

# Coordinated and Efficient Huge Page Management with Ingens

Youngjin Kwon, Hangchen Yu, Simon Peter, Christopher J. Rossbach, Emmett Witchel



The University of Texas at Austin



VMware Research Group

## Huge pages reduce address translation cost

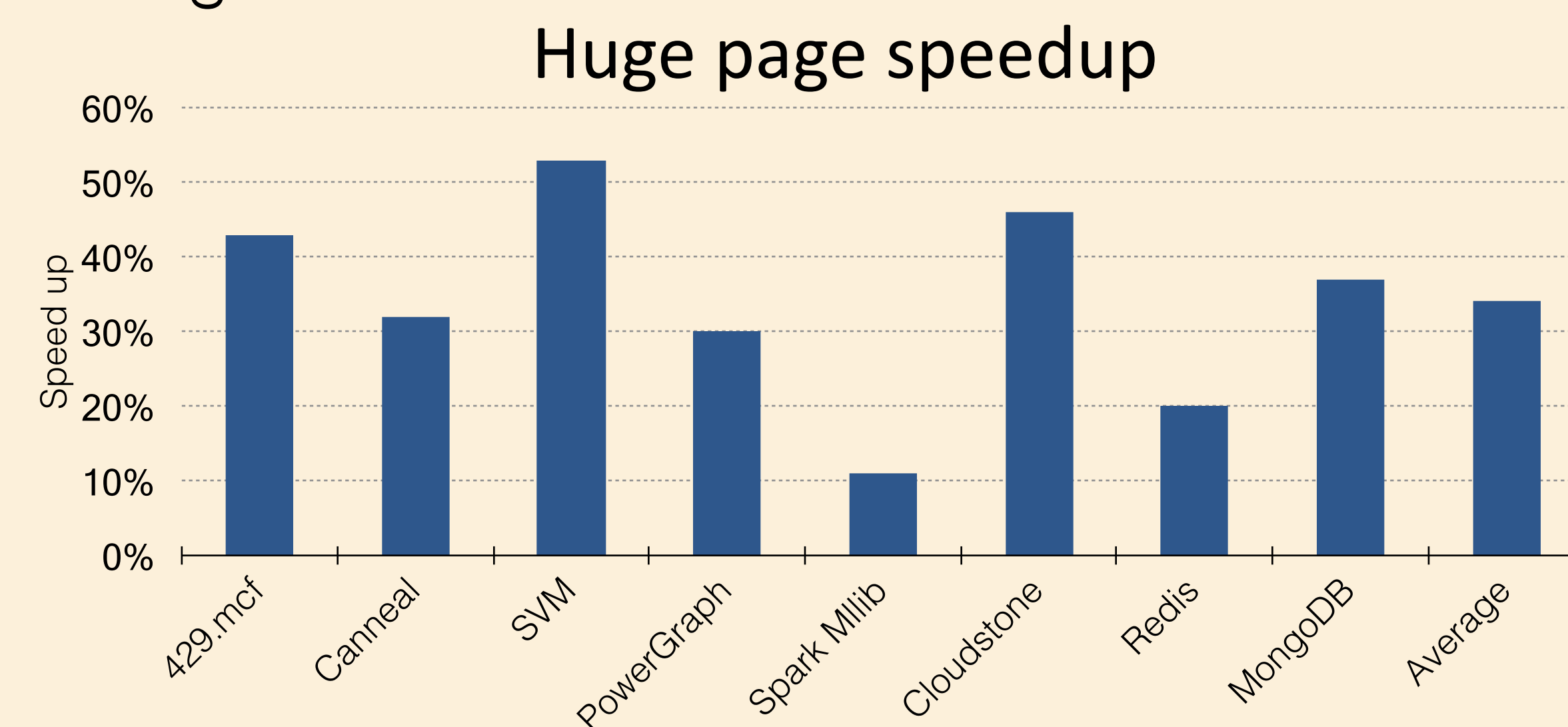
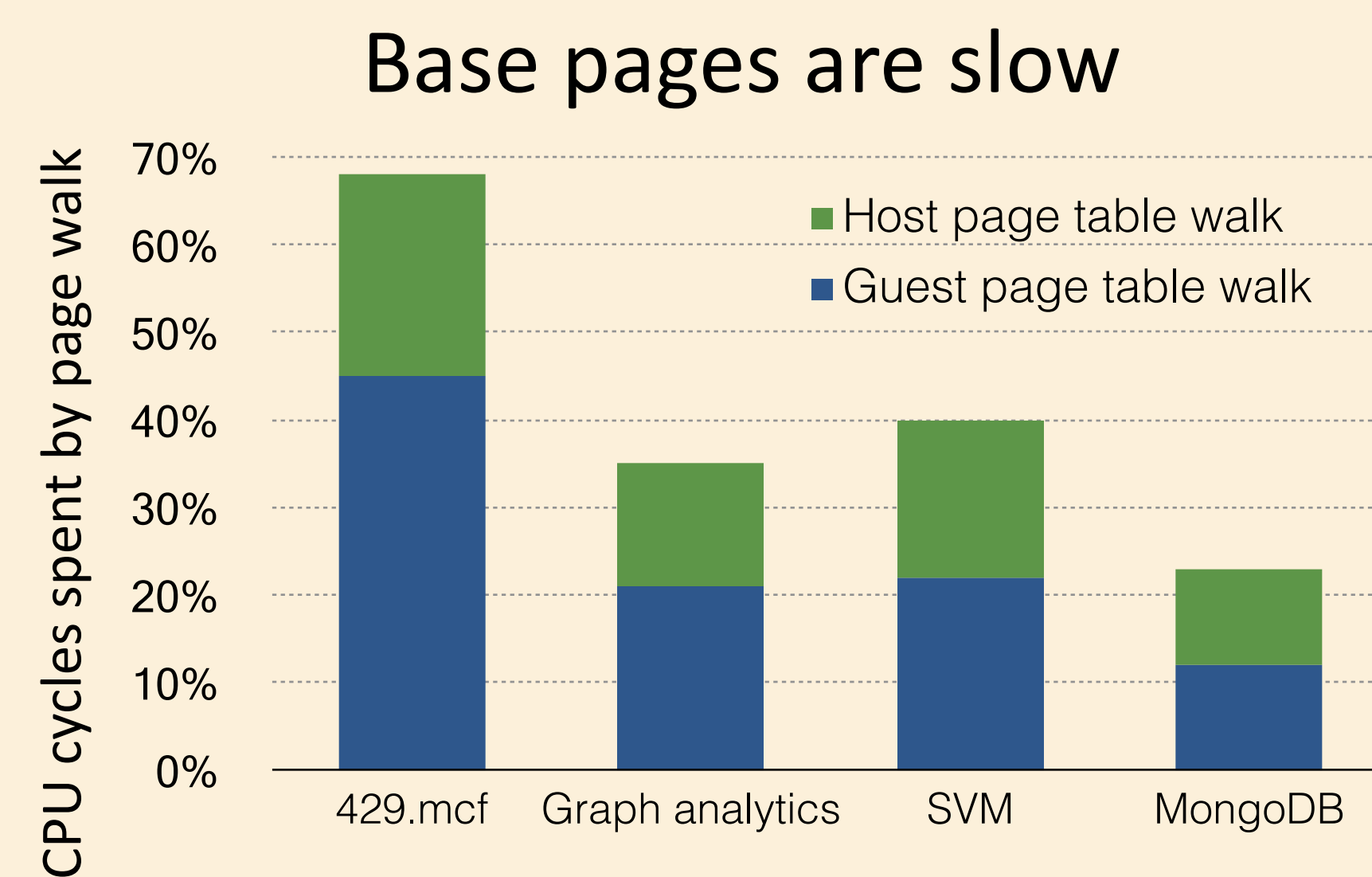
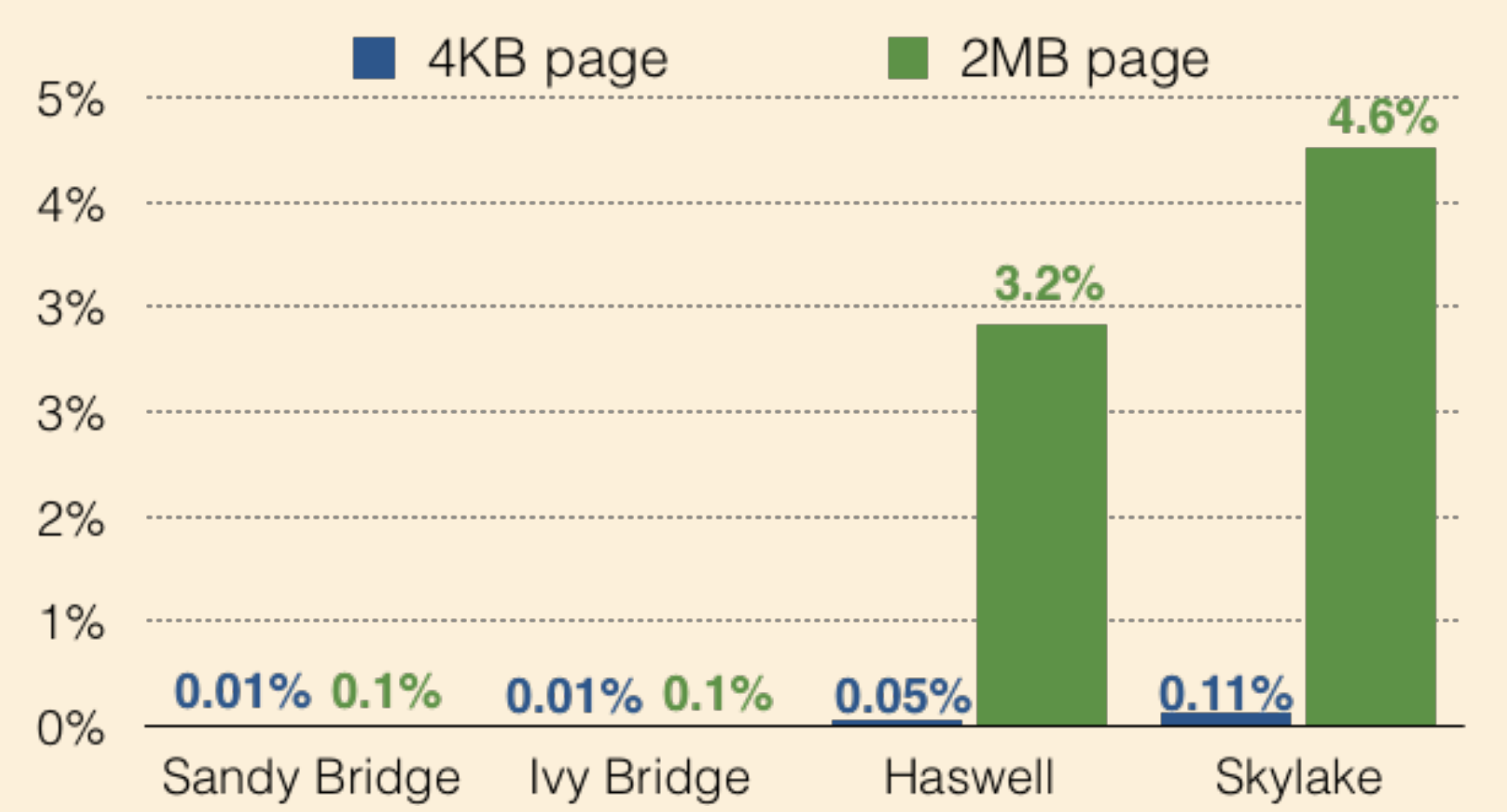
Modern applications

- Large memory footprint
- Low memory locality

Huge page TLB coverage increasing

- Huge pages now useful

TLB coverage proportional to 64 GB DRAM



## Pathologies in current huge page management

**Internal fragmentation**

**External fragmentation**

**Memory bloating**

- Huge pages increase fragmentation
- Fragmentation happens fast in any size of physical memory

	Using huge page	Using only base page
Redis	20.7GB (1.69x)	12.2GB
MongoDB	12.4GB (1.23x)	10.1GB

**High huge page fault latency**

4KB page	3.6 us
2MB page	378 us

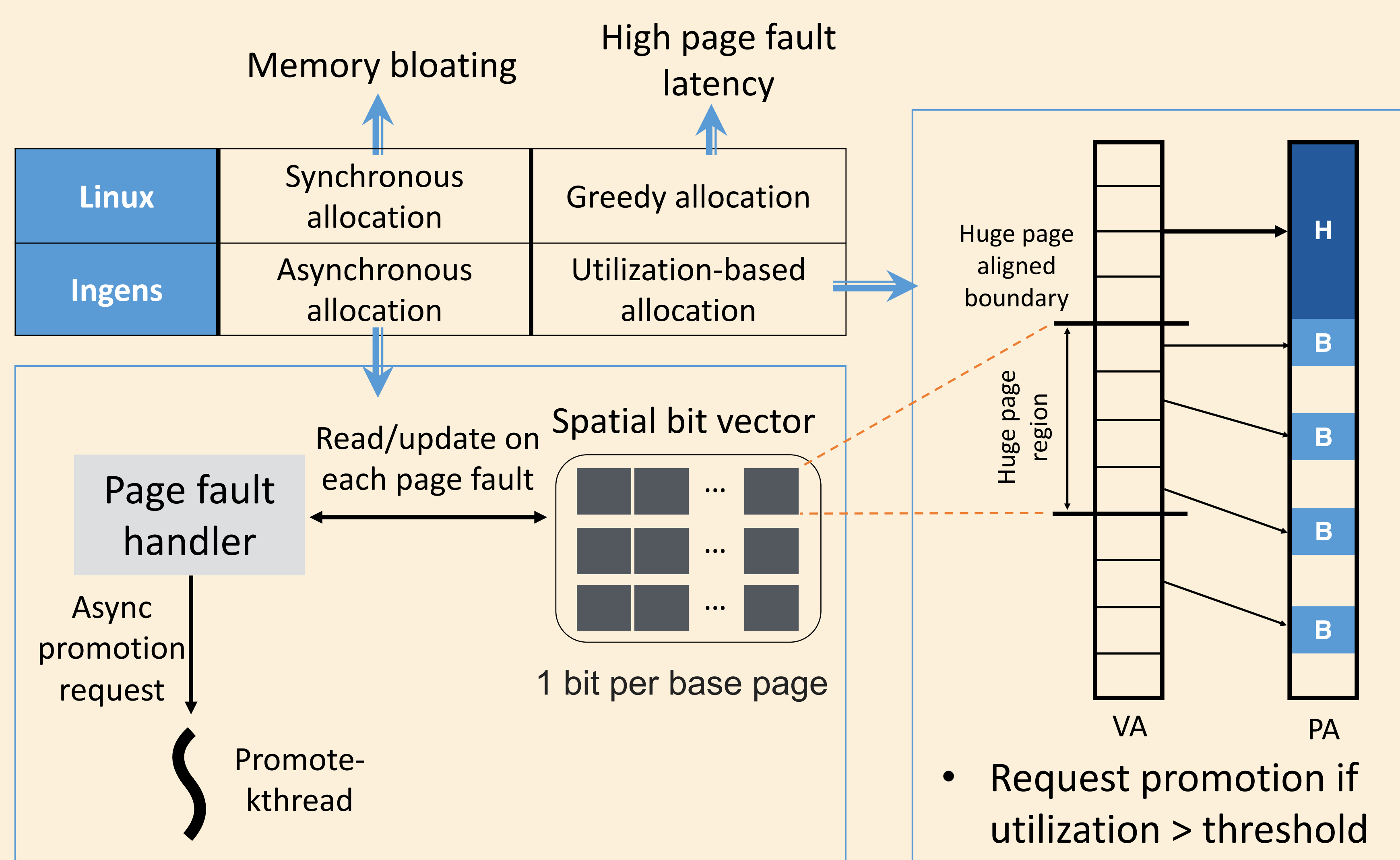
Linux synchronously compacts memory

- Further increases latency

Flowchart: Application pause → Allocate page(s) → Get page(s) from free page list → Compact physical memory → Zero the page(s) → Map the page(s) to page table → Application resume. Includes a 'Not enough contiguous memory' block and 'Page fault handler' / 'Physical memory manager' labels.

## Ingens: Asynchronous and utilization-based huge page promotion

### Ingens design



### Evaluation

**CloudStone WEB 2.0 Benchmark**

	Linux	Ingens
Throughput (req/s)	922.3	1091.9 (+18%)

**Read-dominant operations**

Operation	Linux (ms)	Ingens (ms)
View event (Average)	478	338
View event (90th)	605	354
Visit homepage (Average)	236	207
Visit homepage (90th)	372	226

**Eliminates Redis memory bloating**

	Linux (base only)	Linux (huge)	Ingens
Memory	12.2 GB	20.7 GB	12.3 GB

**GET latency (ms)**

	Throughput	90th lat.	99.9th lat.
Linux (base only)	19.0K	4	109
Linux (huge)	21.7K	3	8
Ingens	20.9K	3	64

**Negligible overhead**

- 2% in the worst case