Towards Speculative Offset-Based Address Translation

Chloe Alverti, Vasileios Karakostas
xalverti@cslab.ece.ntua.gr, vkarakos@cslab.ece.ntua.gr
Computing Systems Laboratory - National Technical University of Athens

1. MOTIVATION & BACKGROUND

Research Problem: Virtual Memory Overhead

Translation Lookaside Buffers miss penalty (page walks).

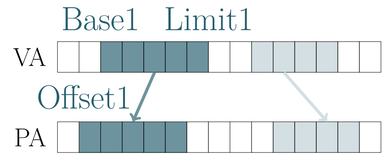
- Performance bottleneck for data intensive workloads

Prior Work: Redundant Memory Mappings [4]

Ranges→Unaligned contiguous virtual-to-physical mappings.
Eager paging→Allocation at request time to boost contiguity.
Range TLB→Accelerate translation.

Motivation: Robust Translation Performance

Eager paging→ insufficient to extract available contiguity, sensitive to fragmentation, narrows memory overcommitment.
Range TLB→Ranges can gradually break→Thrashing.



RangeTLB (FA,<>=)

$Base_1$,$Limit_1$	$Offset_1$,Per_1
$Base_2$,$Limit_2$	$Offset_2$,Per_2
...	...
$Base_N$,$Limit_N$	$Offset_N$,Per_N

2. CONTIGUITY-AWARE PAGING

Demand paging: single random page allocation at access time.
Eager paging: high order block preallocation at request time.

Idea

Enhance system allocator with ***contiguity awareness (CA)***.
Build vast offset-based ranges on a best effort basis.

- Boost and shield contiguity
- Opportunity to relax preallocation

Mechanism

Linux *struct vma* → contiguous virtual address range.

- new *Offset* attribute, track and store *pa - va*

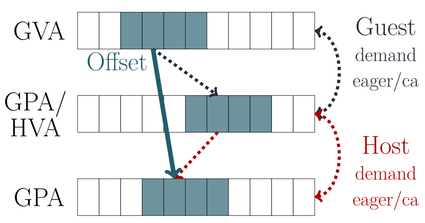
How CA-paging works on allocation requests:

(a) Identify physical block with respect to *vma*→Offset
(b) Available? check → *struct page* mem_map array*
(c) If free, reserve it from buddy allocator
Fallback → default, pick a random block
(d) Establish mapping → Range extended

Advantages

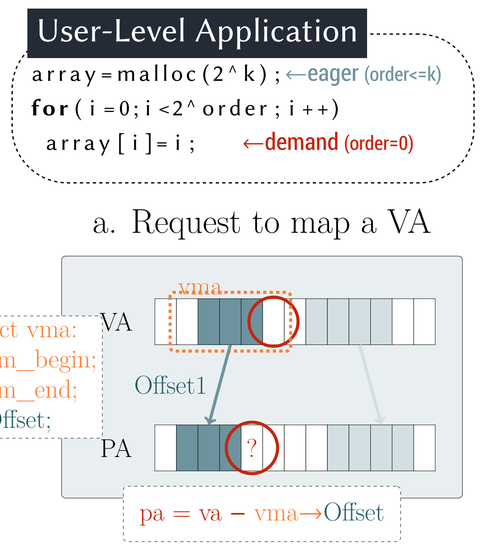
- Range construction across requests
- Resilience to external fragmentation
- Independence of alignment restrictions
- Orthogonal to eager or demand paging

Virtualized Environments



Range → GVA to HPA contiguous mapping.

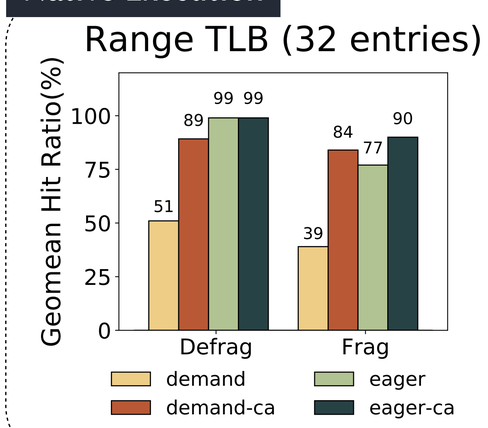
- Non-coordinated Guest⟷Host eager paging
- Work in Progress



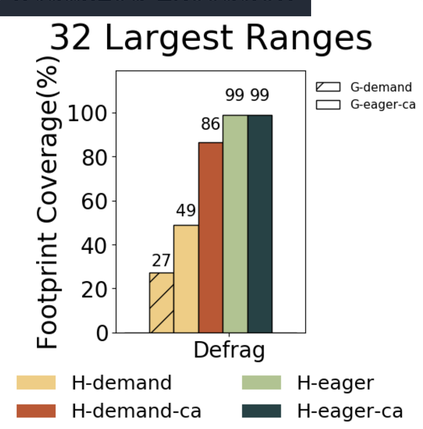
Methodology

We implement CA-paging in Linux 3.15.5.
Badger Trap [2] → RangeTLB performance native execution.
In-house VM Introspection tool → GVA to HVA mappings.
Benchmarks: data-intensive SPEC2006/Parsec workloads.

Native Execution



Virtualized Execution



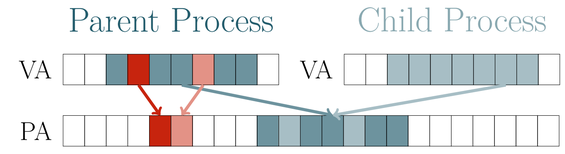
~84-89% performance with demand paging
~13% improvement fragmented environment

3. DISCONTINUITIES IN RANGES

Challenge

Discontinuities in ranges may appear due to:

- External Fragmentation
- Less Aggressive Allocator (i.e demand, eager)
- Lightweight Memory Management Techniques gradually breaking ranges
  - Copy-on-write
  - Working set sampling
  - Memory deduplication (KSM)



Multiple non-contiguous ranges share the same Offset

4. OFFSET-BASED SPECULATION

Motivation

As the number of ranges increases, scaling Range TLB alone is not a sustainable solution due to:

- fully-associative lookups
- range comparisons

Idea

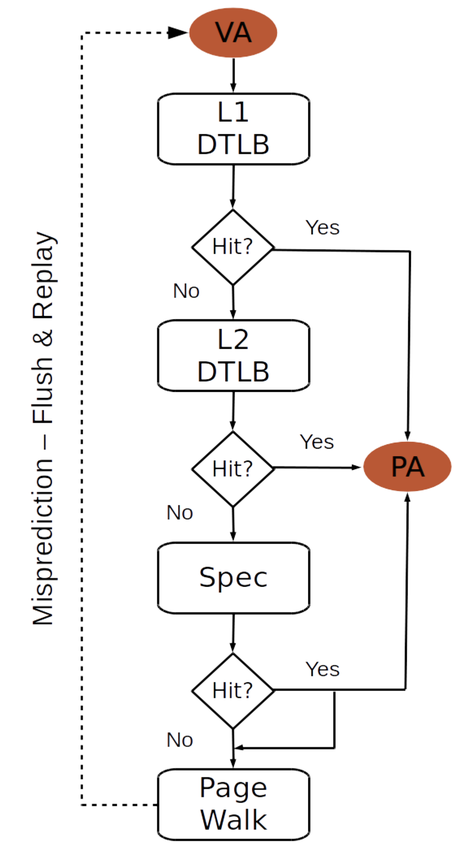
Ranges Discontinuities → opportunity to ***speculate*** the address translation based on **Offset**.

Our Proposal

- Decouple Offsets from virtual range boundaries
- Speculate *pa = va - Offset*
- Verification page table walks on the background
- Mispeculation → flush and replay

Spec Structure

PC_1	$Offset_1$
PC_2	$Offset_2$
...	...
PC_N	$Offset_N$



Preliminary Mechanism

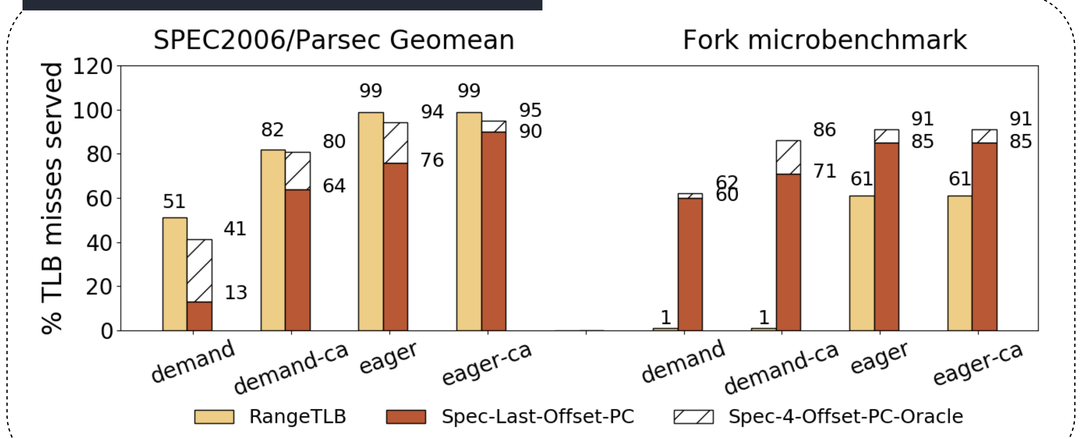
Known observation → Only a few instructions are responsible for the majority of the TLB misses.
Index speculation structure by Program Counter.

(a) Track the last Offset per PC
(b) Track 4 LRU Offsets per PC → Oracle selects the correct one → evaluate potential

Methodology

Speculation could be an auxiliary translation mechanism. For preliminary evaluation, we adopt the challenging scenario of replacing RangeTLBs.
Badger Trap [2] → RangeTLB and Speculation.

Speculation vs RangeTLB – Native



~90% performance at optimal (eager-ca) conditions
Potentially close to RangeTLB performance
~20% better performance than RangeTLB for Fork microbenchmark

- Preliminary results from (VM Introspection tool) indicate that on a multiple VM environment with KSM, tracking offsets instead of ranges covers 10% more of an application's footprint.

[1] T. Barr, A. Cox, and S. Rixner. SpecTLB: A mechanism for speculative address translation. In *ISCA 2011.*
[2] J. Gandhi and A. Basu et al. Badgertrap: A tool to instrument x86-64 tlb misses. *SIGARCH Comput. Archit. News*, 2014.
[3] S. Haria, M. Hill, and M. Swift. Devirtualizing memory in heterogeneous systems. In *ASPLOS 2018.*
[4] V. Karakostas and J. Gandhi et al. Redundant memory mappings for fast access to large memories. In *ISCA 2015.*
[5] Binh Pham et al. Large pages and lightweight memory management in virtualized environments: Can you have it both ways? In *MICRO 2015.*
[6] T. Zheng, H. Zhu, and M. Erez. SIPT: Speculatively Indexed, Physically Tagged Caches. In *HPCA 2018.*