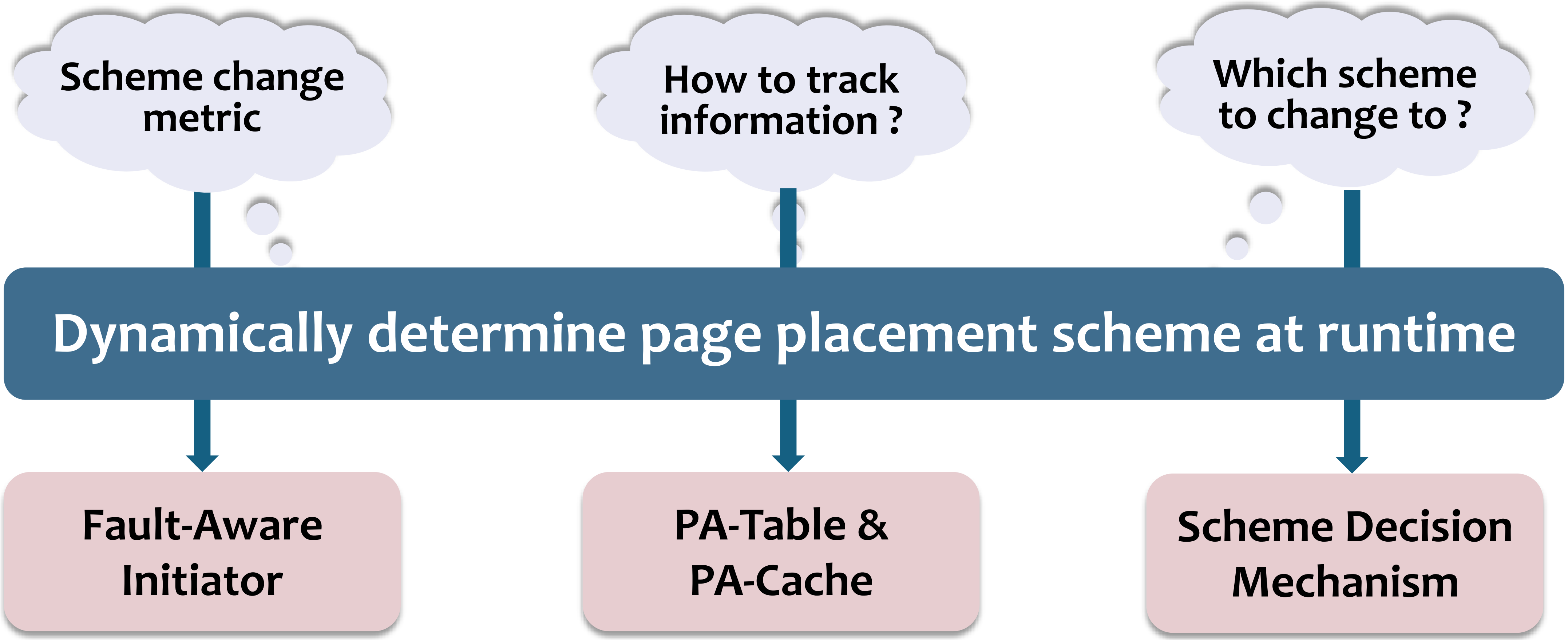
GRIT – Which Scheme

63	62:54	53:52	51:12	11	10:9
XD	Unused Bits (UB)		4 KB Page Frame Number (PFN)	UB	Scheme Bits

Scheme information is stored in host PTE.



GRIT: Dynamic Page Placement Scheme



Scheme change metric

How to track information ?

Which scheme to change to ?

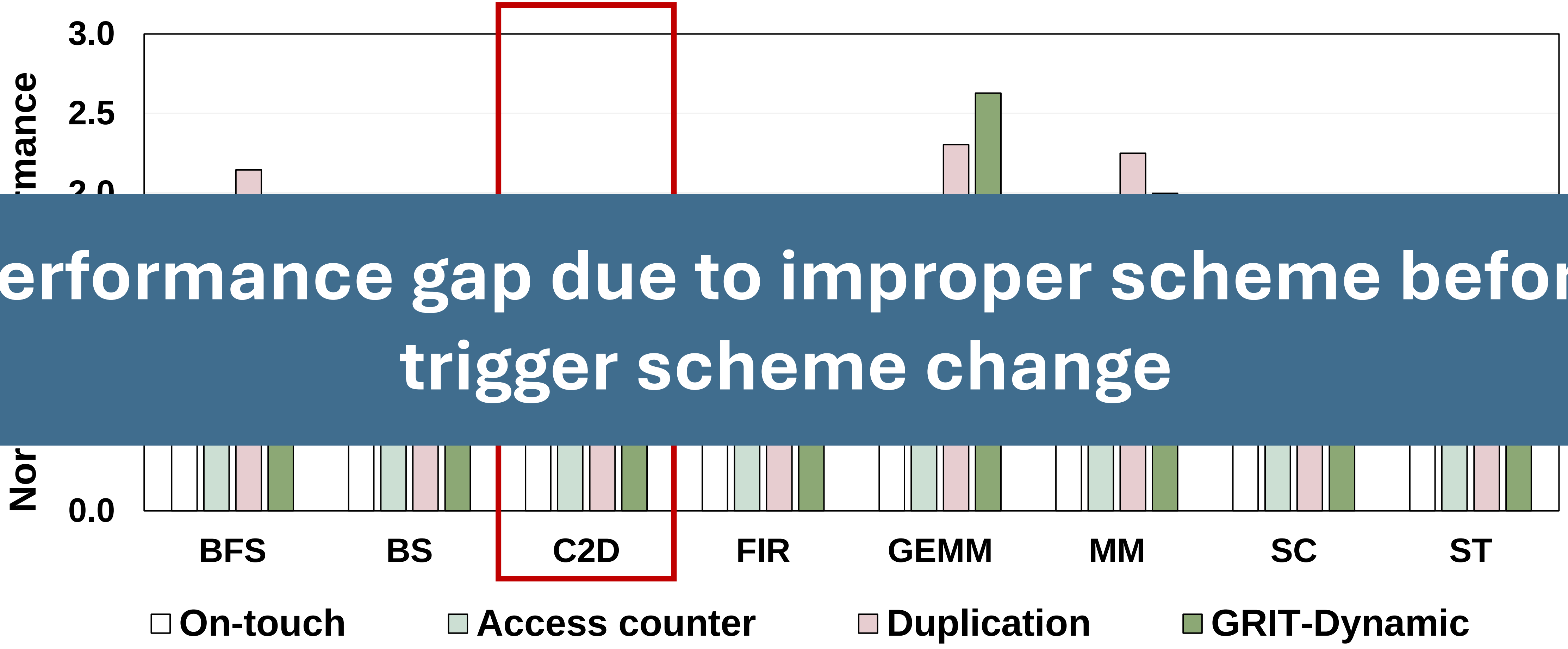
Dynamically determine page placement scheme at runtime

Fault-Aware Initiator

PA-Table & PA-Cache

Scheme Decision Mechanism

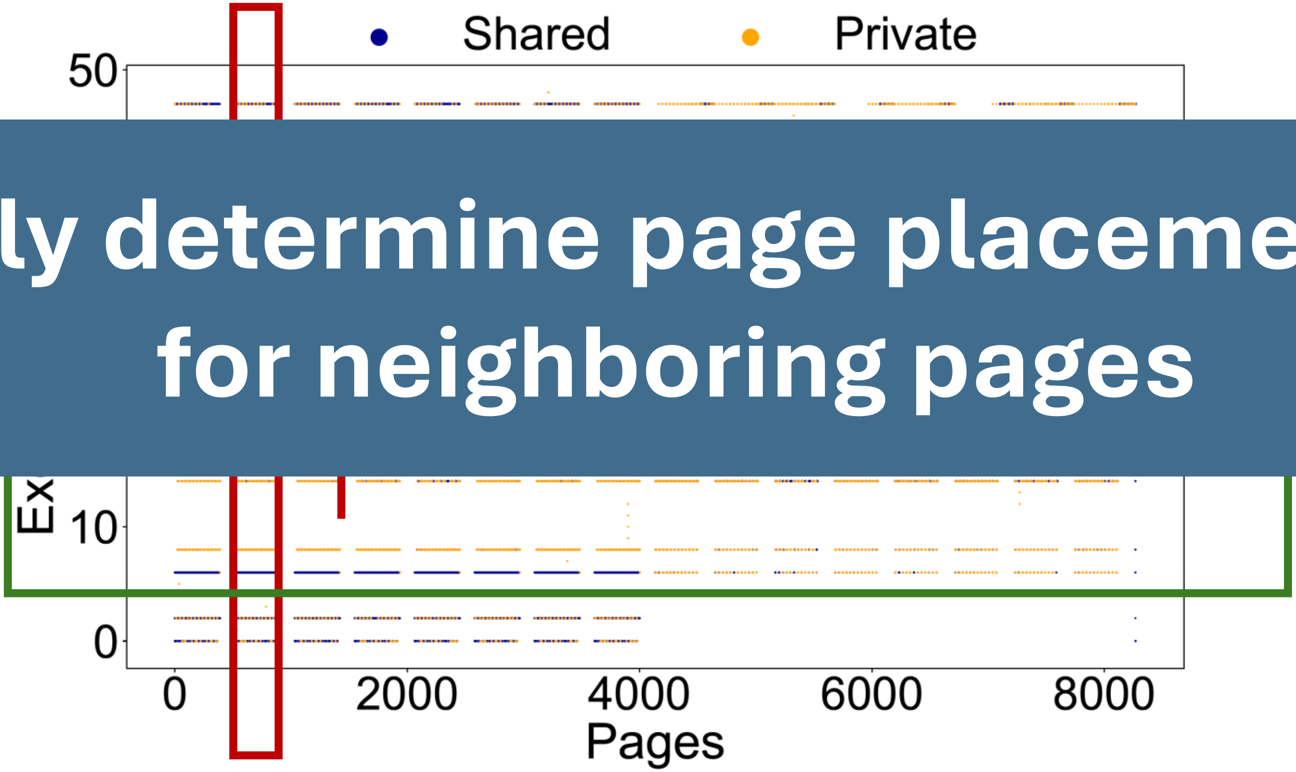
GRIT: Dynamic Page Placement Scheme



Performance gap due to improper scheme before trigger scheme change

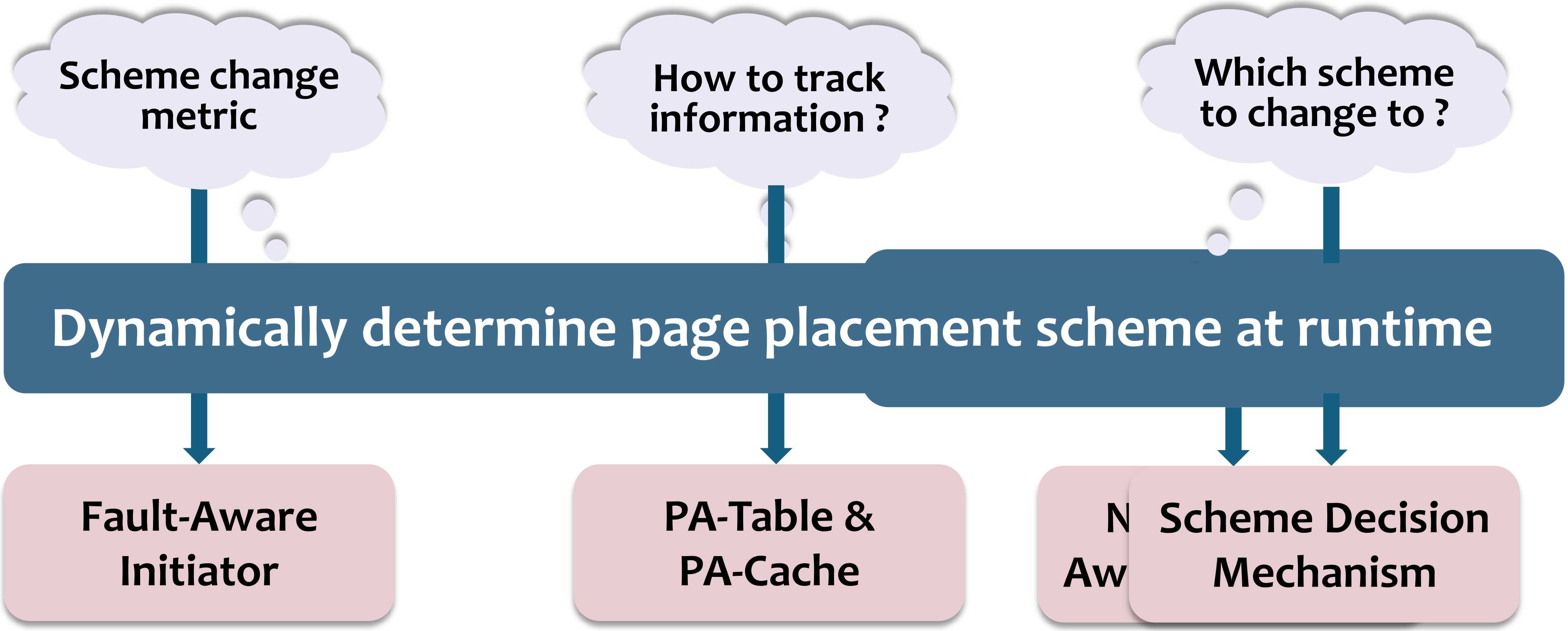
Page Attributes Characterization

- ➢ The **neighboring pages** tend to exhibit **similar access attributes**



Proactively determine page placement scheme for neighboring pages

GRIT: Neighboring-Aware Prediction



Scheme change metric

How to track information ?

Which scheme to change to ?

Dynamically determine page placement scheme at runtime

Fault-Aware Initiator

PA-Table & PA-Cache

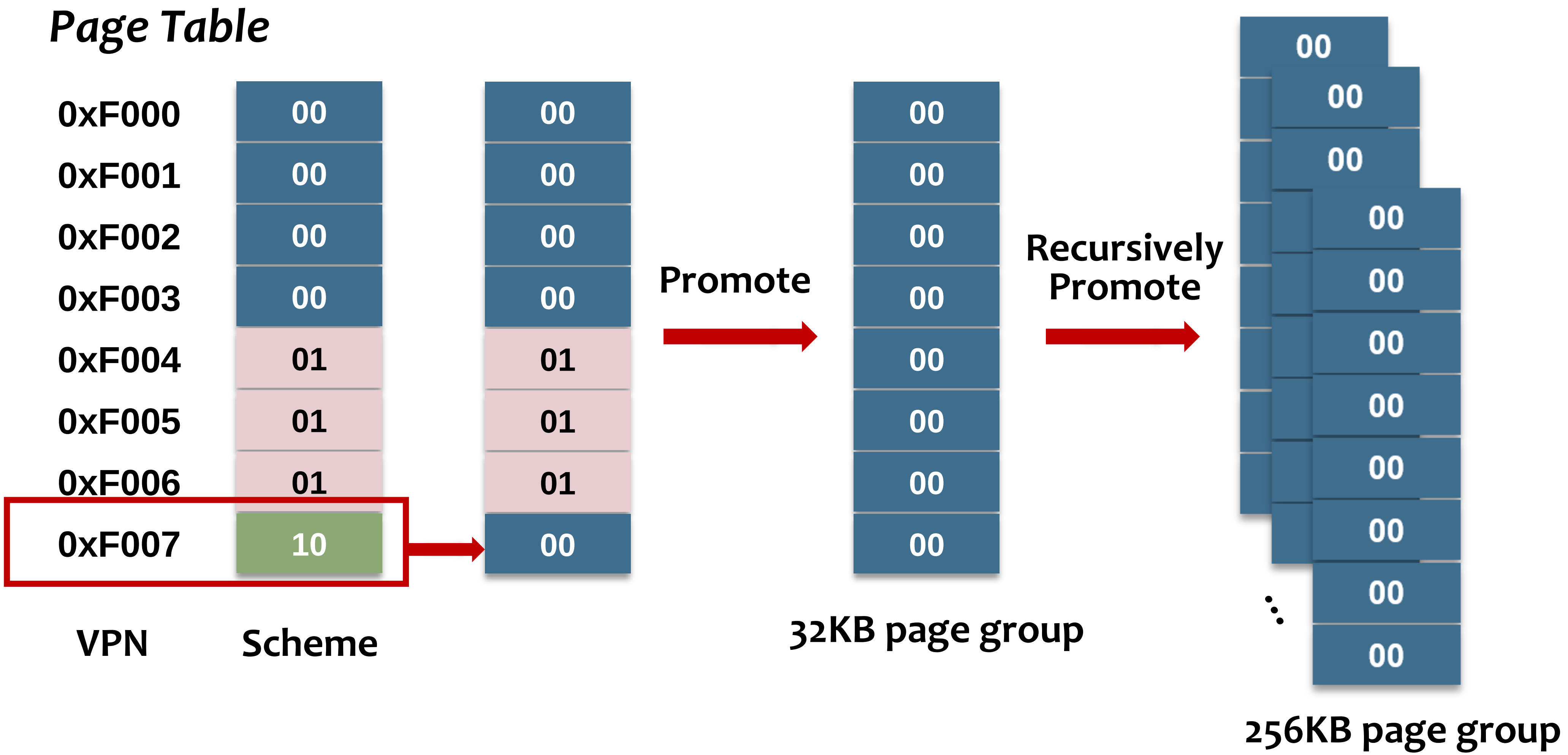
N

Aw

Scheme Decision Mechanism

GRIT: Neighboring-Aware Prediction

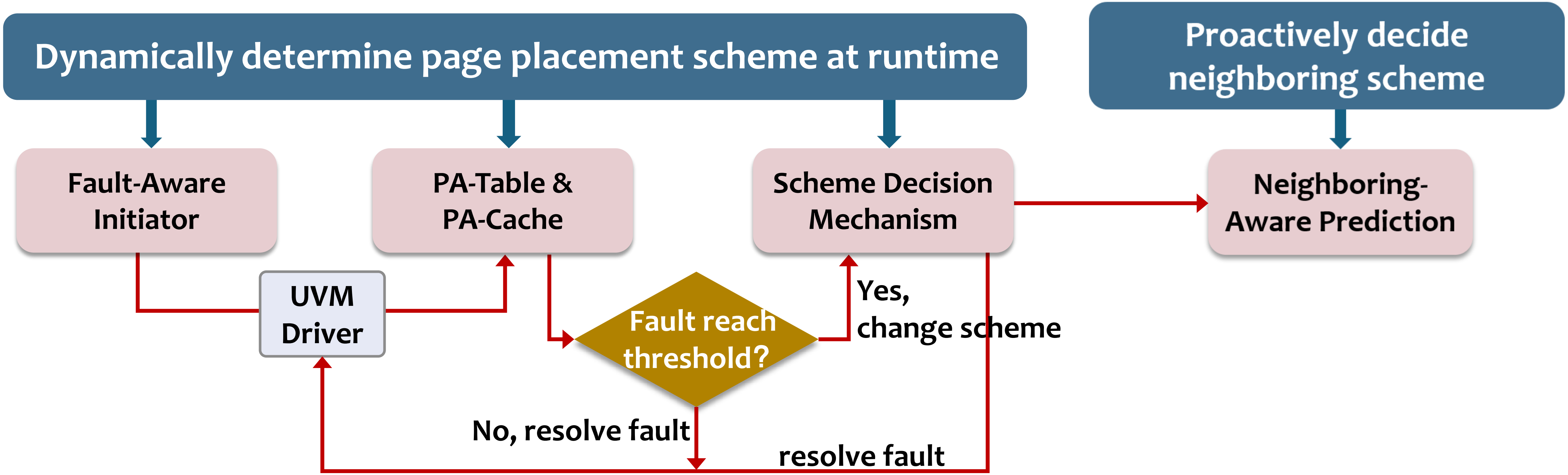
63	62:54	53:52
X D	Unused Bits (UB)	Group Bits



Page Table

VPN	Scheme
0xF000	00
0xF001	00
0xF002	00
0xF003	00
0xF004	01
0xF005	01
0xF006	01
0xF007	10

GRIT – Put All Together



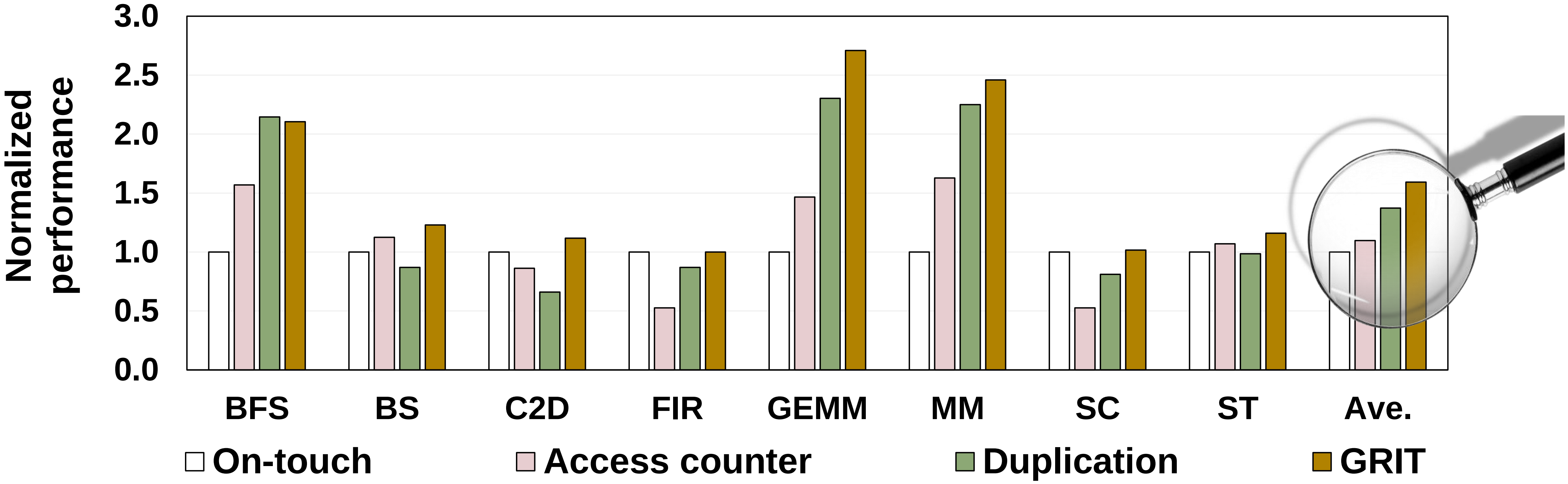
Dynamically determine page placement scheme at runtime
Proactively decide neighboring scheme
Fault-Aware Initiator
PA-Table & PA-Cache
Scheme Decision Mechanism
Neighboring-Aware Prediction
UVM Driver
Fault reach threshold?
Yes, change scheme
No, resolve fault
resolve fault

Methodology

- Simulator: MGPUSim [ISCA 19’]
- Workloads: 8 applications from Hetero-Mark, AMDAPPSDK , SHOC, and DNN Mark benchmark suites, including random, adjacent, and scatter-gather access patterns.

Detailed page placement scheme modeling in paper

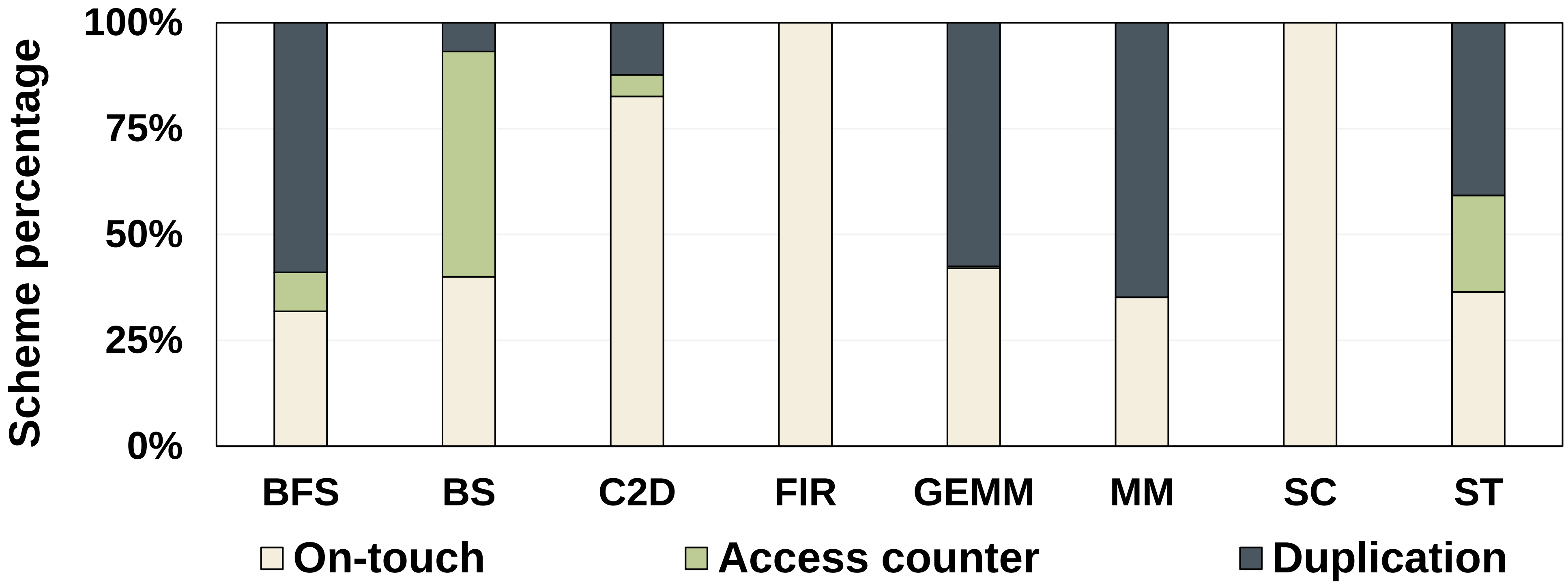
Evaluation – Overall Performance



GRIT achieves *60%, 49%, and 29%* performance improvement compared to uniformly employing on-touch, access counter-based, and page duplication scheme.

University of Pittsburgh

Evaluation – Scheme Breakdown



GRIT is able to distinguish page attributes and consistently select the most suitable scheme accordingly.

Summary

Problem: NUMA overheads in multi-GPU systems

- *No "**one-size-fits-all**" page placement scheme*

GRIT:

A. Dynamic page placement scheme **determines schemes in a fine-grained manner**

B. Neighboring-aware prediction **proactively determines adjacent page scheme**

Improves **performance** by **60%** on average.

Thanks! Q&A

GRIT: Enhancing Multi-GPU Performance with Fine-Grained Dynamic Page Placement

Yueqi Wang*[1], ***Bingyao Li****[1], Aamer Jaleel[2], Jun Yang[1], Xulong Tang[1]

[1]University of Pittsburgh, [2]NVIDIA