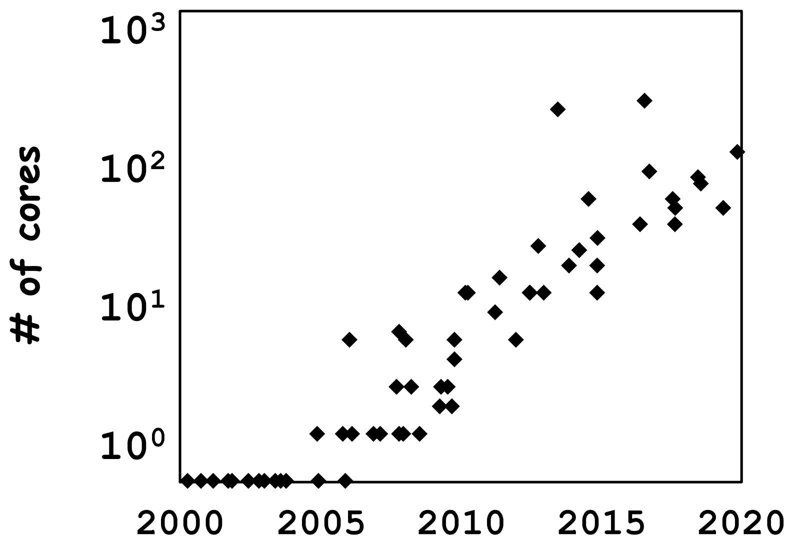
LODIC : Logical Distributed Counting for Scalable File Access

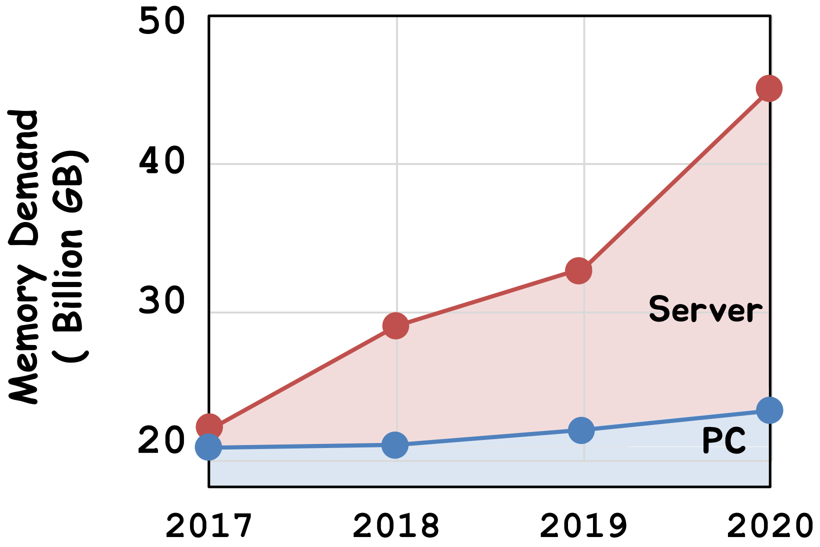
Jeoungahn Park , Taeho Hwang, Jongmoo Choi, Changwoo Min, Youjip Won

Background

- The number of cores is rapidly increasing
- Main memory is getting larger and larger
- Manycore scalability becomes a serious issue in the modern OS design



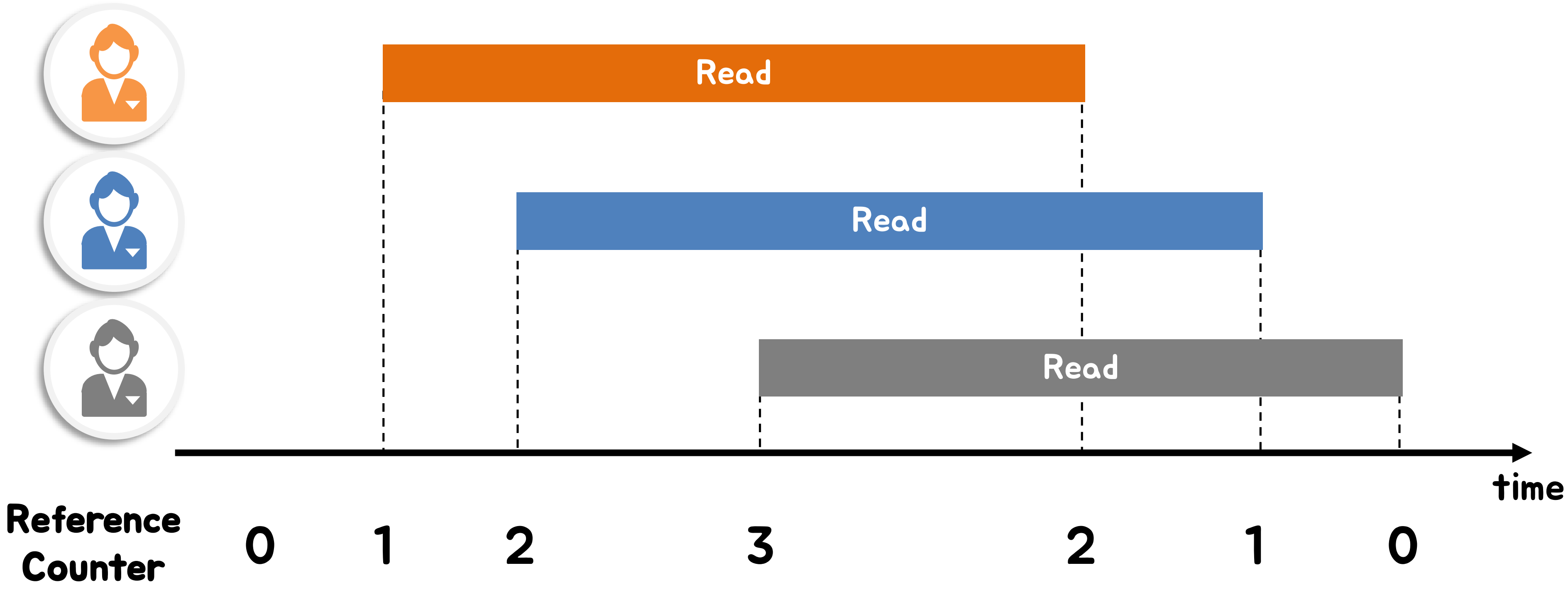
M. Horowitz et al. at MIT (~2010)
K. Rupp's Github (2010 ~ 2019)



IHS Markit, 2020

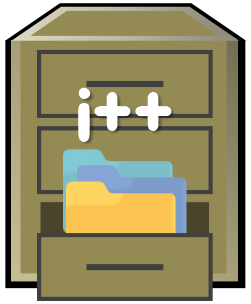
Reference counter

- Number of accesses for a given object



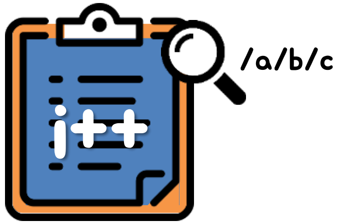
Reference counter in the kernel object

File descriptor table
`struct files_struct`

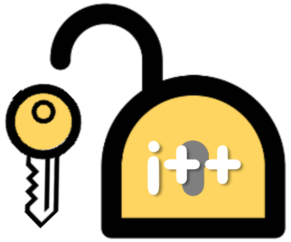


File object
`struct file`

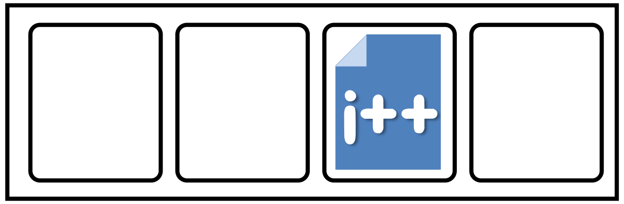
Directory entry
`struct dentry`



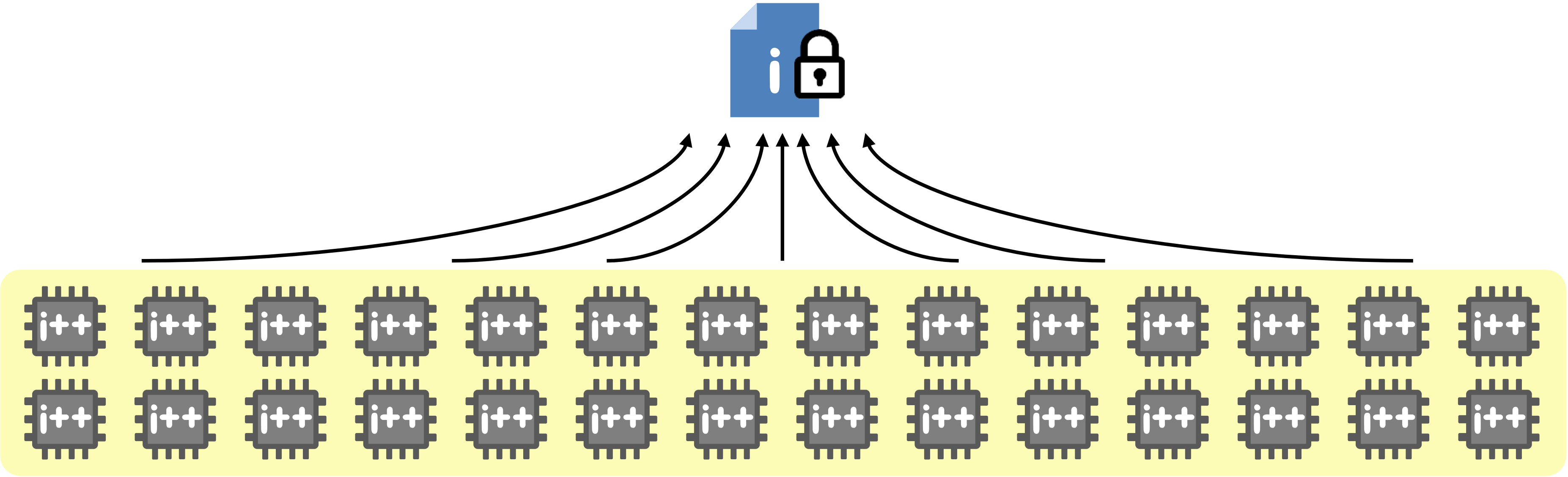
read-write semaphore
`struct rw_semaphore`

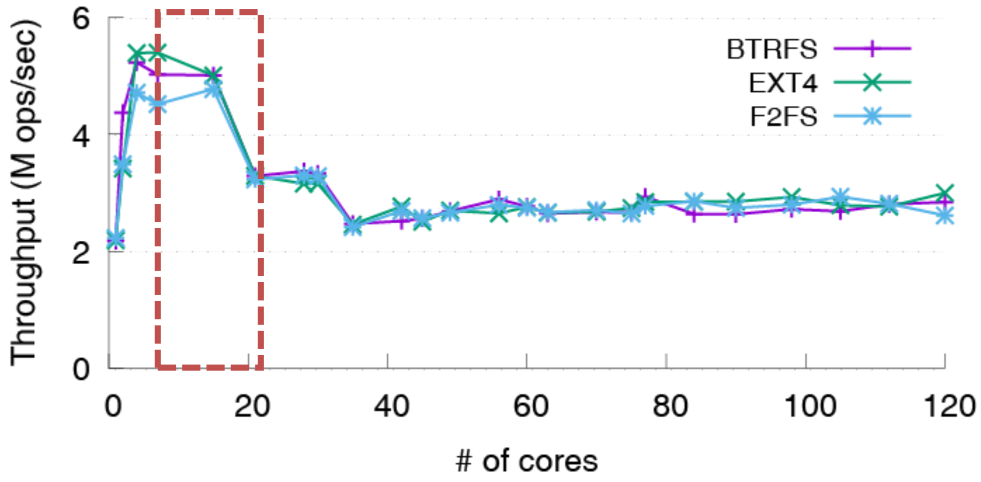


Physical page frame
`struct page`



Manycore scalability in kernel object access



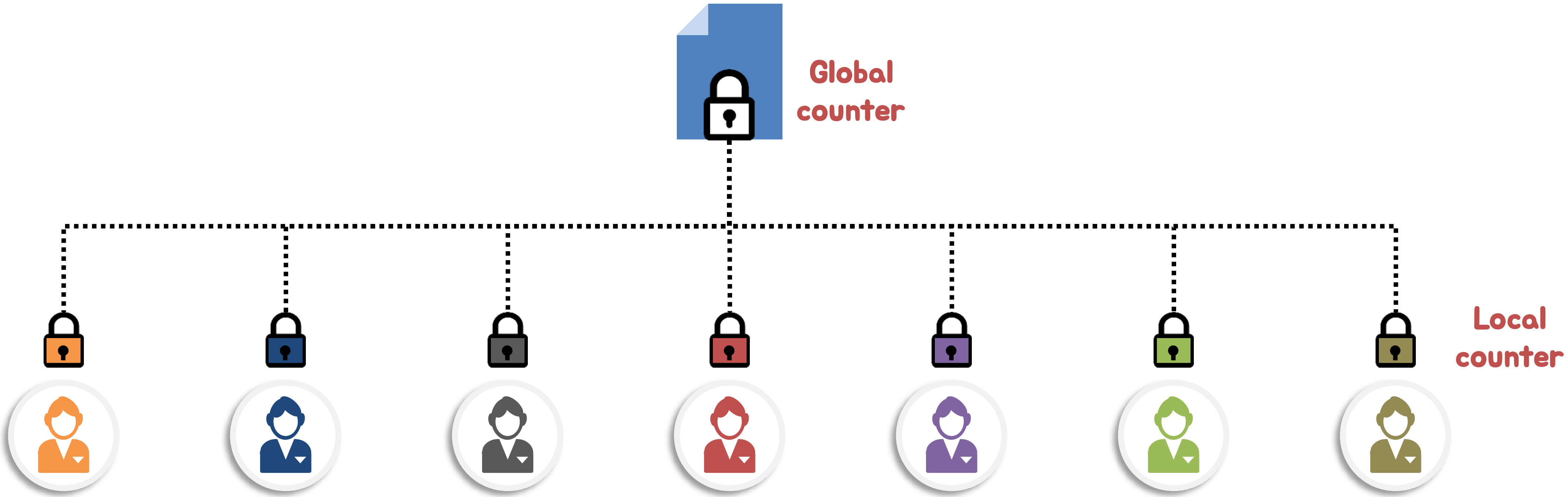


Performance collapse due to
cacheline contention

KAIST VIRGINIA TECH.

Distributed reference counter

- Allocate local counter for each core
- Update operation : update the local counter
- Counter query : scan all local counters

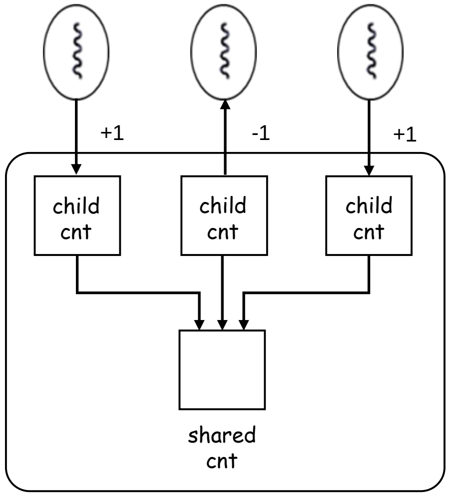


Issues in distributed reference counter

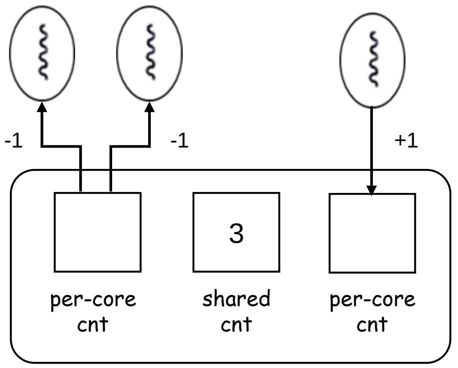
- Memory pressure
  - Memory overhead increase in proportion to number of CPUs and objects
- Query latency
  - For reclaim the object, checking all local counter increase query latency
  - Overhead of obtaining the global state of the counter

Existing works for per-core reference counter

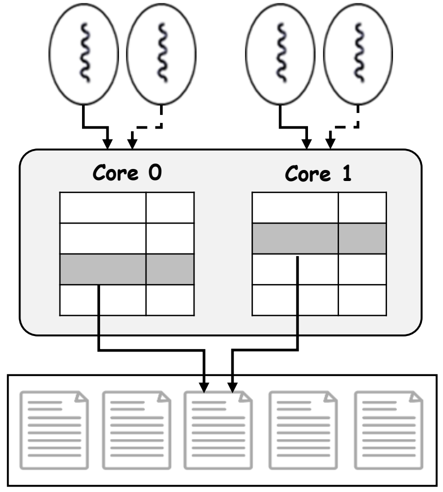
SNZI [PODC'07]



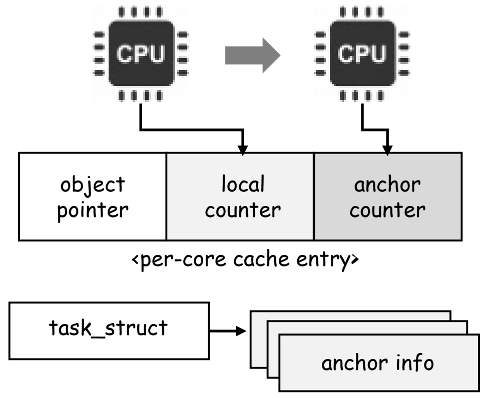
Sloppy Counter [OSDI'10]



RefCache [EuroSys'13]



PayGo [FAST'19]



SNZI: Memory overhead

Sloppy Counter: Memory overhead

Query overhead

RefCache: Overhead of handling hash collision

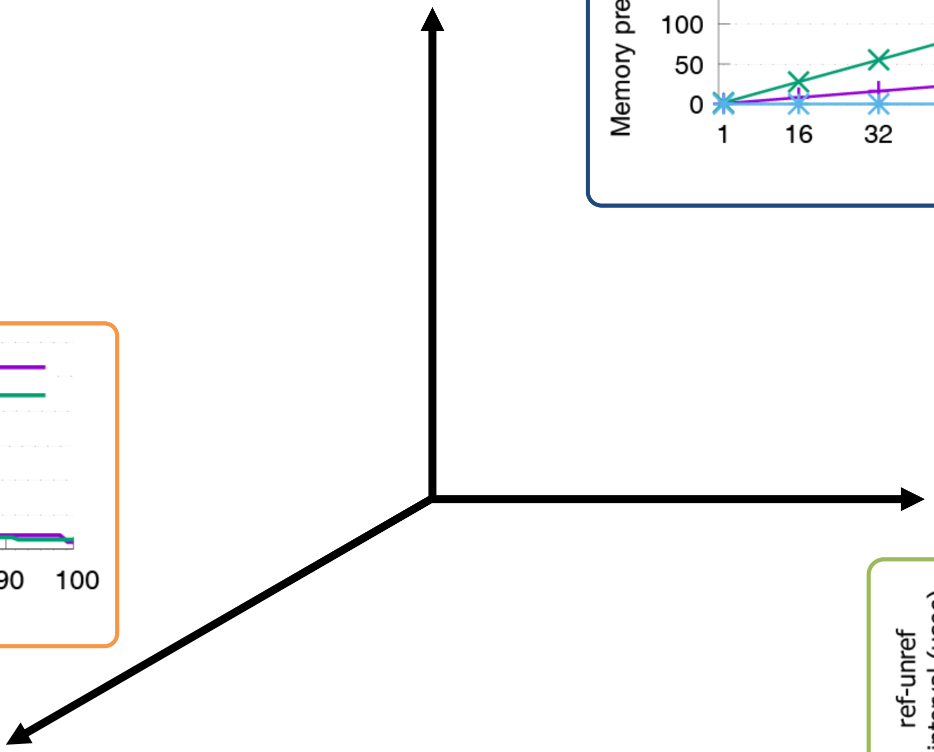
Memory overhead for counter cache

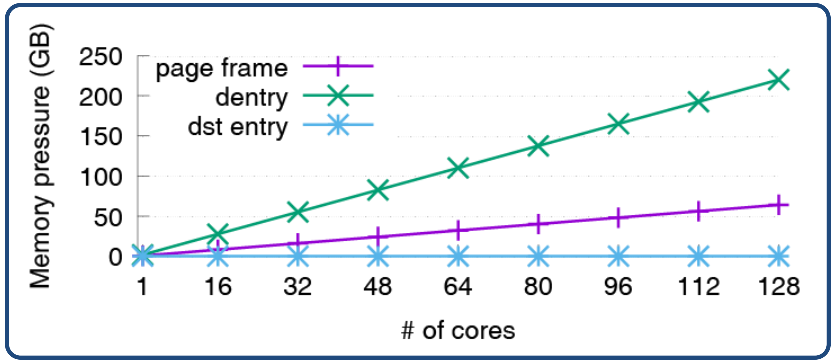
PayGo: Query overhead

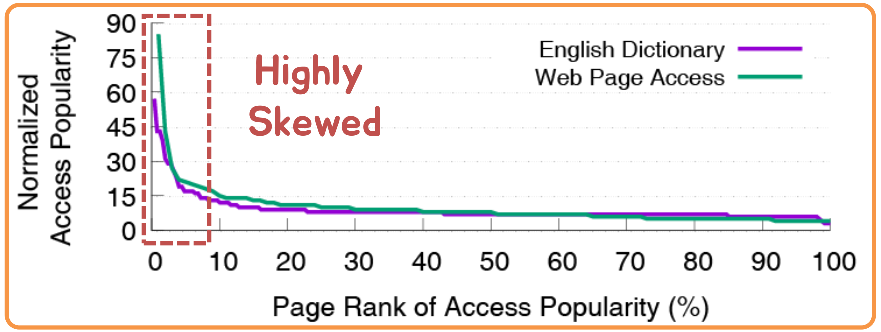
Memory overhead for counter cache

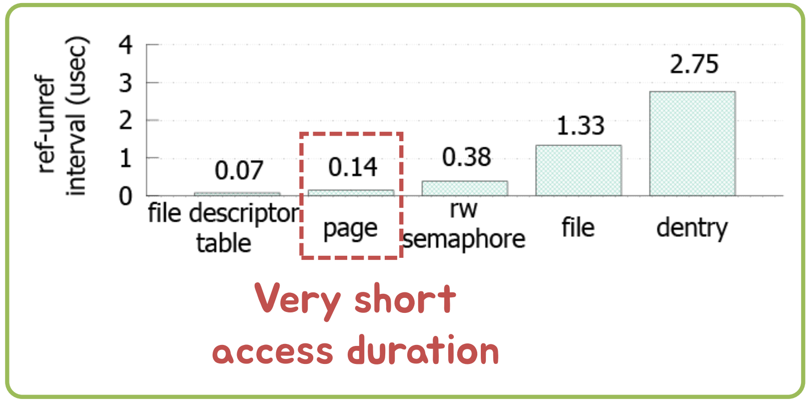
Design of Logical Distributed Counter

Characteristics of kernel objects



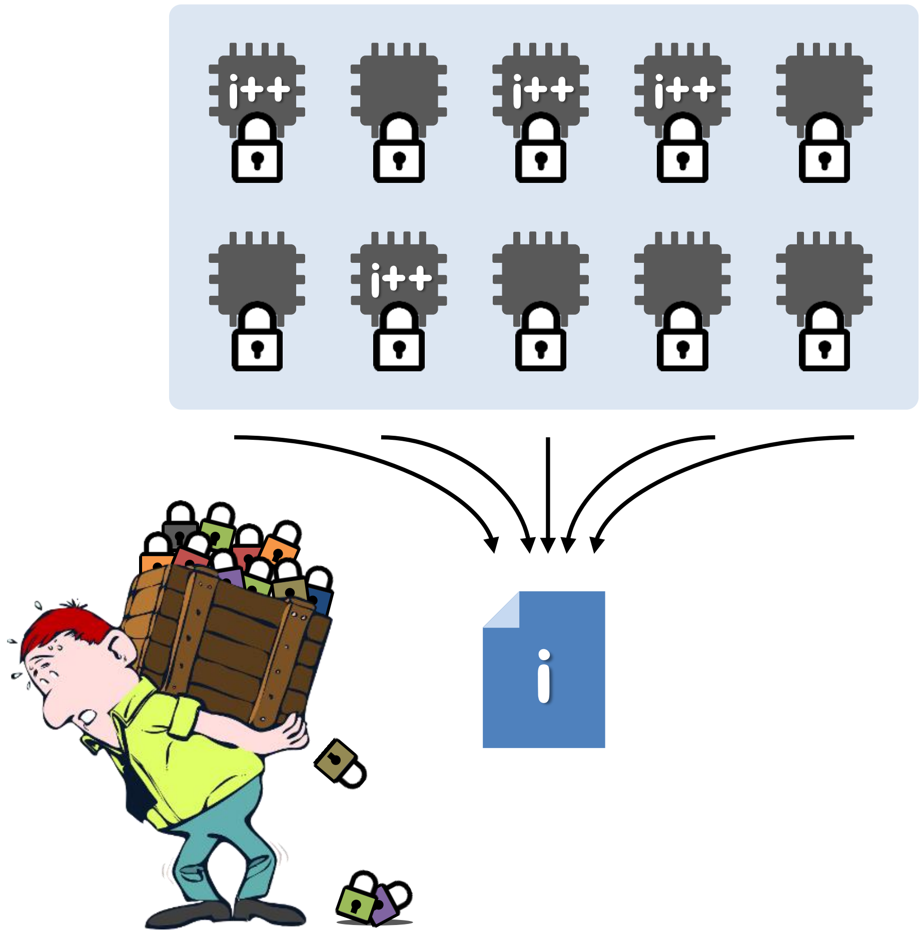




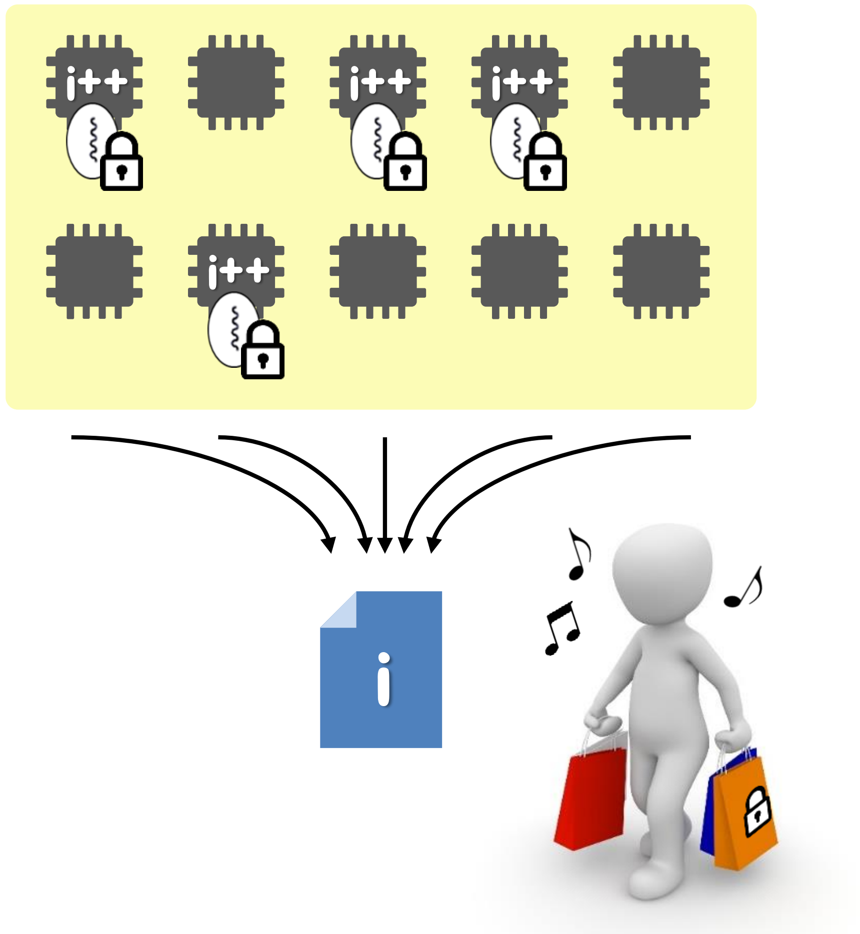


Per-core vs. Per-process view

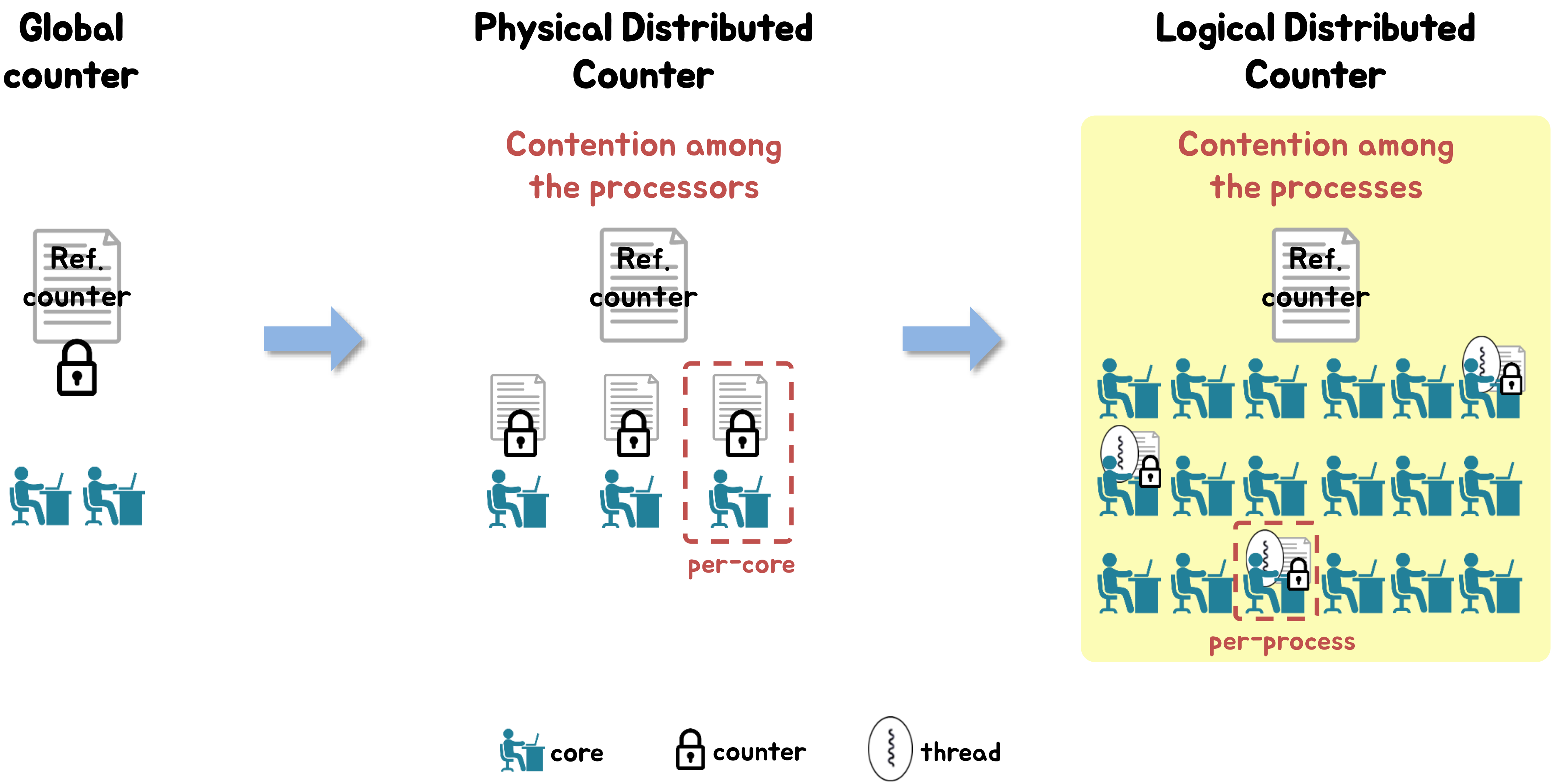
Per-core



Per-process



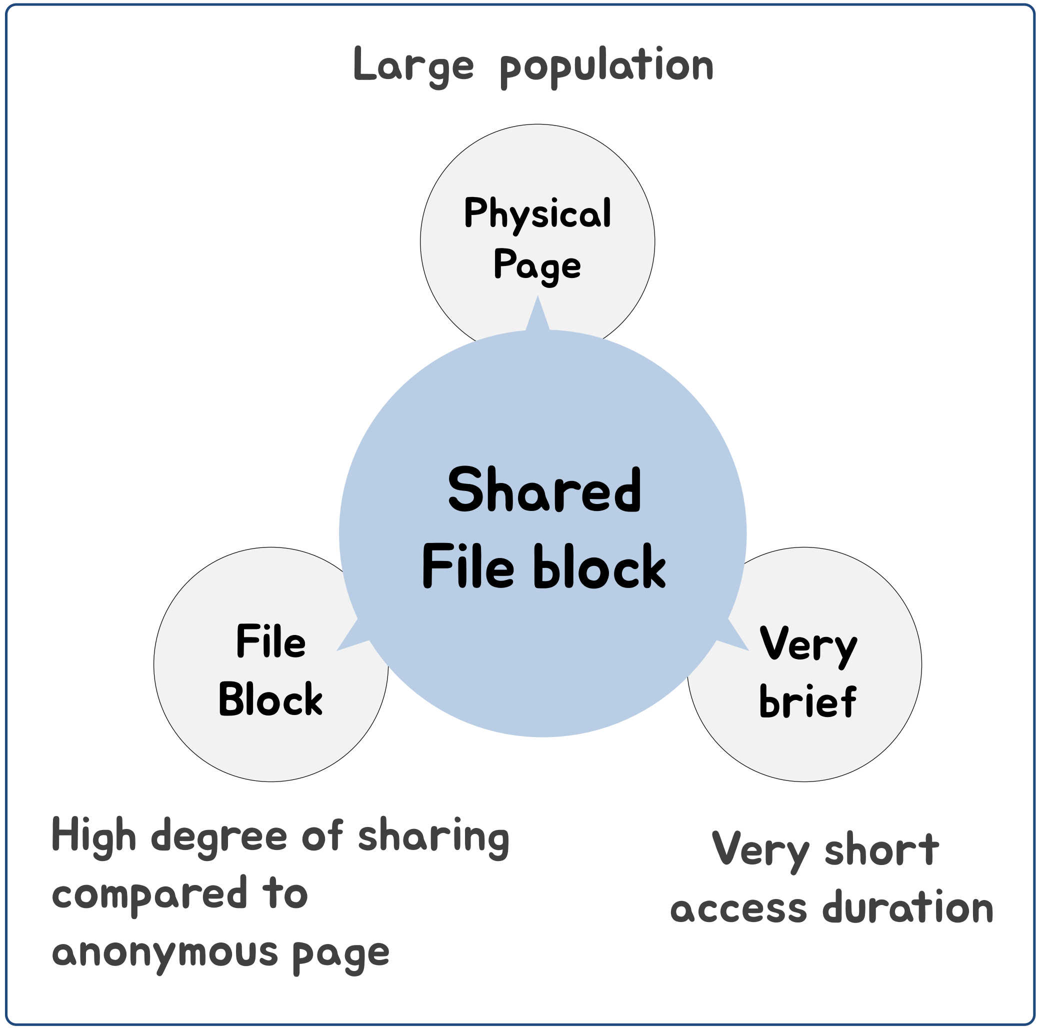
Cause for counter contention



LODIC : Logical Distributed Counter

- LODIC
  - Counter contention is caused by the contention among the processes
  - Distributed counter with local counters are defined in per-process basis

- Used characteristics
  - Popularity : Define the counter with respect to the degree of sharing
  - Access brevity : Not consider the reference split



Objective

Scalability

Memory overhead

Query latency

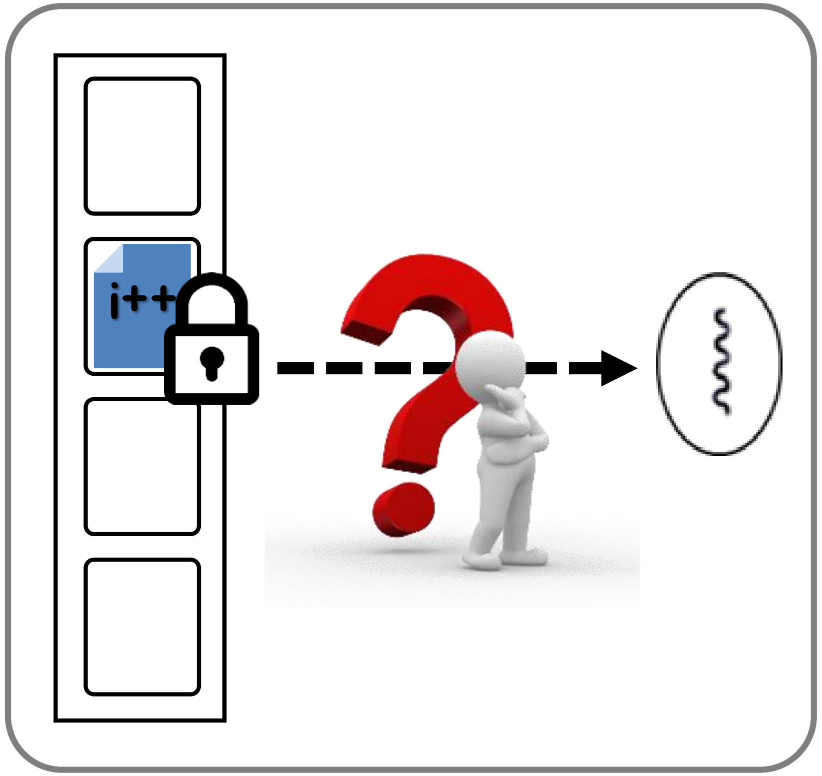
The number of counters are proportional to the degree of sharing

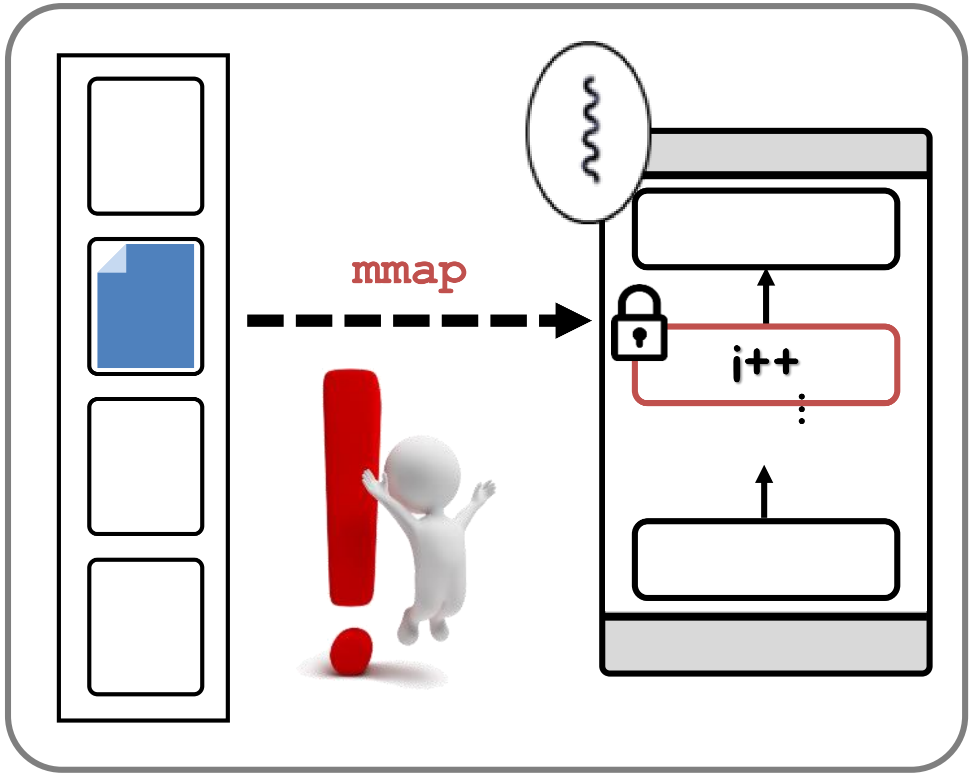
Key techniques

- File mapping
  - Map the file block to the process address space
- Reverse Mapping
  - Do not use existing `rmap` mechanism that is not scalable
  - Reverse Mapping based upon the process address space, file's address space
- Counter Embedding
  - Use the un-used bits in the page table entry

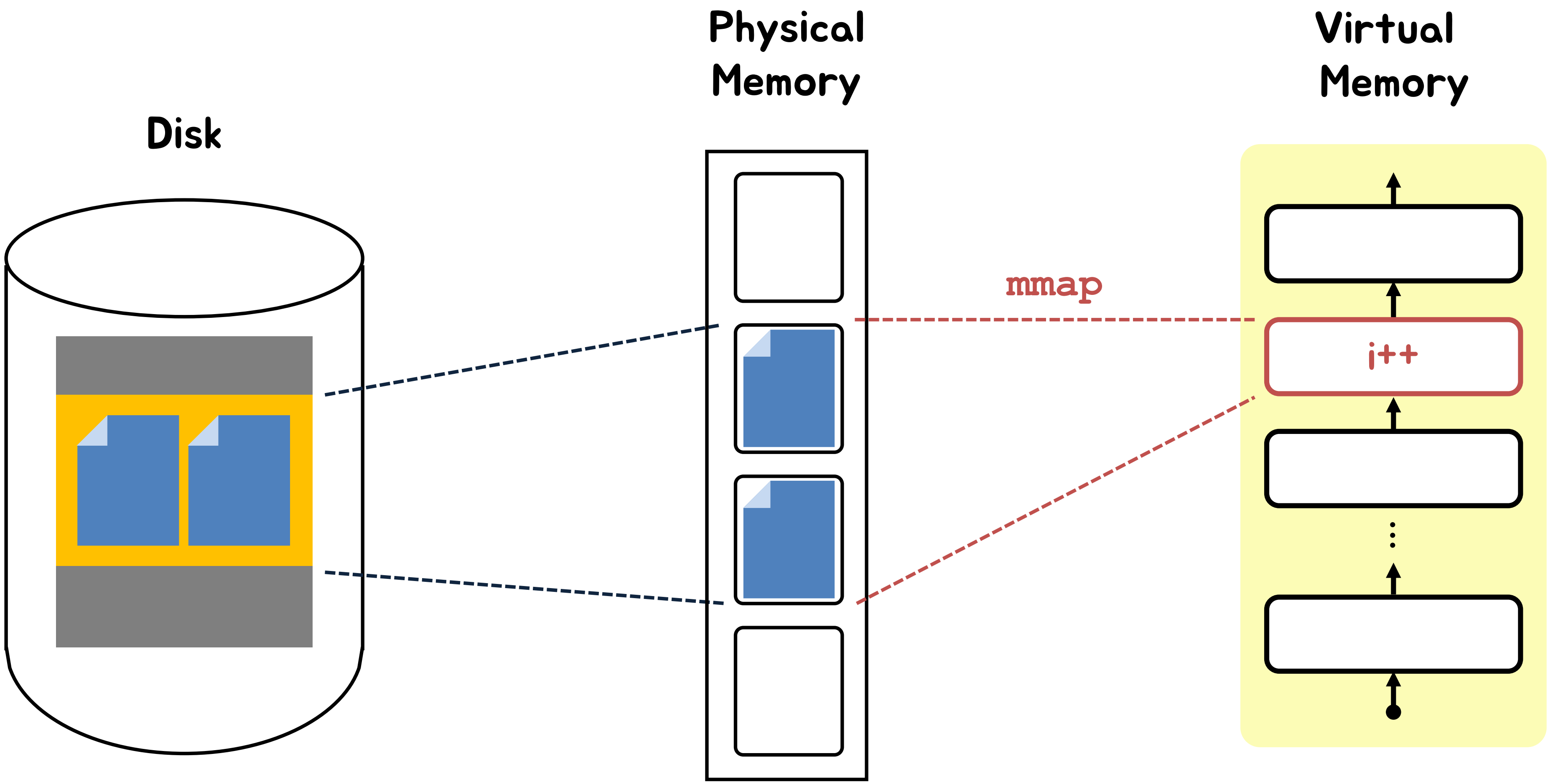
Allocate the counter with per-processes basis

How to allocate a counter of physical page on process address space ?



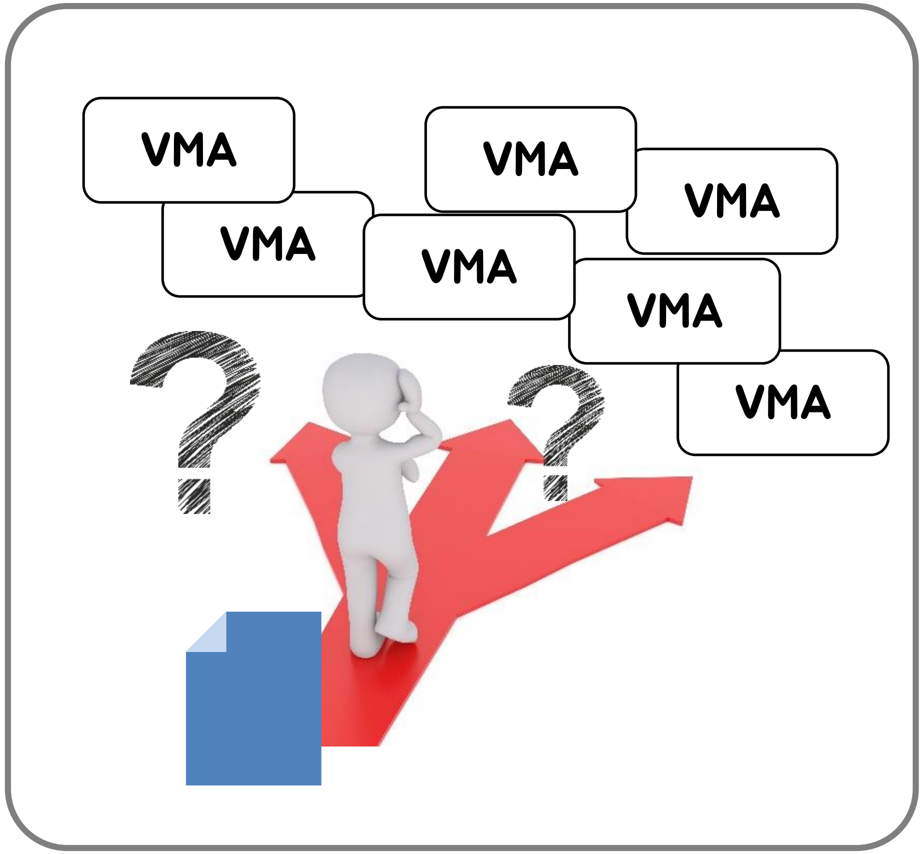


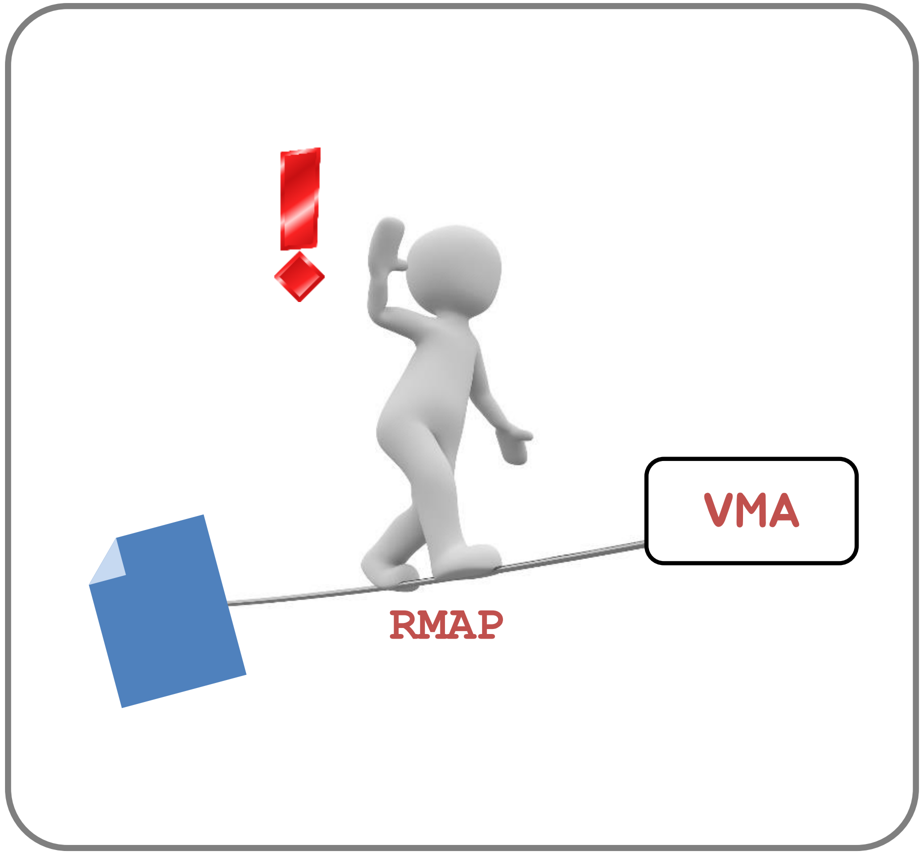
Key technique 1: File Mapping



Find VMA of mapped page

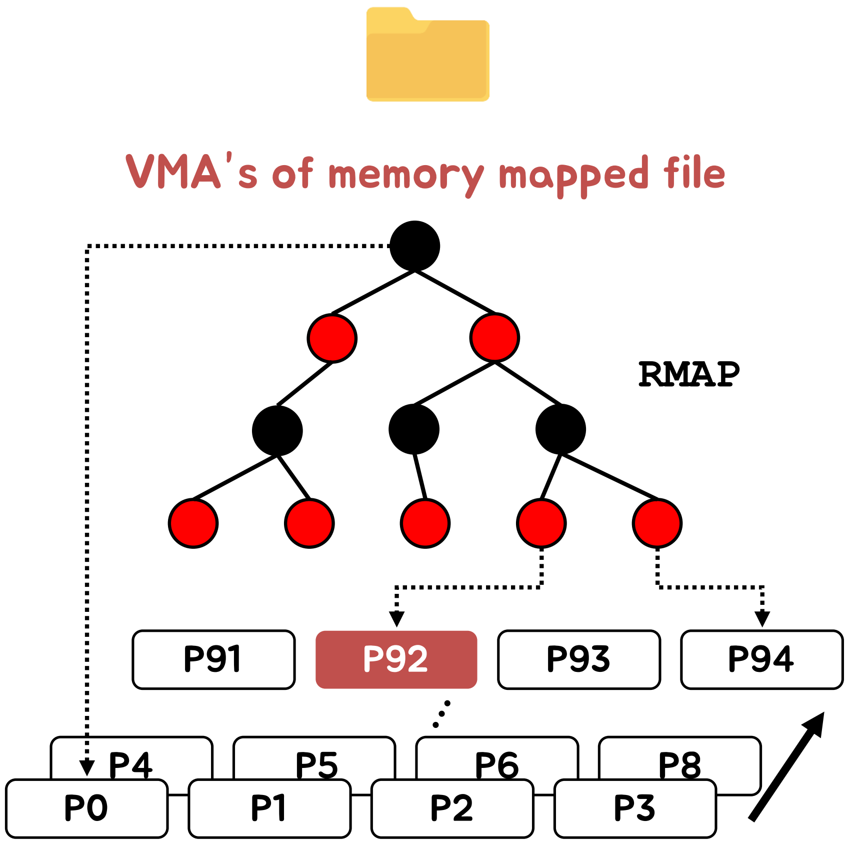
How to quickly find the VMA of page mapped shared file block ?



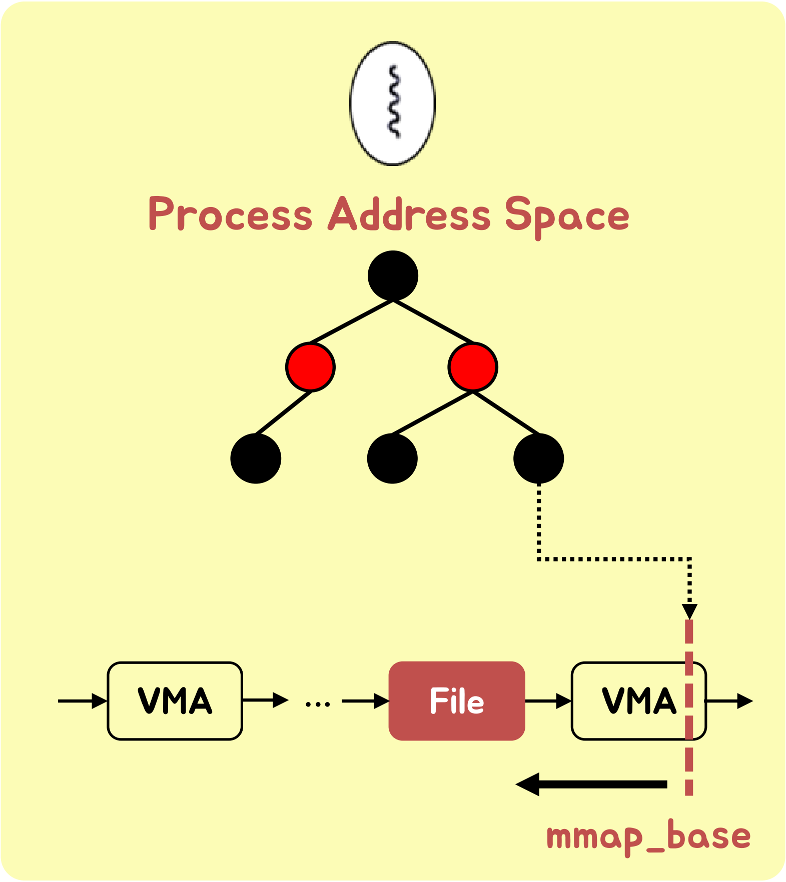


Key Technique 2: Reverse Mapping

File-Space based reverse mapping

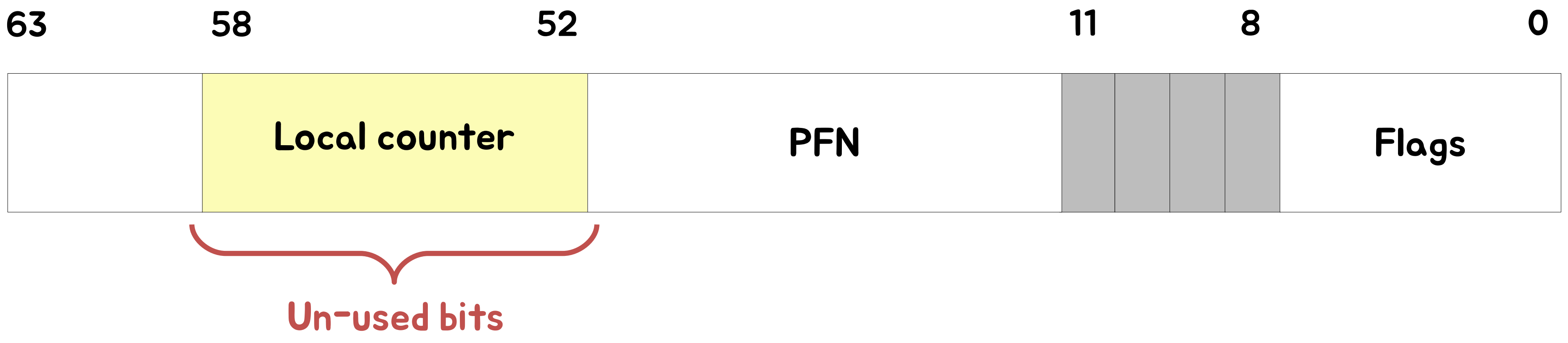


Process-Space based reverse mapping

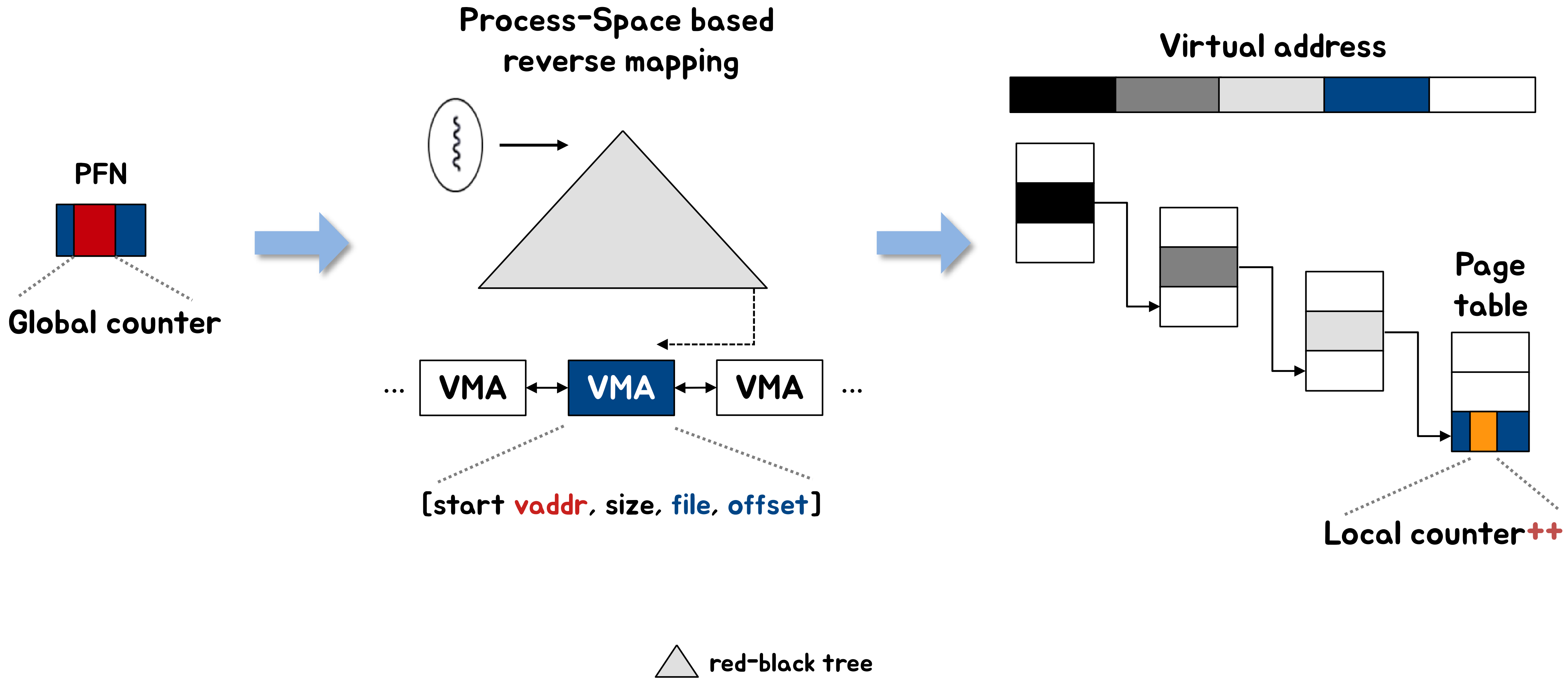


Key technique 3: Counter Embedding

- Embed the local counter at PTE
- For local counter, use un-used bits in PTE.



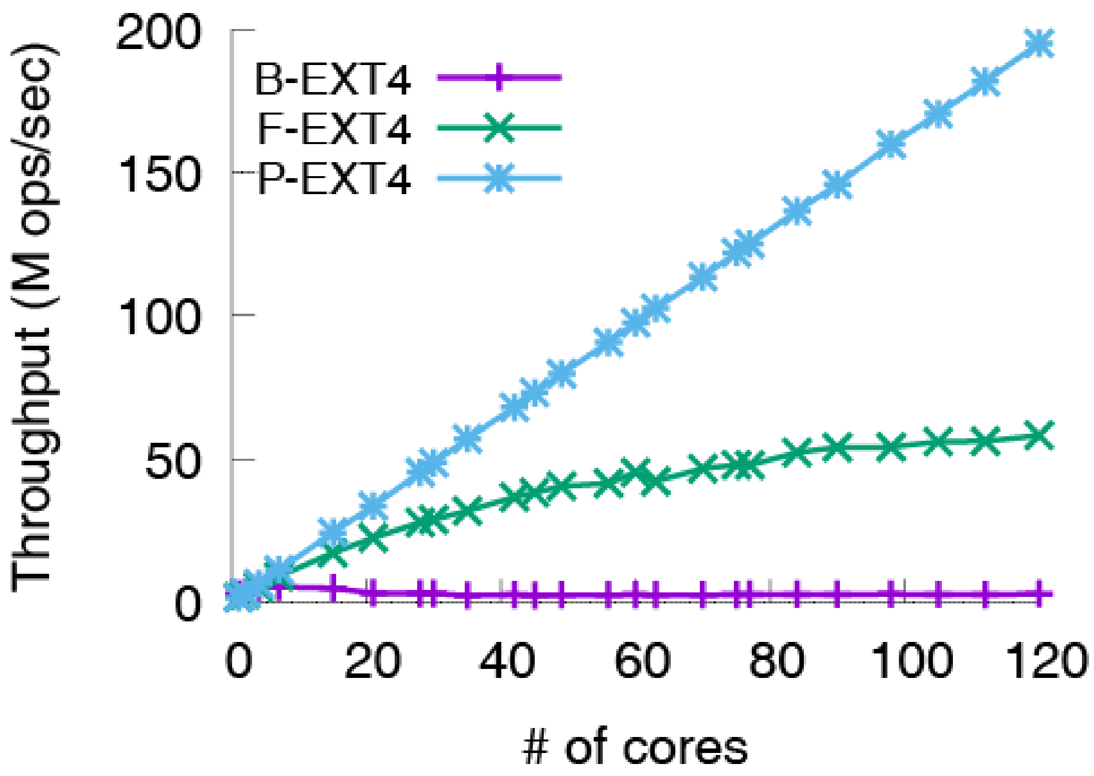
All in one view



Evaluation

Throughput on shared file block read

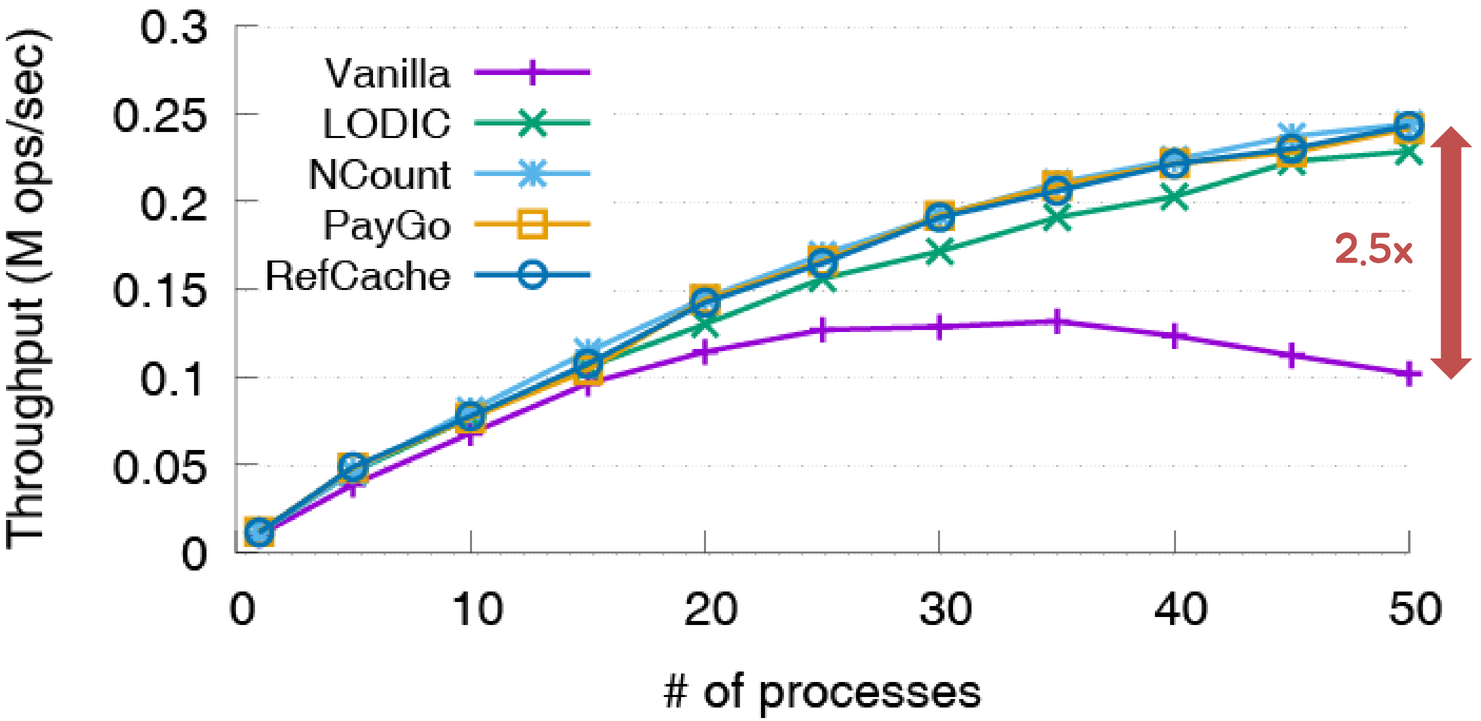
- 120-cores (15 cores/CPU, 8 socket, Intel Xeon E7-8870), 780 GB DRAM, Linux 4. 11. 6
- DRBH Workload on Fxmark



B : Baseline , F : File-based reverse mapping , P : Process-based reverse mapping

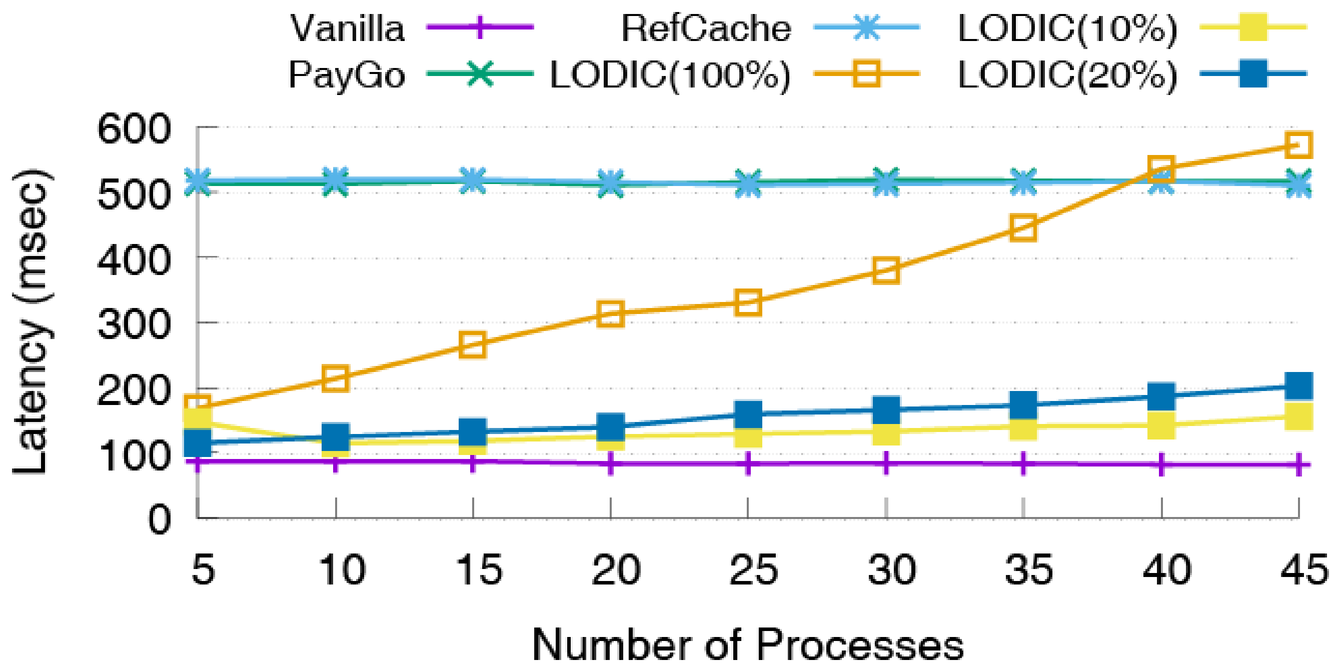
Web server throughput

- 50 client processes, 50 server processes
- NGINX : Reverse proxy server that handles client request
- wrk benchmark : Make the client process to read request for the same file



Counter query latency

- `fadvise()` : System call to reclaim the page
- File size : 1GB
- LODIC(10%) : 10% of file blocks are mapped
- LODIC(20%) : 20% of file blocks are mapped
- LODIC(100%) : 100% of file blocks are mapped



Conclusion

- We take process-centric view in designing the distributed counting scheme

" Counter contention is caused by the contention among the processes,
not by the contention on the processors "

- Number of local counters : With respect to the actual degree of sharing
- Memory pressure : Almost none

- Throughput on the shared block read increases by 65x
- Web server performance increases by 2.5x
- Memory pressure decreases by 13x against per-core distributed reference counters

Q & A

Email : arsd098@kaist.ac.kr