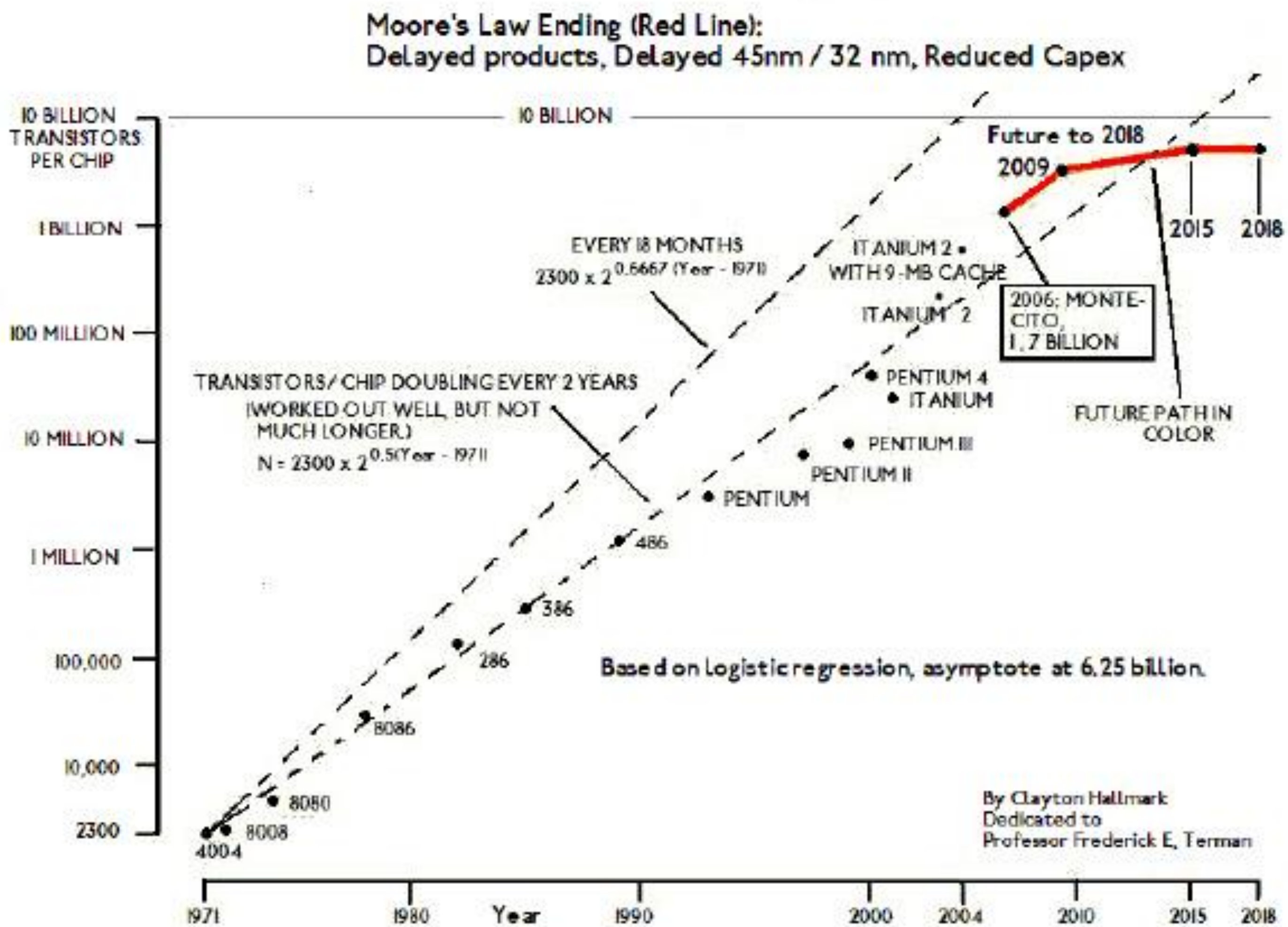
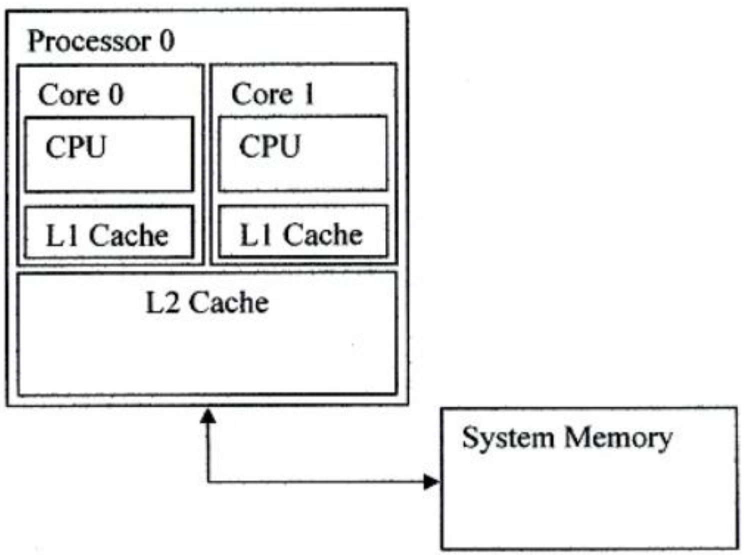
Multiprocessors

Deniz Altinbuken
09/29/09

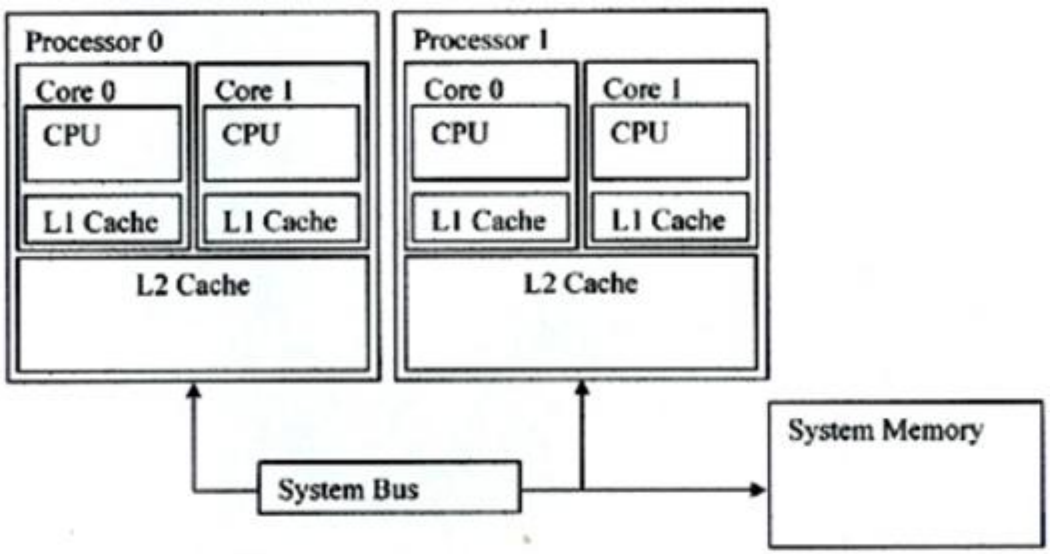
End of Moore's Law?



Multi-Core vs. Multi-Processor

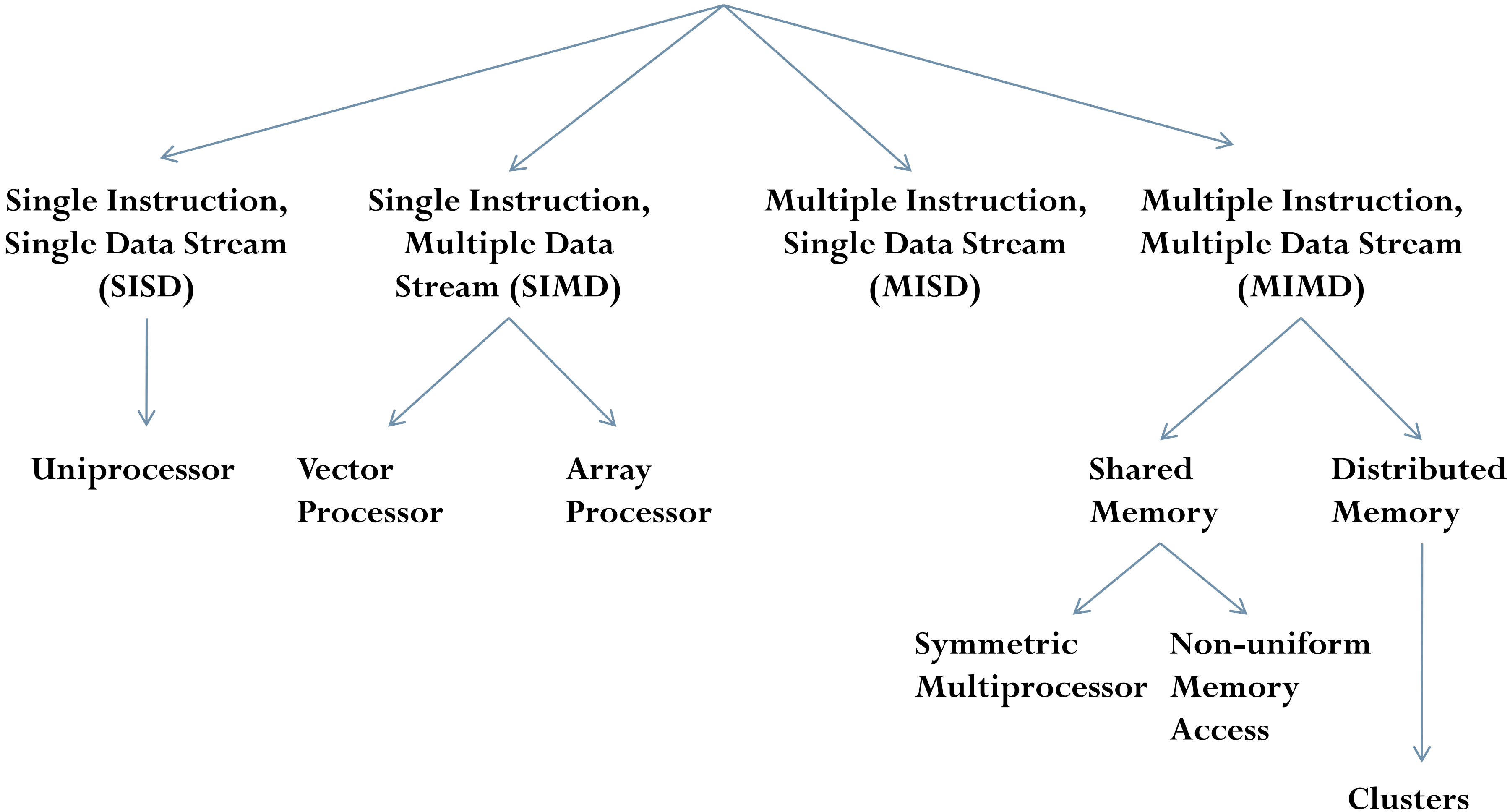


Multi-Core Processor with Shared L2 Cache

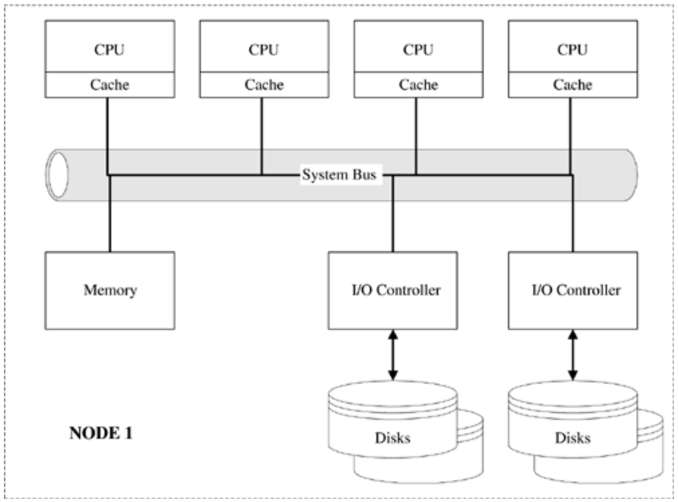


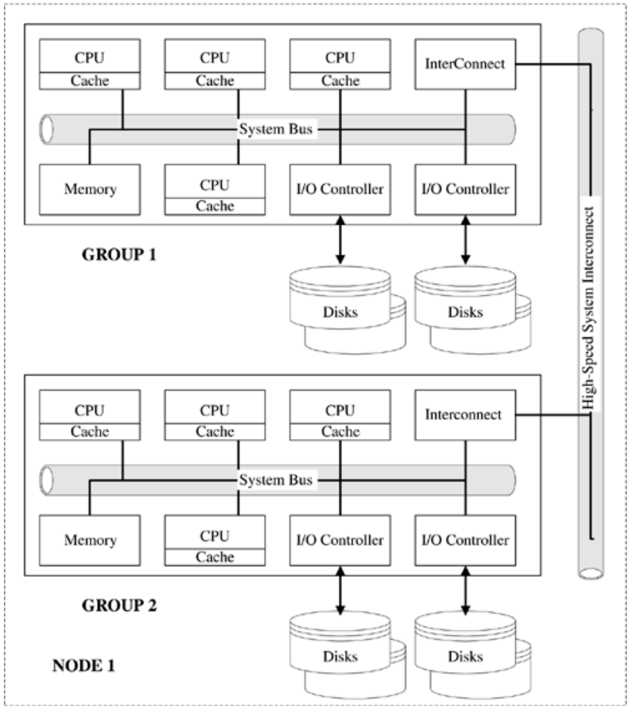
Multi-Processor System with Cores that share L2 Cache

Processor Organizations



Shared Memory Access





Uniform memory access

- Access time to all regions of memory the same
- Access time by all processors the same

Non-uniform memory access

- Different processors access different regions of memory at different speeds

Why Multiprocessors?

- Goal: Taking advantage of the resources in parallel

What is crucial?

- Scalability
  - Ability to support large number of processors
- Flexibility
  - Supporting different architectures
- Reliability and Fault Tolerance
  - Providing Cache Coherence
- Performance
  - Minimizing Contention, Memory Latencies, Sharing Costs

Approaches

- DISCO (1997)
  - Using a software layer between the hardware and multiple virtual machines that run independent operating systems.
- Tornado (1999)
  - Using an object-oriented structure, where every virtual and physical resource in the system is represented by an independent object

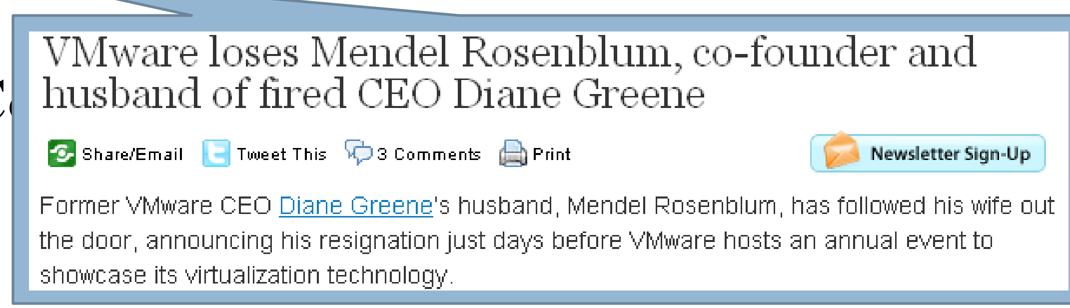
DISCO: Running Commodity Operating Systems on Scalable Multiprocessors

Edouard Bugnion, Scott Devine

- Key members of the SimOS and Disco VM research teams
- Co-founders of VMware
- Ph.D. candidate in Computer Science at Stanford University

Mendel Rosenblum

- Key member of the SimOS and Disco VM research teams
- Co-founder of VMware
- Associate Professor of C

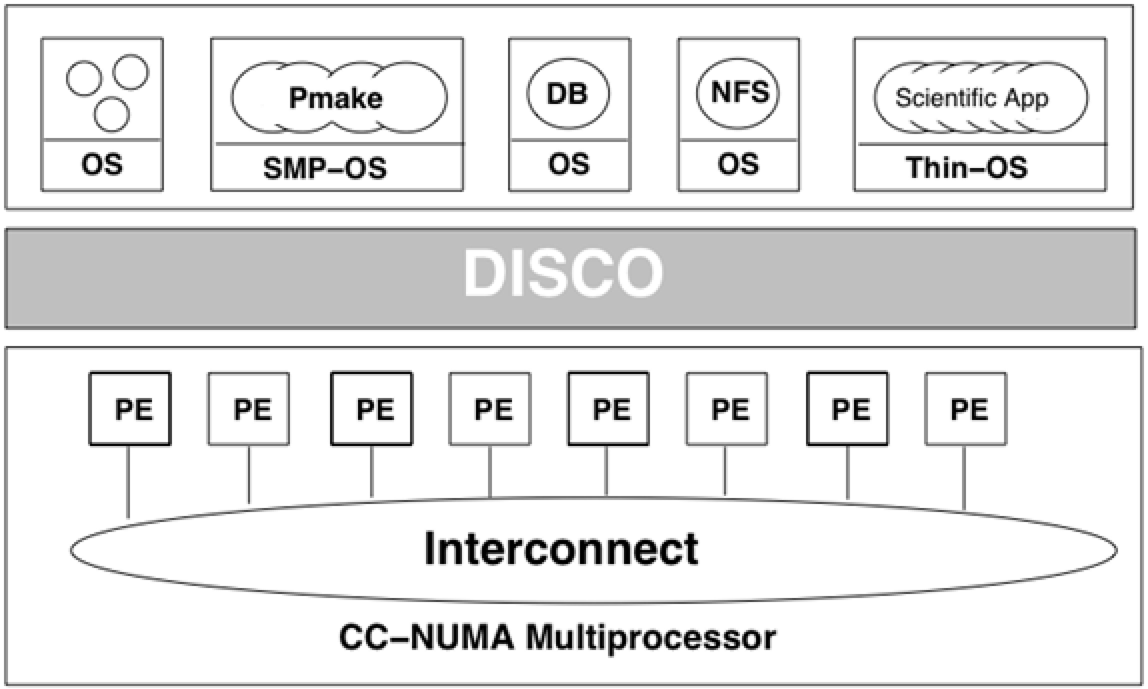


VMware loses Mendel Rosenblum, co-founder and husband of fired CEO Diane Greene

Share/Email Tweet This 3 Comments Print Newsletter Sign-Up

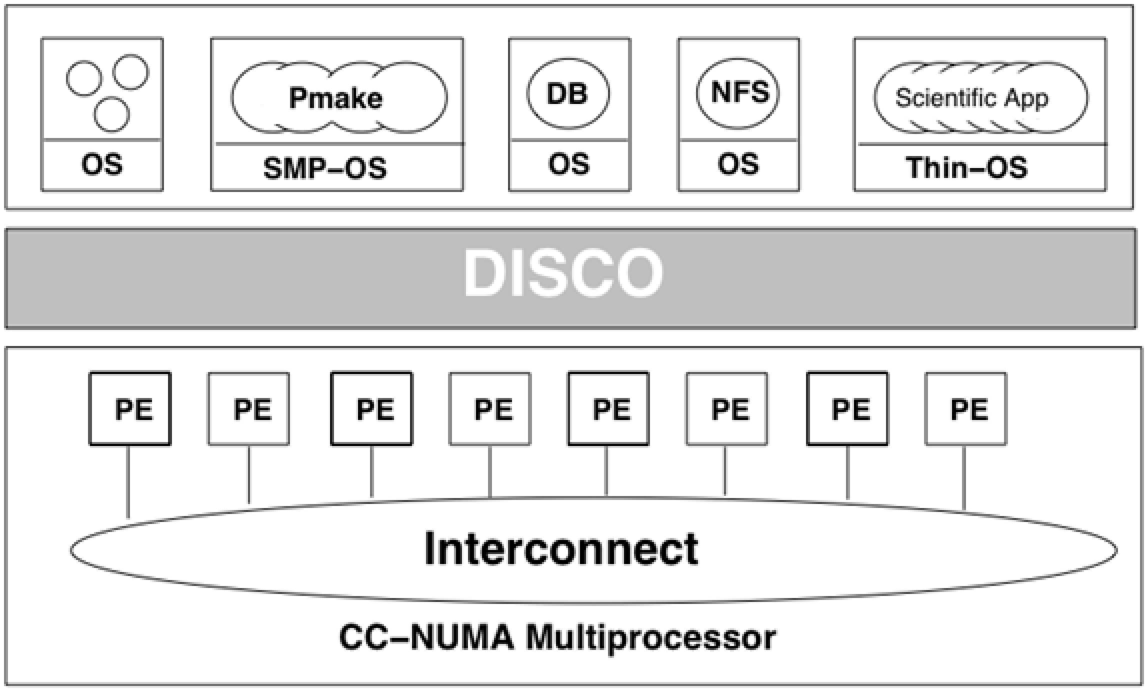
Former VMware CEO Diane Greene's husband, Mendel Rosenblum, has followed his wife out the door, announcing his resignation just days before VMware hosts an annual event to showcase its virtualization technology.

Virtual Machine Monitors



- Additional layer of software between the hardware and the operating system

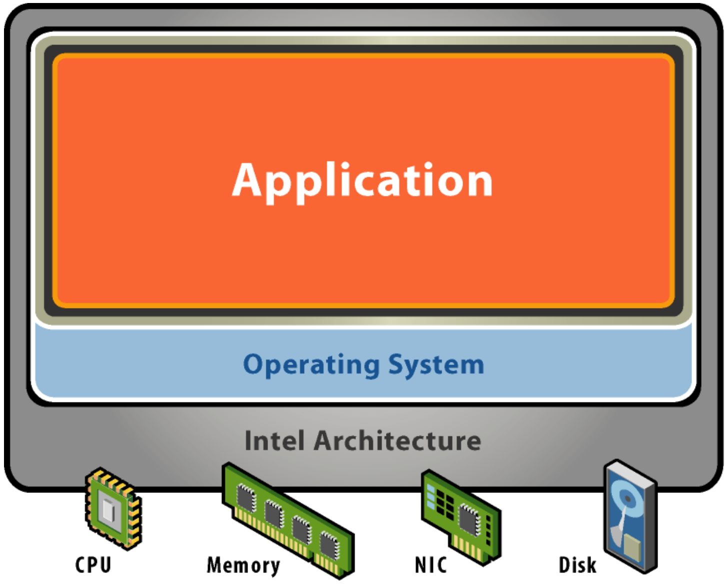
Virtual Machine Monitors



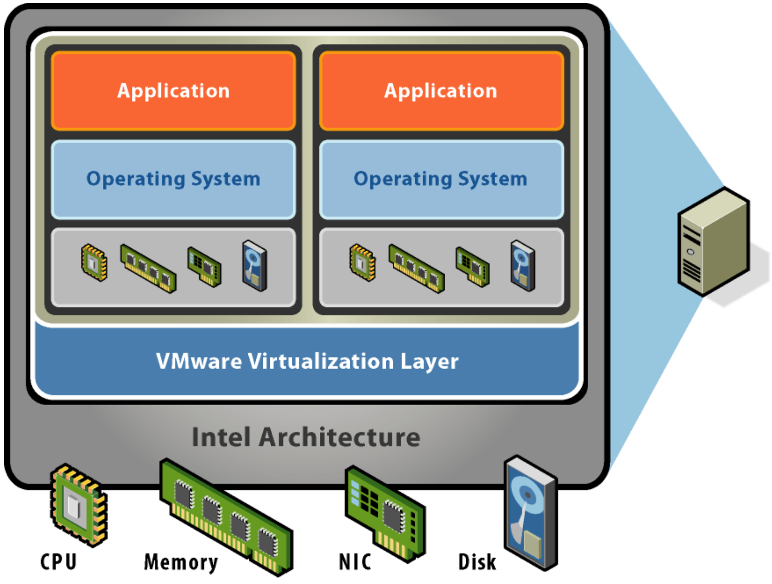
- Virtualizes and manages all the resources so that multiple virtual machines can coexist on the same multiprocessor

VMware Architecture

System ***without*** VMware Software



System ***with*** VMware Software



DISCO: Contributions

- Scalability
  - Explicit memory sharing is allowed
- Flexibility
  - Support for specialized OSs
- ccNUMA: Scalability and Fault-Containment
  - Failures in the system software is contained in VM
- NUMA: Memory Management Issues
  - Dynamic page migration and page replication

DISCO: Disadvantages

- Overheads
  - Virtualization of the hardware resources
- Resource Management Problems
  - The lack of high-level knowledge
- Sharing and Communication Problems
  - Running multiple independent operating systems

DISCO: Interface

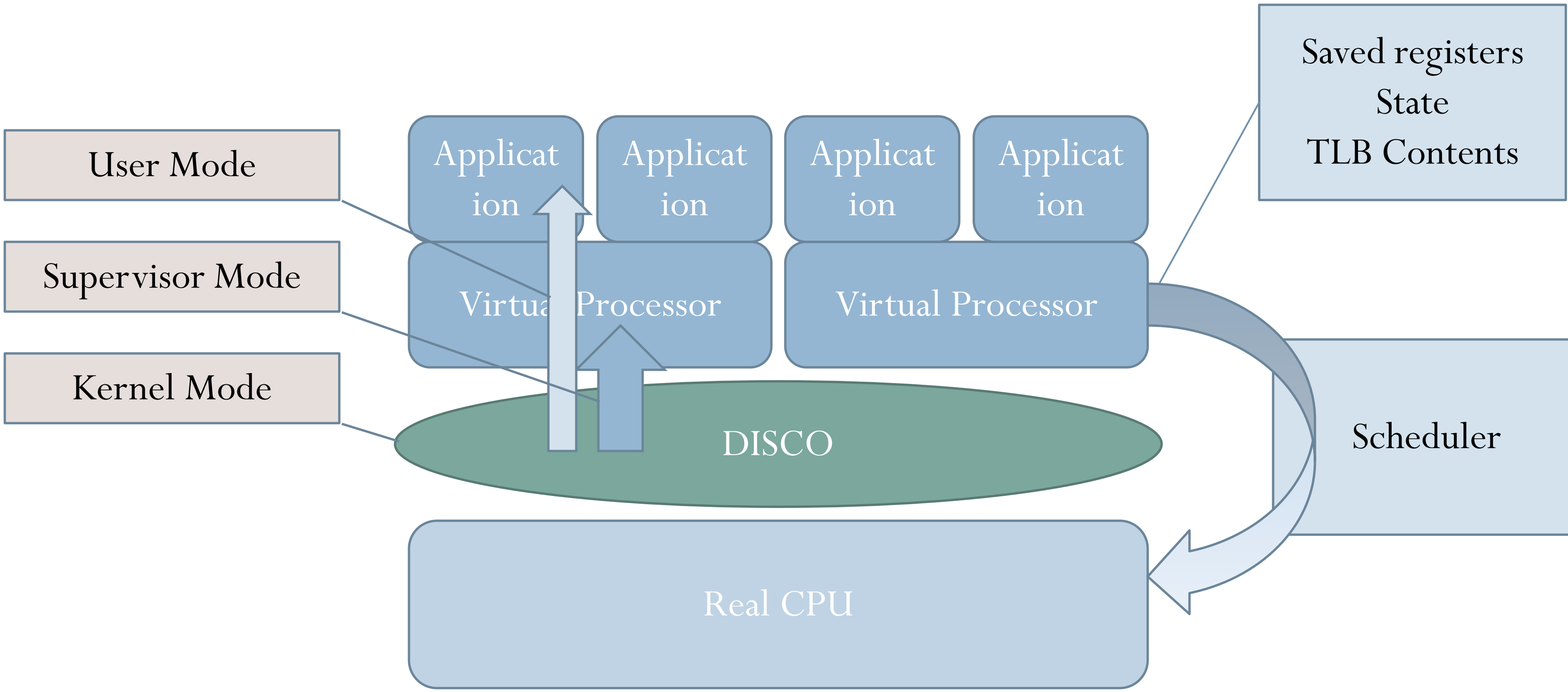
- Processors 
  - The virtual CPUs of DISCO provide the abstraction of a MIPS R10000 processor.
- Physical Memory 
  - Abstraction of main memory residing in a contiguous physical address space starting at address zero.
- I/O Devices   
  - Each I/O device is virtualized
  - Special abstractions for the disk and network devices

DISCO: Implementation

- Implemented as a multi-threaded shared memory program
  - NUMA memory placement
  - cache-aware data structures
  - inter-processor communication patterns

- Code segment of DISCO is replicated into all the memories of FLASH machine to satisfy all instruction cache misses from the local node.

DISCO: Virtual CPUs



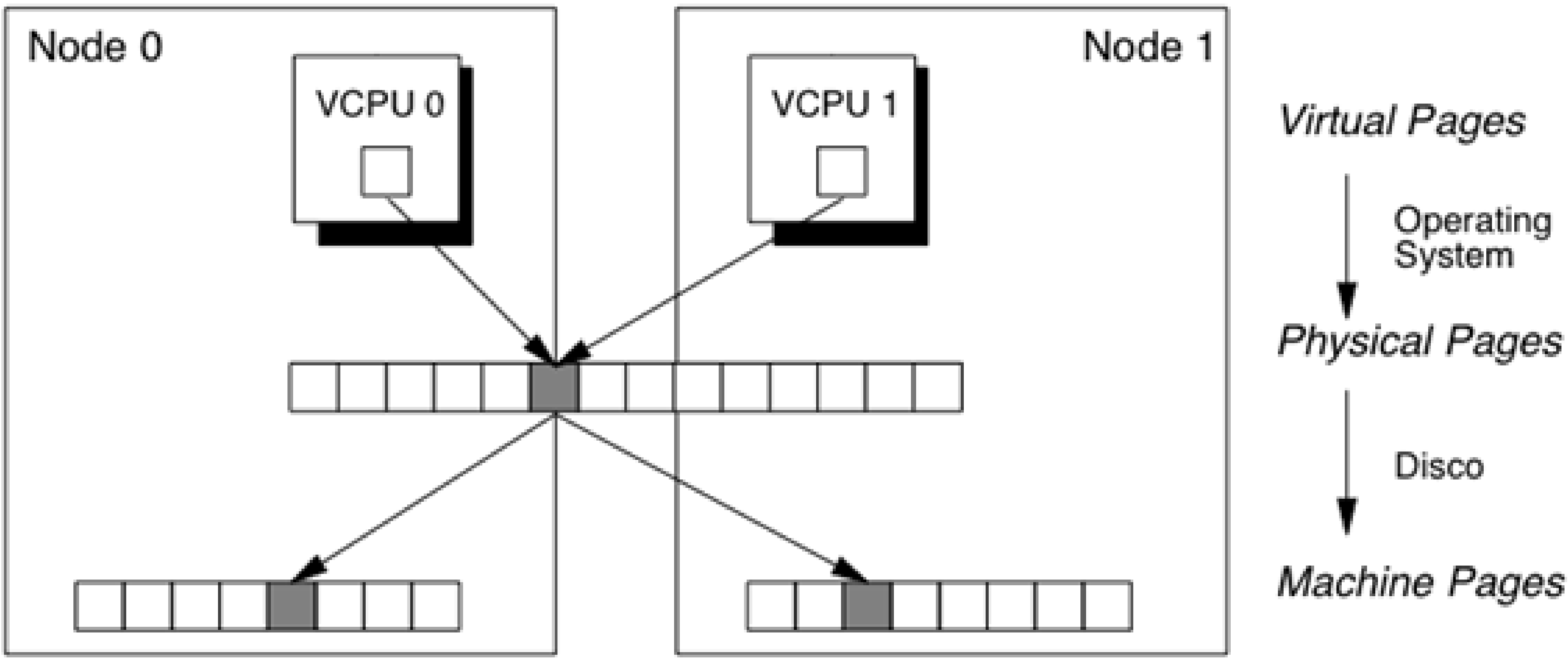
DISCO: Virtual Physical Memory

- DISCO maintains physical-to-machine address mappings.
  - The VMs use physical addresses, and DISCO maps them to machine addresses.
  - DISCO uses the software-reloaded TLB for this.
- TLB must be flushed on virtual CPU switches; Disco caches recent virtual-to-machine translations in a second-level software TLB.

DISCO: NUMA Memory Management

- fast translation of the virtual machine's physical addresses to real machine pages
- the allocation of real memory to virtual machines
- dynamic page migration and page replication system to reduce long memory accesses
  - Pages heavily used by one node are migrated to that node
  - Pages that are read-shared are replicated to the nodes most heavily accessing them
  - Pages that are write-shared are not moved
  - Number of moves of a page limited

DISCO: Transparent Page Replication

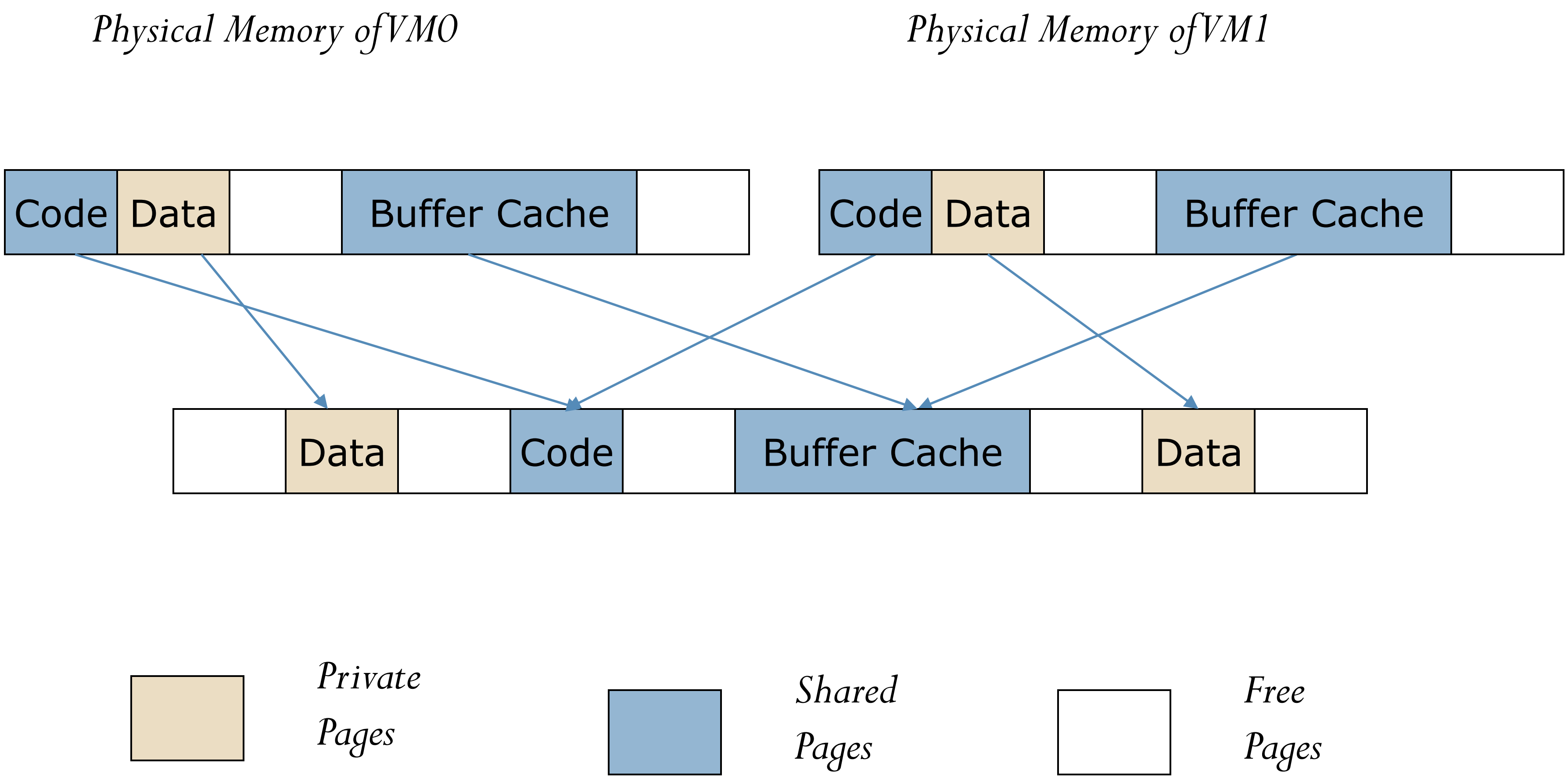


- Two different virtual processors of the same virtual machine read-share the same physical page, but each virtual processor accesses a local copy.
- memmap tracks which virtual page references each physical page.

DISCO: Virtual I/O Devices

- DISCO intercepts all device accesses from the VM and eventually forwards them to the physical devices.
- Installing drivers for DISCO I/O in the guest OS.
- DISCO must intercept DMA requests to translate the physical addresses into machine addresses. DISCO's device drivers then interact directly with the physical device.
- All the virtual machines can share the same root disk containing the kernel and application programs.

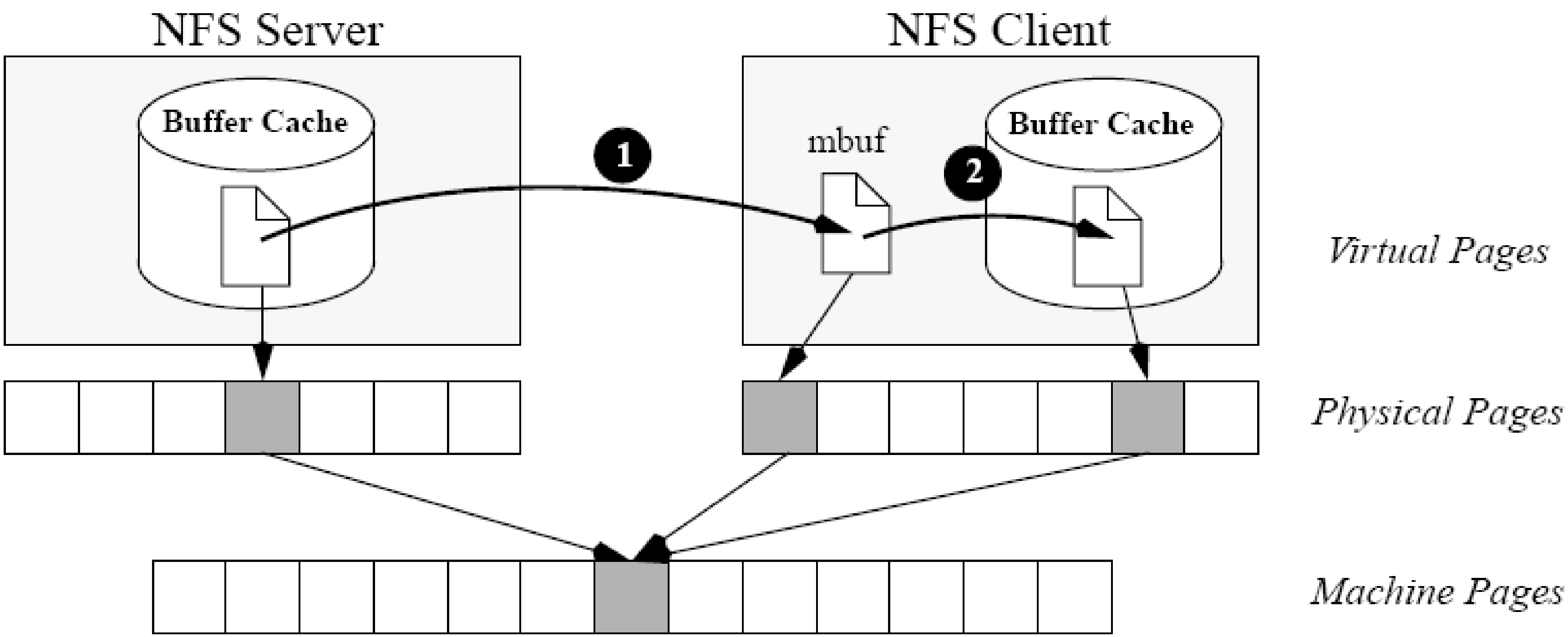
DISCO: Copy-on-write Disks



DISCO: Virtual Network Interface

- Virtual subnet and network interface use copy-on-write mapping to share the read only pages
- Persistent disks can be accessed using standard system protocol NFS
- Provides a global buffer cache that is transparently shared by independent VMs

DISCO: Transparent Sharing of Pages Over NFS



- The monitor's networking device remaps the data page from the source's machine address space to the destination's.
- The monitor remaps the data page from the driver's mbuf to the client's buffer cache.

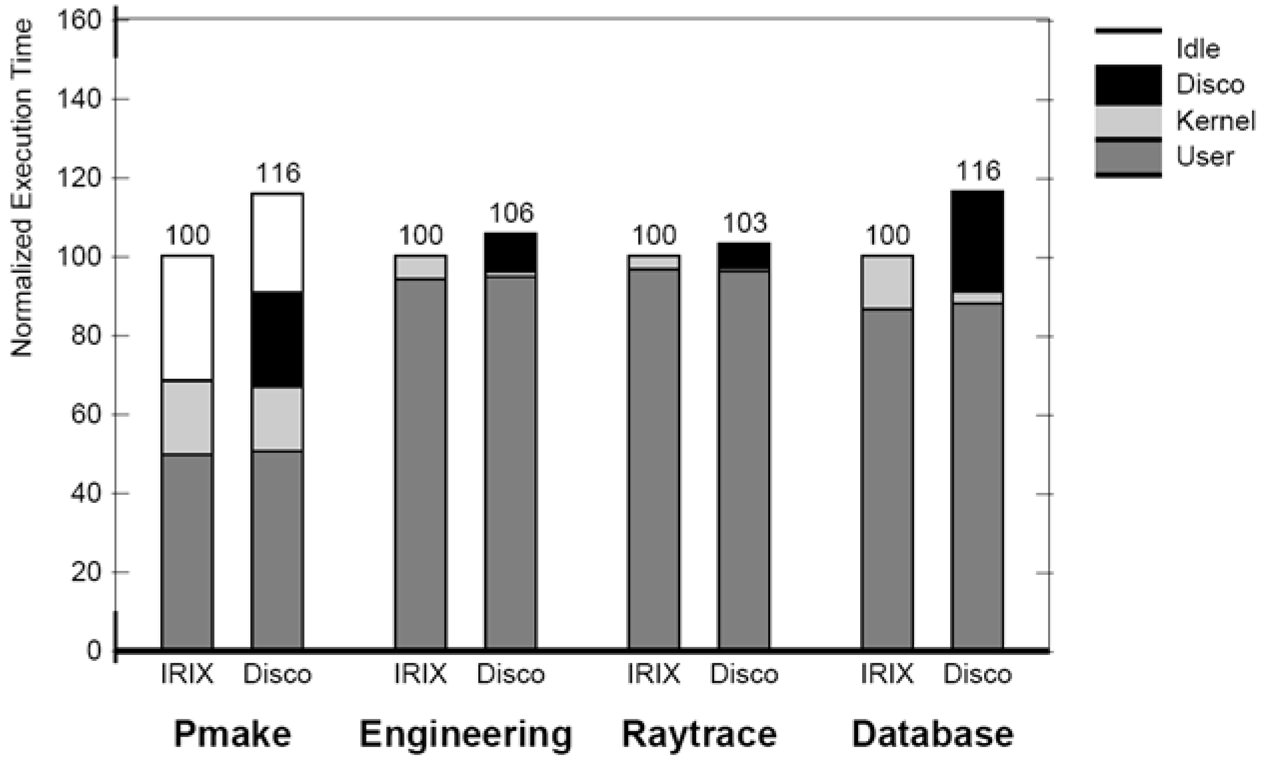
DISCO: Performance

- SimOS is configured to resemble a large-scale multiprocessor with performance characteristics similar to FLASH.
- The processors have the on-chip caches of the MIPS R10000 (32KB split instruction/data) and a 1MB board-level cache.
- Simulation models are too slow for the workloads planned.

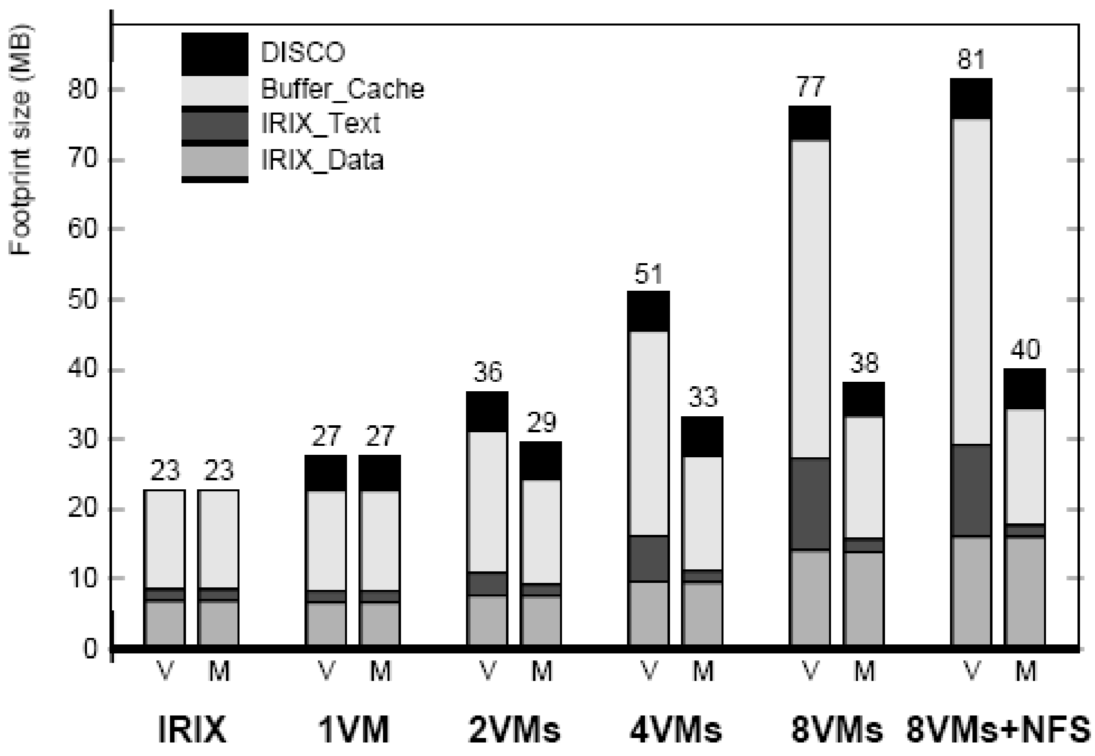
DISCO: Workloads

Workload	Environment	Description	Characteristics	Execution Time
Pmake	Software Development	Parallel compilation (-J2) of the GNU chess application	Multiprogrammed, short-lived, system and I/O intensive processes	3.9 sec
Engineering	Hardware Development	Verilog simulation (Chronologics VCS) + machine simulation	Multiprogrammed, long running processes	3.5 sec
Splash	Scientific Computing	Raytrace from SPLASH-2	Parallel applications	12.9 sec
Database	Commercial Database	Sybase Relational Database Server decision support workload	Single memory intensive process	2.0 sec

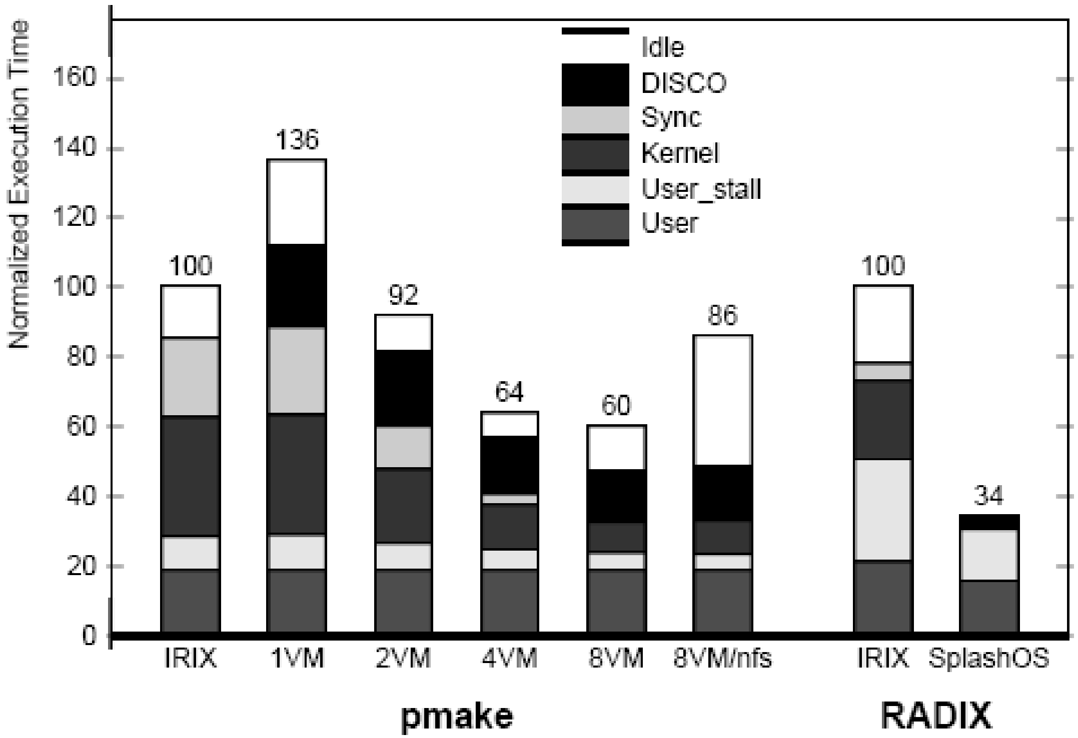
DISCO: Execution Overheads



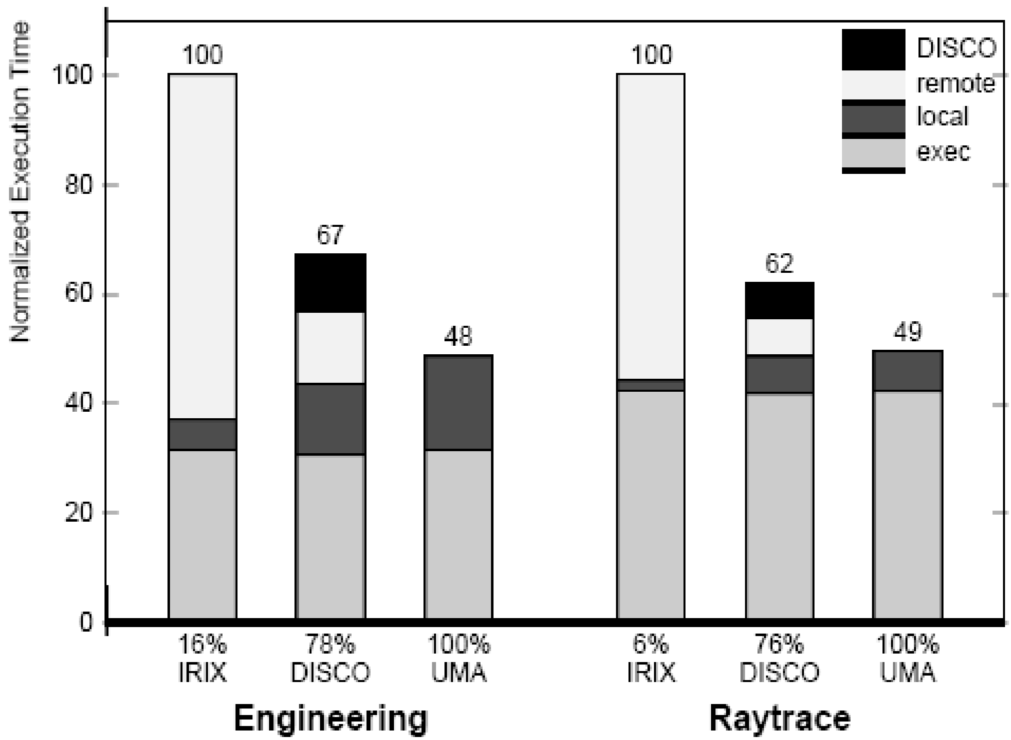
DISCO: Memory Overheads



DISCO: Workload Scalability



DISCO: Page Migration and Replication



DISCO vs. Exokernel

- The Exokernel safely multiplexes resources between user-level library operating systems.
- Both DISCO and Exokernel support specialized operating systems.
- DISCO differs from Exokernel in that it virtualizes resources rather than multiplexing them, and can therefore run commodity operating systems without significant modifications.

Tornado: Maximizing Locality and Concurrency in a Shared Memory Multiprocessor Operating System

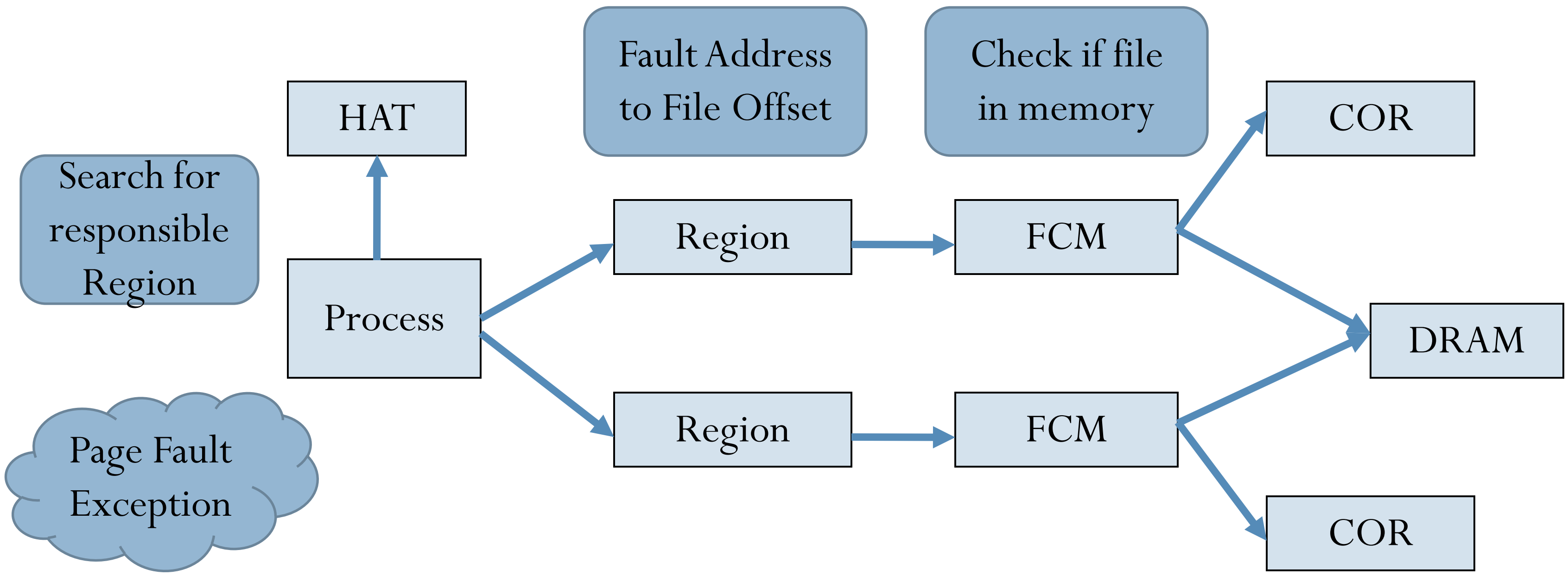
- Locality is as important as concurrency.

- Three Key Innovations:
  - Clustered Objects
  - New Locking Strategy
  - Protected Procedure Call

Memory Locality Optimization

- minimizing read/write and write sharing so as to minimize cache coherence overheads
- minimizing false sharing
- minimizing the distance between the accessing processor and the target memory module

Tornado: Object Oriented Structure



HAT	Hardware Address Translation
FCM	File Cache Manager
COR	Cached Object Representative
DRAM	Memory manager

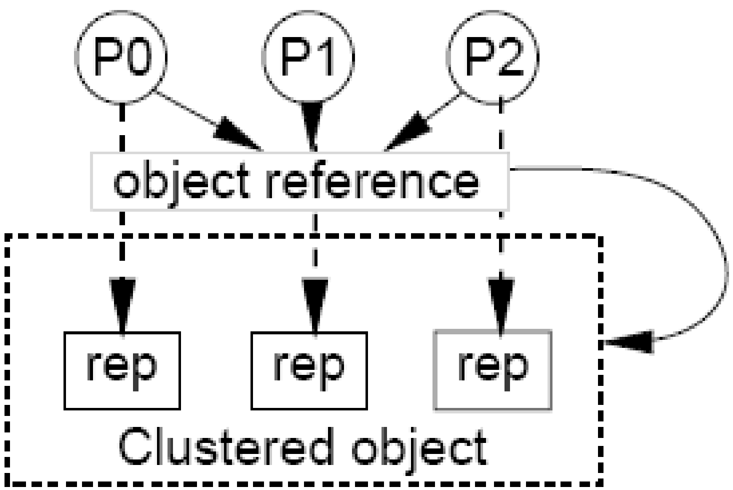
File cached in Memory

Return addr of Physical Page Frame to Region, Ragion calls HAT to map the page.

File not cached in Memory

Request new physical page frame from DRAM and ask COR to fill the page.

Tornado: Clustered Objects

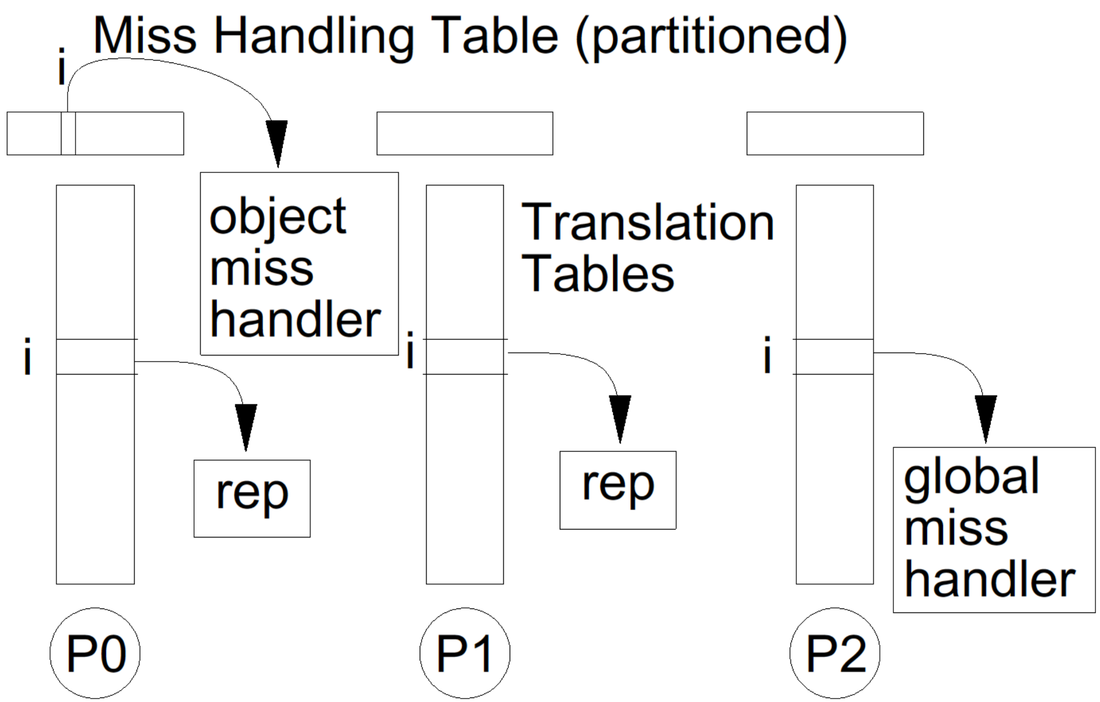


Consistency preserved by coordination via shared memory or PPC

Tornado: CO Implementation

Translation Tables store pointer to the responsible rep

reps created on demand when first accessed



First call is accepted by **Global Miss Handler**. **Object Miss Handler** creates a rep if necessary and installs it in the Translation Table. The call is restarted using this rep.

Tornado: A New Locking Strategy

- Aim is to minimize the overhead of locking instructions
- Lock contention is limited
  - Encapsulating the locks in objects to limit the scope of locks
  - Using clustered objects to provide multiple copies of a lock
- Semi-automatic garbage collection is employed
  - A clustered object is destroyed only if
    - No persistent references exist
    - All temporary references are eliminated

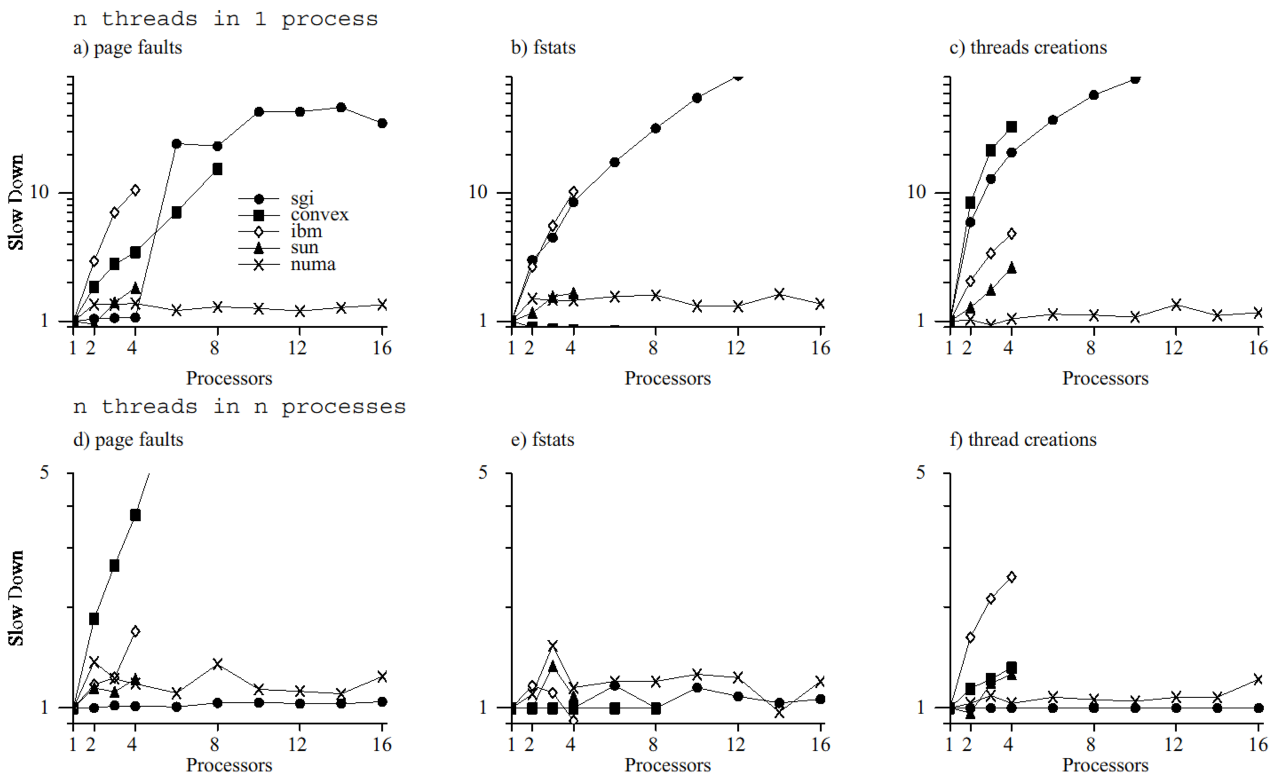
Persistent References are those stored in (shared) memory; they can be accessed by multiple threads.

Tornado: Protected Procedure Calls

- Objective: Providing locality and concurrency during IPC
- A PPC is a call from a client object to a server object
  - Acts like a clustered object call
  - On demand creation of server threads to handle calls

- Advantages of PPC
  - Client requests are always serviced on their local processor
  - Clients and servers share the processor in a manner similar to handoff scheduling
  - There are as many threads of control in the server as client requests

Tornado: Performance



n threads in one process

n processes, each with one thread

In-Core Page Fault Each worker thread accesses a set of in-core unmapped pages in independent memory regions.

File Stat Each worker thread repeatedly fstats an independent file.

Thread Creation Each worker successively creates and then joins with a child thread

Conclusion

- DISCO
  - Virtual Machine layer
  - OS independent
  - manages resources, optimizes sharing primitives, and mirrors the hardware
- Tornado
  - Object Oriented Design
  - flexible and extensible OS
  - Locality addressed by clustered objects and PPC

Thank You..