高性能计算与云计算

第三讲 并行计算机系统及性能评测

胡金龙 董守斌
{jlhu, sbdong}@scut.edu.cn

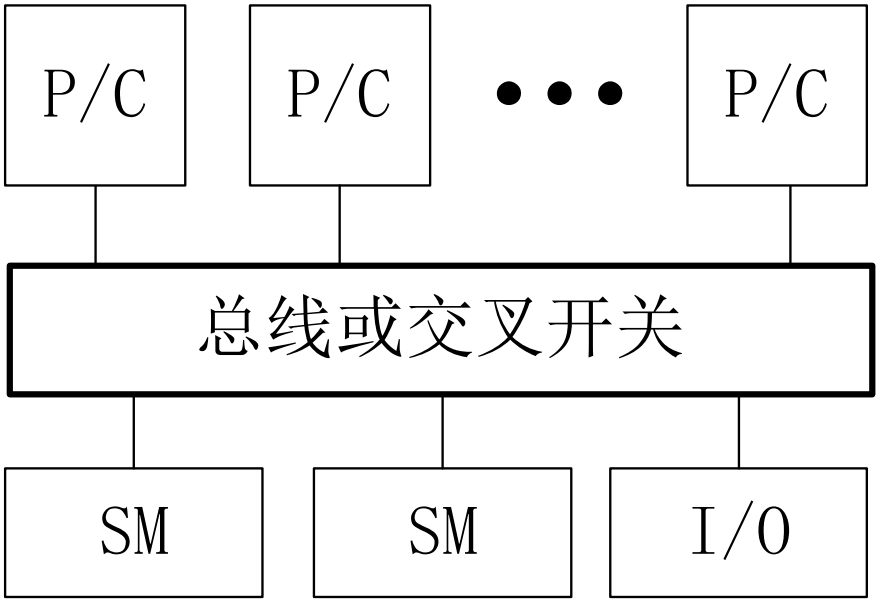
华南理工大学计算机学院
广东省计算机网络重点实验室
Communication & Computer Network Laboratory (CCNL)

主要内容

- 并行计算机系统
  - SMP
  - MPP
  - Cluster
- 并行计算性能评测

对称多处理机SMP

- **SMP（Symmetric Multiprocessing）: 采用商用微处理器，通常有片上和片外Cache，基于总线连接，集中式共享存储，UMA结构**
- **例子：SGI Power Challenge, DEC Alpha Server,曙光（Dawning）1，HP SuperDome, Sun SunFire, IBM Regatta**



商业 SMP系统

System Features	HP 9000 Superdome	Sun Fire 15K	IBM P690 Regatta
CPUs	128	106	32
CPU Types	PA-8800 1.0GHZ	UltraSPARC 3 1.2GHz	POWER4+ 1.90GHz
Partitions	16/128	18	32
Maximum memory per partition	1TB	1/2 TB	1TB
Total hot-swap PCI-X I/O slots	96/192 slots	72 slots	160 slots
Interconnect Network	Crossbar 64GB/s (Peak)	150 MHz Sun Fireplane	Crossbar
I/O bandwidth	32GB/s (Peak)	21.6 GB/s (Sustained)	44GB/s (Aggregated)
Memory bandwidth (Peak)	256GB/s	172.8GB/s	205GB/s

SMP的优缺点

- 优点
  - 对称性
  - 单地址空间，易编程性，动态负载平衡，无需显示数据分配
  - 高速缓存及其一致性，数据局部性，硬件维持一致性
  - 低通信延迟，Load/Store完成
- 问题
  - 可靠性差：BUS，OS，SM
  - 可观的通信延迟（相对于CPU）
  - 带宽增加缓慢（存储器总线带宽每3年只增加2倍）
  - 不可扩放性（总线是不可扩放的）---〉改用交叉开关，改用CC-NUMA或集群

高速缓存一致性非均匀存储访问（CC-NUMA）

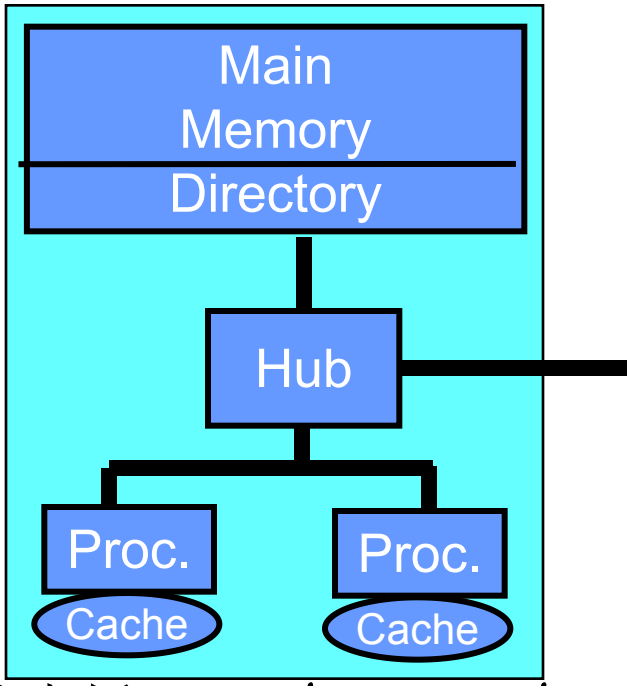
- CC: Cache Coherent

  NUMA: Non-Uniform Memory Access
  - Memory是物理分布的
  - Memory是全局可存取的
  - 本地存储快于远程存储的（Non-Uniform Memory Access = NUMA）
  - 不同节点的本地存储不互相影响
- 例子：
  - Stanford Dash：第一个使用基于目录的缓存一致性解决方案
  - SGI Origin 2000（Silicon Graphics Inc.）：可支持多至1024个处理器，曾经占据95% 的 cc-NUMA型机器的市场

实例分析：SGI Origin 2000

- O2K的体系结构
  - CC-NUMA
  - 节点板：双CPU
  - 互联：超立方（Hypercube）
  - 缓存一致和虚拟内存
    - ✓分布式存储构成单一的（逻辑的）存储空间，任何处理器都可以访问
  - 可扩展型：时延和带宽

Origin 2000 节点板



节点板（Node Board）

- MIPS R12000 CPU
  - 64-bit RISC design, 400MHz
  - 5 fully-pipelined execution units
  - 8MB L2 cache
  - 32KB 2-way set-associative instruction and data caches

路由和互联结构

- 6路的非阻塞交叉开关（6-way non-blocking crossbar）：9.3 GBytes/sec
  - 每个端口1.56 Gbyte/sec （双工）
- 虫蚀选路（wormhole routing）技术
- 源同步驱动器SSD （Source Synchronous Driver）和源同步接收器SSR（Source Synchronous Receiver）
  - 将390MHz的外部信号转换为核心频率97.5MHz
  - 64位/16位转换
- 链路级协议LLP（Link-Level Protocal）
  - 确保消息的可靠传输

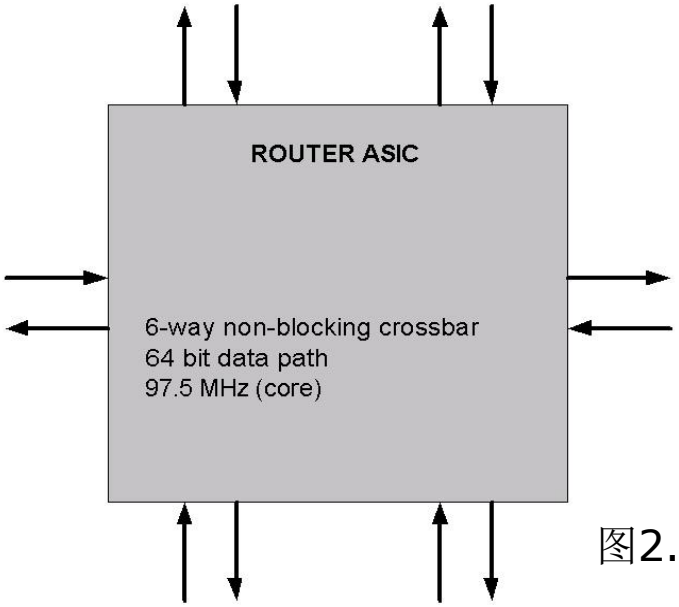
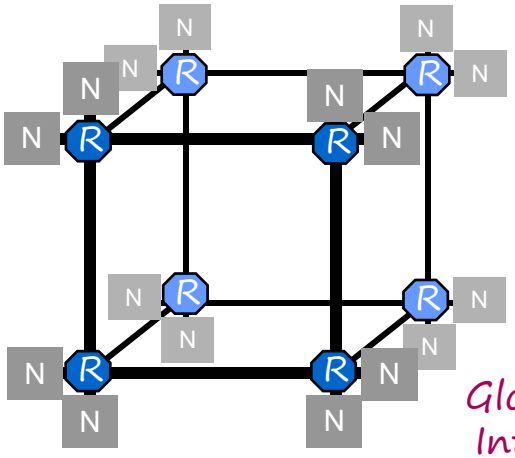


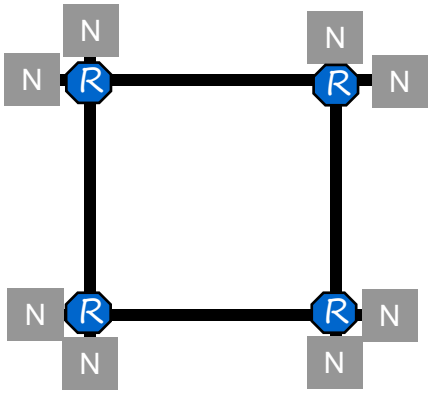
图2.5

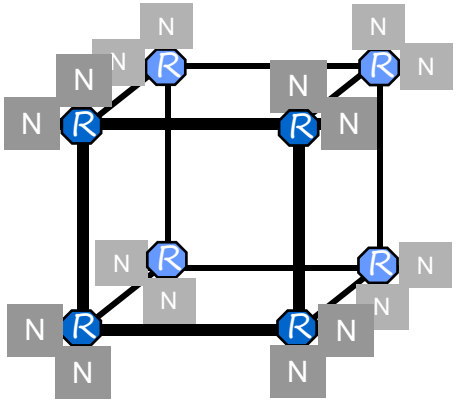


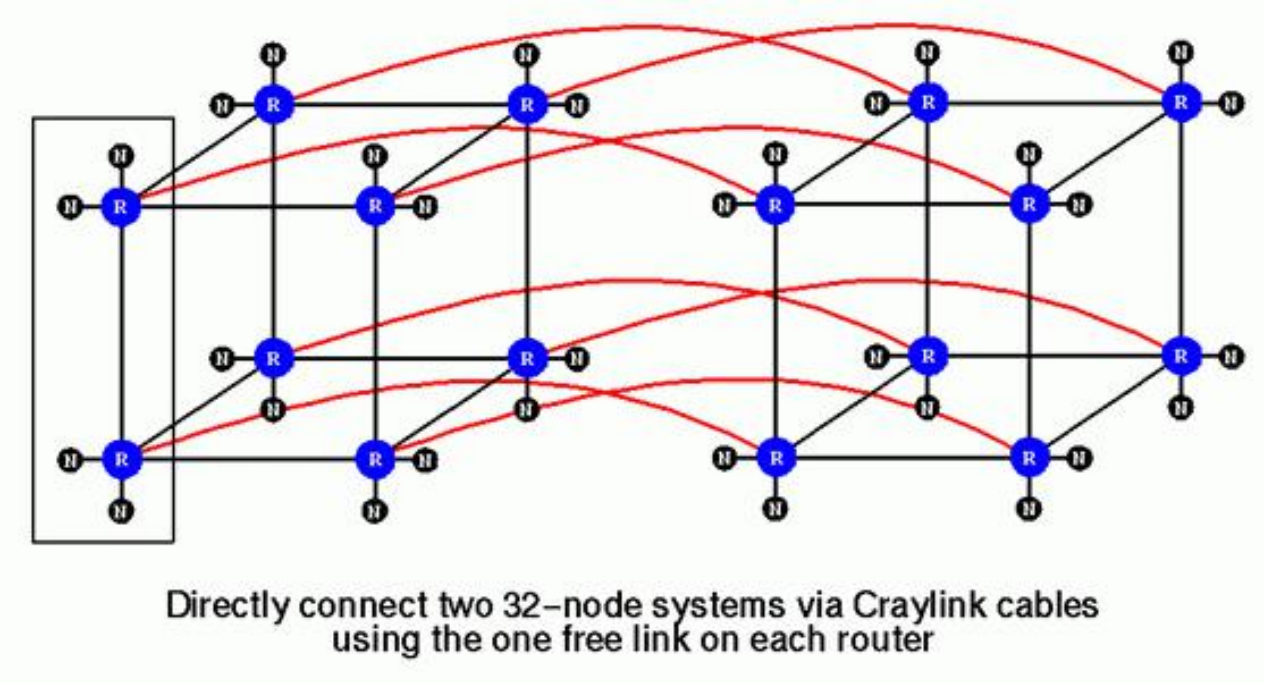
Global Switch Interconnect

互联拓扑

- 互联拓扑：“fat hypercube”(胖超立方)
- 每个路由器（router）连接一对节点
- 路由器通过超立方连接



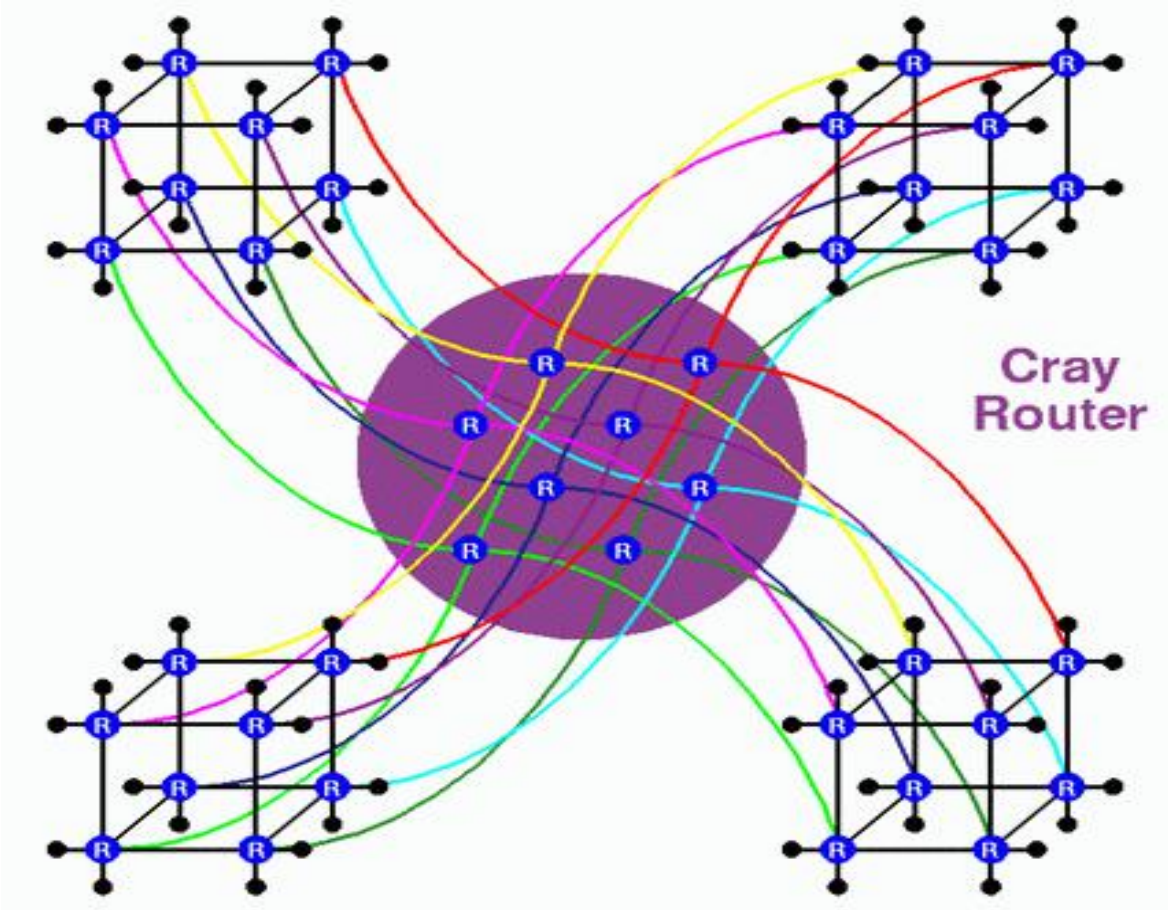




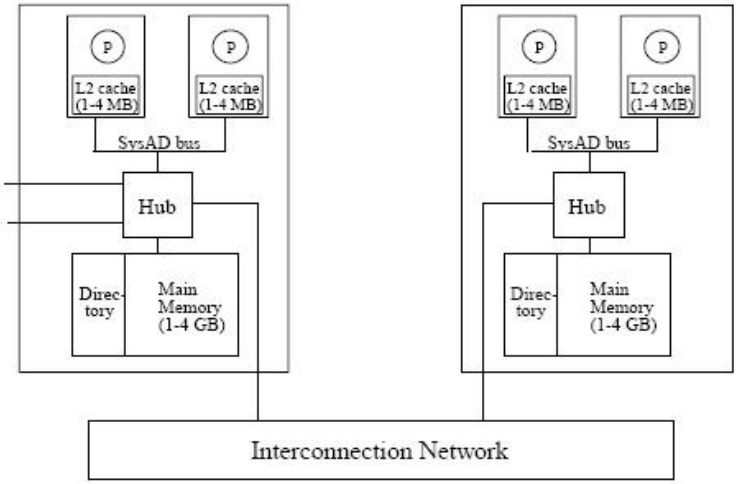
Directly connect two 32-node systems via Craylink cables using the one free link on each router

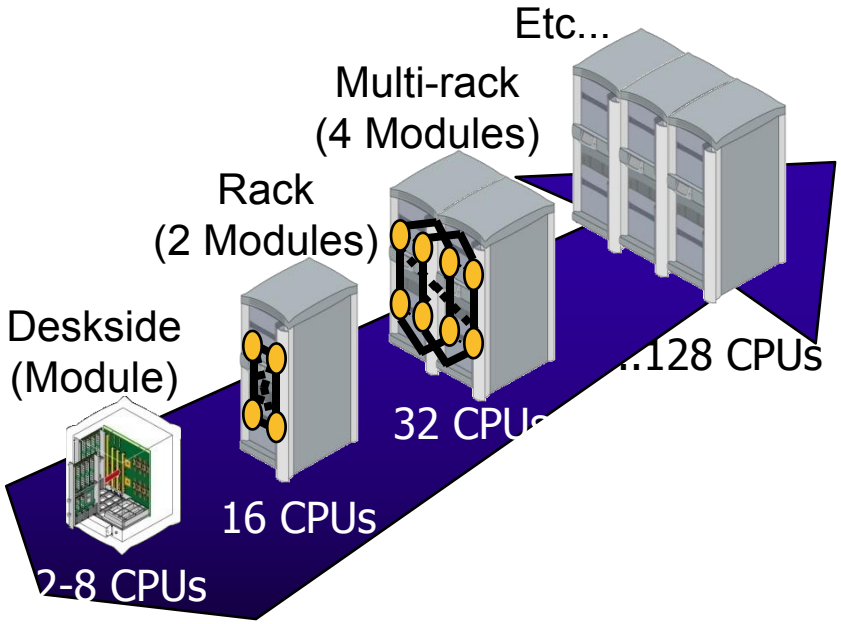
元路由器

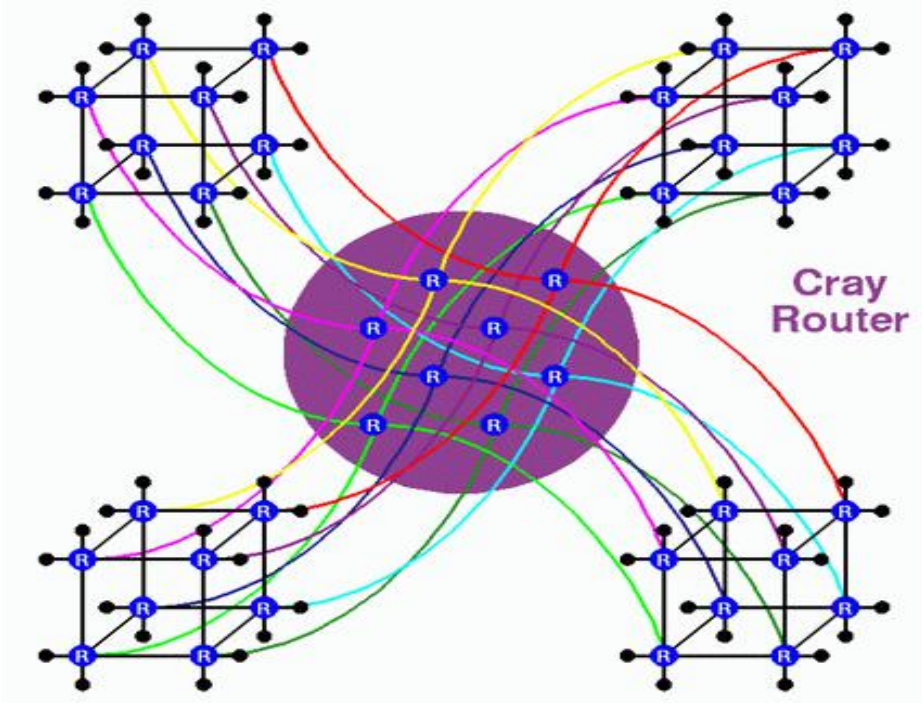
- 64个节点



Origin 2000 – 128 Processors







O2K存储时延

数据位置	读时延（时钟数）
寄存器	0
L1 Cache（片上）	1-3
L2 Cache（片外）	10
本地存储器	61
远程存储器 （1个路由器以远）	117
远程存储器 （2个路由器以远）	137
远程存储器 （3个路由器以远）	157
远程存储器	97+20*(number of router hops)

表2.3

带宽 (GB/s)和延迟 (ns)

系统配置 I/O:Node:CPU	总计峰值存储器带宽	总计峰值I/O带宽	互联网络对剖带宽	Router Hops (avg/max)	Memory Latency (avg)
1:1:2	0.78	1.56	-	-	343
2:2:4	1.56	3.12	-	-	441
2:4:8	3.12	6.24	1.56	0.75/1	623
4:8:16	6.24	12.48	3.12	1.63/2	691
8:16:32	12.48	24.96	6.24	2.19/3	674
16:32:64	24.96	51.20	12.48	2.97/5	851
32:64:128	49.92	99.84	24.96	3.98/6	959

表3.7

互联网络对剖带宽/CPU数 = ?

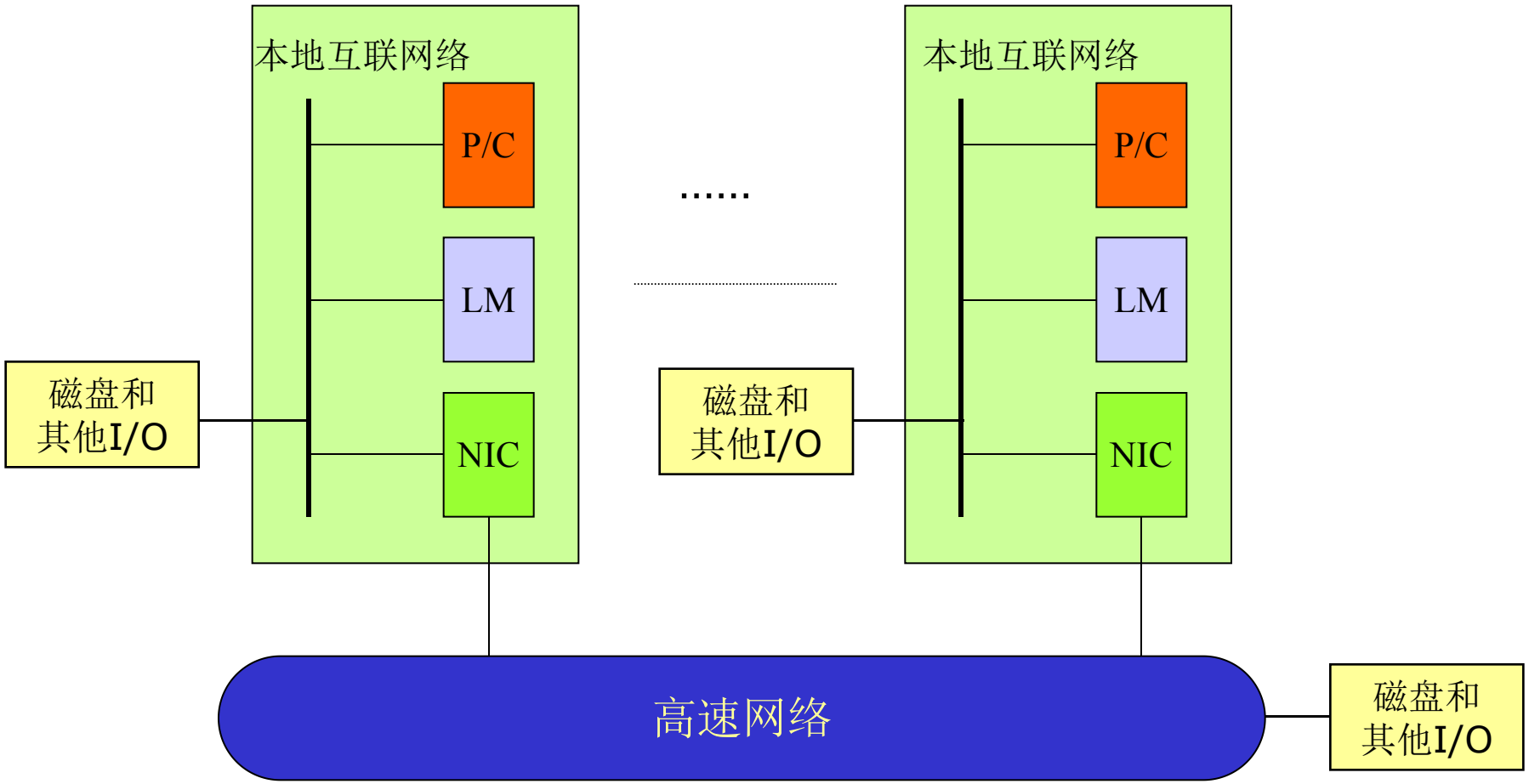
主要内容

- 并行计算机系统
  - SMP
  - MPP
  - Cluster
- 并行计算性能评测

大规模并行处理机MPP概述

- 大规模并行处理机MPP（Massively Parallel Processor） 通常是指具有下列特点的大规模的计算机系统：
  - 节点中使用商品化微处理器，且每个节点有一个或多个微处理器
  - 节点内使用物理上分布的存储器
  - 具有高通信带宽和低延迟的互连网络，节点间紧耦合
  - 能扩展成具有成百上千个处理器

MPP的结构图


本地互联网络
P/C
LM
NIC
磁盘和
其他I/O
.....
本地互联网络
P/C
LM
NIC
磁盘和
其他I/O
高速网络
磁盘和
其他I/O

ASCI可扩放设计策略

- 加速战略计划创新ASCI（Accelerated Strategic Computing Initiative）
  - 1994年DOE
  - 1996年 1Tflop/s系统，2000年 10至30Tflop/s系统，2004年 100Tflop/s系统，且这些系统应该成本相近
  - 不仅瞄准峰值速度，而且总的系统持续的应用性能要10^5倍于1994年
- 平衡的可扩放设计
  - 着重用于科学计算应用的高端平台，而非大批量市场平台和市场热点应用；
  - 使用尽可能多的商品化市售（COTS）硬件和软件部件，着重开发主流计算机公司未有效提供的关键技术；
  - 使用大规模并行体系结构，着重于缩放和集成技术，将数千个COTS节点纳入一个有单一系统映象的高效平台

ASCI平台


SCI Blue Mountain

四个ASCI比较

特性	Option Red	Option Blue		Option White
		Blue Pacific	Blue Mountain	
制造商	Intel	IBM	SGI	IBM
安装场所	Sandia	Livermore	Los Alamos	Livermore
完成日期	1997年6月	1998年12月	1998年12月	2000年12月
成本(百万美元)	55	94	<110	85
所选用处理器	Pentium Pro 200MHz 200Mflop/s	PowerPC 604 332MHz 664Mflop/s	MIPS 10000 250MHz 500Mflop/s	POWER3 311MHz 1244Mflop/s
系统体系结构	NORMA-MPP	SMP机群 4CPU/节点 1464节点	CC-NUMA机群 128CPU/节点 48节点	SMP机群 16CPU/节点 512节点
节点内连接	总线	交叉开关	胖超立方体	交叉开关
节点间连接	分离2D网孔	Omega开关	千兆位开关	Omega开关
处理器数量	9216	5856	6144	8192
峰值速度	1.8Tflop/s	3.888Tflop/s	3.072Tflop/s	10.2Tflop/s
主存容量	594GB	2.5TB	1.5TB	4TB
磁盘容量	1TB	75TB	75TB	150TB

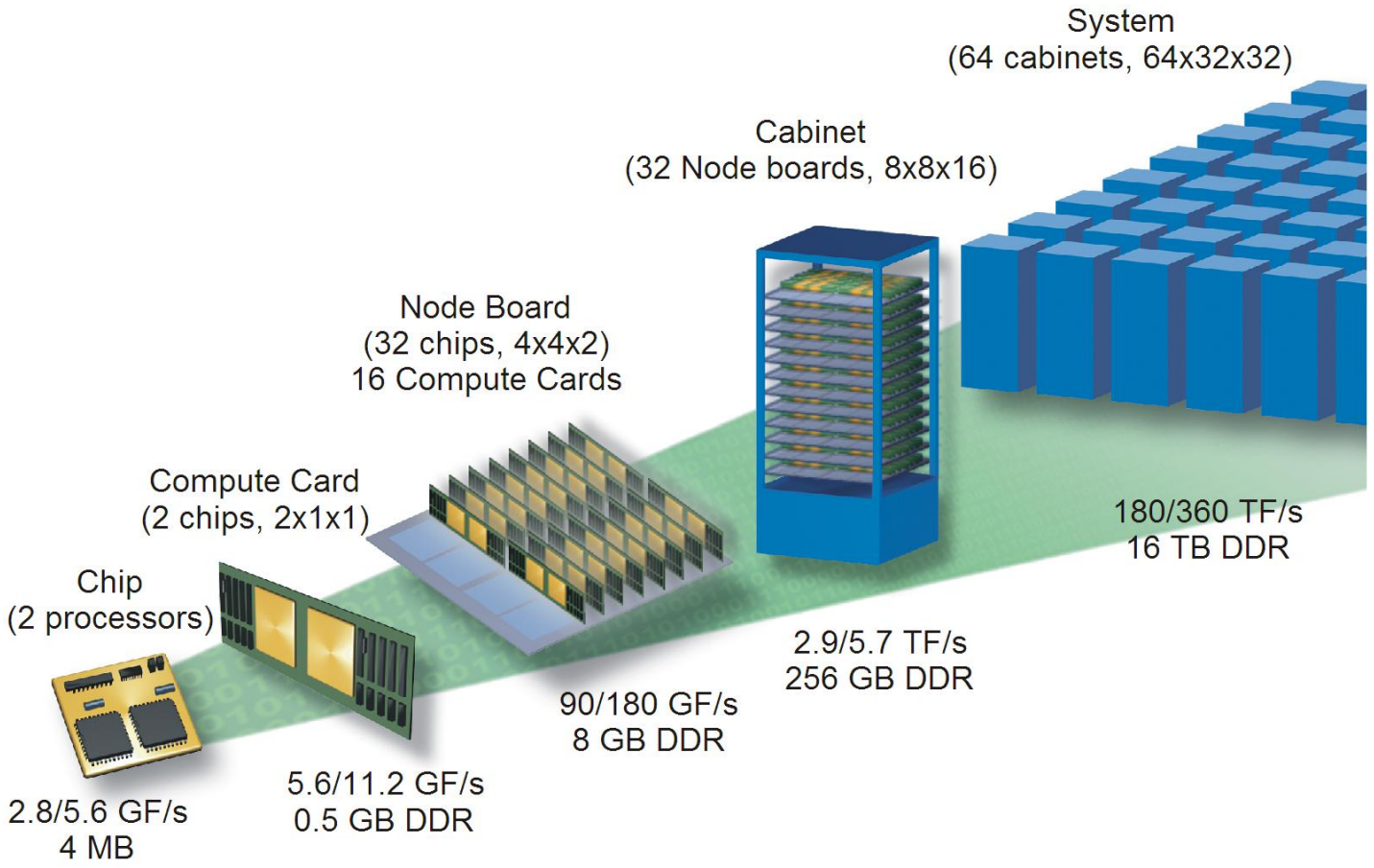
实例分析：蓝色基因（Blue Gene）

- IBM BG/L是IBM Blue Gene系列的第一台机器。其最初的设计目标是满足蛋白质折叠的需求，以及更广泛的商业应用
- IBM BG/L的体系结构特性使得它具备了功耗低、造价低、体积小的特点，同时性能也达到了每秒百万亿次的级别
- 2007年全世界已经有28台BG系统，2017年TOP500中有16台BG系统
- 安装在美国国家实验室LLNL的最大的BG/L （安装时间：2004年第二季度到2005年第三季度）
  - 65,000+ 计算节点 (131,000+ 个处理器)，放在 64个机柜中
  - 每个节点有2个低功耗的700-MHz PowerPC处理器
  - 互联：三维环网（3D torus） +树（ tree）
  - 峰值： 367 Tflop/s； Linpack测试： 280 Tflop/s（2007.11世界第一）

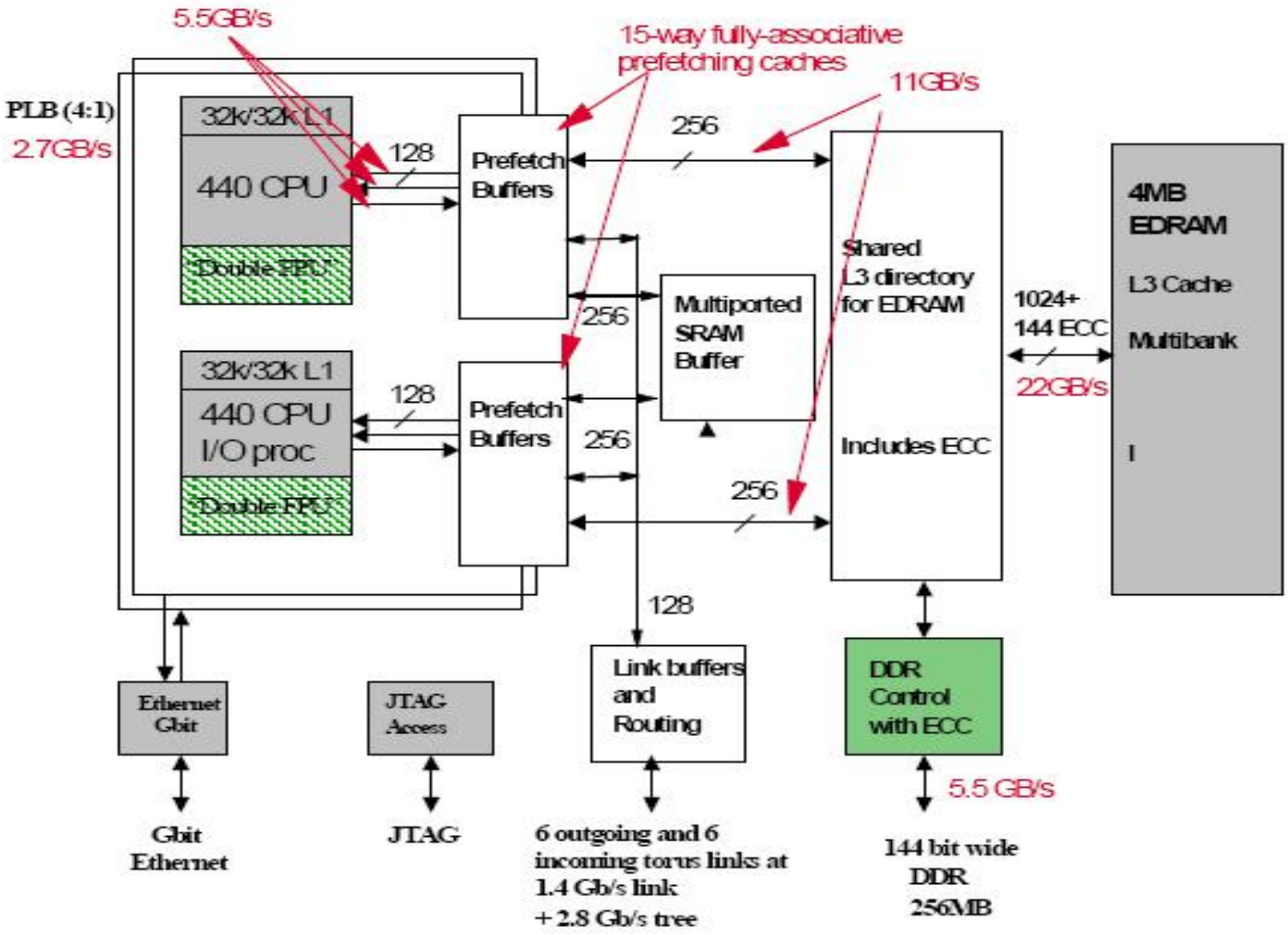
TOP 500中的Blue Gene（2019.6）

Name	Site	Cores	Rmax (TFLOPS)	Rpeak (TFLOPS)
Sequoia	#13: DOE/NNSA/LLNL, USA, 2011	1572864	17173.224	20132.659
Mira	#24: DOE/SC/Argonne National Laboratory, USA, 2012	786432	8586.612	10066.33
Vulcan	#44: DOE/NNSA/LLNL, USA, 2012	393216	4293.306	5033.165
Blue Joule	#288: Science and Technology Facilities Council - Daresbury Laboratory United Kingdom	131072	1431.1	1677.7

从芯片到机架


System
(64 cabinets, 64x32x32)
Cabinet
(32 Node boards, 8x8x16)
Node Board
(32 chips, 4x4x2)
16 Compute Cards
Compute Card
(2 chips, 2x1x1)
Chip
(2 processors)
2.8/5.6 GF/s
4 MB
5.6/11.2 GF/s
0.5 GB DDR
90/180 GF/s
8 GB DDR
2.9/5.7 TF/s
256 GB DDR
180/360 TF/s
16 TB DDR

BG/L处理器芯片
(System-on-a-chip: SoC)

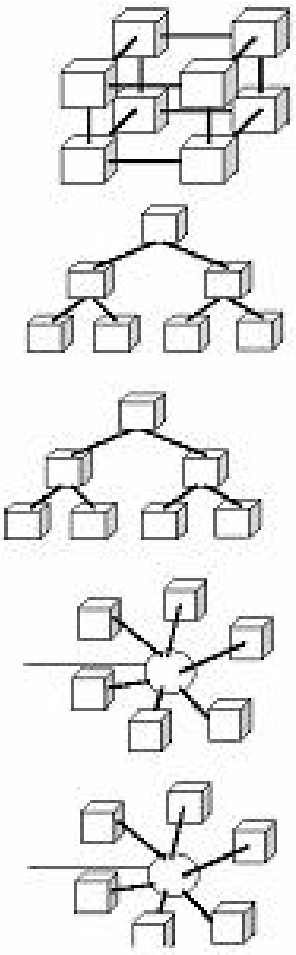


BG/L处理器芯片

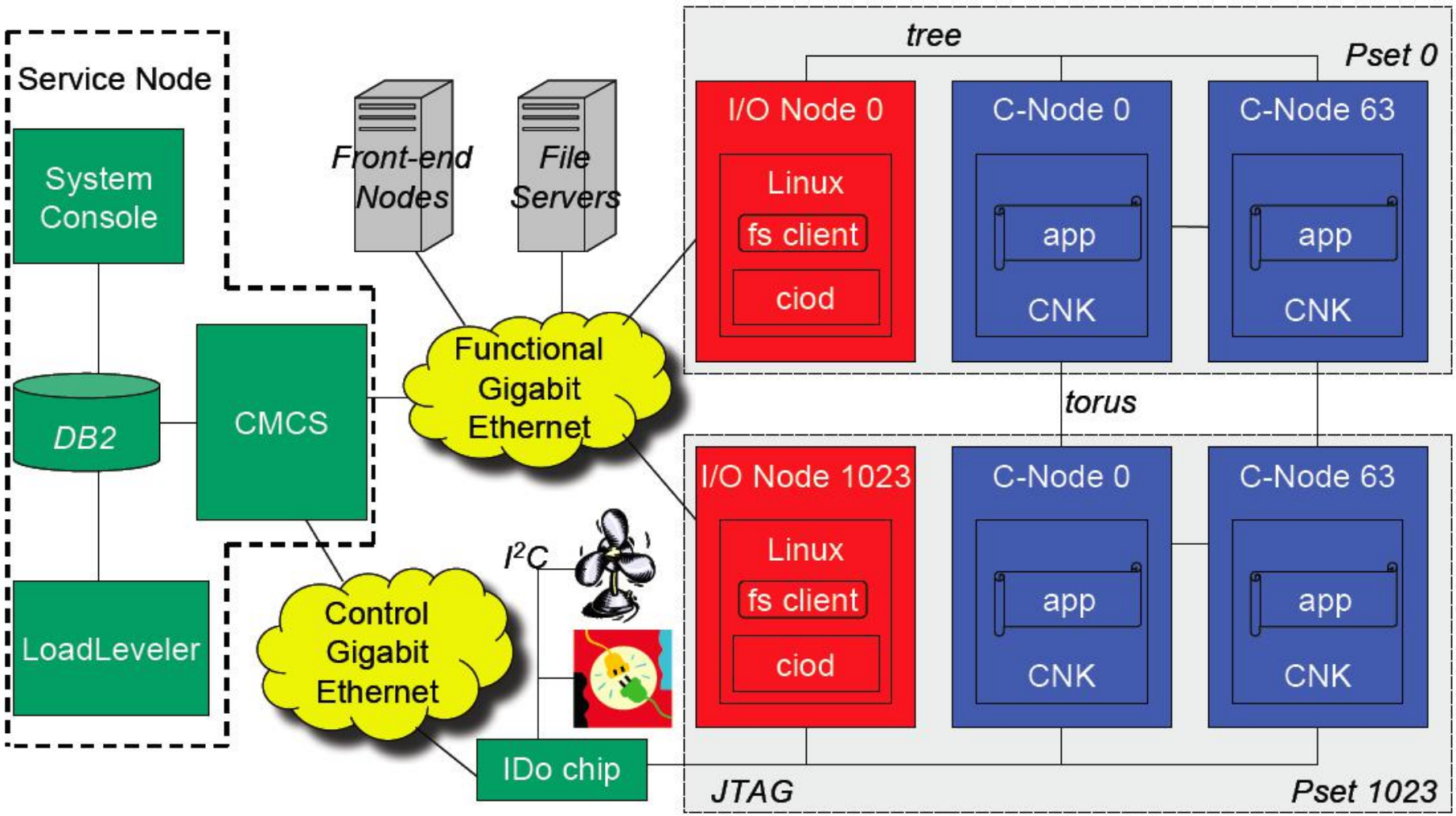
- SoC芯片，每个芯片里面集成了多处理器、内存、通信逻辑。就通过复制这样一个个芯片来搭建整个系统。
- 2个700-MHz PowerPC 440 处理器
  - 各有两个浮点计算单元（floating-point units）
  - 各有32-kB L1数据缓存（ data caches）
  - 4 flops/proc-clock peak (=2.8 Gflop/s per processor)
  - 2 8-B loads or stores / proc-clock peak in L1 (=11.2 GB/s per processor)
- 共享的 2-kB L2 cache
- 共享的 4-MB L3 cache
- 片外（ off-chip ）存储控制器： 512 MB 或1 GB 的共享存储
  - 每个节点的峰值存储带宽为5.6 GB/s

5个网络控制器

- 三维环网络（3D Torus）：可以实现任意两个计算结点之间的点对点传输。在每个三维环的单个链路方向上的硬件带宽是175MB每秒。
- 两个全局的树型网络（Tree）
  - 一个用于实现全局广播操作
  - 另一个用于实现全局的中断响应
- 两个千兆以太网（Gigabit Ethernet）
  - 一个用于实现计算节点与文件系统以及主机之间的通信
  - JTAG（Joint Test Action Group）：用于全局的控制和监视



集成的BG系统和软件


Service Node
System Console
DB2
CMCS
LoadLeveler
Front-end Nodes
File Servers
Functional Gigabit Ethernet
Control Gigabit Ethernet
I2C
IDo chip
tree
Pset 0
I/O Node 0
Linux
fs client
ciod
C-Node 0
app
CNK
C-Node 63
app
CNK
torus
I/O Node 1023
Linux
fs client
ciod
C-Node 0
app
CNK
C-Node 63
app
CNK
JTAG
Pset 1023

BG操作系统和功能

- 计算节点：运行CNK（Compute Node Kernel）
  - 每个节点一次运行一个任务
  - CNK只占用很少的内存
- I/O节点（I/O nodes）：运行嵌入式Linux
  - 运行后台程序CIOD来管理计算节点
  - 进行文件I/O
  - 运行并行文件系统（GPFS）
- 前端节点（Front-end nodes）：运行SuSE Linux
  - 支持用户登陆
  - 运行交叉编译和连接
  - 提交作业和管理作业
- 服务节点（Service node）：运行SuSE Linux
  - 使用DB2来管理4个系统数据可
  - 运行系统管理软件

MPP小结

- 八十年代后期及九十年代中前期迅速发展
  - 主要被用于科学计算
- 九十年代后期，随着一些专门生产并行机的公司的倒闭或被兼并，MPP系统慢慢从主流的并行处理市场退出
  - 由于消息传递系统相对共享存储系统比较容易实现，它仍成为实现超大规模并行处理的重要手段，不过由于价格和应用领域的原因，基于消息传递的MPP系统的研制逐渐成为了政府和大公司的行为
  - 新涌现的高性能计算系统绝大多数都将是由可扩放的高速互连网络连接的基于商用微处理器的对称多处理机（SMP） 集群

Top500中的MPP（2017.6）

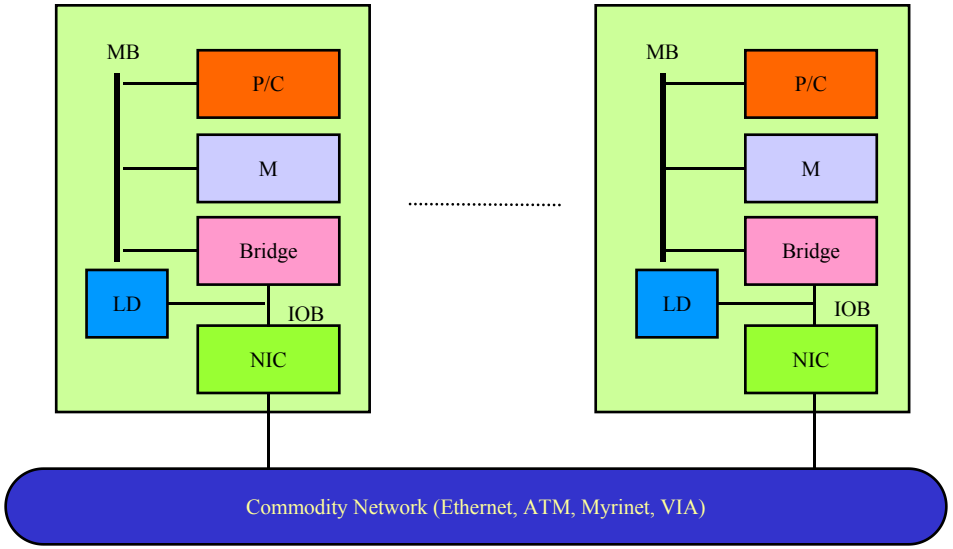
Name/Country	Computer	Interconnect	Cores	Rmax (GFLOPS)	Rpeak (GFLOPS)
#1: Sunway TaihuLight, China,2016	Sunway MPP, Sunway SW26010 260C 1.45GHz, Sunway	Custom Interconnect	10649600	93014.594	125435.904
#3: Piz Daint, Switzerland, 2017	Cray XC50, Xeon E5-2690v3 12C 2.6GHz, Aries interconnect , NVIDIA Tesla P100	Custom Interconnect	361760	361760	25326.264
#4: Titan, USA, 2012	Cray XK7, Opteron 6274 16C 2.200GHz, Cray Gemini interconnect, NVIDIA K20x	Custom Interconnect	560640	17590	27112.55
#5: Sequoia, USA, 2011	BlueGene/Q, Power BQC 16C 1.60 GHz, Custom	Custom Interconnect	1572864	17173.224	20132.659

主要内容

- 高性能计算机系统
  - SMP
  - MPP
  - Cluster
- 并行计算性能评测

工作站集群COW

- 分布式存储，MIMD，工作站+商用互连网络，每个节点是一个完整的计算机，有自己的磁盘和操作系统，而MPP中只有微内核
- 优点：
  - 投资风险小
  - 系统结构灵活
  - 性能/价格比高
  - 能充分利用分散的计算资源
  - 可扩放性好
- 问题
  - 通信性能
  - 并行编程环境
- 例子：Berkeley NOW，Alpha Farm，FXCOW

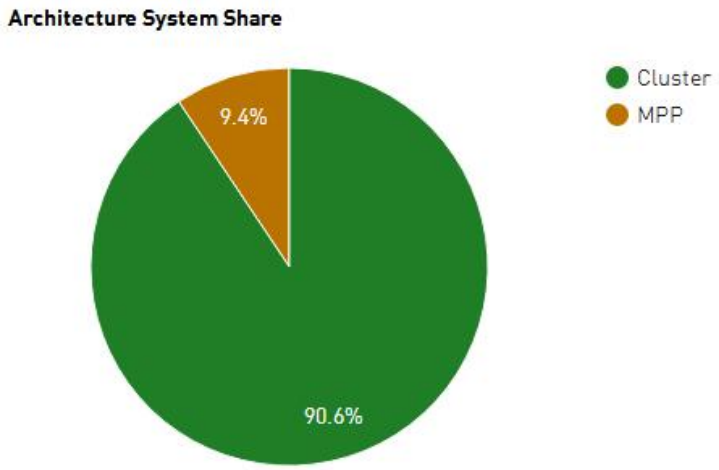


基本概念

- 集群（机群）是一组独立的计算机（节点）的集合体，通常有以下特征：
  - 各节点都是一个完整的系统：工作站，PC机或SMP机器；
  - 互连网络通常使用商品化网络，如以太网、FDDI、ATM等；
  - 网络接口与节点的I/O总线松耦合相连；
  - 各节点通常有一个本地磁盘；
  - 各节点有自己的完整的操作系统。
  - 各节点除了可以作为一个单一的计算资源供交互式用户使用外，还可以协同工作并表现为一个单一的、集中的计算资源供并行计算任务使用。
- 集群与分布式系统的区别：
  - 集群继承了分布式系统的大部分知识
  - 分布式系统通常是一个计算机的动物园，具有许多不同种类的计算机
  - 集群通常是同构，耦合度较紧密，节点间互为信任关系

TOP500中的集群

- 集群系统在高性能计算机中所占比例迅速增
  - Ø TOP500中最普通的并行机体系结构
  - Ø TOP500中目前有453个集群系统（2019.6）
  - Ø 导致了高性能计算机的“平民化”——如2003年11月排名第3的系统X（System X）是由美国弗吉尼亚工学院（Virginia Polytechnic Institute and State University, Virginia Tech）的一群师生采用商用部件花了4个月时间制造出来。



Architecture	Count	System Share (%)	Rmax (GFlops)	Rpeak (GFlops)	Cores
Cluster	453	90.6	1,254,784,968	2,031,045,875	38,869,402
MPP	47	9.4	304,790,412	432,826,680	20,236,964

集群系统的迅速发展的原因

- 集群价格便宜并且易于构建；“穷人”的解决方案，Gordon Bell奖
- 作为集群节点的工作站系统的处理性能越来越强大
- 局域网上新的网络技术和新的通信协议的引入，高带宽底延迟的节点间通信
- 集群系统比传统的并行计算机更易于融合到已有的网络系统中去
- 集群上的开发工具更成熟，传统并行计算机上缺乏一个统一的标准
- 集群的可扩放性良好

Top500中的Linux集群（2017.6）

Name/Country	Computer	Interconnect	Cores	Rmax (GFLOPS)	Rpeak (GFLOPS)
#2 Tianhe-2 (MilkyWay-2), China， 2013	TH-IVB-FEP Cluster, Intel Xeon E5-2692 12C 2.200GHz, TH Express-2, Intel Xeon Phi 31S1P	TH Express-2	3120000	33862.7	54902.4
#7: Oakforest-PACS, Japan, 2016	PRIMERGY CX1640 M1, Intel Xeon Phi 7250 68C 1.4GHz, Intel Omni-Path	Omnipath	556104	13554.6	24913.459
#8: K Computer, Japan, 2011	K computer, SPARC64 VIIIfx 2.0GHz, Tofu interconnect	Custom Interconnect	705024	10510	11280.384

2017年6月

实例分析：天河二号

- 运算速率：54.9PFLOPS（理论峰值），33.86PFLOPS（实际峰值）
- 处理器
  - 16,000个运算节点，每节点配备两个Xeon E5 12核心的中央处理器、三个Xeon Phi 57核心的协处理器，共312万个计算核心
  - 中央处理器：时钟频率为2.2GHZ的Xeon E5-2692 12，峰值性能0.2112TFLOPS
  - 协处理器：英特尔集成众核架构的Xeon Phi 31S1P协处理器，运行时钟为1.1GHz，峰值性能为1.003TFLOPS

- 互联网络：使用光电混合传输技术（Optoelectronics Hybrid Transport Technology），使用自制的TH Express-2主干拓扑结构网络连接，以13个大型路由器通过576个连接端口以光电传输介质与各个运算节点互联

SMP、MPP、集群的比较

系统特征	SMP	MPP	集群
节点复杂度	中粒度或细粒度	细粒度或中粒度	中粒度或粗粒度
节点间通信	共享存储器	消息传递 或共享变量（有DSM时）	消息传递
节点操作系统	1	N(微内核) 和1个主机OS(单一)	N (希望为同构)
支持单一系统映像	永远	部分	希望
地址空间	单一	多或单一（有DSM时）	多个
作业调度	单一运行队列	主机上单一运行队列	协作多队列
网络协议	非标准	非标准	标准或非标准
可用性	通常较低	低到中	高可用或容错
性能/价格比	一般	一般	高
互连网络	总线/交叉开关	定制	商用

主要内容

- 并行计算机系统
- 并行计算性能评测
  - 加速比性能定律
  - 可扩展性评测标准
  - 基准测试程序

什么是性能评测（Performance Evaluation）？

- 性能评测：性能评价和性能分析
- 性能评价和性能分析可以揭示高性能计算机、并行算法和并行应用程序的性能特点和性能瓶颈，指导高性能计算机、并行算法和应用程序的设计与改进
- 性能评测的目的：提高性能
  - 性能预测
  - 性能诊断/优化

并行计算性能评测的意义

- 发挥高性能计算机长处，提高高性能计算机的使用效率
- 减少用户购机盲目性，降低投资风险
- 改进系统结构设计，提高机器的性能
- 促进软/硬件结合，合理功能划分
- 优化“结构－算法－应用”的最佳组合
- 提供客观、公正的评价高性能计算机的标准

如何进行并行计算性能评测？

- 机器级的性能评测：CPU和存储器的某些基本性能指标；并行和通信开销分析；并行机的可用性与好用性以及机器成本、价格与性/价比
- 算法级的性能评测：加速比、效率、扩展性
- 程序级的性能评测：基准程序（Benchmark）

计算机的性能

- 性能（Performance）：通常是指机器的速度，它是程序执行时间的倒数
- 程序执行时间（Elapsed time）：是指用户的响应时间（访问磁盘和访问存储器的时间，CPU时间，I/O时间以及操作系统的开销）
  - 机器的时钟周期为TC，程序中指令总条数为IN，执行每条指令所需的平均时钟周期数为CPI（Cycle Per Instruction），则一个程序在CPU上运行的时间 为：IN×CPI×TC
- CPU时间：它表示CPU的工作时间，不包括I/O等待时间和运行其它任务的时间

工作负载（Workload）和速度

- 工作负载：计算操作的数目
  - 百万**浮点计算数**MFLOP (million floating point operations)
  - 百万**指令数**MI (million instruction)

- 速度：
  - 每秒百万浮点计算数MFLOPS (million floating point operations per second):

  - 每秒百万指令数目MIPS (million instructions per second):
    - instruction count/(execution time × 10^6)

并行机的性能指标

名 称	符 号	含 意	单 位
机器规模	n	处理器的数目	无量纲
时钟速率	f	时钟周期长度的倒数	MHz
工作负载	W	计算操作的数目	Mflop
顺序执行时间	T_1	程序在单处理机上的运行时间	s（秒）
并行执行时间	T_n	程序在并行机上的运行时间	s（秒）
速度	$R_n = W/T_n$	每秒百万次浮点运算	Mflop/s
加速	$S_n=T_1/T_n$	衡量并行机有多快	无量纲
效率	$E_n = S_n/n$	衡量处理器的利用率	无量纲
峰值速度	$R_{peak} = n\,R'_{peak}$	所有处理器峰值速度之积， R'_{peak}为一个处理器的峰值速度	Mflop/s
利用率	$U = R_n/R_{peak}$	可达速度与峰值速度之比	无量纲
通信延迟	t_o	传送0-字节或单字的时间	μs
渐近带宽	r_∞	传送长消息通信速率	MB/s

算法级性能评测

- 加速比（Speedup）：对于一个给定的应用，并行算法（或并行程序）相对于串行算法（或串行程序）的性能提高程度
  - Amdahl 定律
  - Gustafson定律
  - Sun Ni定律
- 可扩展性（ Scalability ）：当系统和问题规模增大时，可维持相同性能的能力，即指应用、算法和结构能否充分利用不断增长的处理器的能力
  - 等效率度量标准
  - 等速度度量标准
  - 平均延迟度量标准

并行性能测度

- 加速比（Speedup）：性能的提高通常表现为执行速度的加快

$$S_p = \frac{\text{Sequential Execution Time}}{\text{Parallel Execution Time}}$$

- 并行效率（Parallel Efficiency）：处理器的利用率
  - Efficiency = (Sequential execution time) / (Number of processors * Parallel execution time)
  - Efficiency = Speedup / Number of processors

主要内容

- 并行计算机系统
- 并行计算性能评测
  - 加速比性能定律
  - 可扩展性评测标准
  - 基准测试程序

性能提高的测度－Speedup

$$performance = \frac{work}{time}$$

$$speedup(p) = \frac{performance(p)}{performance(1)} = \frac{work(p)/time(p)}{work(1)/time(1)}$$

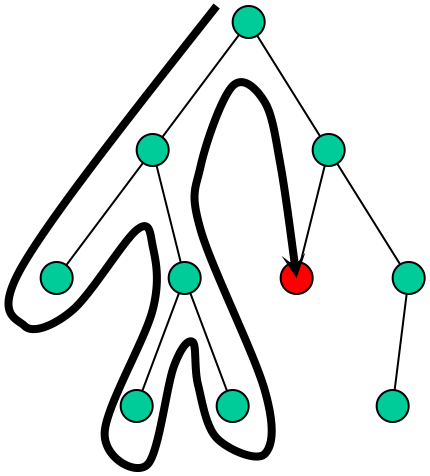
多处理器性能的基本测度

加速比的特性

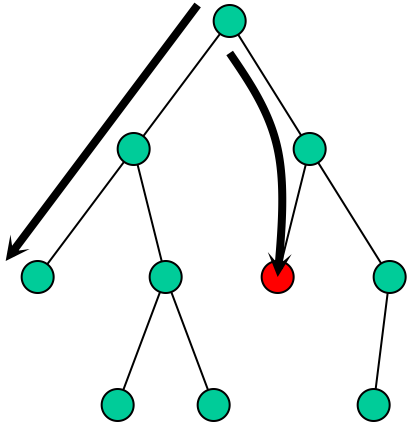
- 一般地，我们想象“2>1+1>1”
  - 基本原因：通信、同步、协调开销过大
- 不幸的是，也可能会有“1+1<1”！（学习并行计算技术的目的之一就是要理解这种情况发生的条件，避免其发生）
- 是否存在1+1>2？
  - 超线性加速比（Superlinear Speedup）

超线性加速比

- 例如在搜索问题时的不同搜索策略



Sequential search – 12 moves



Parallel search – 2 moves

超线性加速比

- 并行计算机有更多的内存。由于高速缓存（cache）的影响导致的“超线性”加速现象
  - 问题的数据规模大 – 一台处理机的cache放不下，多台处理机的cache分摊能够容纳；命中与否对应的访问时间有很大差别。
  - 访问模式随机 – 一台处理机也就没有“局部性”可利用
  - 数据并行问题 – 多处理机之间没有太多的通信，在这个数据集上的反复循环迭代操作

不同约束条件下的加速比

- 固定问题规模：Problem constrained (PC）
- 固定时间：Time constrained (TC)
- 固定存储：Memory constrained (MC)

Amdahl 定律

- P：处理器数
- W：问题规模（计算负载、工作负载，给定问题的总计算量）
  - W_s：应用程序中的串行分量,
  - f：串行分量比例（$f = W_s/W$，$W_1 = W_s + W_o$）；
  - W_P：应用程序中可并行化部分，1-f为并行分量比例
  - $W_s + W_p = W$
- T_1：串行执行时间，T_p：并行执行时间；
- S：加速比
- 出发点：
  - 固定不变的计算负载，分布在多个处理器上
  - 增加处理器加快执行速度，从而达到了加速的目的
    - ⑩ 例子：视频压缩，搜索引擎，OLTP等

固定规模（ Fixed-Size）的加速比

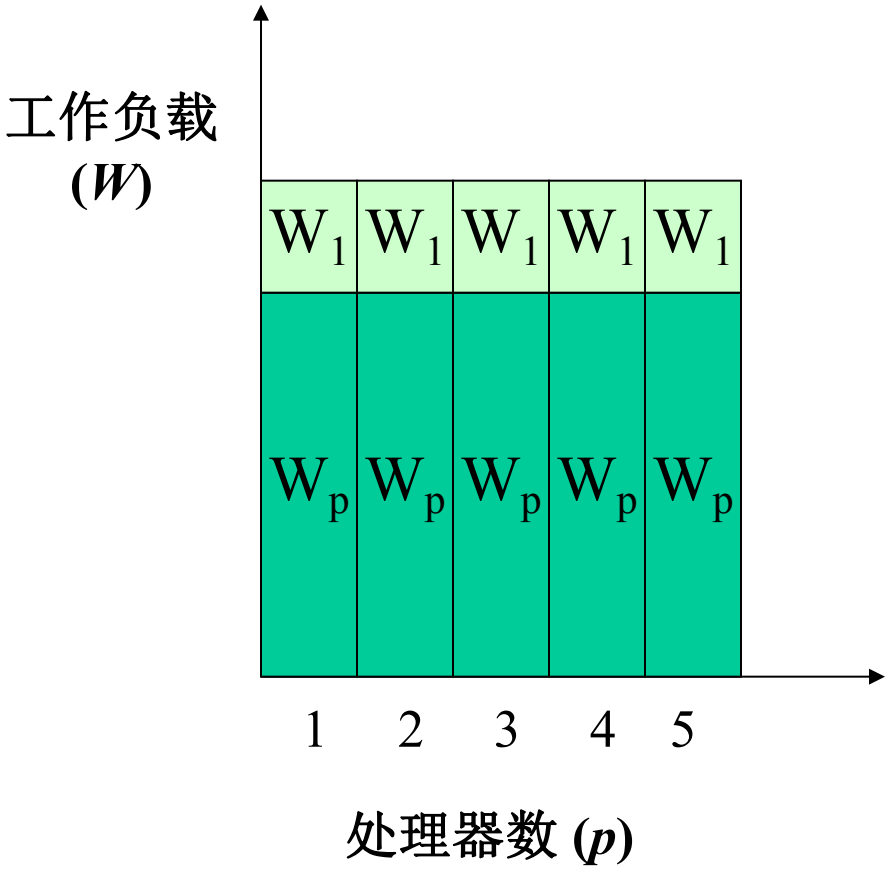
$$performance = \frac{work}{time}$$

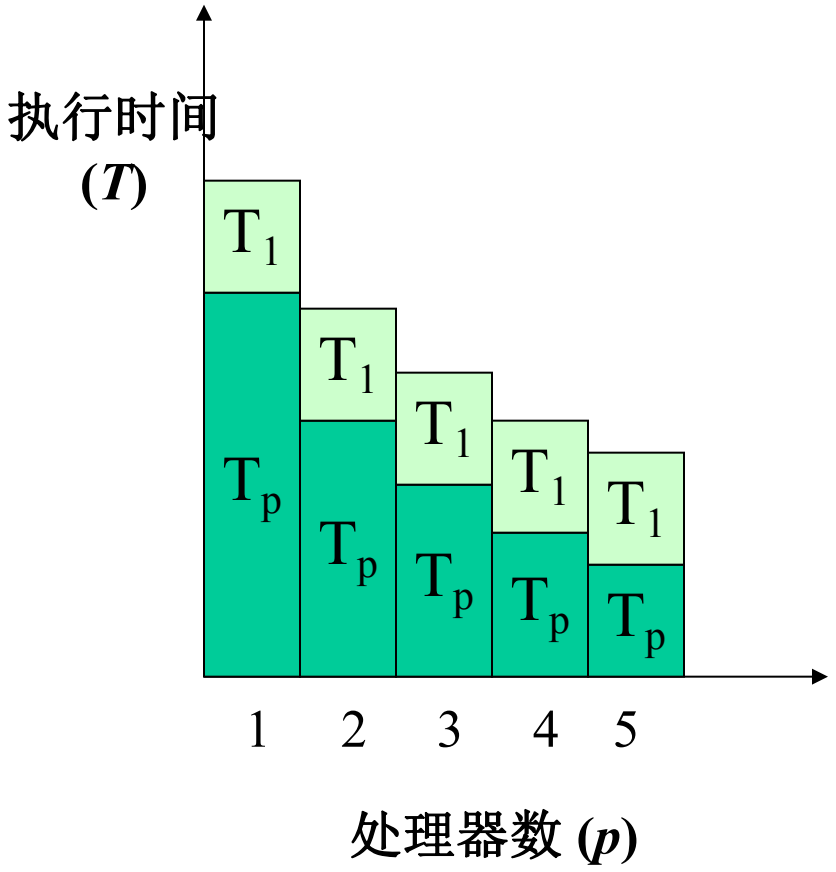
当工作固定

$$speedup(p) = \frac{performance(p)}{performance(1)} \Rightarrow speedup(p) = \frac{time(1)}{time(p)}$$

多处理器性能的基本测度

固定规模的加速比


工作负载
(W)
W1 W1 W1 W1 W1
Wp Wp Wp Wp Wp
1 2 3 4 5
处理器数 (p)


执行时间
(T)
T1
Tp
T1
Tp
T1
Tp
T1
Tp
T1
Tp
1 2 3 4 5
处理器数 (p)

Amdahl定律

$$S_{PC} = \frac{W_S + W_p}{W_S + W_P / p} = \frac{f + (1-f)}{f + \frac{1-f}{p}} = \frac{p}{1 + f(p-1)} \to \frac{1}{f} \quad as\ p \to \infty$$

Portion of sequential computation

# of processors

式4.5

例如：如果 f=10%, 则最大的加速比是10，无论用多少个个处理器

增强的Amdahl定律

考虑开销W_o，包括并行和通信的开销

$$S_{PC} = \frac{W_S + W_P}{W_S + \frac{W_P}{p} + W_o} = \frac{W}{fW + \frac{W(1-f)}{p} + W_o} = \frac{p}{1 + f(p-1) + W_o p / W}$$

$$\rightarrow \frac{1}{f + \frac{W_o}{W}} \quad as\ p \rightarrow \infty$$

式4.8

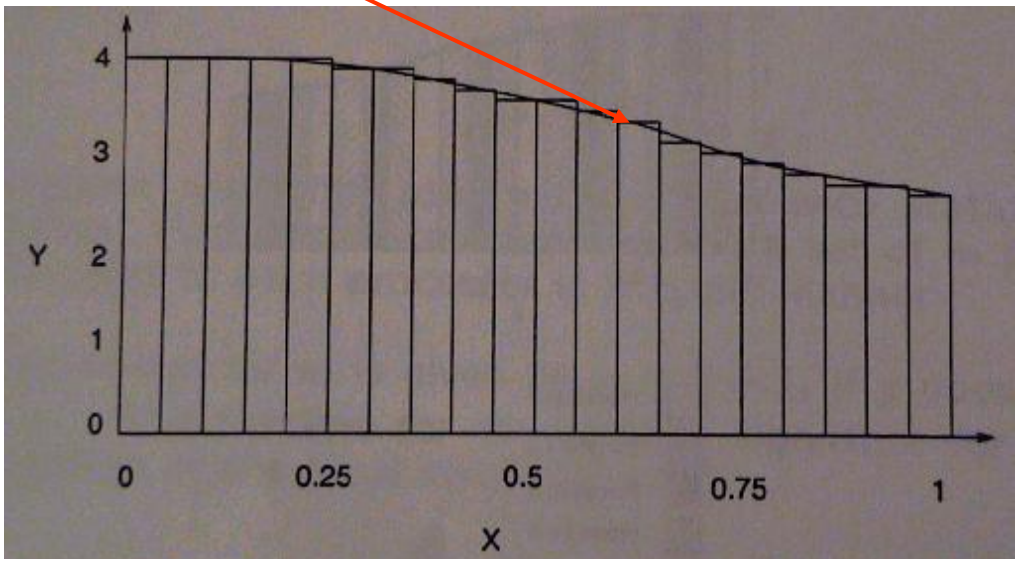
例子：计算π

- 如何用数值方法计算π=3.1415…？ (P367)

$$\pi = \int_0^1 \frac{4}{1+x^2}\,dx \approx \sum_{0 \le i < N} \frac{4}{1+\left(\frac{i+0.5}{N}\right)^2} \cdot \frac{1}{N}$$

串行算法

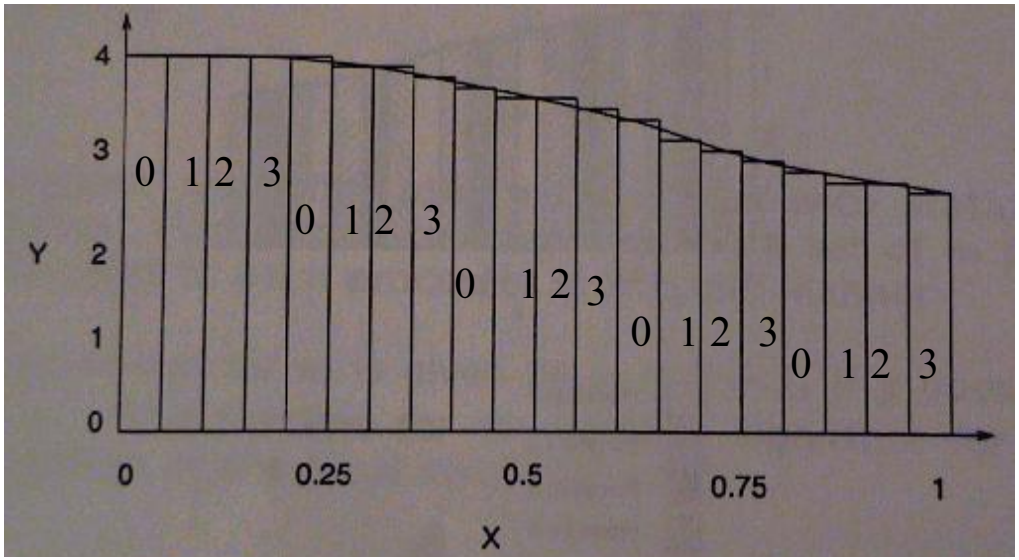
$\frac{4}{1+x^2}$




```
computepi()
{
    h=1.0/n;
    sum =0.0;
    for (i=0;i<n;i++) {
        x=h*(i+0.5);
        sum=sum+4.0/(1+x*x);
    }
    pi=h*sum;
}
```

计算π：并行算法

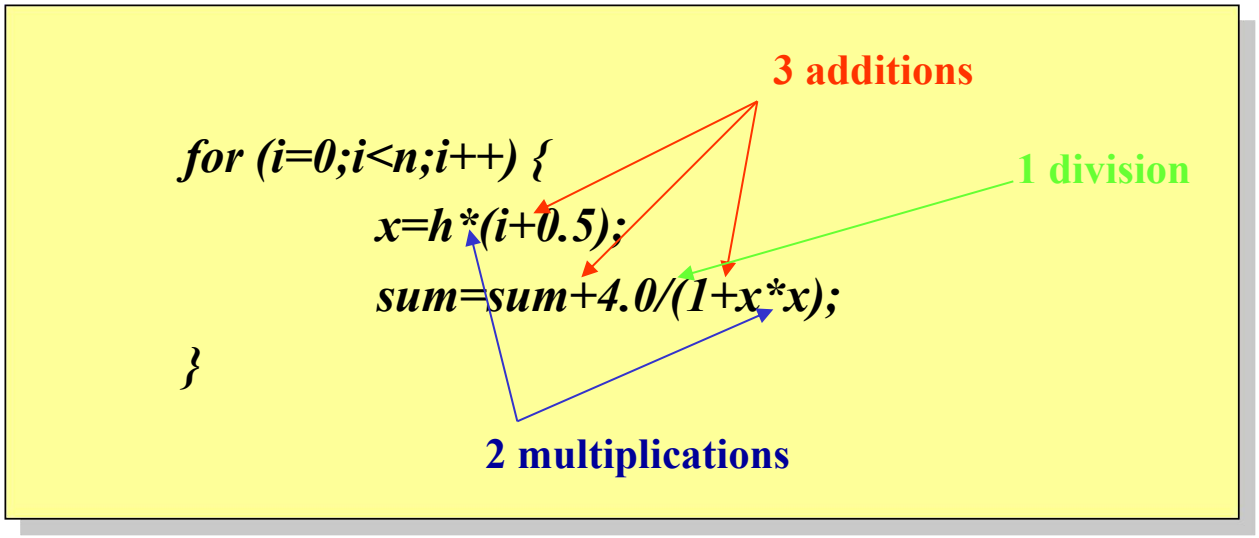
- 循环分配计算任务，每个处理器计算n/p 个点
- 将p个处理器上的本地和相加就得到π的值



计算π：分析

- 假设π的计算是在 n 个点上进行的。串行算法对x轴上每个点都要进行6次操作（两次乘法、一次除法、三次加法）
- 因此，对于n个点，串行算法要执行的操作数是（t_0是单位计算时间）：

$$T_s = 6n * t_0$$



计算π：分析（2）

- 使用p个处理器的并行算法，每个处理器计算 m 个点

$$m \leq \frac{n}{p} + 1$$

- 因此计算π的并行算法的运行时间是：

$$T_p = 6m * t_0 = (6\frac{n}{p} + 6)t_0$$

计算π：分析（3）

- 最后得到π，还要进行本地和的累加，这可以通过基于树的合并算法，可在$\log_2(p)$ 个步骤内完成
- 因此并行算法的总的计算时间是（t_c是单位通信时间）：

$$T_p = 6m * t_0 = (6\frac{n}{p} + 6)t_0 + \log(p)(t_0 + t_c)$$

- 并行算法的加速比是：

$$S_p = \frac{T_s}{T_p} = \frac{6n}{6\frac{n}{p} + 6 + \log(p)(1 + t_c / t_0)}$$

计算π：分析（4）

- 改写上式：

$$S_p = \frac{p}{1+\frac{p}{n}+\frac{pc\log(p)}{6n}} \Rightarrow S_p = \frac{p}{1+(p-1)f(n,p)}$$

- 可以得到Amdahl定律的串行分量f(n,p)为：

$$f(n,p) = \frac{p}{(p-1)n} + \frac{pc\log(p)}{6n(p-1)}$$

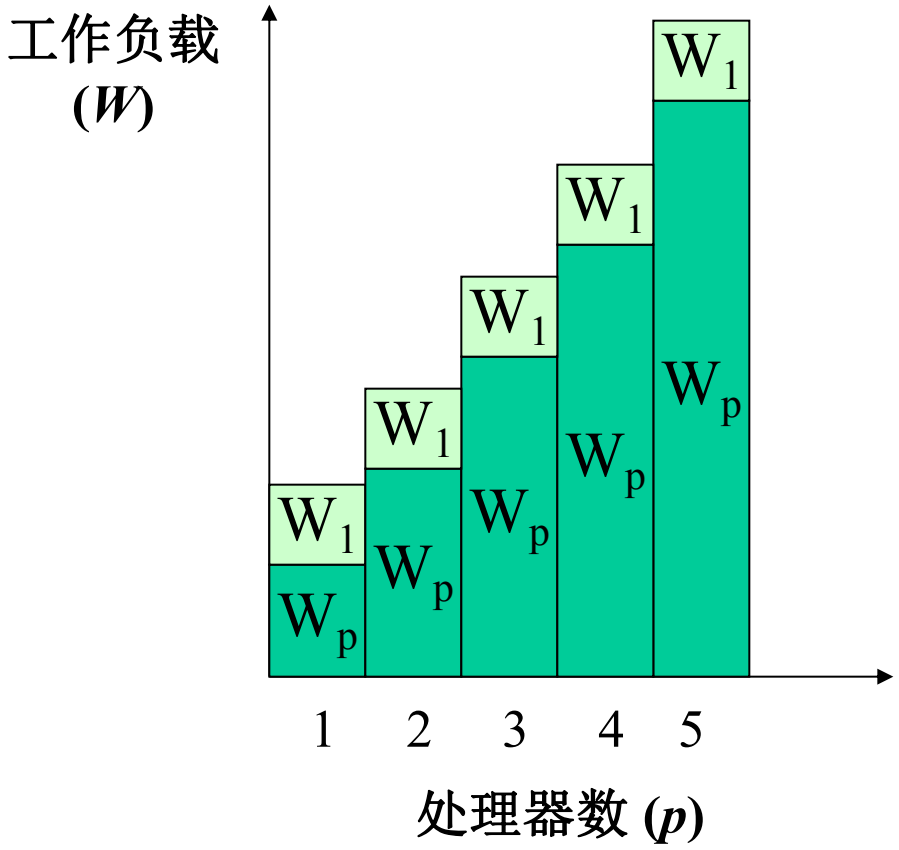
- 这样的并行算法是有效（effective）的，因为：

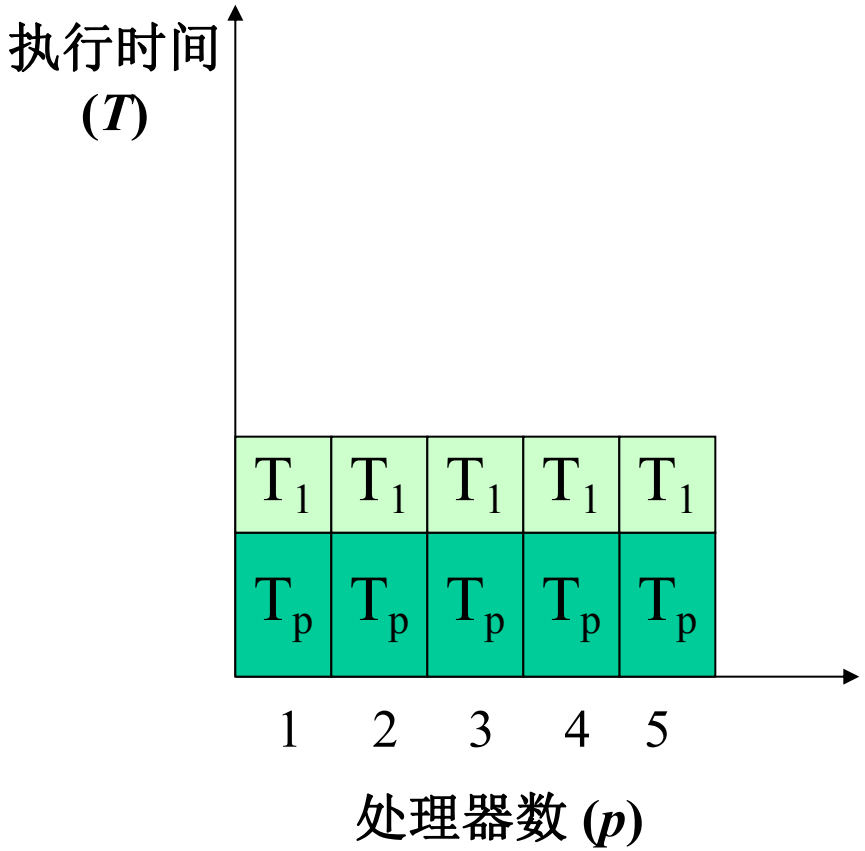
$$f(n,p) \to 0 \text{ as } n \to \infty \text{ for fixed } p$$

Gustafson定律

- 许多重要应用要求在固定时间内能处理的数据越多越好（例如数值天气预报）；而在固定时间里，依靠增加处理器数总可以指望能处理更多的数据（尽管它们之间的关系随问题而变）
- 其他例子：
  - 结构分子中的有限元方法FEM（Finite Element Method）
  - 流体力学的有限差分方法FDM（Finite Difference Method）

时间固定（Fixed-Time）的加速比





Gustafson定律

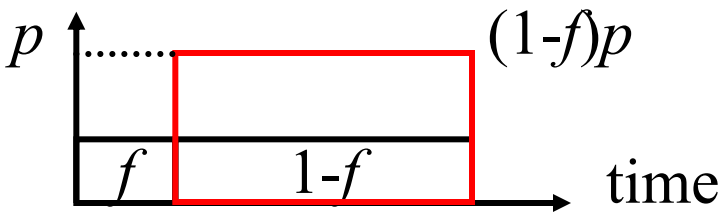
- Gustafson加速定律：

$$S_{TC} = \frac{W_S + pW_P}{W_S + p \cdot W_P / p} = \frac{W_S + pW_P}{W_S + W_P}$$

式4.9

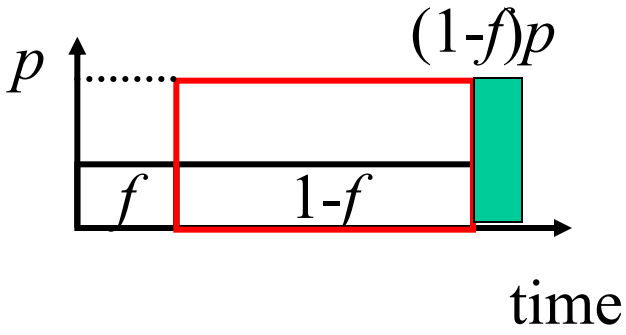
$$S_{TC} = f + p\,(1\text{-}f) = p + f\,(1\text{-}p) = p\text{-}f\,(p\text{-}1)$$

式4.10



Gustafson定律

- 考虑并行开销W_o：



$$S_{TC} = \frac{W_S + pW_P}{W_S + W_P + W_o} = \frac{f + p(1-f)}{1 + W_o / W}$$

式4.11

Sun & Ni定律

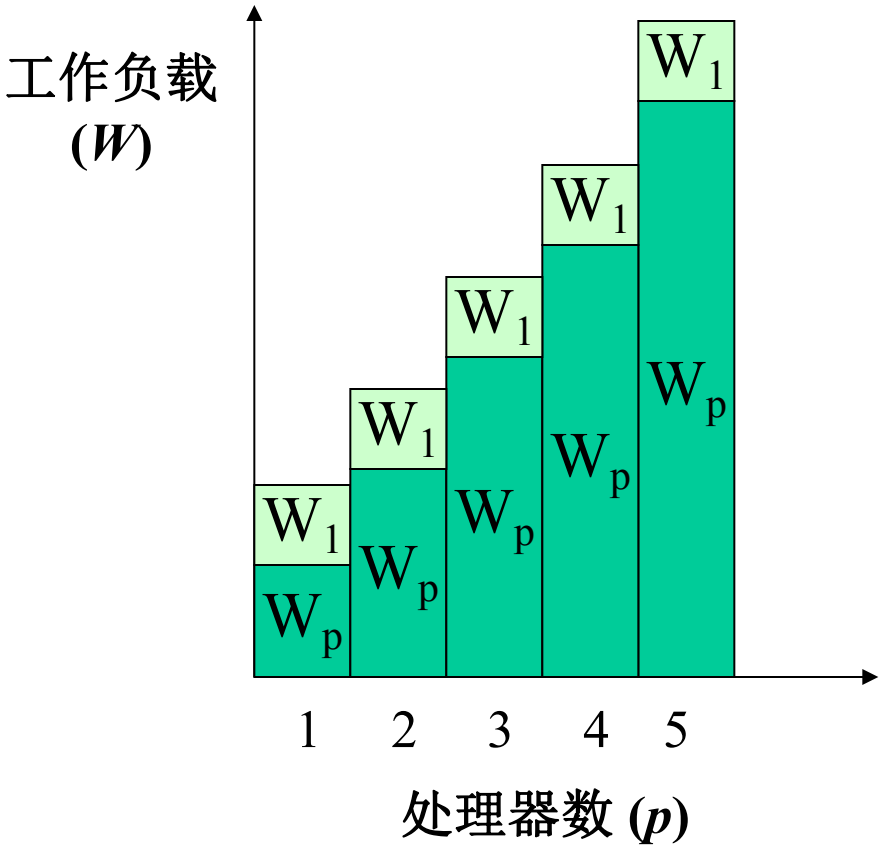
- 只要存储空间许可，应尽量增大问题规模以产生更好和更精确的解（此时可能使执行时间略有增加）
  - 例如N体（N-body）问题
- 假定在单节点上使用了全部存储容量M并在相应于W的时间内求解之，此时工作负载：

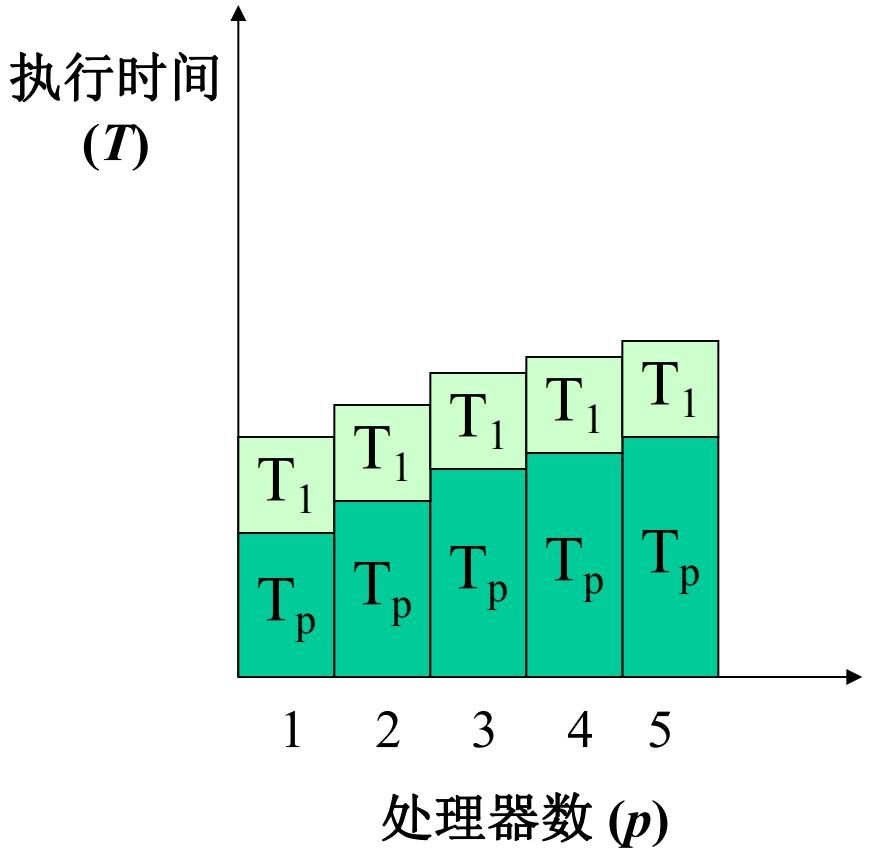
$$W = fW + (1-f)W$$

- 在p 个节点的并行系统上，能够求解较大规模的问题是因为存储容量可增加到pM。令因子$G(p)$表示存储容量增加到p倍时并行工作负载的增加量，所以扩大后的工作负载：

$$W = fW + (1-f)G(p)W$$

存储受限（Memory-Boundary）的加速比



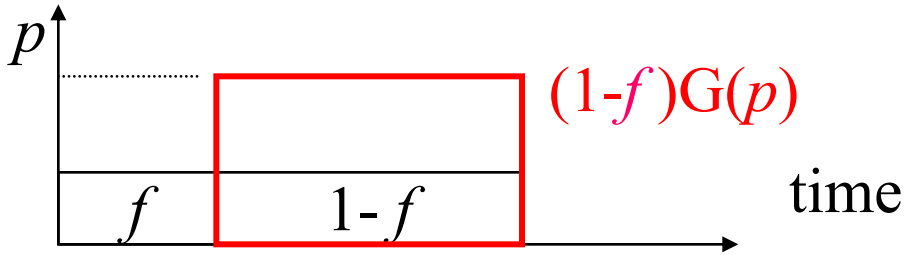


Sun & Ni定律

- 存储受限的加速公式：

$$S_{MC}=\frac{fW+(1-f)G(p)W}{fW+(1-f)G(p)W/p}=\frac{f+(1-f)G(p)}{f+(1-f)G(p)/p}$$

式4.13



- 并行开销W_o:

$$S_{MC}=\frac{fW+(1-f)WG(p)}{fW+(1-f)G(p)W/p+W_o}=\frac{f+(1-f)G(p)}{f+(1-f)G(p)/p+W_o/W}$$

式4.14

Sun Ni定律特性

- G(p)=1时就是Amdahl加速定律；
- G(p)=p 变为 $f + p(1-f)$，就是Gustafson加速定律
- G(p)>p时，相应于计算机负载比存储要求增加得快，此时 Sun Ni 加速均比 Amdahl 加速和 Gustafson 加速为高。

加速比讨论

- 参考的加速比经验公式： $p/\log p \le S \le P$
  - 线性加速比：很少通信开销的矩阵相加、内积运算等
  - $p/\log p$的加速比：分治类的应用问题
- 通信密集类的应用问题： $S = 1 / C(p)$ (式4.16)
- 超线性加速
- 绝对加速（科学研究）：最佳并行算法与串行算法
- 相对加速（工程应用）：同一算法在单机和并行机的运行时间

主要内容

- 并行计算机系统
- 并行计算性能评测
  - 加速比性能定律
  - 可扩展性评测标准
  - 基准测试程序

可扩展性

- 影响加速比的因素：系统规模与问题规模
  - 求解问题中的串行分量
  - 并行处理所引起的额外开销（通信、等待、竞争、冗余操作和同步等）
  - 加大的处理器数超过了算法中的并发程度
- 增加问题的规模有利于提高加速的因素：
  - 较大的问题规模可提供较高的并发度
  - 额外开销的增加可能慢于有效计算的增加
  - 算法中的串行分量比例不是固定不变的（串行部分所占的比例随着问题规模的增大而缩小）
- 增加系统规模（处理器数）会增大额外开销和降低处理器利用率，所以对于一个特定的并行系统（算法或程序），它们能否有效利用不断增加的处理器的能力应是受限的，而度量这种能力就是可扩放性这一指标。

可扩展性（2）

- 经常，在一定规模下求解某个问题的“执行时间”对理解系统的“性能”来说还不够
- 针对并行计算机系统的性能，关心它在系统规模和应用数据规模变化时的执行时间往往更有意义
- 可扩展性（Scalability）：系统的“规模性能”（通常译为“可扩展性”，有时也称为“可扩放性”，“可伸缩性”）

{性能,系统规模,数据规模}的综合测度

对scalability的追求

- 当系统规模扩大时，希望系统的处理能力能相应增强
- 能力增强
  - 同样的时间能处理更多的数据
  - 处理同样的数据花更少的时间
- 相应
  - 成线性比例，对数比例，…

- 可扩展性：调整什么和按什么比例调整
  - 并行计算要调整的是处理数p和问题规模W，
  - 两者可按不同比例进行调整，此比例关系（可能是线性的，多项式的或指数的等）就反映了可扩放的程度。

等效率函数

Speedup:

$$S = \frac{T_e}{T_p} = \frac{T_e}{\frac{T_e + T_o}{p}} = \frac{p}{1 + \frac{T_o}{T_e}} = \frac{p}{1 + \frac{W_o}{W}}$$

式4.18

Efficiency:

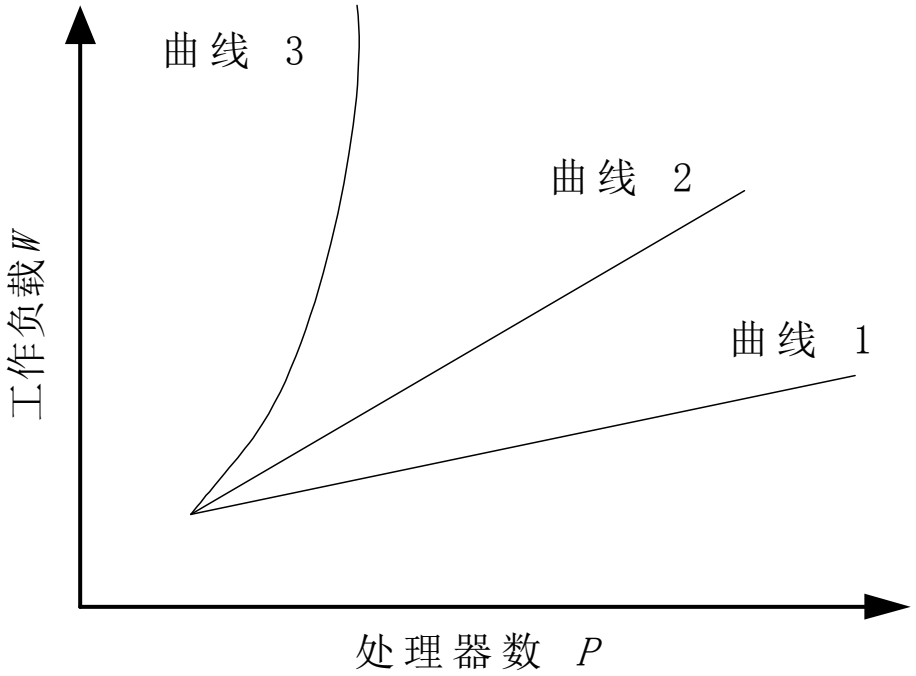
$$E = \frac{S}{P} = \frac{1}{1 + \frac{T_o}{T_e}} = \frac{1}{1 + \frac{W_o}{W}}$$

式4.19

- 如果问题规模W保持不变，处理器数p增加，开销T_o增大，效率E下降。为了维持一定的效率（介于0与1之间），当处理数p增大时，需要相应地增大问题规模W的值。由此定义函数$f_E(p)$为问题规模W随处理器数p变化的函数，为**等效率函数**（ISO-efficiency Function）

等效率函数曲线

- 曲线1表示算法具有很好的扩放性；曲线2表示算法是可扩放的；曲线 3表示算法是不可扩放的。



主要内容

- 并行计算机系统
- 并行计算性能评测
  - 加速比性能定律
  - 可扩展性评测标准
  - 基准测试程序

程序级性能评测

- 基准测试程序（Benchmark）
  - 一组标准的测试程序
  - 提供一组控制测试条件
  - 步骤的规则说明（测试平台环境、输入数据、输出结果和性能指标等 ）
- 基准测试程序的分类
  - 宏观测试程序（Macro-benchmark）：计算机系统作为一个整体来测试其性能
  - 微观测试程序（Micro-benchmark ）：测试机器的某一特定方面的性质

测试程序分类

- 按生成方式：真实、核心、小、综合程序
- 按应用类型：科学计算、商业应用、信息处理等
- 按程序功能：宏观测试程序、微观测试程序

基准测试程序（Benchmark Suites）

类　型	名　称	意 义 用 途
宏观测试程序	PARKBENCH	并行计算
	NAS	并行计算CFD
	SPEC	混合基准测试程序
	Splash	并行计算
	STAP	信号处理
	TPC	商业应用
微观测试程序	LINPACK	数值计算（线性代数）
	LMBECH	系统调用和数据移动（UNIX）
	STREAM	存储器带宽

LINPACK测试程序

- Linpack是国际上流行的用于测试高性能计算机系统浮点性能的benchmark
- 通过对高性能计算机采用LU分解求解线性代数方程组能力的测试，评价高性能计算机的浮点处理性能
  - Linpack 100：早期版本，程序不可修改
  - Linkpack 1000：用全精度64位字长的子程序求解1000阶线性方程组的速度，测试的结果以Mflops（每秒百万次浮点运算）为单位
  - HPL（High Performance Linpack）：可以修改问题规模、CPU数量及通信、问题分块等，用在Top500的测试中

HPL简介

- 高性能Linpack：**HPL**(High Performance Linpack)
  - 针对大规模的并行计算机系统的测试， 2000年9月发布1.0版
  - 是TOP500超级计算机排名的主要依据
  - 采用MPI实现，基于BLAS库或VSIPL库
  - 用户可以选择矩阵的大小（问题规模）、使用各种优化方法来执行测试程序，寻求最佳的测试结果
  - 浮点峰值＝总计算量 / 计算时间
    - ✓ 总计算量＝$2/3 * N^3 - 2*N^2$
    - ✓ 计算时间：可通过系统时钟获得

课程小结

- 高性能计算机系统
  - SMP，MPP，Cluster
- 工作负载
  - 执行时间，Mflops, MIPS
- 加速比
  - 加速比的性质
  - 不同约束条件下的加速比
- 可扩展性
  - 可扩展性的含义和目标
  - 测度：等效率测度
- 基准测试
  - 程序级性能评测基准测试
  - LINPACK测试

推荐读物和网站

- 《并行计算—结构、算法、编程》（第三版）
  - 第3章：典型并行计算机系统介绍
  - 第4章：并行计算性能评测

- The BlueGene/L Team, “An Overview of the BlueGene/L Supercomputer”, Supercomputing 2002 Technical Papers
- X. H. Sun and L. Ni, “Scalable Problems and Memory-Bounded Speedup”, Journal of Parallel and Distributed Computing, Vol.19, Sept. 1993, pp27-37
- LINPACK Benchmark: http://www.netlib.org/benchmark

下一讲

- 并行计算模型及并行算法设计策略
  - 《并行计算－结构、算法、编程》（第三版）第5，6章