Jin-Soo Kim
(*jinsoo.kim@snu.ac.kr*)

Systems Software &
Architecture Lab.

Seoul National University

Spring 2026

Disco

(E. Bugnion, S. Devine, and M. Rosenblum, SOSP 1997)

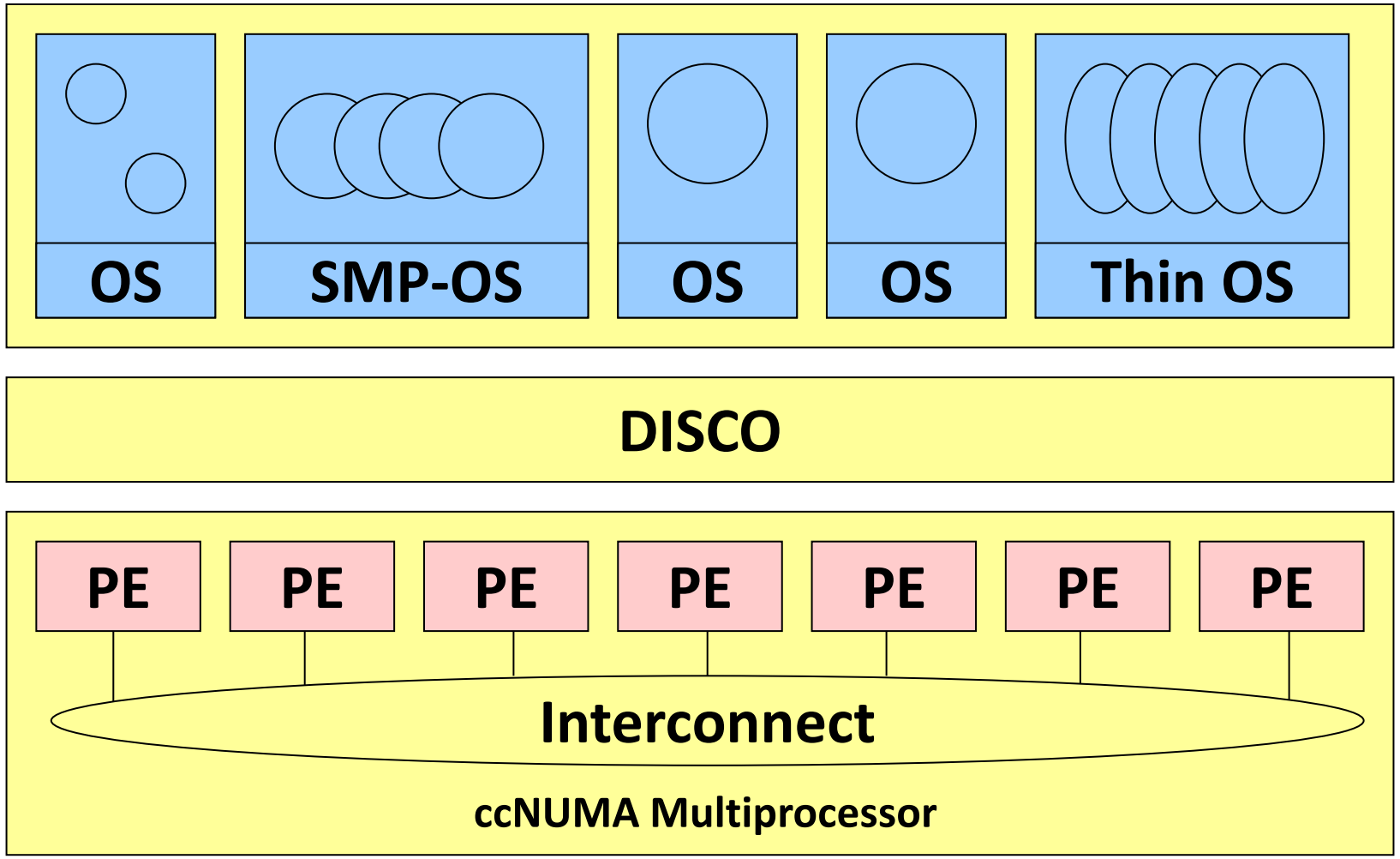
Background

- ccNUMA: Cache-coherent non-uniform memory architecture
  - Multiprocessor with high-performance interconnect
- Non-uniform memory
  - Global address space
  - But memory distributed amongst processing elements
- Cache-coherence
  - Issue: How to ensure that memory in processor caches is consistent?
  - Solutions: Bus snooping, directory
- Targeted system: FLASH, Stanford's own ccNUMA
- "Commodity OS": SGI IRIX

The Challenge

- Commodity OSes not well-suited for ccNUMA
  - Do not scale: ***Why?***
  - Do not isolate/contain faults: More processors → more failures
- Customized operating systems
  - Take time to build, lag hardware
  - Cost a lot of money
- Disco: 13,000 lines of code
  - Code segment size: 72KB (replicated to all NUMA memories)

The Solution: DISCO



How to Virtualize?

- Virtualize physical resources
  - CPU: instructions → Trap all privileged instructions
  - Memory: address spaces → Map "physical pages" managed by the guest OS to machine pages, handle translation, dynamic page migration & replication, etc.
  - Devices → Any I/O communication needs to be trapped and passed through/handled appropriately, virtual disks & networks
- Dispatch events
  - e.g., forward page fault trap to guest OS
- Manage resources
  - e.g., divide real memory between the physical memory of each guest OS

Virtualizing CPUs

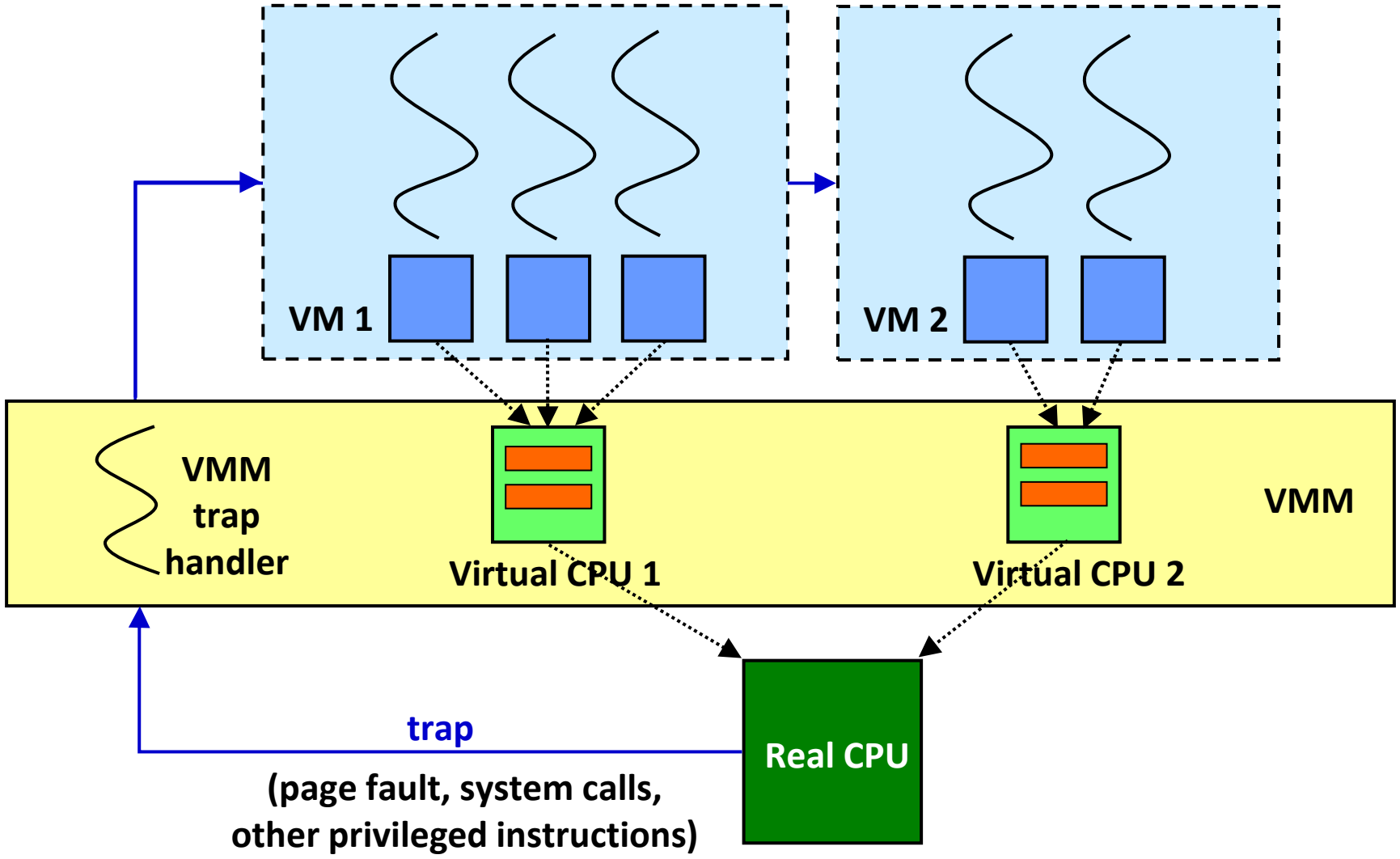
- MIPS R10000 in FLASH: the (unfortunate) choice for Disco
- MIPS R10000 has three operating modes
  - Kernel mode: Disco
  - Supervisor mode: Guest OS (no access to privileged instructions or physical memory)
  - User mode: applications
- MIPS R10000 does not support the complete virtualization
  - A processor running in supervisor mode cannot access the KSEG0 segment efficiently, that bypasses the TLB
  - IRIS 5.3 places the kernel code and data in the KSEG0 segment
  - Requires modifications to the IRIX kernel to relocate the kernel to the mapped supervisor segment

Virtualizing CPUs: Virtual CPU

- For each virtual CPU, Disco keeps a data structure for
  - The saved registers
  - _____
  - _____
  - Other state of a virtual CPU

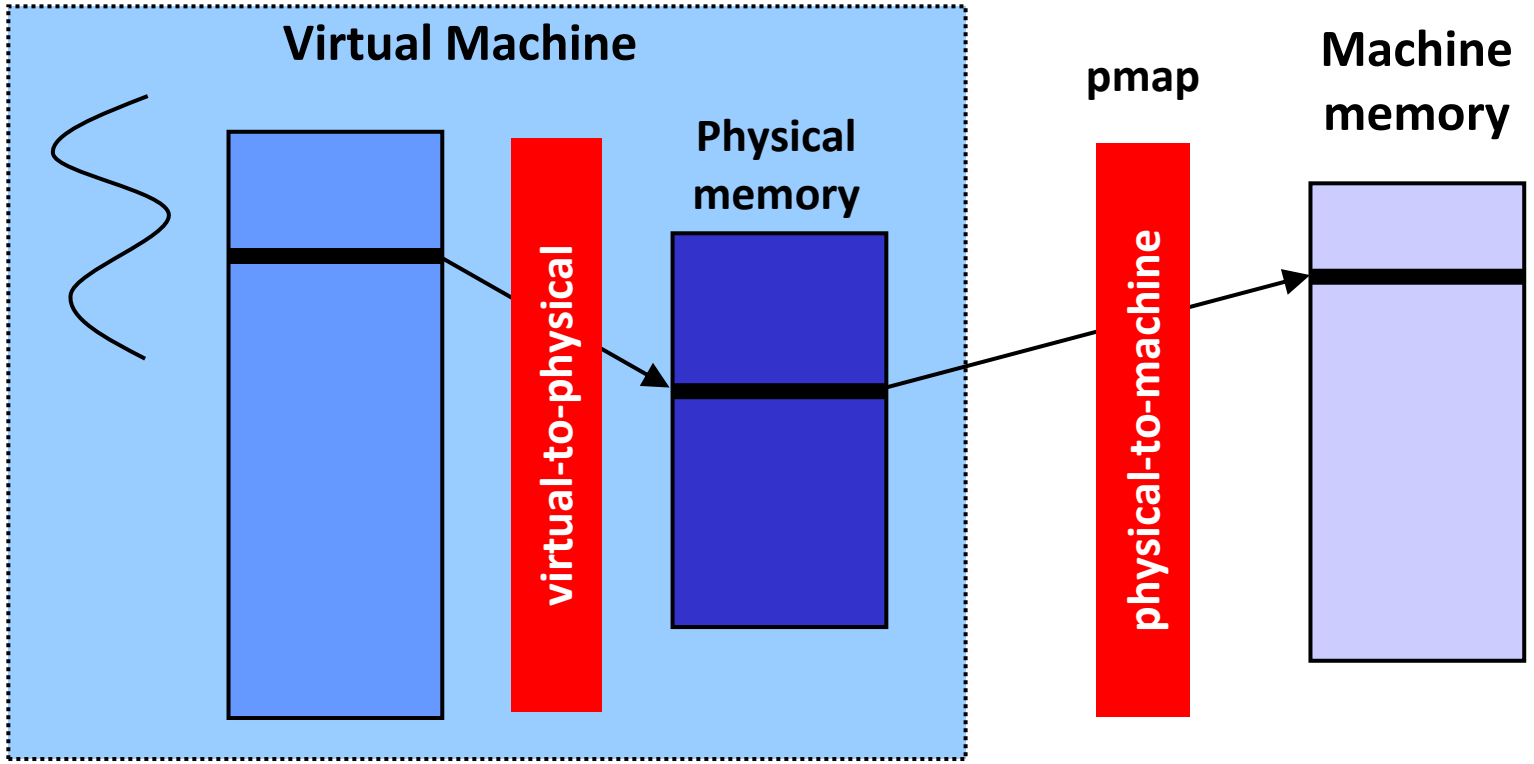
- Virtualizing CPUs
  - To schedule a virtual CPU, Disco sets the real machines' registers to those of the virtual CPU and jumps to the current PC of the virtual CPU
  - Disco emulates the operations that cannot be issued in the supervisor mode
  - Disco simply time-shares the virtual processors

Virtualizing CPUs: Example



Virtualizing Memory: Machine Address

- Address used by the (physical) memory system of the FLASH machine
- Adds a level of address translation: physical-to-machine



Virtualizing Memory

- When a guest OS attempts to insert a virtual-to-physical mapping into the TLB:
  - Disco translates the physical address into the corresponding machine address, and inserts this corrected TLB entry
- Pmap data structure accelerates the computation of the corrected TLB entry
  - ***How?***

Virtualizing Memory: TLB Handling

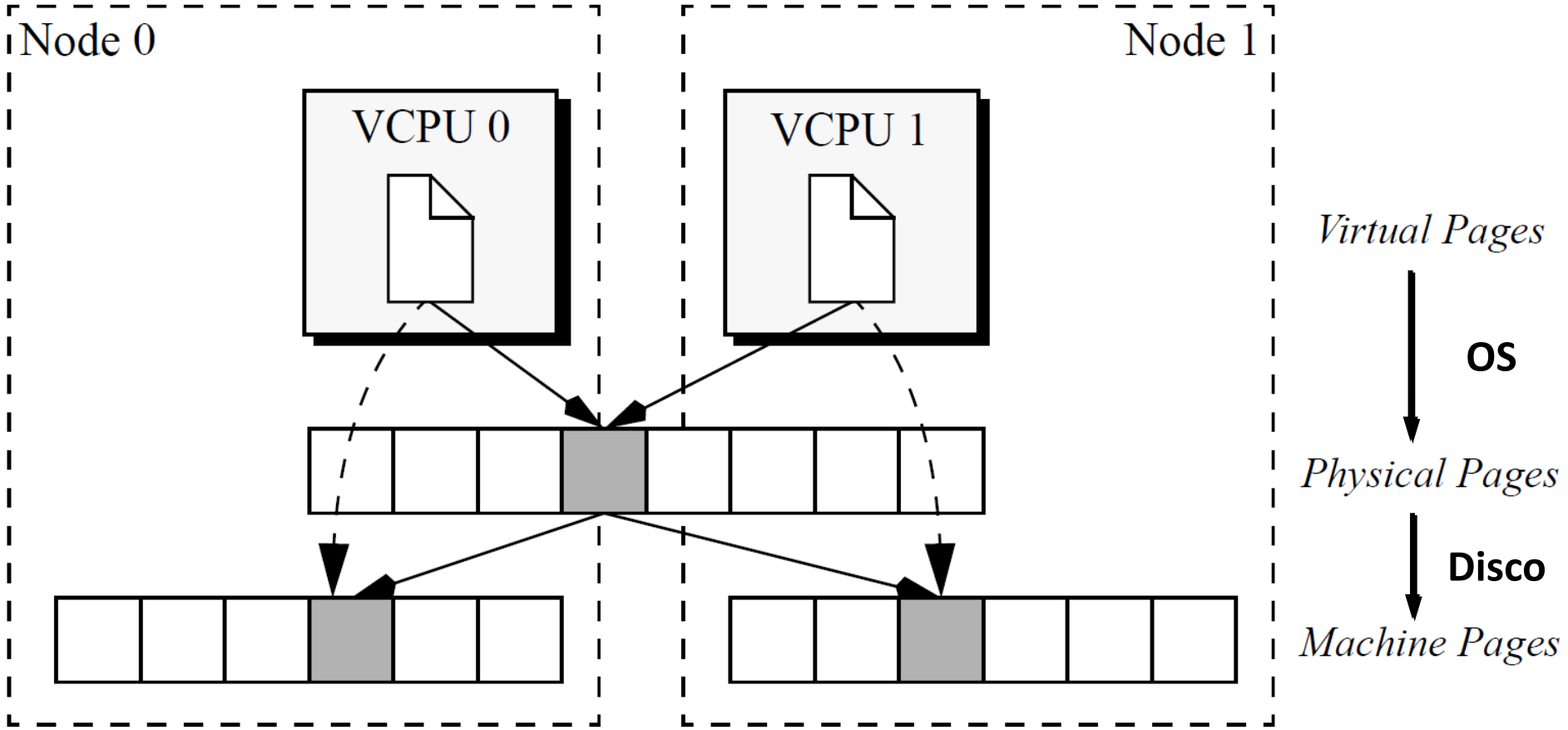
- TLB in Disco
  - MIPS TLB is software-managed and supports ASID (Address Space Identifier)
  - TLB is flushed on virtual CPU switches
  - TLB miss handling is expensive
    - Emulation of the trap architecture
    - Emulation of privileged instructions in the OS's TLB miss handler
    - Remapping of physical addresses
- L2TLB (second-level software TLB)
  - L2TLB caches recent virtual-to-machine translations
  - On a TLB miss, Disco consults L2TLB first
  - If there is no match, Disco forwards the TLB miss exception to the OS

Virtualizing Memory: NUMA Handling

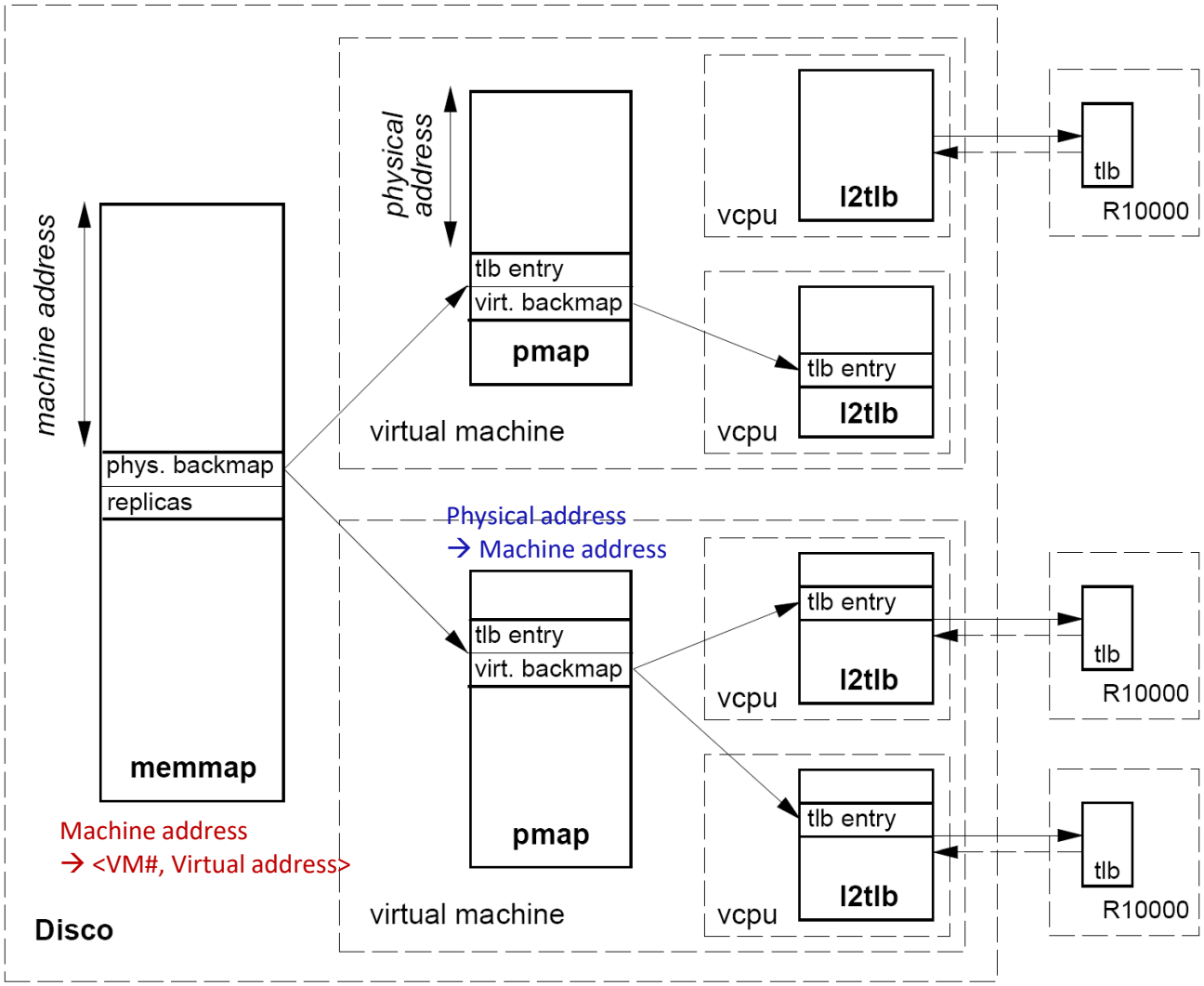
- Dynamic page migration
  - Pages that are heavily accessed by only one node are migrated to that node
- Page replication
  - Pages that are primarily read-shared are replicated to the nodes most heavily accessing them
- FLASH detects a hot page by counting cache misses to each page from every physical processor
- Memmap data structure
  - A list of the virtual machines using the machine page and the virtual addresses used to access them
  - Used for TLB shootdown during page migration and replication

Virtualizing Memory: Page Replication

- Transparent page replication



Virtualizing Memory: Data Structures



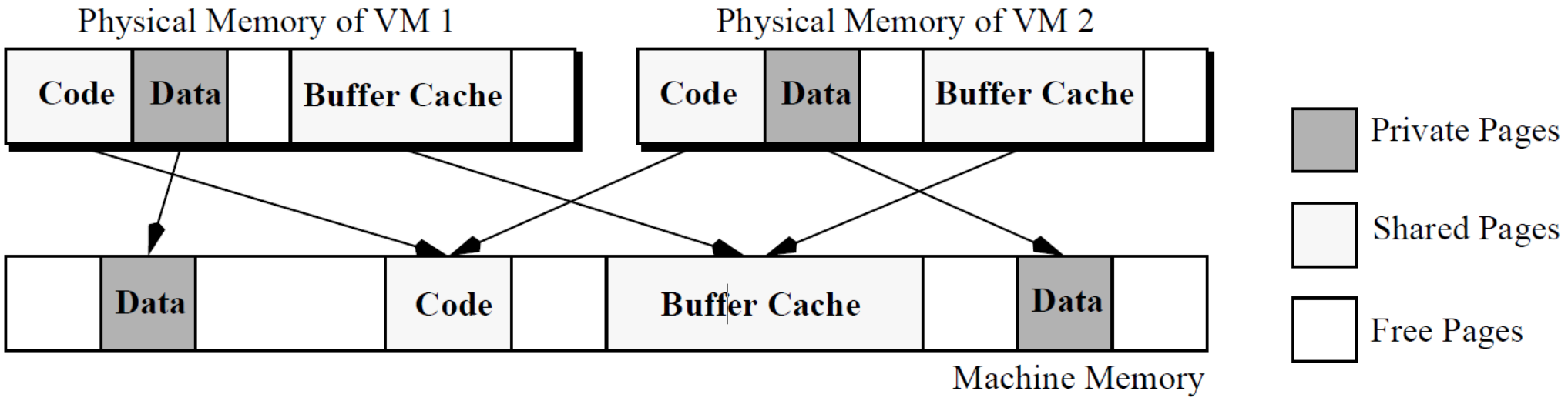
Virtualizing I/O Devices

- Intercept the programmed I/O from the guest OS and emulate the functionality of the hardware device
  - Complex, specific to each device, and require many traps

- Disco: add special device drivers into the guest OS
  - Use a(an) _____ to pass all command arguments in a single trap
  - DMA target addresses (physical addresses) should be translated into machine addresses
  - Disco supports UART, SCSI disks, and ethernet drivers

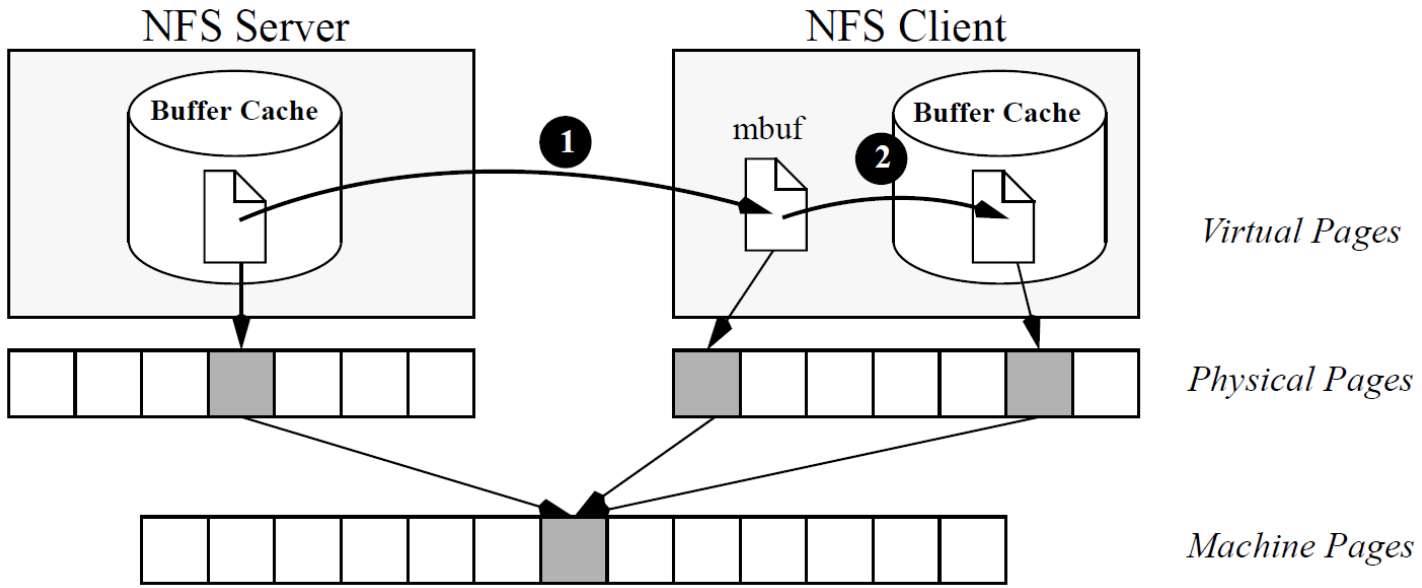
Virtualizing I/O Devices: Disks

- Share disk blocks by mapping the page into the VM's physical memory
- For code and other read-only data (e.g., root disks)
- Sharing read-write data
  - Mount separate disk partition
  - Use NFS



Virtualizing I/O Devices: NICs

- Use copy-on-write mappings to reduce copying and to allow for memory sharing
- Send buffer is remapped to receive buffer
- Receive buffer is remapped again to the buffer cache



Discussion

- Running commodity OS on VM
- Disco still requires kernel modification
  - Some for inherent CPU restrictions
  - Some for optimizations

- Disco and after …
  - Cellular Disco (SOSP '99): For SMPs
  - VMware founded in 1998: For Windows/x86