操作系统　Operation System

第九章 分布式OS

沃天宇
woty@buaa.edu.cn

Scalable Computing: the Way to High-performance

IBM BG/P

Source: ANL ALCF

Petaflops System

72 Racks

1 PF/s
144 TB

Rack

32 Node Cards
1024 chips, 4096 procs

Cabled 8x8x16

14 TF/s
2 TB

Node Board

(32 chips 4x4x2)
32 compute, 0-2 IO cards

435 GF/s
64 GB

Compute Card

1 chip, 20 DRAMs

13.6 GF/s
2.0 GB DDR
Supports 4-way SMP

Chip

4 cores

850 MHz
8 MB EDRAM

Front End Node / Service Node
System p Servers
Linux SLES10

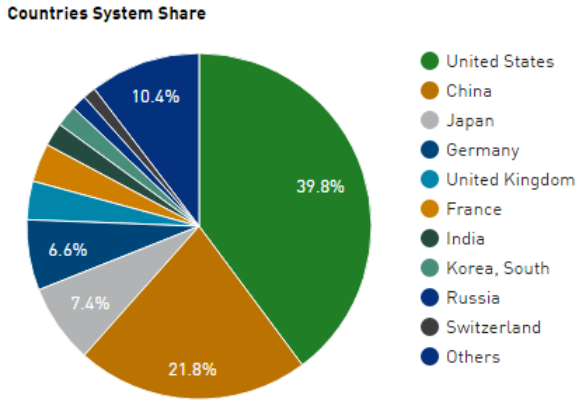
Maximum System

256 racks
3.5 PF/s
512 TB

HPC SW:
Compilers
GPFS
ESSL
Loadleveler

天河2号

- **峰值：55PFlops（Top1 Since 2013/6）**
  - 1000TB Mem, 17808kW, Kylin OS
- 美国建设中
  - Summit 150-300 Pflops（2018）
  - Sierra 100 Pflops

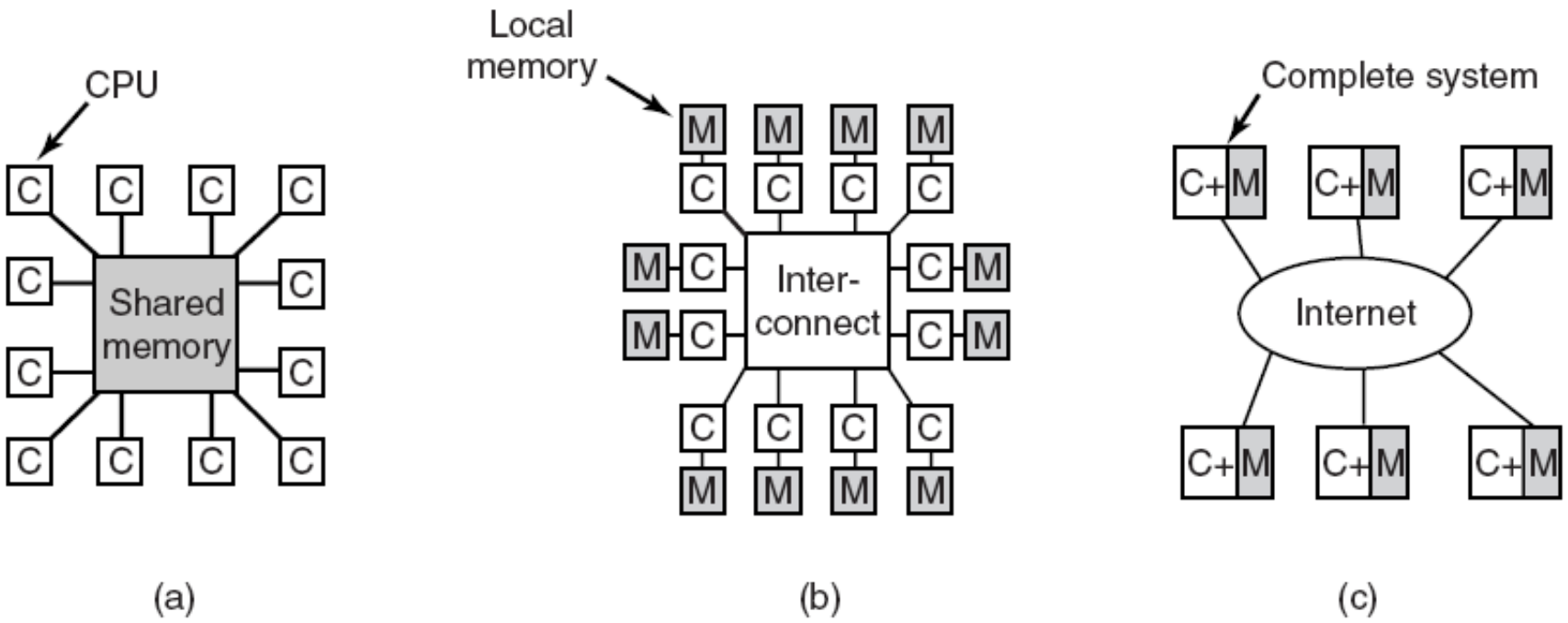


TOP 10 Sites for November 2015

For more information about the sites and systems in the list, click on the links or view the complete list.

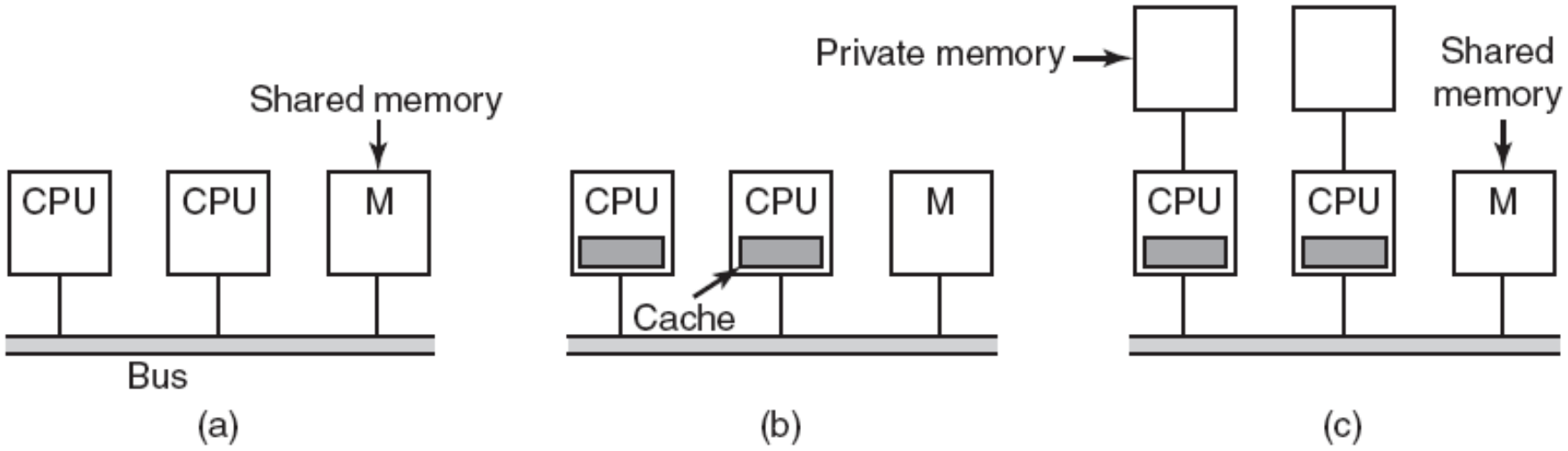
RANK	SITE	SYSTEM	CORES	RMAX (TFLOP/S)	RPEAK (TFLOP/S)	POWER (KW)
1	National Super Computer Center in Guangzhou China	**Tianhe-2 (MilkyWay-2)** - TH-IVB-FEP Cluster, Intel Xeon E5-2692 12C 2.200GHz, TH Express-2, Intel Xeon Phi 31S1P NUDT	3,120,000	33,862.7	54,902.4	17,808
2	DOE/SC/Oak Ridge National Laboratory United States	**Titan** - Cray XK7 , Opteron 6274 16C 2.200GHz, Cray Gemini interconnect, NVIDIA K20x Cray Inc.	560,640	17,590.0	27,112.5	8,209
3	DOE/NNSA/LLNL United States	**Sequoia** - BlueGene/Q, Power BQC 16C 1.60 GHz, Custom IBM	1,572,864	17,173.2	20,132.7	7,890

多处理机系统的三种模型



(a) 共享存储的多处理机. (b) 基于消息传递的多处理机. (c) 广域分布式系统.

统一存储器访问UMA的总线型多处理机体系结构

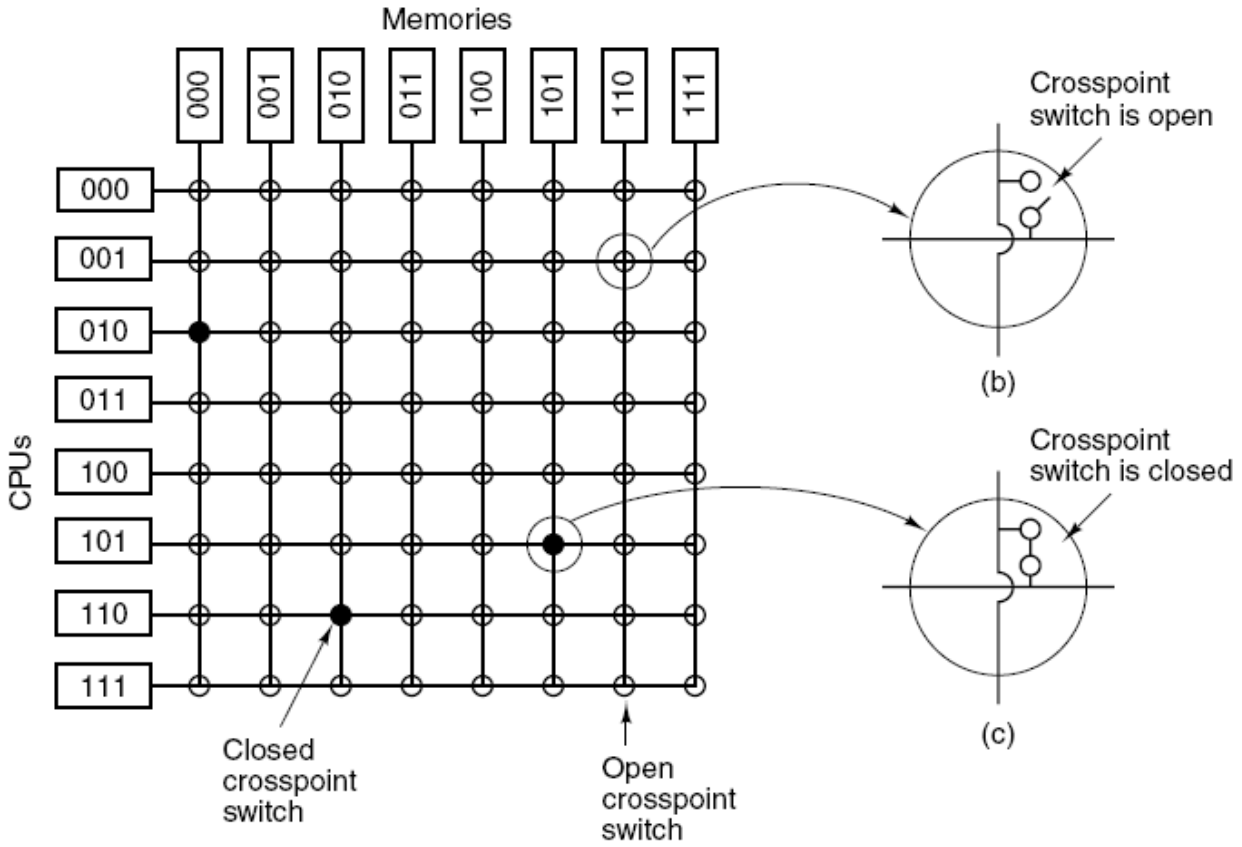


（a）无缓存.（b）有缓存.（c）有缓存和私有存储

问题：单总线的可扩展性：16-32 CPU

统一存储器访问UMA的交叉开关型多处理机体系结构

问题：交叉开关的可扩展性，考虑1000个CPU和1000个内存



问题：8*8无阻塞交叉网络开关有多少种不同开关设置组合？

(a) 8 × 8 交叉开关. (b) 打开的交叉点. (c) 闭合的交叉点.

使用多阶段交换网络的UMA多处理机(1)

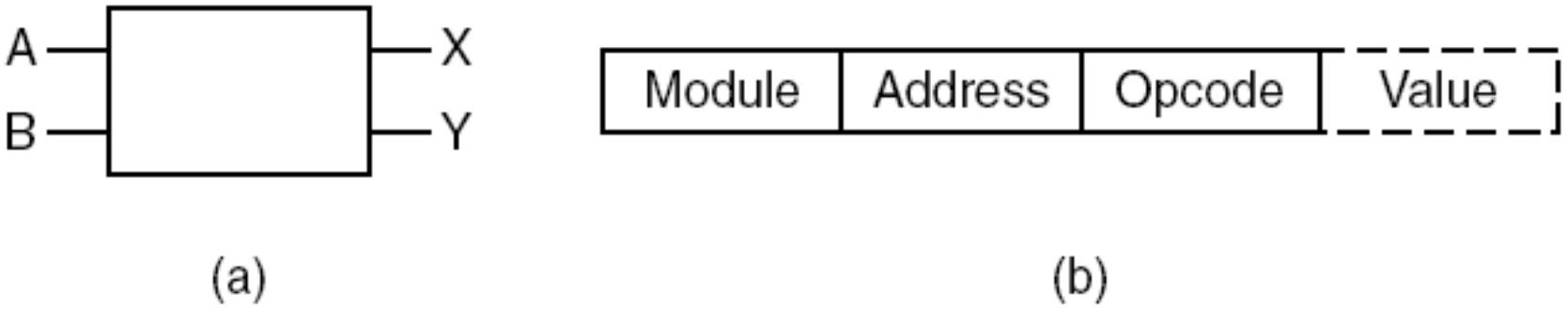
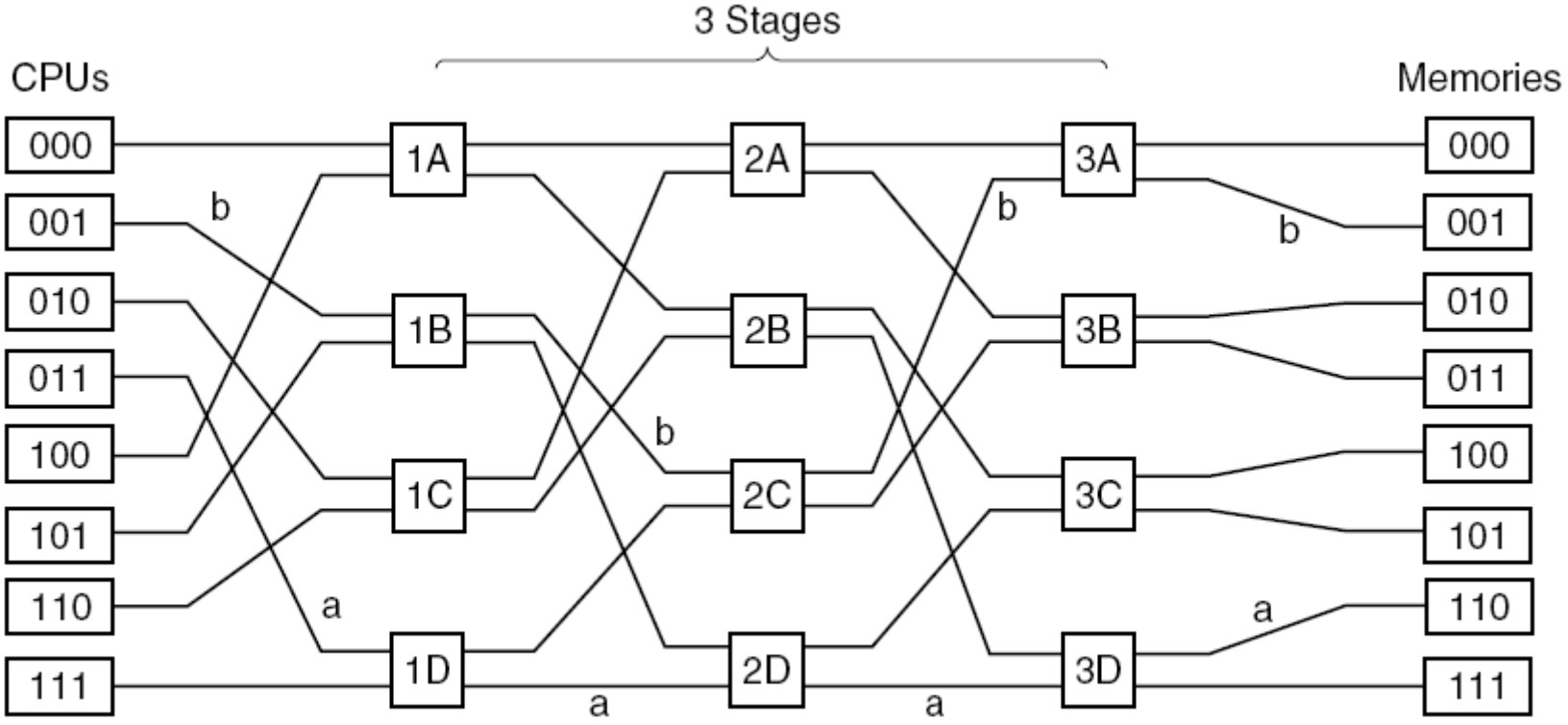


Figure 8-4. (a) A 2 × 2 switch with two input lines, A and B, and two output lines, X and Y. (b) A message format.

Module域指明如何设置开关（X or Y）

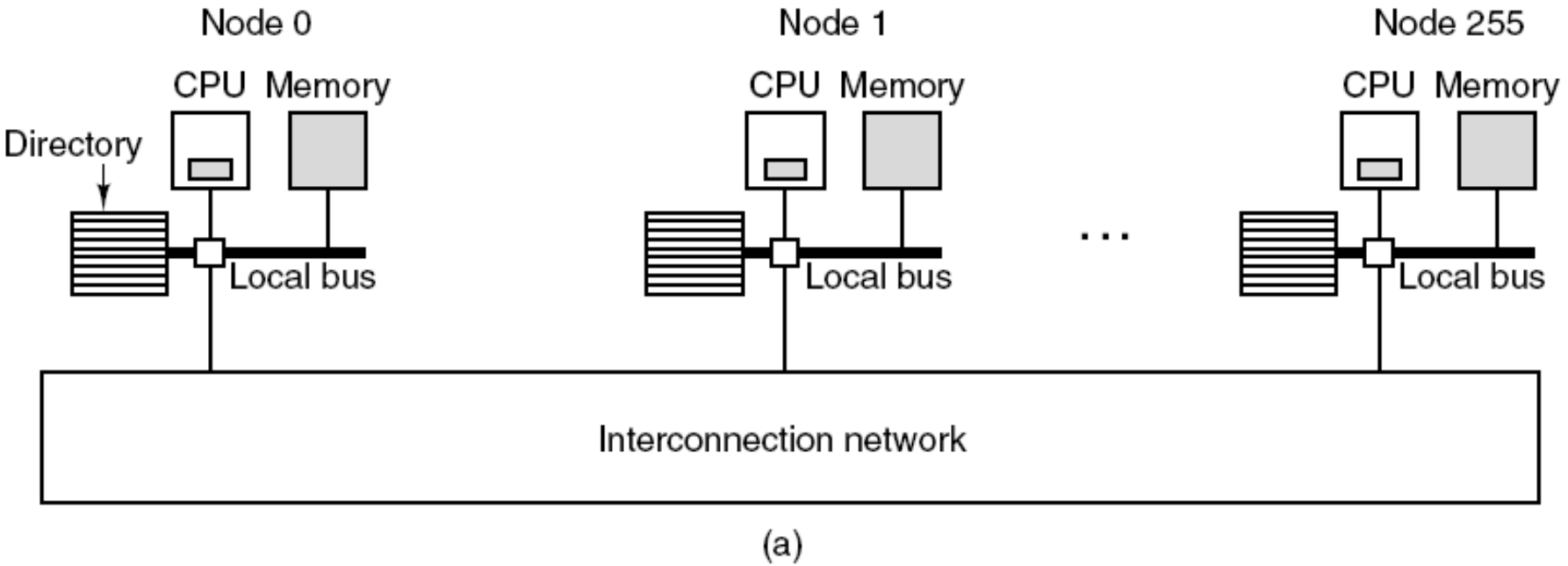
使用多阶段交换网络的UMA多处理机(2)

Figure 8-5. An omega switching network.



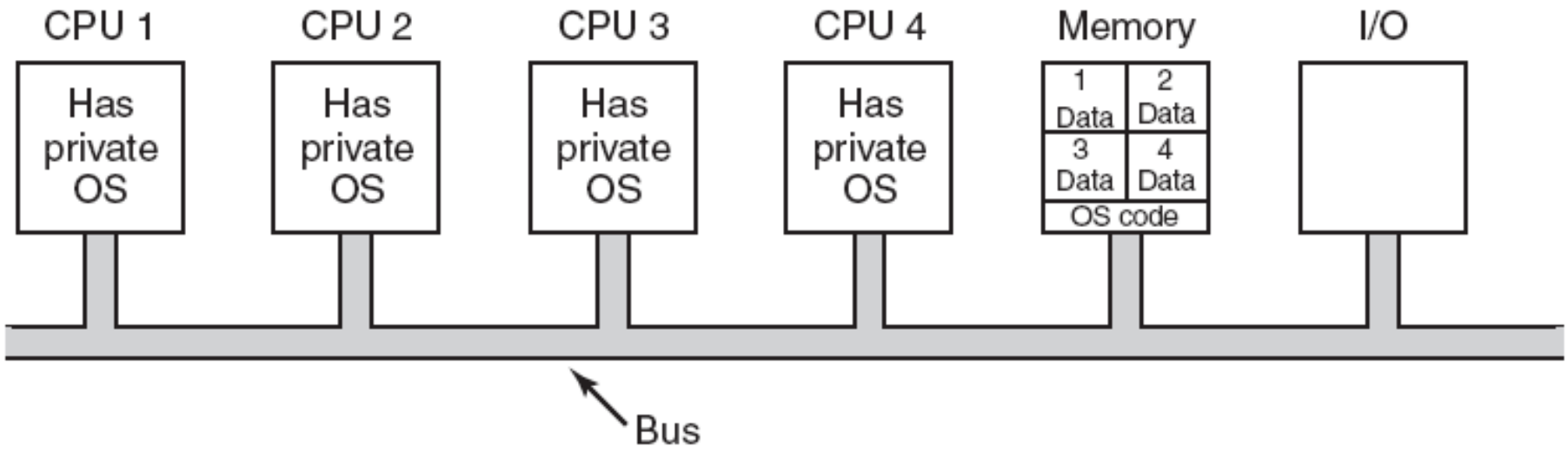
共需要$(n/2)\log_2 n$个交叉开关，比n^2可扩展

非一致内存访问NUMA的多处理机



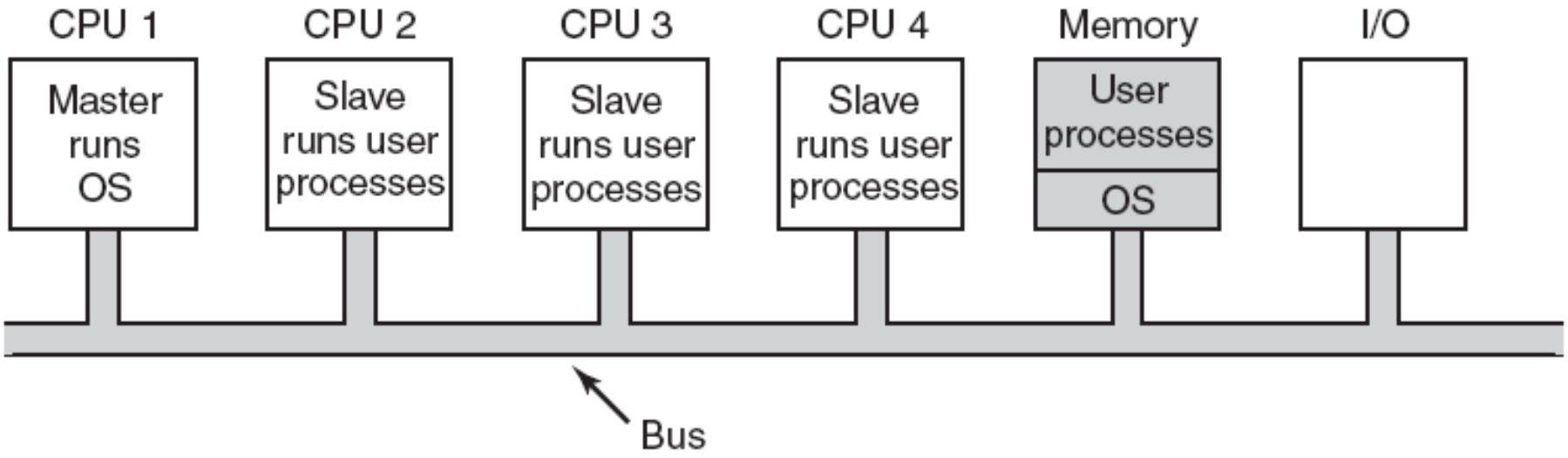
访问远程内存比访问本地内存要慢

每个CPU拥有各自的操作系统



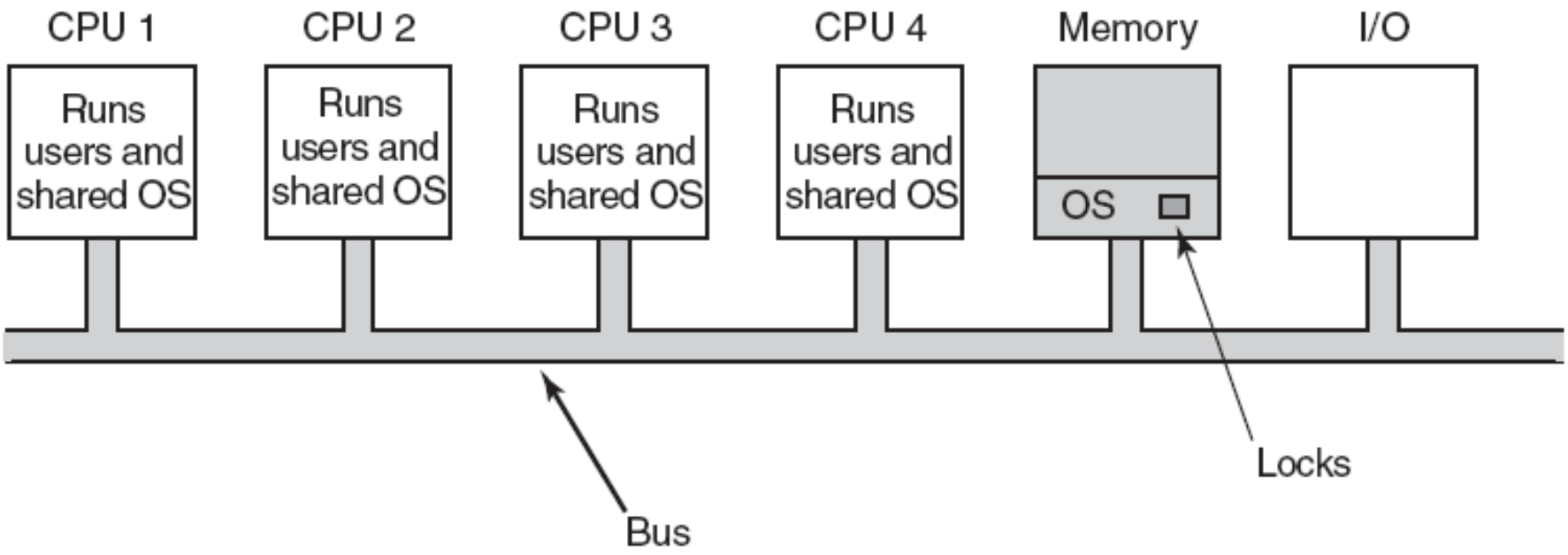
每个CPU具有自己的内存，用于保存私有数据；
操作系统的代码被所有CPU进行共享。

主从式多处理机



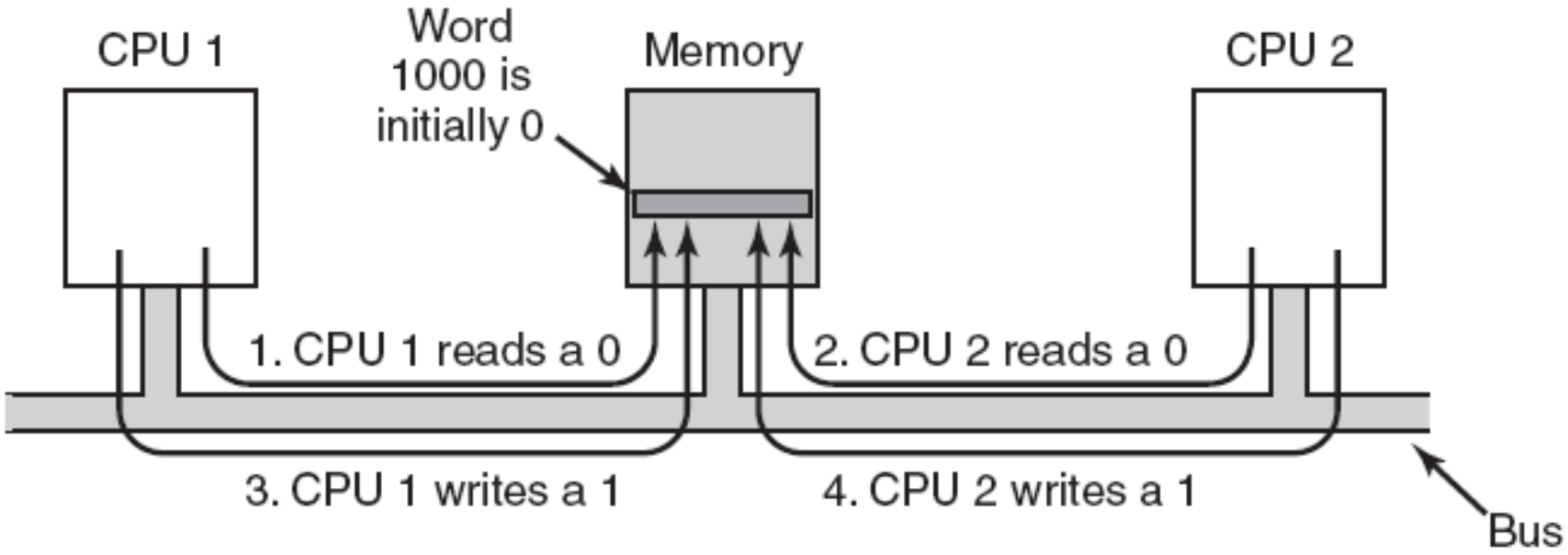
所有的系统调用都由主CPU处理，从CPU只负责运行用户进程

对称多处理机



互斥访问、死锁处理

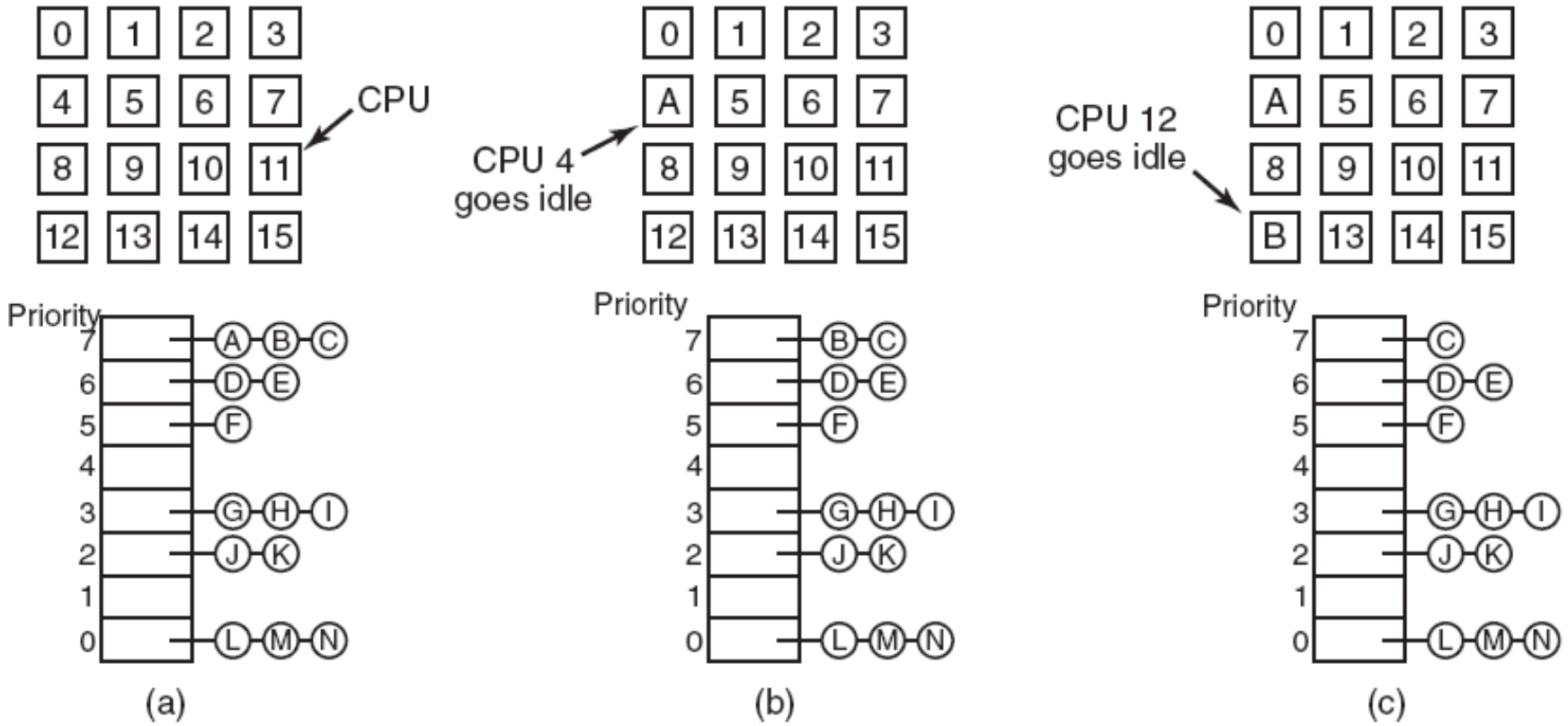
多处理机同步与互斥



TSL指令：必须锁住总线，否则会失效

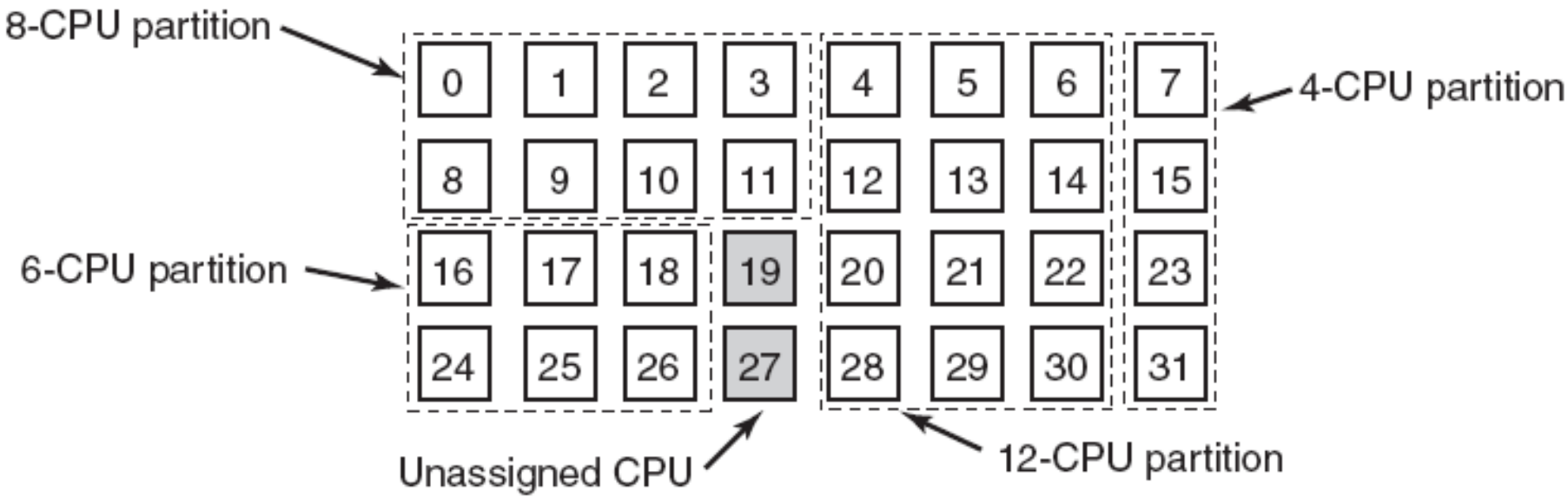
纯软件的同步协议？Peterson

多处理机的调度–分时系统



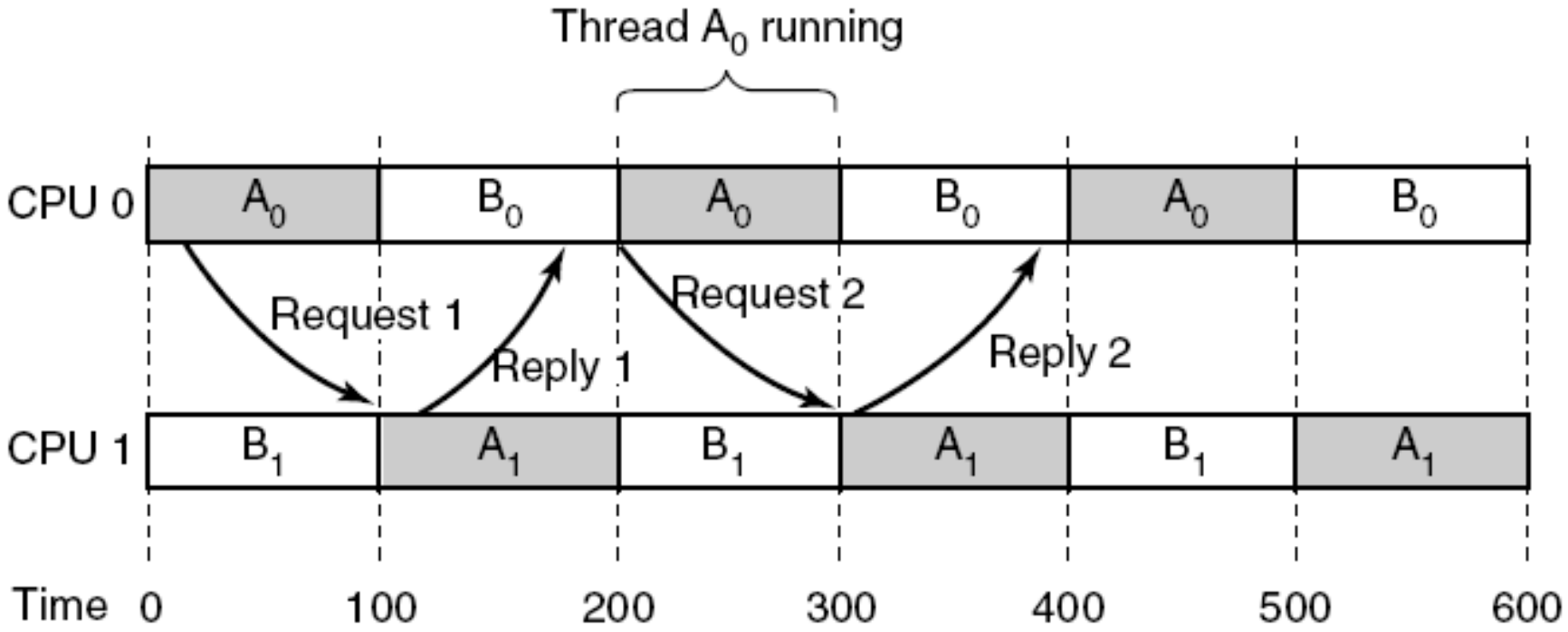
使用一个数据结构进行调度（线程之间无依赖关系）

多处理机调度-空间共享



32个CPU分成4个区域，两个CPU可用
（某些线程需要同时进行调度）

群调度Gang Scheduling（1）



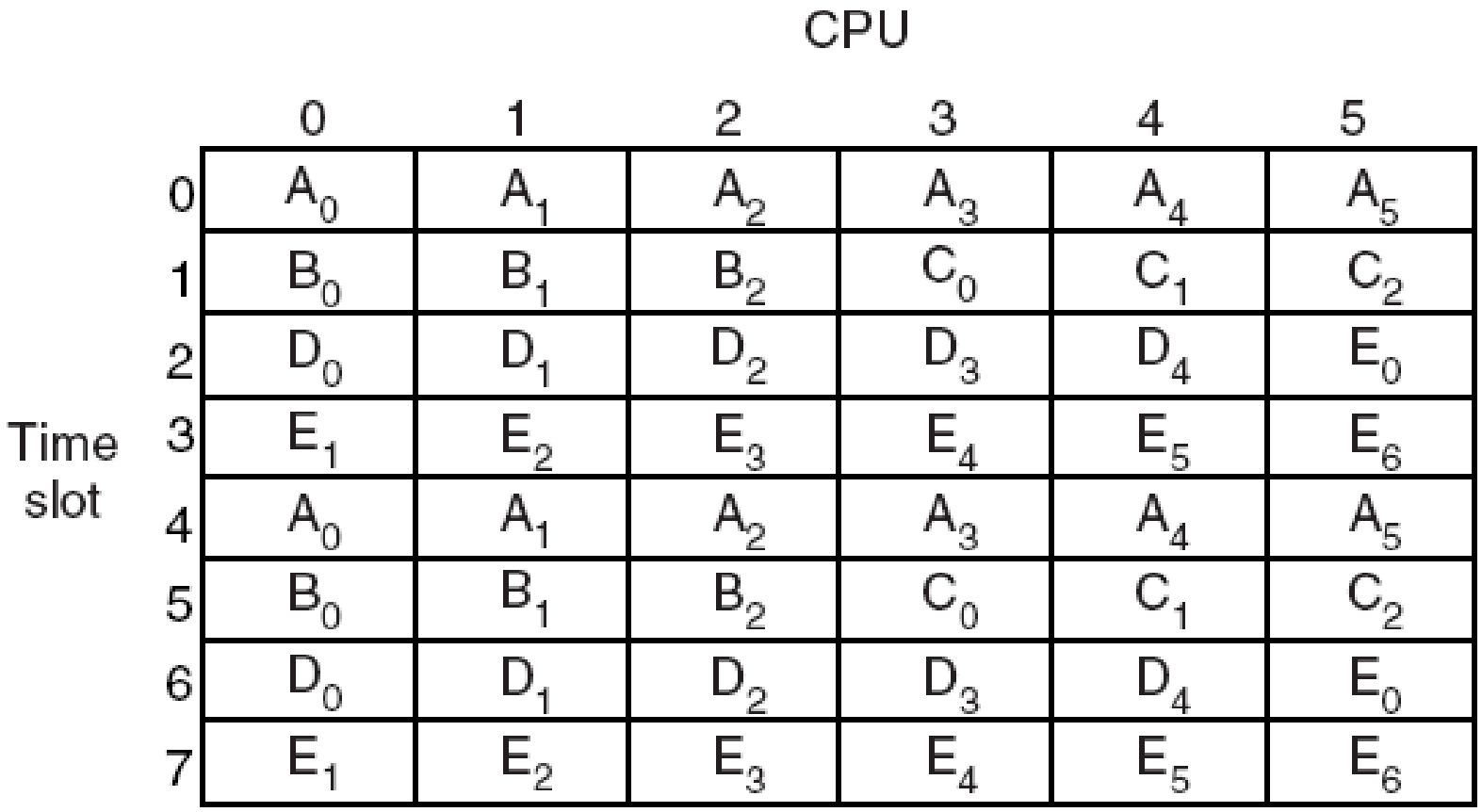
空间共享与分时的结合，让一个进程的所有线程一起调度运行。

Gang Scheduling (2)

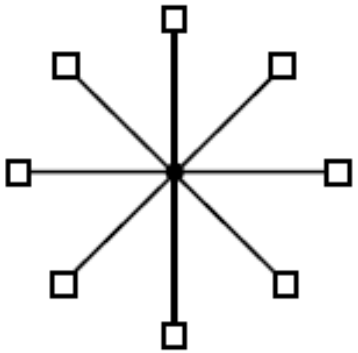
The three parts of gang scheduling:

1. Groups of related threads are scheduled as a unit, a gang.
2. All members of a gang run simultaneously, on different timeshared CPUs.
3. All gang members start and end their time slices together.

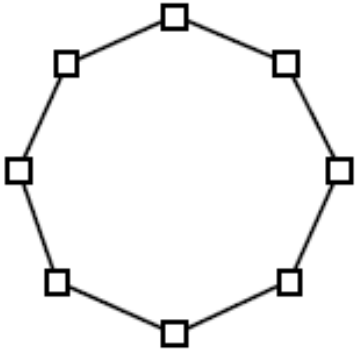
Gang Scheduling (3)



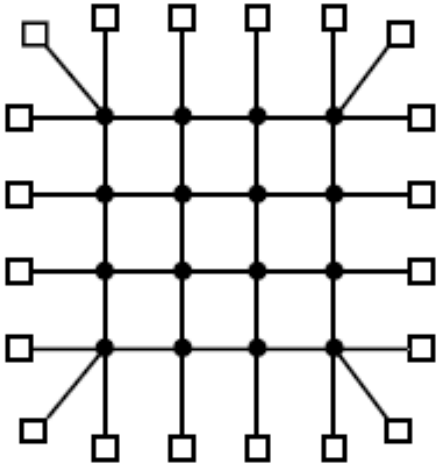
多计算机之间的互联：Interconnection Technology (1)



(a)

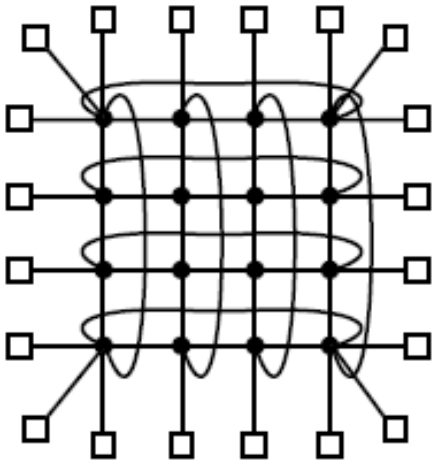


(b)

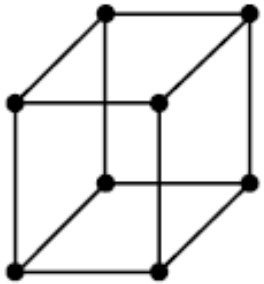


(c)

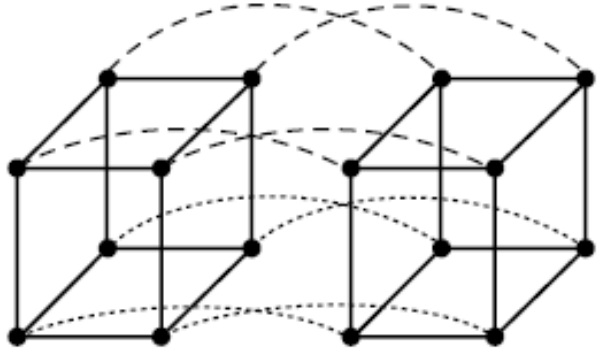
Interconnection Technology (2)



(d)

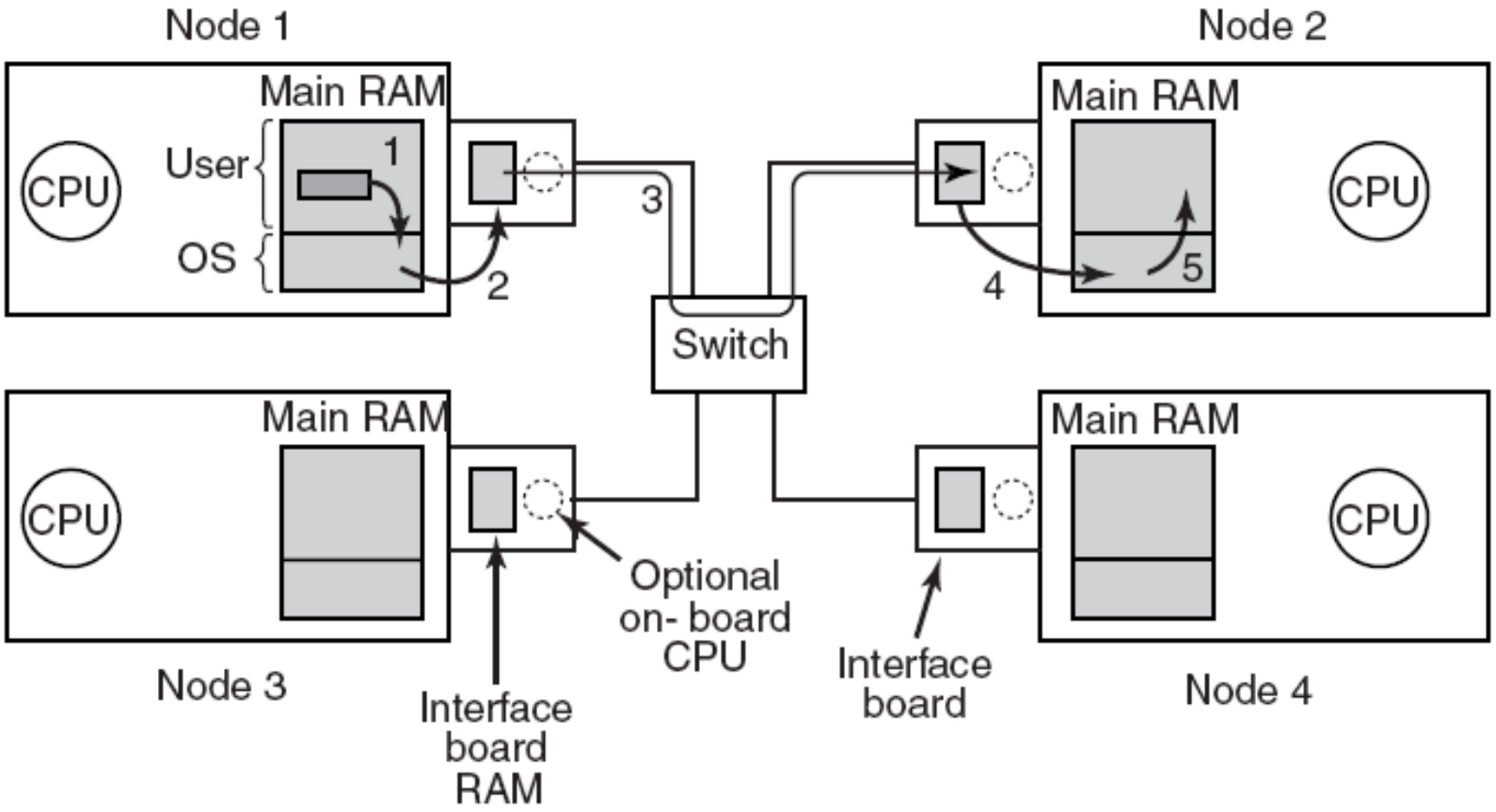


(e)



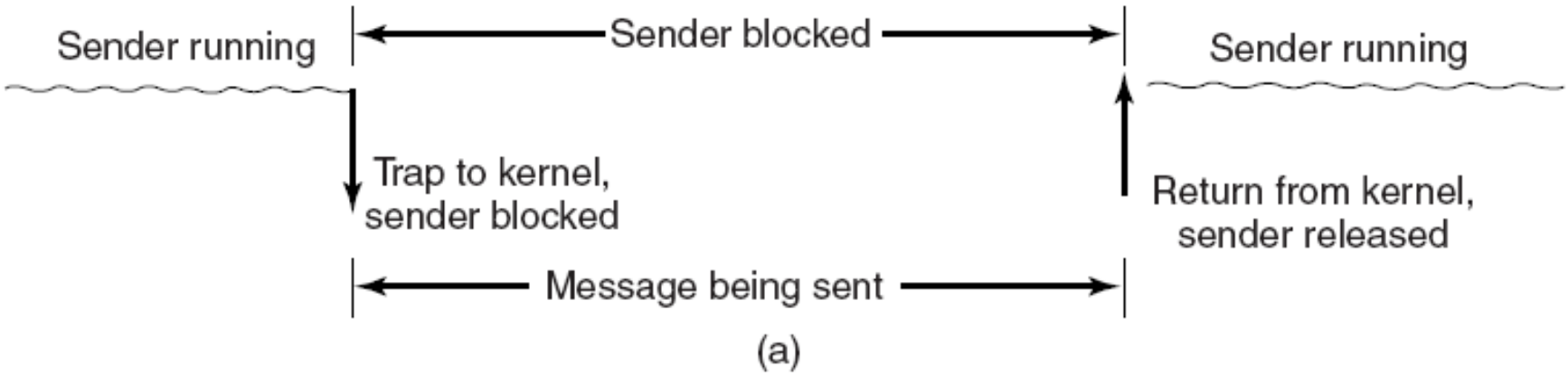
(f)

Network Interfaces



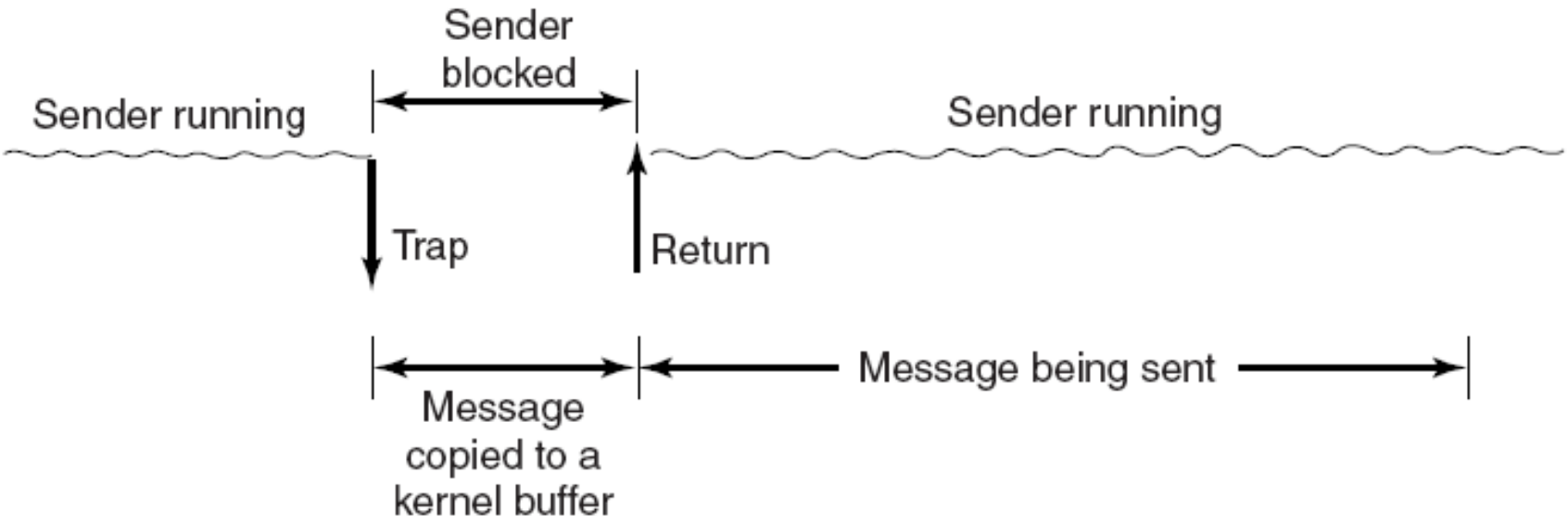
阻塞与非阻塞调用(1)

(a) 阻塞的send调用.



阻塞与非阻塞调用(2)

非阻塞的send调用



(b)

问题：直到消息发送出去才能修改其消息缓冲区，否则消息会丢失，发送进程仍然无法连续发送消息。

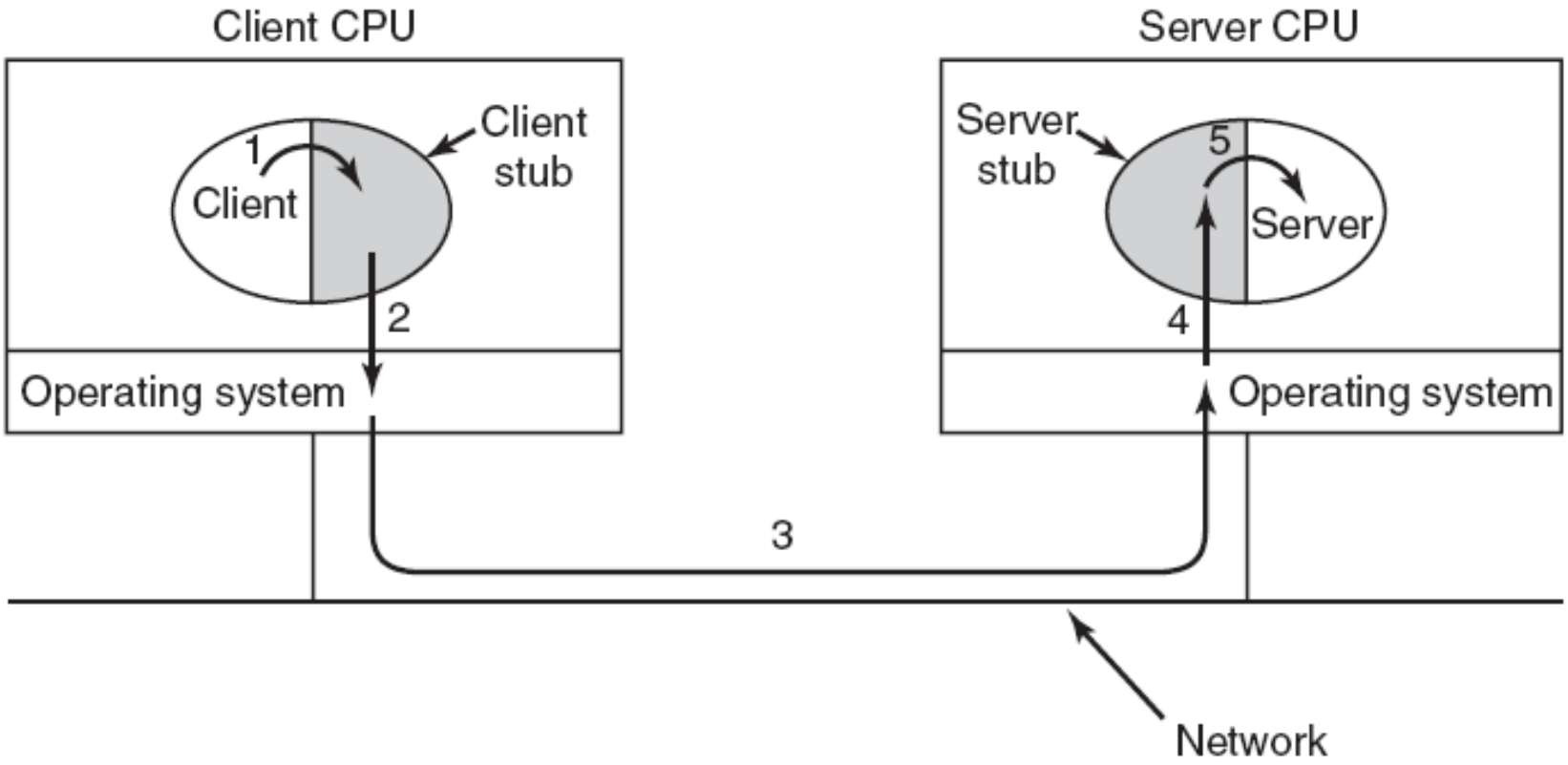
三种方法：复制消息到内核缓冲区、发送结束之后中断发送者、缓冲区写时复制

阻塞与非阻塞调用(3)

因此发送方有四种选择:

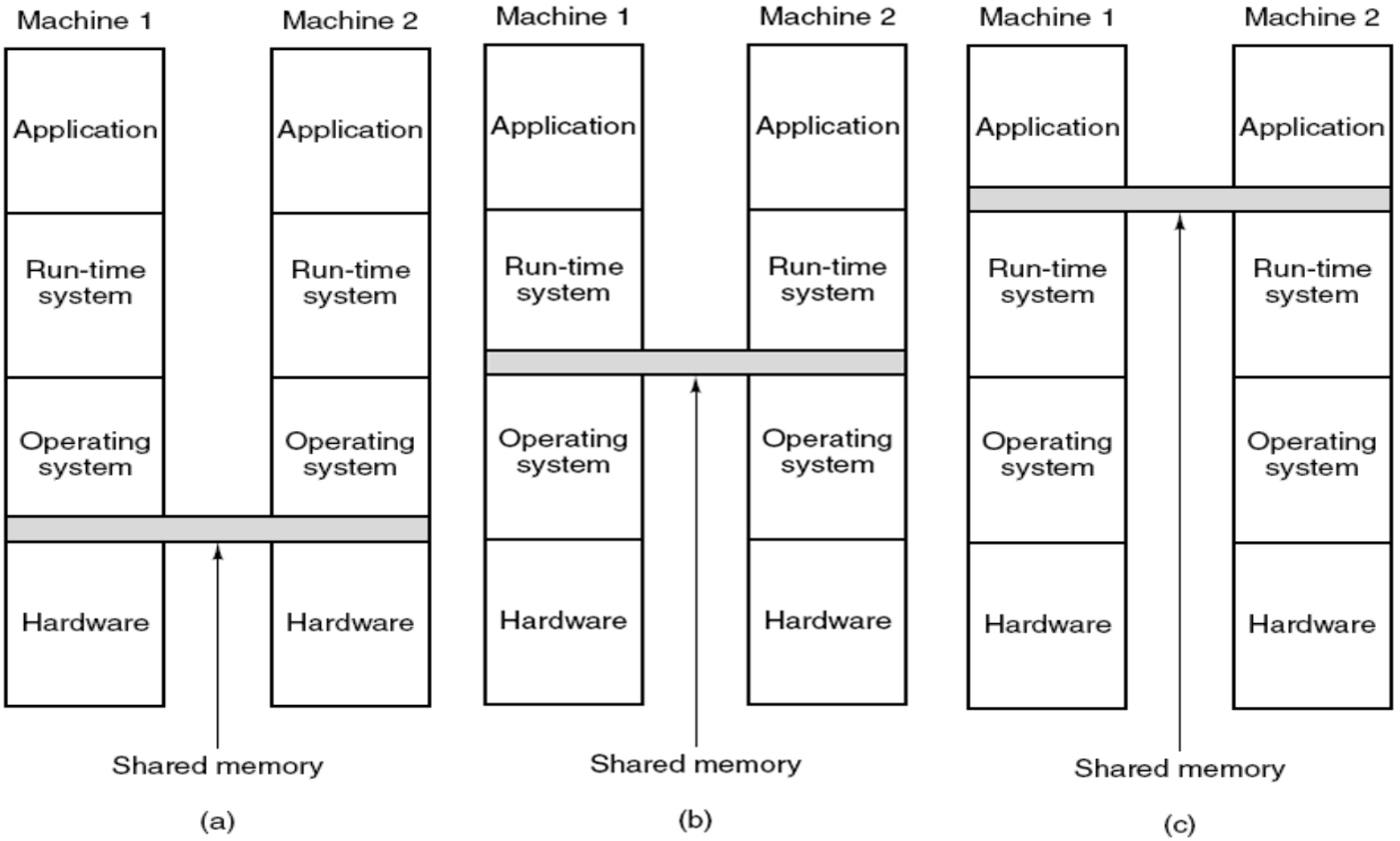
1. Blocking send (CPU idle during message transmission).
2. Nonblocking send with copy (CPU time wasted for the extra copy).
3. Nonblocking send with interrupt (makes programming difficult).
4. Copy on write (extra copy probably needed eventually).

远程过程调用：Remote Procedure Call



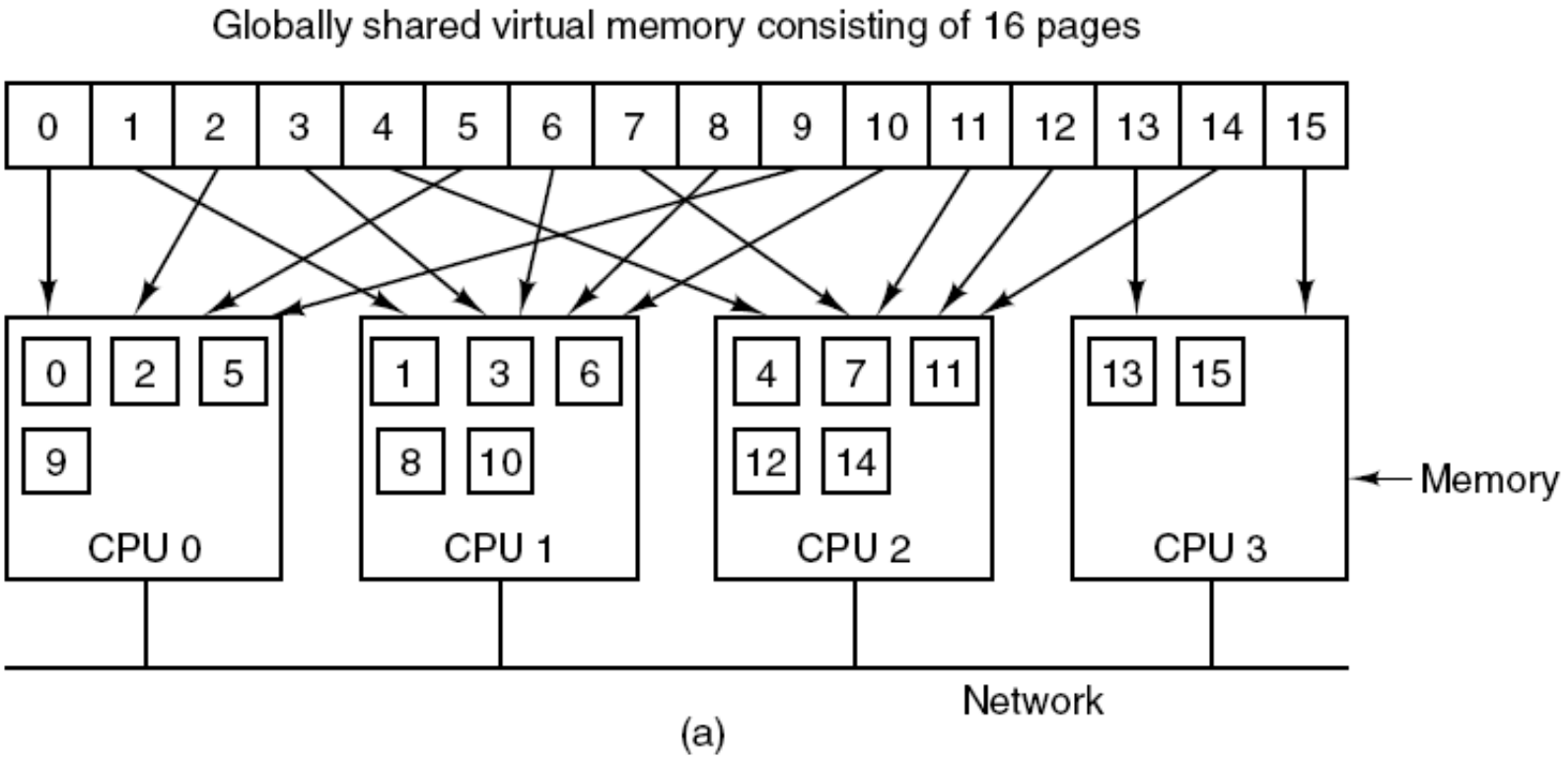
屏蔽I/O层的细节

分布式共享内存(1)



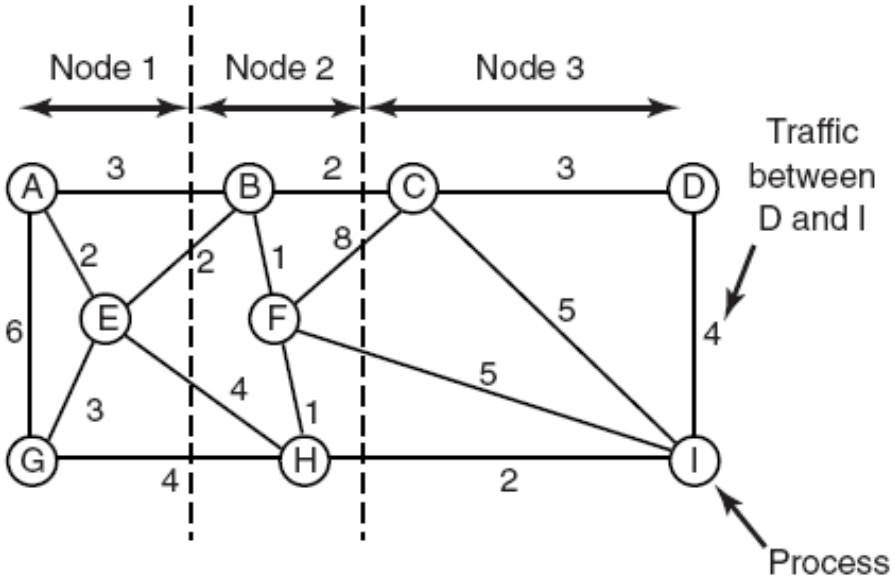
不同层次实现存储器共享

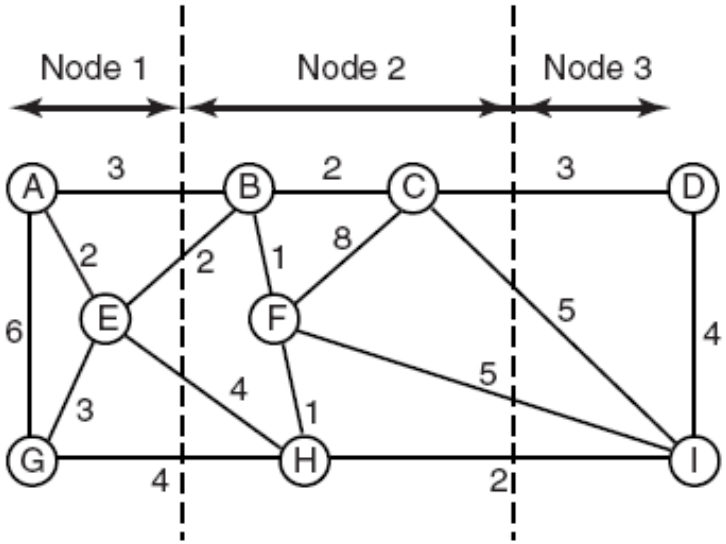
分布式共享内存(2)



(a) Pages of the address space distributed among four machines.

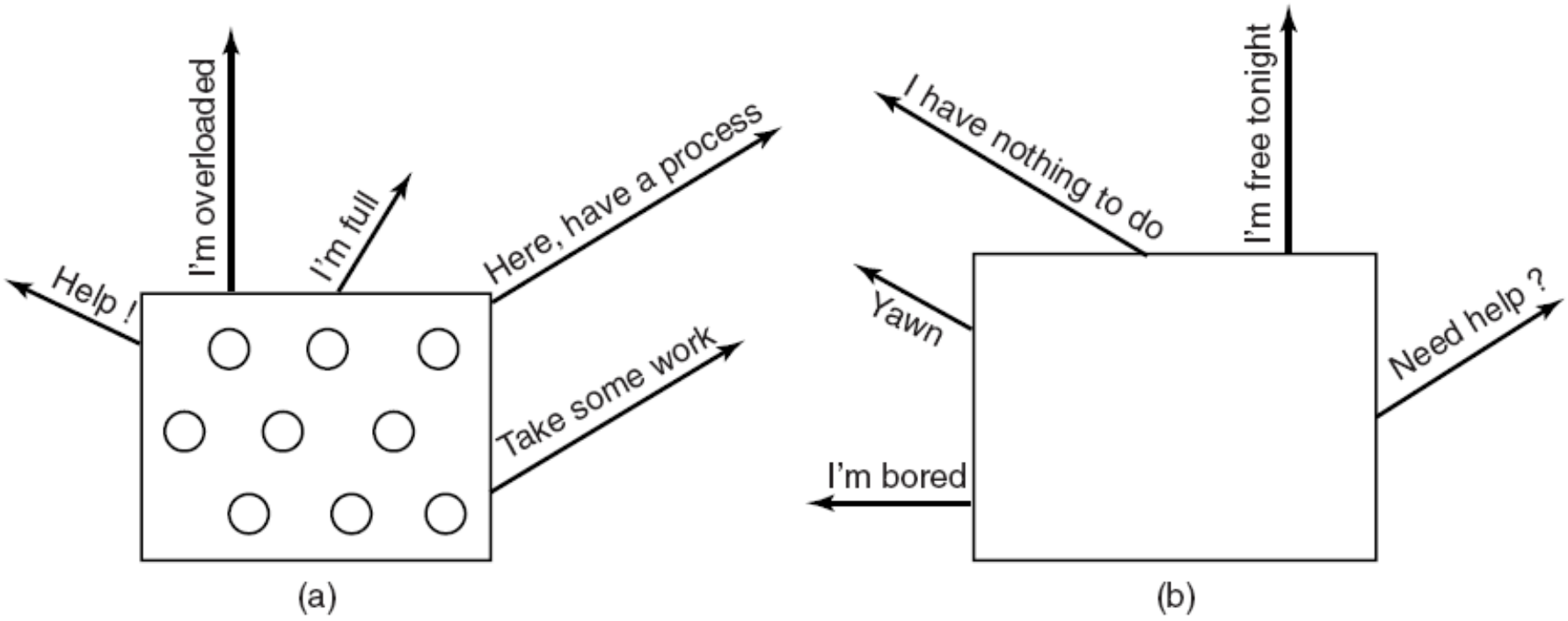
多计算机间的调度算法：基于图论的确定性调度算法





目标：最小网络流

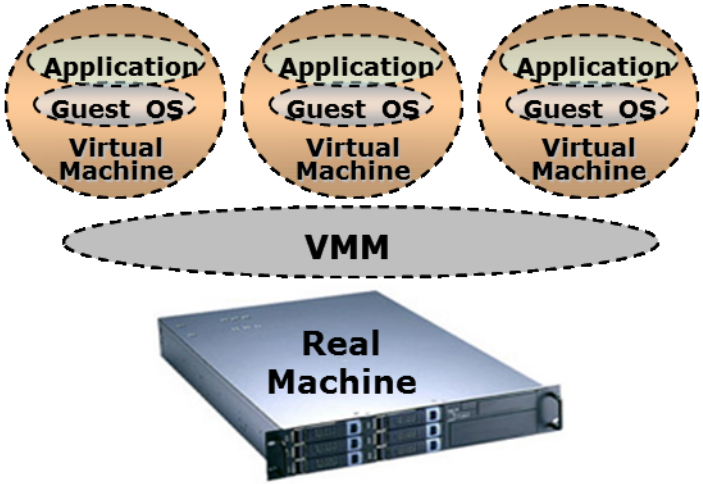
A Sender-Initiated Distributed Heuristic Algorithm



过载的节点主动询问空闲节点

虚拟化技术

- 系统虚拟化：实现操作系统与物理计算机的分离，使得在一台物理计算机上可以同时安装和运行一个或多个虚拟的操作系统
- 虚拟机：使用系统虚拟化技术，运行在一个隔离环境中、具有完整硬件功能的逻辑计算机系统，包括操作系统和其中的应用程序。



最早在IBM 370中提出

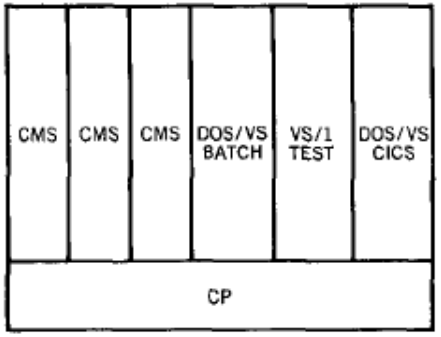
VM/370—a study of multiplicity and usefulness

by L. H. Seawright and R. A. MacKinnon

The productivity of data processing professionals and other professionals can be enhanced through the use of interactive and time-sharing systems. Similarly, system programmers can benefit from the use of system testing tools. A systems solution to both areas can be the virtual machine concept, which provides multiple software replicas of real computing systems on one real processor. Each virtual machine has a full complement of input/output devices and provides functions similar to those of a real machine. One system that implements virtual machines is IBM's Virtual Machine Facility/370 (VM/370).[1]

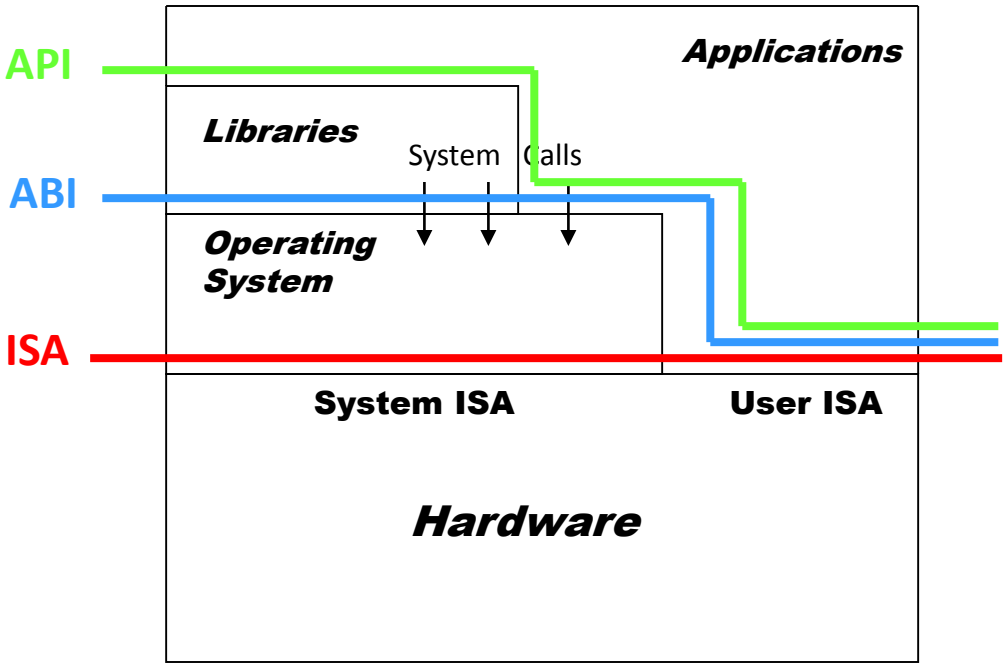
IBM Systems Journal, vol. 18, no. 1, 1979, pp. 4-17.

Figure 1 A VM/370 environment



- **Concurrent execution of multiple production operating systems**
- **Testing and development of experimental systems**
- **Adoption of new systems with continued use of legacy systems**
- **Ability to accommodate applications requiring special-purpose OS**
- **Introduced notions of "handshake" and "virtual-equals-real mode" to allow sharing of resource control information with CP**
- **Leveraged ability to co-design hardware, VMM, and guestOS**

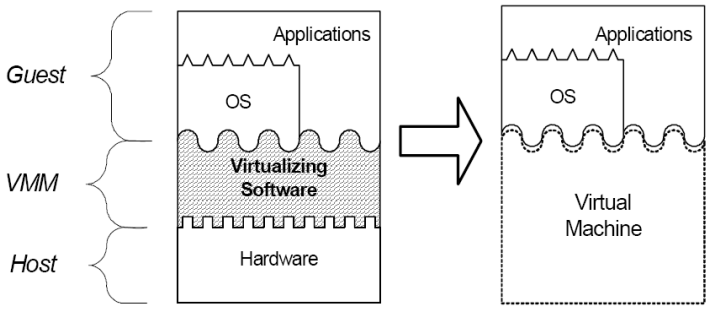
计算机系统结构与接口



- API – application programming interface
- ABI – application binary interface
- ISA – instruction set architecture

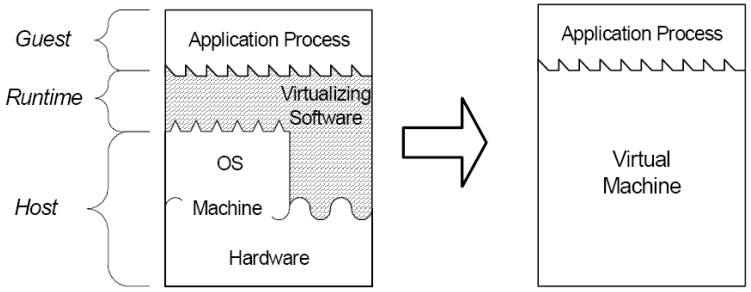
VMM 类型

- 系统级



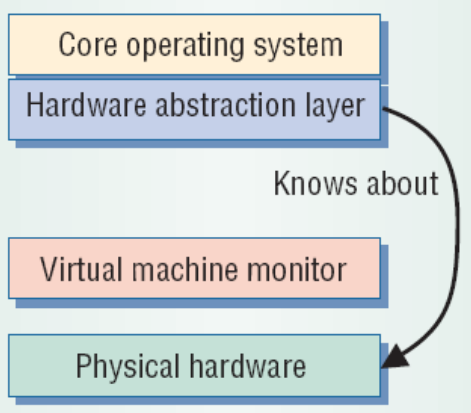
- Provides ABI interface
- Efficient execution
- Can add OS-independent services (e.g., migration, intrustion detection)

- 进程级

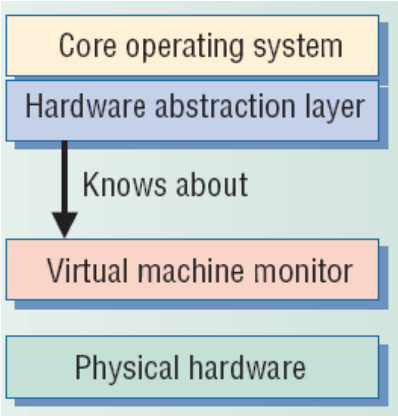


- Provdes API interface
- Easier installation
- Leverage OS services (e.g., device drivers)
- Execution overhead (possibly mitigated by just-in-time compilation)

系统级的虚拟化设计

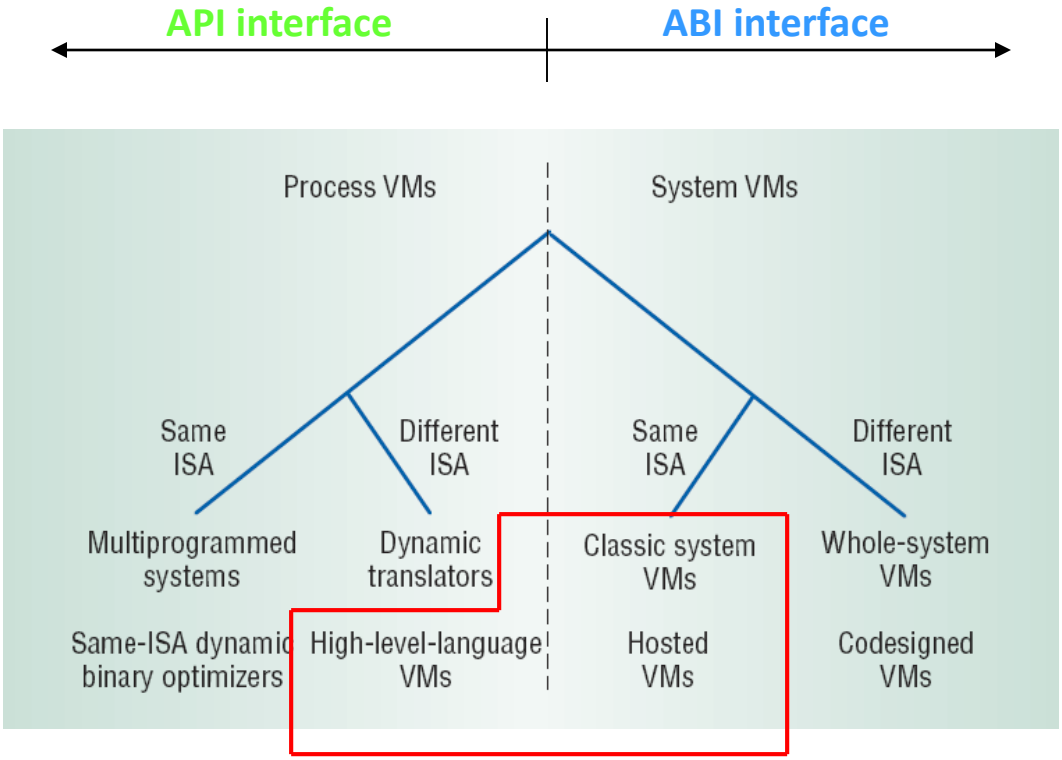


- 完全虚拟化(direct execution)
  - **Exact hardware exposed to OS**
  - **Efficient execution**
  - **OS runs unchanged**
  - **Requires a “virtualizable” architecture**
  - **Example: VMWare**



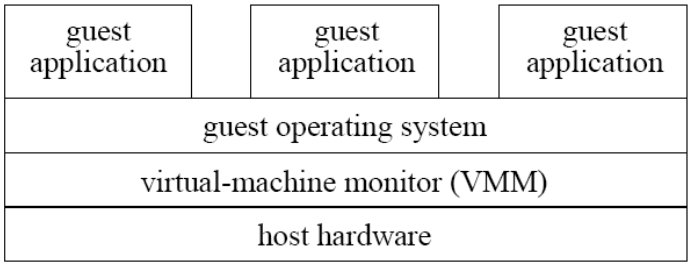
- 半虚拟化Paravirtualization
  - **OS modified to execute under VMM**
  - **Requires porting OS code**
  - **Execution overhead**
  - **Necessary for some (popular) architectures (e.g., x86)**
  - **Examples: Xen, Denali**

Design Space (level vs. ISA)

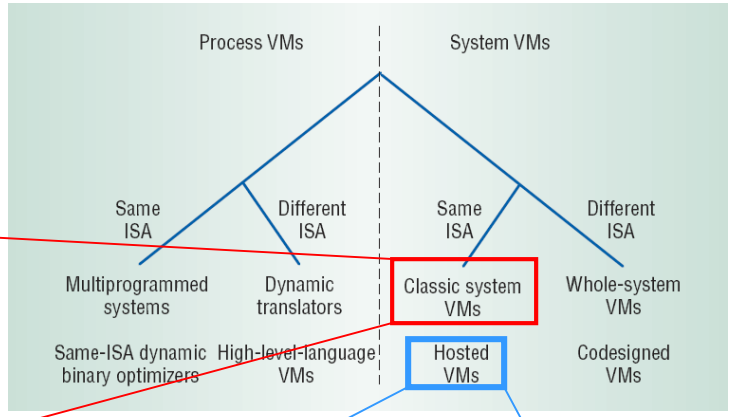


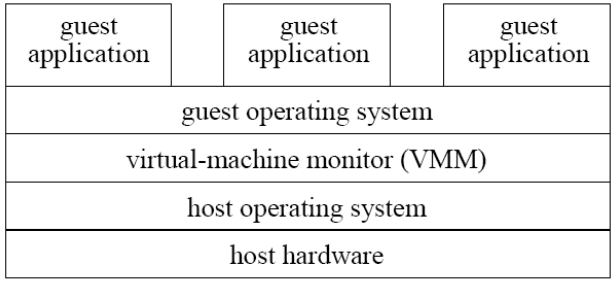
- Variety of techniques and approaches available
- Critical technology space highlighted

系统级VMM



Type 1



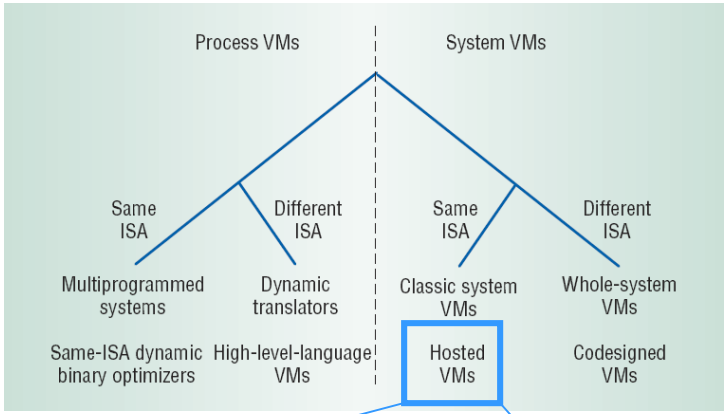


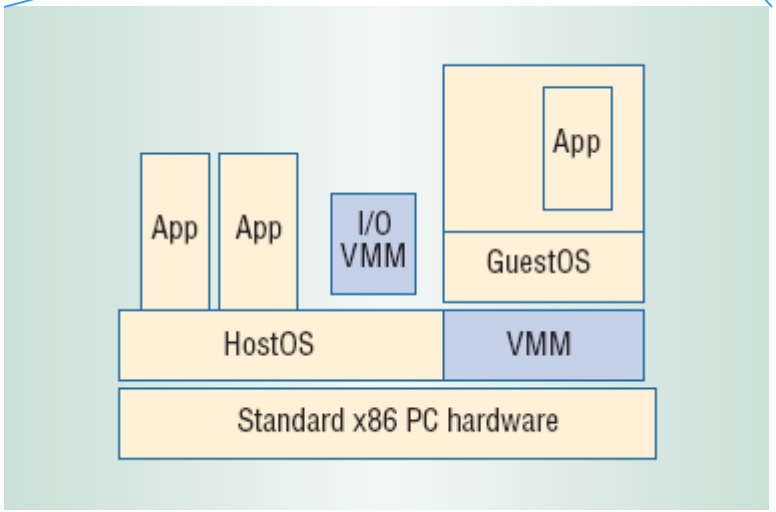
Type 2

- Structure
  - Type 1: runs directly on host hardware
  - Type 2: runs on HostOS
- Primary goals
  - Type 1: High performance
  - Type 2: Ease of construction/installation/acceptability
- Examples
  - Type 1: VMWare ESX Server, Xen, OS/370
  - Type 2: User-mode Linux

Hosted VMMs

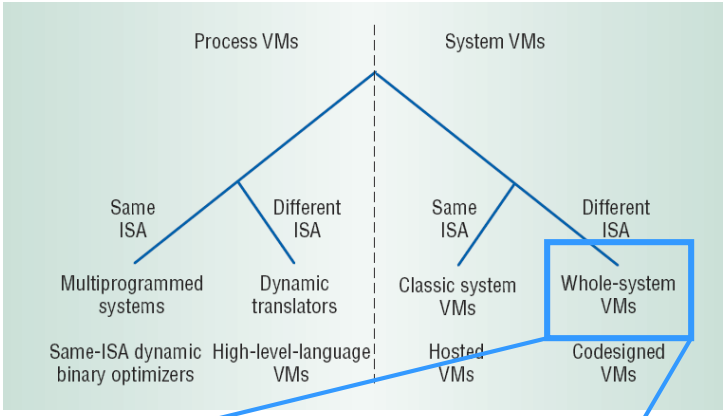
- Structure
  - Hybrid between Type1 and Type2
  - Core VMM executes directly on hardware
  - I/O services provided by code running on HostOS
- Goals
  - Improve performance overall
  - leverages I/O device support on the HostOS
- Disadvantages
  - Incurs overhead on I/O operations
  - Lacks performance isolation and performance guarantees
- Example: VMWare (Workstation)

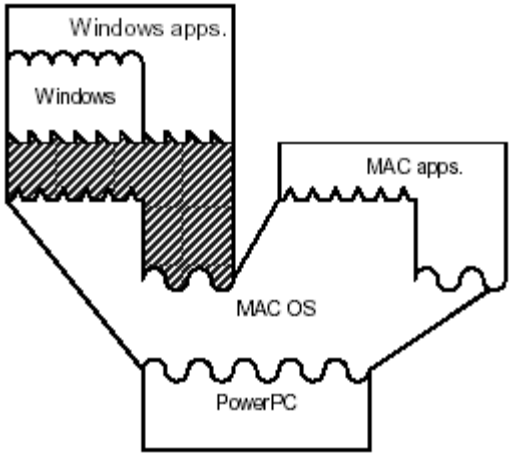




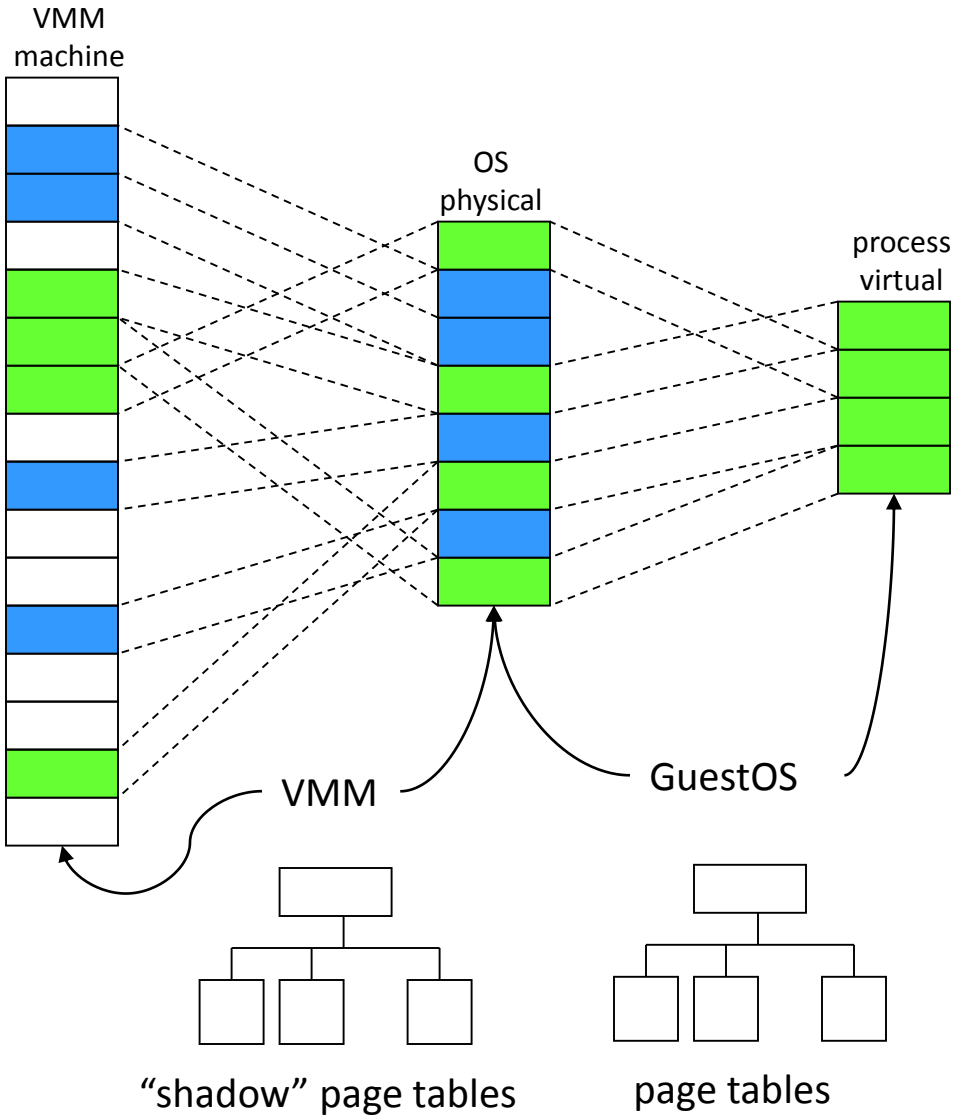
Whole-system VMMs

- **Challenge: GuestOS ISA differs from HostOS ISA**
- **Requires full emulation of GuestOS and its applications**
- **Example: VirtualPC**



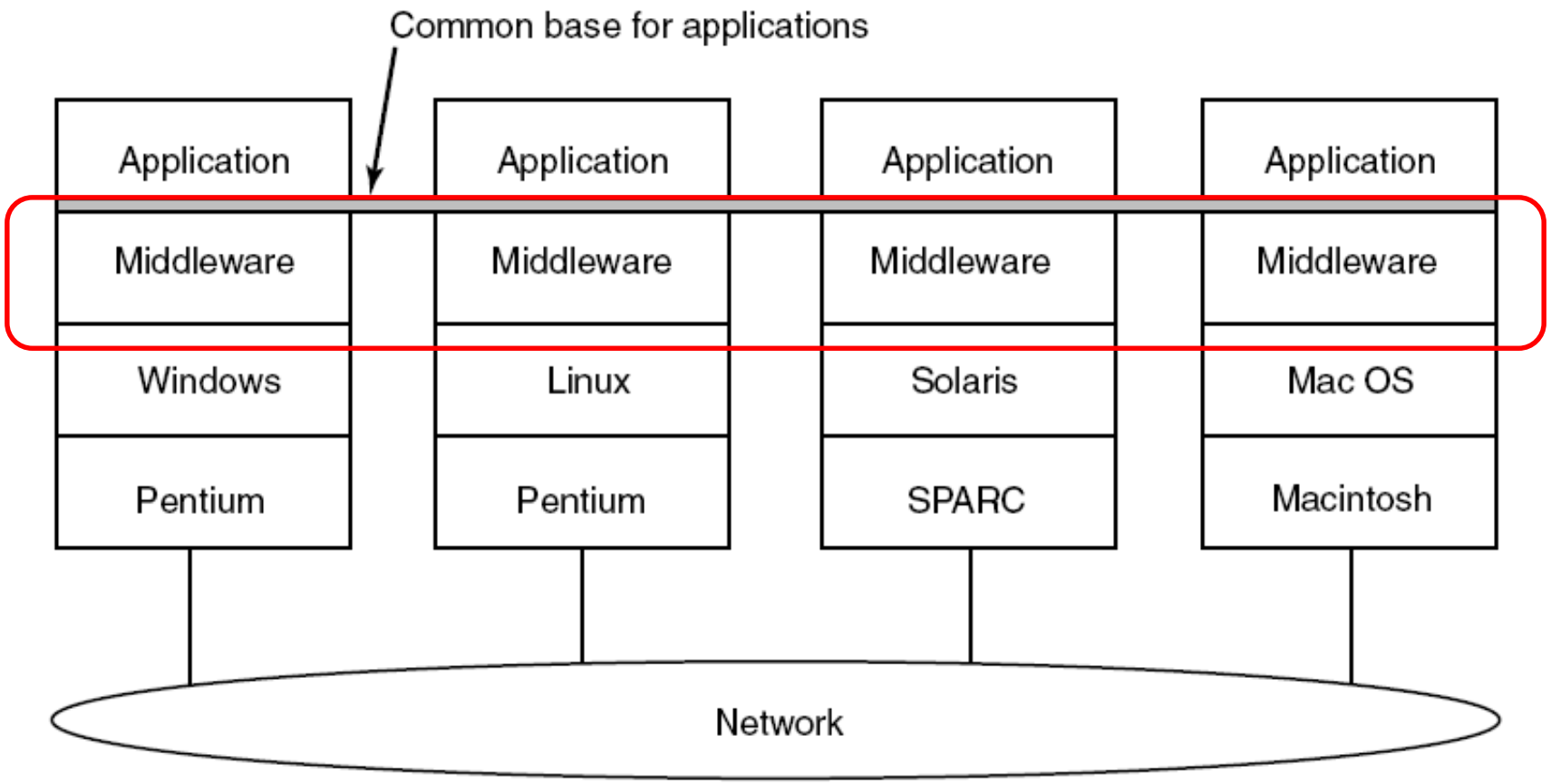


虚拟机环境下的内存管理



- Isolation/protection of Guest OS address spaces
- Efficient MM address translation

分布式系统



小结：多处理机系统

.

Item	Multiprocessor	Multicomputer	Distributed System
Node configuration	CPU	CPU, RAM, net interface	Complete computer
Node peripherals	All shared	Shared exc. maybe disk	Full set per node
Location	Same rack	Same room	Possibly worldwide
Internode communication	Shared RAM	Dedicated interconnect	Traditional network
Operating systems	One, shared	Multiple, same	Possibly all different
File systems	One, shared	One, shared	Each node has own
Administration	One organization	One organization	Many organizations

问题？