Coordinated and Efficient Huge Page Management with Ingens

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

The University of Texas at Austin

vmware RESEARCH

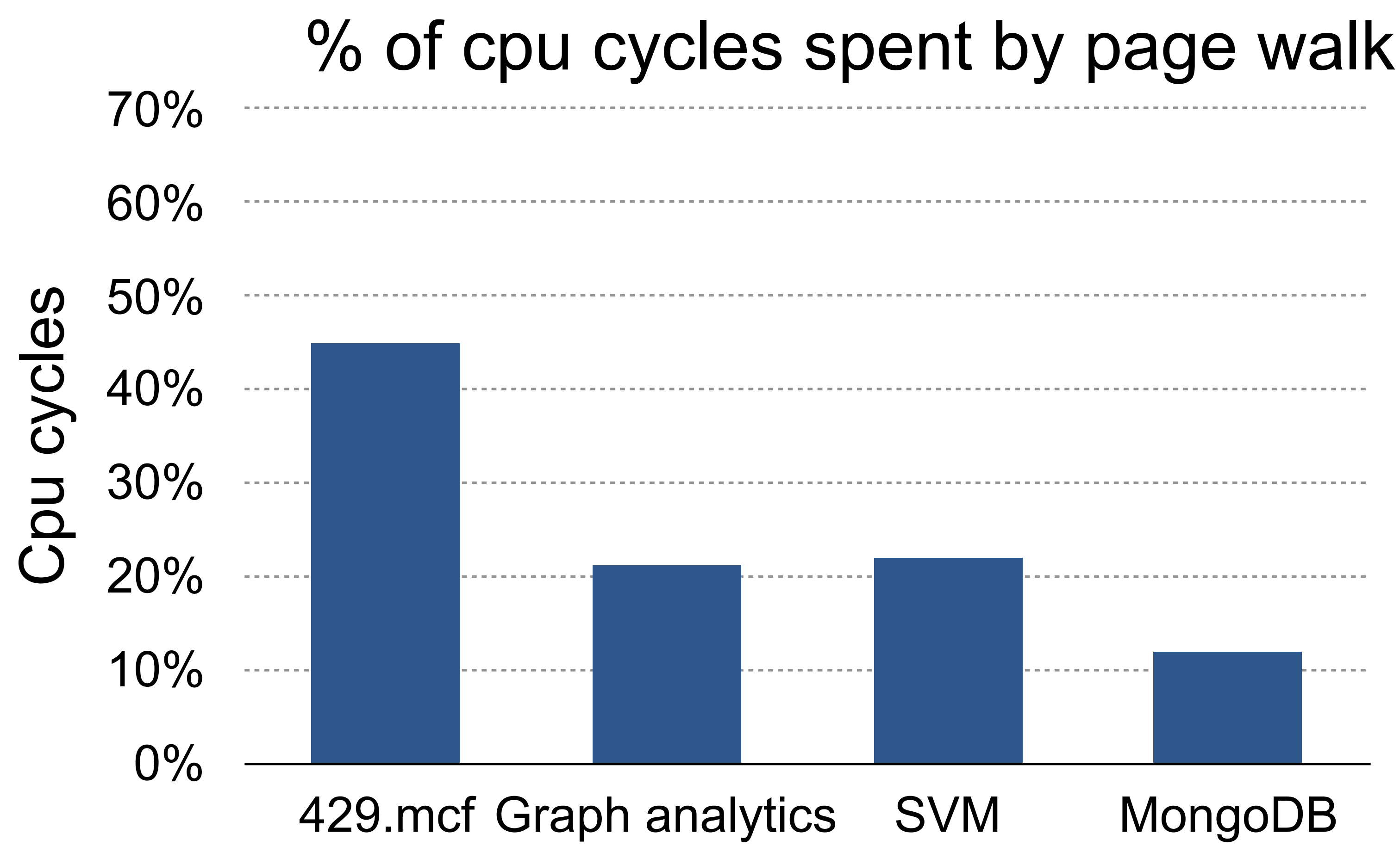
High address translation cost

- Modern applications: large memory footprint, low memory access locality
- TLB coverage using base pages is insufficient

Virtual address

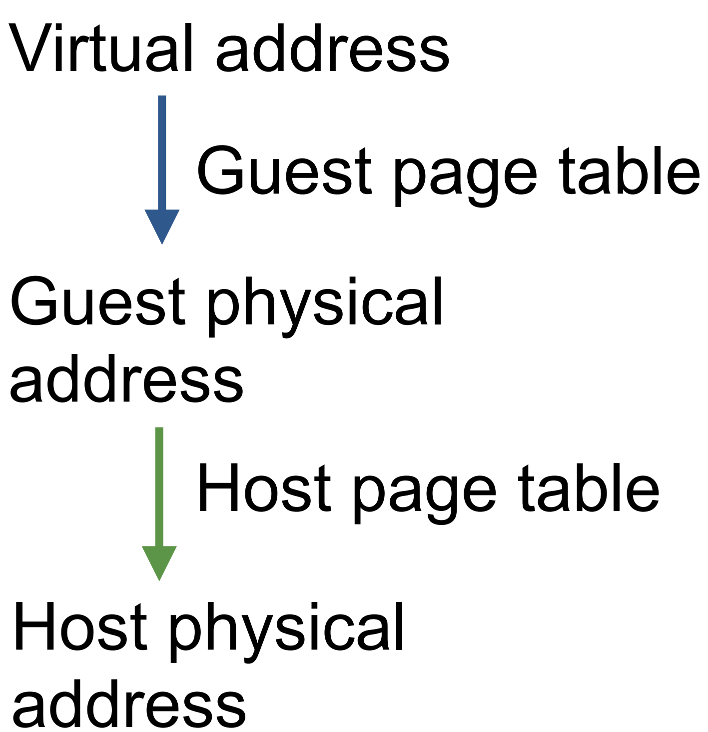
↓ Page table

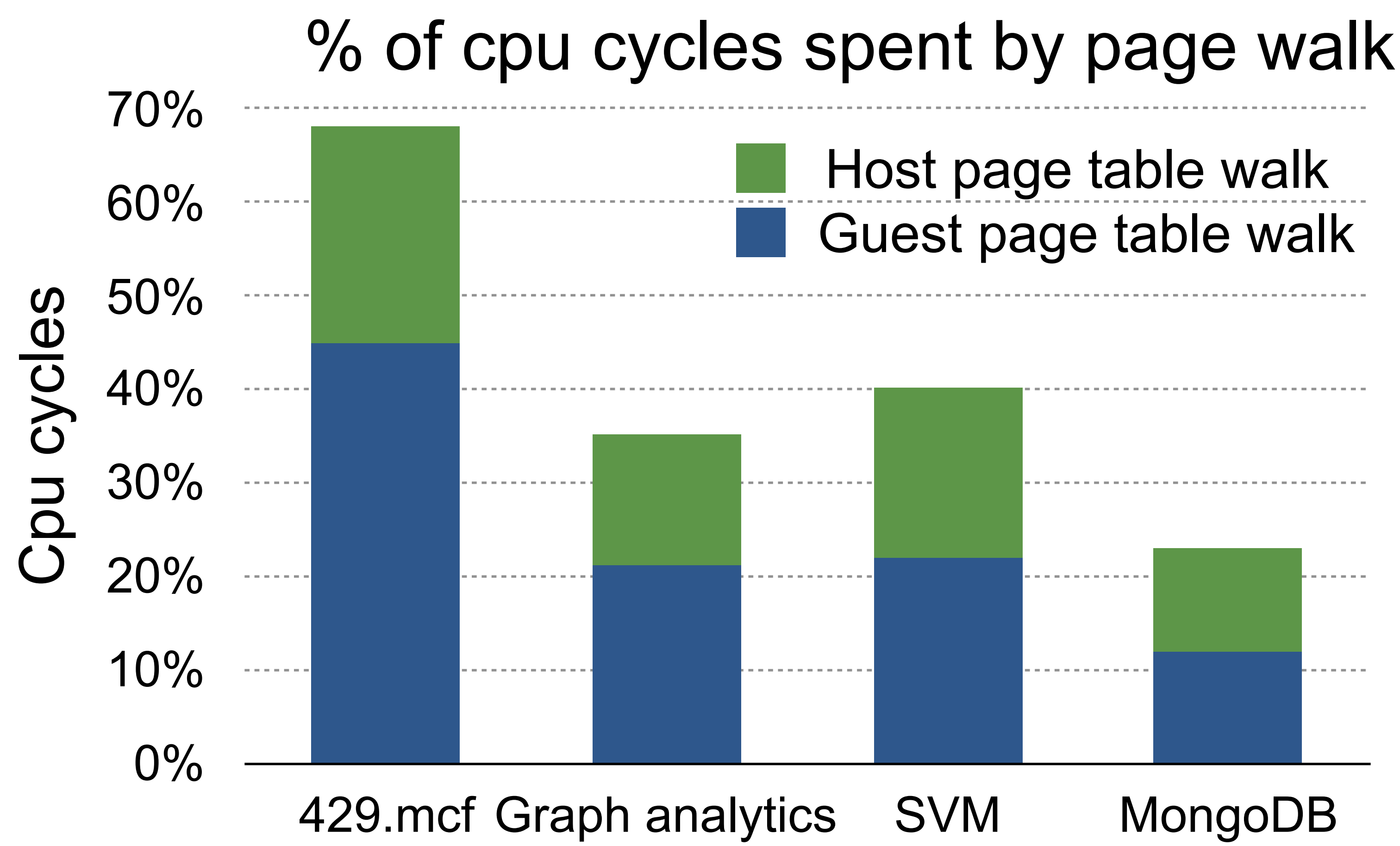
Physical address



High address translation cost

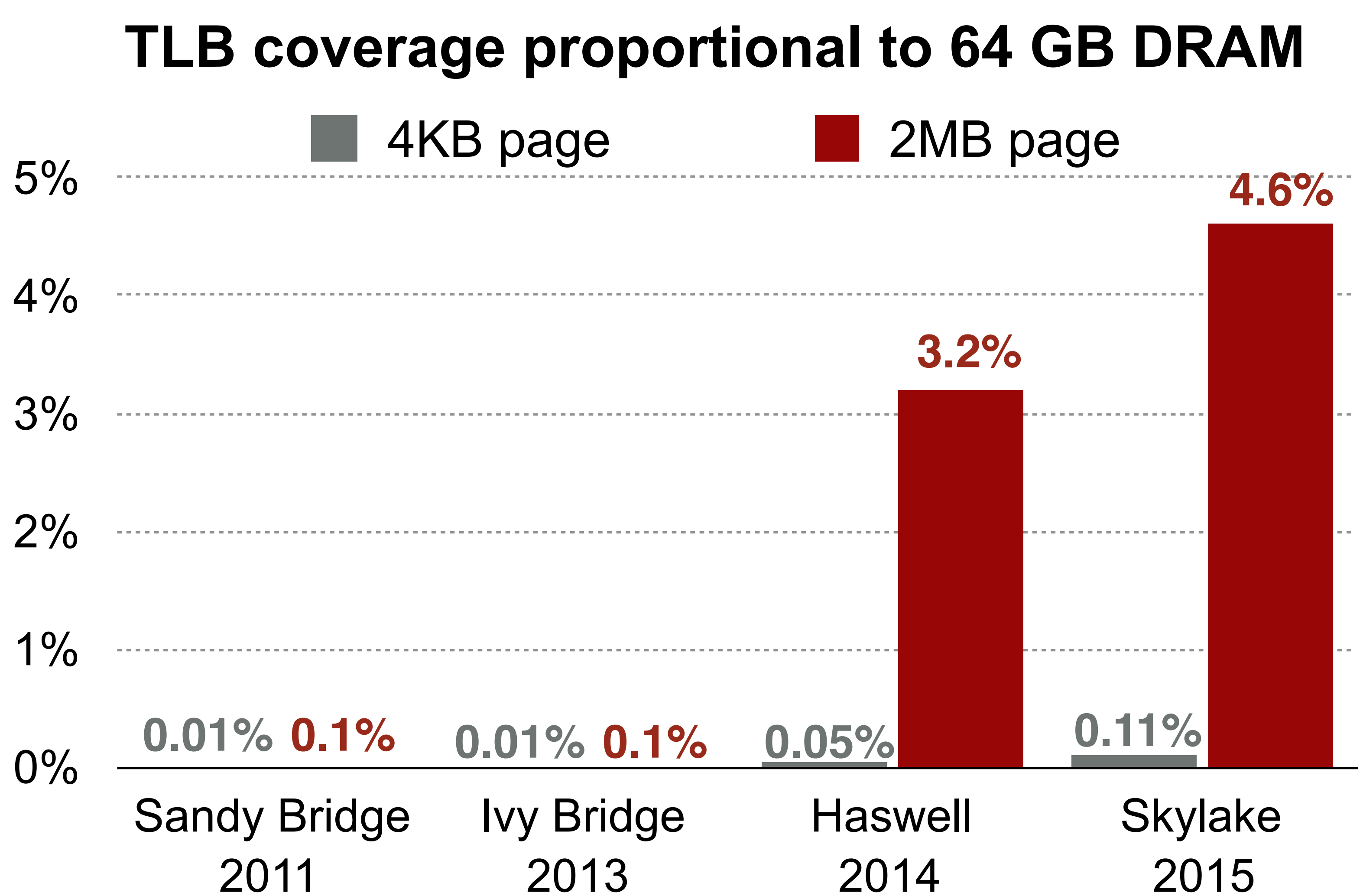
- Virtualization requires additional address translation





Huge pages improve TLB coverage

- Architecture supports larger page size (e.g., 2MB page)
  - Intel: 0 to 1,536 entries in 2 years (2013 ~ 2015)
- Operating system has the burden of better huge page support



Operating system support for huge pages

- OS transparently allocates/deallocates huge pages
- Huge pages in both guest and host

FreeBSD **Practical, transparent operating system support for superpages**

Juan Navarro[†] Sitaram Iyer[†] Peter Druschel[†] Alan Cox[†]

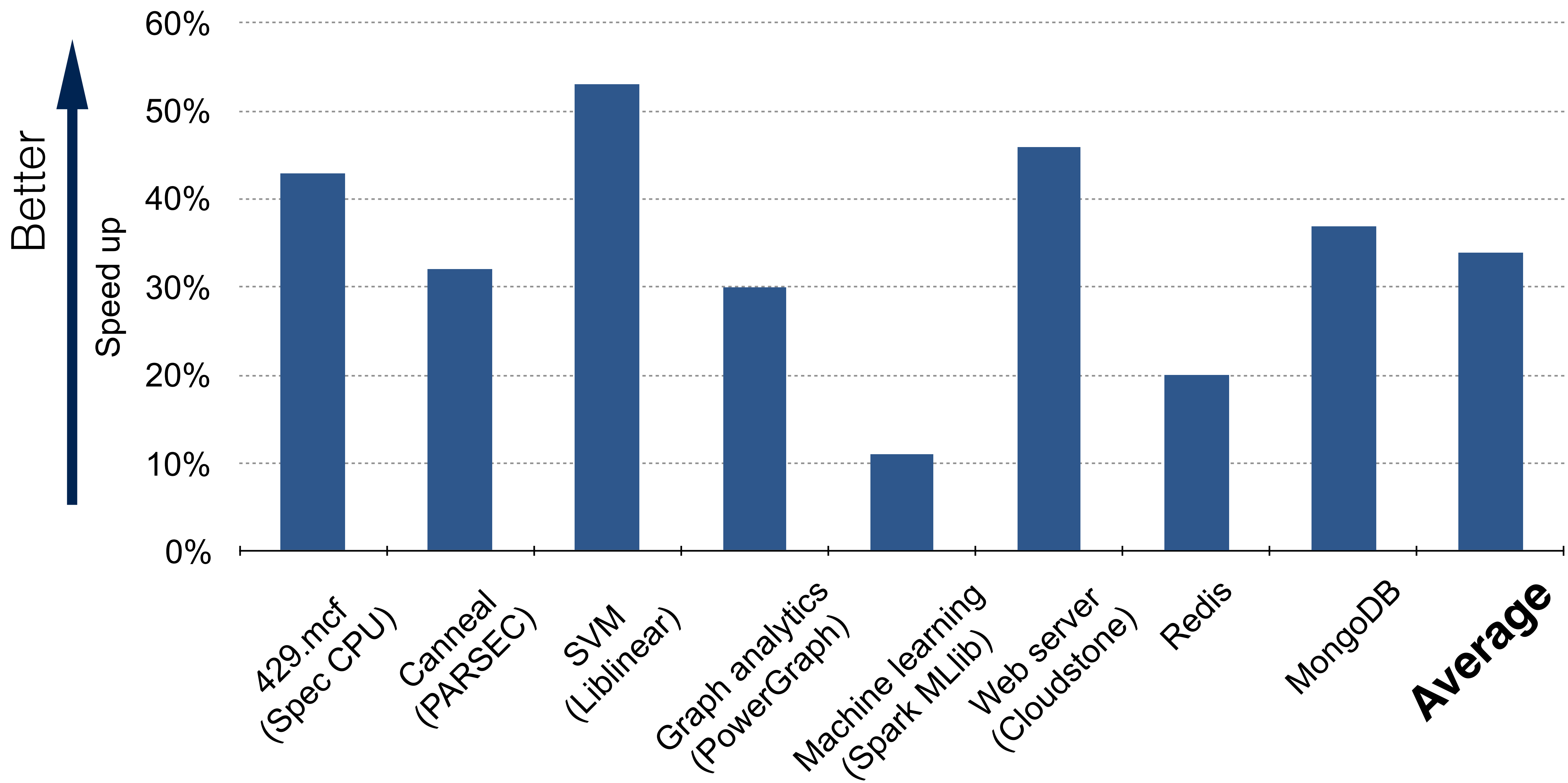
OSDI '02

Linux **Transparent huge pages in 2.6.38**

LWN.net, 2011

Huge pages improve performance

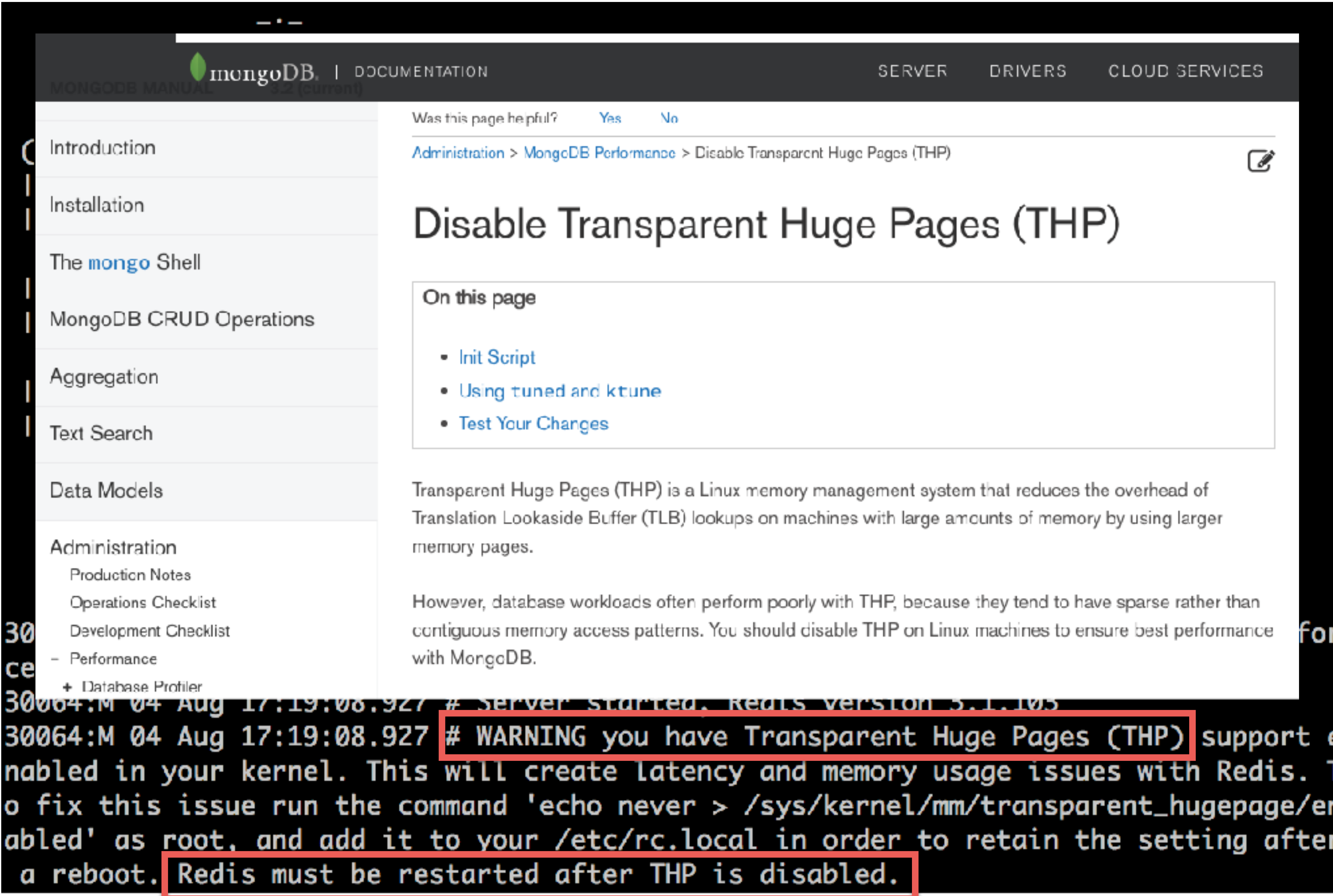
- Application speed up over using base pages only



Are huge pages a free lunch?

```
                _._
           _.-``__ ''-._
      _.-``    `.  `_.  ''-._           Redis 3.1.103 (3bba4842/1) 64 bit
  .-`` .-```.  ```\/    _.,_ ''-._
 (    '      ,       .-`  | `,    )     Running in standalone mode
 |`-._`-...-` __...-.``-._|'` _.-'|     Port: 6379
 |    `-._   `._    /     _.-'    |     PID: 30064
  `-._    `-._  `-./  _.-'    _.-'
 |`-._`-._    `-.__.-'    _.-'_.-'|
 |    `-._`-._        _.-'_.-'    |           http://redis.io
  `-._    `-._`-.__.-'_.-'    _.-'
 |`-._`-._    `-.__.-'    _.-'_.-'|
 |    `-._`-._        _.-'_.-'    |
  `-._    `-._`-.__.-'_.-'    _.-'
      `-._    `-.__.-'    _.-'
          `-._        _.-'
              `-.__.-'

30064:M 04 Aug 17:19:08.927 # WARNING: The TCP backlog setting of 511 cannot be enfor
ced because /proc/sys/net/core/somaxconn is set to the lower value of 128.
30064:M 04 Aug 17:19:08.927 # Server started, Redis version 3.1.103
30064:M 04 Aug 17:19:08.927 # WARNING you have Transparent Huge Pages (THP) support e
nabled in your kernel. This will create latency and memory usage issues with Redis. T
o fix this issue run the command 'echo never > /sys/kernel/mm/transparent_hugepage/en
abled' as root, and add it to your /etc/rc.local in order to retain the setting after
 a reboot. Redis must be restarted after THP is disabled.
```



mongoDB. | DOCUMENTATION   SERVER   DRIVERS   CLOUD SERVICES

- Introduction
- Installation
- The mongo Shell
- MongoDB CRUD Operations
- Aggregation
- Text Search
- Data Models
- Administration
  - Production Notes
  - Operations Checklist
  - Development Checklist
  - − Performance
    - + Database Profiler

Was this page helpful? Yes No

Administration > MongoDB Performance > Disable Transparent Huge Pages (THP)

Disable Transparent Huge Pages (THP)

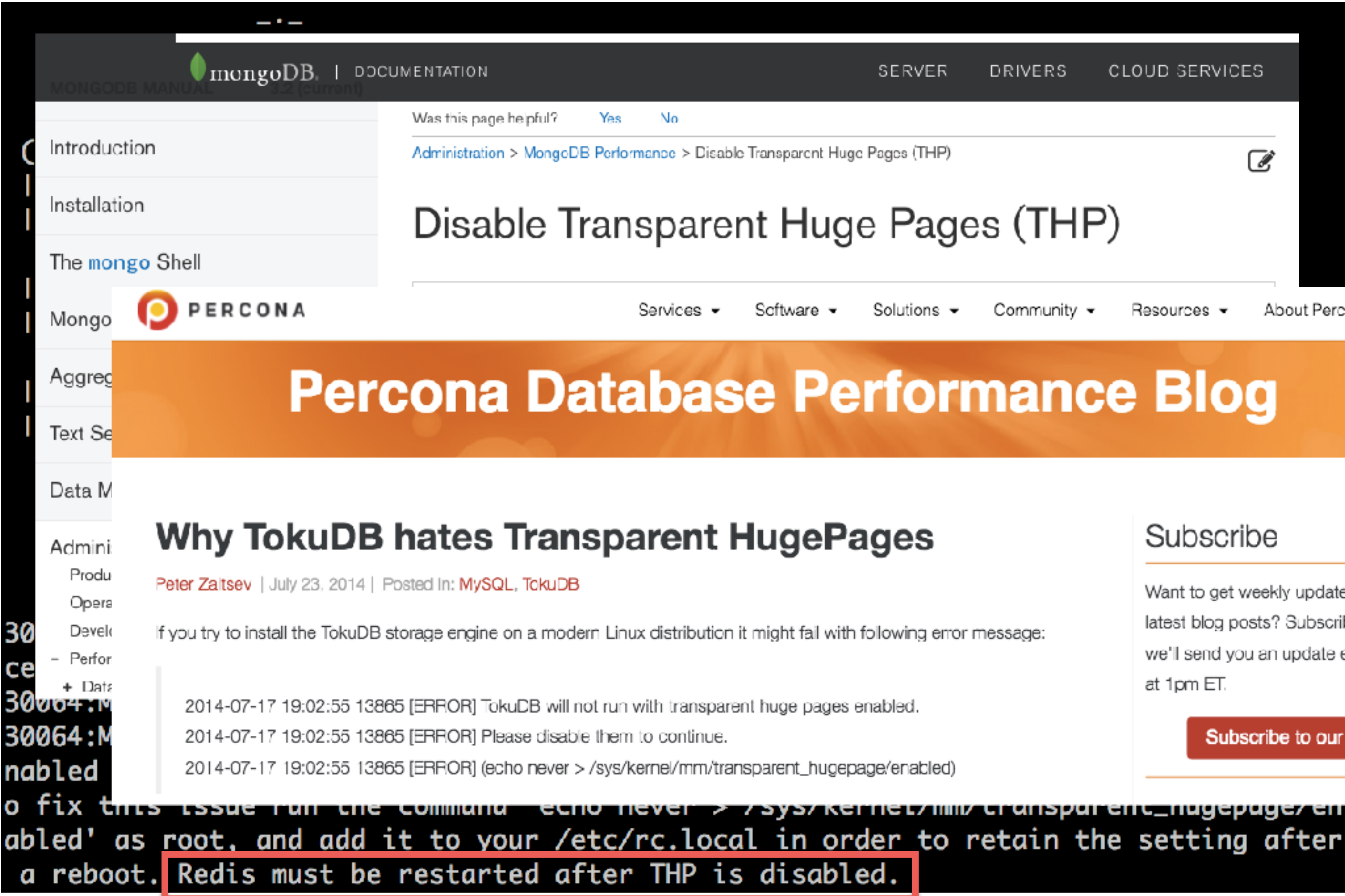
On this page

- Init Script
- Using tuned and ktune
- Test Your Changes

Transparent Huge Pages (THP) is a Linux memory management system that reduces the overhead of Translation Lookaside Buffer (TLB) lookups on machines with large amounts of memory by using larger memory pages.

However, database workloads often perform poorly with THP, because they tend to have sparse rather than contiguous memory access patterns. You should disable THP on Linux machines to ensure best performance with MongoDB.

```
30064:M 04 Aug 17:19:08.927 # WARNING you have Transparent Huge Pages (THP) support e
nabled in your kernel. This will create latency and memory usage issues with Redis. T
o fix this issue run the command 'echo never > /sys/kernel/mm/transparent_hugepage/en
abled' as root, and add it to your /etc/rc.local in order to retain the setting after
 a reboot. Redis must be restarted after THP is disabled.
```


mongoDB. | DOCUMENTATION
SERVER
DRIVERS
CLOUD SERVICES
Was this page helpful? Yes No
Administration > MongoDB Performance > Disable Transparent Huge Pages (THP)
Disable Transparent Huge Pages (THP)
Introduction
Installation
The mongo Shell
PERCONA
Services
Software
Solutions
Community
Resources
Percona Database Performance Blog
Why TokuDB hates Transparent HugePages
Peter Zaitsev | July 23, 2014 | Posted In: MySQL, TokuDB
If you try to install the TokuDB storage engine on a modern Linux distribution it might fail with following error message:
2014-07-17 19:02:55 13865 [ERROR] TokuDB will not run with transparent huge pages enabled.
2014-07-17 19:02:55 13865 [ERROR] Please disable them to continue.
2014-07-17 19:02:55 13865 [ERROR] (echo never > /sys/kernel/mm/transparent_hugepage/enabled)
Subscribe
Subscribe to our
abled' as root, and add it to your /etc/rc.local in order to retain the setting after
a reboot. Redis must be restarted after THP is disabled.

Disable Transparent Huge Pages (THP)

2.3.2. Disable Transparent Huge Pages

IBM Support

splunk> docs PRODUCTS SOLUTIONS CUSTOMERS COMMUNITY SPLEXICON Support & Services My Account

cloudera Why Cloudera Products Services & Support Solutions

okta PRODUCT DOCS DISCUSSION SUPPORT

Transparent Huge Pages: Thanks for your help...please don't help

By the next morning CPU contention was worse.

The alarmingly high system CPU usage that we'd seen in the previous 3 months was always due to MySQL using kernel mutex. But sin
problem, *what the heck was this?*

We discussed turning off TCMalloc, but that would've been a mistake. Implementing TCMalloc was a critical link in the chain of problem
ultimately strengthened our platform.

We discovered very quickly that the culprit this time was a *khugepaged* enabled by a Linux kernel flag called Transparent Huge Pages
default in most Linux distributions). Huge pages are designed to improve performance by helping the operating system manage large
They effectively increase the page size from the standard 4kb to 2MB or 1Gb (depending on how it is configured).

THP makes huge pages easier to use by, among other things, arranging your memory into larger chunks. It works great for app servers
memory-intensive operations.

- High Availability
- Backup and Disaster Recovery
- Cloudera Manager Administration
- Cloudera Navigator Data Management Component Administration

Disabling Transparent Hugepage Compaction

Most Linux platforms supported by CDH 5 include a feature called **transparent hugepage compaction** whic
interacts poorly with Hadoop workloads and can seriously degrade performance.

Huge page pathologies in Linux

- High page fault latency
- Memory bloating
- Unfair huge page allocation
- Uncoordinated memory management

Huge page pathologies in Linux

- High page fault latency
- Memory bloating
- Unfair huge page allocation
- Uncoordinated memory management

Ingens

Efficient huge page management system

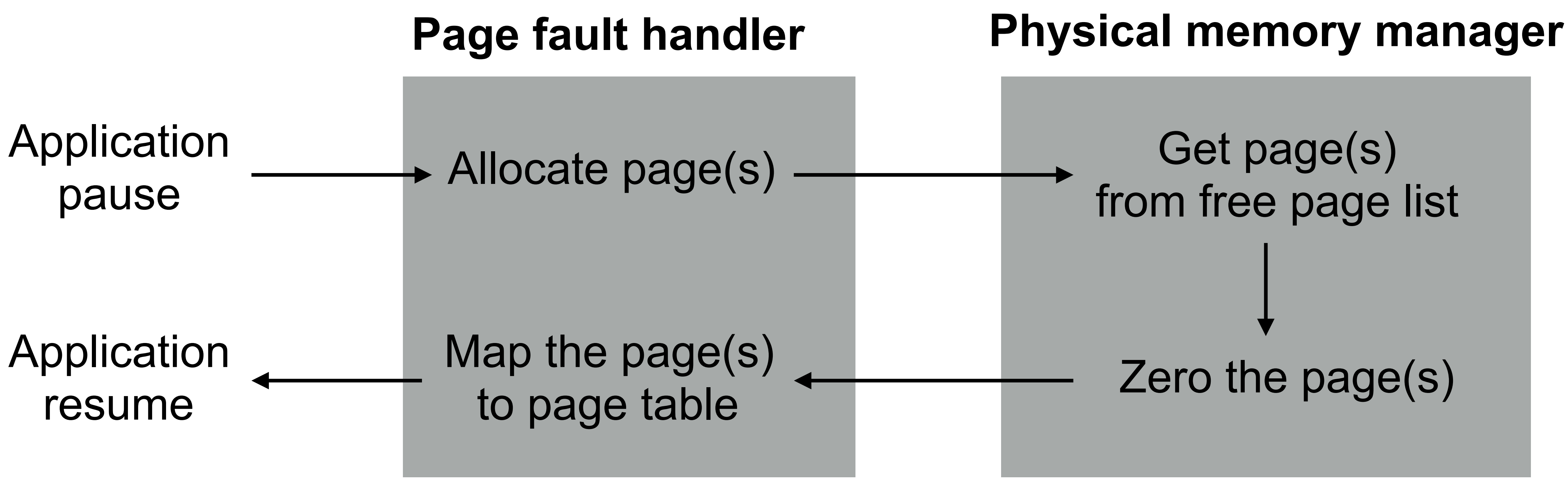
How to allocate huge pages?

Problems		Linux	Ingens
High page fault latency	←	Synchronous allocation	Asynchronous allocation
Memory bloating	←	Greedy allocation	Spatial utilization based allocation

High page fault latency

Huge page allocation increases page fault latency

- Page allocation path of both base and huge page

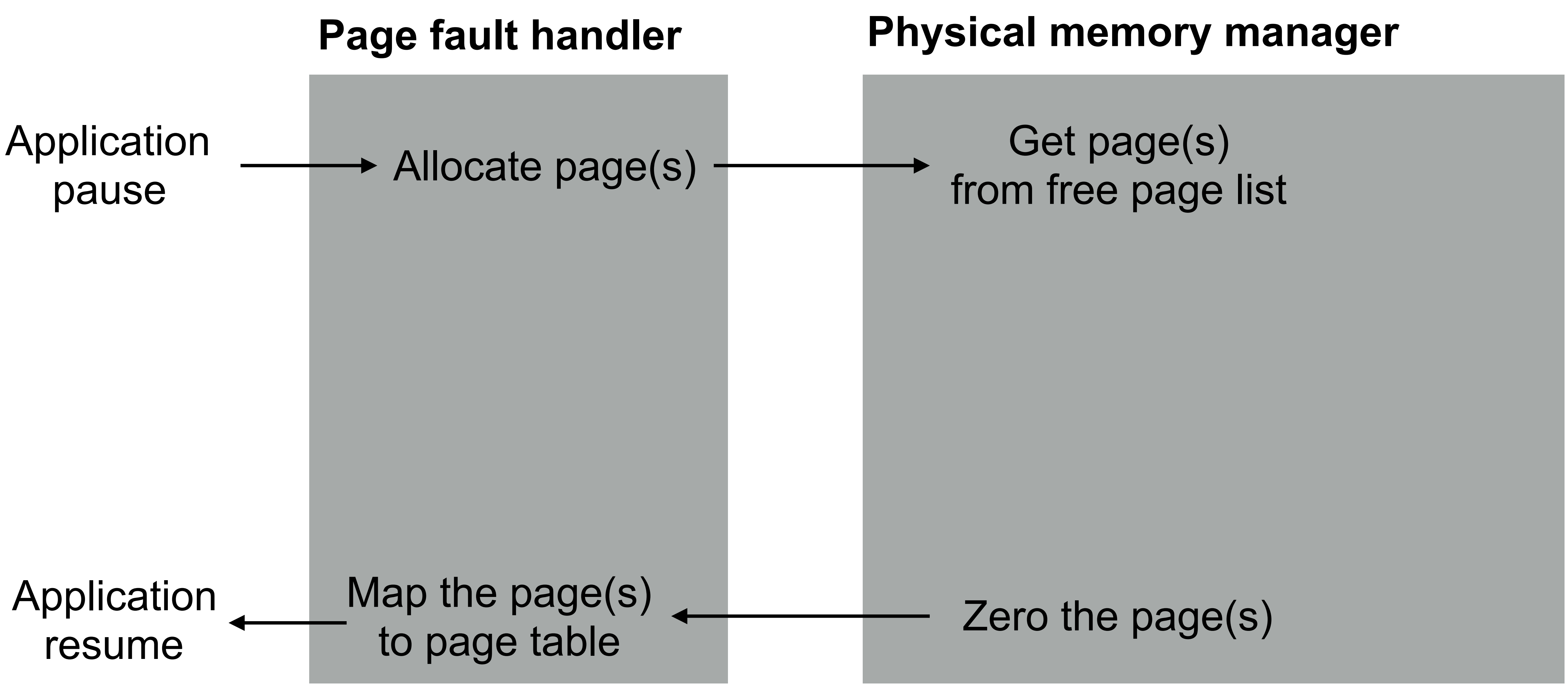


Page fault latency

- 4KB page : 3.6 us
- 2MB page : 378.0 us (mostly from page zeroing)
- Increases tail latency

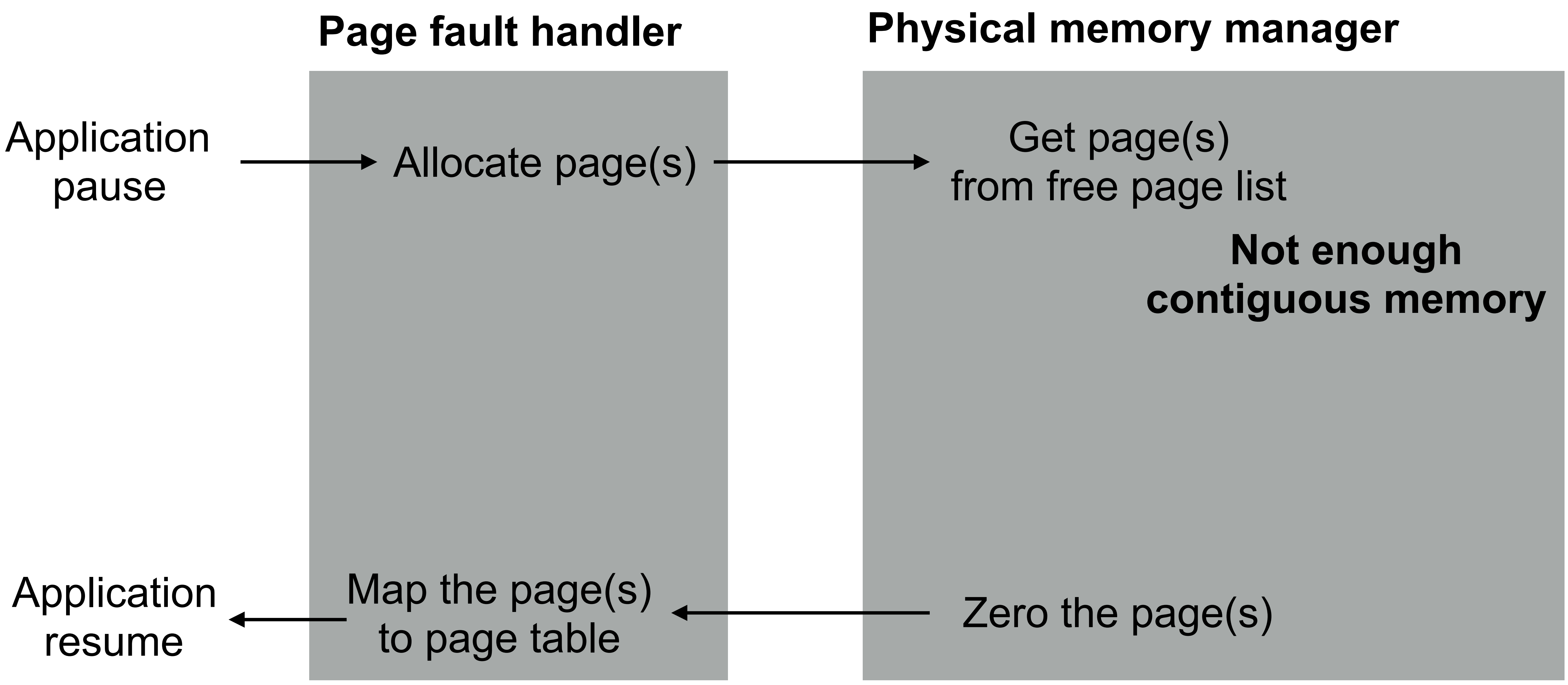
Huge page allocation might require extra memory copying

- Page allocation path of huge page



Huge page allocation might require extra memory copying

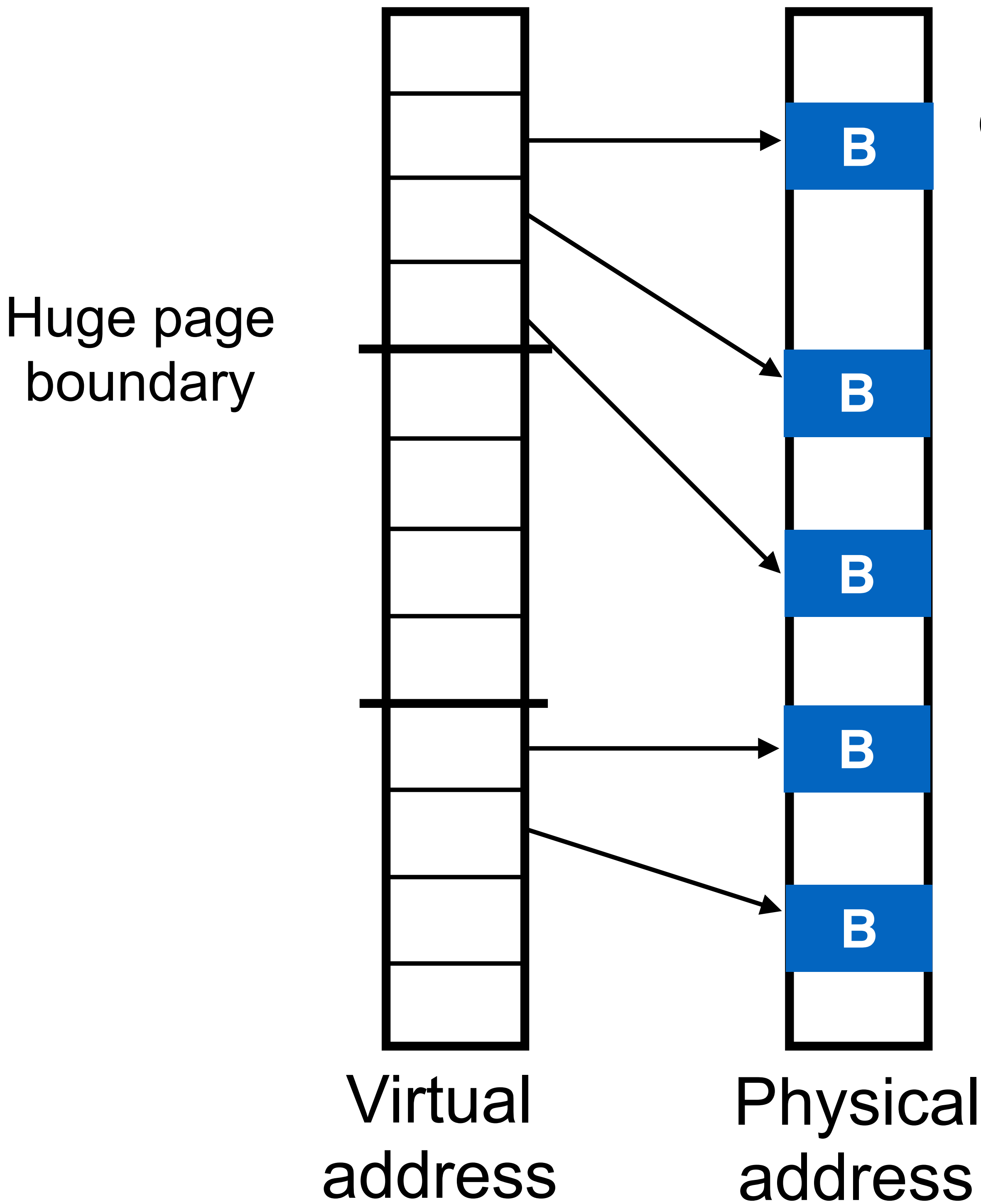
- Page allocation path of huge page



External fragmentation

Not enough
contiguous memory

External fragmentation

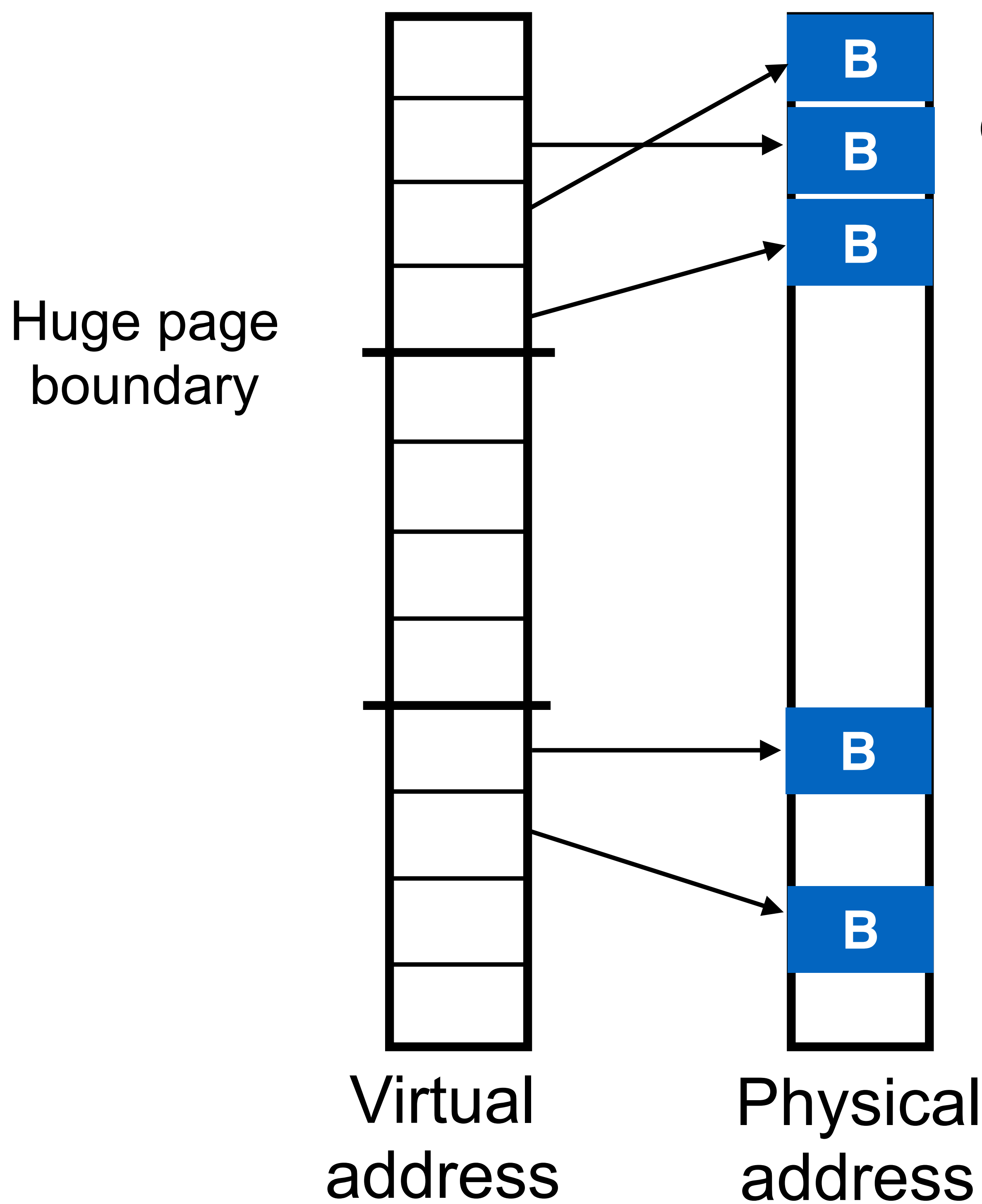


Not enough contiguous memory

- As system ages, physical memory is fragmented
  - 2 minutes to fragment 24 GB
  - All memory sizes eventually fragment
- Linux compacts physical memory to create contiguous pages

Allocated Base page

External fragmentation

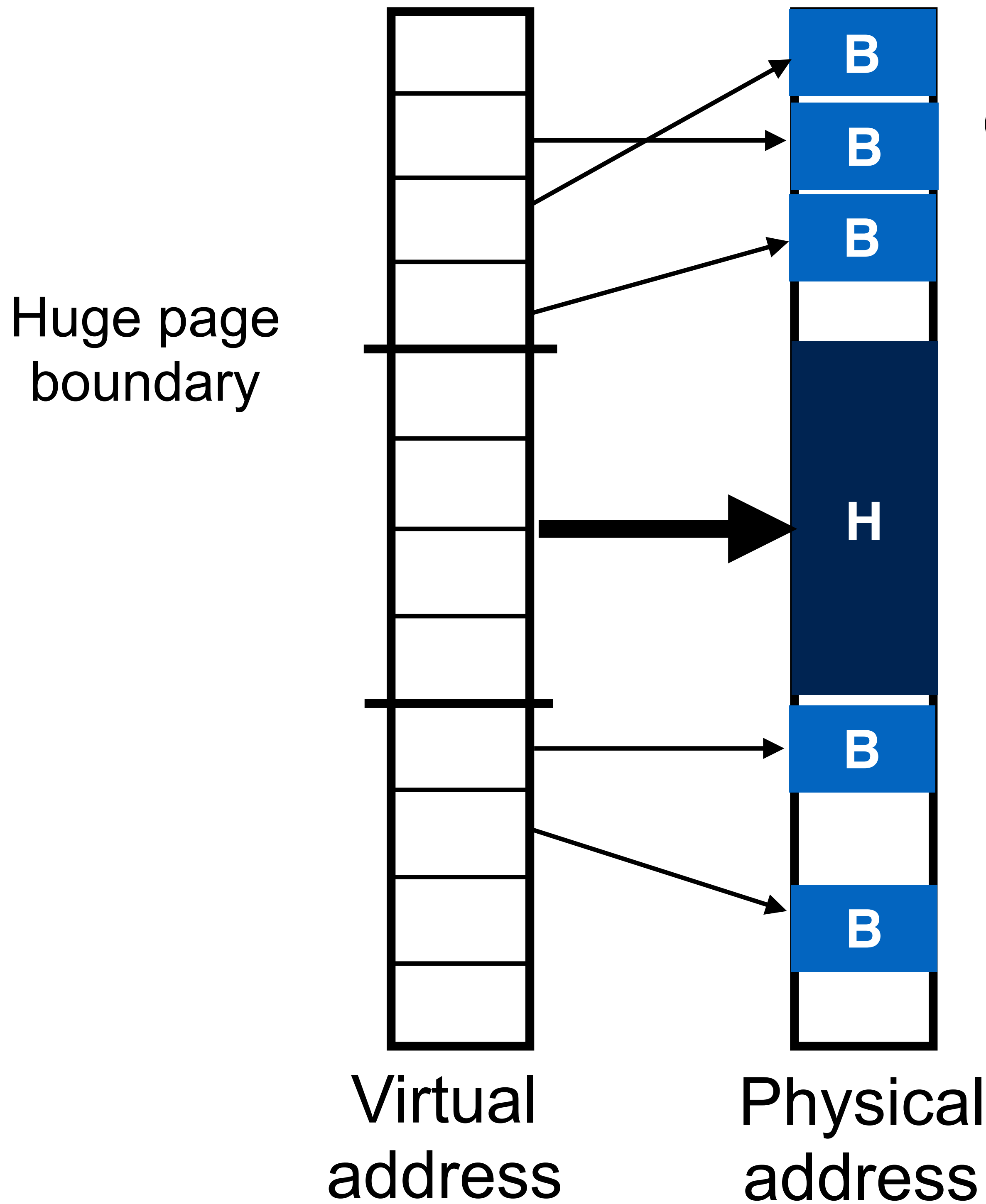


Not enough contiguous memory

- As system ages, physical memory is fragmented
  - 2 minutes to fragment 24 GB
  - All memory sizes eventually fragment
- Linux compacts physical memory to create contiguous pages

B Allocated Base page

External fragmentation



Not enough contiguous memory

- As system ages, physical memory is fragmented
    - 2 minutes to fragment 24 GB
    - All memory sizes eventually fragment
- Linux compacts physical memory to create contiguous pages

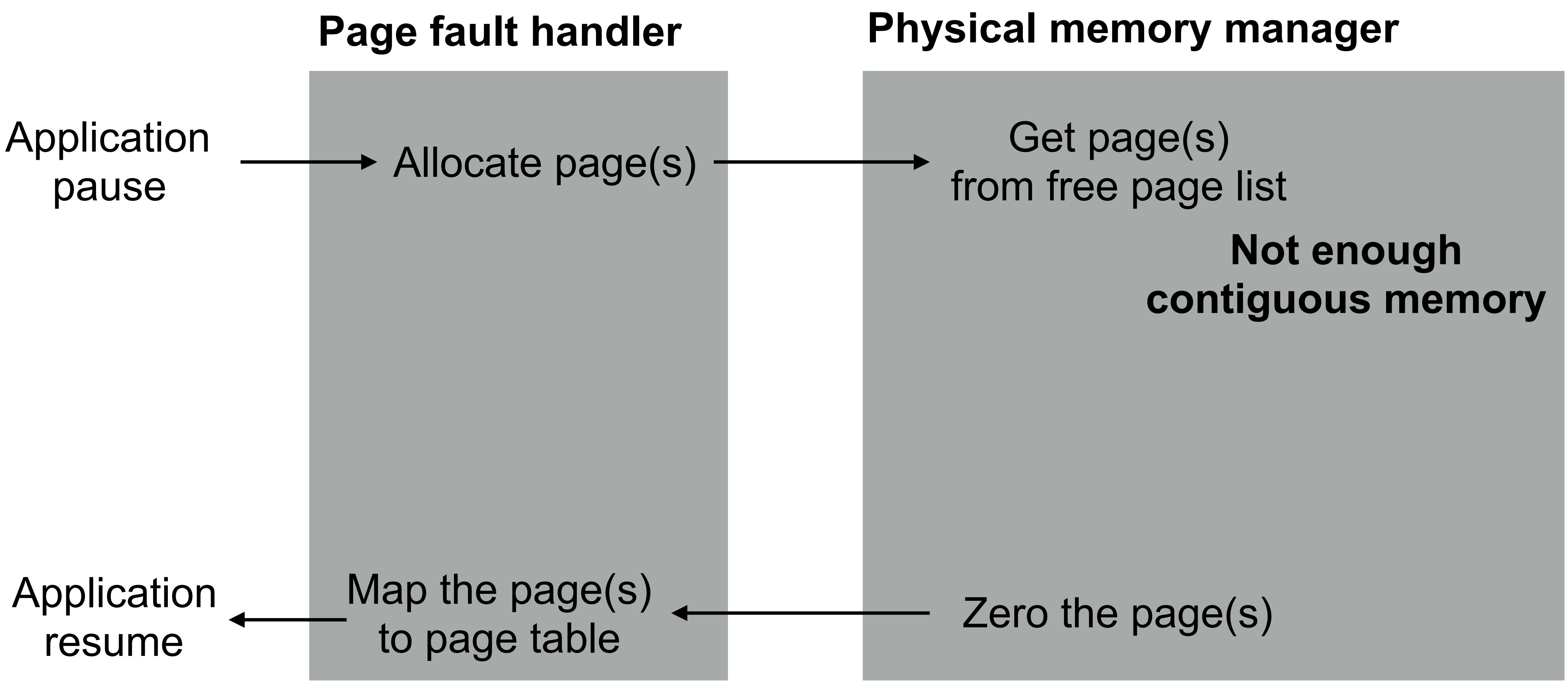
B Allocated Base page

External fragmentation

Not enough contiguous memory

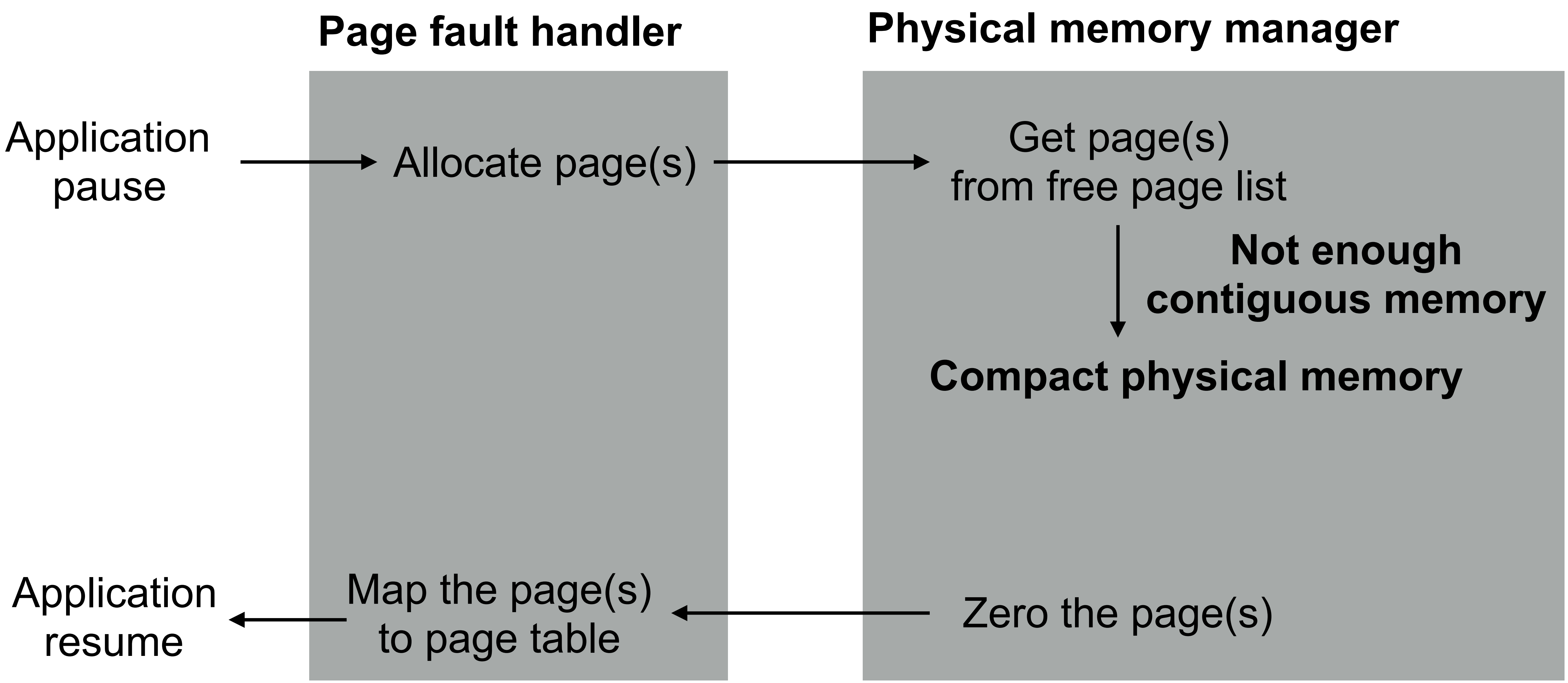
Huge page allocation might require extra memory copying

- Page allocation path of huge page includes memory compaction



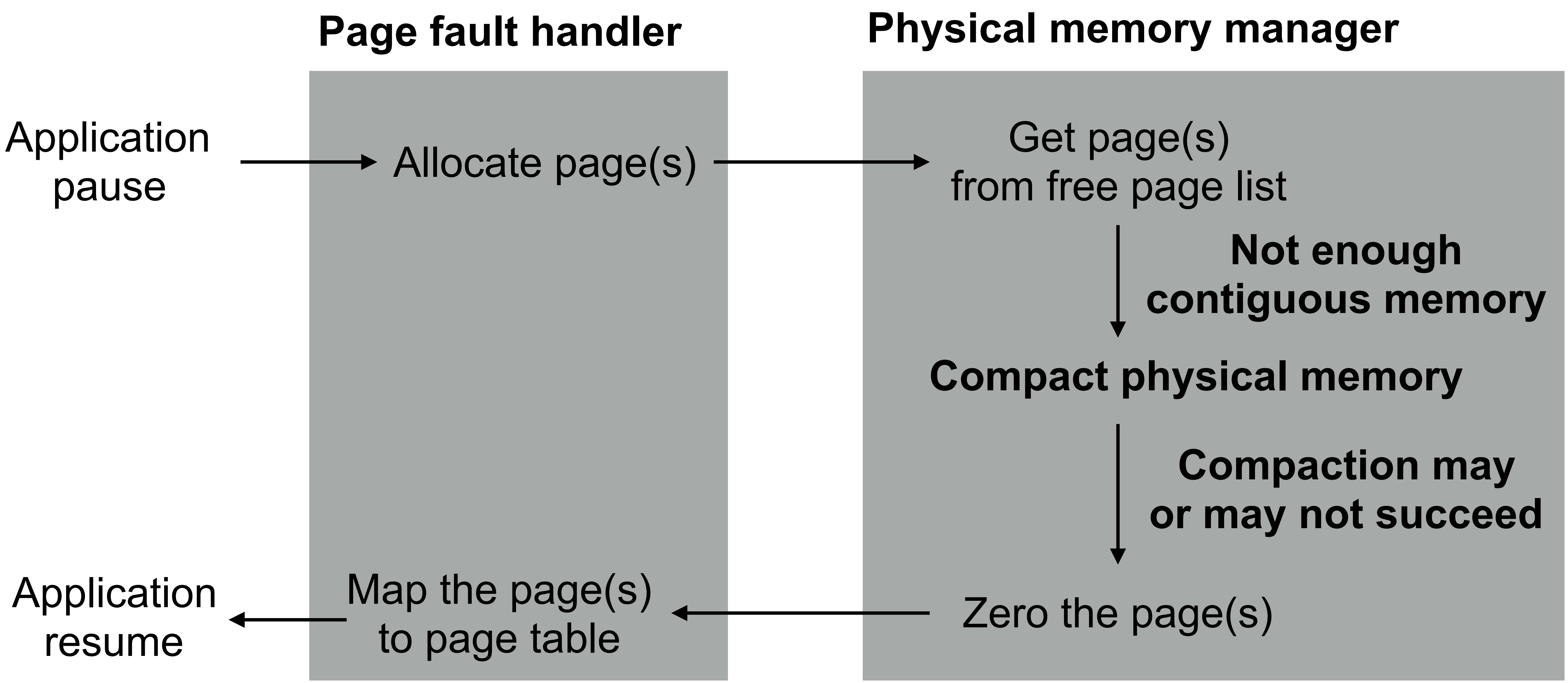
Huge page allocation might require extra memory copying

- Page allocation path of huge page includes memory compaction

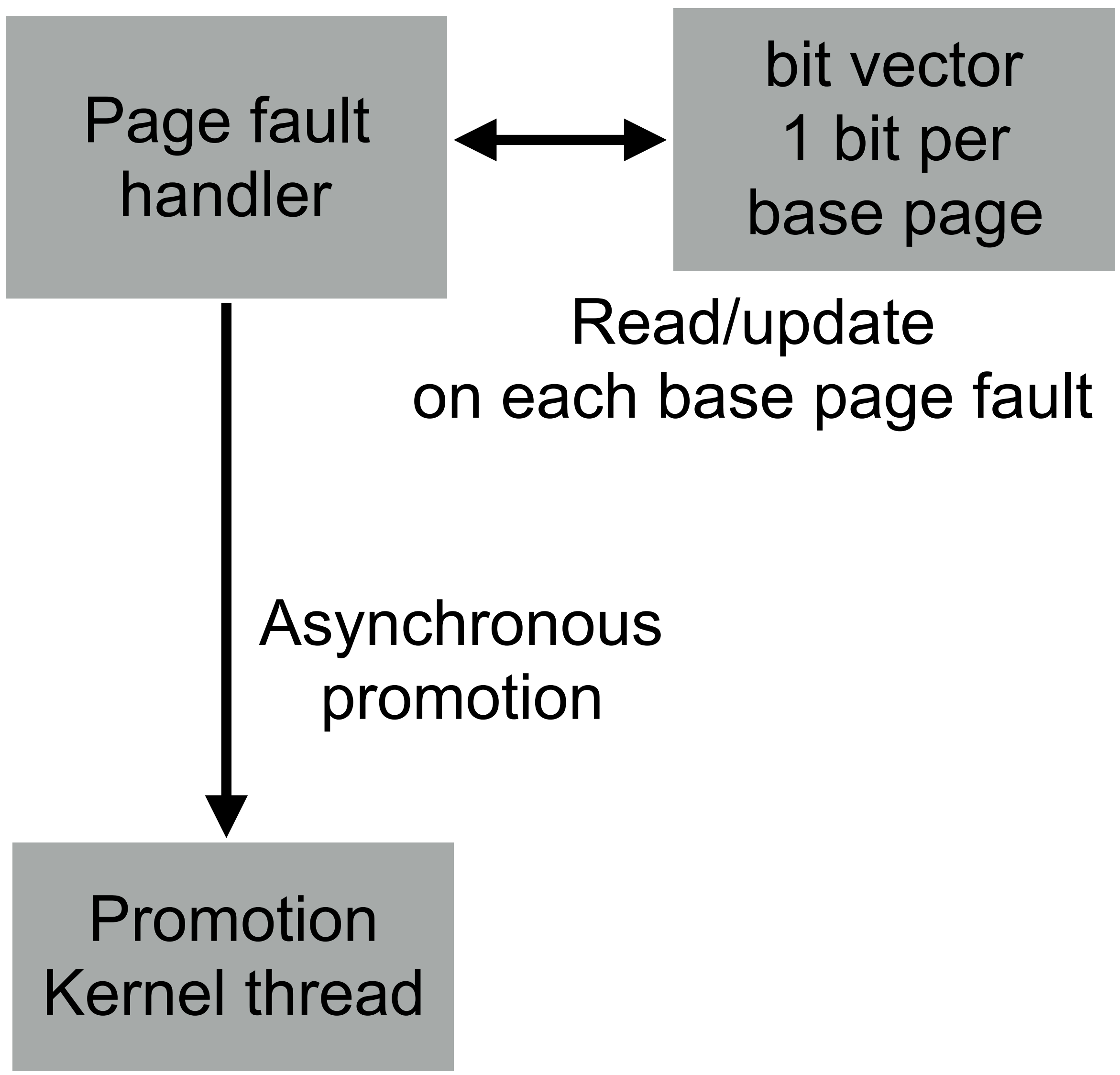


Huge page allocation might require extra memory copying

- Page allocation path of huge page includes memory compaction



Ingens: asynchronous allocation



- Page fault handler **only** allocates base pages
- Huge page allocation in background
- Memory compaction in background
- No extra page fault latency
  - No huge page zeroing
  - No compaction

Fast page fault handling

Page fault latency experiment

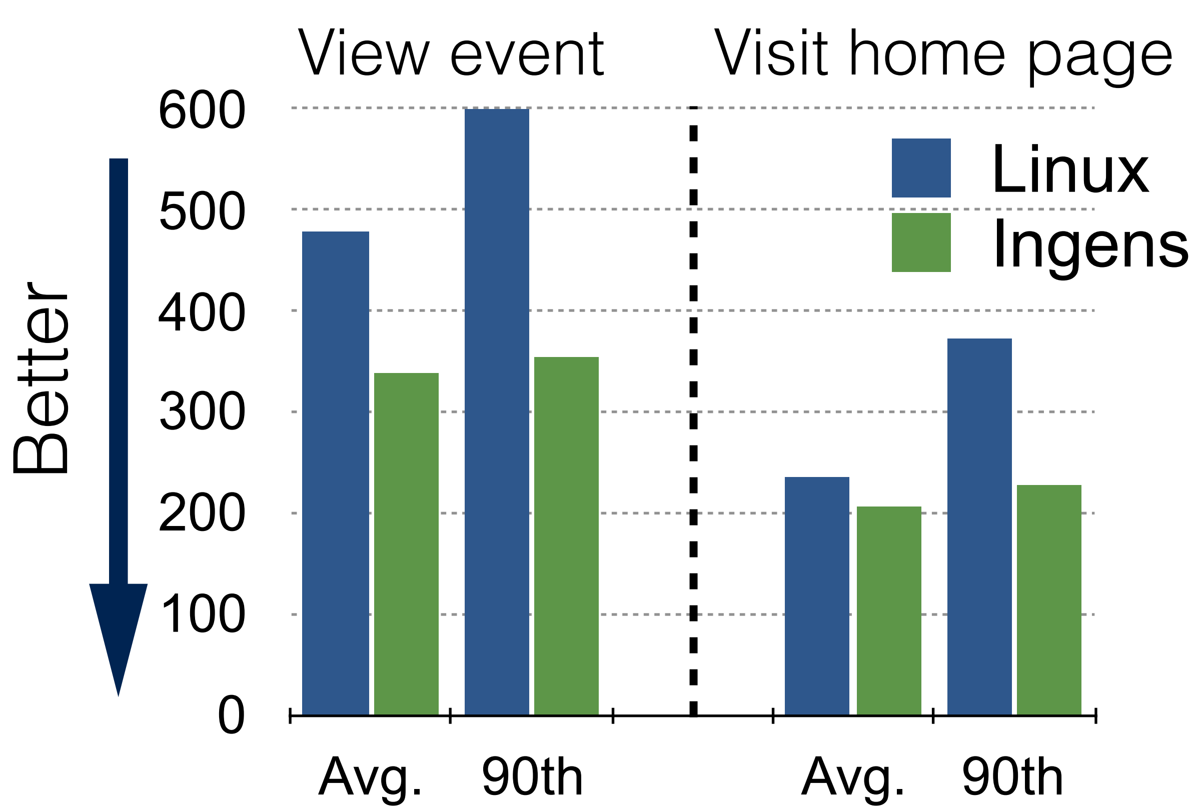
- Machine specification
  - Two Intel Xeon E5-2640 2.60GHz CPUs
  - 64GB memory and two 250 MB SSDs
- Cloudstone workload (latency sensitive)
  - Web service for social event planning
  - nginx/PHP/MySQL running in virtual machines
  - 85% read, 10% login, 5% write workloads
  - 2 of 7 web pages modified to use modern web page sizes
    - The average web page is 2.1 MB
      https://www.soasta.com/blog/page-bloat-average-web-page-2-mb/

Cloudstone result

Throughput (requests/s)

Linux	Ingens
922.3	1091.9 (+18%)

Latency (millisecond)



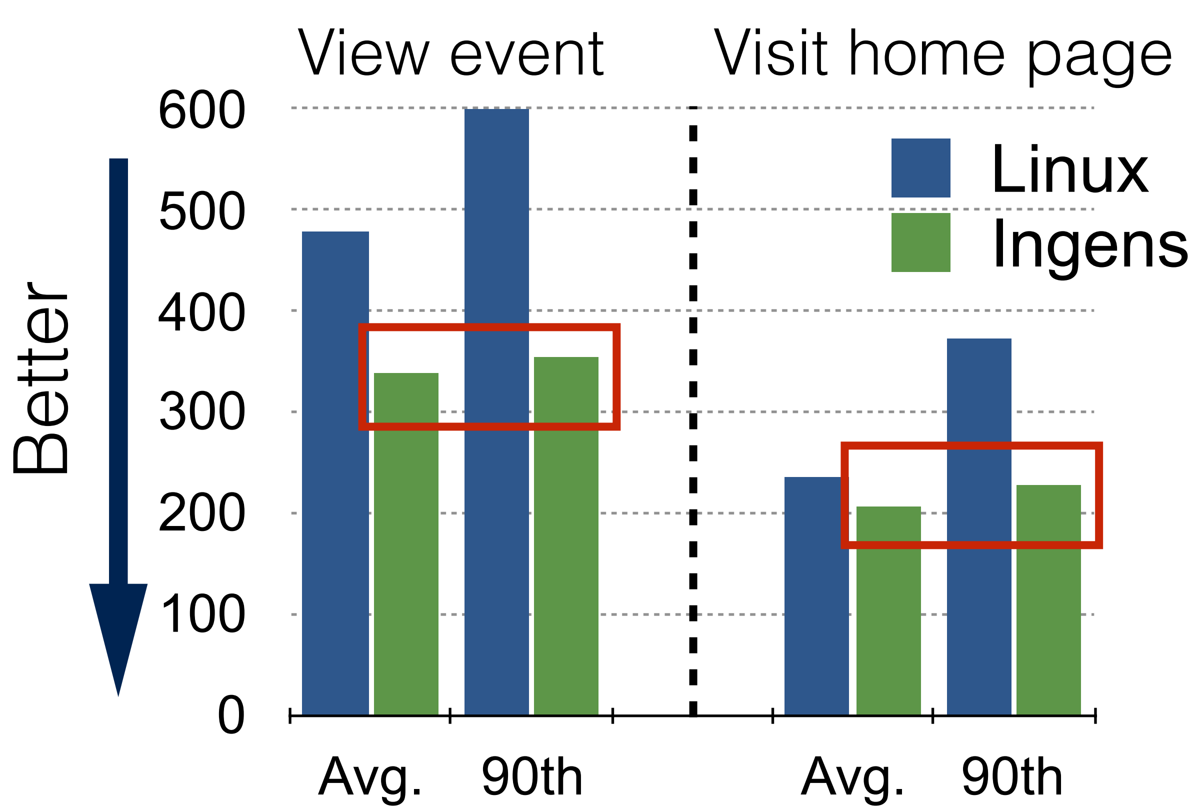
- Memory is highly fragmented
- Ingens reduces
  - average latency up to 29.2%
  - tail latency up to 41.4%
- Linux page fault handler performs 461,383 memory compactions

Cloudstone result

Throughput (requests/s)

Linux	Ingens
922.3	1091.9 (+18%)

Latency (millisecond)

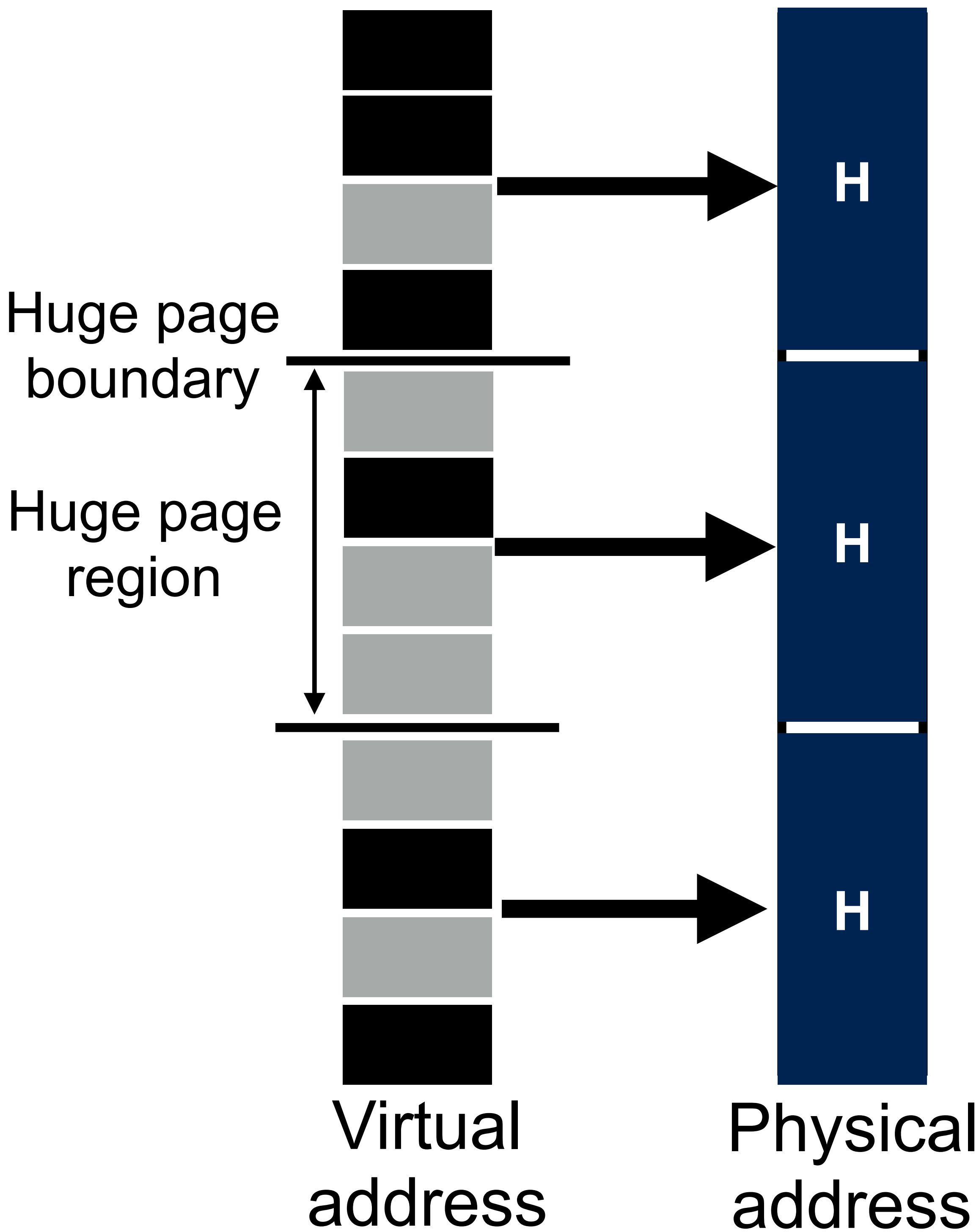


- Memory is highly fragmented
- Ingens reduces
  - average latency up to 29.2%
  - tail latency up to 41.4%
- Linux page fault handler performs 461,383 memory compactions

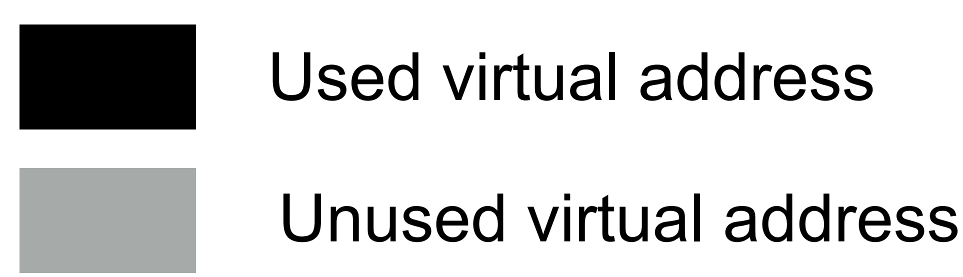
Memory bloating

Application occupies more memory than it uses

Internal fragmentation



- Greedy allocation in Linux
  - Allocate a huge page on first fault to huge page region
  - The huge page region may not be fully used
- Greedy allocation causes severe internal fragmentation
  - Memory use often sparse



Memory bloating experiment

- Redis
  - Delete 70% objects after populating 8KB objects
- MongoDB
  - 15 million get requests for 1KB object with YCSB

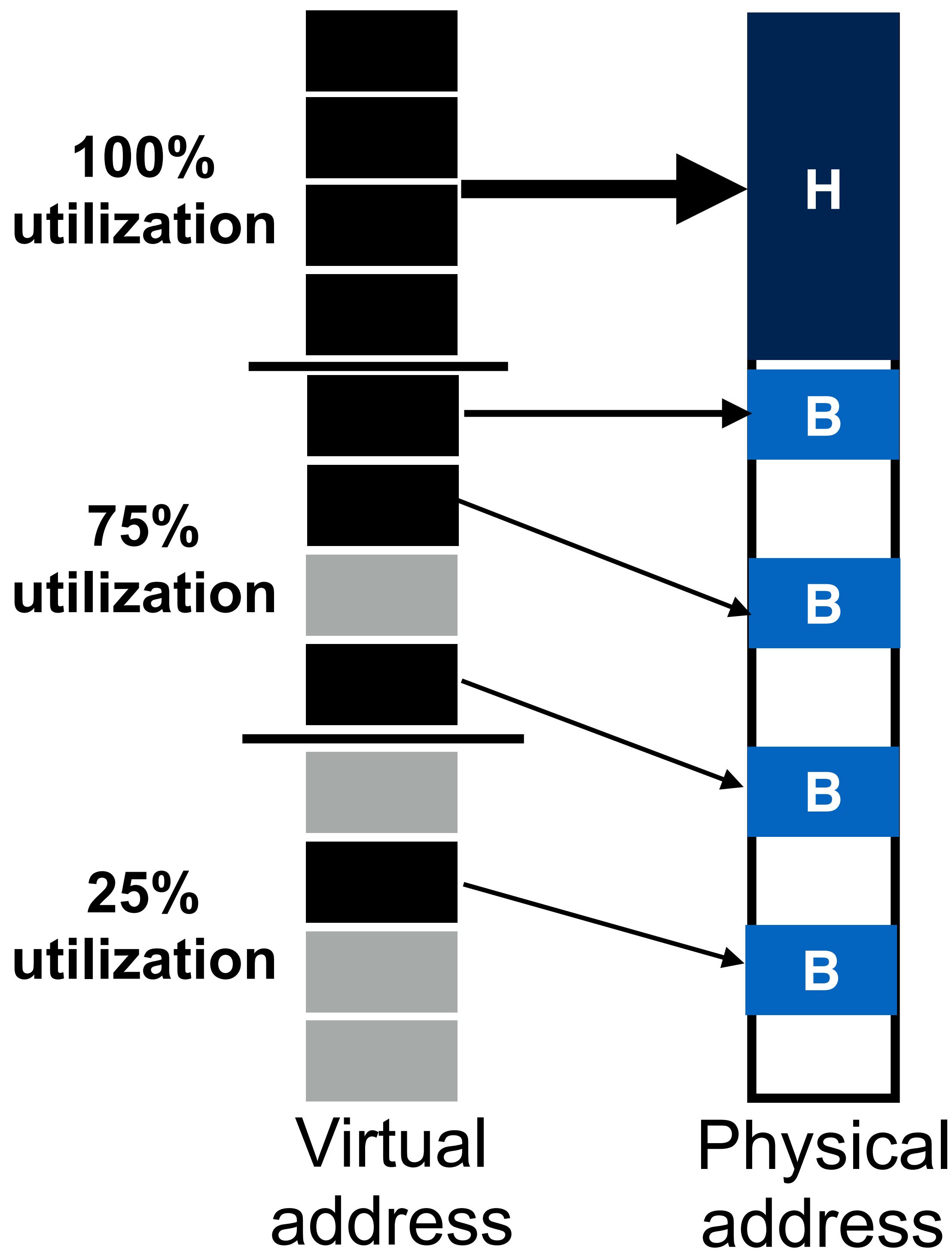
Physical memory consumption

	Using huge page	Using only base page
Redis	20.7GB (+69%)	12.2GB
MongoDB	12.4GB (+23%)	10.1GB

Bloating makes memory consumption unpredictable

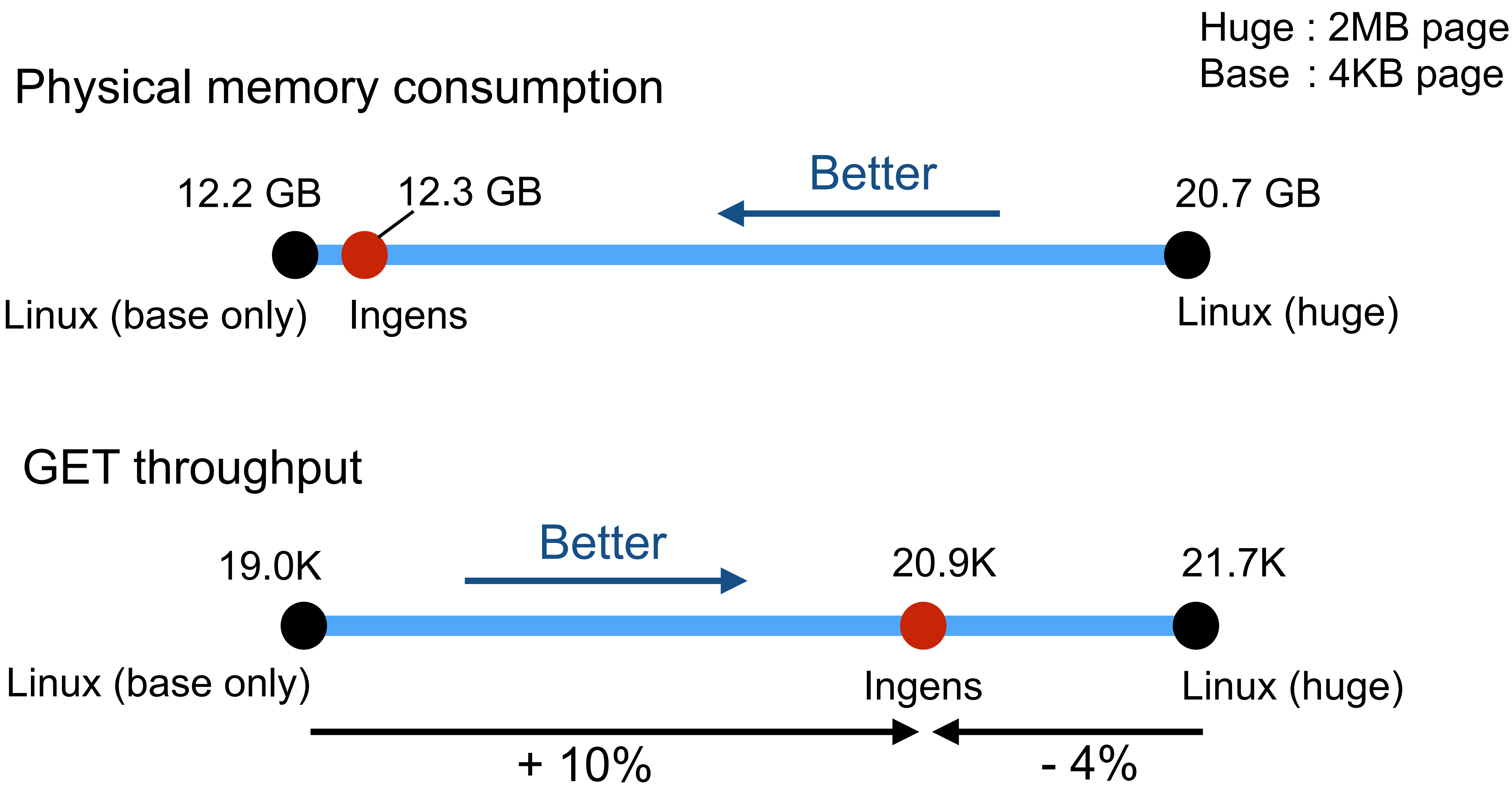
Memory-intensive applications can't provision to avoid swap

Ingens: Spatial utilization based allocation



- Ingens monitors spatial utilization of each huge page region
- Utilization-based allocation
  - Page fault handler requests promotion when the utilization is beyond a threshold (e.g., 90%)
  - Bounds the size of internal fragmentation

Redis memory bloating experiment



Ingens overhead

- Overhead for memory intensive application

429.mcf	Graph	Spark	Canneal	SVM	Redis	MongoDB
0.9%	0.9%	0.6%	1.9%	1.3%	0.2%	0.6%

- Overhead for non-memory intensive application

Kernel build	Grep	Parsec 3.0 Benchmark
0.2%	0.4%	0.8%

Ingens overhead is negligible

Ingens

Make huge pages widely used in practice

Linux	Ingens		Advantages
Synchronous allocation	Asynchronous allocation	→	No extra page fault latency
Greedy allocation	Spatial utilization based allocation	→	Bound memory bloating

Source code is available at

https://github.com/ut-osa/ingens

Backup slides

Other operating systems

- Window, MacOS
  - Does not support transparent huge page
- FreeBSD
  - Very conservative approach
  - No memory compaction functionality
  - Performance speedup in Linux and FreeBSD

	SVM	Canneal	Redis
FreeBSD	1.28	1.13	1.02
Linux	1.30	1.21	1.15
Ingens	1.29	1.19	1.15

Operating system support for huge pages

- User-controlled huge page management
  - Admin reserves huge page in advance
  - New APIs for memory allocation/deallocation
  - It could fail to reserve huge pages when memory is fragmented
- Transparent huge page management
  - Developers do not know about huge page
  - OS Transparently allocates/deallocates huge pages
  - OS manages memory fragmentation