

# Punch holes in your large pages for benefits (performance, flexibility) with less overhead

## Perforated Pages: Supporting Fragmented Memory Allocation for Large Pages (ISCA 2020)



Chang Hyun Park, Sanghoon Cha, Bokyeong Kim, Youngjin Kwon, David Black-Schaffer, and Jaehyuk Huh



### 1 Large pages for performance

- Large page mapping:
  - 512x TLB coverage  $\uparrow$
  - ~68% performance  $\uparrow$ <sup>1</sup>

But

### 2 Strict Requirements

- 512 contiguous 4KB pages
- 2MB alignments
- Homogeneous properties

Cause

### 3 Overhead 1: Data movement

- Compact to create large pages
- Takes up to 35% of execution time<sup>2</sup>
- Immovable pages hinder compaction
  - Kernel allocations, I/O buffers, etc.
- Existed in 50% of 2MB regions

Fragmented → After compaction

Legend: ■ Used page, ■ Free page, ■ Immovable page

### 4 Overhead 2: Wasted Memory

- Sparse access/use → Wasted space
- Example: Redis using large pages
  - 20% more memory consumption
  - 45% fewer TLB misses → 29% higher throughput

### 5 Our Solution: Perforated Pages

- Perforated page (2MB): maps common case
  - Large page-like **coverage** and **mapping**
- Hole Page (4KB): maps special case
  - Flexibility** for specific mappings

### Example Usages $\downarrow$

- Map to another physical page
- Different permissions/properties
- Leave free (save memory space)

### 6 How it works - Page table

- Hole Bitmap: Identifies hole pages - stored in DRAM
- Shadow L2: provides an additional pointer in the page table
  - Main L2 PTE points to perforated page mapping (2MB)
  - Shadow L2 PTE points to L1 PT that holds hole page mappings (4KB)

### 7 How it works - L2 TLB

- Cache and translate perforated/hole pages at L2 TLB
- L2 TLB caches additional entries:
  - Perforated mappings: 2MB page-like coverage
  - Hole mappings: 4KB flexible mappings
  - Hole bitmaps: cached in L2 TLB
- Optimization: bitmap filter
  - Identify non-hole regions
  - Stored in perforated mappings

Legend: ■ Perforated pages, ■ 2MB pages, ■ Regular/hole 4KB, ■ Hole bitmaps

Annotations:
 

- Large pages cannot be created with fragmentation
- L2 TLB is filled with hole bitmaps
- Perforated pages step in to improve TLB coverage

### 8 Evaluation Summary

Legend: — Baseline 4KB, - - - Baseline 2MB+4KB, — Perforated 25% holes

Annotations:
 

- Result summary based on 50% fragmentation scenario (Observed on real system)
- Large pages cannot be created with fragmentation
- Perforated pages provide most of the Large page benefits
- Perforated pages retain performance of ideal large page mappings

Real benchmarks show 2-11% improvement (at 50% fragmentation) and 93-99% of ideal mapping performance (at 100% fragmentation).

<sup>1</sup> Margaritov et al., MICRO '19  
<sup>2</sup> Parwar et al., ASPLOS '18