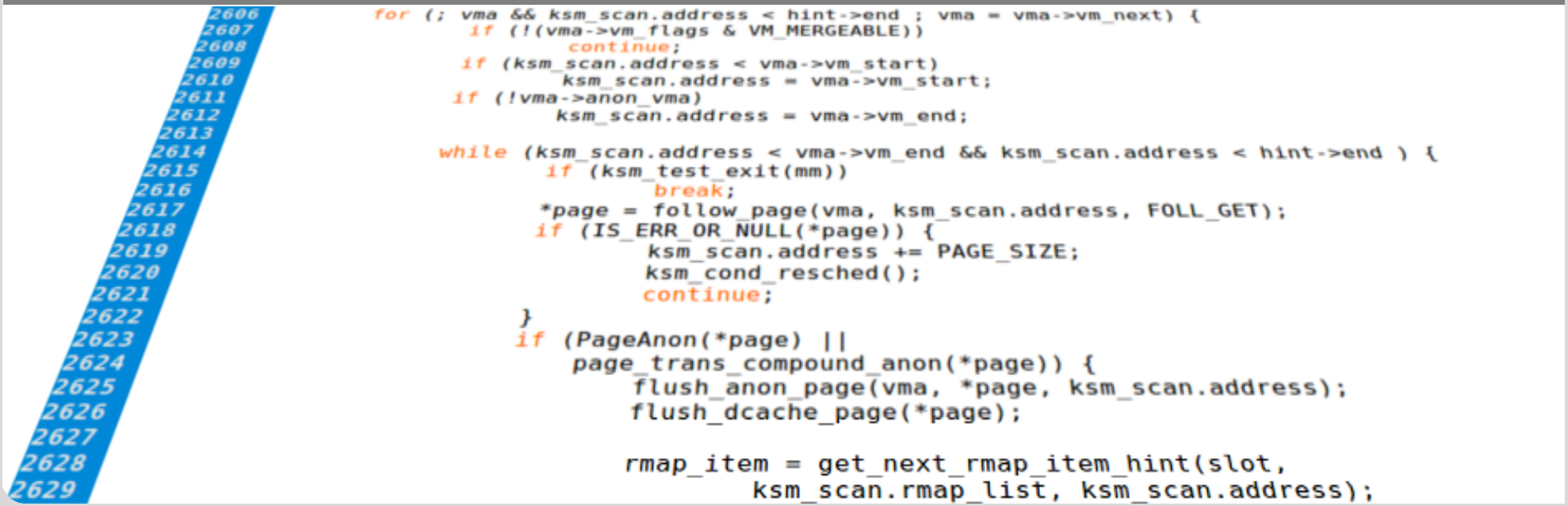
Karlsruhe Institute of Technology

# KSM++: Using I/O-based hints to make memory-deduplication scanners more efficient

Konrad Miller

KIT - System Architecture Group | RESoLVE, March 2012

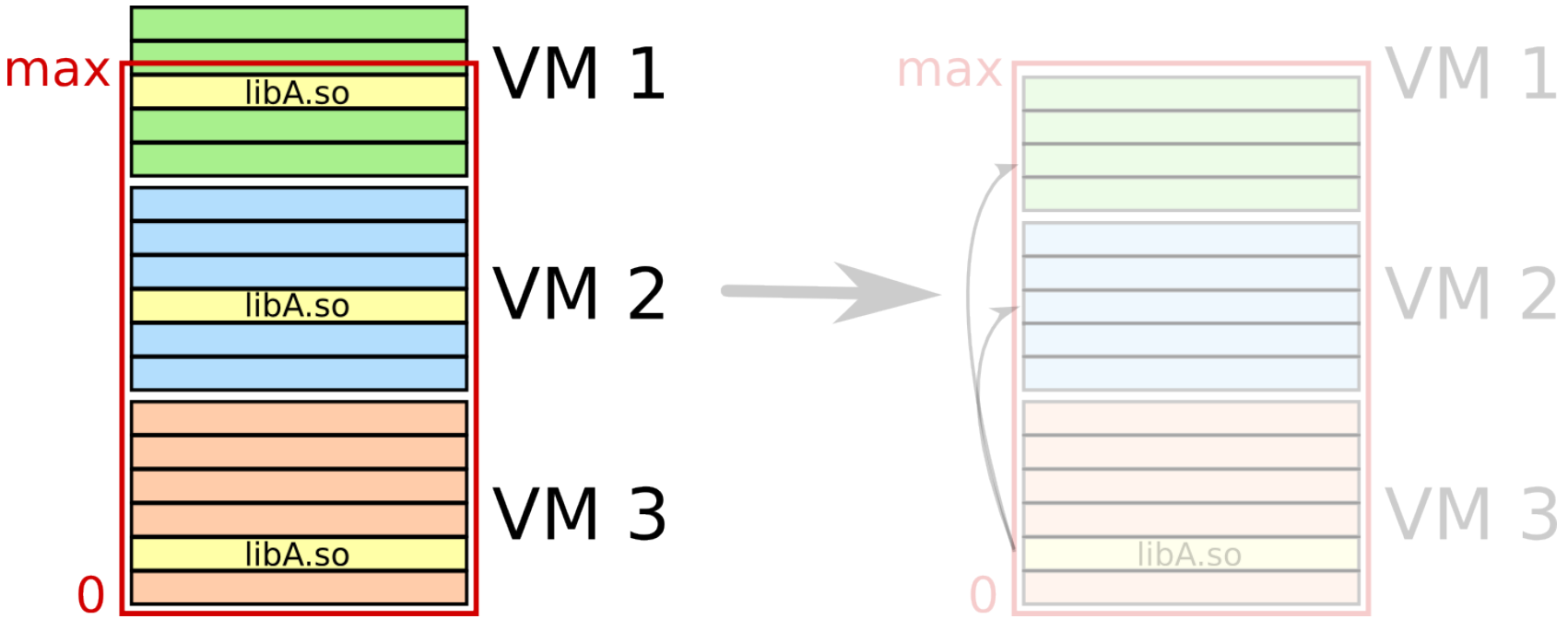
KONRAD MILLER FABIAN FRANZ THORSTEN GRÖNINGER MARC RITTINGHAUS MARIUS HILLENBRAND FRANK BELLOSA



```
2606  for (; vma && ksm_scan.address < hint->end ; vma = vma->vm_next) {
2607          if (!(vma->vm_flags & VM_MERGEABLE))
2608                  continue;
2609          if (ksm_scan.address < vma->vm_start)
2610                  ksm_scan.address = vma->vm_start;
2611          if (!vma->anon_vma)
2612                  ksm_scan.address = vma->vm_end;
2613
2614          while (ksm_scan.address < vma->vm_end && ksm_scan.address < hint->end ) {
2615                  if (ksm_test_exit(mm))
2616                          break;
2617                  *page = follow_page(vma, ksm_scan.address, FOLL_GET);
2618                  if (IS_ERR_OR_NULL(*page)) {
2619                          ksm_scan.address += PAGE_SIZE;
2620                          ksm_cond_resched();
2621                          continue;
2622                  }
2623                  if (PageAnon(*page) ||
2624                      page_trans_compound_anon(*page)) {
2625                          flush_anon_page(vma, *page, ksm_scan.address);
2626                          flush_dcache_page(*page);
2627
2628                          rmap_item = get_next_rmap_item_hint(slot,
2629                                  ksm_scan.rmap_list, ksm_scan.address);
```

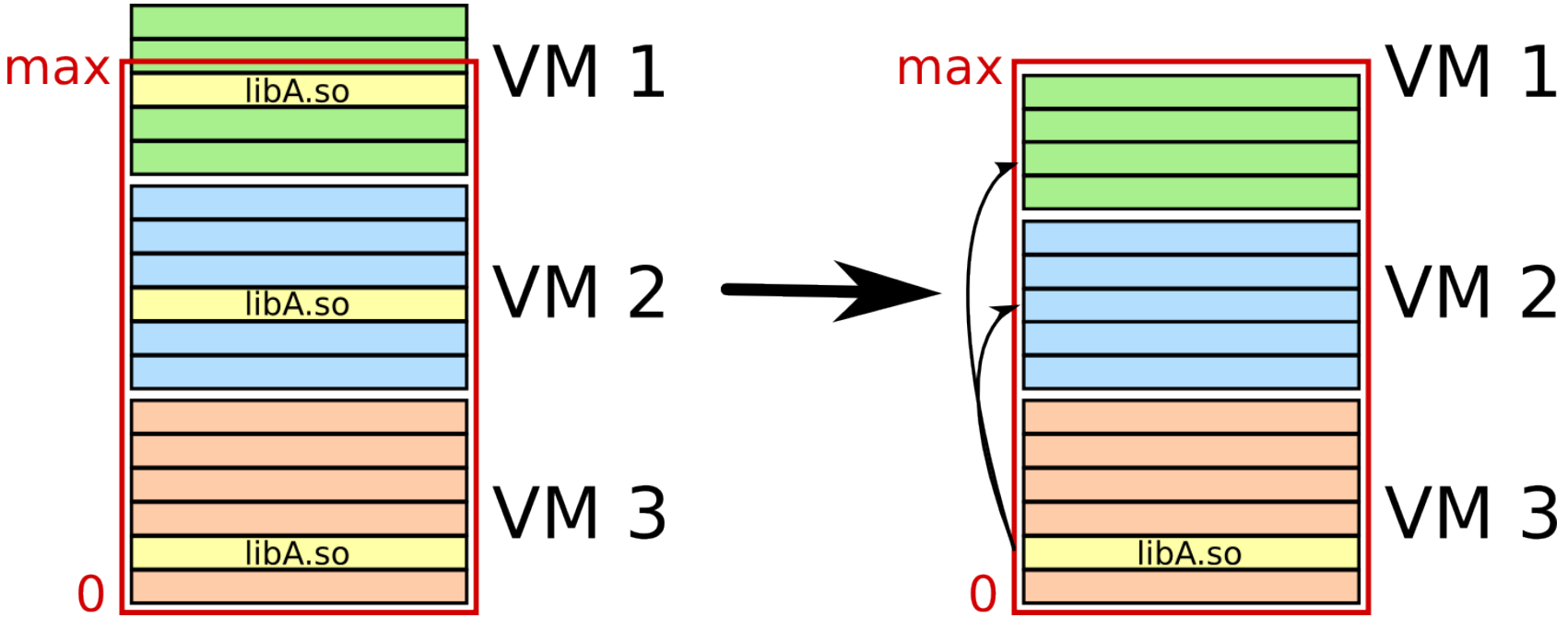
# Deduplication in VM Environments

- Main memory is the primary bottleneck when consolidating VMs
- Different VMs often contain pages with equal content



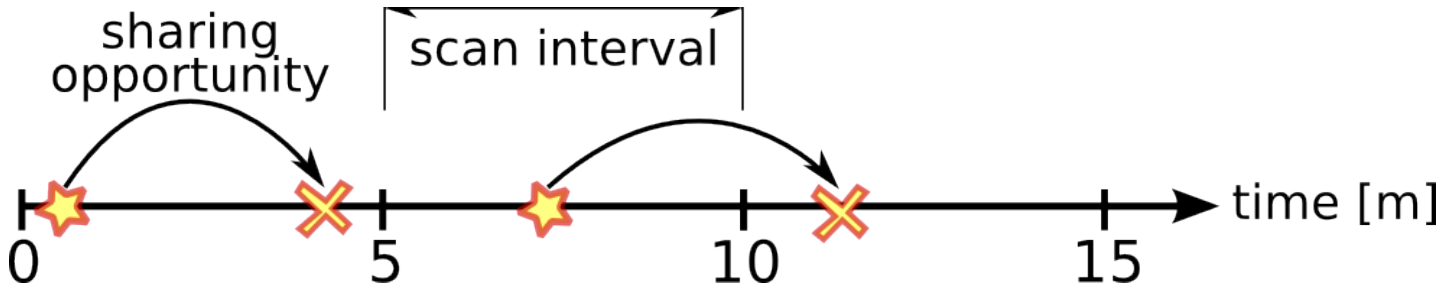
# Traditional Sharing vs. Semantic Gap

- Memory footprint can be reduced: merge equal pages
- Finding well suited merge candidates is not trivial, VMs have different name spaces, VMM has no semantic information about VMs' memory



# Getting Around the Semantic Gap

- Memory scanners directly address page contents
  - Continuously catalog page contents
    - In random order (VMware ESX, OSDI'02)
    - In linear order (Linux Kernel Samepage Merging (KSM) Linux Symposium'09)
  - Merge and COW equal pages
  - Tame scan rate: Only effective for long-lived pages ($>$ 5 min)
  - Aggressive scan rate: High CPU/memory bandwidth overhead




- Initial benchmarks: more than 70% of mergable pages modified...
  - ...late enough to amortize the merge cost
  - ...too early to be caught by scanner

# Closing the Semantic Gap

- Many deduplication candidates stem from Virtual Disk Image (VDI)
  - Libraries, programs, data
  - Guests' I/O target pages are prime deduplication candidates
- Assumption: There is a correlation between I/O and memory deduplication candidates
- Paravirtualization/Introspection closes the semantic gap
  - Modify guests' VDI driver (Satori, USENIX'09)
  - Hook guests syscalls (Disco, SOSP'97)
  - Efficiently catch duplicates that stem from VDI
  - Need to process *all* I/O requests $\rightarrow$ I/O-intensive workloads?

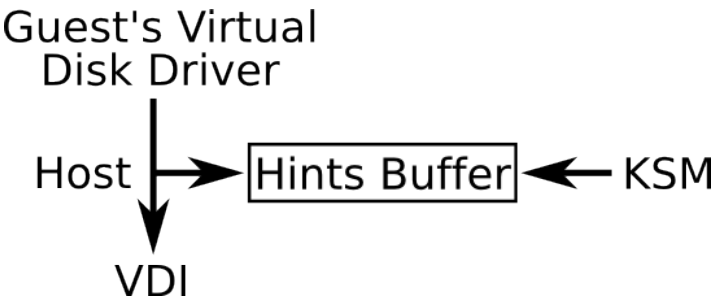
# KSM++: Hints for Memory Scanners

- Observation
  - Host/Hypervisor does I/O on behalf of guest VMs
  - I/O-operations target guests' buffer caches and mmap areas


- KSM++
  - Extension of KSM
  - Generate hints for scanner in host, on I/O calls from guest
  - Visit I/O-pages earlier in memory scanner
  - Quick detection of I/O-based sharing opportunities
  - Only modified/added ∼400 LOC (Linux kernel)

# Hint Generation, Storage, and Processing

- Intercept VFS calls
  - Host-VFS target memory area is used as hint
  - Works for all processes, not limited to VMs
  - Purely in the host, no paravirtualization

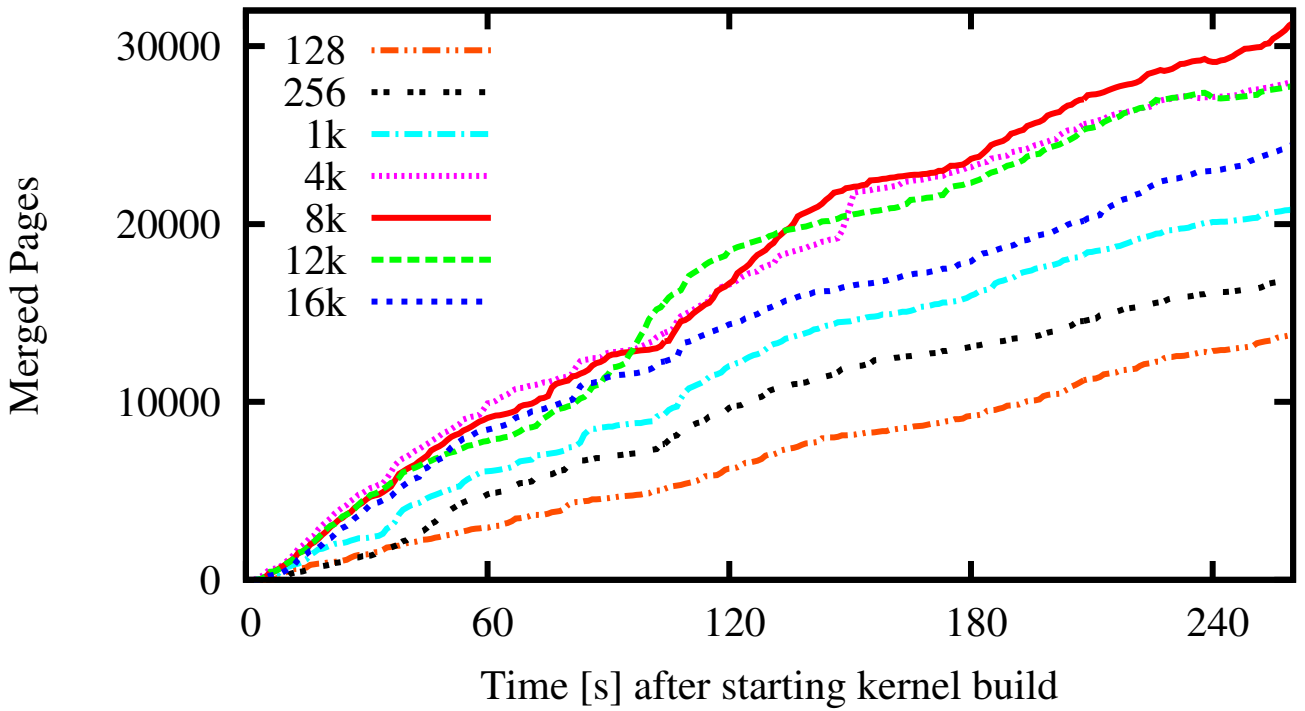



- I/O is bursty
  - Buffer a fixed number of unprocessed hints
  - Lossy buffer overwrites old hints

- Process hints interleaved to regular KSM scan process
  - Don’t starve non-I/O scan: catch duplicates from all sources
  - Obey to scan-rate limits (can limit CPU/IO resource-consumption)

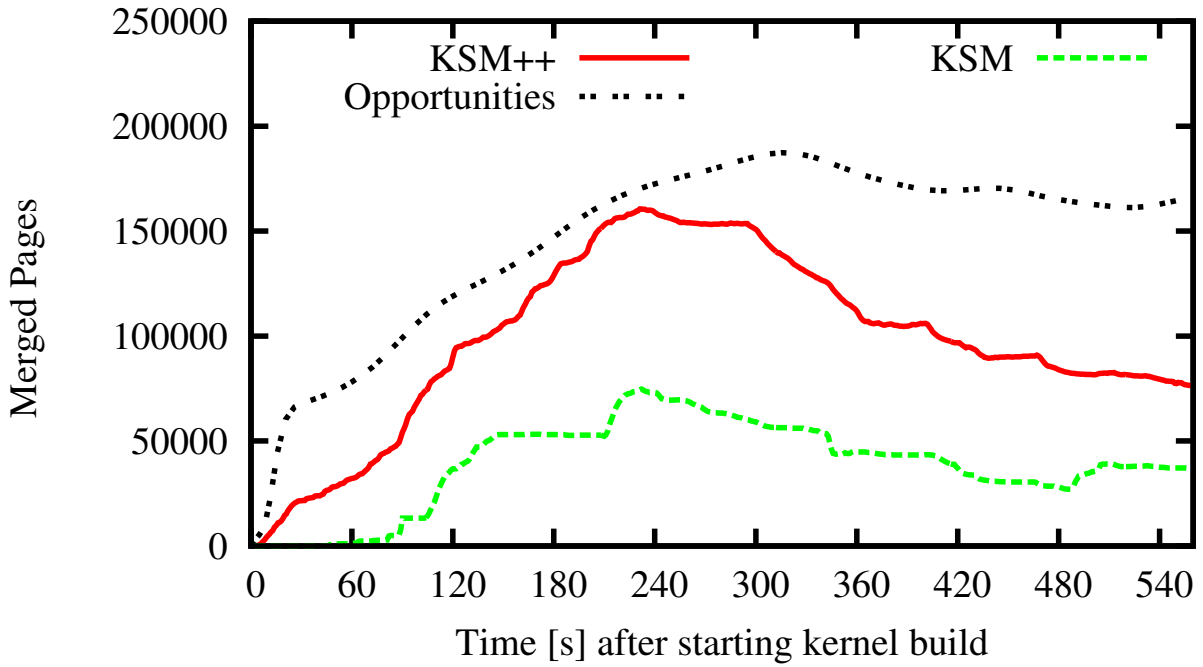
# Evaluation Results - Hint Buffer Size

- In our experiments I/O was so bursty, that we couldn't store/process *all* generated hints within rate limits
- Yet, small hint buffers were "large enough"
- Too large buffers are actually hurting performance (16k line)



# Evaluation Results - Merge Performance

- KSM++ needs to visit less pages to find a sharing opportunity
- KSM datastructures are suboptimal – fixed for fair comparison




- Sharing opportunities peak at about 20% of total memory assigned to VM in this benchmark (measured with memory snapshots)

# Conclusion

- Motivation:
  - Main memory is scarce in virtualized environments $\rightarrow$ deduplication
  - Finding sharing opportunities efficiently is not trivial
  - Memory scanners can find long lived sharing opportunities
- KSM++ keeps properties of memory scanners:
  - Not limited to I/O pages; catch duplicates from all sources
  - Configurable, maximum overhead
  - No Paravirtualization
- KSM++ hints help detecting 2-10x more sharing opportunities than pure random or linear scanning in our benchmarks