

Elastic Cuckoo Page Tables: Rethinking Virtual Memory Translation for Parallelism

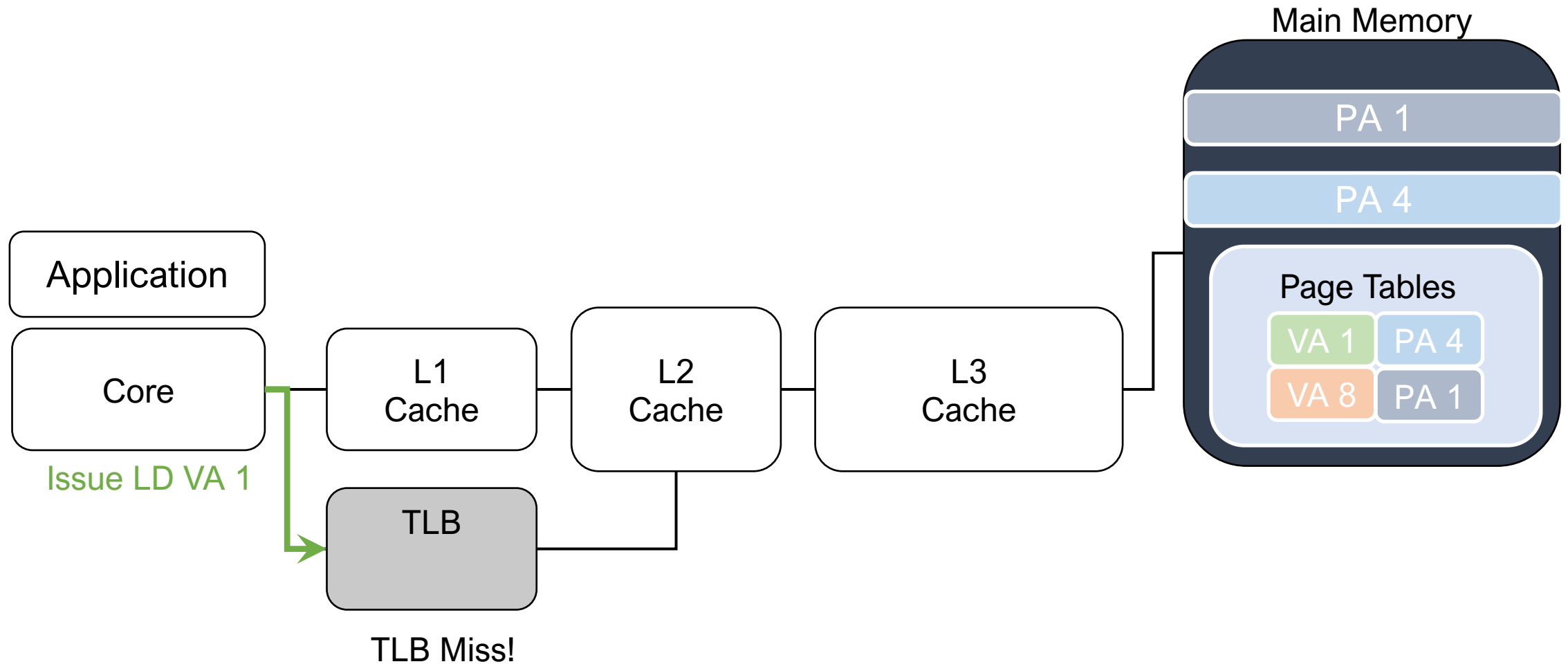
Dimitrios Skarlatos, Apostolos Kokolis, Tianyin Xu, Josep Torrellas

University of Illinois at Urbana-Champaign

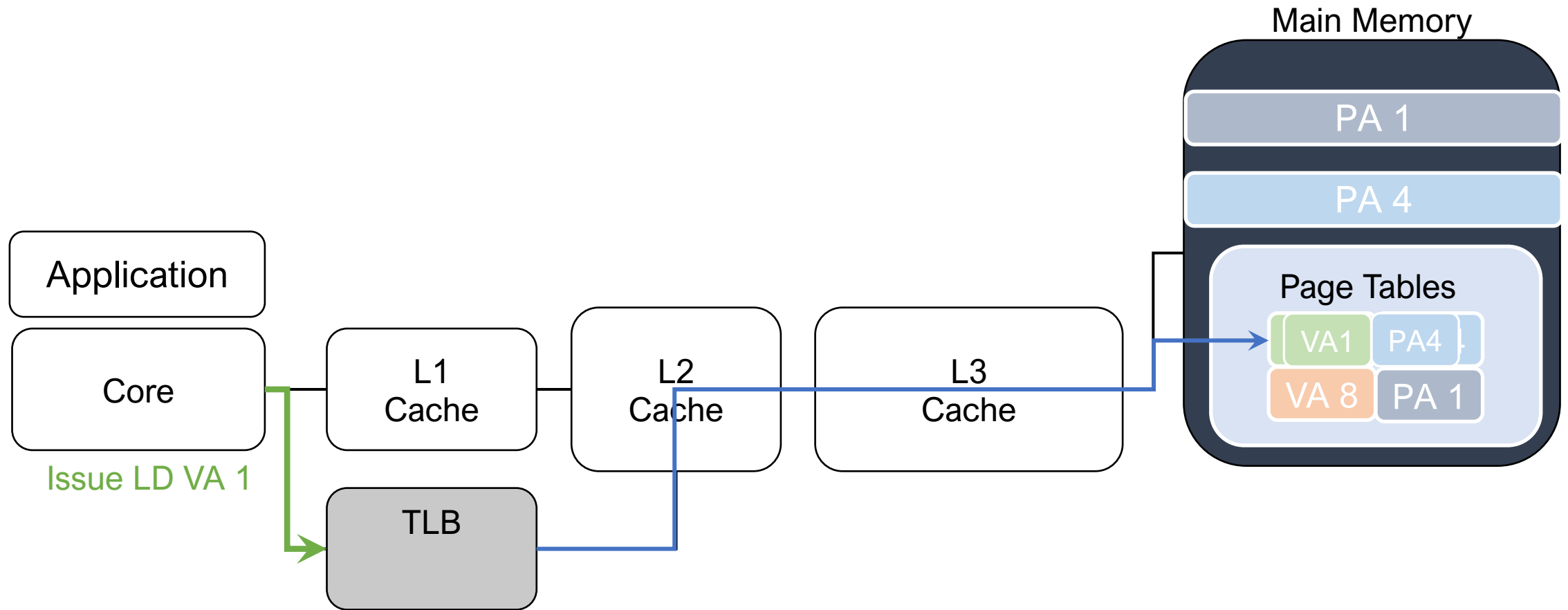
skarlat2.web.engr.illinois.edu

ASPLOS 2020

Virtual Memory Translation is Expensive



Virtual Memory Translation is Expensive

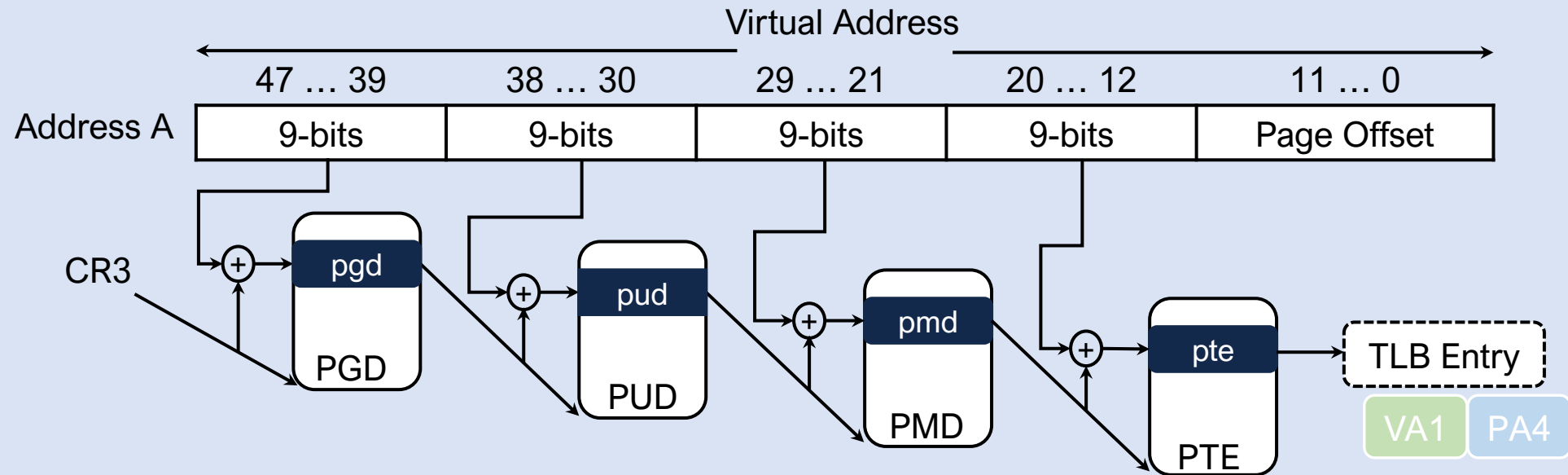


TLB Miss → "Page Walk" = Fetch entry from page table

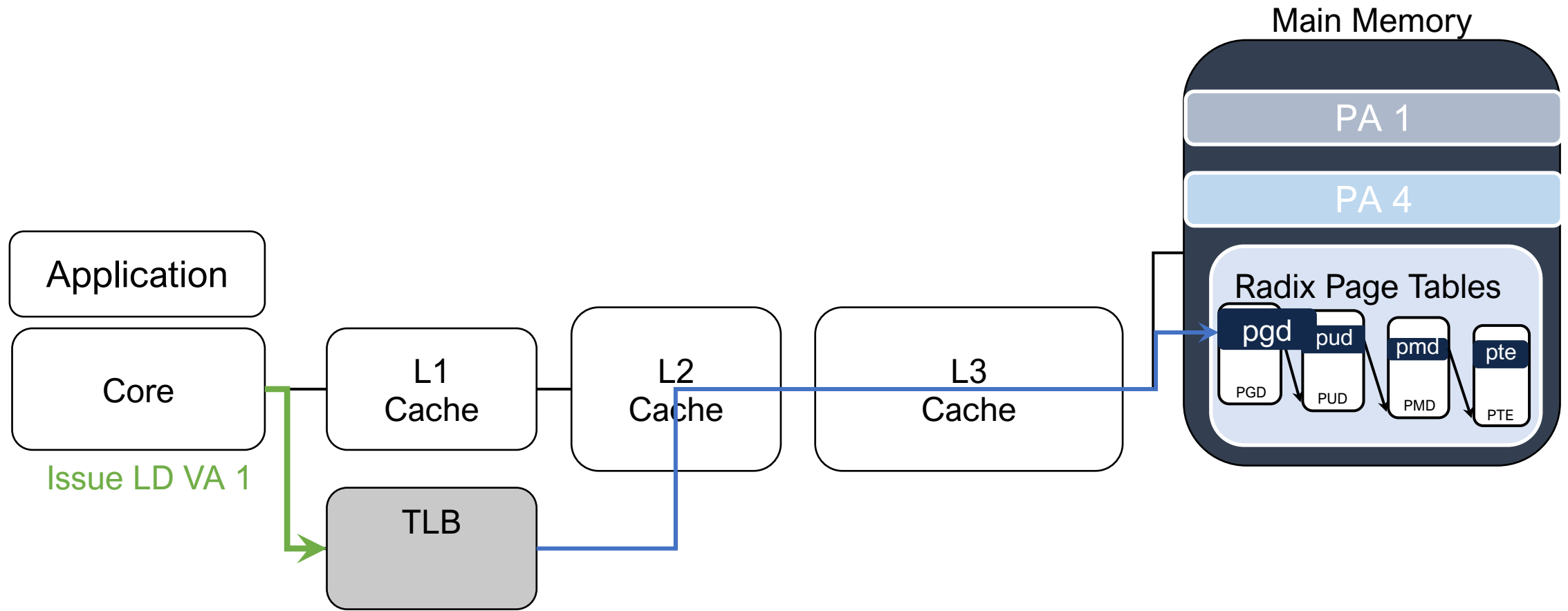
x86-64 Radix Page Tables



x86-64 Radix Page Tables

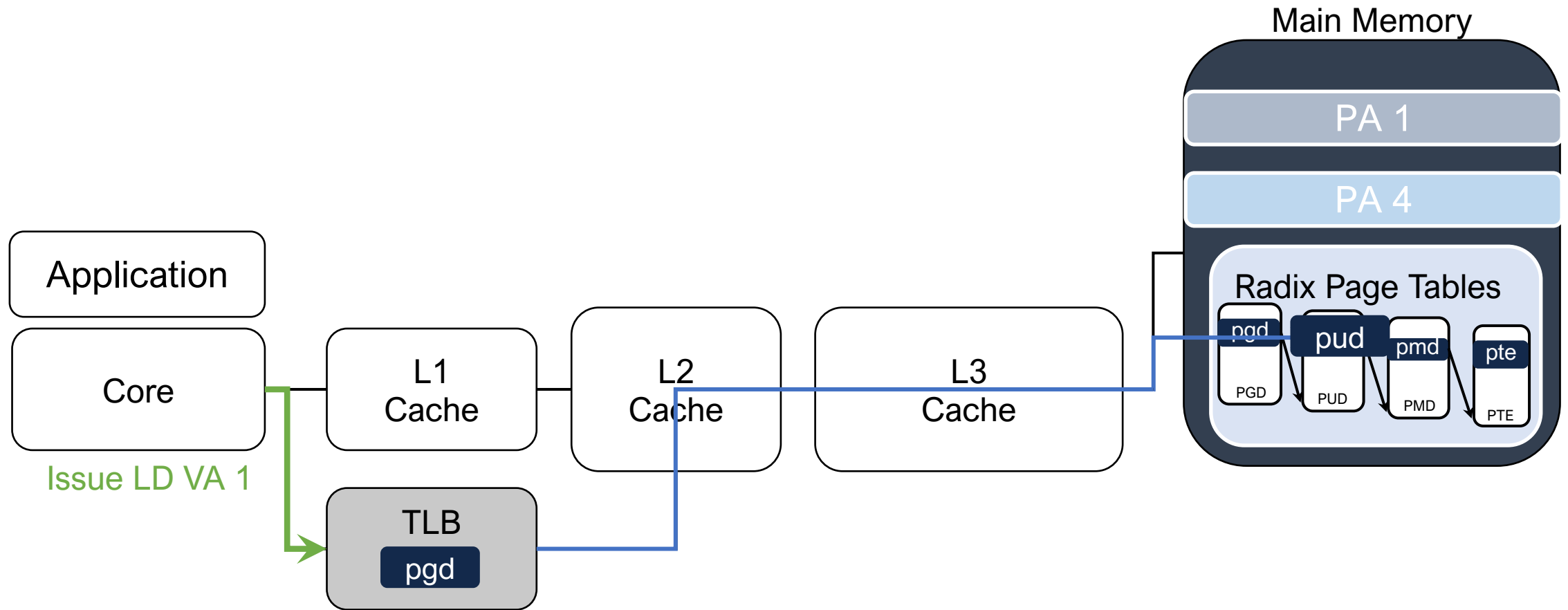


Virtual Memory Translation is Expensive



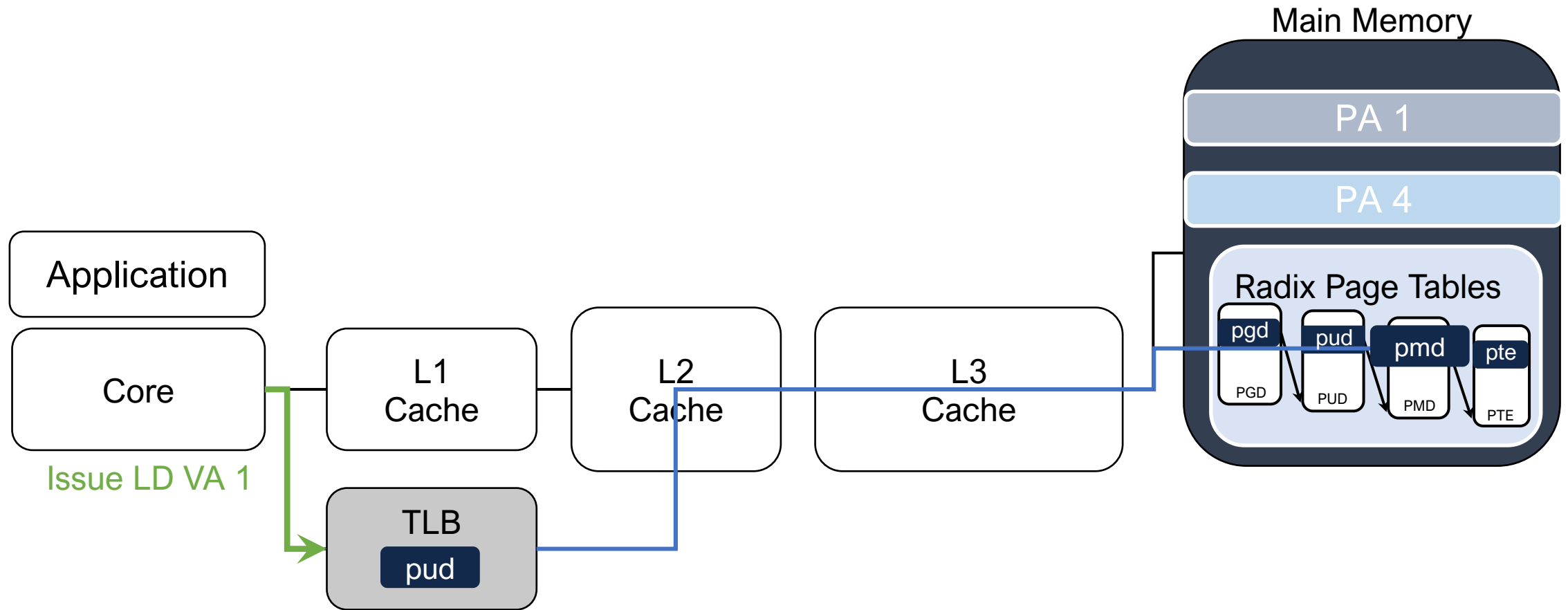
TLB Miss → “Page Walk” = Fetch entry from radix page table

Virtual Memory Translation is Expensive



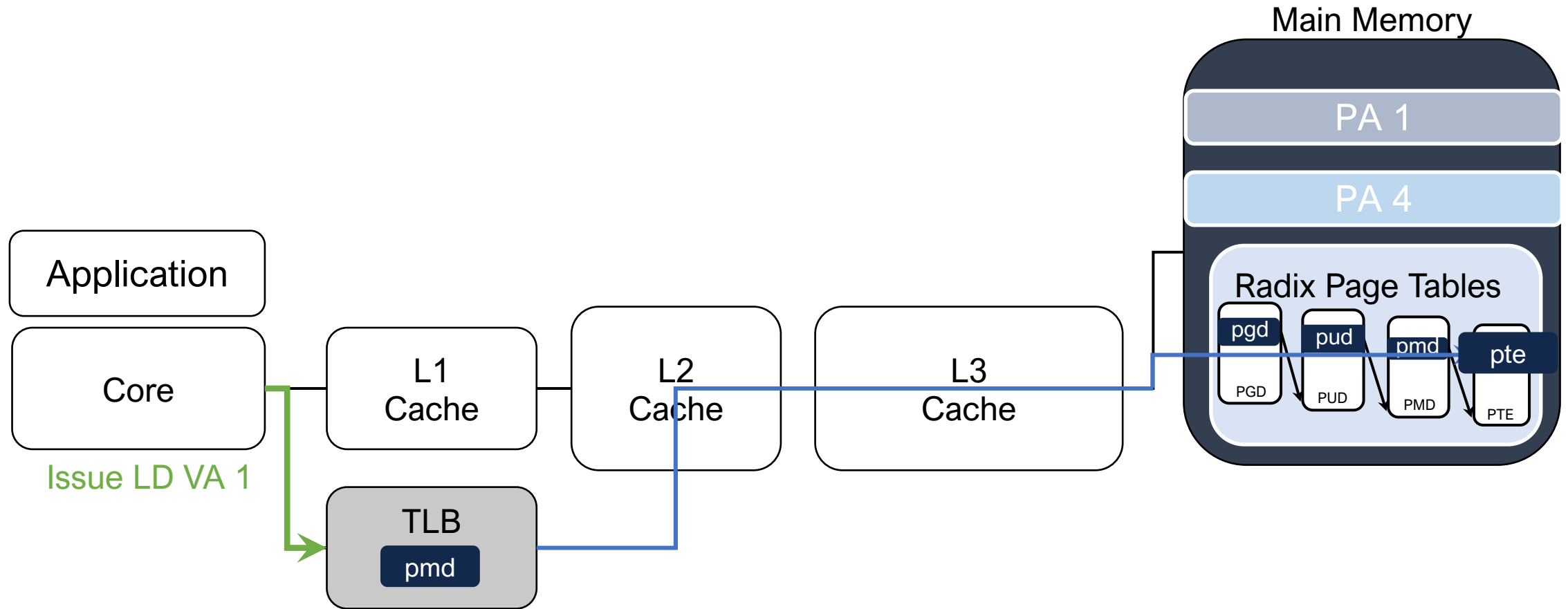
TLB Miss → **“Page Walk”** = Fetch entry from radix page table

Virtual Memory Translation is Expensive



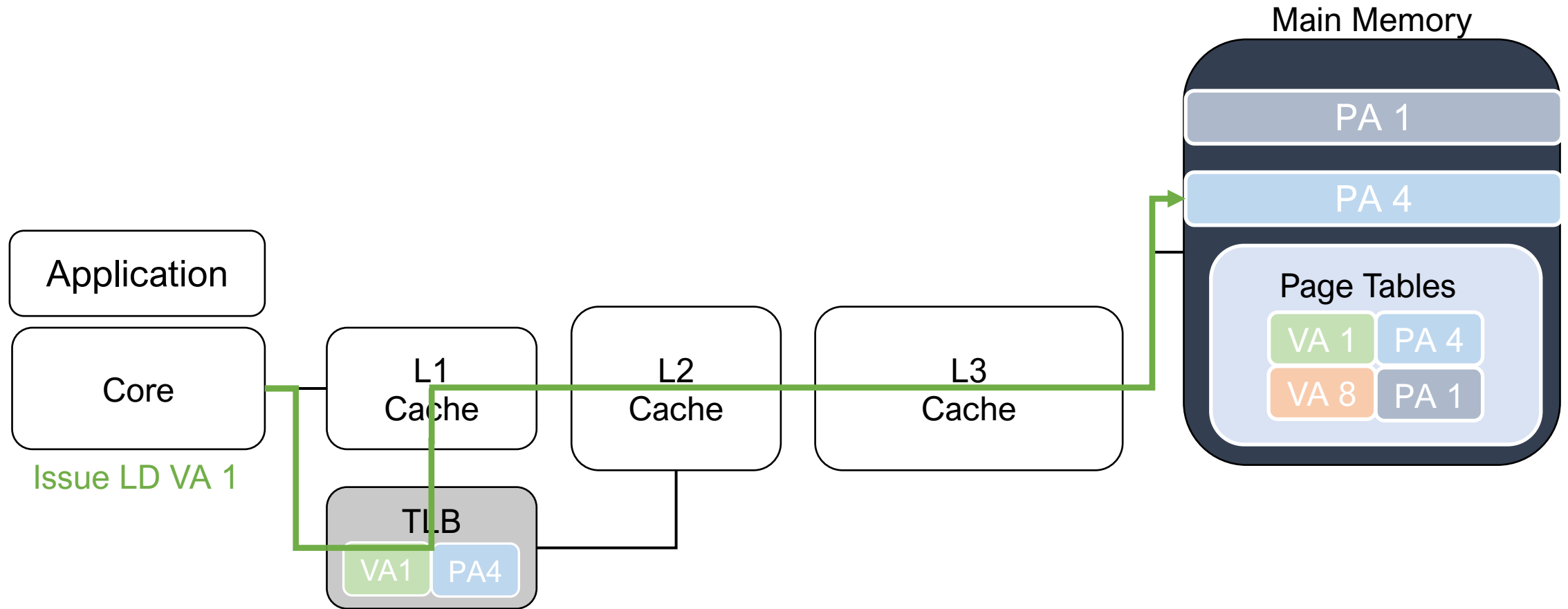
TLB Miss → **“Page Walk”** = Fetch entry from radix page table

Virtual Memory Translation is Expensive

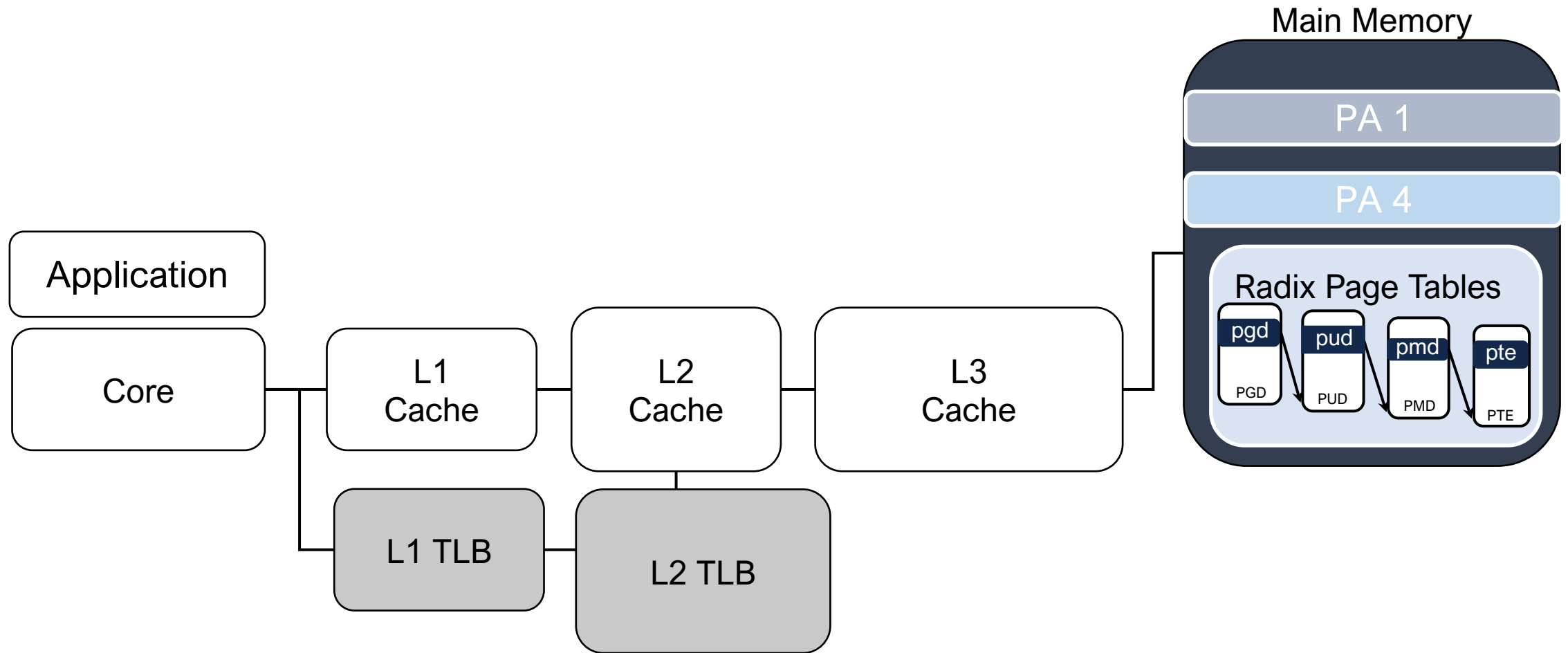


TLB Miss → **“Page Walk”** = Fetch entry from radix page table

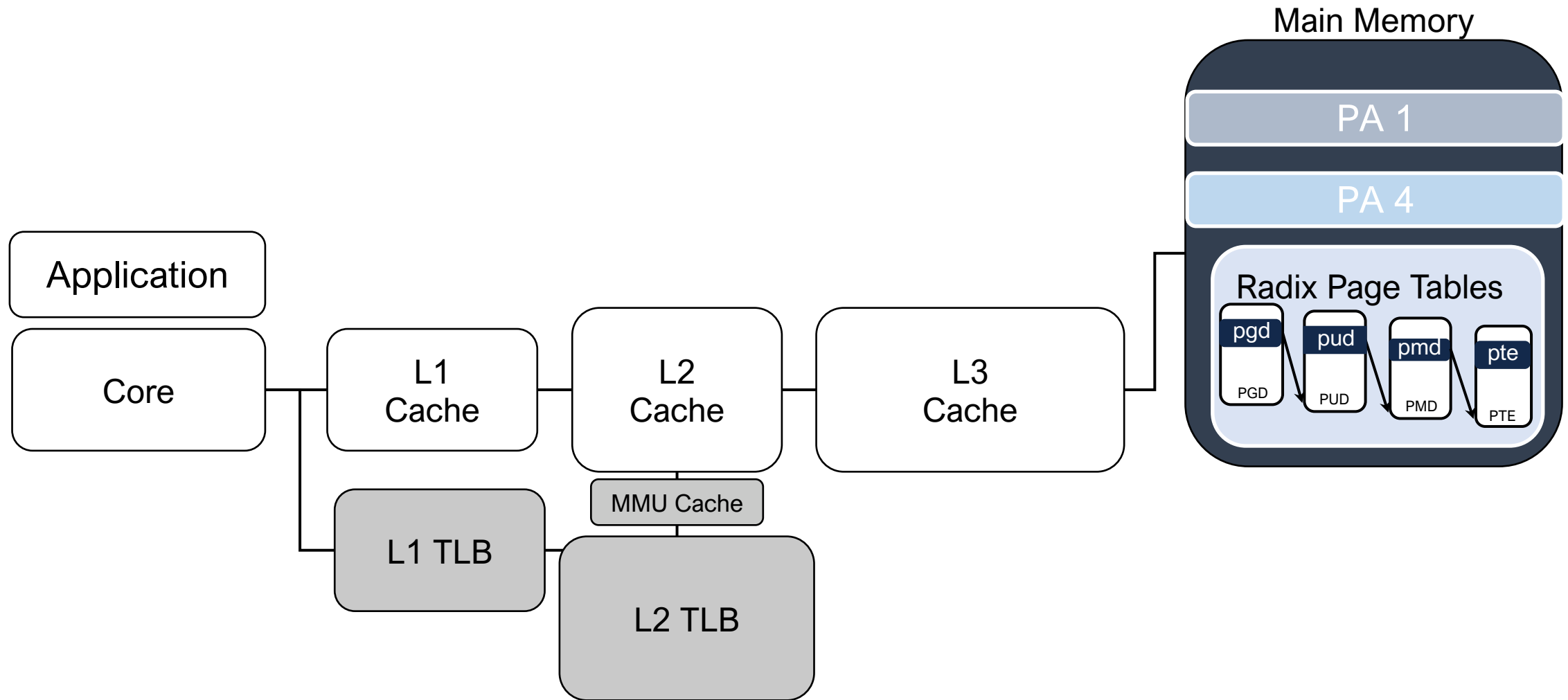
Virtual Memory Translation is Expensive



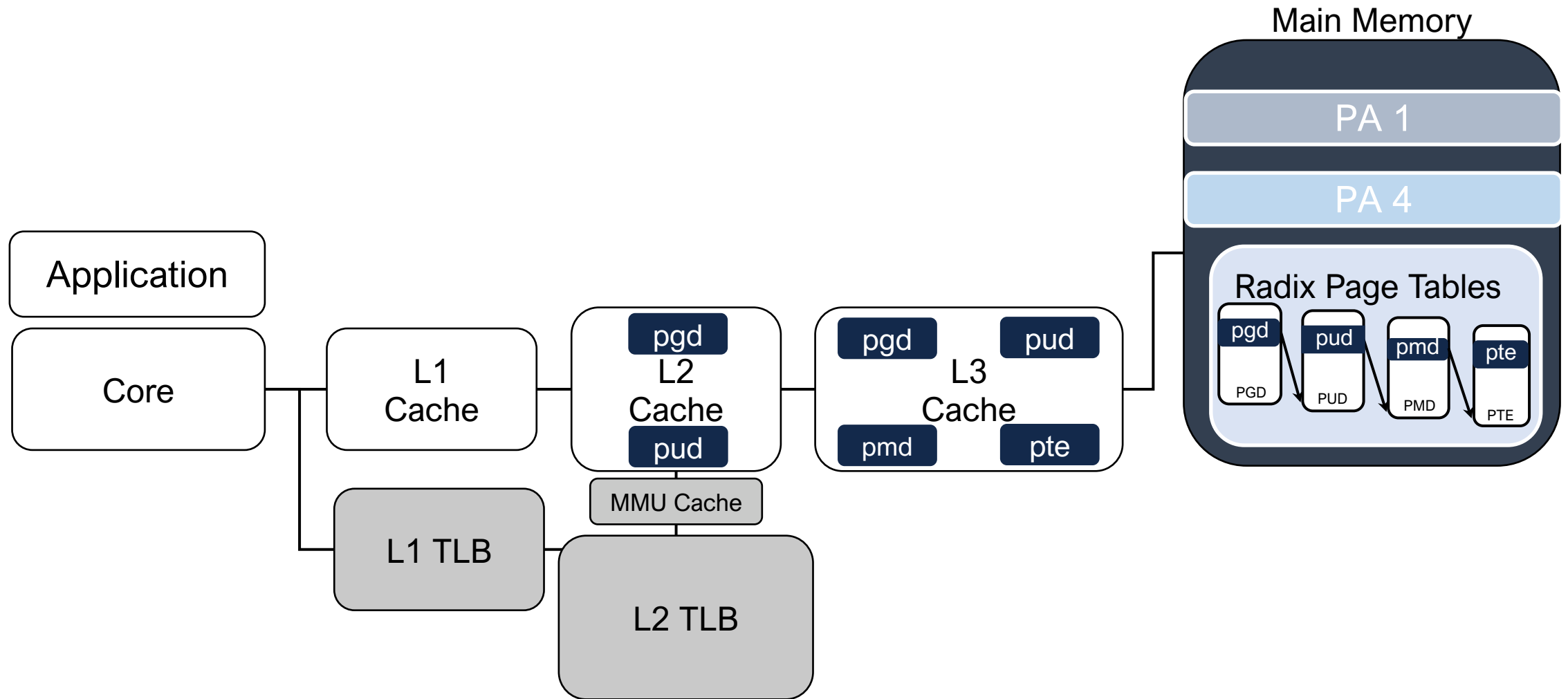
Multilevel TLBs



Memory Management Unit (MMU) Cache

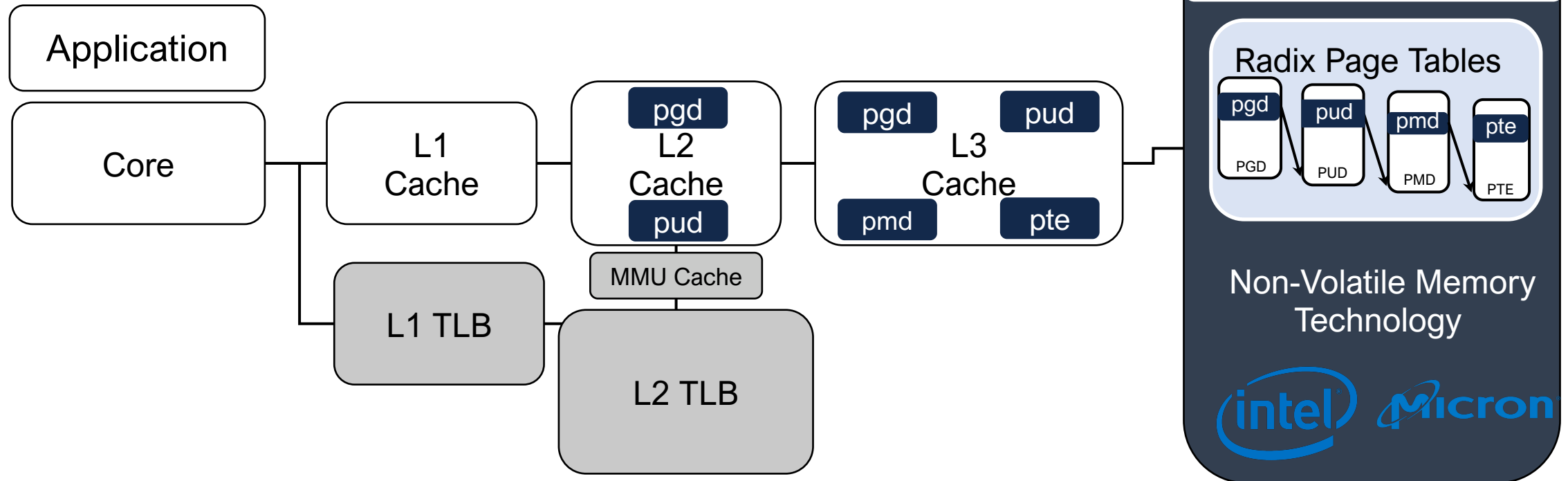


Translations in Data Caches



NVM will Make the Problem Worse

 Sunny Cove introduces 5-Level Radix Page Tables!!



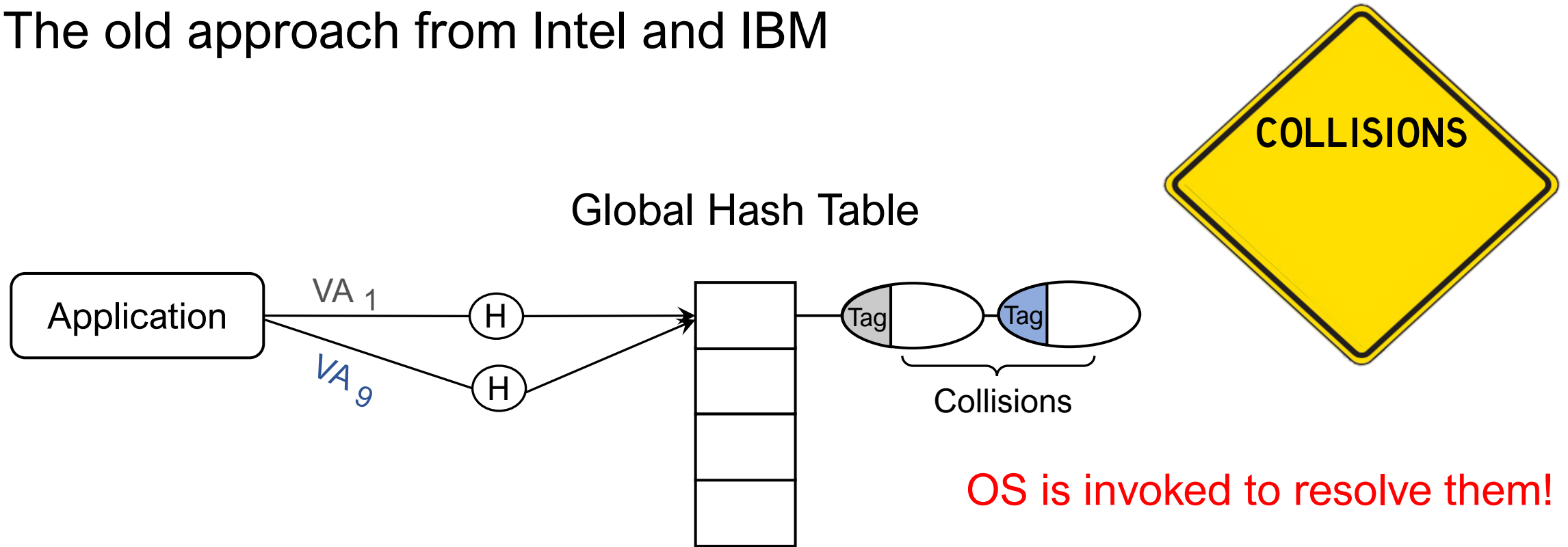
Contribution: Elastic Cuckoo Page Tables

- Rethinking virtual memory translation for parallelism
- Idea: Dynamically resizable page tables based on cuckoo hashing
- No sequential page table lookups → parallel single-step lookups
- Application speedup over state-of-the-art:
 - 3-28% with 4KB pages
 - 3-18% with Huge pages

Alternative: A Global Hashed Page Table

Alternative: A Global Hashed Page Table

The old approach from Intel and IBM



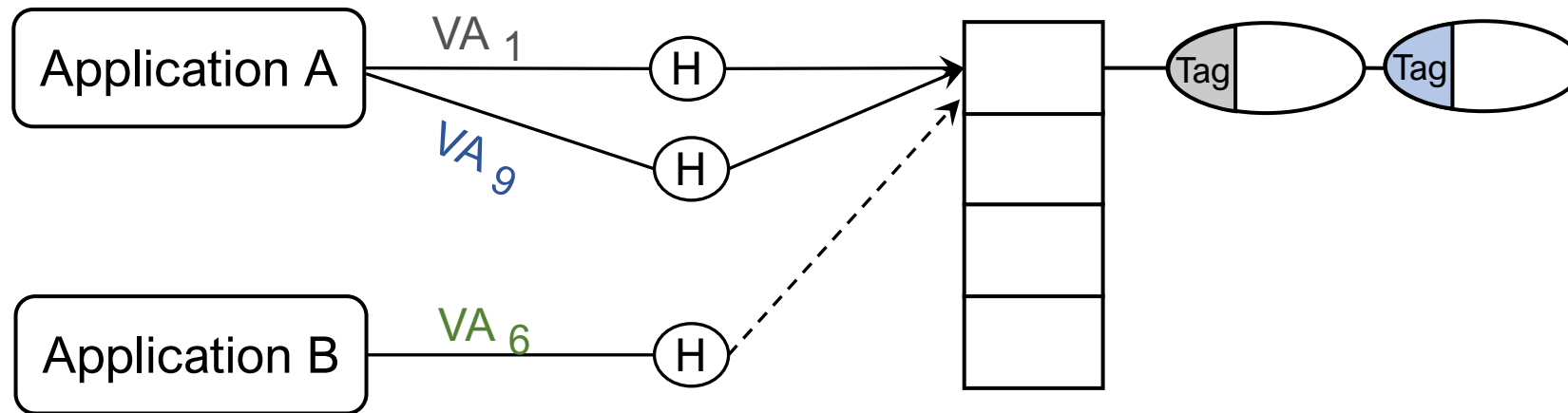
Alternative: A Global Hashed Page Table

The old approach from Intel and IBM

How to share pages?

Multiple page sizes?

Global Hash Table



Alternative: A Global Hashed Page Table

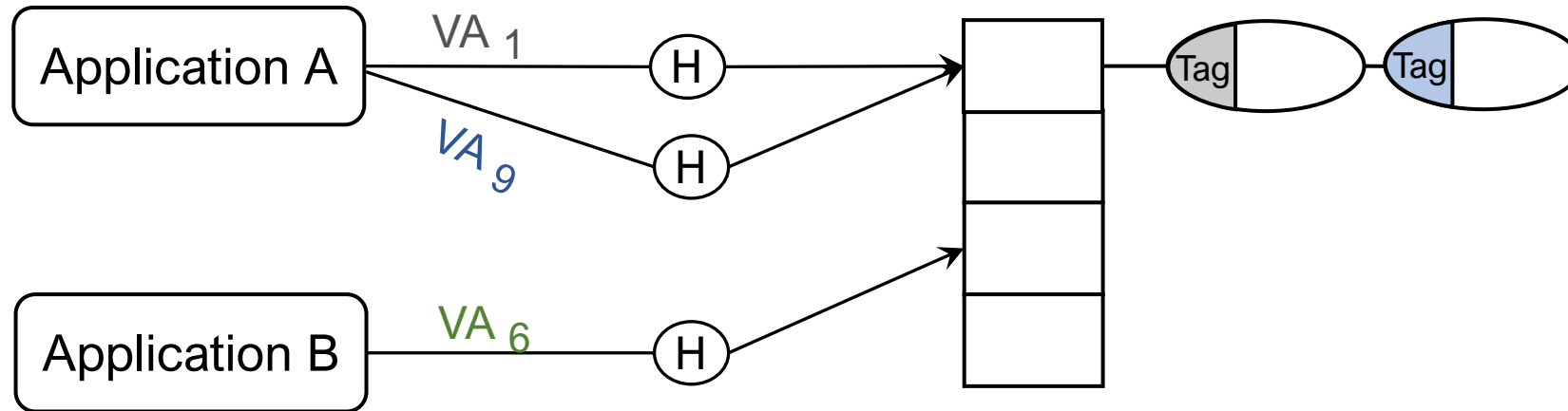
The old approach from Intel and IBM

How to share pages?

Multiple page sizes?

New level of indirection!!

Global Hash Table



Alternative: A Global Hashed Page Table

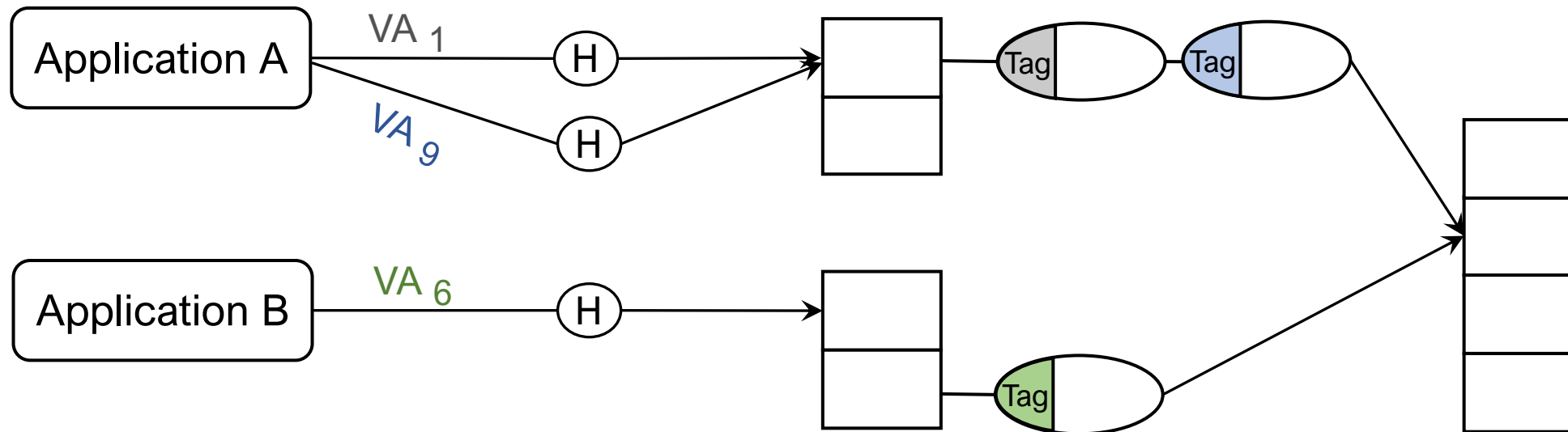
The old approach from Intel and IBM

How to share pages?

Multiple page sizes?

New level of indirection!!

Global Hash Table



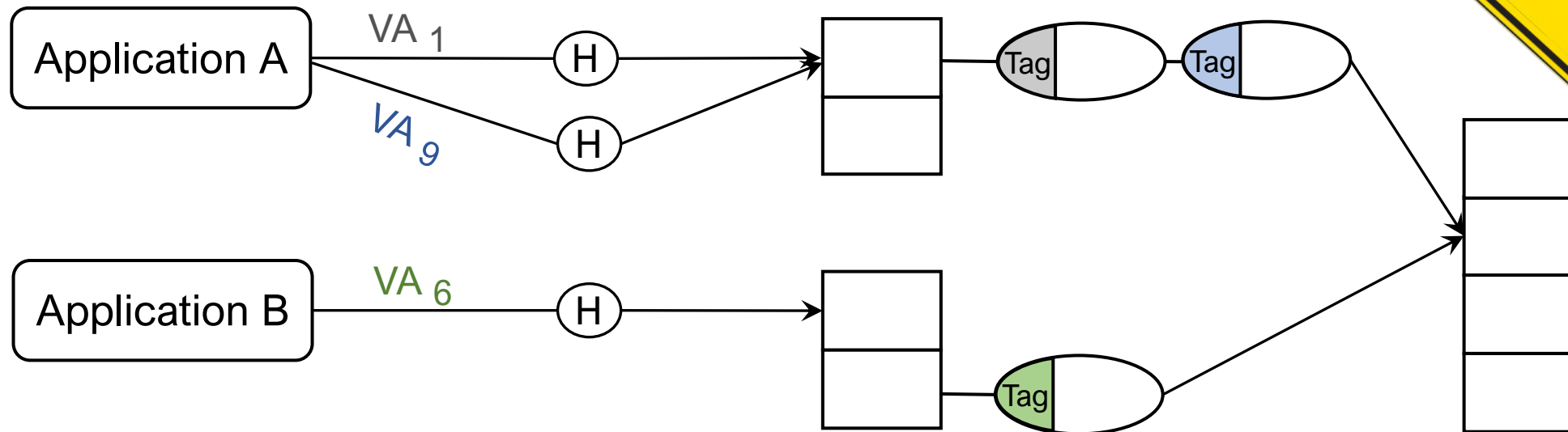
Alternative: A Global Hashed Page Table

The old approach from Intel and IBM

How to share pages?
Multiple page sizes?

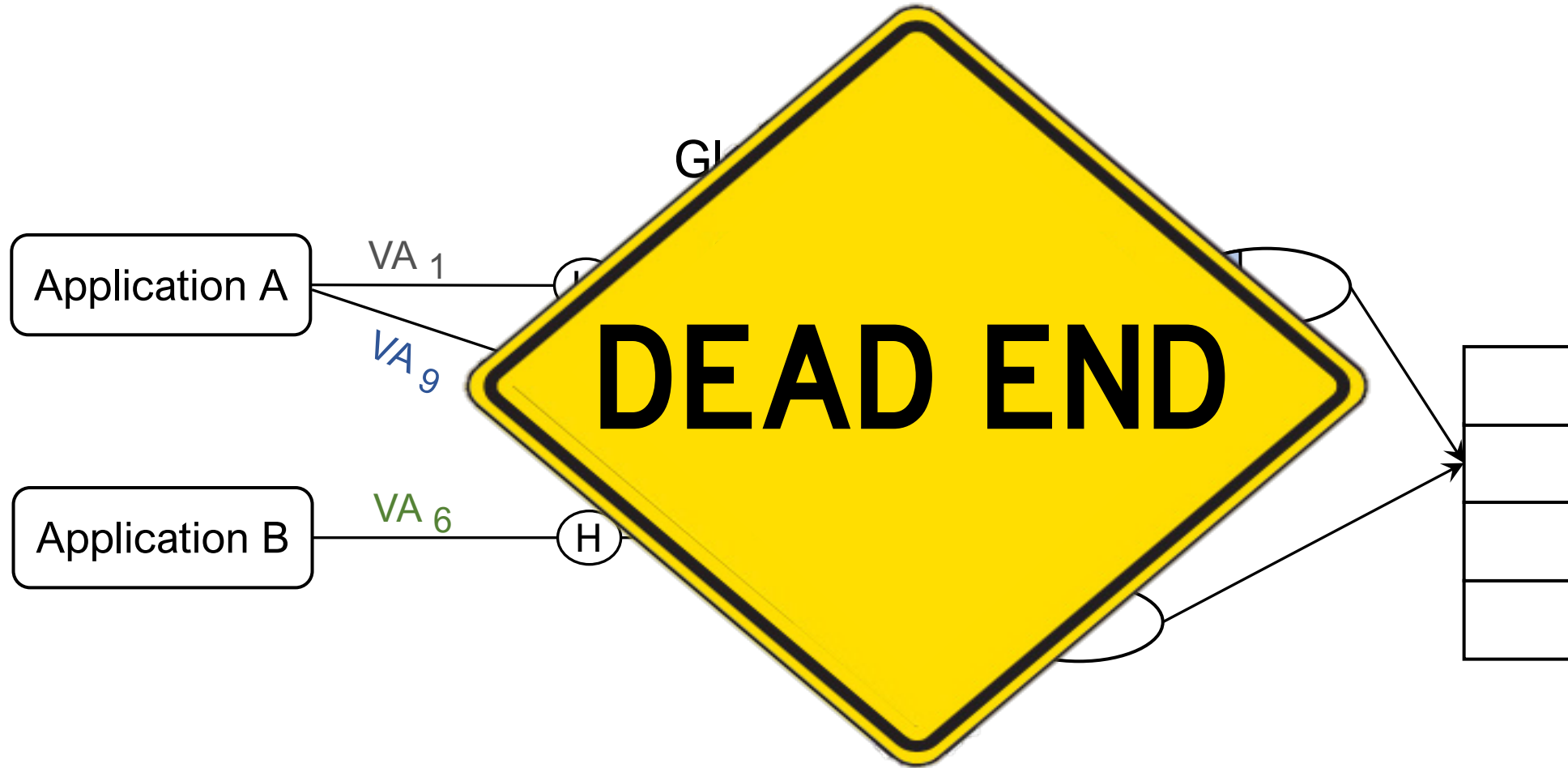
New level of indirection!!

Global Hash Table



Alternative: A Global Hashed Page Table

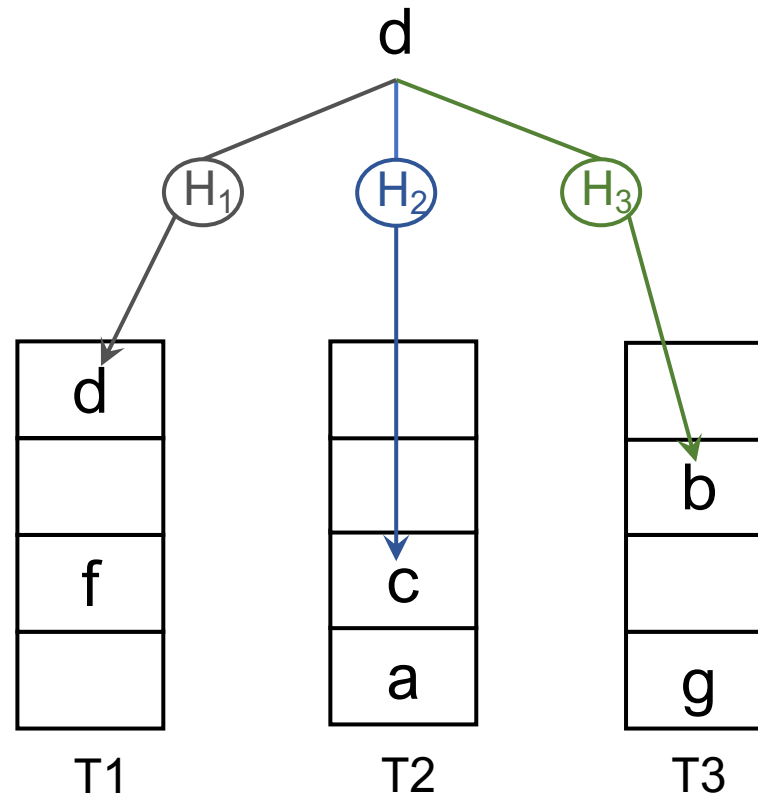
The old approach from Intel and IBM Switched to radix page tables!



Elastic Cuckoo Page Tables

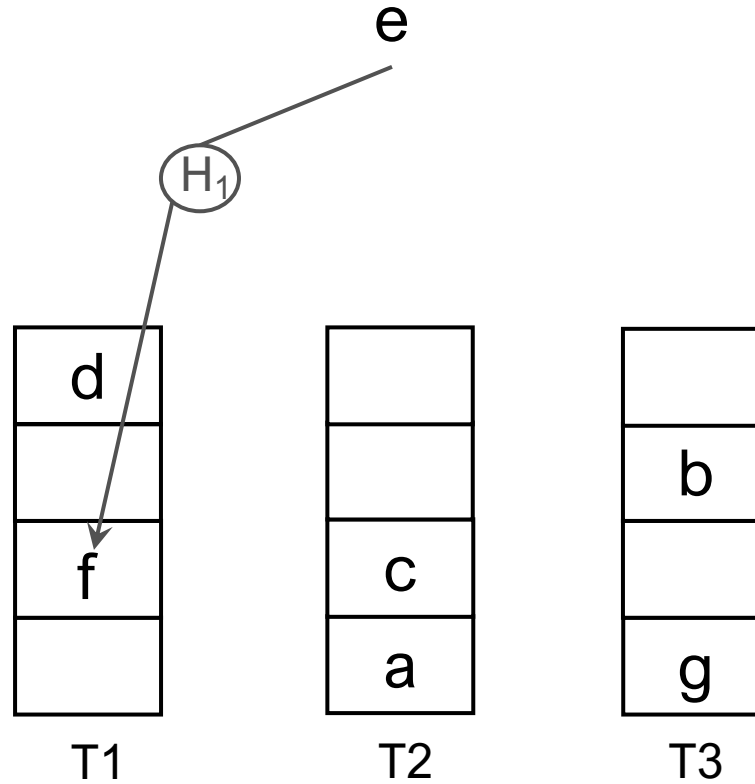
Rethinking virtual memory translation for parallelism

Cuckoo Hashing [Pagh 2001, Fotakis 2005]



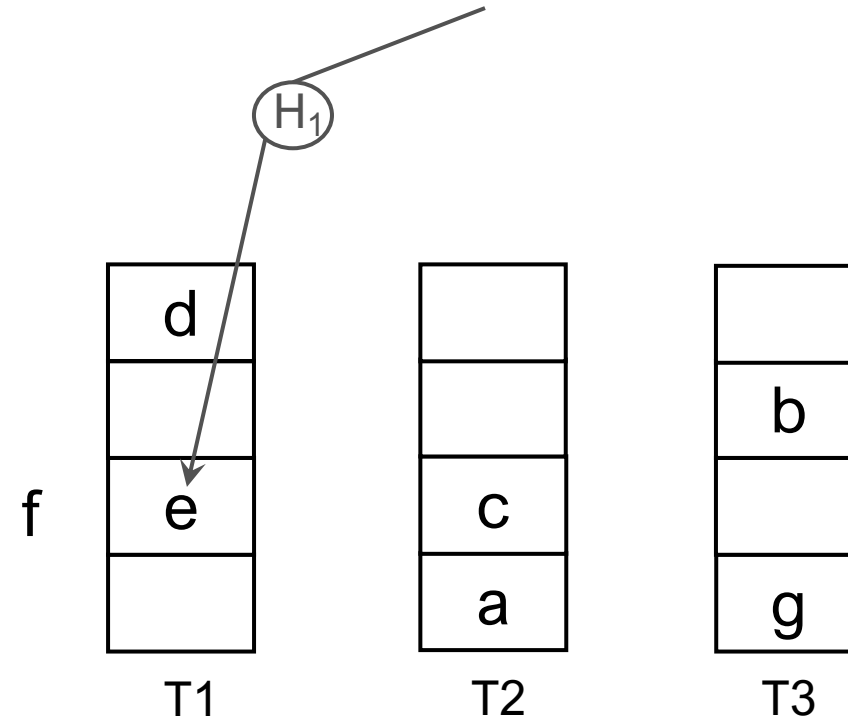
d -ary Cuckoo Hash Table

Insertions with Cuckoo Hashing



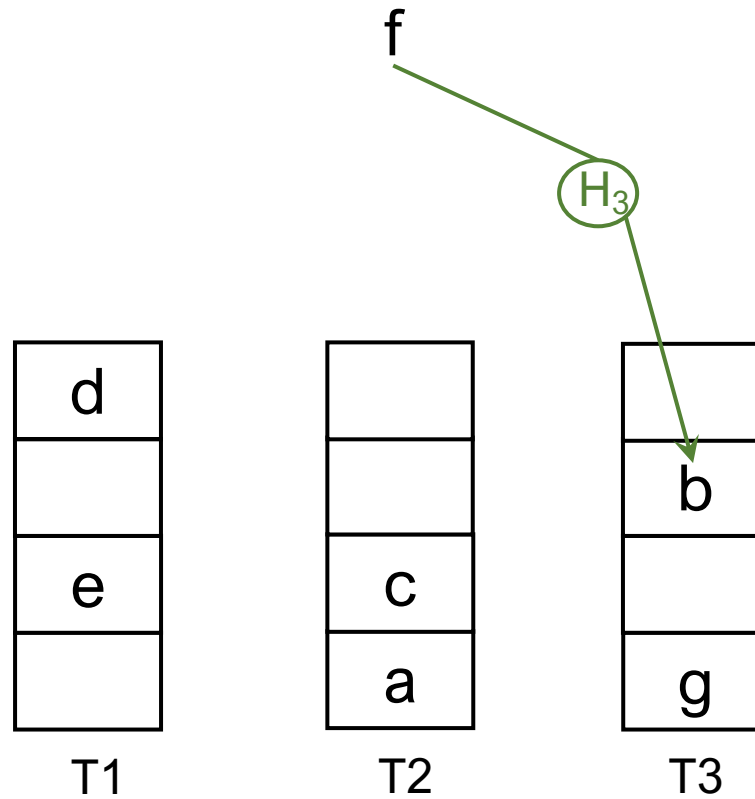
d -ary Cuckoo Hash Table

Insertions with Cuckoo Hashing



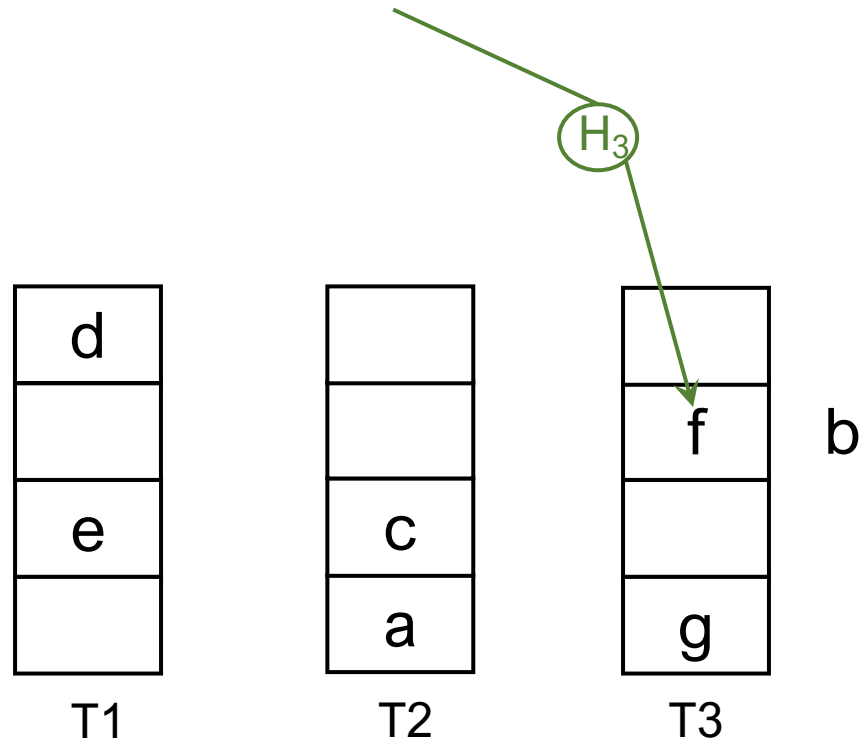
d -ary Cuckoo Hash Table

Insertions with Cuckoo Hashing



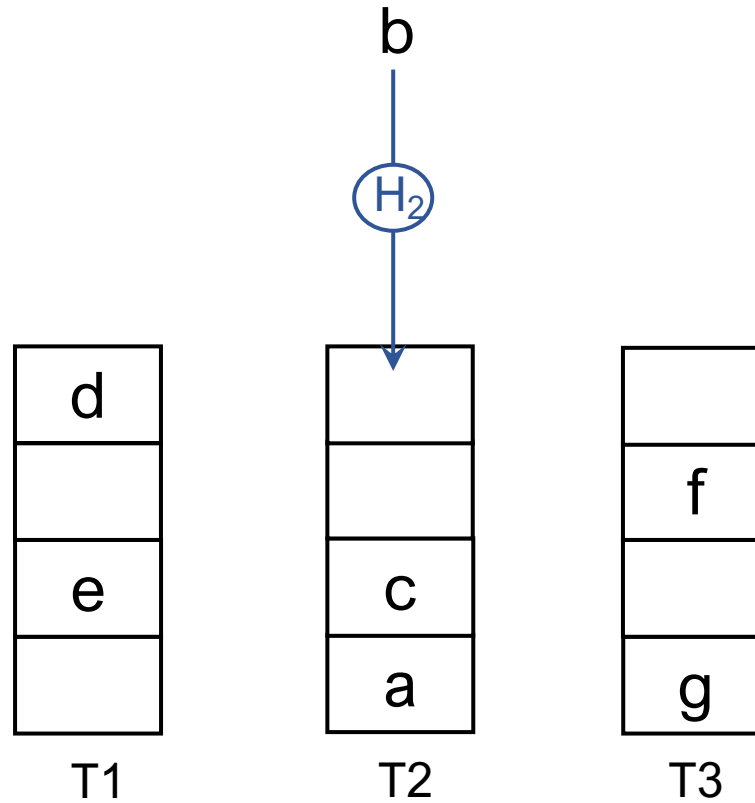
d -ary Cuckoo Hash Table

Insertions with Cuckoo Hashing



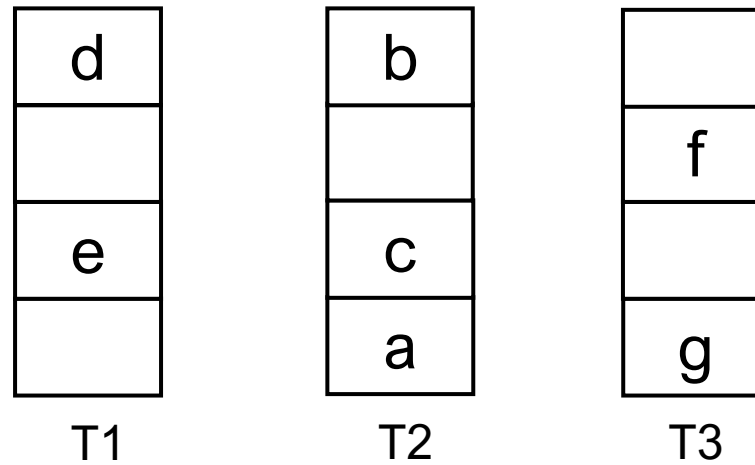
d -ary Cuckoo Hash Table

Insertions with Cuckoo Hashing



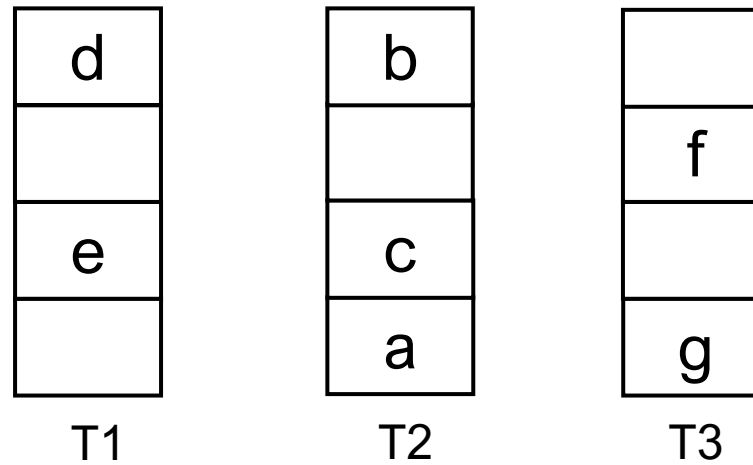
d -ary Cuckoo Hash Table

Insertions with Cuckoo Hashing



d -ary Cuckoo Hash Table

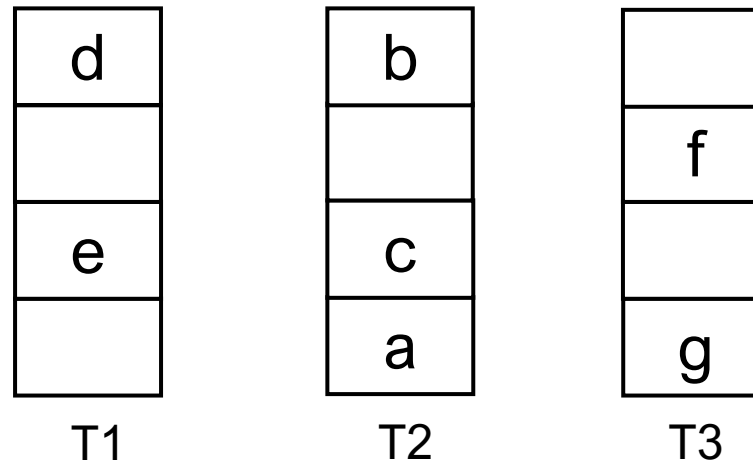
Insertions with Cuckoo Hashing



d-ary Cuckoo Hash Table



Private Hashed Page Tables



d-ary Cuckoo Hash Table



Cannot Be Too Big → Waste Memory



App A

Main Memory

Page Tables A

Page Tables B

App B



Private page tables cannot be too big

Need to Dynamically Resize



App A

Main Memory

Page Tables A

VA 1

PA 4

VA 8

PA 1

Private page tables cannot be too big



Need to dynamically resize



Need to Dynamically Resize



App A

Main Memory

Page Tables A

VA 1	PA 4
VA 8	PA 1

New Page Tables A

Private page tables cannot be too big



Need to dynamically resize



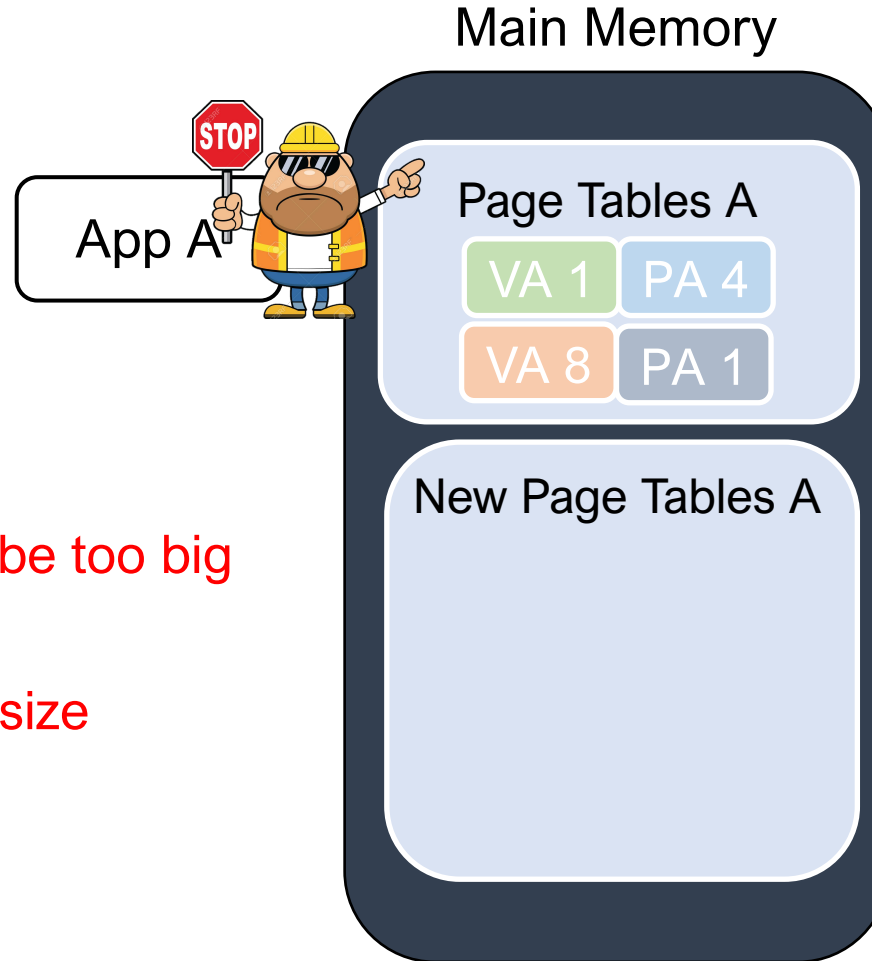
Cannot Rehash All Entries at Once



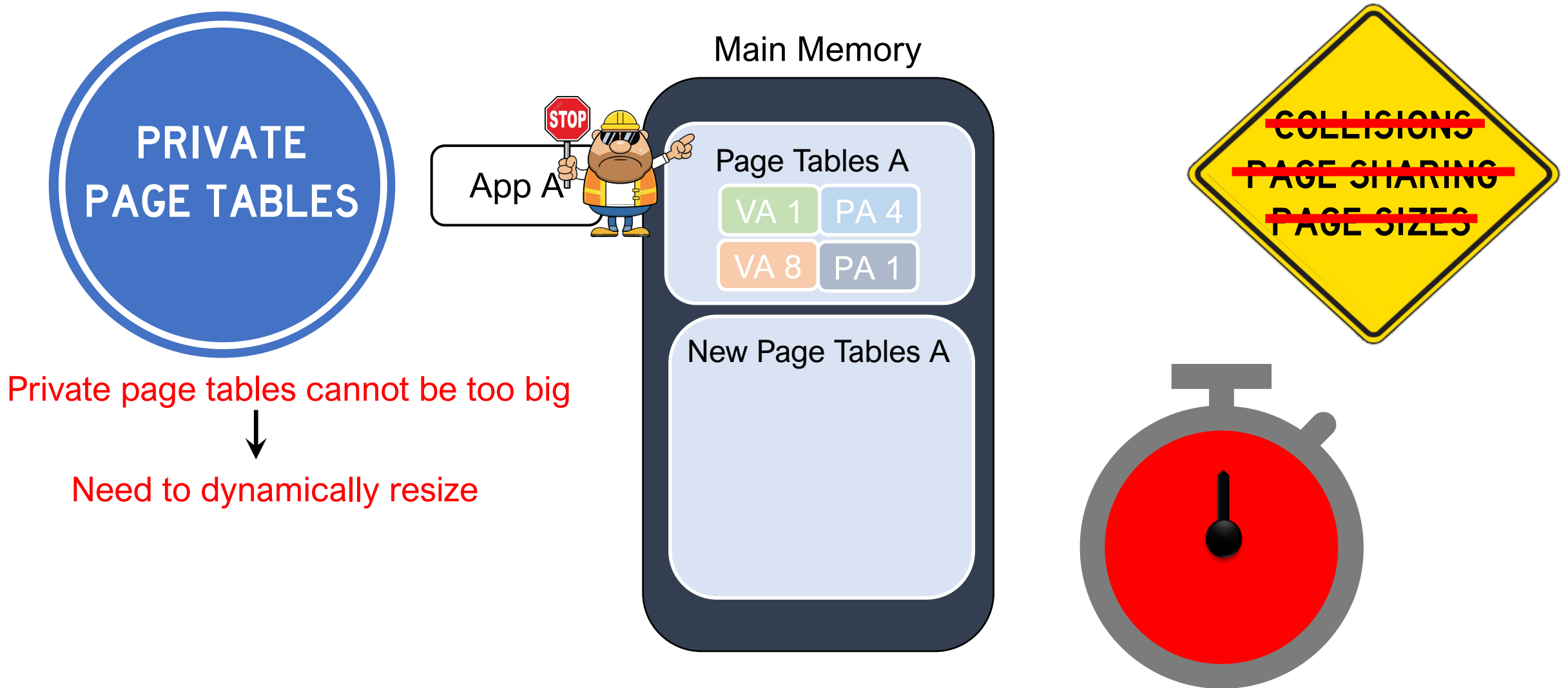
Private page tables cannot be too big



Need to dynamically resize



Cannot Rehash All Entries at Once



Cannot Rehash All Entries at Once



App A

Main Memory

Page Tables A

VA 1	PA 4
VA 8	PA 1

New Page Tables A



Private page tables cannot be too big



Need to dynamically resize



While the program is running



Gradual Resizing!

Gradual Resizing Cuckoo Hash Tables

At every insert $\rightarrow$ Rehash one element

m	b	
k		f
e	c	l
	a	g

T1

T2

T3

Old d -ary Cuckoo Hash Table

T1'

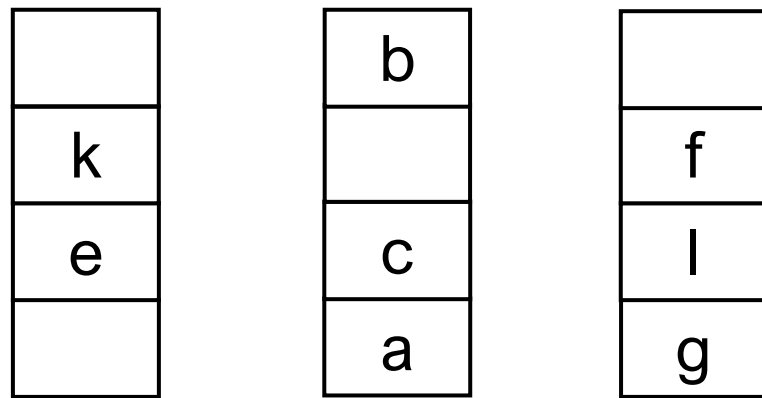
T2'

T3'

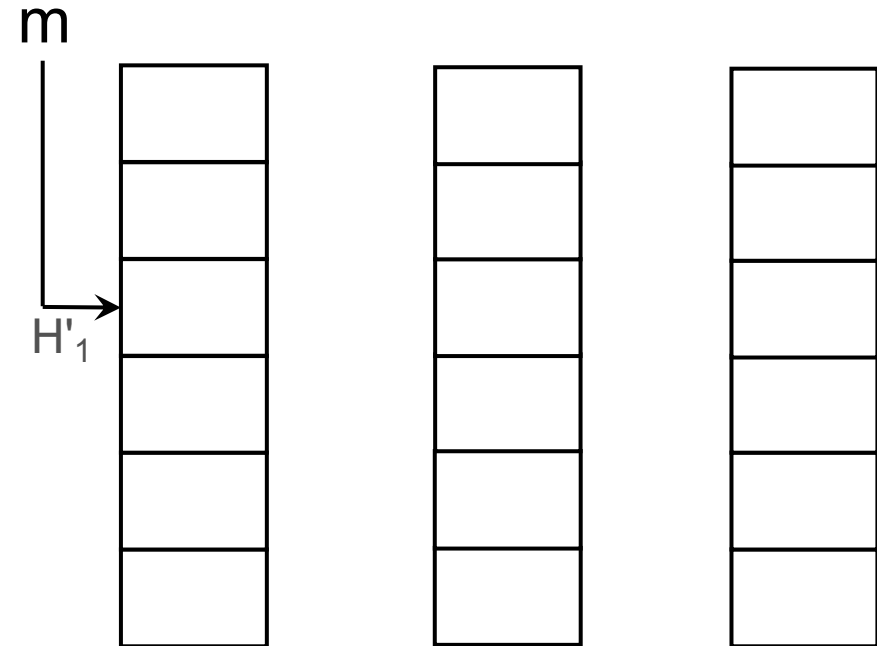
New d -ary Cuckoo Hash Table

Gradual Resizing Cuckoo Hash Tables

At every insert $\rightarrow$ Rehash one element

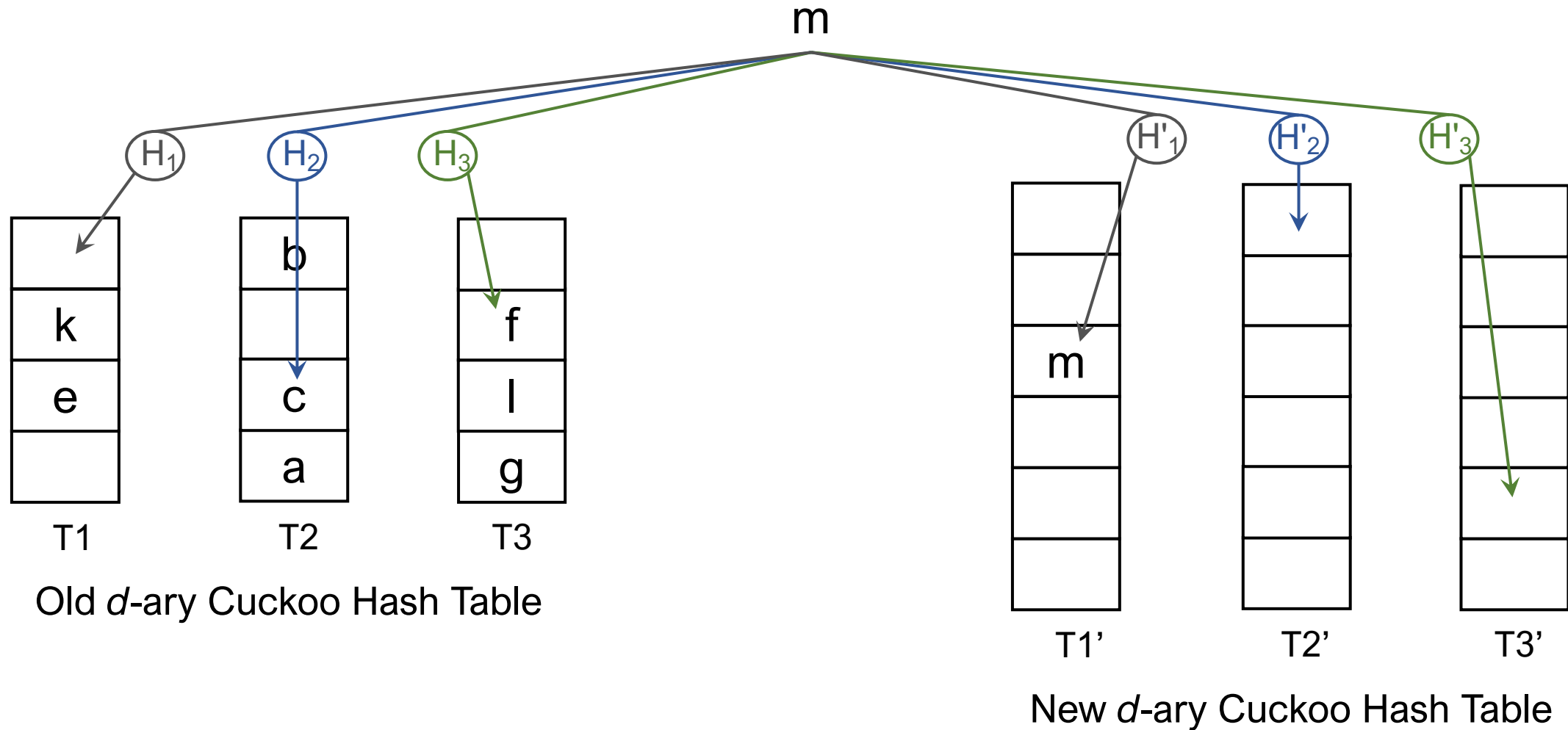


T1 T2 T3
Old d -ary Cuckoo Hash Table

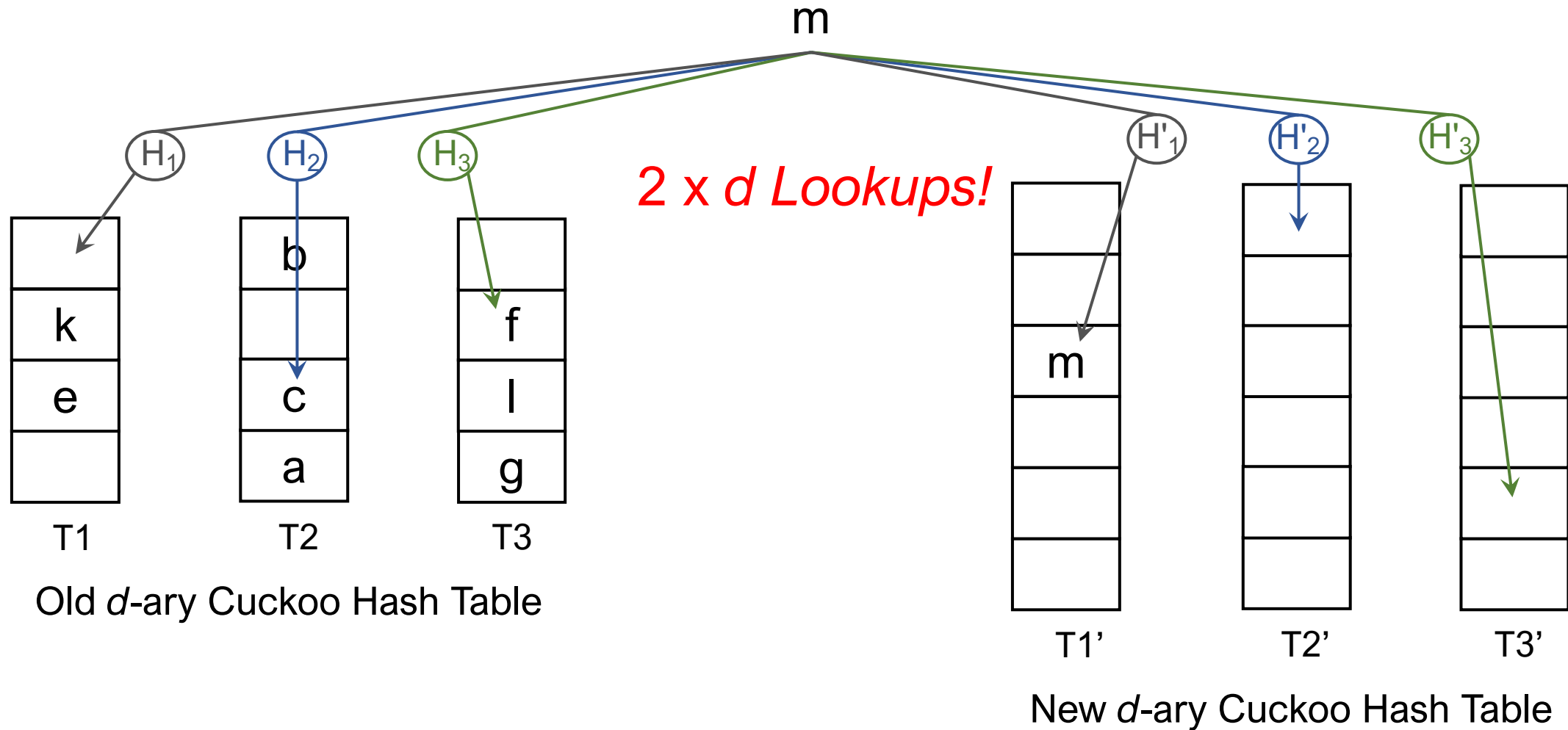


T1' T2' T3'
New d -ary Cuckoo Hash Table

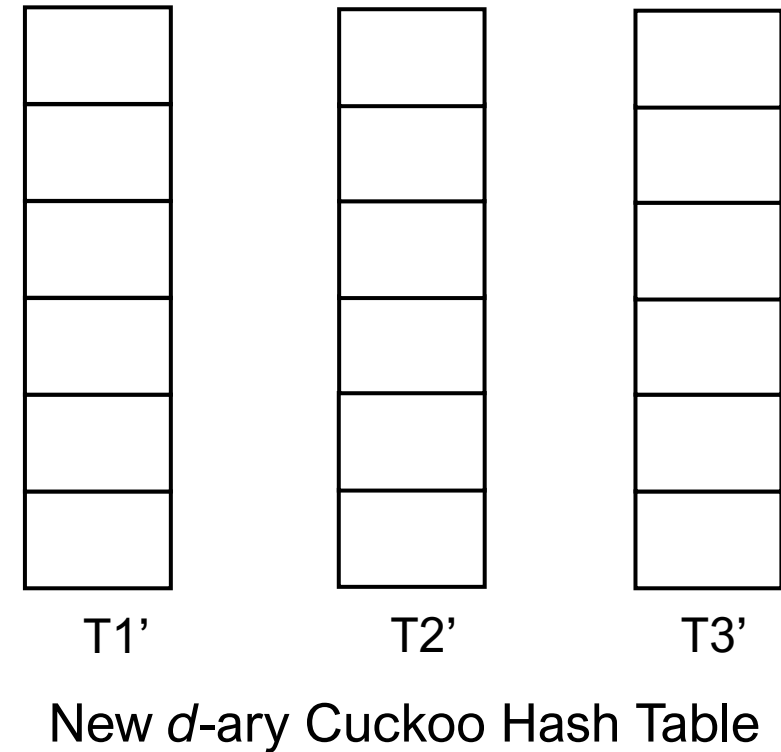
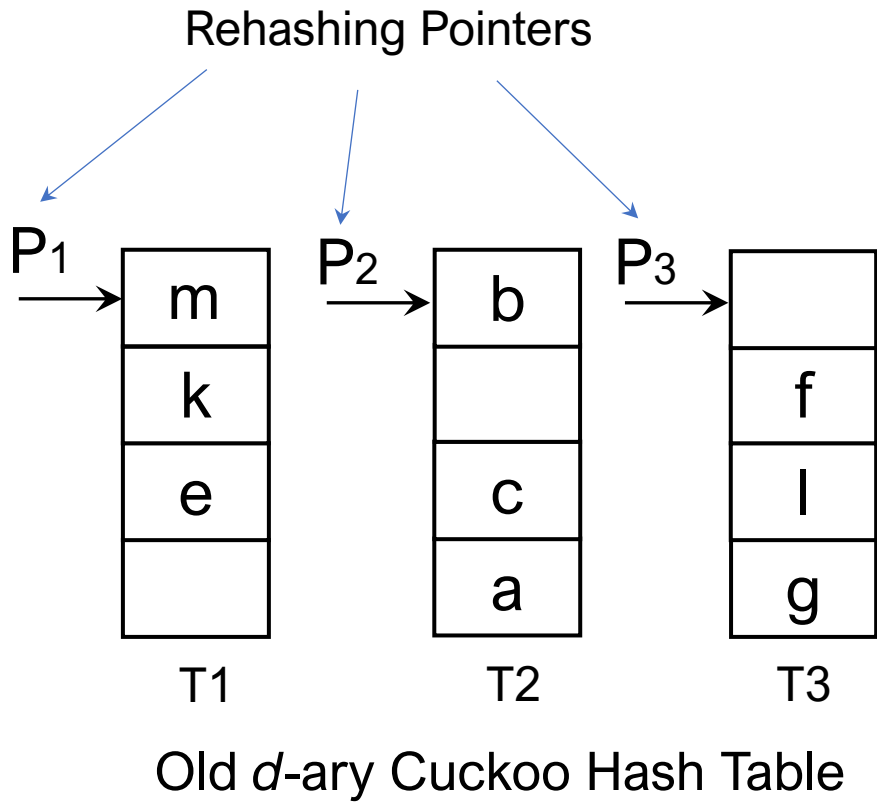
Lookup During Gradual Resizing



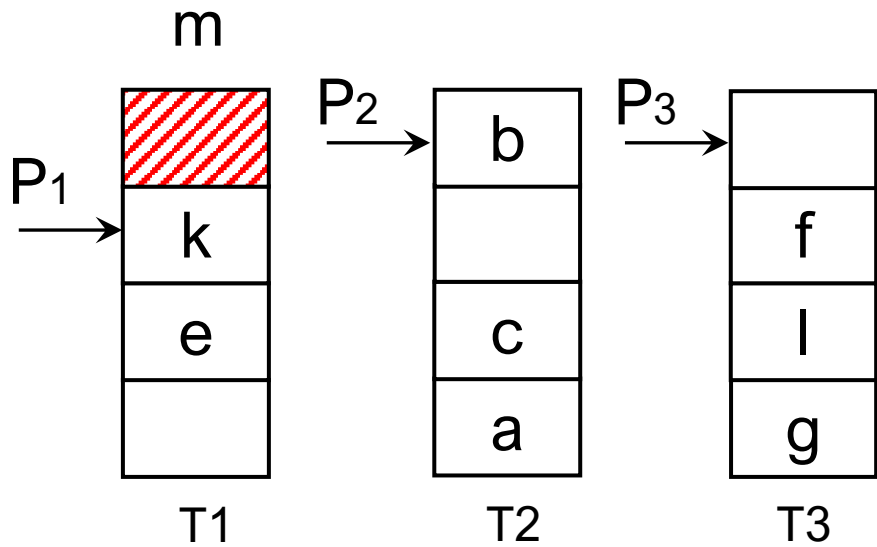
Problem of Resizing: Double #Lookups



Contribution: Elastic Cuckoo Hashing

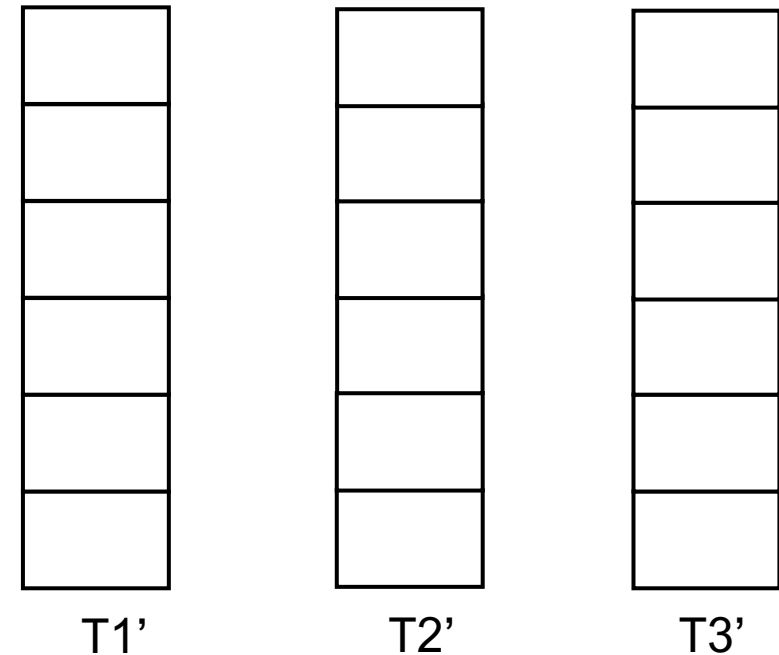


Elastic Cuckoo Migration



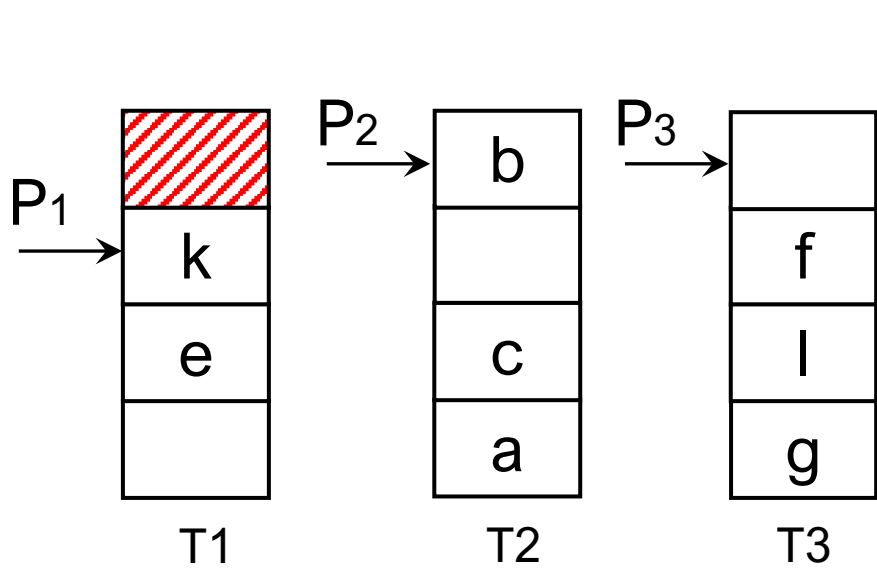
Old d -ary Cuckoo Hash Table

 Migrated Region



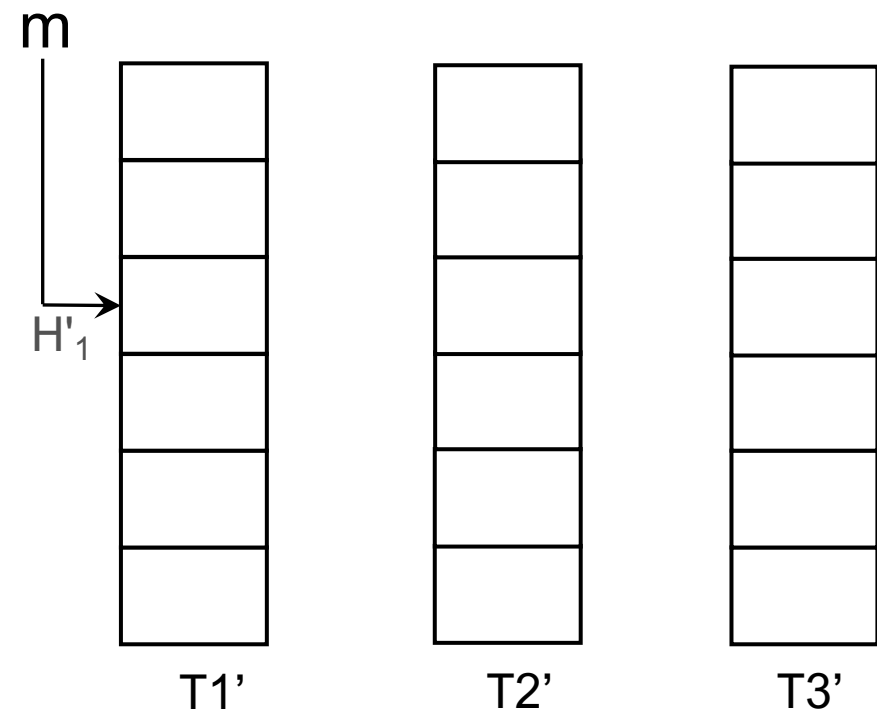
New d -ary Cuckoo Hash Table

Elastic Cuckoo Migration



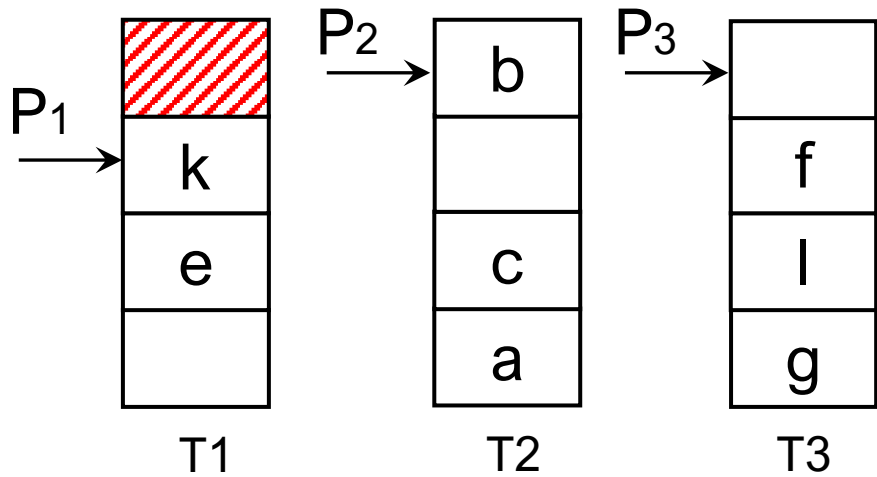
Old d -ary Cuckoo Hash Table

 Migrated Region



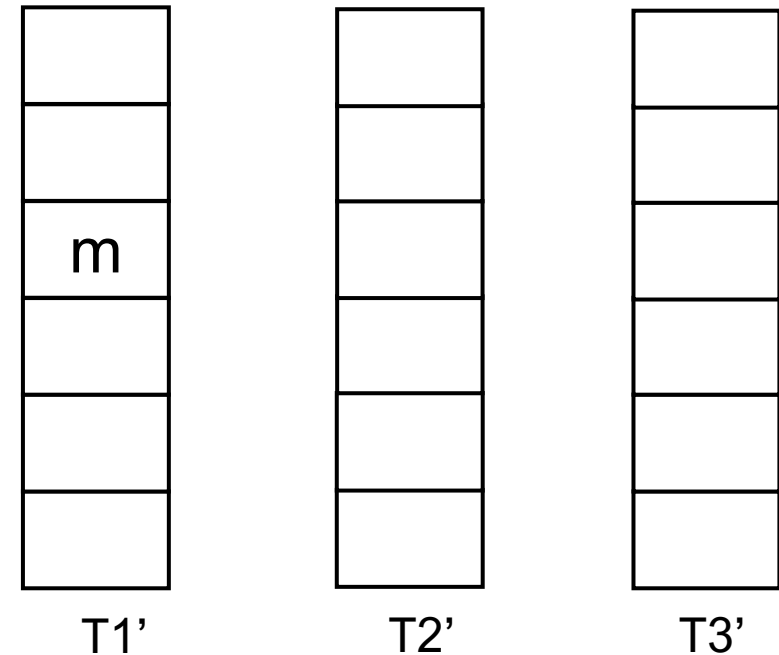
New d -ary Cuckoo Hash Table

Elastic Cuckoo Migration



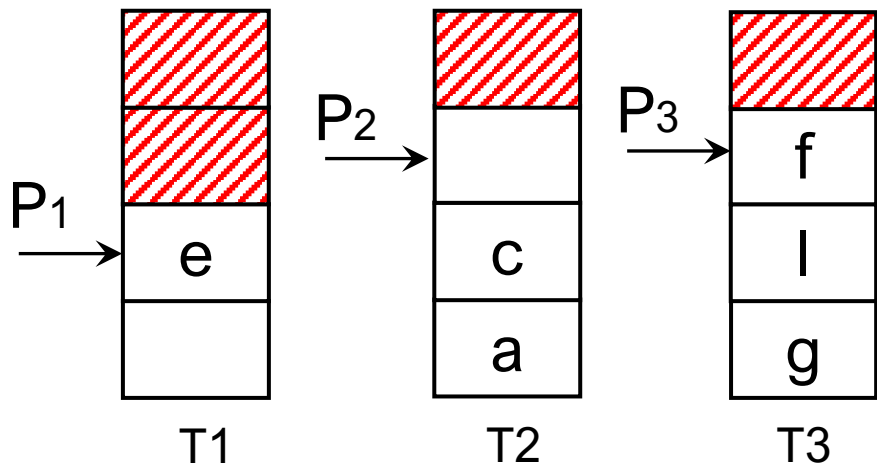
Old d -ary Cuckoo Hash Table

 Migrated Region



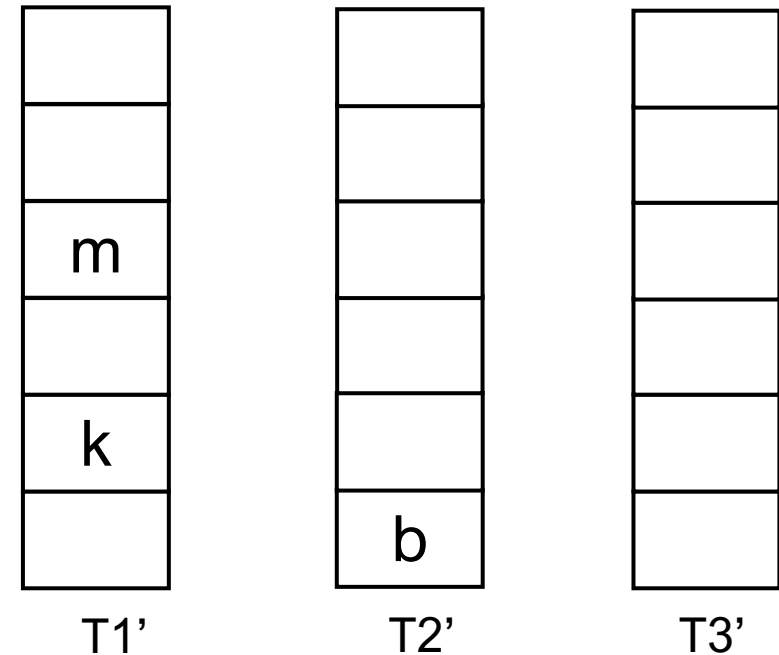
New d -ary Cuckoo Hash Table

Elastic Cuckoo Migration



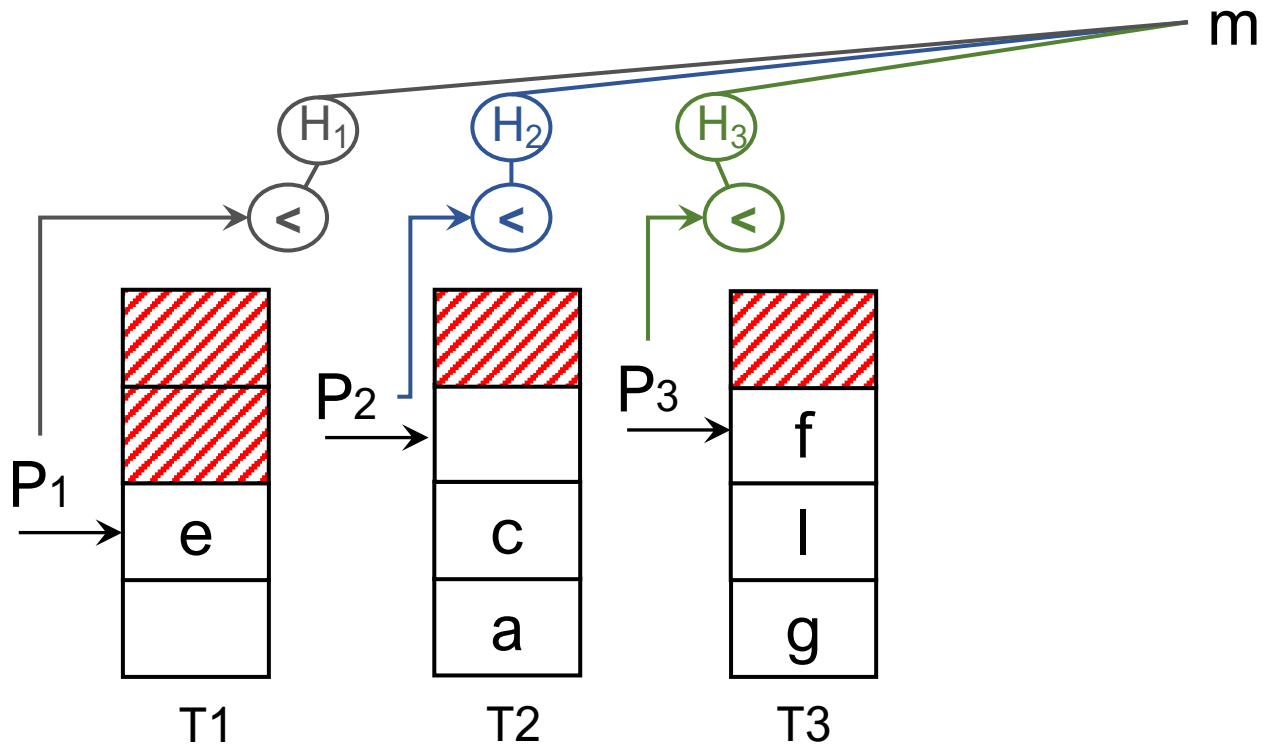
Old d -ary Cuckoo Hash Table

 Migrated Region



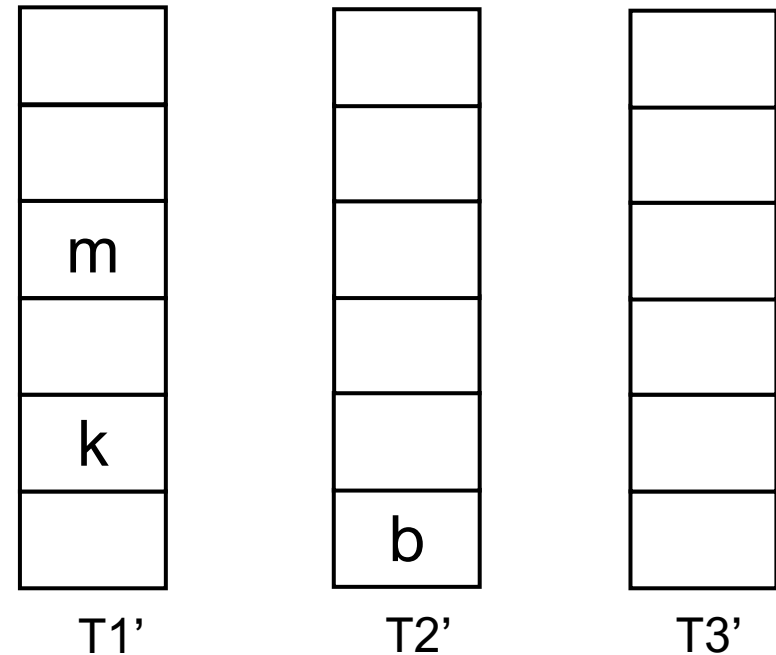
New d -ary Cuckoo Hash Table

Elastic Cuckoo Lookup



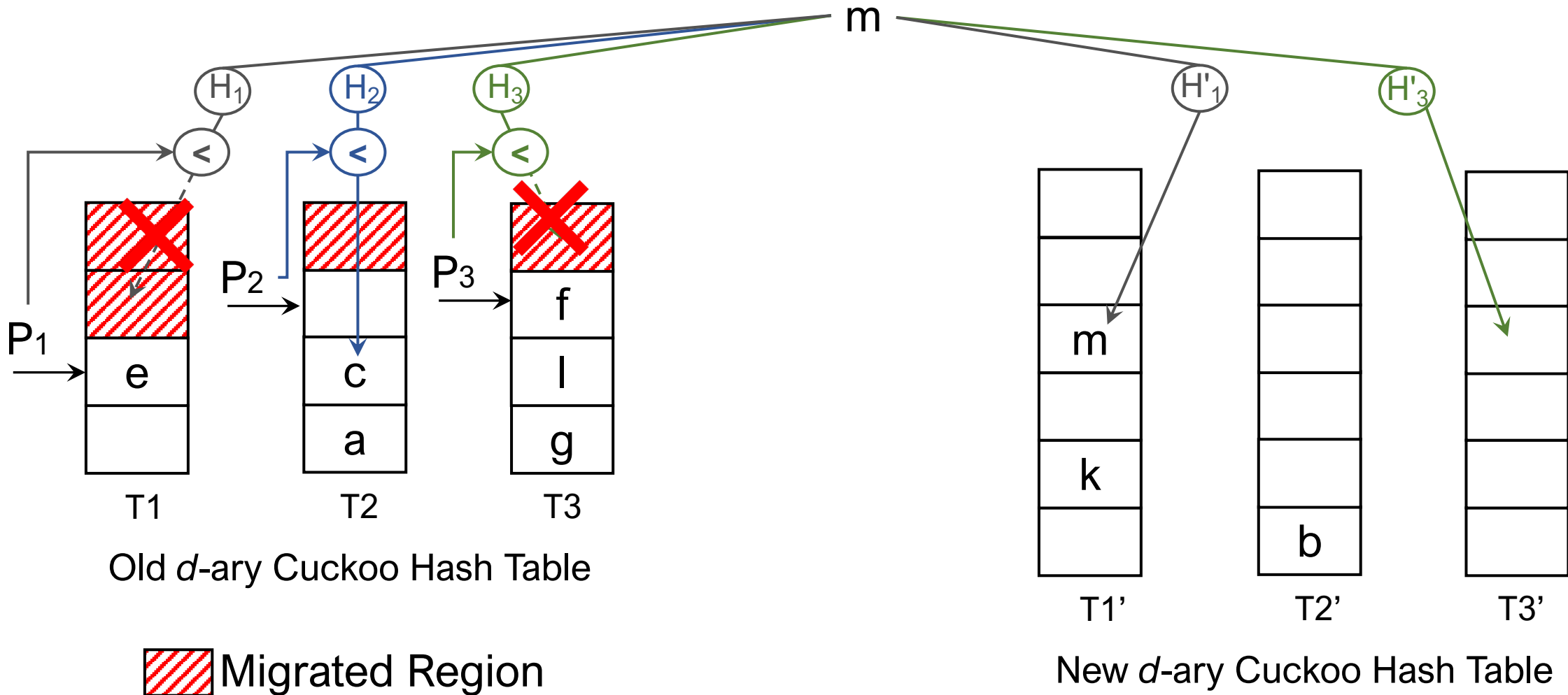
Old d -ary Cuckoo Hash Table

 Migrated Region

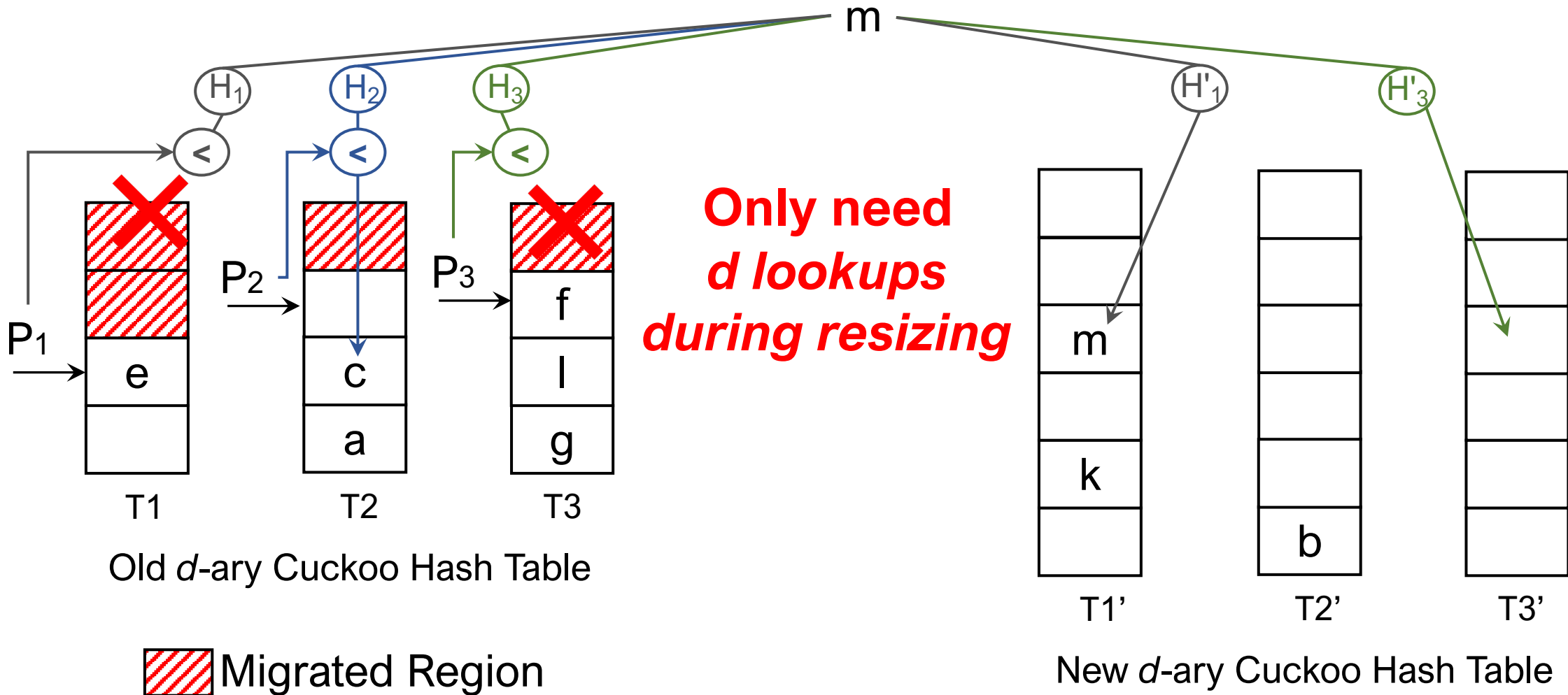


New d -ary Cuckoo Hash Table

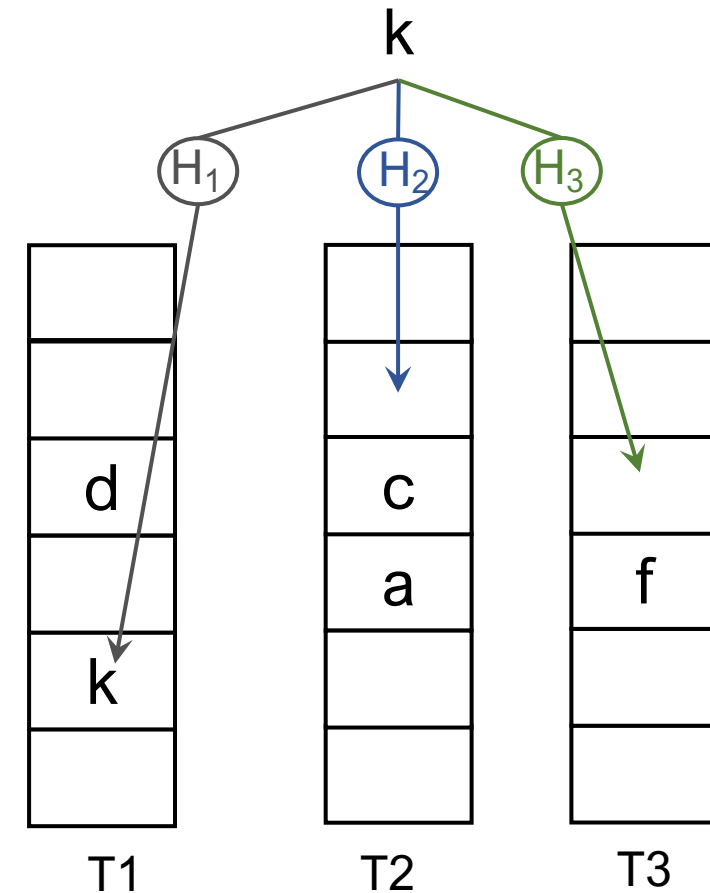
Elastic Cuckoo Lookup



Elastic Cuckoo Lookup

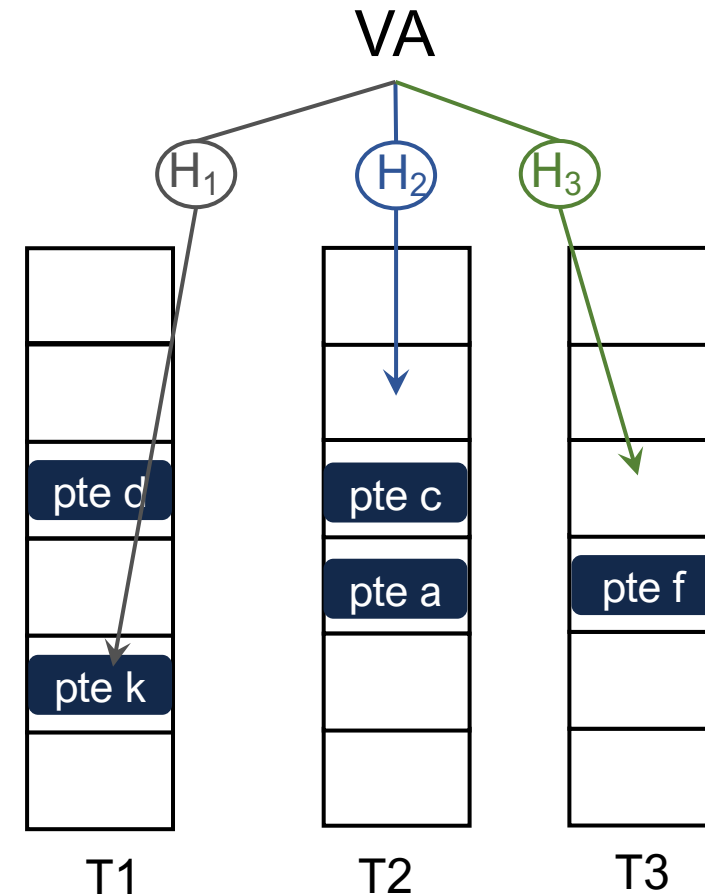


Exploiting Parallelism in Virtual Translation



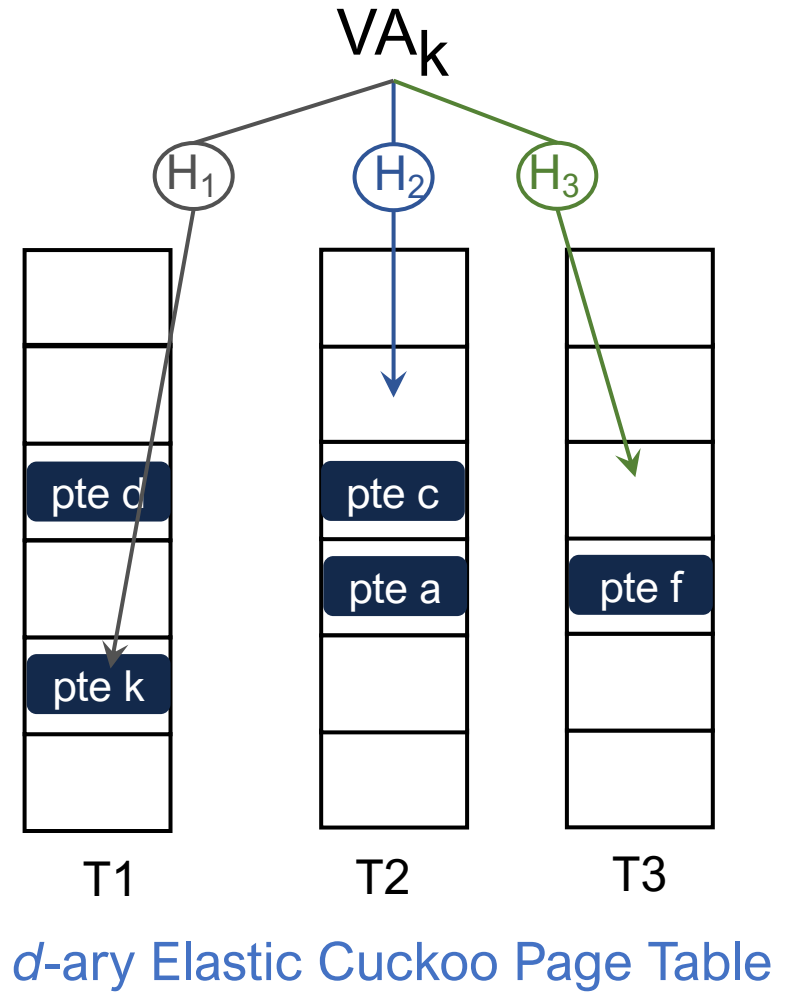
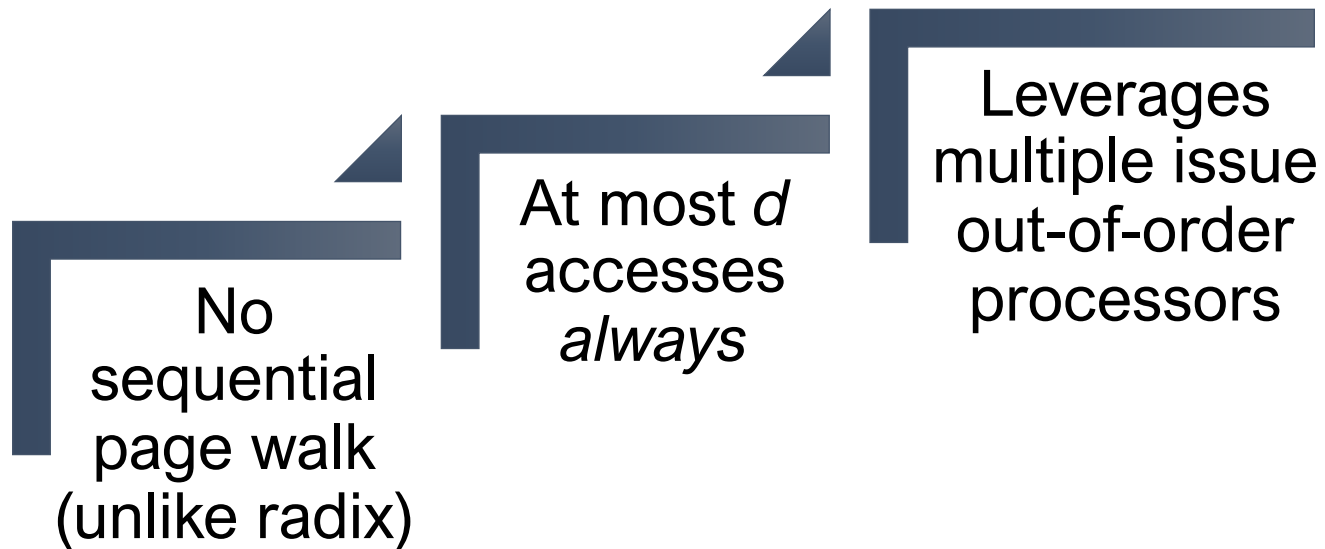
d -ary Elastic Cuckoo Hash Table

Exploiting Parallelism in Virtual Translation

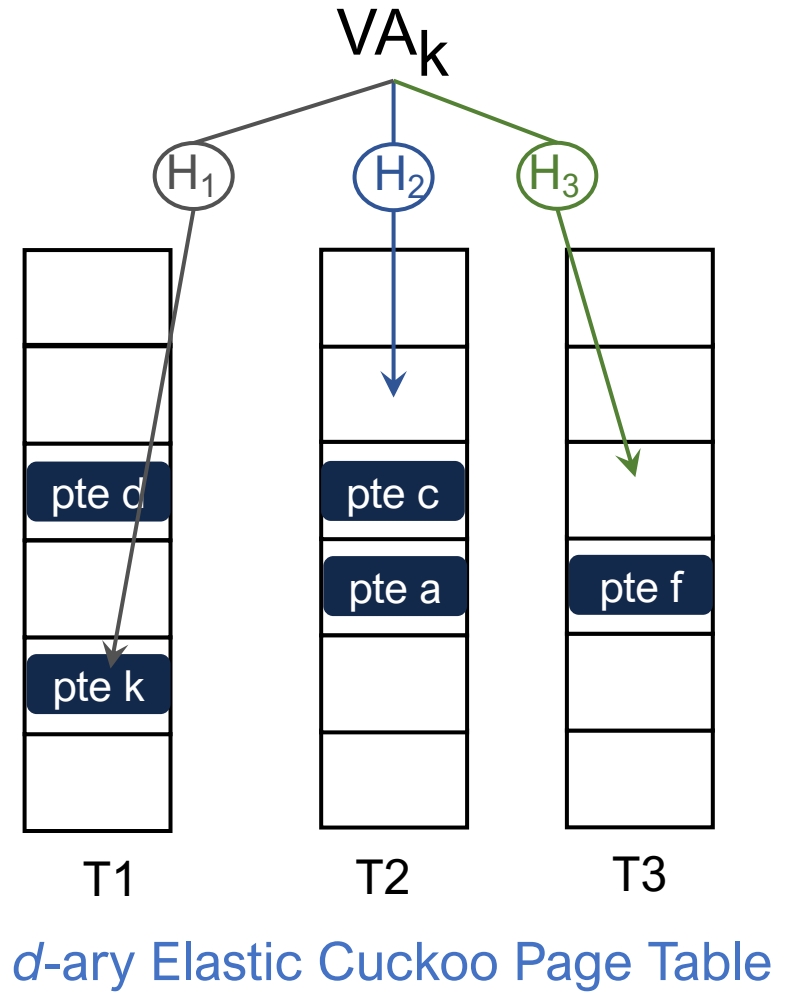
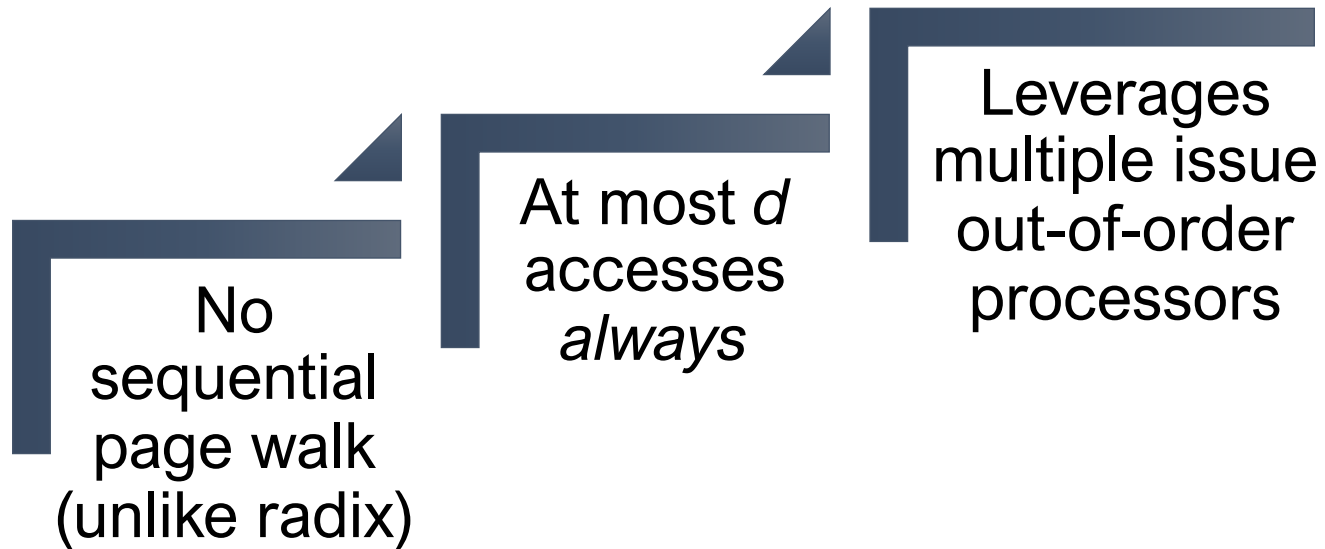


d -ary Elastic Cuckoo Page Table

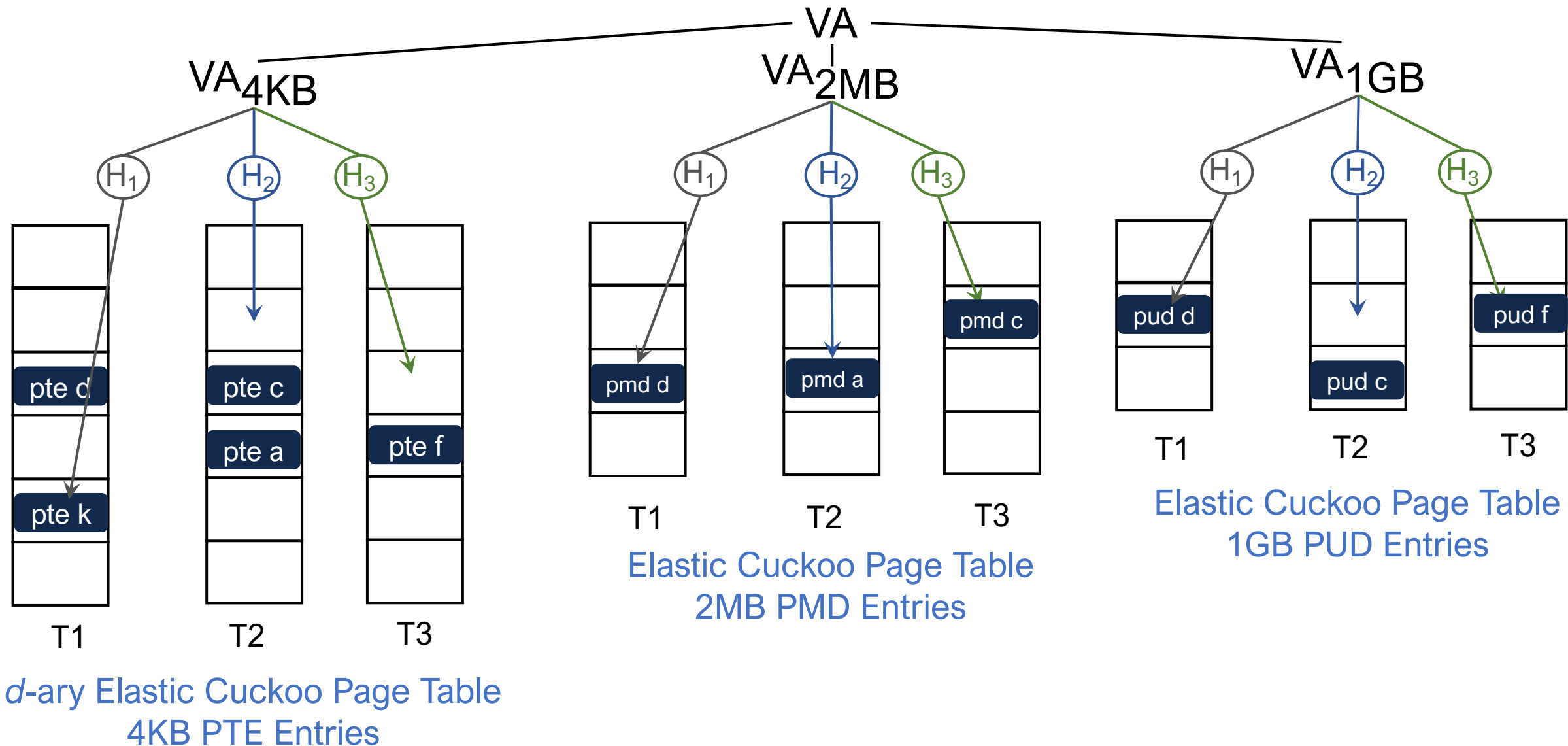
Exploiting Parallelism in Virtual Translation



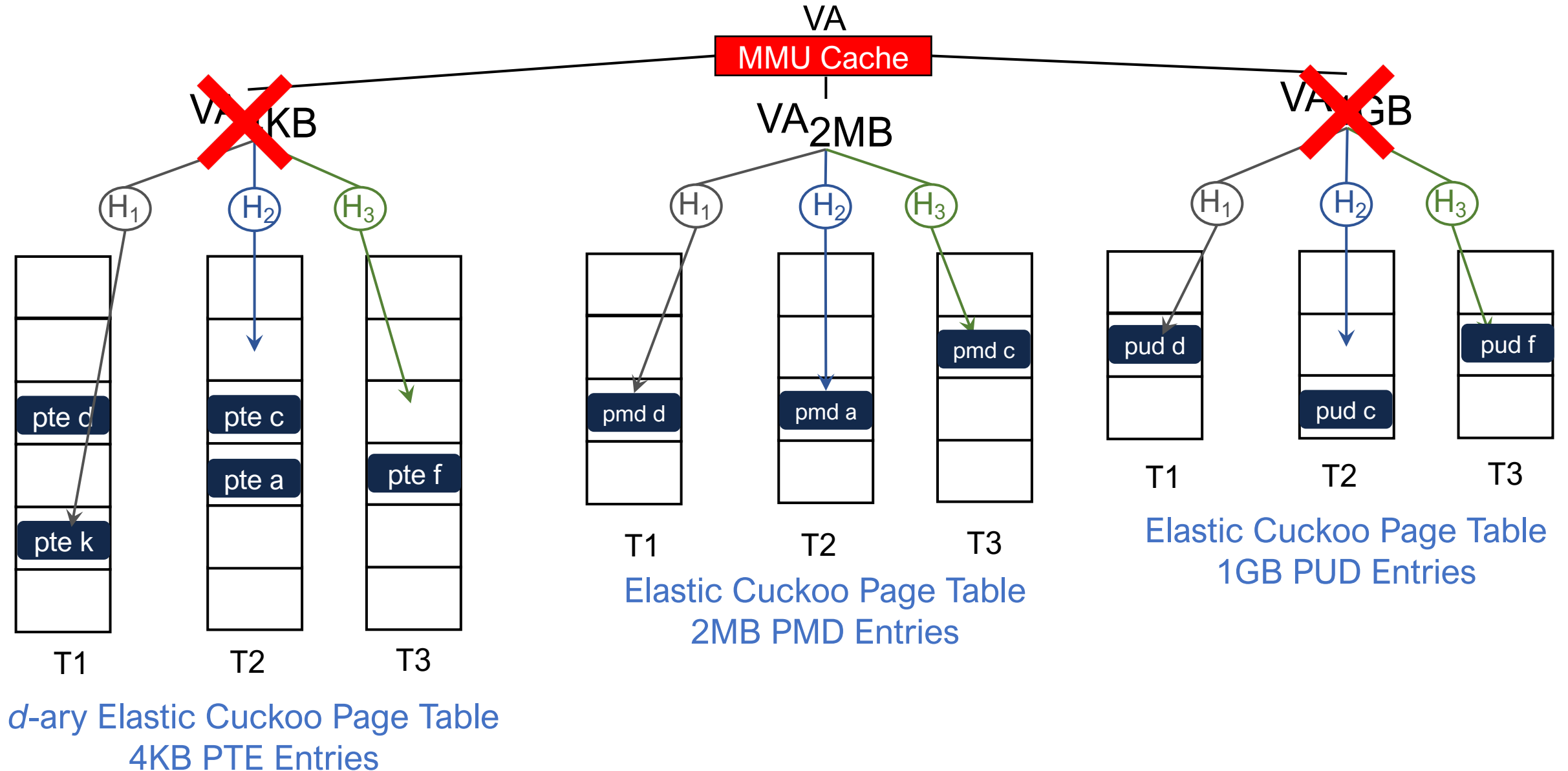
Exploiting Parallelism in Virtual Translation



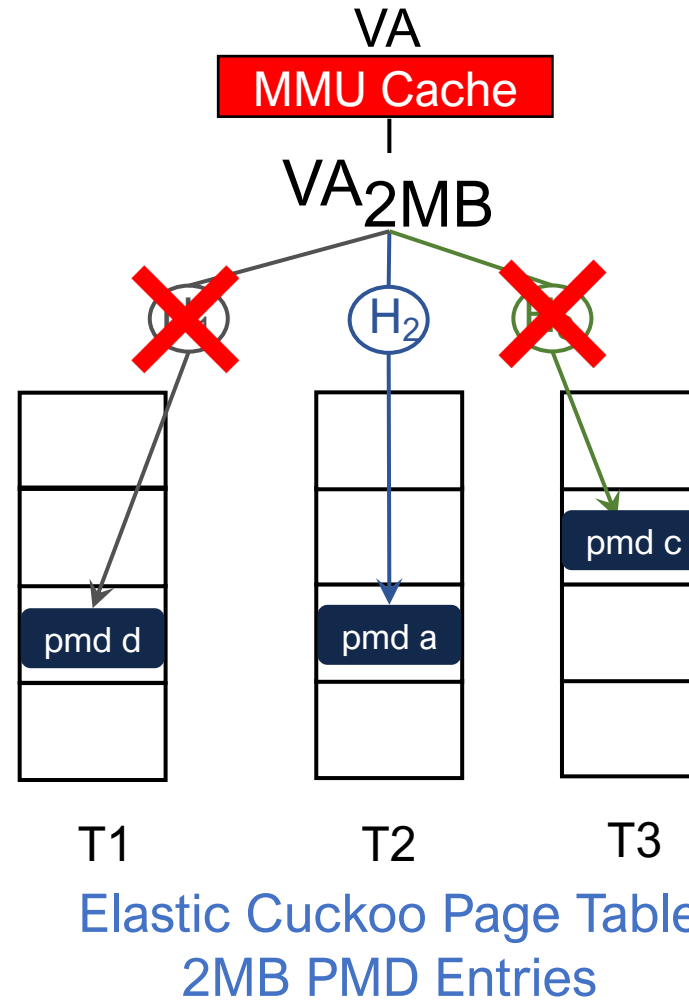
Lookup Multiple Page Sizes in Parallel



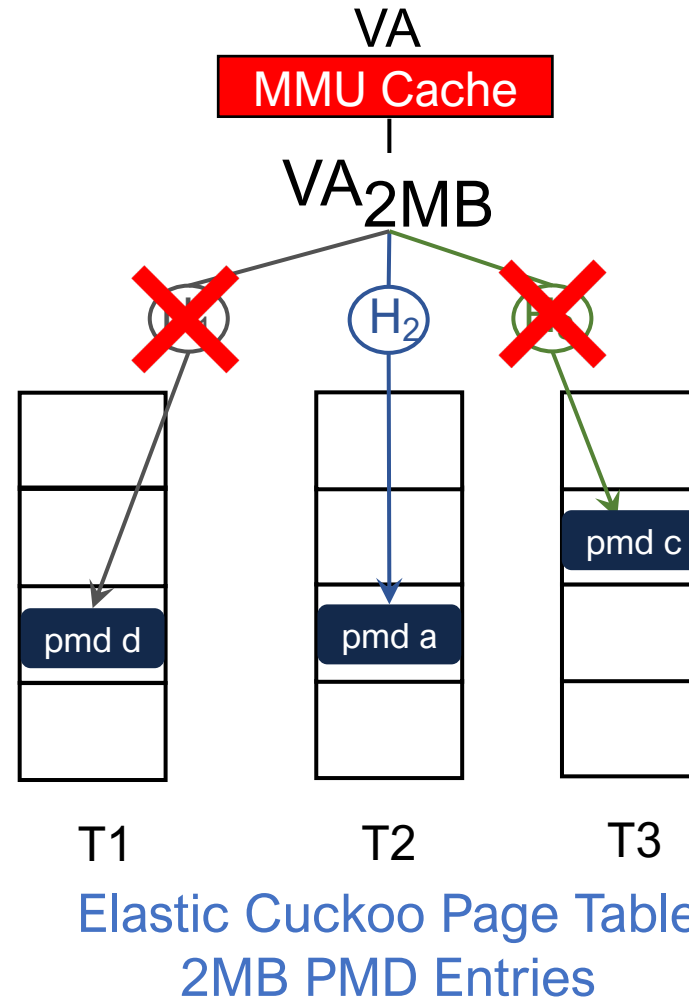
New MMU Cache to Prune Parallelism



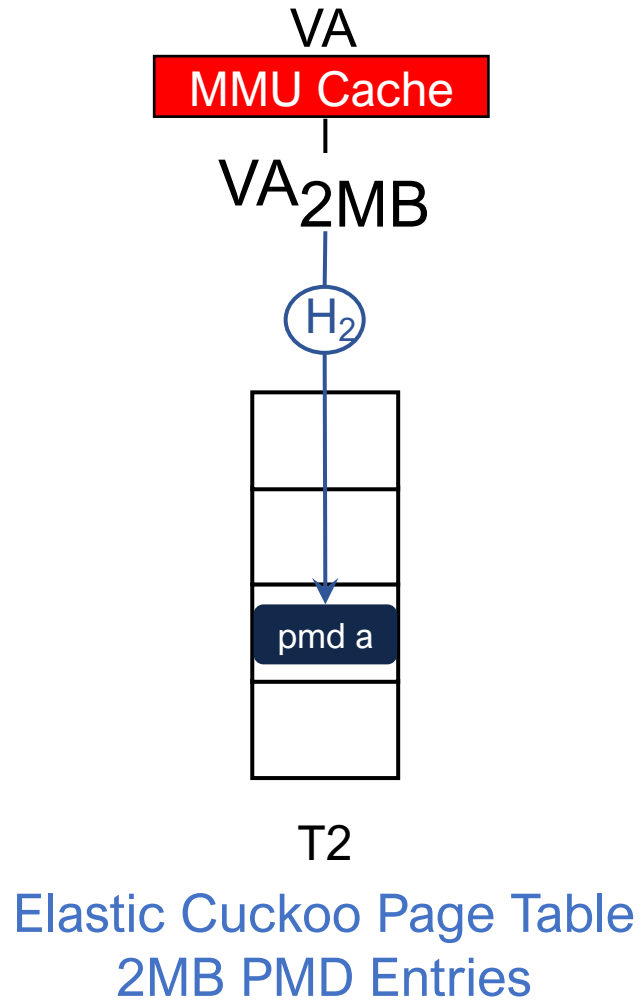
New MMU Cache to Prune Parallelism



New MMU Cache to Prune Parallelism



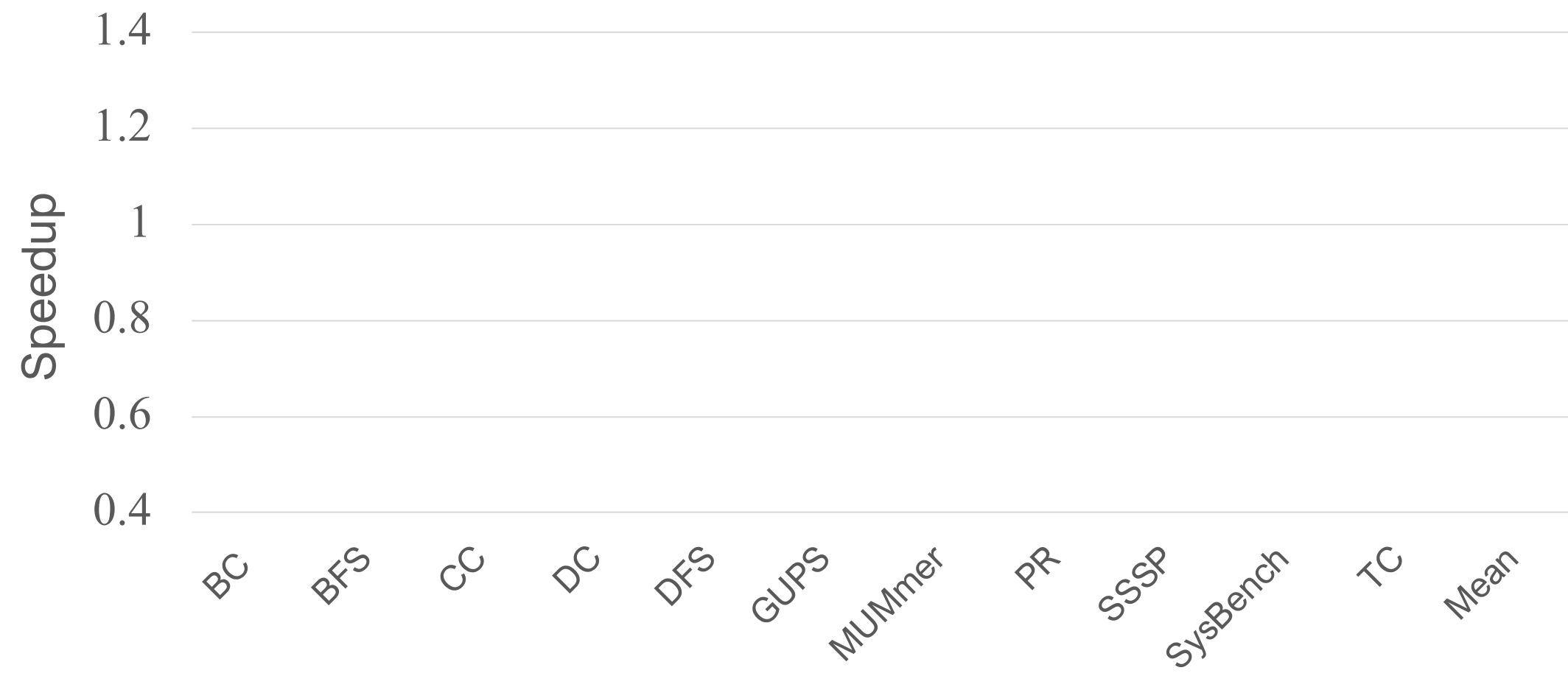
New MMU Cache to Prune Parallelism



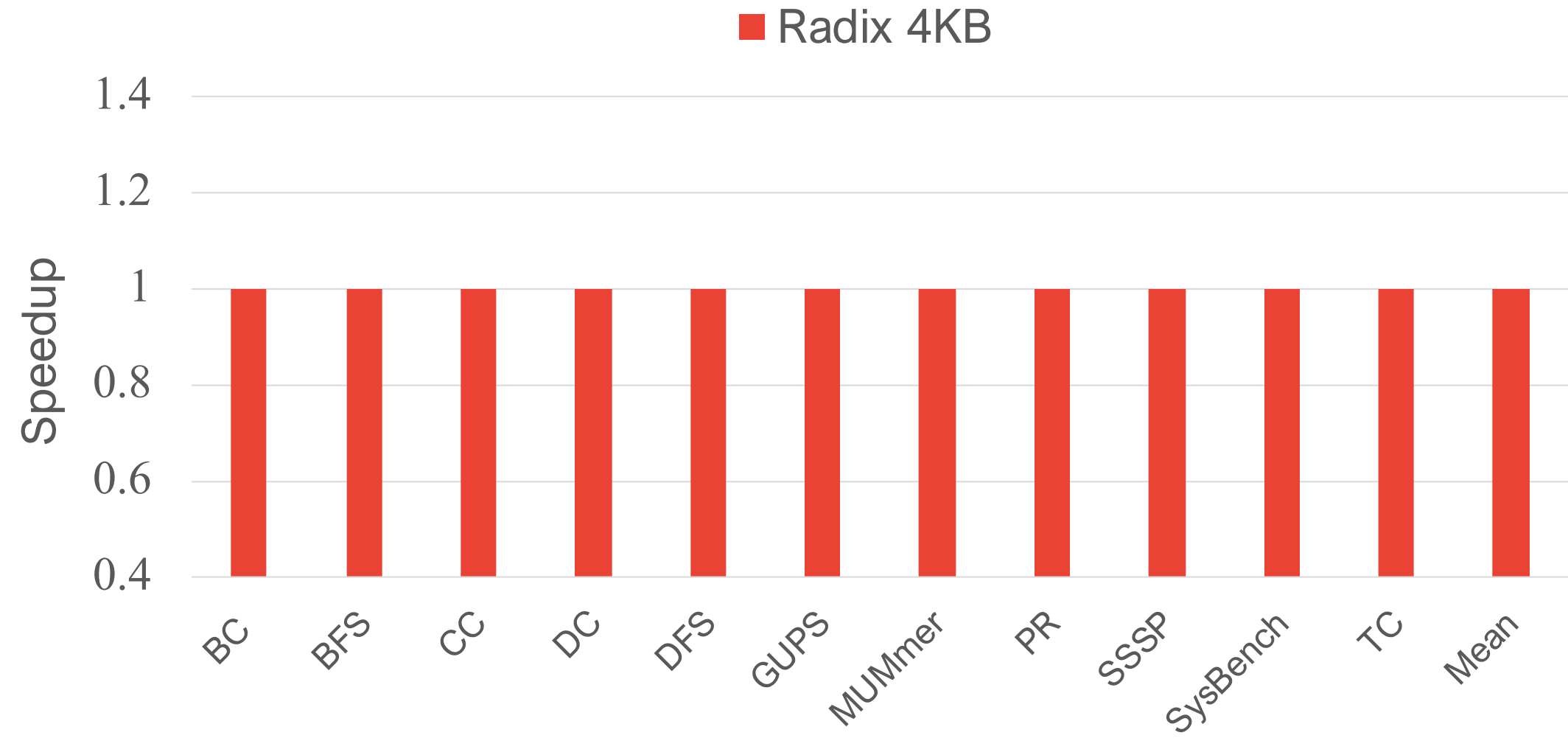
The background features a series of thin, light gray curved lines that sweep across the frame, creating a sense of motion and depth. These lines are more densely packed on the left side and become sparser towards the right.

Evaluation

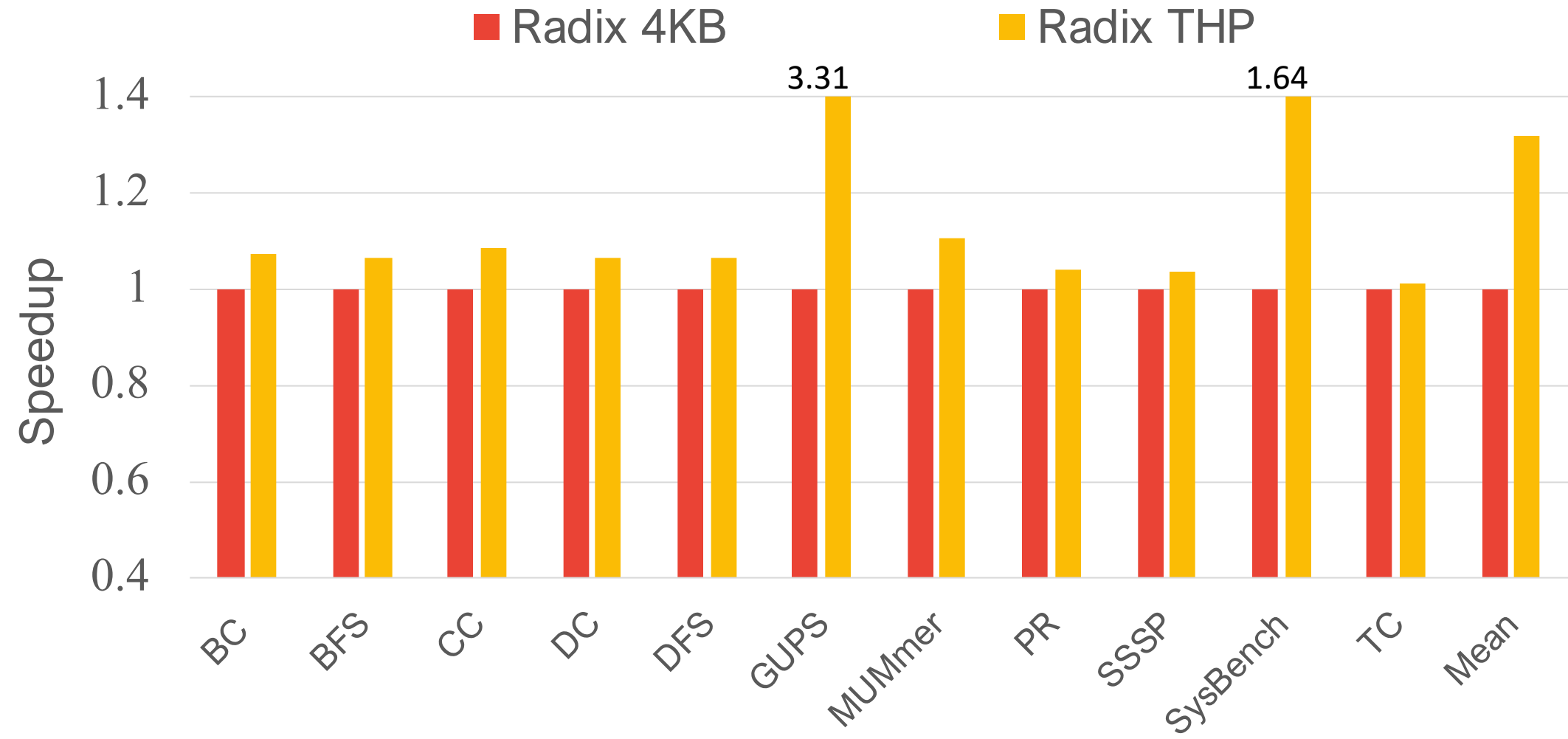
Application Speedup



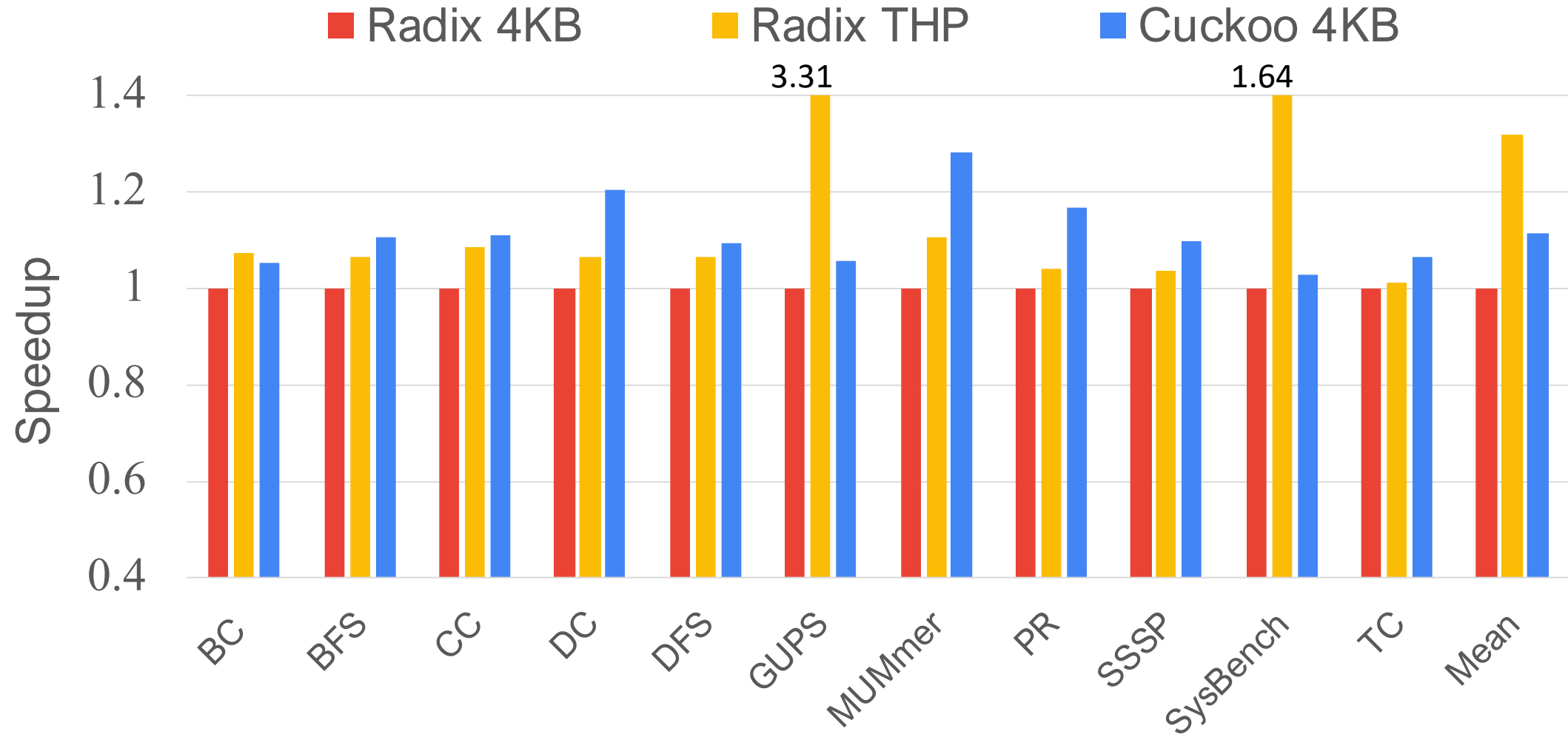
Application Speedup



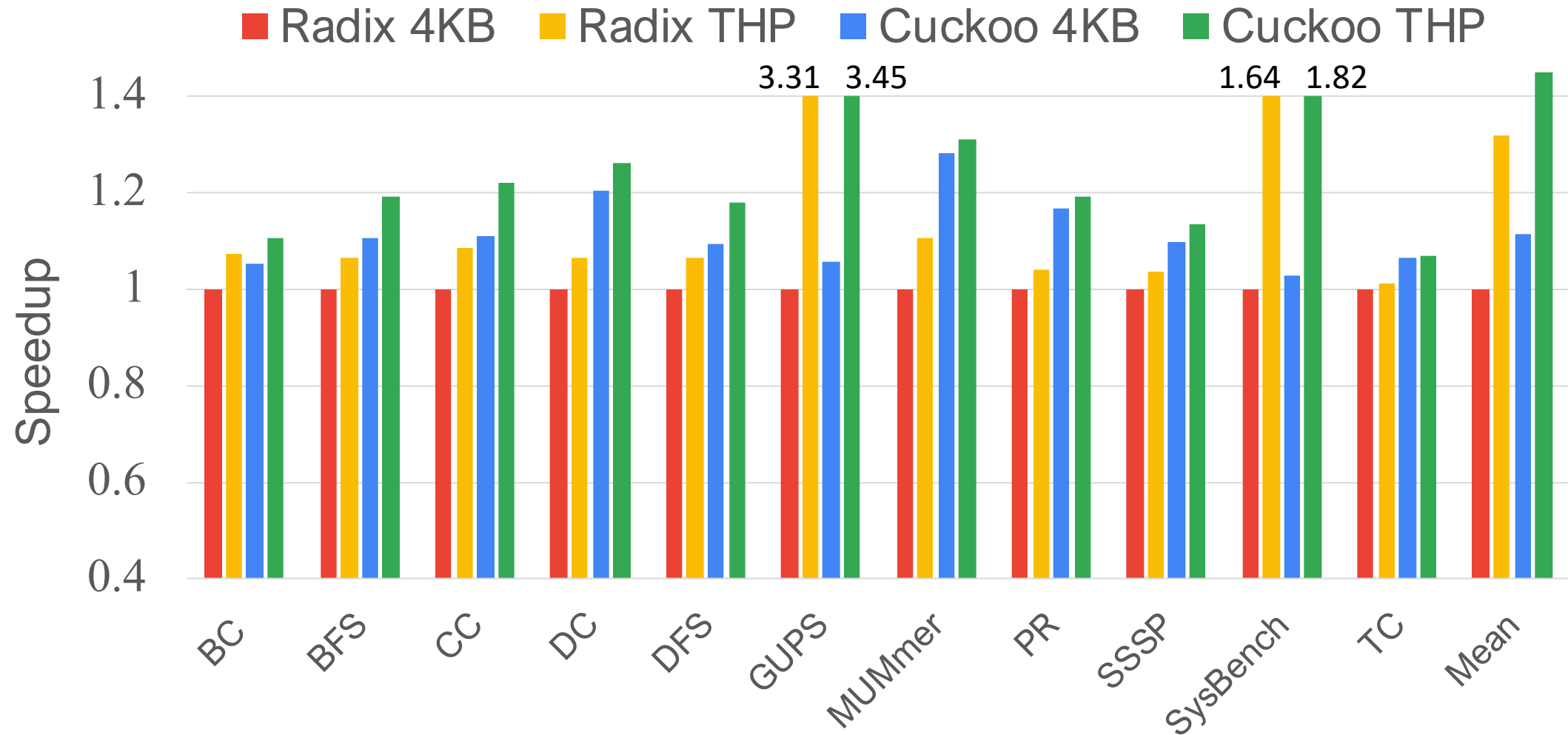
Application Speedup



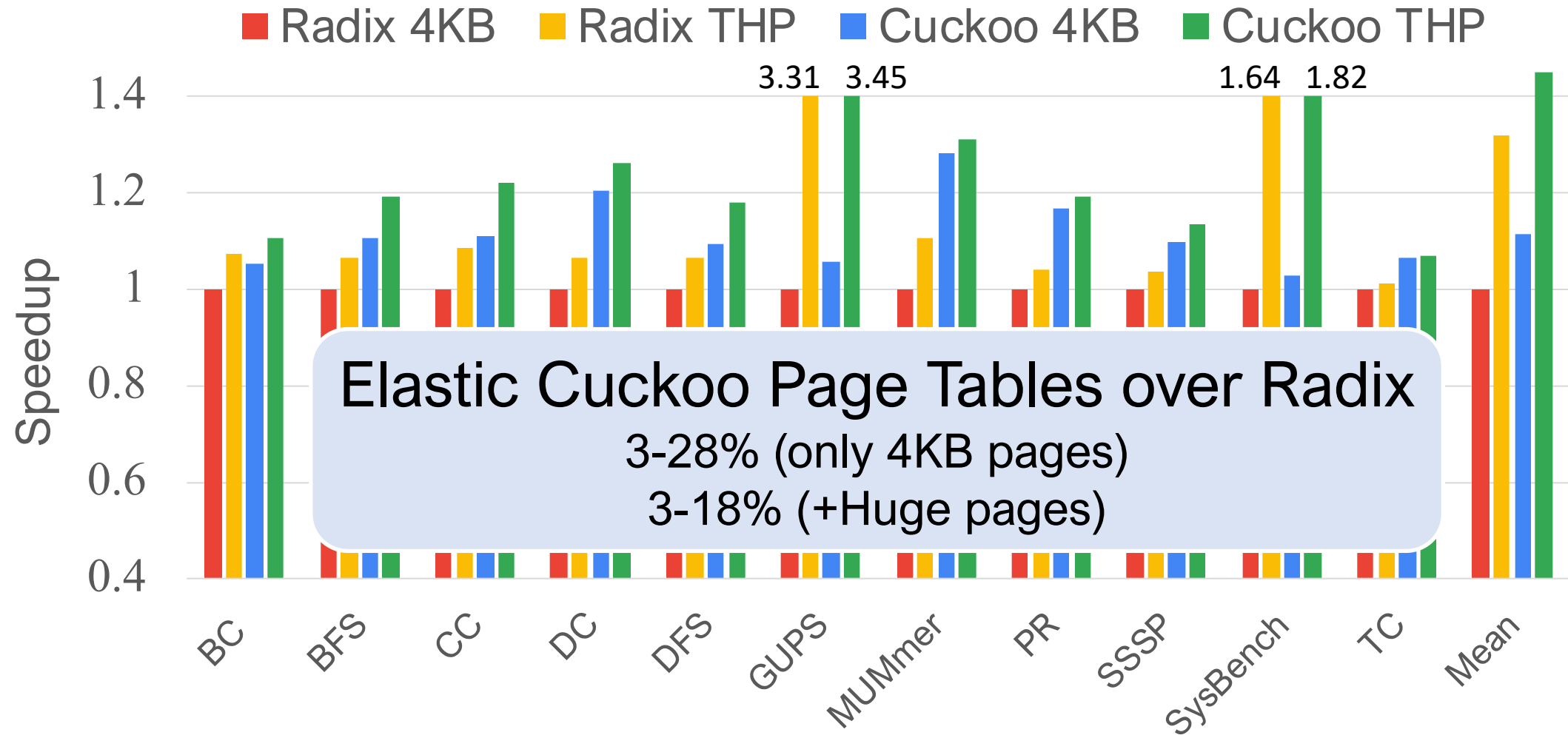
Application Speedup



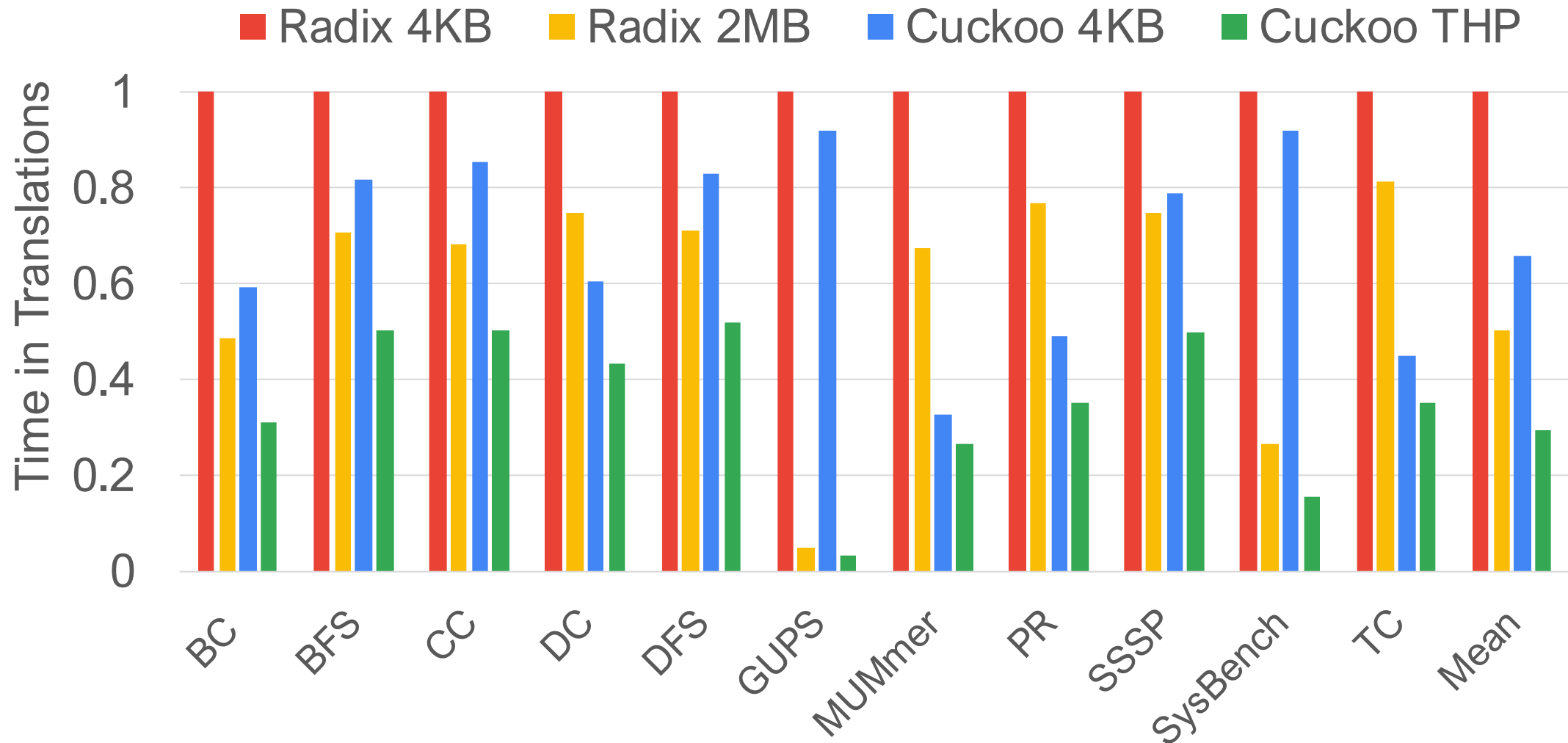
Application Speedup



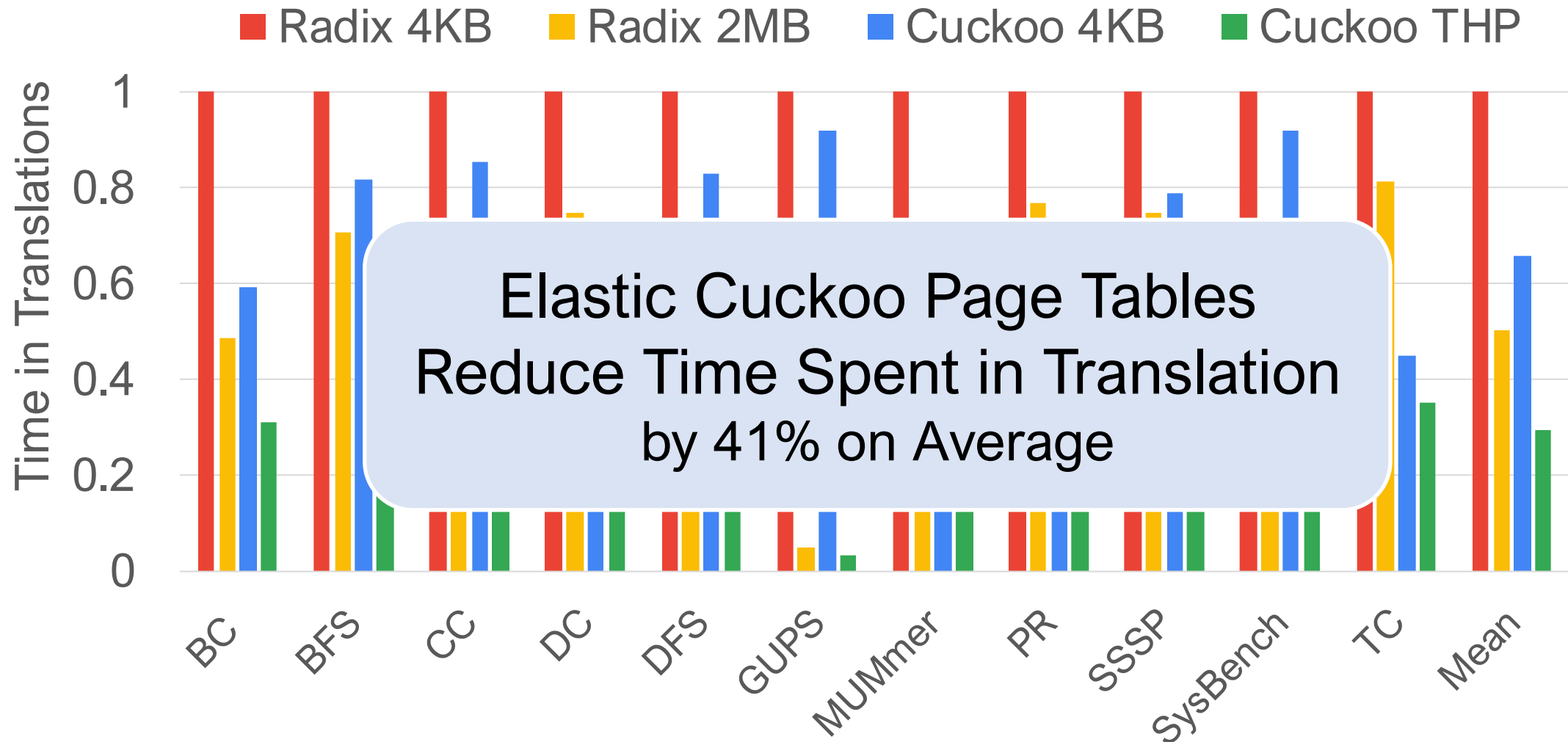
Application Speedup



Time Spent in Translations



Time Spent in Translations



More in the Paper

- Elastic cuckoo hashing operation
- Design of MMU cache and other structures
- More evaluation:
 - MMU and cache-subsystem characterization
 - Cuckoo walks characterization
 - Memory consumption of page tables

Takeaway: Elastic Cuckoo Page Tables

Better alternative to existing radix page tables

- Exploits parallelism in virtual translation for the first time
- Reduces the cost of dynamic resizing of hash tables
- Application speedup over state-of-the-art:
 - 3-28% with 4KB pages
 - 3-18% with Huge pages
- Expected high performance impact on:
 - Virtualized environments and nested page tables (ongoing work)

Elastic Cuckoo Page Tables: Rethinking Virtual Memory Translation for Parallelism

Dimitrios Skarlatos, Apostolos Kokolis, Tianyin Xu, Josep Torrellas

University of Illinois at Urbana-Champaign

skarlat2.web.engr.illinois.edu

ASPLOS 2020