



華東師範大學

EAST CHINA NORMAL UNIVERSITY

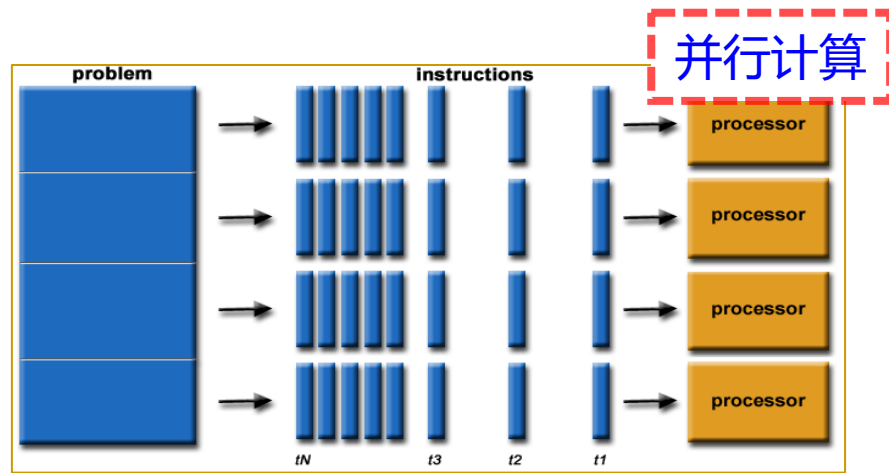
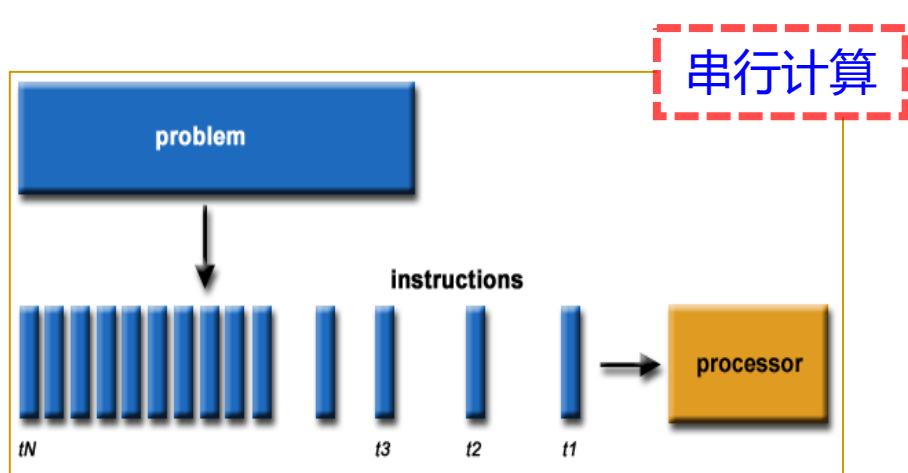
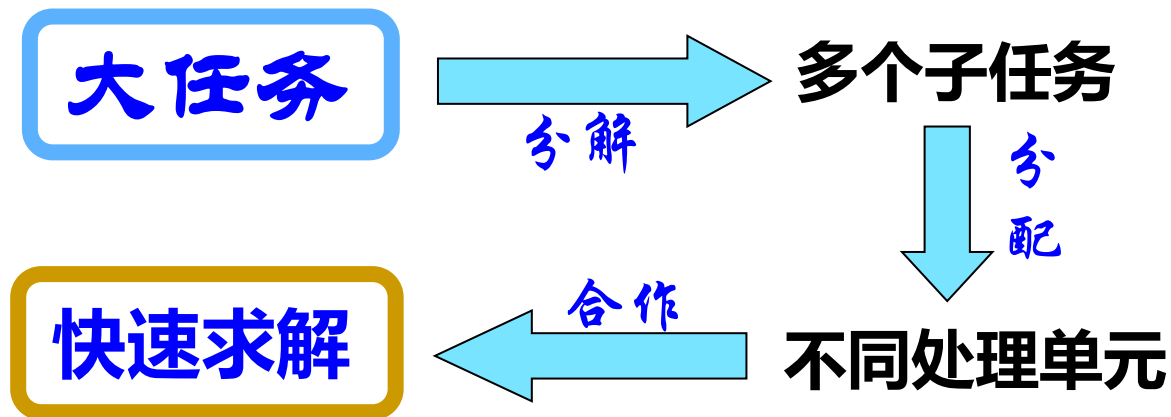


并行计算介绍

潘建瑜

MATH@ECNU

并行计算/高性能计算/超级计算



并行计算研究内容和目标

研究内容

- ▶ 并行计算机的体系结构
- ▶ 并行算法设计与分析
- ▶ 并行实现技术：编程实现，优化性能
- ▶ 并行应用：开发并行应用软件

主要目标

(1) 求解问题**更快**； (2) 求解问题规模**更大**

并行计算基本条件

并行计算三要素

- 硬件：并行计算机/体系结构
- 算法：并行算法设计/应用问题的并行度
- 软件：并行编程环境/Linux/Fortran/C/C++/MPI/OpenMP



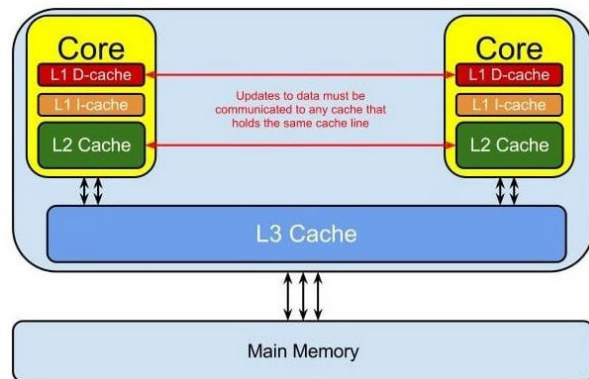
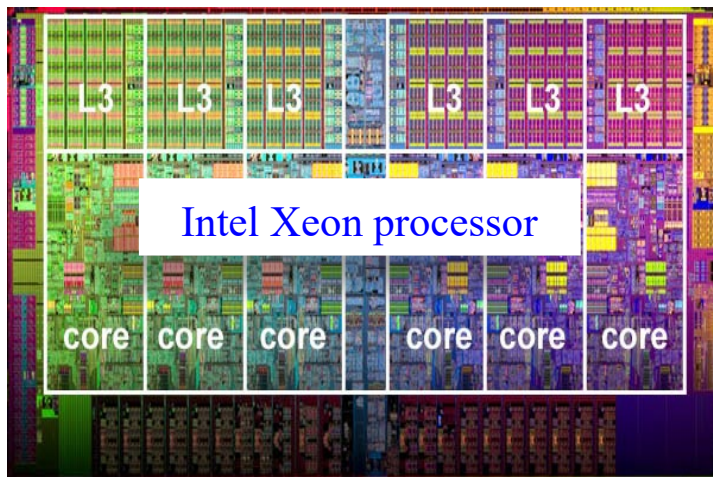
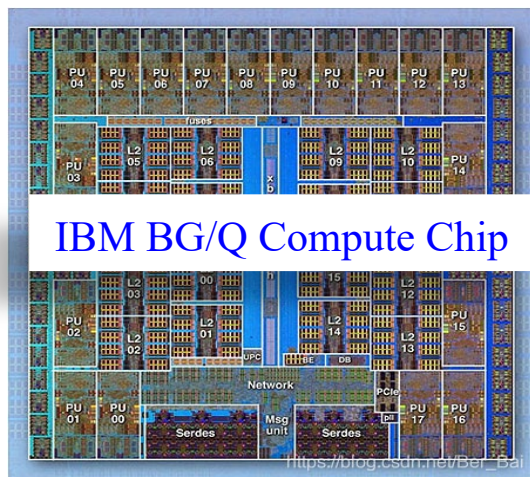
1

并行计算机

—— 体系结构与发展历史

并行计算机

- 从硬件角度讲，当前所有计算机都可以被认为是并行的
多功能单元、多执行单元、多硬件线程



并行计算机



Frontier, 处理器核芯: 1,110,144个

神威太湖之光, 处理器核芯: 10,649,600个



并行计算机架构

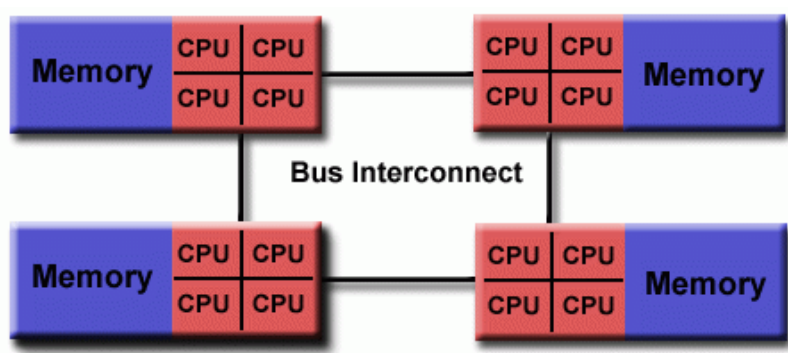
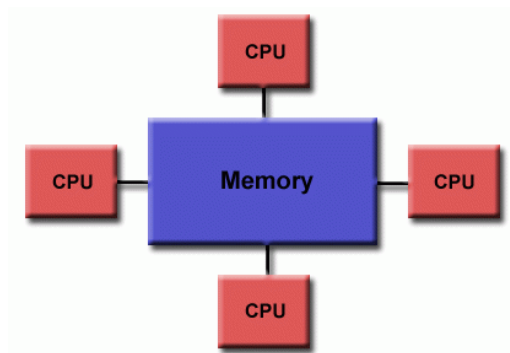
- ▶ 共享内存系统
- ▶ 分布式内存系统
- ▶ 混合内存系统

并行计算机架构 – 共享内存

□ 共享内存系统

- **统一内存寻址**：所有处理器以全局寻址的方式访问所有的内存空间。

对称多处理器机器 (Symmetric Multi-processor, SMP, 左图)



- **非统一内存寻址**：由多个物理上相连的 SMP 组成，每个 SMP 都可以访问其它 SMP 的内存，但访问是有差别的 (右图)

并行计算机架构 – 共享内存

共享内存优点

- (1) 全局地址空间提供了一种**用户友好的编程方式**
- (2) 内存与 **CPU** 的物理连接使得任务之间的数据共享既快速又统一

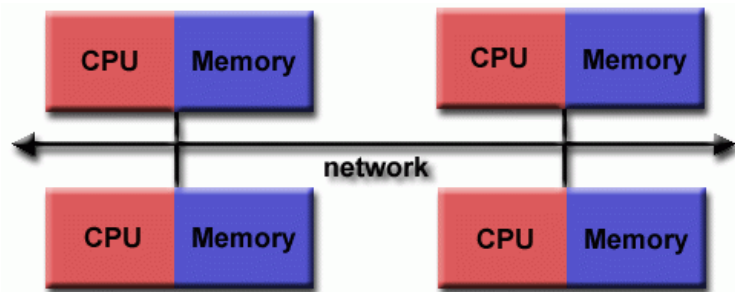
共享内存缺点

内存和 **CPU** 之间缺少较好的可扩展性，过多 **CPU** 会大幅增加缓存和内存管理的负担（几何级别）

并行计算机架构 – 分布式内存

□ 分布式内存系统

- 处理器会有各自所对应的内存，不存在全局内存地址空间
- 数据交换：通过**网络**传输完成/通讯

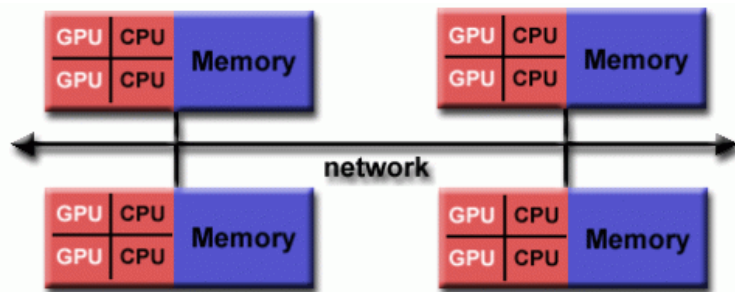
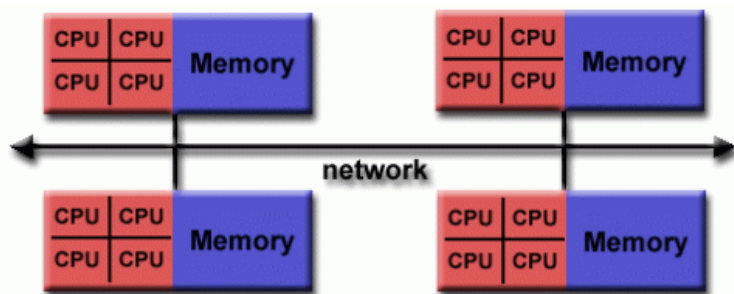


优点：扩展性好；访问自身数据速度快；维护成本低

缺点：（1）编程复杂；（2）访问其他结点上的数据需要长得多的时间

并行计算机架构 – 混合内存

□ 分布式 --- 共享内存 混合系统



- 继承了共享内存和分布式内存的优缺点
- 是当前的主流模式，扩展性强，但编程复杂

MPP (Massively Parallel Processing)

每个节点相对独立，有一个或多个处理器，有自己的操作系统和独立内存，避免内存访问瓶颈，各个节点只能访问自己的内存模块，扩展性较好。

并行计算机发展

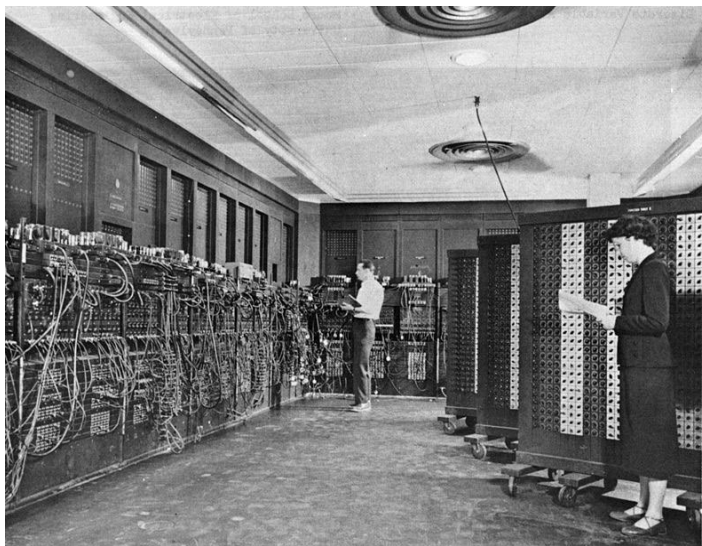
- ▶ 国际并行计算机发展介绍
- ▶ 我国并行计算机发展介绍
- ▶ 当前世界超级计算机情况

第一台计算机

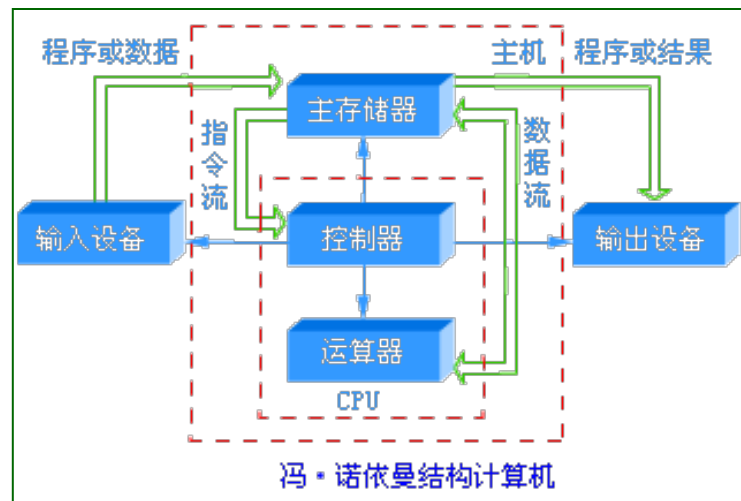
❑ 1946 年，世界上第一台计算机 ENIAC 诞生

Electronic Numerical Integrator And Computer

电子数值积分计算机，美国宾夕法尼亚大学



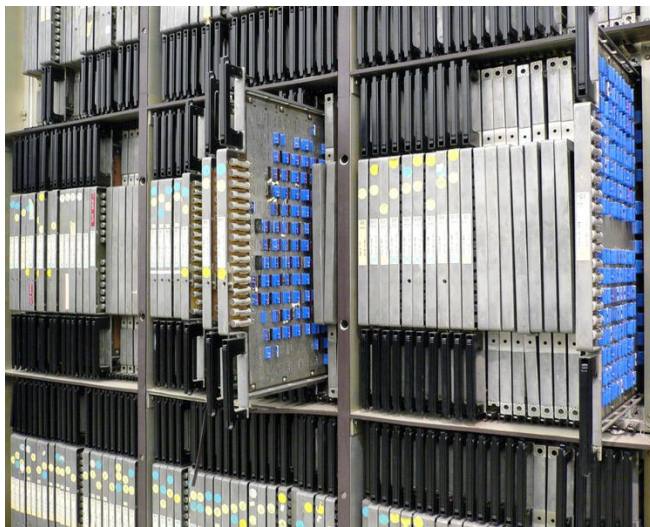
冯·诺依曼
现代计算机之父
数学博士



- ▶ 占地约 170 平方米，重约 30 吨，耗电150千瓦
- ▶ 运算速度：5000 次加法/秒或 500 次乘法/秒，15分钟换一个零件（真空管）
- ▶ 用途：弹道计算和氢弹研制

第一台并行计算机

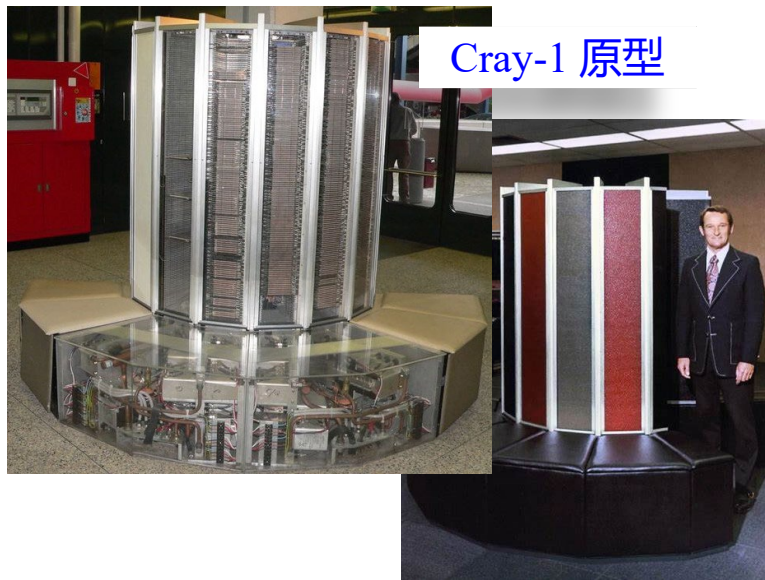
□ 1972 年，第一台并行计算机 ILLIAC IV (美国伊利诺依大学)



- ▶ 1960 年代末开始建造，1972 年建成
- ▶ 1974 年运行第一个完整程序，1976 年运行第一个应用程序
- ▶ 64 个处理器，公认的 1981 年前最快，1982 年退役
- ▶ 可扩展性好，但可编程性差

超级计算元年

❑ 超级计算元年的标志：Cray-1 向量机



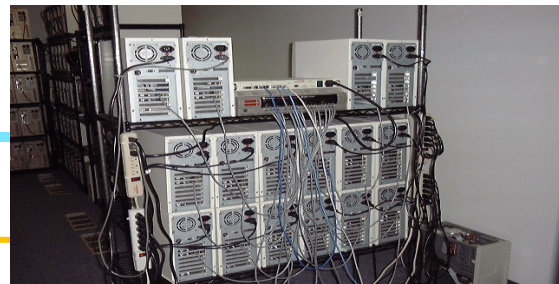
西摩·克雷 Seymour Cray (1925-1996)
电子工程学学士，应用数学硕士
超级计算之父，Cray 研究公司的创始人
亲手设计 Cray 机型的全部硬件与操作系统，作业系统用机器码编写完成。
1984 年时，公司占据了超级计算机市场 70% 的份额。1996 年 Cray 研究公司被 SGI 收购，2000 年被出售给 Tera 计算机公司，2019 年被 HPE 收购。2022 年 11 月发布的全球最快超级计算机排名，前 10 中有三个是 HPE Cray 制造，包括第 1 名。

- ▶ 一般将 Cray-1 投入运行的 1976 年称为“超级计算元年”
- ▶ 编程方便，但可扩展性差

21世纪发展现状

Cluster 机群 / 集群

- ▶ 每个节点含多个商用处理器，节点内部共享存储
- ▶ 采用商用机群和交换机，通过前端总线连接节点，节点分布存储
- ▶ 各个结点采用 Linux 操作系统、GNU编译系统和作业管理系统



世界第一台 Beowulf 集群, 1994

Constellation 星群

- ▶ 每个节点是一台小并行机
- ▶ 各个节点运行专用的节点操作系统、编译系统和作业管理系统

MPP 大规模并行机

并行计算机发展趋势

- ❑ 异构计算：CPU + 加速器（GPU/斜处理器等）
- ❑ 绿色计算：计算性能与绿色节能兼备，低能耗，回收利用等，
- ❑ 软件开发：并行编程工具的发展远远落后于硬件的发展
- ❑ 应用开发
- ❑ 网格计算，云计算，AI 计算，.....
- 技术自主创新，超算软件人才培养，底层软件库建设，...

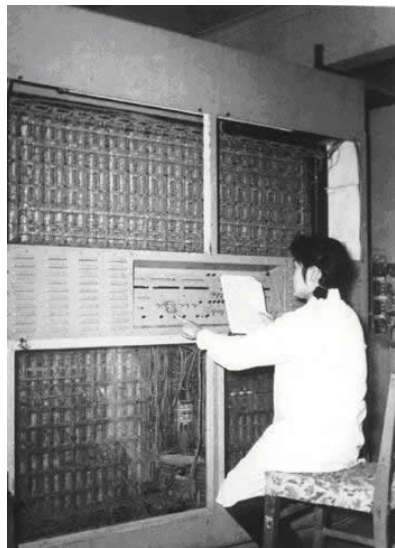
- ▶ International Conference for HPC, Networking, Storage, and Analysis
<https://sc23.supercomputing.org>
- ▶ ISC High Performance <https://www.isc-hpc.com>
- ▶ CCF 全国高性能计算学术年会 <https://hpcchina.ccf.org.cn>
- ▶ Top 500 <https://top500.org>

我国的并行计算机发展

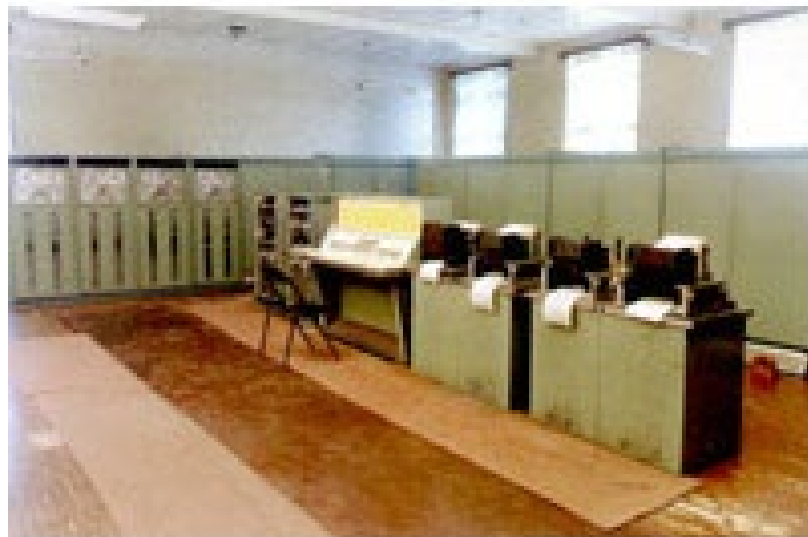
- 银河，天河：国防科大
- 神威：国家并行计算机工程技术研究中心
- 曙光：中科曙光
- 深腾：联想集团
- 其他：浪潮、华为（鲲鹏）等

我国第一台计算机

□ 1958 年第一台国产计算机诞生 — 103 型计算机



103 型，运行速度 1500 次每秒



109 丙机 (1968) ，两弹一星 “功勋机”

并行机 —— 银河系列

□ 银河 I 号 1983年12月，我国第一台每秒运算达 1 亿次以上的计算机



- ▶ 1992年，银河-II，峰值达 10 亿次/秒
- ▶ 1997年，银河-III，峰值达 100 亿次/秒
- ▶ 2000年，银河-IV，峰值达 10000 亿次/秒

并行机 —— 天河系列

□ 天河一号 / 天河二号 / 天河三号

2009 年国内首台超千万亿次，继美国后第二个能研制千万亿次超级计算机国家

- ▶ 2010年10月，升级后的天河一号排名 Top500 第一
- ▶ 2013年6月，天河二号再次问鼎世界第一
- ▶ 2017年11月，用国产加速器 Matrix-2000 升级后的天河二号排名第 2，当前排名第 10
- ▶ 2018年，据称天河三号 E 级计算原型机通过验收，但未参与排名



国家超级计算天津中心



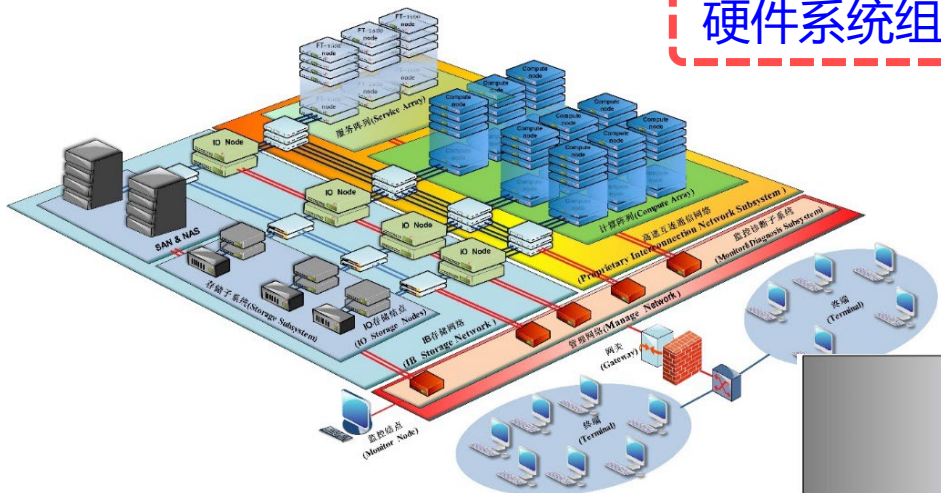
广州超级计算中心



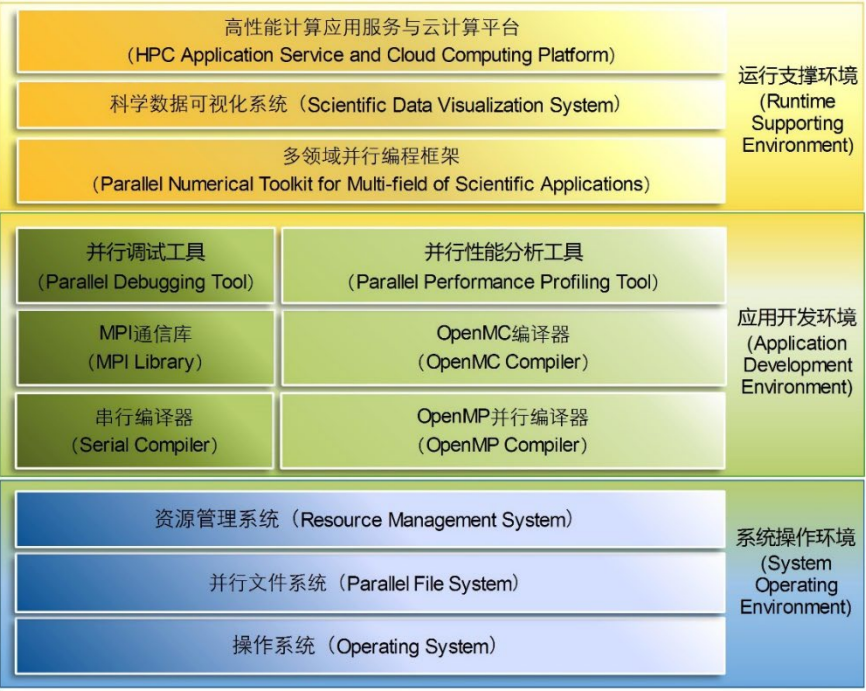
国家超级计算天津中心

天河二号

硬件系统组成



软件系统架构



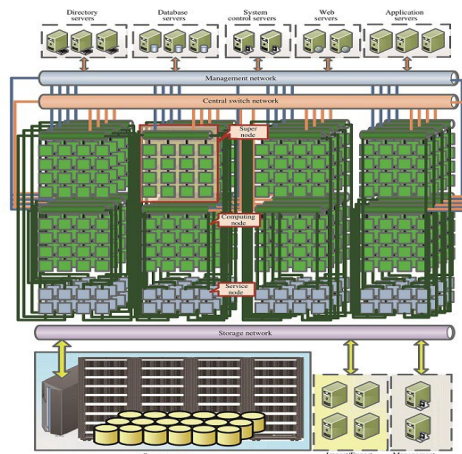
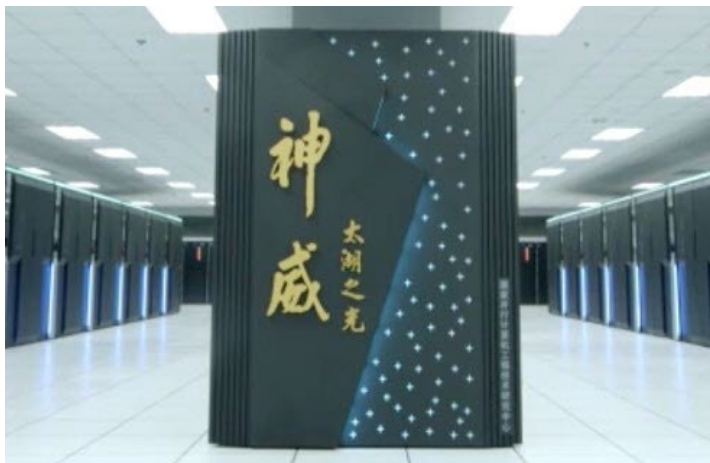
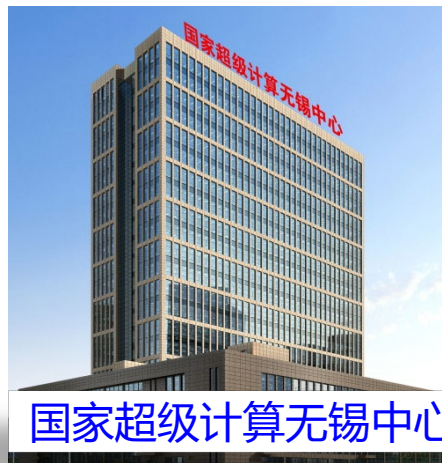
并行机 —— 神威系列

□ 神威 I 1999年，国家并行计算机工程技术研究中心牵头研制，峰值 3840 亿次/秒

□ 神威·太湖之光

2016 年，全球首台峰值超 10 亿亿次每秒，连续四次 Top500 第一，目前排名第 7
安装 40960 个自主研发“申威26010”众核处理器，采用64位自主申威指令系统

2021 年，据称海洋之光 E 级超算做了基准测试，目前已陆续投入使用



世界最快超级计算机



<http://www.top500.org>

2022.06

Rank	System	Cores	Rmax (PFlop/s)	Rpeak (PFlop/s)	Power
1	Frontier/HPE Cray/USA	8,730,112	1,102.00	1,685.65	21,100
2	Fugaku/Japan	7,630,848	442.01	537.21	29,899
3	LUMI/HPE Cray/Finland	1,110,144	151.90	214.35	2,942
4	Summit/IBM/USA	2,414,592	148.60	200.79	10,096
5	Sierra/IBM/USA	1,572,480	94.64	125.71	7,438
6	Sunway TaihuLight/China	10,649,600	93.01	125.44	15,371
7	Perlmutter/HPE Cray/USA	761,856	70.87	93.75	2,589
8	Selene/NVIDIA/USA	555,520	63.46	79.22	2,646
9	Tianhe-2A/China	4,981,760	61.44	100.68	18,482
10	Adastra/HPE Cray/France	319,072	46.10	61.61	921

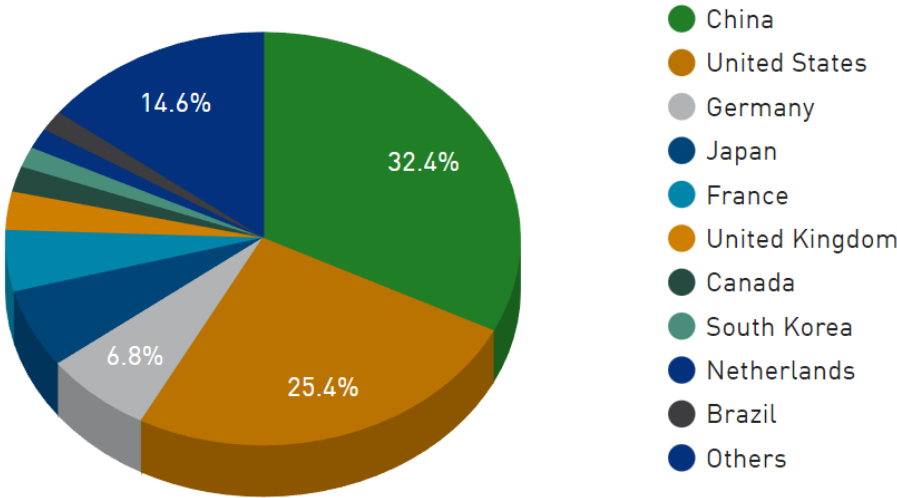
世界最快超级计算机



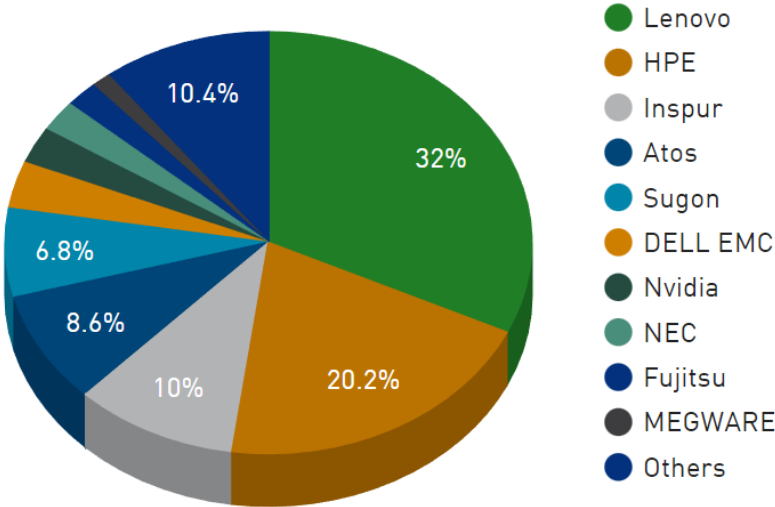
2022.11

Rank	System	Cores	Rmax (PFlop/s)	Rpeak (PFlop/s)	Power
1	Frontier/HPE Cray/USA	8,730,112	1,102.00	1,685.65	21,100
2	Fugaku/Japan	7,630,848	442.01	537.21	29,899
3	LUMI/HPE Cray/Finland	2,220,1288	309.10	428.70	6,016
4	Leonardo/BullSequana/Italy	1,463,616	174.70	255.75	5,610
5	Summit/IBM/USA	2,414,592	148.60	200.79	10,096
6	Sierra/IBM/USA	1,572,480	94.64	125.71	7,438
7	Sunway TaihuLight/China	10,649,600	93.01	125.44	15,371
8	Perlmutter/HPE Cray/USA	761,856	70.87	93.75	2,589
9	Selene/NVIDIA/USA	555,520	63.46	79.22	2,646
10	Tianhe-2A/China	4,981,760	61.44	100.68	18,482

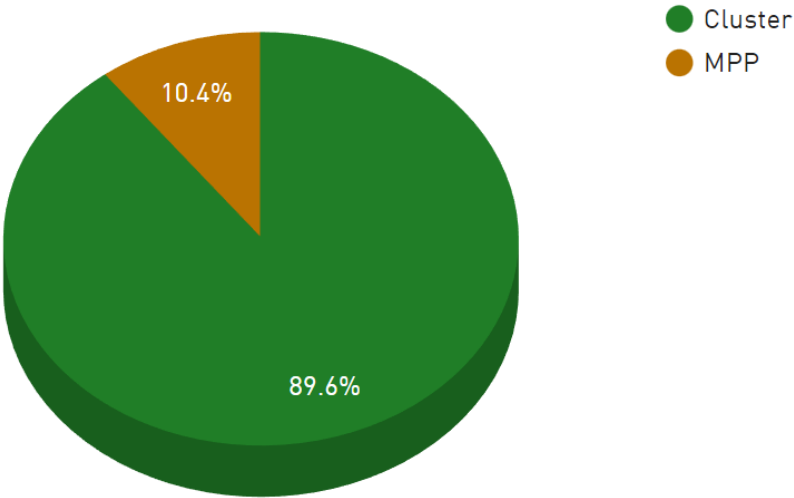
Countries System Share



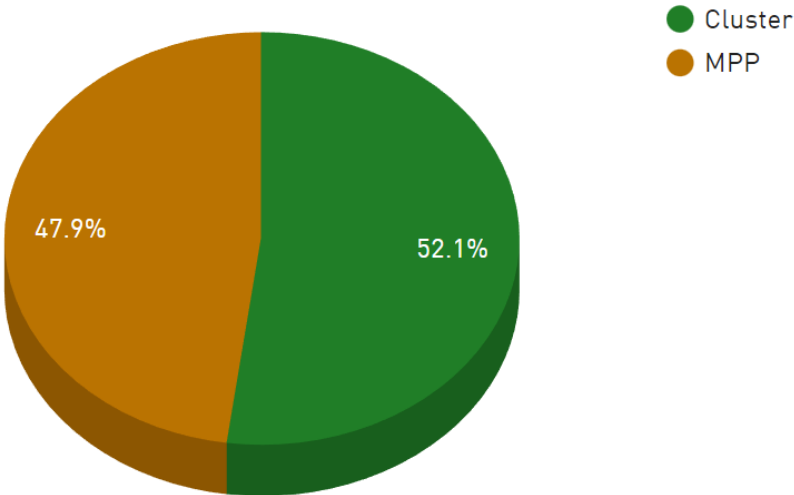
Vendors System Share



Architecture System Share

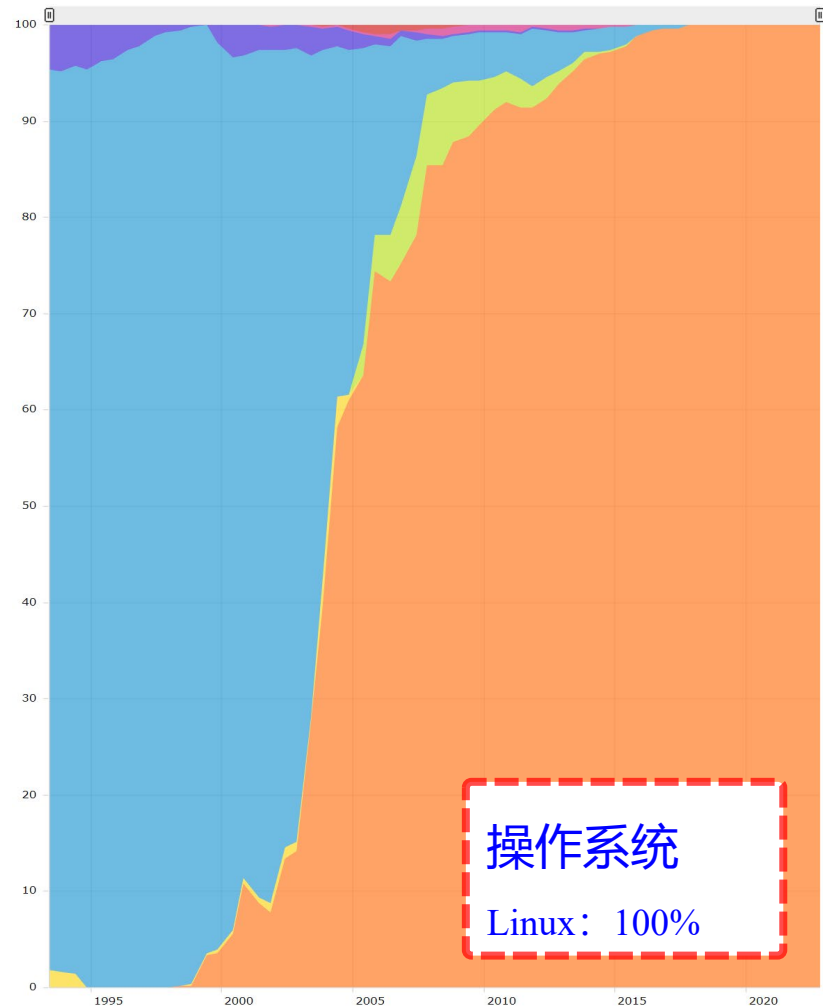
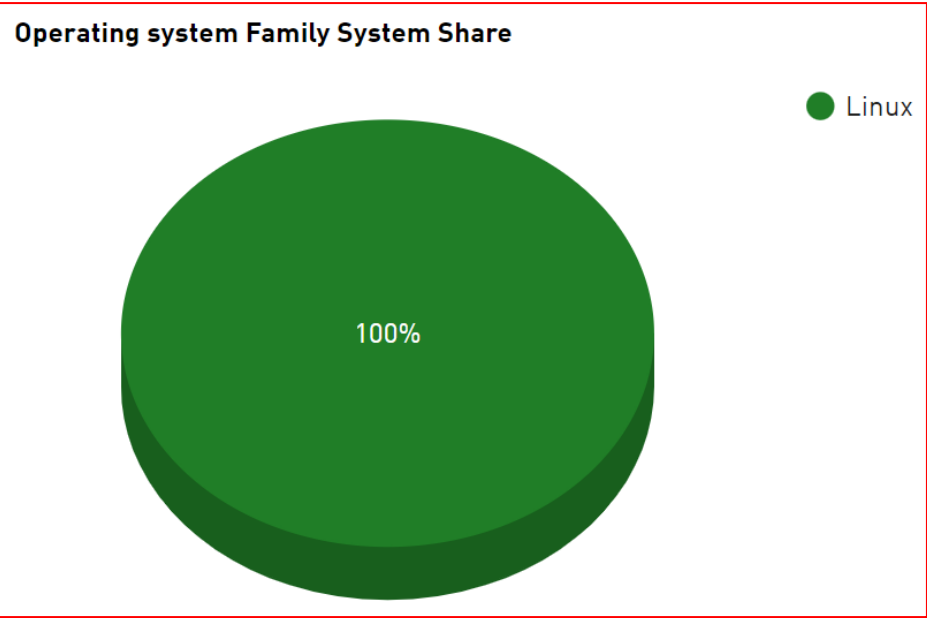


Architecture Performance Share

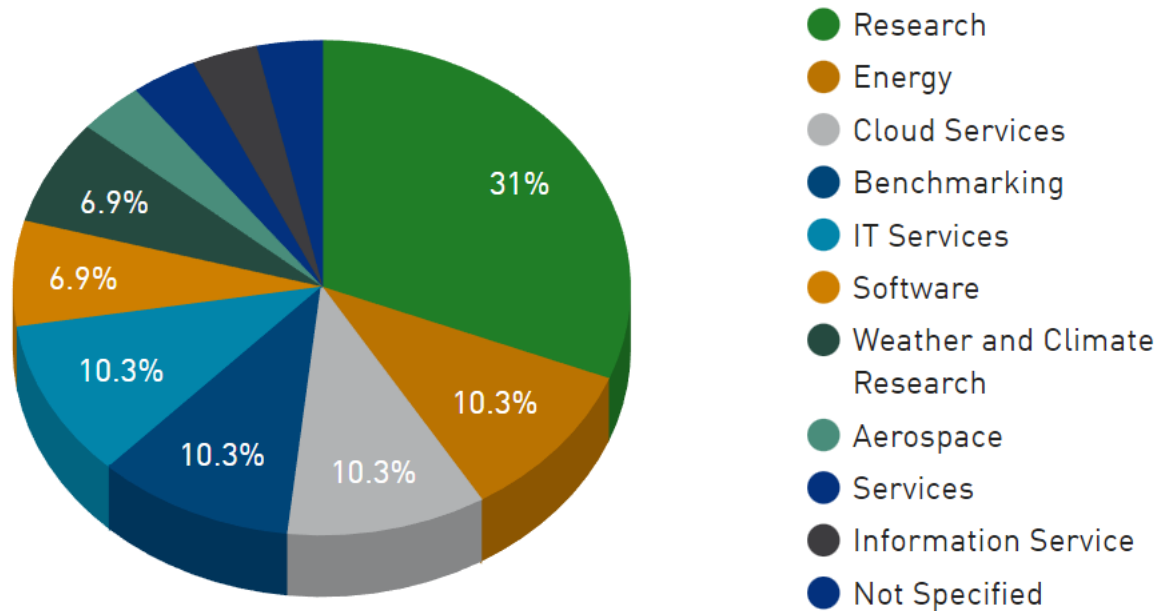


数据统计：操作系统

Top 500 @ 2022.11

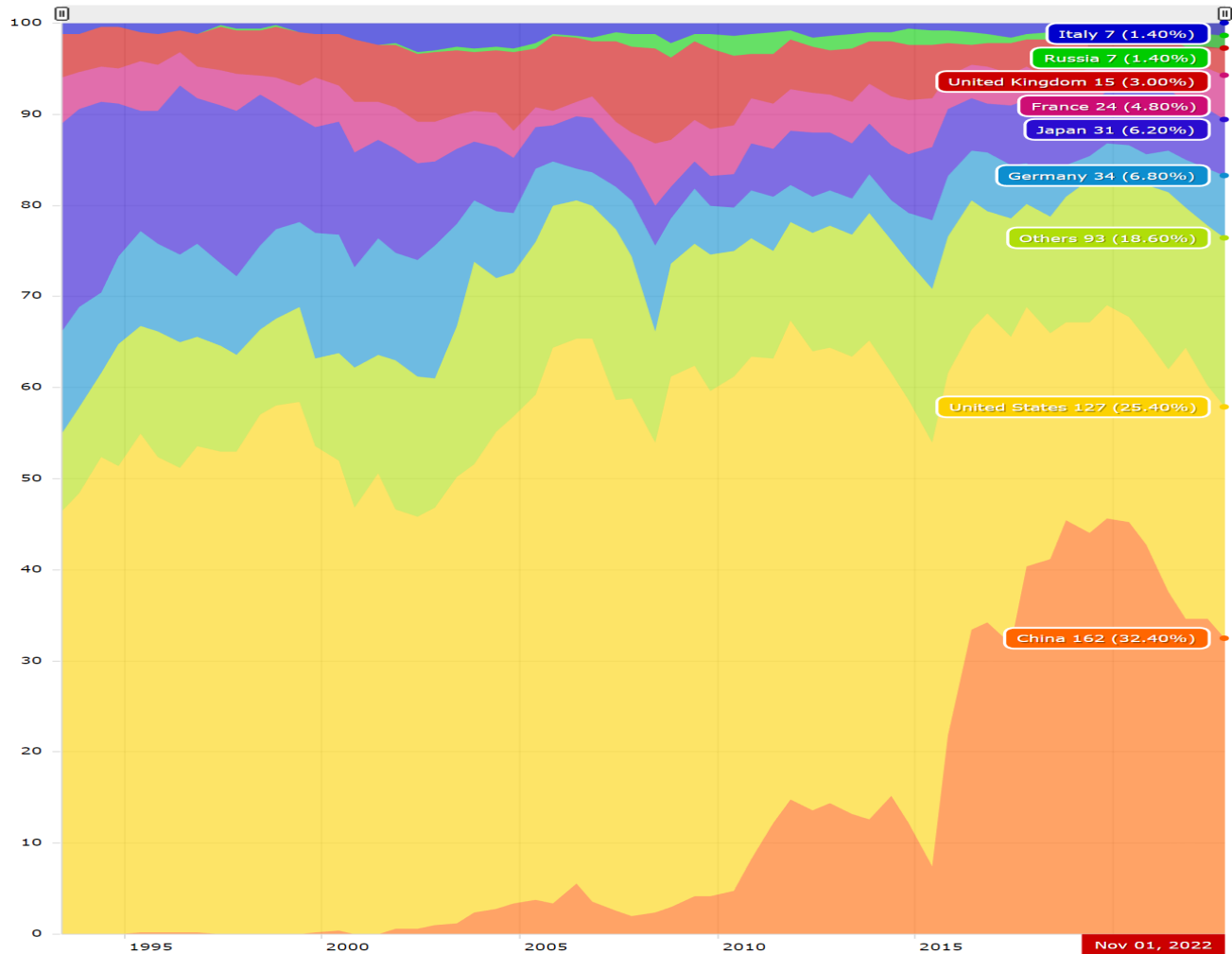


Application Area System Share



数据统计：历史数据

Top 500 @ 2022.11



国家与地区
(2022.11)

中国：162

美国：127

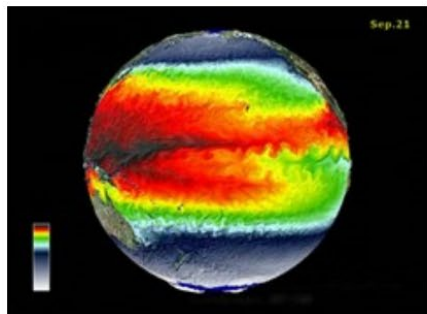
德国：34

日本：31

法国：24

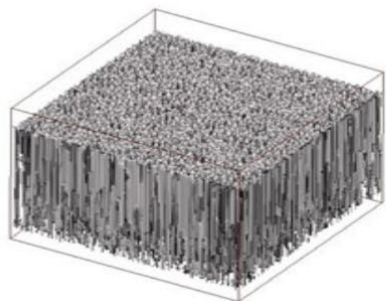
为什么并行计算

- ▶ 需求，需求，需求
- ▶ 科学界，工程界，工业界，商业界，...



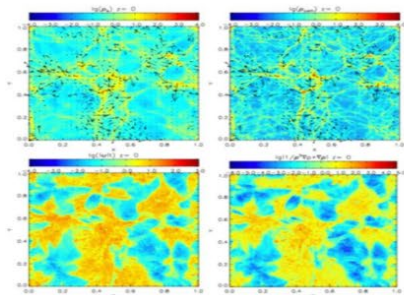
全球海洋模拟数据可视化

海洋流速是研究海洋气候变化的关键问题之一，超算中心开发了海洋流场分析模块，可交互式操作的向前、向后多种轨迹线技术直观再现海水的流动，帮助科学家分析海洋环流问题。



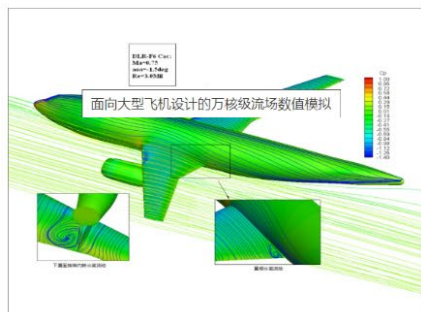
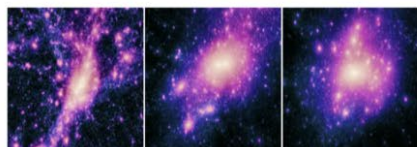
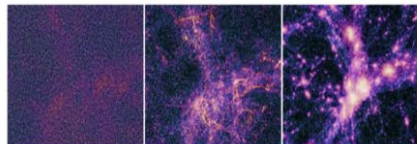
高温变形钛合金组织演化的大规模并行模拟

该课题由中科院金属研究所的徐东生研究员承担。中科院超算中心完成了相关程序的并行化工作。在课题组人员的努力和超级计算中心人员的通力合作支持下，在5千核规模上...



星系尺度上的恒星形成过程和动力学反馈的大规...

该课题由中科院紫金山天文台的冯珑珑研究员承担。而基于GADGET的N体数值模拟则有望获得超大规模的宇宙学数值模拟样本，该样本将对我国的重大科学工程如LAMOST红移巡天计...

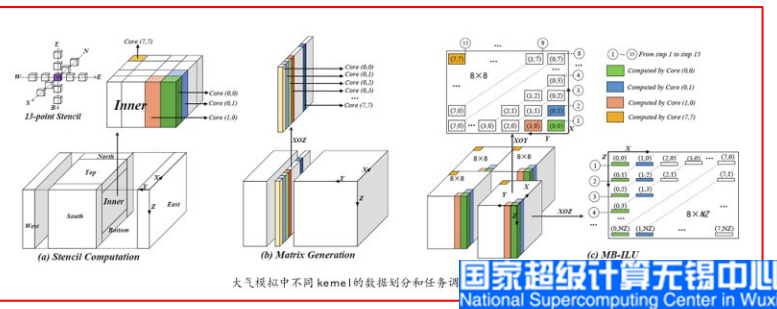


大规模高动态范围时序粒子数据可视化

超级计算中心为国家天文台提供海量高动态范围时序粒子数据的可视化处理服务。根据数据的特点，提出了基于直方图均衡的时序映射算法和合并树特征跟踪算法，有效解决了相...

面向大型飞机设计的万核级流场数值模拟

用户在深腾7000上开展了湍流问题的高分辨率数值计算及湍流机理研究，使用自主研发的混合网格数值模拟软件CCFD-UG软件对某型民用飞机高升力流场进行了数值模拟。



千万核可扩展全球大气动力学全隐式模拟

本项目设计并开发了一种用于大气动力学中经常出现的强时间依赖问题的高可扩展性全隐式求解器。该求解器使用异构多重网格局部分解算法，显著加快了求解器的收敛过程，并利用粗粒度并行。获得2016年戈登贝尔 (Gorden Bell) 奖



2

并行算法设计

—— 并行计算基础知识

一些基本概念

□ 进程 (Process) 和线程 (Thread)

- ▶ 进程是程序在某个数据集合上的一次运行活动，也是操作系统进行资源分配的基本单位，通俗来说，进程就是运行中的程序，或者程序的一次执行过程。
- ▶ 进程不仅包含正在运行的程序实体，并且包括这个运行的程序中占据的所有系统资源，比如说 CPU、内存、网络资源等。
- ▶ 线程是进程的一个执行单元，是比进程更小的独立运行的基本单位。一个进程中可以开启多个线程，它们共享这个进程的资源，比进程更容易创建，也更容易撤销。

Table 1.1 Difference in initialization overhead between the creation of a thread and the creation of process on an Intel i5 CPU using Visual Studio.

Function call	Time
CreateProcess(..)	12.76 ms
CreateThread(..)	0.037 ms

一些基本概念

□ 节点 (Node)

一个独立的“计算机单元”，通常由多个 CPU 组成，有自己的内存。

□ 通信/通讯 (Communications)

并行任务（进程/线程）之间的数据交换，无论采用何种实现方式。

□ 同步 (Synchronization)

并行任务之间的实时协调，所有任务都执行到同一点后才能往后继续。

□ 计算通信比 (Computation-to-Communication Ratio) / 粒度 (Granularity)

计算与通信的比率，通常更高的比值会带来更好并行效率

粒度：两次通信之间处理器计算工作量的粗略估计，分粗粒度和细粒度

缓存内存速度比较

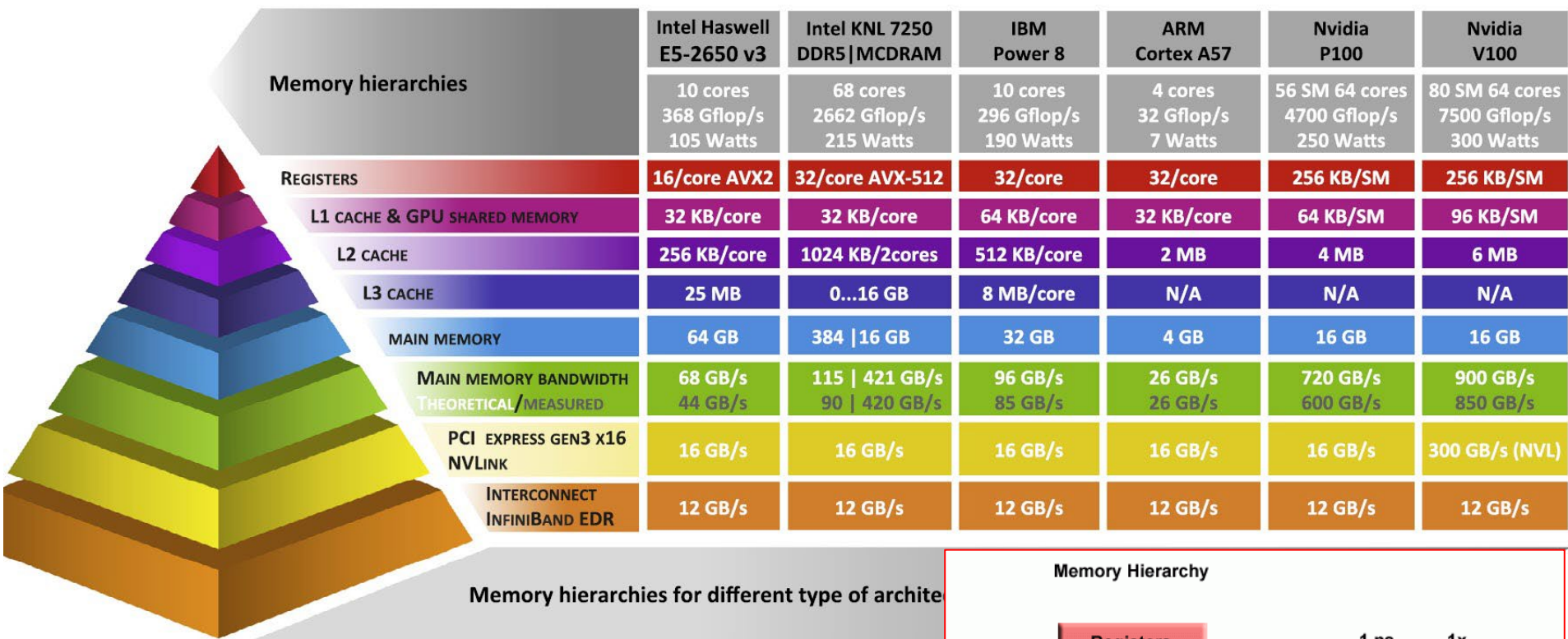
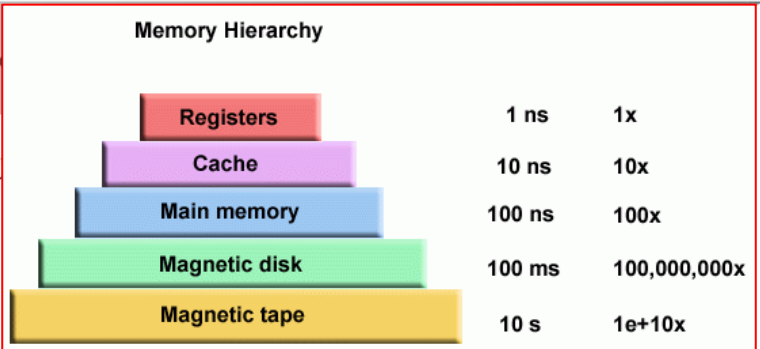
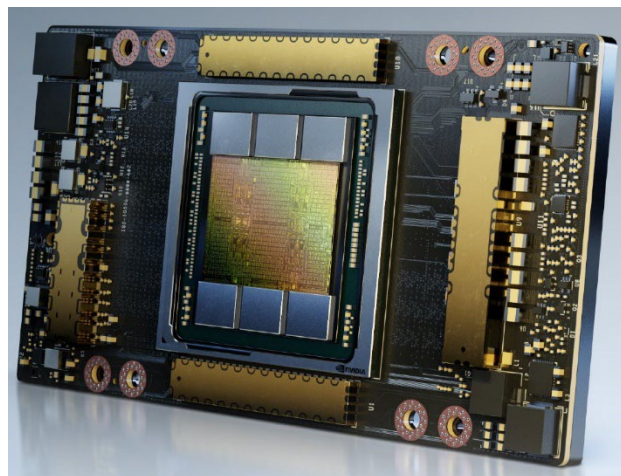


Fig. 2. Memory hierarchies of the experimental CPU



GPU

NVIDIA A100 TENSOR CORE GPU 规格 (SXM4 和 PCIe 外形规格)



	A100 80GB PCIe	A100 80GB SXM
FP64	9.7 TFLOPS	
FP64 Tensor Core	19.5 TFLOPS	
FP32	19.5 TFLOPS	
Tensor Float 32 (TF32)	156 TFLOPS 312 TFLOPS*	
BFLOAT16 Tensor Core	312 TFLOPS 624 TFLOPS*	
FP16 Tensor Core	312 TFLOPS 624 TFLOPS*	
INT8 Tensor Core	624 TOPS 1248 TOPS*	
GPU 显存	80GB HBM2e	80GB HBM2e
GPU 显存带宽	1935GB/s	2039GB/s
最大热设计功耗 (TDP)	300 瓦	400 瓦 ***
互连技术	搭载 2 个 GPU 的 NVIDIA® NVLink® 桥接器 : 600GB/s** PCIe 4.0 : 64GB/s	NVLink : 600GB/s PCIe 4.0 : 64GB/s

一些基本概念

□ 加速比 (Speedup)

串行计算所需时间与并行计算所需时间的比值 $S = \frac{T(1)}{T(p)}$

► 理想情况：线性加速比

□ 效率 (Efficiency) 与开销 (Cost)

$$E = \frac{S}{p}, \quad C = p * T(p)$$

□ 可扩展性 (Scalability)

计算单元增加时并行程序的行为，影响因素：硬件、算法、并行开销、问题特征

并行算法

并行算法

- 适合在并行机上实现的算法
- 好的并行算法应充分发挥并行机计算机的潜在性能

并行算法分类

- 按运算对象：数值并行算法、非数值并行算法
- 按并行进程执行顺序：同步并行算法、异步并行算法、独立并行算法
- 按计算任务：细粒度并行算法、中粒度并行算法、粗粒度并行算法

并行程序设计需考虑的因素

□ 划分 (Partitioning)

将问题分解成可以并行的子问题/任务：数据分解（区域分解）、功能分解

□ 通信 (Communication)

确认各任务之间需要的通信量和通信类型，评估任务划分的合理性

□ 组合 (Agglomeration)

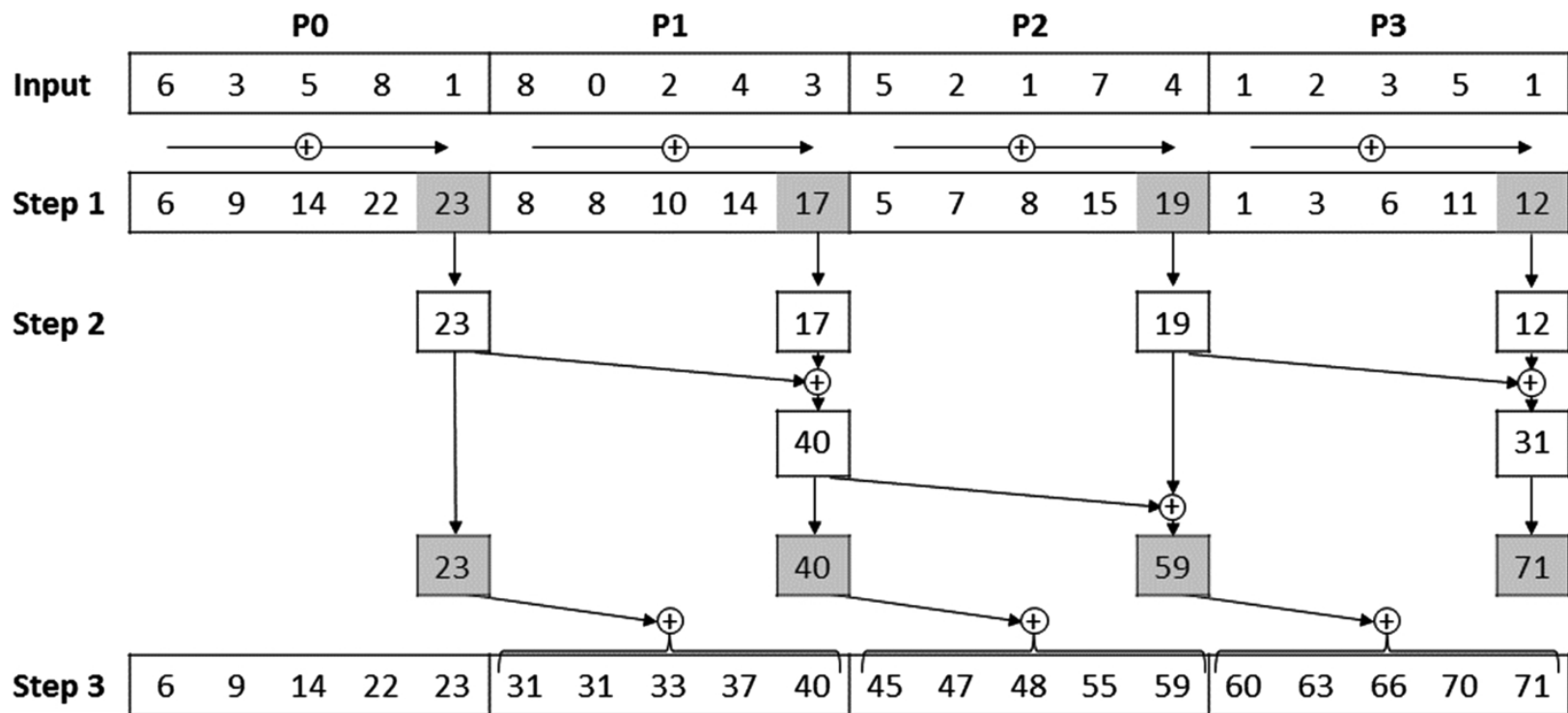
依据任务的局部性，将小任务组合成大任务，减少通信

□ 映射 (Mapping)

将组合后的任务分配到各个进程/线程，力争负载均衡

举例

例：给定一个数组 $a_1, a_2, a_3, \dots$, 计算 $a_i = a_i + a_{i-1}$, $i = 2, 3, \dots$



并行程序设计基本原则

并行算法设计基本原则

- ▶ 与并行机体系结构相结合
- ▶ 以 MIMD 为主 (Multiple Instruction Stream Multiple Data Stream)
- ▶ 具有可扩展性、可移植性
- ▶ 各节点负载均衡
- ▶ 减少节点间的通信
- ▶ 粗粒度并行 (数值计算)
- ▶ 优化性能

并行层次

□ 节点间的并行

基于分布式内存架构：MPI、UPC++

□ 节点内的并行

基于共享内存架构：C++多线程、OpenMP

□ 加速卡级的并行

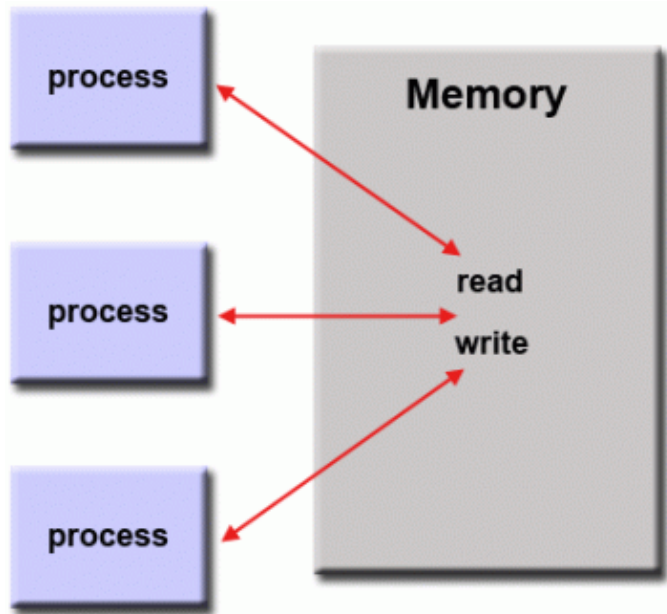
部分任务可分配给加速卡，如 GPU、Xeon Phi 等

† 在当前的超级计算机上编写能够扩展到巨大数量计算核心的代码非常具有挑战性，常常涉及多种语言的混合编程。

并行计算模型

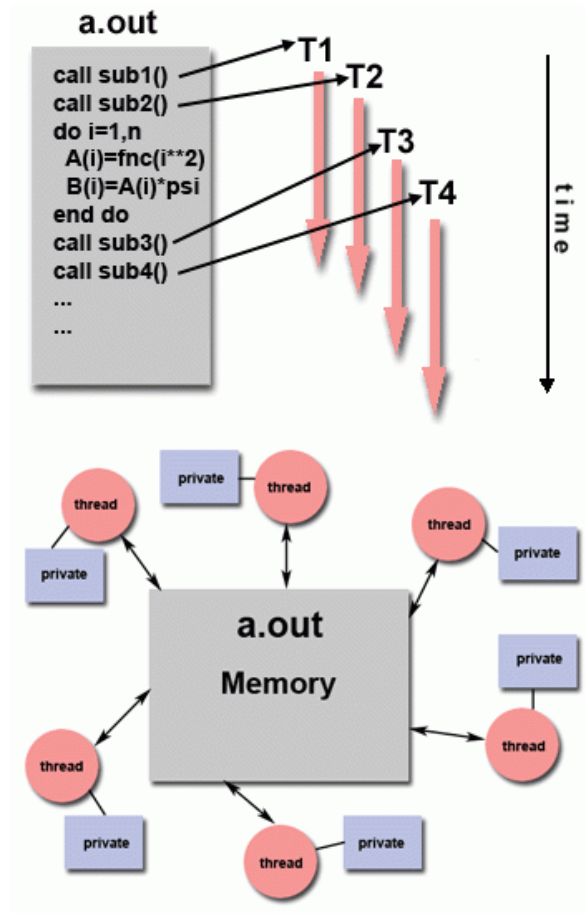
- 共享内存模型（无线程、线程）
- 分布式内存/消息传递模型
- 混合模型
- 数据并行模型
- 单程序多数据模型
- 多程序多数据模型

共享内存模型（无线程）



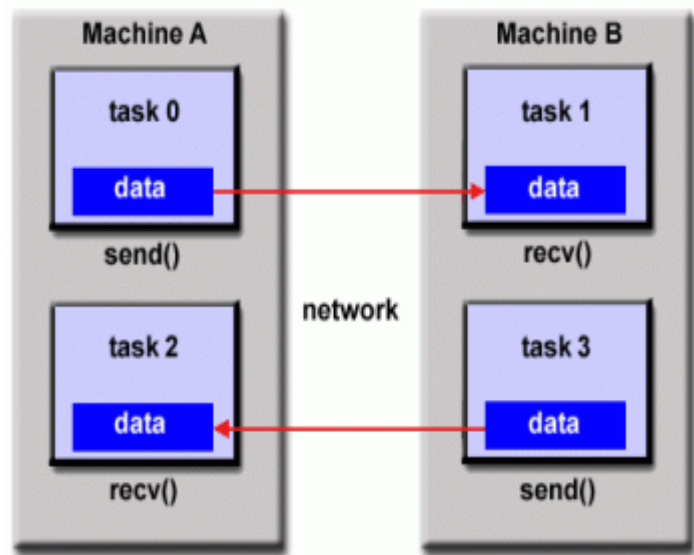
- ▶ 在这种并行计算模型中，处理器/任务**共享内存空间**，可以**异步对内存进行读写**，程序开发比较容易
- ▶ 需设计特定机制控制对内存的存取，解决内存访问冲突以及避免死锁
- ▶ 实现：共享内存单机，本地操作系统，编译器及其对应的硬件都支持共享内存编程

共享内存模型（线程）



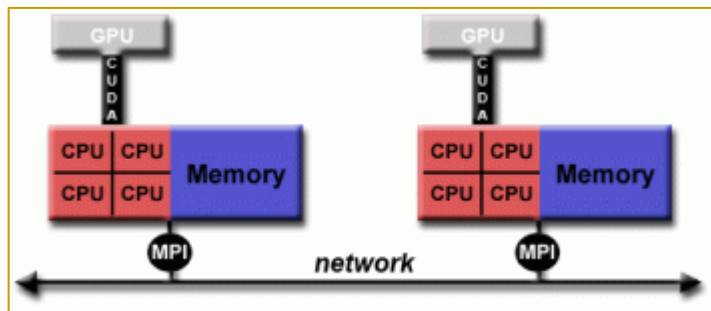
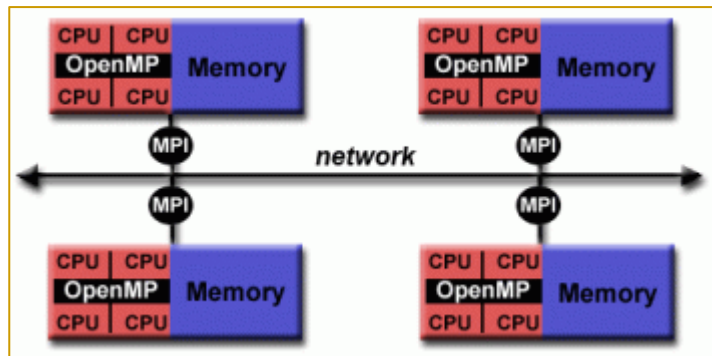
- ▶ 该模型中，有一个主进程，主进程首先执行一些串行工作，然后生成一系列任务（线程），这些线程并发执行。
- ▶ 每个线程具有自己的数据，但同时也共享主进程的所有资源，线程间通过全局内存实现通信。
- ▶ 线程可以随时生成和结束，但主进程一直存在，以提供所需的共享资源，直到整个程序结束。
- ▶ 实现：Pthreads、OpenMP、CUDA threads

分布式内存/消息传递模型



- ▶ 每个任务仅使用自身的本地内存。
- ▶ 任务之间的数据交换是通过发送和接收消息实现。
- ▶ 数据传输通常需要不同进程之间的协同操作才能实现。例如，一个发送操作需要同时对应一个接收操作。
- ▶ 实现：MPI（MPICH、OpenMPI）、PVM

混合并行模型



- ▶ MPI + OpenMP
- ▶ MPI + CUDA
- ▶ MPI + Pthreads
- ▶ MPI + non-GPU accelerators

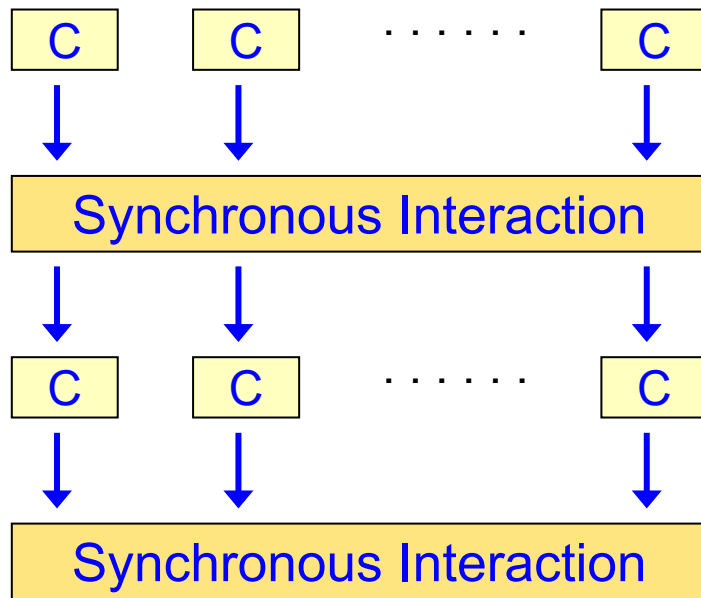
并行化方法

- ❑ 相并行 (Phase Parallel)
- ❑ 流水线并行 (Pipeline Parallel)
- ❑ 主从并行 (Master-Slave Parallel)
- ❑ 分而治之并行 (Divide and Conquer Parallel)
- ❑ 工作池并行 (Work Pool Parallel)

相并行

□ 相并行 (Phase Parallel)

- ▶ 一组超级步 (相)
- ▶ 步内各自计算
- ▶ 步间通信同步
- ▶ 方便查错和性能分析
- ▶ 计算和通信不能重叠



流水线并行

□ 流水线并行 (Pipeline Parallel)

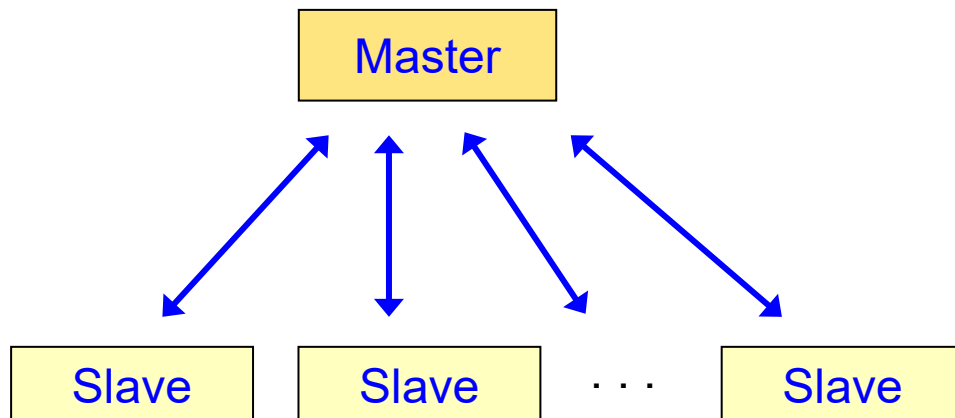
- ▶ 将分成一系列子任务 $t_1, t_2, \dots, t_m$, 一旦 t_1 完成, 后继的子任务就立即开始, 并以同样的速率进行计算
- ▶ 一组进程, 流水线设计技术



主从并行

□ 主从并行 (Master-Slave Parallel)

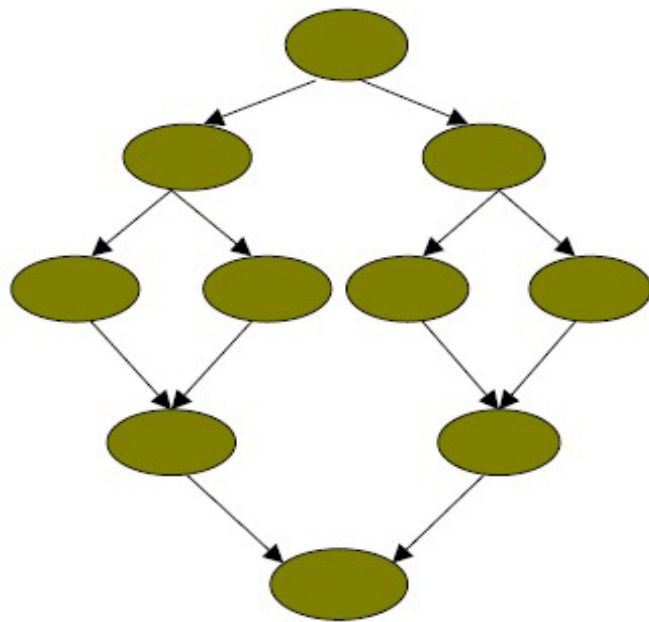
- ▶ 主进程：串行，协调任务
- ▶ 子进程/线程：计算子任务
- ▶ 与相并行结合
- ▶ 主进程易成为瓶颈



分而治之并行

□ 分而治之并行 (Divide and Conquer Parallel)

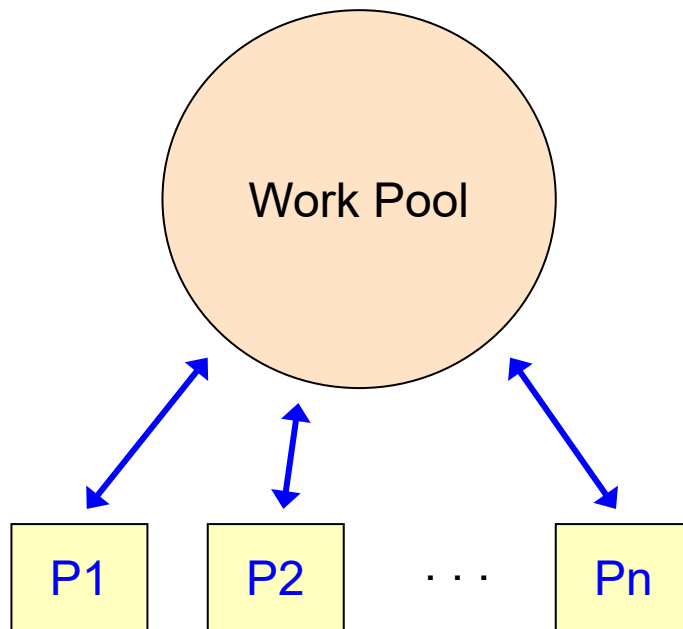
- ▶ 将问题分解成若干特征相同的子问题，分而治之
- ▶ 父进程把负载分割并指派给子进程/线程
- ▶ 重点在于归并
- ▶ 难以负载平衡



工作池并行

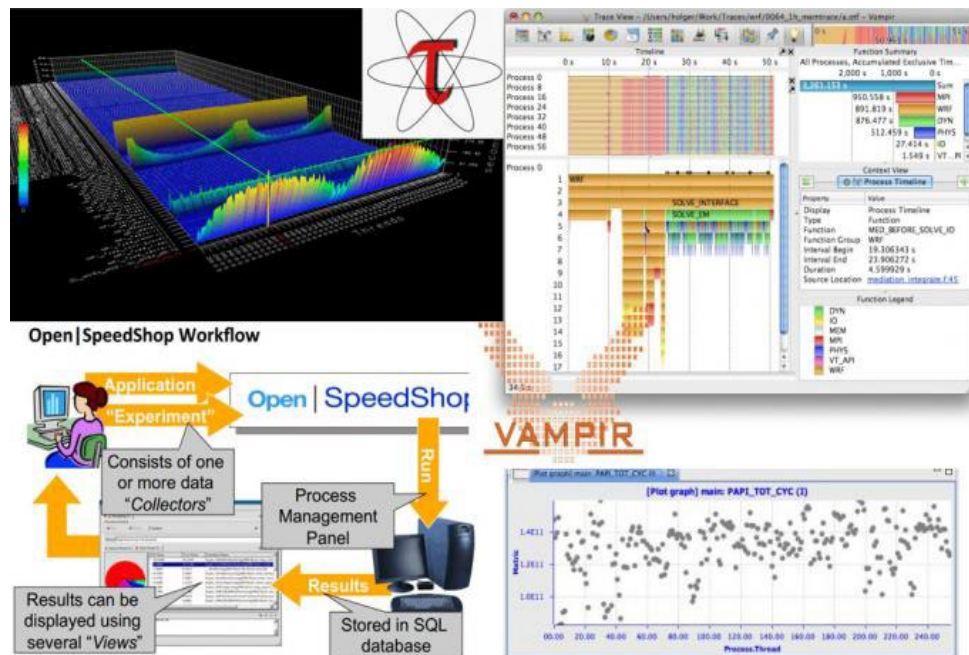
□ 工作池并行 (Work Pool Parallel)

- ▶ 初始状态：一件或多件工作
- ▶ 进程从池中取任务执行
- ▶ 可产生新任务放回池中
- ▶ 直至任务池为空
- ▶ 易于负载均衡



调试与性能分析

□ 并程序的调试和性能分析比串程序更具挑战性



TotalView

Allinea DDT (Distributed Debugging Tool)

Intel Inspector

HPC Toolkit

TAU

Vampir/VampireServer

Open| Speedshop

Intel VTune Amplifier

mpitrace

.....



3

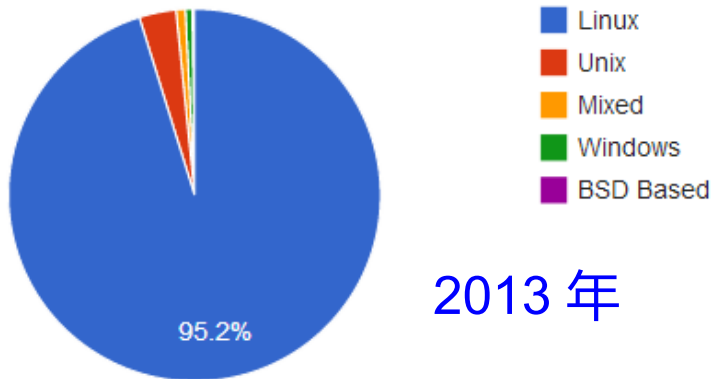
并行编程环境

—— 操作系统/OpenMP/MPI

—— Pthreads/CUDA/UPC++

操作系统 —— Linux

Operating system Family System Share



- ❑ In 2013: 95%
- ❑ In 2014: 97%
- ❑ In 2015: 97.2%
- ❑ In 2016: 99.6%
- ❑ In 2017: 99.6%
- ❑ In 2018: 100%
- ❑ In 2019: 100%
- ❑ In 2020: 100%

Linux might be struggling for a decent desktop market share but it is definitely ruling the world of supercomputers. Linux is the supercomputer operating system by choice.



As per the [latest report from Top 500](#), Linux now runs on all the fastest 500 supercomputers in the world. The previous number was 498 as the remaining two supercomputers ran Unix back in around 2017.

基于共享内存的并行编程环境

OpenMP

基于共享内存的并行编程环境 —— OpenMP



<https://www.openmp.org>

- ❑ 通过在源代码（串行）中添加 OpenMP 指令和调用 OpenMP 库函数来实现在共享内存系统上的并行执行。
- ❑ 为共享内存并行程序员提供了一种简单灵活的开发并行应用的接口模型，使程序既可以在台式机上执行，也可以在超级计算机上执行，具有良好的可移植性。

Jointly defined by a group of major computer hardware and software vendors and major parallel computing user facilities, the OpenMP API is a portable, scalable model that gives shared-memory parallel programmers a simple and flexible interface for developing parallel applications on platforms ranging from embedded systems and accelerator devices to multicore systems and shared-memory systems.

OpenMP 的使用说明

- ❑ FORTRAN/C/C++ 程序库，无需另外安装
- ❑ 编译时不打开 OpenMP 编译选项，则编译器自动忽略 OpenMP 指令，从而生成串行可执行程序（串行等价性）
- ❑ 打开 OpenMP 编译选项，编译器将对 OpenMP 指令进行处理，编译生成并行可执行程序
- ❑ 并行线程数可以根据需要在程序中进行动态设置
- ❑ 支持与 MPI 混合编程

OpenMP 是基于 **线程** 的并行编程模型。

OpenMP 的发展历史

- ❑ 起源于 ANSI X3H5 (1994) 草案, 1997年, 部分设备商和编译器开发商组成 ARB (架构审查委员会), 着手制定 OpenMP 标准化规范
- ❑ 目标: 编程简单, 增量化并行, 移植性好, 可扩展性好, 支持主流编译器
- ❑ 支持 Unix, Linux, Windows 等操作系统
- ❑ ARB 成员: AMD, ARM, Intel, IBM, Cray, NEC, HP, NVIDIA,

OpenMP

OpenMP
Application Programming
Interface

Version 5.2 November 2021

- ▶ FORTRAN version 1.0 (1997), C/C++ version 1.0 (1998)
- ▶ FORTRAN version 2.0 (2000), C/C++ version 2.0 (2002)
- ▶ OpenMP 3.0 – (May 2008)
- ▶ OpenMP 4.0 – (July 2013)
- ▶ OpenMP 4.5 – (Nov 2015)
- ▶ OpenMP 5.0 – (Nov 2018)
- ▶ OpenMP 5.2 – (Nov 2021)

基于分布式内存的并行编程环境

消息传递编程接口MPI

基于消息传递的并行编程环境 —— MPI

MPI

Message Passing Interface

<https://www.mpi-forum.org>

- 消息传递编程标准，提供一个高效、可扩展、统一的并行编程环境，目前最为通用的分布式并行编程方式。
- MPI 是一种消息传递编程模型，是一种标准或规范，MPI 实现通过提供库函数实现进程间通信，从而进行并行计算，目前所有并行机制造商都提供对MPI 的支持。

The advantages of developing message passing software using MPI closely match the design goals of **portability**, **efficiency**, and **flexibility**.

MPI is not an IEEE or ISO standard, but has in fact, become the “**industry standard**” for writing message passing programs on HPC platforms.

OpenMP 的发展历史

- ❑ 1994 年公布 MPI-1.0, 当前版本 MPI-4.0 (2021)
 - ❑ 支持 C 和 Fortran
 - ❑ MPI 实现: MPICH, OpenMPI
 - ❑ 商业实现: Intel MPI, IBM BG/Q MPI, HP-MPI, MS-MPI,
 - ❑ 所有版本都遵循 MPI 标准, 可以不加修改地运行
-
- ▶ MPI-1.0 (1994), MPI-1.1 (1995), MPI-1.2 (1997), MPI-1.3 (2008)
 - ▶ MPI-2.0 (1997), MPI-2.1 (2008), MPI-2.2 (2009)
 - ▶ MPI-3.0 (2012), MPI-3.1 (2015)
 - ▶ MPI-4.0 (2021)

MPI 是基于 进程 的并行编程模型。

谢谢
THANK YOU

