

---

# 并行计算

## 第一讲：并行计算概述

中国科学院计算技术研究所  
高性能计算机研究中心  
邵恩

# 授课教师

---

- 邵恩，计算所高性能计算机研究中心



- 杨帆，计算所高性能计算机研究中心



- 肖俊敏，计算所高性能计算机研究中心

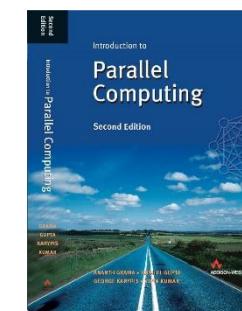


# 参考书籍

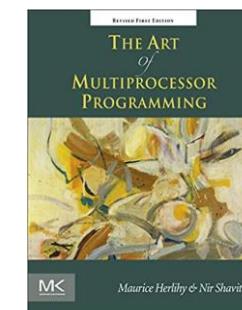
- **David E. Culler, Jaswinder Pal Singh.**  
Parallel Computer Architecture:  
A Hardware/Software Approach (有中文版)



- **Gramma, A., Gupta, A., Kumar V.**  
Introduction to Parallel Computing



- **Maurice Herlihy, Nir Shavit.**  
The Art of Multiprocessor Programming



# 一些预备课程或知识

---

- 编译原理
- 计算机系统结构
- 操作系统（Linux）
- 算法导论

# Exam & Project

---

- 课程周期: **11月29日~1月11日**
  - 周三 18:10 - 19:30 - 21:00
  - 周四 8:30 - 9:40 - 11:20
- 考勤 – **30%**
- 考试 – **70%**
  - 以课程总结的方式: 比如结合实验课的实践或者自己科研项目制作报告演讲

# 报告要求

---

## ■ 形式

- PPT展示
- 每人报告**10分钟**, 提问**3分钟**
- 使用排练计时, 不允许超时

## ■ 内容

- 展示个人对并行计算的理解和思考
- 选题必须在并行计算课程讲解的内容范围内
- 可以选择**2018年以后**（包含**2018年**）**CCF**列表中**A、B**类期刊或会议论文进行讨论
- 也可以结合自己的工程或项目经历论述
- 讲清研究背景、相关工作、主要思路或创新点、实现方法、评测结果、结论或评价

## ■ 评分

- 报告选题紧扣课程内容, 话题新颖
- 报告时间安排合理
- 思路清晰, 观点明确
- 能清晰回答听众的问题, 也善于提问

# Outline

---

- ① 计算机中无处不在的并行 Including your laptops and handhelds
- ② 工程与科学计算需要高性能计算机 Commercial problems too
- ③ 高性能计算机（超级计算机）的发展历史
- ④ 如何在高性能计算机上编程（难写的并行程序）  
But things are improving

---

# 第一部分： 计算机（计算设备）中 无处不在的并行

# 日常生活中随处可见的并行设备

海光高性能服务器



- 2颗海光7000处理器
- 每处理器16~32核心

龙芯台式机



- 1颗龙芯3A 4000处理器
- 每处理器4核心

飞腾笔记本



- 1颗飞腾2000处理器
- 每处理器4核心

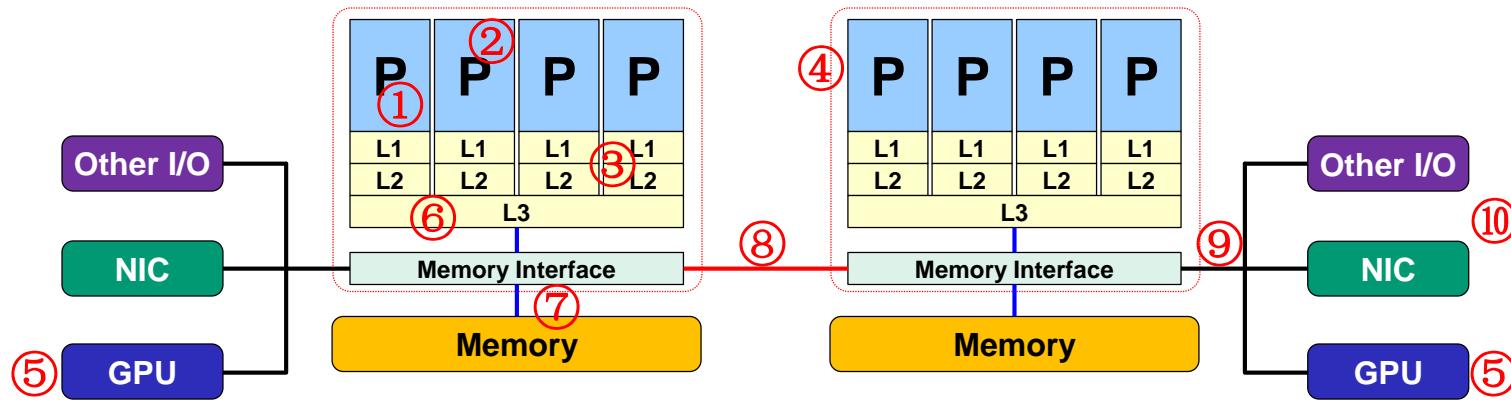
华为手机



- 1颗麒麟9000处理器
- 每处理器8个ARM核心、24个Mali核心、数个NPU核心

# 从计算机系统结构角度看并行

- 一个双路服务器内的资源并行和共享（**why mentioned**）



并行资源:

- ① 执行单元: e.g. SIMD units
- ② 处理器核 cores
- ③ 核内的cache
- ④ 不同的处理器 (Sockets)
- ⑤ 不同的加速器 (GPU)

共享的资源:

- ⑥ 每个处理器 (Socket) 内的核间Cache
- ⑦ 每个处理器 (Socket) 的内存总线
- ⑧ 处理器间 (Intersocket) 的总线
- ⑨ I/O总线 e.g. PCIe
- ⑩ 互连网络 或 其他的I/O资源

思考题: 应用程序是如何使用这些并行资源和共享资源的?

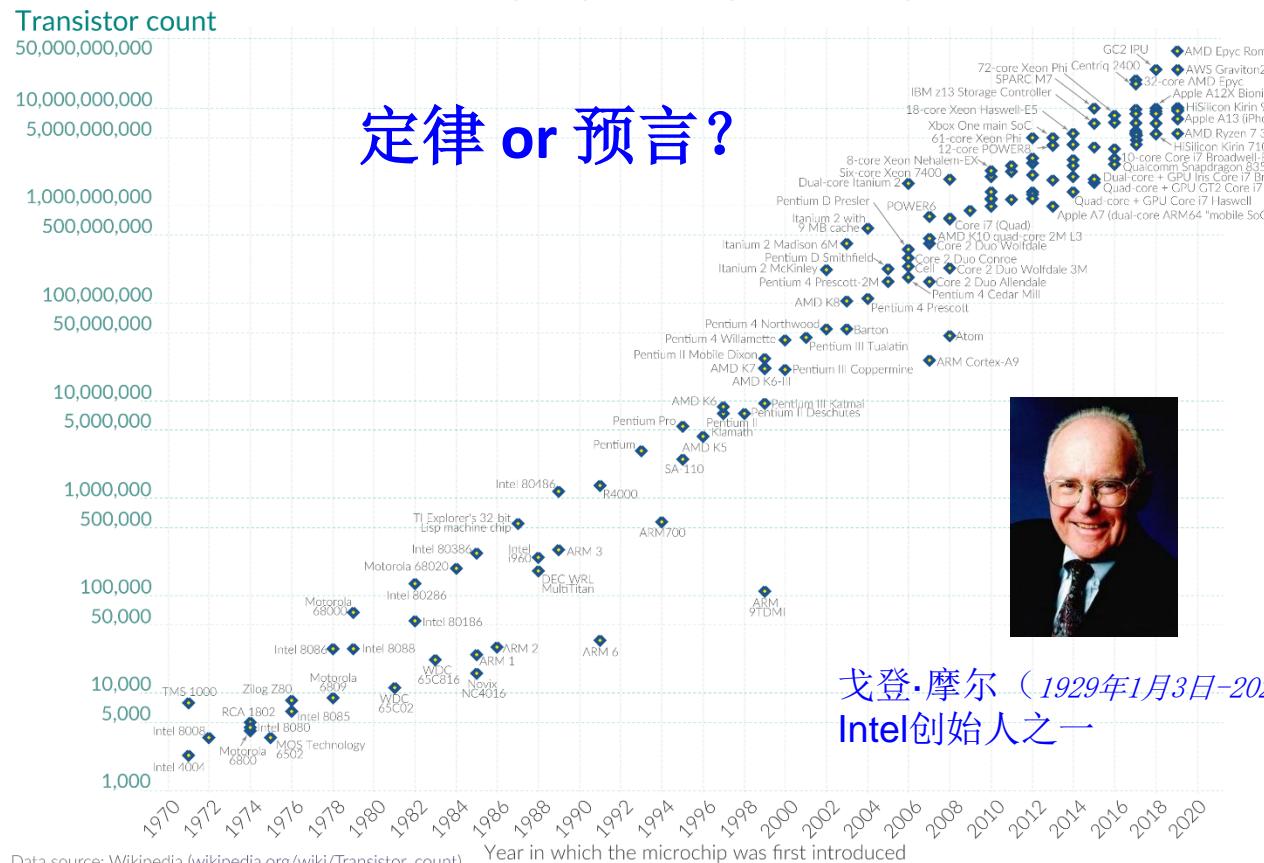
# 为什么会产生并行？

- 并行→性能→摩尔定律
- 摩尔定律：半导体芯片的晶体管密度大约每18个月翻一番

Moore's Law: The number of transistors on microchips doubles every two years

Moore's law describes the empirical regularity that the number of transistors on integrated circuits doubles approximately every two years. This advancement is important for other aspects of technological progress in computing – such as processing speed or the price of computers.

Our World  
in Data

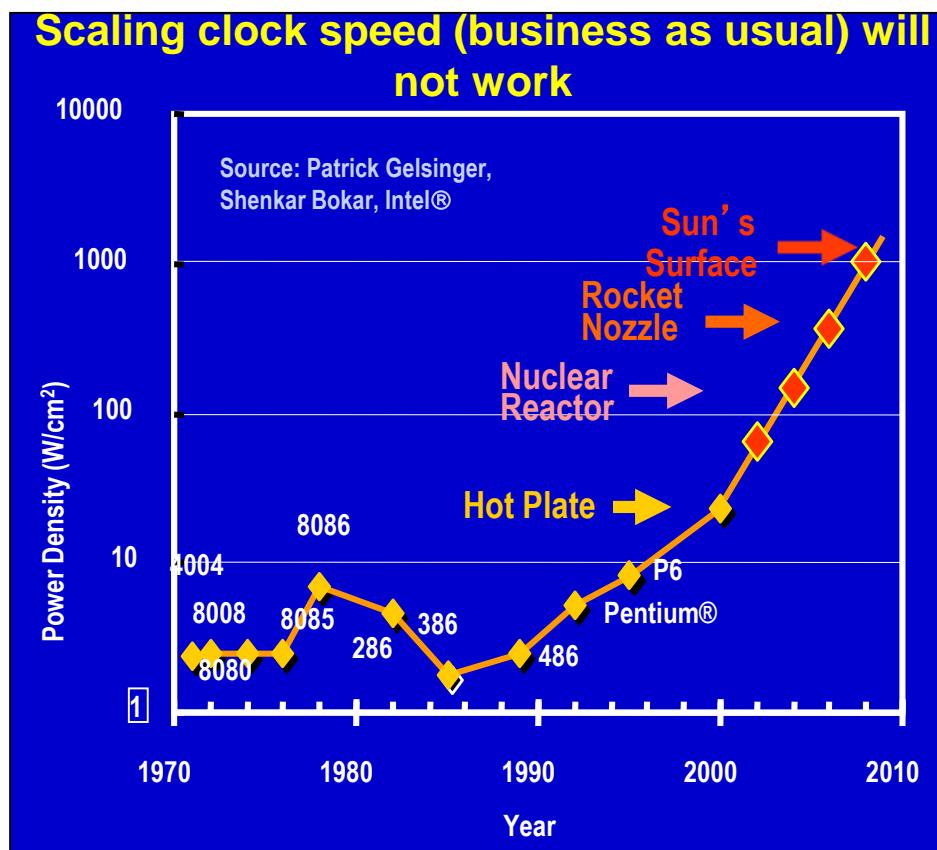


# 摩尔定律下，器件尺寸缩小带来的影响

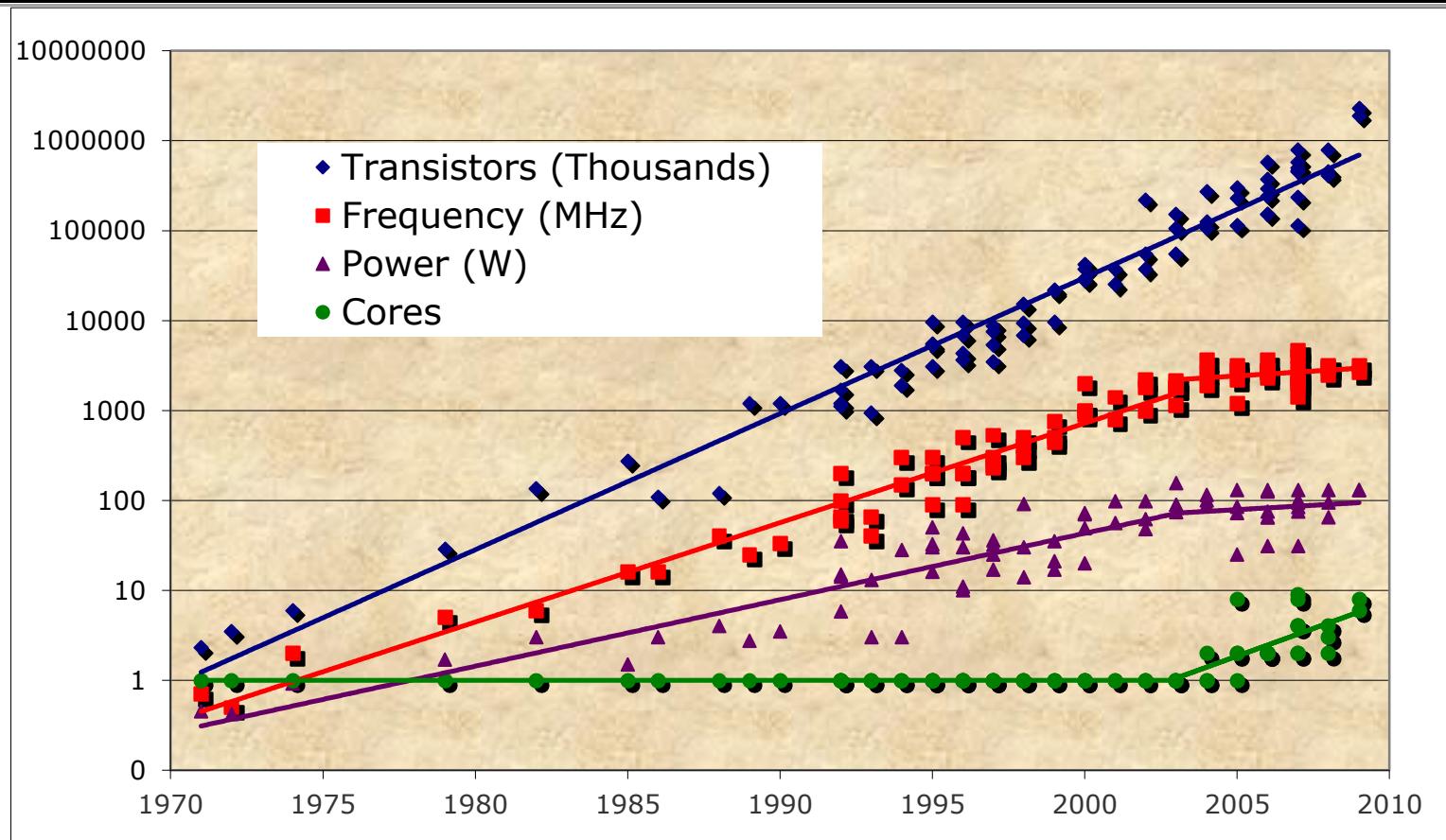
- 器件特征尺寸(transistor size) 缩小 $\times$ 倍?
- 晶体管的时钟速率可以提高 $\times$ 倍
  - 通常由于能量消耗，会小于 $\times$ 倍
- 每单位面积的晶体管数量提高 $\times^2$ 倍
- 裸片尺寸也会增加
  - 系数是与 $\times$ 相关的另一个因子
- 芯片的理论性能提升 $\sim \times^4$ 倍!
  - 其中，大约 $\times^3$ 倍的提升用于或者来源于
    - parallelism: e.g. Instruction-level parallelism (ILP) (既消耗晶体管资源，也需要频率提升)
    - locality: caches
- 理论上来说，原有程序不需要改动，就可以获得 $\times^3$ 倍以上的性能提升

# 功率密度限制串行性能

- 并行技术更加省电
  - 晶体管动态功耗正比于 $V^2fC$
  - 提高频率( $f$ )也会提高电压( $V$ ) → 产生立方效应(cubic effect)
  - 增加核的数量只增加电容( $C$ )仅仅是线性增长
  - 因此通过降低时钟频率来节省功耗效果比较明显
- **提高串行处理器的技术有能力天花板**
  - 多发射, 分支预测 (为了降低失败代价、提高频率), 动态依赖检查, etc.
- 因此, 在摩尔定律支持下, 仅仅依赖串行技术不足以获得足够的性能收益



# 因此，我们看一下40年处理器的演进



- 芯片密度（晶体管数量）~2x 每两年
- 时钟频率提升的速度较缓
- 逐步在用处理器核数增加来提升性能
- 处理器的功耗逐步得到了控制

# 重新审视摩尔定律

---

- 芯片内核的数量每两年翻一番
- 芯片的频率将不再增加(甚至降低)
- 编程需要应对需要具有数百万并发线程的系统
- 编程还需要同时应对芯片内并行和芯片间并行

# Parallelism and Concurrency (名词解释)

## Concurrency 并发

“Concurrency occurs when two or more execution flows are able to run simultaneously.” 当两个或多个**执行流**能够**同时**运行时称之为并发

A **property** of execution flows 程序执行流的一种**属性**

## Parallelism 并行度

“The maximum number of independent subtasks in a given task at a given point in its execution.” 给定任务、在给定执行时间点的**最大独立子任务数**

A **state** of execution flows 程序执行流的一个**状态**

Parallel Computing (并行计算) → Division of tasks in Parallel Computers and side effects (i.e. Memory consistency) 单任务分布到并行计算机上执行

Concurrent Computing (并发计算) → Interactions between tasks in a concurrent system (i.e. Signal Handling) 多个任务之间交互计算

# 在计算机系统中具体并行的类型



# 作业级并行 (Job Level Parallelism)

协调器: OS, Programmer

将作业执行与CPU的其他活动叠加起来

作业间并行:

“处理”冗长的I/O请求时在作业之间切换

	Time 1	Time 2
CPU	Job 1	Job 2
I/O	Job 2	Job 1

作业内并行:

在等待DMA传输时进行作业内其他部分的计算

要求:重复的资源

# 任务级并行 (Task Level Parallelism)

协调器: Compiler, Programmer, OS

Task Level == Program Level:

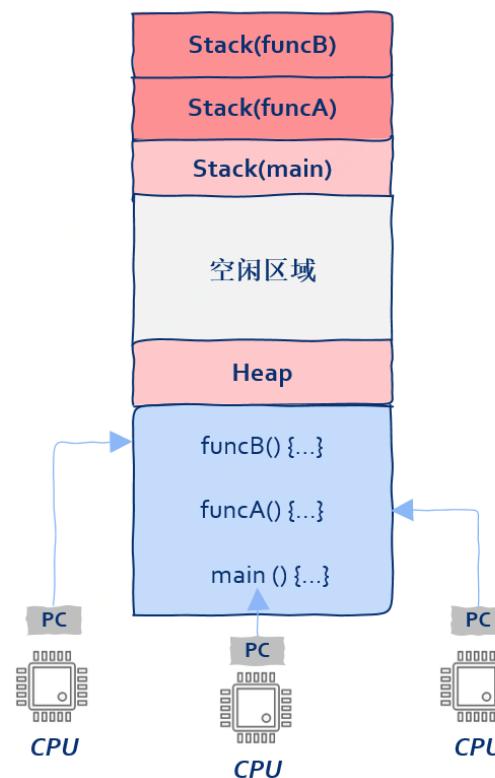
- 在不同的代码段之间并行  
函数调用或其他形式代码抽象 (like code blocks)
- 同一循环的不同迭代之间并行
- 要处理好数据依赖和程序分区

# 线程级并行 (Thread Level Parallelism)

## Thread

- 具有独立 PC (程序计数器) 的一段指令序列;
- 在不同的处理器 (或者核上) 中运行程序的不同部分以充分利用并行性;

## 协调器: Programmer or Compiler

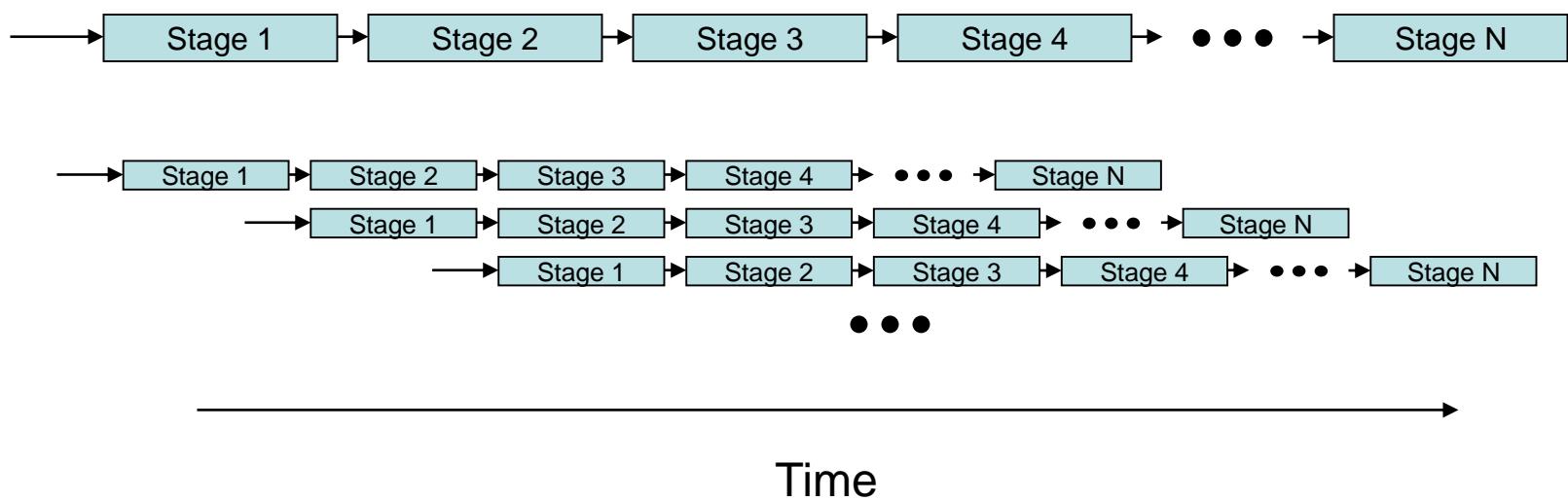


# 指令级并行 (Instruction Level Parallelism)

## ■ 指令之间并行

- 将不同的指令在不同硬件资源上执行
- 前提:硬件资源不止一种或一份 (ALU, Adder, Multiplier, Loader, MAC unit, etc)
- 指令之间不存在数据依赖

## ■ 指令执行的不同阶段并行, Examples: Instruction Pipelines



# 两类大规模并行系统比较

高性能计算机（超算）	数据中心
目标应用	
■ Few, big tasks	■ Many small tasks
系统硬件	
■ Customized ■ Optimized for reliability ■ Low latency interconnect	■ Consumer grade ■ Optimized for low cost ■ Throughput-optimized interconnect
系统软件	
■ Minimal ■ Static scheduling	■ Provides reliability ■ Dynamic allocation
编程	
■ Low-level, processor-centric model ■ Programmer manages resources	■ High level, data-centric model ■ Let run-time system manage resources

# 两类系统逐渐产生了重叠

---

## ■ 高性能计算机的技术应用于数据中心

- Data center computers sometimes used to solve problem
  - E.g., learn neural network for language translation
- Data center computers sometimes equipped with GPUs
- Data center network uses RDMA technology

## ■ 数据中心的特征出现在高性能计算机中

- Also used to process many small–medium jobs with docker

# 高性能计算机是不可或缺的

- 在任何给定时间，高性能计算机都是解决科学和工程问题的最强大的计算机
- 在任何给定时间，高性能计算机都是解决各类问题（包括AI）的最强大的计算机，也是技术最先进的计算机



---

## 第二部分： 工程与科学计算需要 高性能计算机

# 计算机科学对传统科学产生影响

“An important development in sciences is occurring at the intersection of computer science and the sciences that has the potential to have a profound impact on science. It is a leap from the application of computing ... to the *integration of computer science concepts, tools, and theorems* into the very fabric of science.”

-*Science 2020 Report, March 2006*

将计算机科学的概念、工具、理论整合进传统自然科学框架中，将使科学产生飞跃式进展



Nature, March 23, 2006



# 从科学研究的范式（Paradigm）说起

范式是常规科学所赖以运作的理论基础和实践规范，是从事某一科学的科学家群体所共同遵从的世界观和行为方式



托马斯·库恩  
(Thomas Samuel Kuhn)



# 科学研究第一范式：实验科学

描述记录自然现象，基于实验或经验的归纳为主



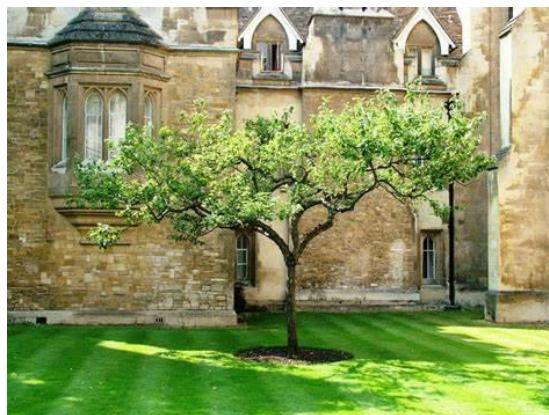
钻木取火



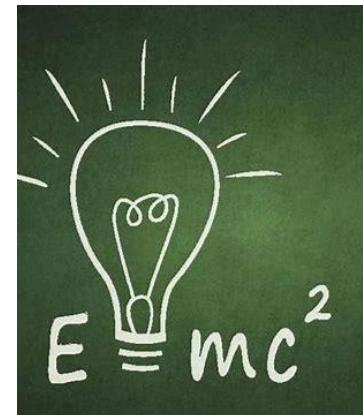
比萨斜塔实验

# 科学研究第二范式：理论科学

在自然现象基础上进行了抽象简化，以构建数学模型为主



牛顿力学



相对论

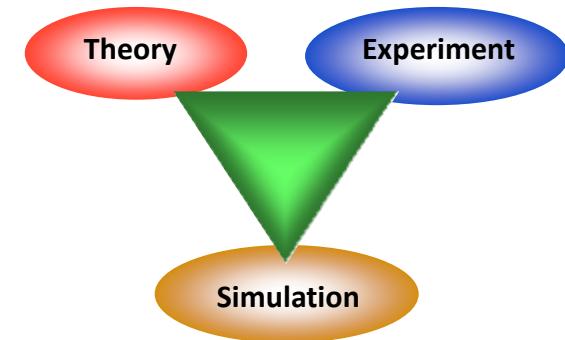
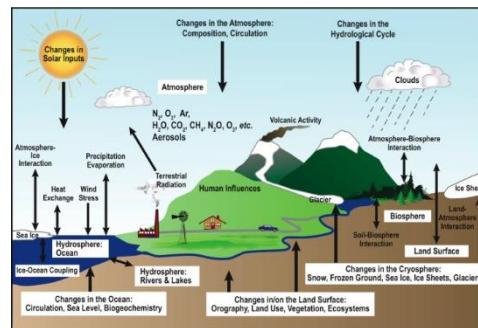
# 科学研究的第三范式：计算科学

## ■ 传统方法的局限性：

- –Too difficult—例如：建造大型风洞
- –Too expensive—例如：建造一架一次性客机
- –Too slow—例如：等待气候或银河系演化
- –Too dangerous—例如：武器、药物设计、气候实验

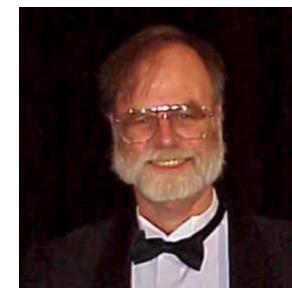
## ■ 计算科学范式：

- 利用计算机对现象进行模拟分析
- 基于已知的物理定律和高效的数值方法
- 使用手动无法实现的计算工具和方法分析仿真结果

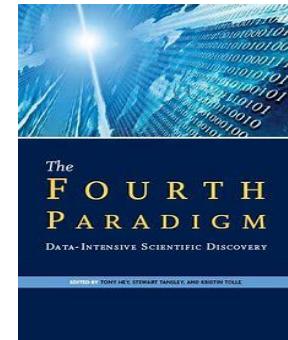


# 科学研究的第四范式：数据科学

- 实验数据持续指数增长
  - 摩尔定律也适用于传感器
  - 需要分析所有这些数据
  - **气候建模**: 政府间气候变化专门委员会 (IPCC) 预估一次的数据是10s of petabytes
  - **基因组**: 美国能源部联合基因组研究中心 (JGI) 目前有0.5 petabyte data, 每年增长一倍
  - **粒子物理学**: 欧洲大型强子对撞机 (LHC) 每年产生16 petabytes
  - **天体物理学**: 大型综合巡天望远镜 (LSST) 及其他类似装置每年产生 5 petabytes
- **数据科学**: 基于数据密集型的相关关系分析



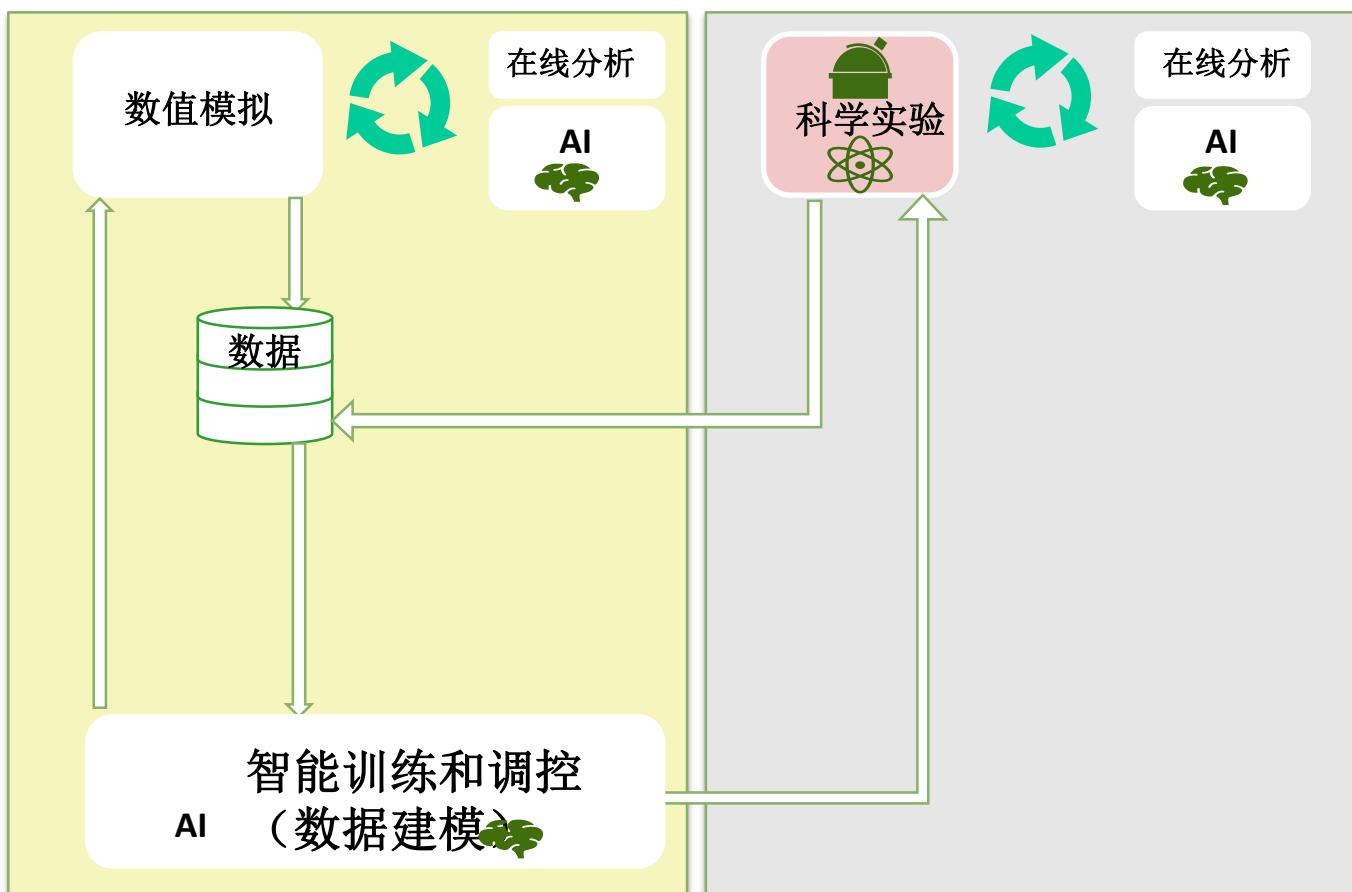
*Jim Gray*



*Microsoft Research*

# 科学研究的第五范式：智能科学

利用AI技术实现更大规模的建模，并实现理实交融



# 各行各业都有对计算特别具有挑战性的应用

## ■ 自然科学领域 (Science)

- 全球气候模拟 Global climate modeling (天)
- 生物方向 (Biology) : 基因组学 genomics; 蛋白质折叠 protein folding; 药物设计 drug design (人)
- 天体物理建模 Astrophysical modeling (天)
- 计算化学 Computational Chemistry
- 计算材料科学和纳米科学 Computational Material Sciences and Nanosciences

## ■ 工程领域 (Engineering)

- 半导体设计 Semiconductor design
- 地震建模模拟 Earthquake and structural modeling (地)
- 计算流体力学 Computation fluid dynamics (airplane design)
- 燃烧建模计算 Combustion (engine design)
- 碰撞模拟 Crash simulation

## ■ 商业领域 (Business)

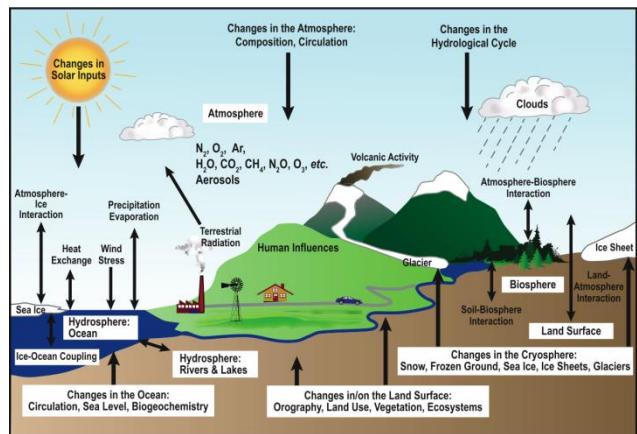
- 金融和经济建模 Financial and economic modeling (人)

## ■ 国防领域 (Defense)

- 核武器模拟 Nuclear weapons -- test by simulations
- 密码学 Cryptography

算天  
算地  
算人

# 科学与工程计算应用举例：全球气候模拟



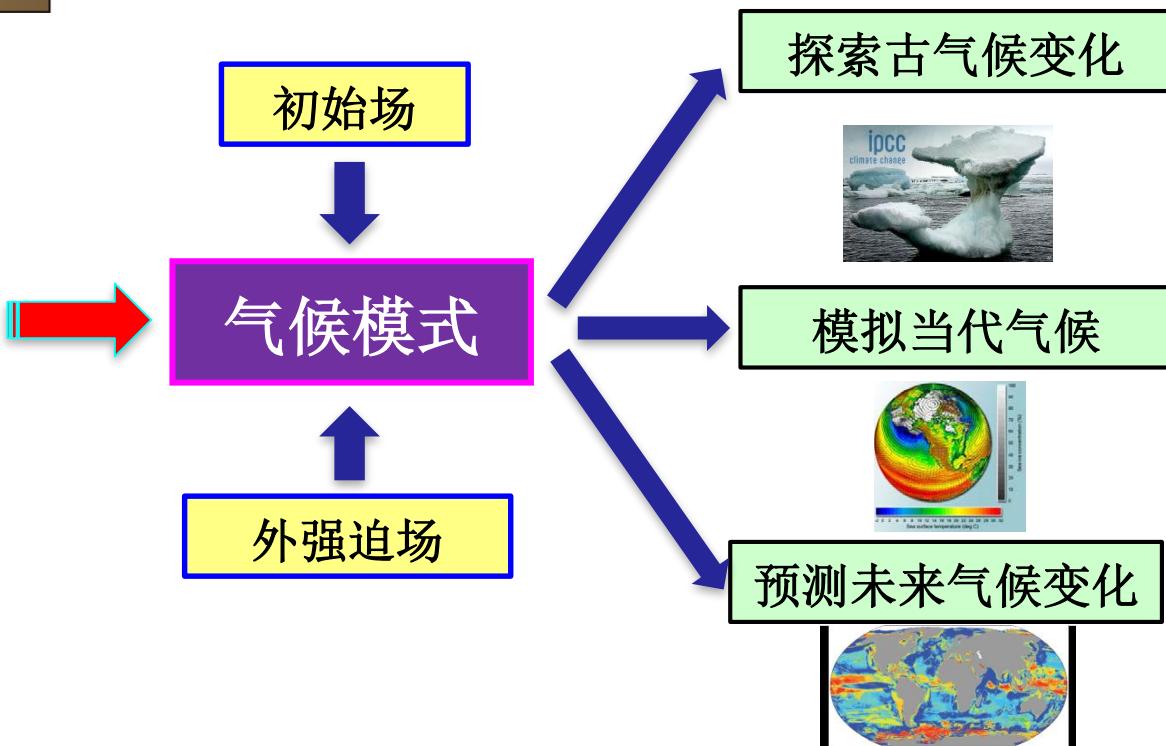
基于气候系统中的  
动力、物理、化学  
和生物过程建立数  
学方程组

问题形式化描述：

$f(\text{纬度}, \text{精度}, \text{海拔}, \text{时间}) \rightarrow \text{"weather"} = (\text{温度}, \text{压力}, \text{湿度}, \text{风速})$

解法：

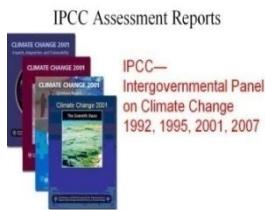
- 离散化域，例如，每 10 公里一个测量点
- 给定  $t$ ，设计一个算法来预测时间  $t+dt$  的天气



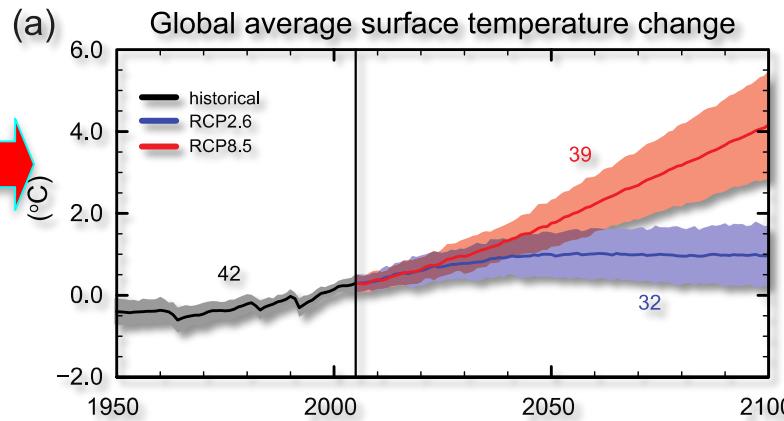
# 全球气候模拟的作用举例

## 参与IPCC的模式

Model name	
ACCESS1.0, ACCESS1.3	Australia
BCC-CSM1.1, BCC-CSM1.1(m)	China
BNU-ESM	China
CanCM4	Canada
CanESM2	
CCSM4	
CESM1 (BGC)	USA
CESM1 (WACCM)	
CESM1 (FASTCHEM)	
CESM1 (CAM5)	USA
CESM1 (CAM5.1-FV2)	
CMCC-CM, CMCC-CMS	Italy
CMCC-CESM	
CNRM-CM5	France
CSIRO-Mk3.6.0	Australia
EC-EARTH	Europe
FGOALS-g2	China
FGOALS-s2	China
FIO-ESM v1.0	China
GFDL-ESM2M, GFDL-ESM2G	
GFDL-CM2.1	USA
GFDL-CM3	
GISS-E2-R, GISS-E2-H	USA
GISS-E2-R-CC, GISS-E2-H-CC	
HadGEM2-ES	
HadGEM2-CC	UK
HadCM3	
HadGEM2-AO	Korea
INM-CM4	Russia
IPSL-CM5A-LR / -CM5A-MR / -CM5B-LR	France
MIROC4h, MIROC5	Japan
MIROC-ESM	
MIROC-ESM-CHEM	
MPI-ESM-LR / -ESM-MR / -ESM-P	Germany
MRI-ESM1	Japan
MRI-CGCM3	
NCEP-CFSv2	USA
NorESM1-M	Norway
NorESM1-ME	



IPCC（政府间气候变化专门委员会，2007年诺贝尔和平奖）给出的未来增温幅度完全是由参与IPCC评估的模式结果得出的。而全球气候变化谈判依据的正是这个结果。



只有努力发展我国自己的气候系统模式，提高模式模拟能力，才能增强我国在全球气候变化谈判中的话语权。

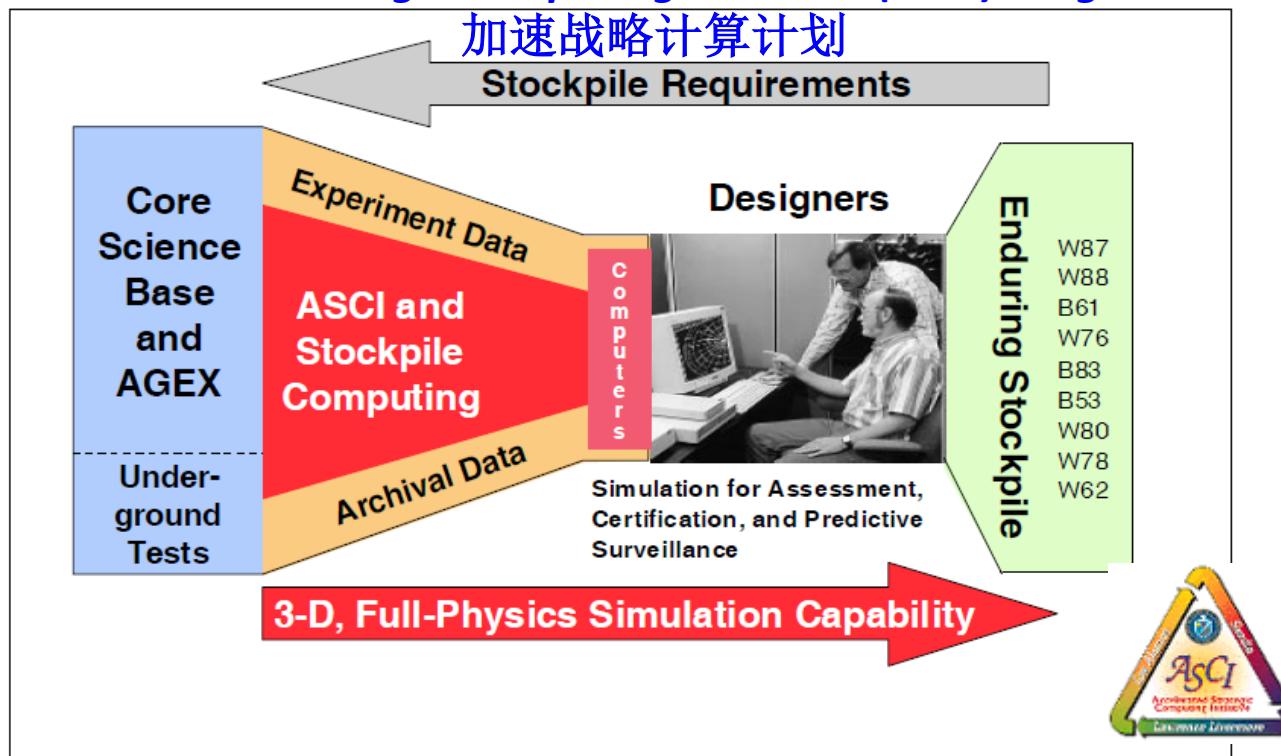
# 全球气候模拟的计算需求

- 模拟大气中的流体流动是其中一项内容
  - 求解Navier-Stokes方程（纳维尔斯托克斯方程，NS方程）
  - 每个网格点大约 100 Flops 的计算能力，模拟时间步长为 1 分钟
  - Computational requirements:
    - 天气预报(7 days in 24 hours) → 56 Gflop/s
    - 气候预测(50 years in 30 days) → 4.8 Tflop/s
    - 用于气候政策谈判(50 years in 12 hours) → 288 Tflop/s
- 将网格分辨率加倍，计算将变为原来的 **8x to 16x**
- 最先进的模型还需要整合大气、云、海洋、海冰、陆地模型，以及碳循环、地球化学等，目前的模型还比较粗糙，因此需要计算机算力进一步提升

# 科学与工程计算应用举例：核模拟

- 持久储备（Enduring Stockpile）是美国在冷战结束后的核武器库
- 《全面禁止核试验条约》1996年要求缔约国承诺：不进行、导致、鼓励或以任何方式参与进行任何核武器试验爆炸或任何其他核爆炸

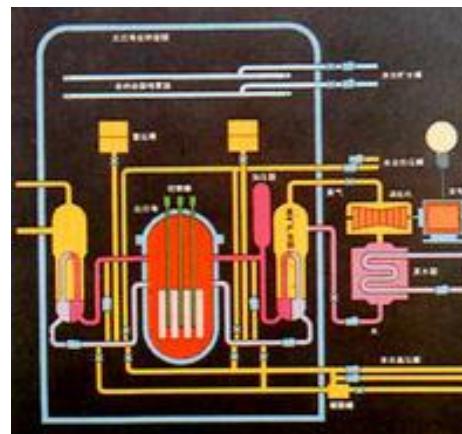
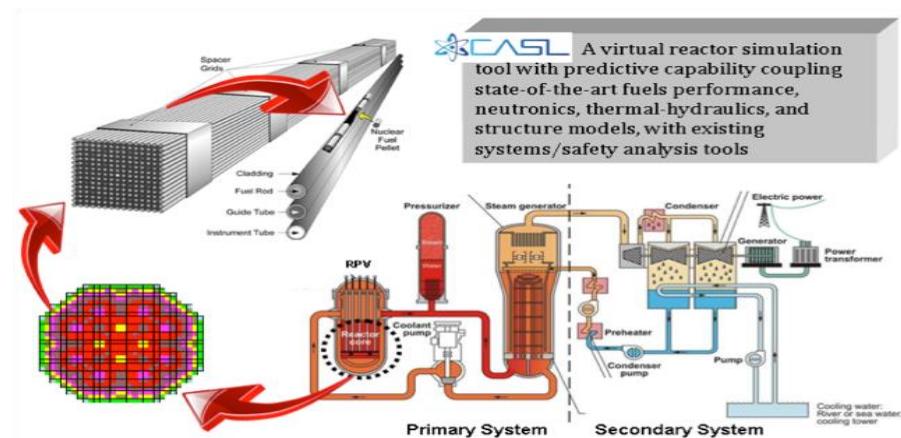
## Accelerated Strategic Computing Initiative (ASCI) Program Plan



**Figure 1.** Simulation, via ASCI, is the only practical, credible path from the known to the unknown for designers to provide an understanding of the enduring stockpile.

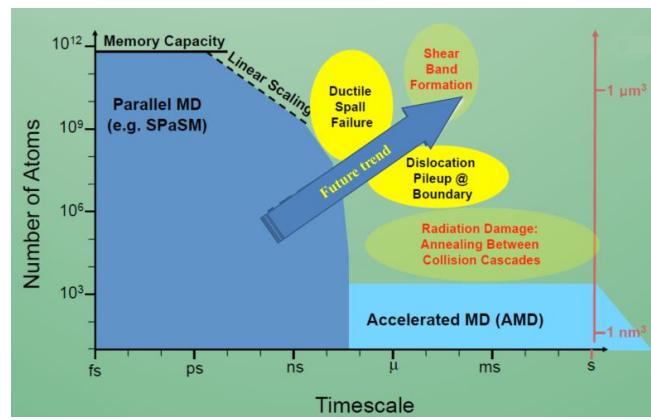
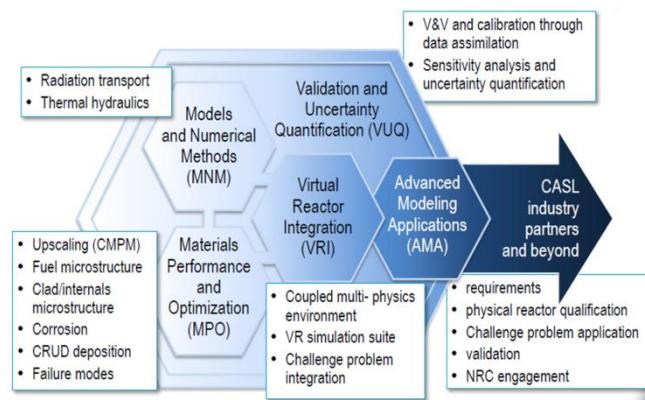
# 核电站的安全性和经济性

- 世界上接近**20%**的电力来自核电。核电的安全性和经济性是政府、业主和公众重点关注的问题。
- 在保证安全的前提下，实现高的组件卸料燃耗（长的燃料循环）、长的核电站使用寿命是核电追求的目标。
- 核电站核岛（一回路）设备在高温、高压、强辐射、冷却剂腐蚀的苛刻环境中工作，设备在服役过程中的性能降级直接关系到核电站的安全性和经济性。
- 核电站关键设备材料（组件、压力容器、堆内构件等）可靠性是保障核电站安全经济运行的基础。

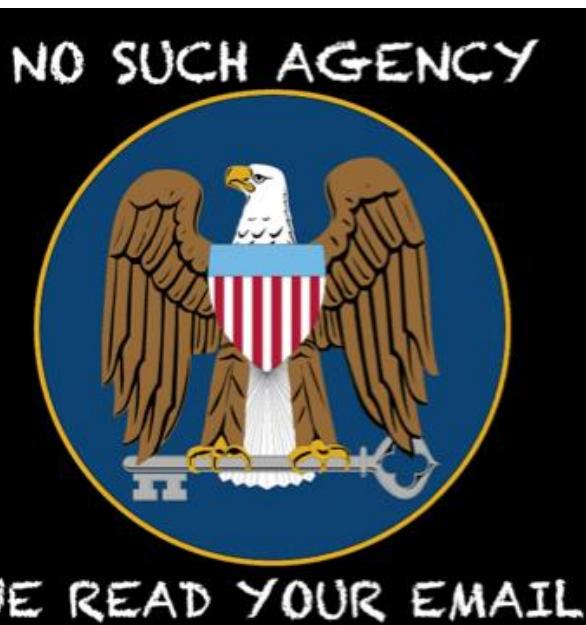


# 核模拟的主要方法：数值反应堆

- 美国、欧洲等国家试图从材料的基本理论出发，结合实验数据和运行经验，运用当代超强的计算机计算能力，通过模拟仿真，模拟核电站运行及部件的性能演化，进行验证和评估（V&V），预测核电站关键设备和材料的使用性能和寿命，提出数值反应堆概念（美国称之为VERA， virtual environment for reactor application）。
- 美国的CASL计划，美欧的PVR计划等，重点关注燃料组件的CILC、GTRF、PCI和压力容器和堆内构件的辐照脆化等挑战性问题。目前这些计划已经取得阶段性成果。
- CASL针对现有轻水堆延寿、升级等任务，利用超级计算机来研究轻水堆性能，开发反应堆分析数值模拟环境（VERA），通过高精度的模型来进行仿真模拟。



# 科学与工程计算应用举例：密码学（信息安全）



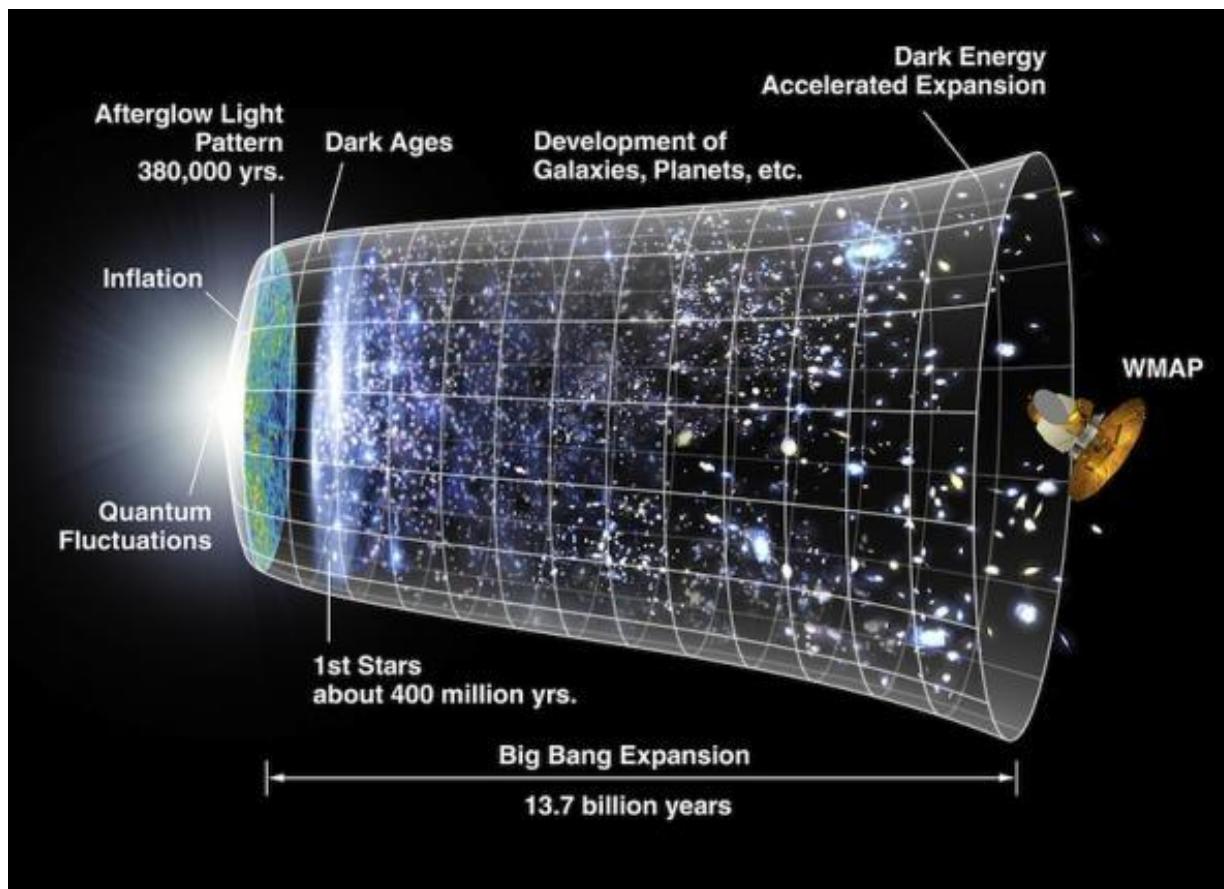
# 众所周知的棱镜计划

棱镜计划（PRISM）是一项由美国国家安全局（NSA）自2007年小布什时期起开始实施的绝密电子监听计划，该计划的正式名号为“US-984XN”。直接进入美国网际网络公司的中心服务器里挖掘数据、收集情报，包括微软、雅虎、谷歌、苹果等在内的9家国际网络巨头皆参与其中。



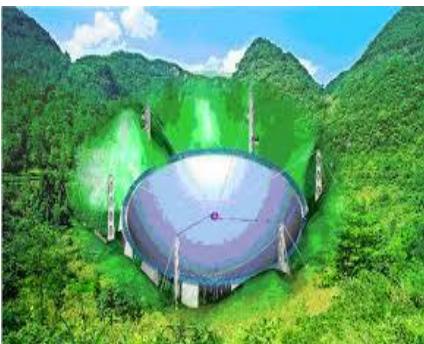
# 科学与工程计算应用举例：天体模拟

## 宇宙大爆炸



# 支撑FAST大科学工程

- **500米口径球面射电望远镜FAST**（Five hundred meters Aperture Spherical Telescope），贵州黔南州平塘县，是世界上最大口径的射电望远镜，将在20-30年内保持世界一流设备的地位。
- 围绕FAST的两个重要科学目标“脉冲星巡天”和“中性氢谱线巡天”，需要针对上百万个星系开展谱线处理、图像处理、噪声处理、干扰处理、宇宙参数模型拟合、脉冲星搜寻、高精度时间模型拟合等工作，需要建设“实时监测+海量数据存储+高效精确高性能计算能力”的高性能计算平台。
- FAST一期，峰值数据将达到40TB/天（ $24\text{小时} \times 3600\text{秒} \times 1\text{GB}/2$ ），峰值计算能力至少需要每秒浮点运算次数达200万亿次以上，存储至少在20PB以上。
- 随着FAST数据的长期积累，对存储和计算能力的需求还将进一步提升。

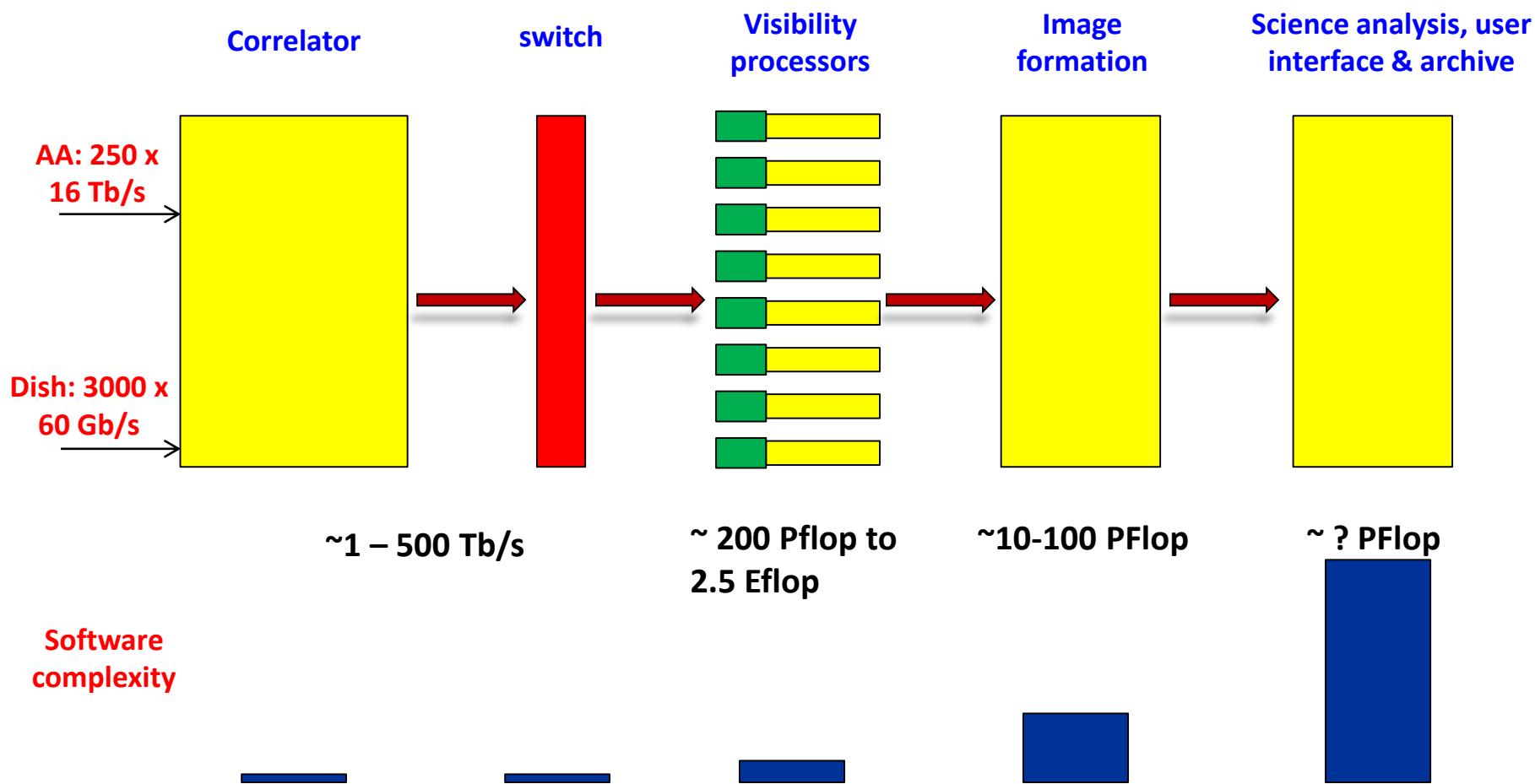


# 支撑SKA大科学工程

- **平方公里阵列射电望远镜SKA**（Square Kilometer Array），我国参与的国际大科学工程，是世界最大的射电望远镜项目，由上千台天线组成，其中有一半天线位于中央直径5公里的区域内，另有四分之一的天线散布在周围150公里的区域内，其余的分布在大约3000公里的范围内，呈螺旋形排列，主要在南非和澳大利亚
- 能探测到宇宙大爆炸之后第一代恒星和星系形成时发出的电磁波、揭示磁场在恒星和星系演化过程中的作用、探测暗能量产生的种种效应
- SKA一期工程于2020年完工，其存储和计算需求将是FAST的10到100倍。



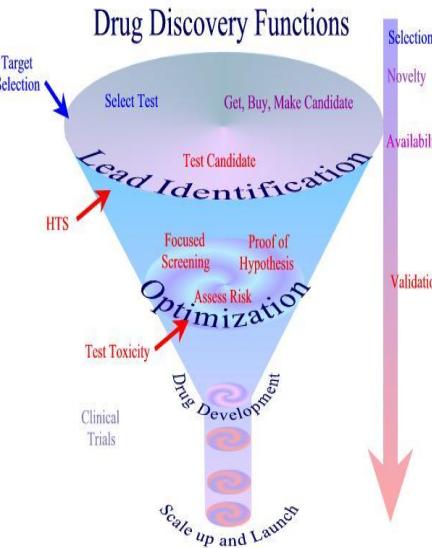
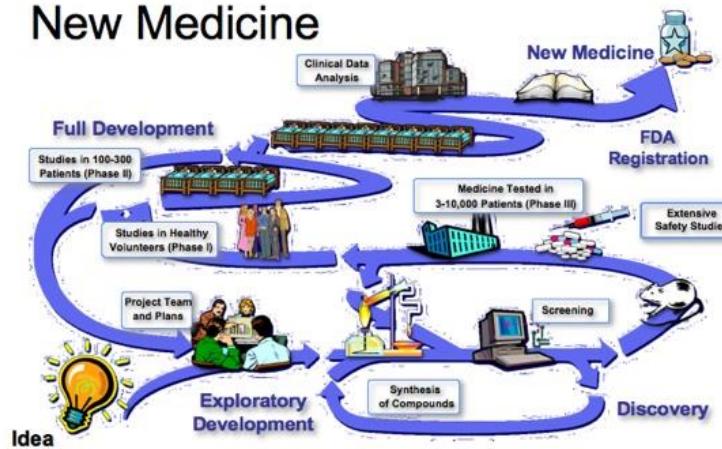
# SKA的计算需求



# 科学与工程计算应用举例：新药研制

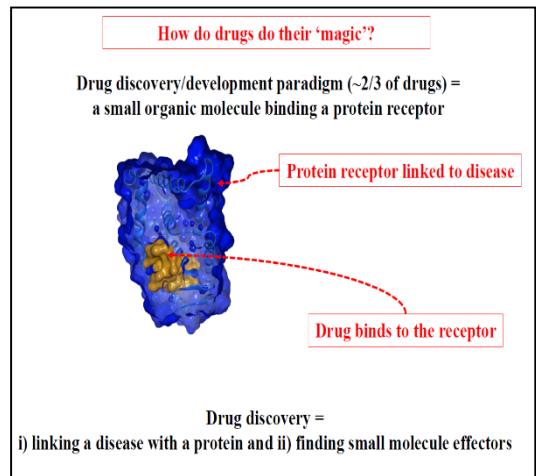
- 通常情况下，研制一款新药的投入从几亿到几十亿美元且需要10到15年的时间
- 工业界需要多快好省的从大量候选药物设计方案中尽快找出可行的方案
- 大部分的药物研制计划都以失败告终
- 每年全球制药企业的研发人员耗费精力研究数万种化合物，其中有些化合物对某种疾病尽管有特效，但由于其对应的人群太小，因而不具备商业开发的价值。这类药物被称为“坠落天使”(fallen angel)。

## The Long Road to a New Medicine

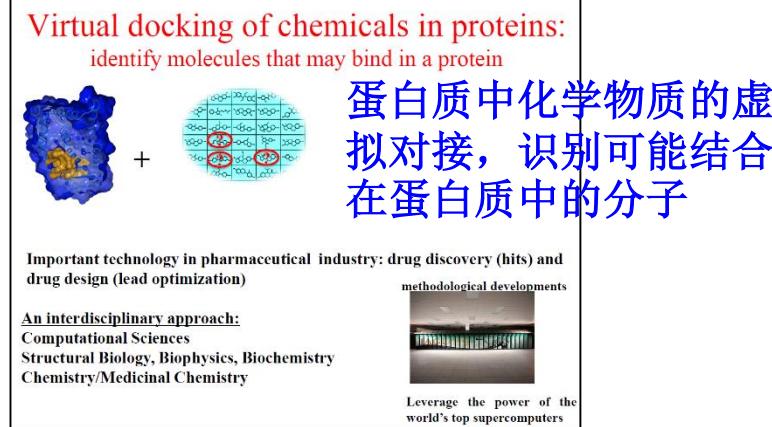
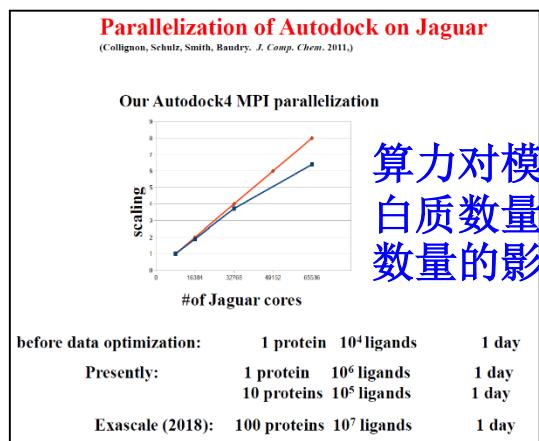
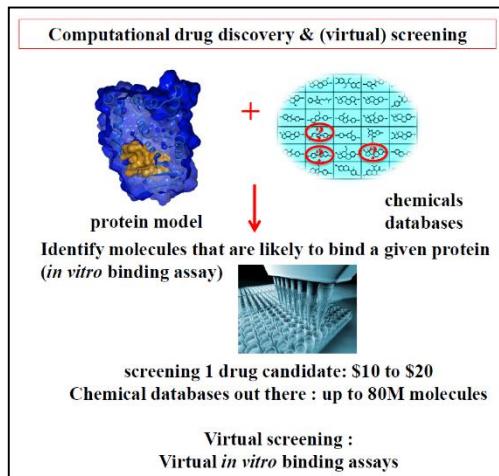


# 药物模拟的实现

现实世界的药物作用：药物有机小分子与蛋白质受体结合，观察结果

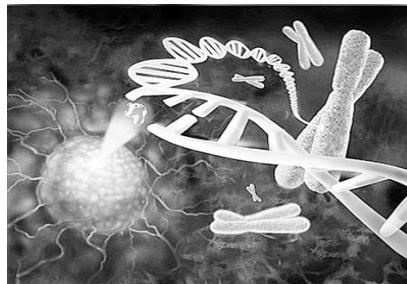


计算模拟：构建蛋白质模型，结合化学数据库进行药物扫描



# 科学与工程计算应用举例：个性化医疗

- 2013年5月，好莱坞影星安吉丽娜·茱莉做出一项轰动全球的决定：切除双侧乳腺。她的乳腺并未病变，但仍然执意手术，支持她的是一次价值4000美元的基因检测：检测显示，她体内携带有BRCA1型基因突变。
- 小小基因的个体差异巨大，会影响药物治疗效果
  - 一个令人惊愕的数据：在所有被服用抗癌药物中，只有25%起作用，其它75%几乎被浪费。
  - 药物对不同个体的疗效差异达300倍，一个不起眼的基因突变往往会使具有高度靶向性的抗癌药物完全无效
- 对所有病人肿瘤样本基因组进行完整测序，把肿瘤细胞基因序列和病人正常细胞基因序列进行对比，验证病人的癌症基因变异，通过基因测序，医生能预先知道某种药物会对哪些人管用。



# 个性化医疗的基础是基因测序和基因序列比对

**Beyond the Human Genome Project**

In 2003 scientists in the Human Genome Project obtained the DNA sequence of the 3 billion base pairs making up the human genome.

- The human genome is nearly the same (99.9%) in all people.
- Only about 2% of the human genome contains genes, which are instructions for making proteins.
- Humans have an estimated 30,000 genes; the functions of more than half of them are unknown.
- Almost half of all human proteins share similarities with those of other organisms, underscoring the unity of life.

*Many new discoveries yet to come!*

**The Path Forward**

**Scientific Discovery**

How does DNA impact HEALTH?

**Discovery Path** Identify and understand the differences in DNA sequence (A, T, C, G) among human populations

What do all the GENES do?

**Discovery Path** Discover the functions of human genes by experimentation and by finding genes with similar functions in the mouse, yeast, fruit fly, and other sequenced organisms

What does most of the human genome DO?

**Discovery Path** Identify important elements in the non-coding regions of DNA that are present in many different organisms, including humans

How does the genome enable LIFE?

**Discovery Path** Explore life at the ultimate level of the whole organism instead of single genes or proteins. The DOE Genomes to Life program provides a foundation for this understanding by using the information found in the genomes of microbes, life's simplest organisms, to study how proteins—the products of genes—carry out all activities of living cells

**Diverse Applications**

**Medicine**

- Develop more accurate and rapid diagnostics
- Design customized treatments

**Microbes for energy and the environment**

- Clean up toxic wastes
- Help mitigate global climate change
- Generate clean energy sources (e.g., hydrogen)

**Bioanthropology**

- Understand human lineage
- Explore migration patterns through time

**Agriculture, livestock breeding, bioprocessing**

- Make crops and animals more resistant to diseases, pests, and environmental conditions
- Grow more nutritious and abundant produce
- Incorporate vaccines into food products
- Develop more efficient industrial processes

**DNA identification**

- Identify kinships, catastrophe victims
- Exonerate or implicate people accused of crimes
- Identify contaminants in air, water, soil, food
- Confirm pedigrees of animals, plants, foods, wines

**The Basics: From DNA to working cells**

DNA contains the hereditary material of all living systems. The genome is an organism's complete set of DNA and is organized into chromosomes. DNA contains genes which sequence specifies how and when to build proteins. Proteins perform most essential life functions, often working together as molecular machines. Molecular machines interact through complex, interconnected pathways and networks to make the working cell come alive. Communities of cells range from a single human (comprising a hundred trillion cells) to associations of microbes (each a single cell in a particular environmental niche).

**Genome**  
The Secret of How Life Works

Made possible by  Smithsonian Arts and Industries Building  
June 7, 2003 - January 4, 2004  
A National Science Foundation Program

Poster prepared by the Human Genome Management Information System, Oak Ridge National Laboratory, U.S. Department of Energy Office of Science (2003). Poster contacts: Dennis Cooper (cooper@ornl.gov) and Judy Wickett (wickett@ornl.gov)

## Is China on its way to dominating the genome market?

Biotechnology is one of the seven key sectors identified in China's 12th Five-Year Plan

- BGI-Shenzhen**  
China's largest genomics company: (10-20% of global DNA analysis)
- Knowledge through acquisition**  
BGI acquired Complete Genomics in 2012 (one of the largest US genomics companies)
- Gaining ground in competing markets**  
Opened a DNA data analysis centre in the US in 2011
- Diversifying activities**  
Sequencing DNA for research on autism, obesity, agriculture, cancer
- Expanding customer base**  
Clients include research centres, non-profits, foundations

**Cost and time of sequencing human genome**

	2003	2013
<b>COST:</b>	\$3 BILLION	\$3,000-5,000
<b>TIME:</b>	13 YEARS	ONE DAY



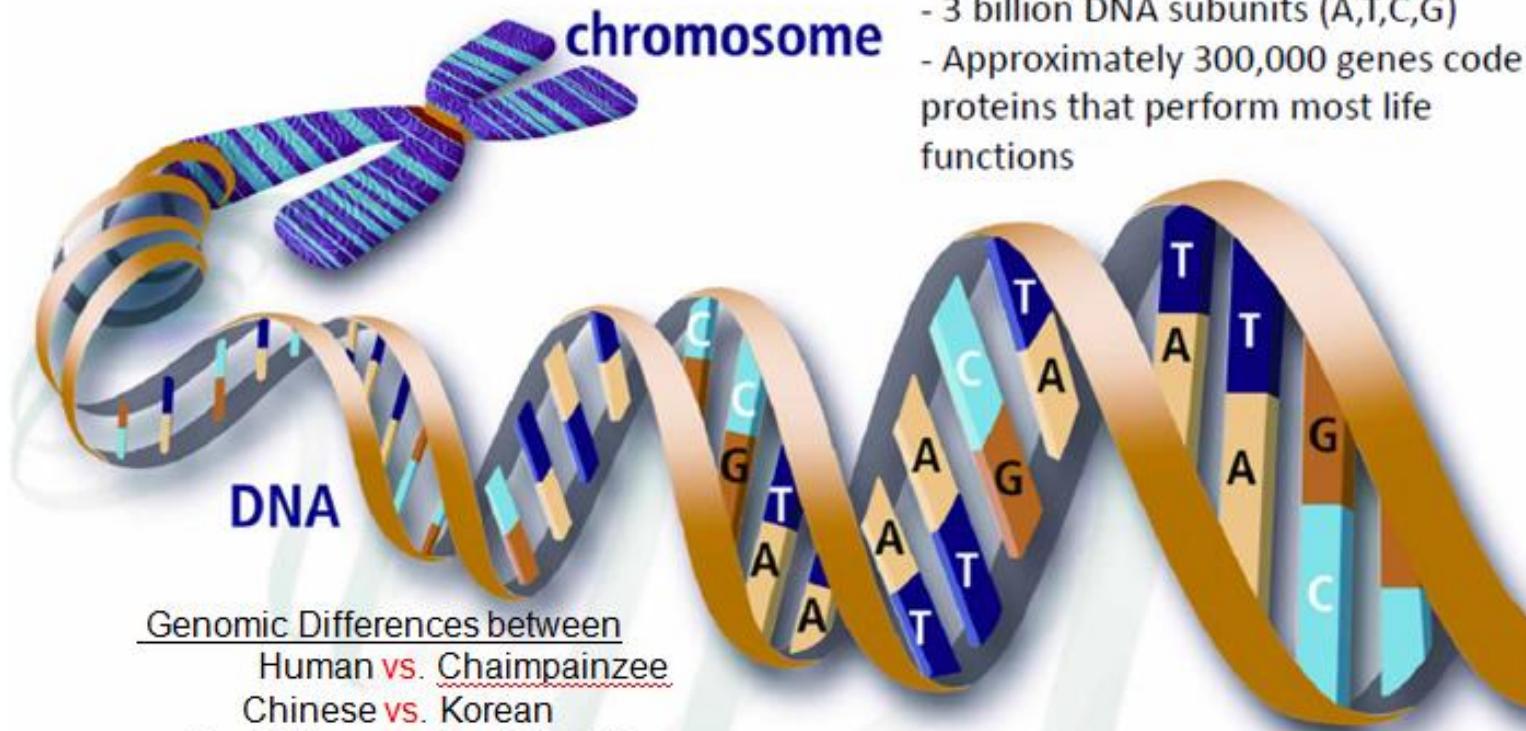
# 私人定制的治疗计划

## Genomic sequencing and personalized medicine

### 基因排序&私人定制医疗

Human genome reference database: 3 GB

Working data set: 40-100 X 3 GB = 120-300 GB



Genomic Differences between  
Human vs. Chimpainzee  
Chinese vs. Korean

David Chen vs. President Hu

Myself vs. my son

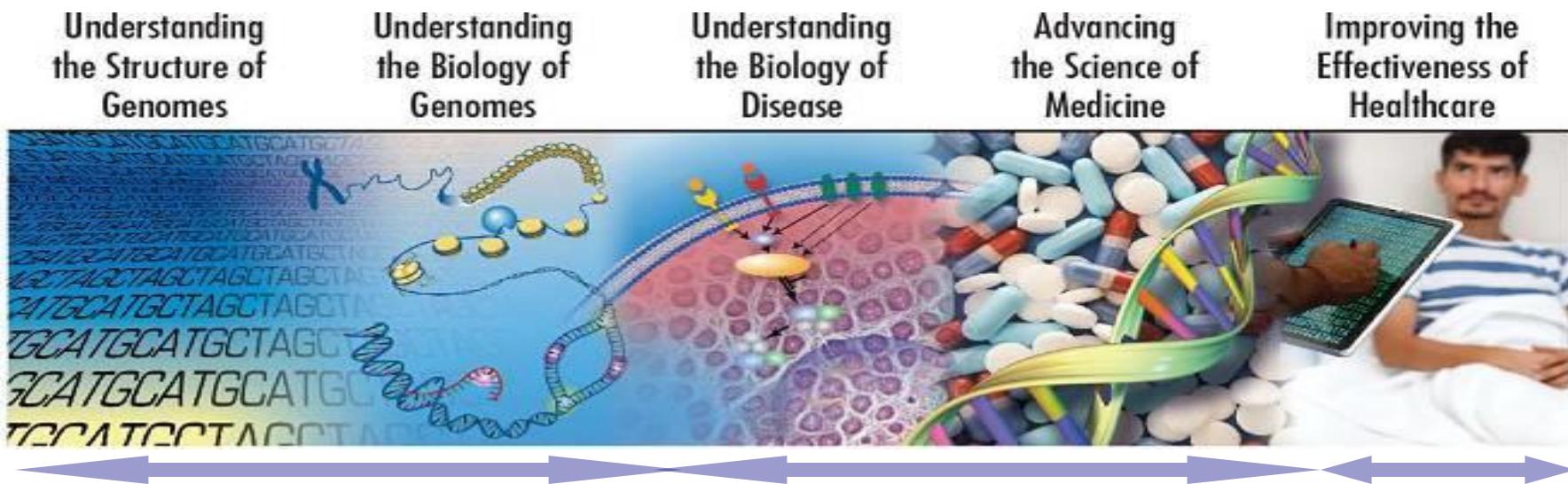
When I am healthy vs. when I have a cancer

#### Facts: Human Genome

- Trillions of cells
- 23 pairs of chromosomes
- 2 meters of DNA
- 3 billion DNA subunits (A,T,C,G)
- Approximately 300,000 genes code for proteins that perform most life functions

From Wang Bingqian's presentation at NVIDIA User Conference, Beijing, Dec 2011

# 私人定制的治疗计划三个阶段



## 测序数据收集

专注于超大数据生成，  
主要来自1000个全基因  
组测序、数据处理及归  
约；包括植物、动物和  
微生物组基因组学

## 转化医学

专注于数据整合，包括  
基因组数据，以及识别  
生物标志物、了解疾病  
机制和识别新医学治疗  
所需的分析

## 个性化医疗

专注于提供基  
因组医学以改  
善结果

NHGRI, a branch of NIH, has defined 5 steps for genomic medicine. ( E. Green et al., Nature 470)

---

# 第三部分： 高性能计算机的发展历史

# 在了解高性能计算机之前先了解算力的度量

## ■ High Performance Computing (HPC) units are:

- Flop: floating point operation, usually double precision unless noted (FP64, 双精度浮点)
- Flop/s: floating point operations per second (/有时候省略)
- Bytes: size of data (a double precision floating point number is 8 bytes)

## ■ Typical sizes are millions, billions, trillions...

Mega	$Mflop/s = 10^6 \text{ flop/sec}$	$Mbyte = 2^{20} \sim 10^6 \text{ bytes}$
Giga	$Gflop/s = 10^9 \text{ flop/sec}$	$Gbyte = 2^{30} \sim 10^9 \text{ bytes}$
Tera	$Tflop/s = 10^{12} \text{ flop/sec}$	$Tbyte = 2^{40} \sim 10^{12} \text{ bytes}$
Peta	$Pflop/s = 10^{15} \text{ flop/sec}$	$Pbyte = 2^{50} \sim 10^{15} \text{ bytes}$
Exa	$Eflop/s = 10^{18} \text{ flop/sec}$	$Ebyte = 2^{60} \sim 10^{18} \text{ bytes}$
Zetta	$Zflop/s = 10^{21} \text{ flop/sec}$	$Zbyte = 2^{70} \sim 10^{21} \text{ bytes}$
Yotta	$Yflop/s = 10^{24} \text{ flop/sec}$	$Ybyte = 2^{80} \sim 10^{24} \text{ bytes}$

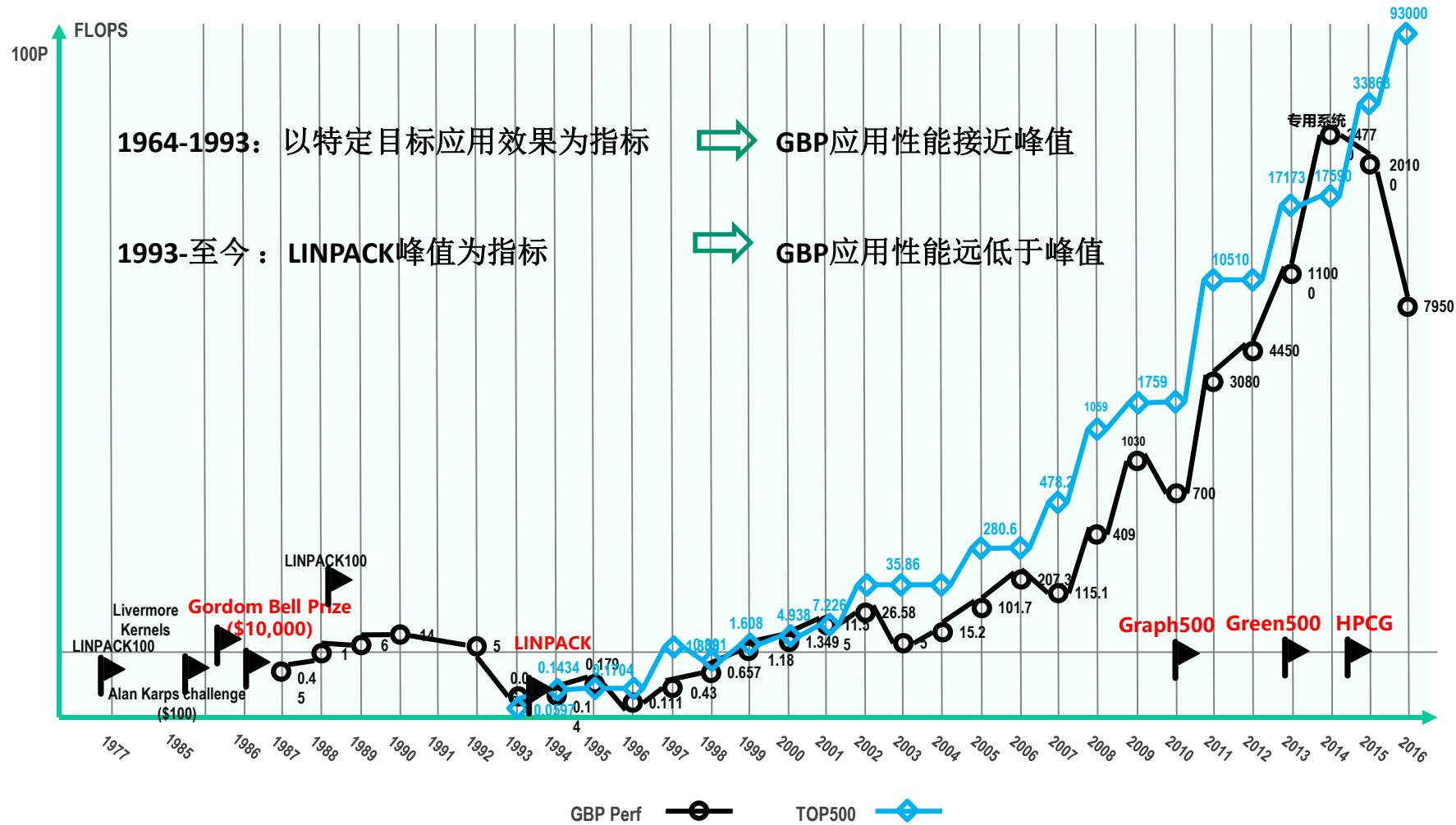
## ■ Current fastest (public) machine **143.5 Pflop/s, 2,397,824 cores**

- Up-to-date list at [www.top500.org](http://www.top500.org)

# 算力评价：The TOP500 Project from 1993

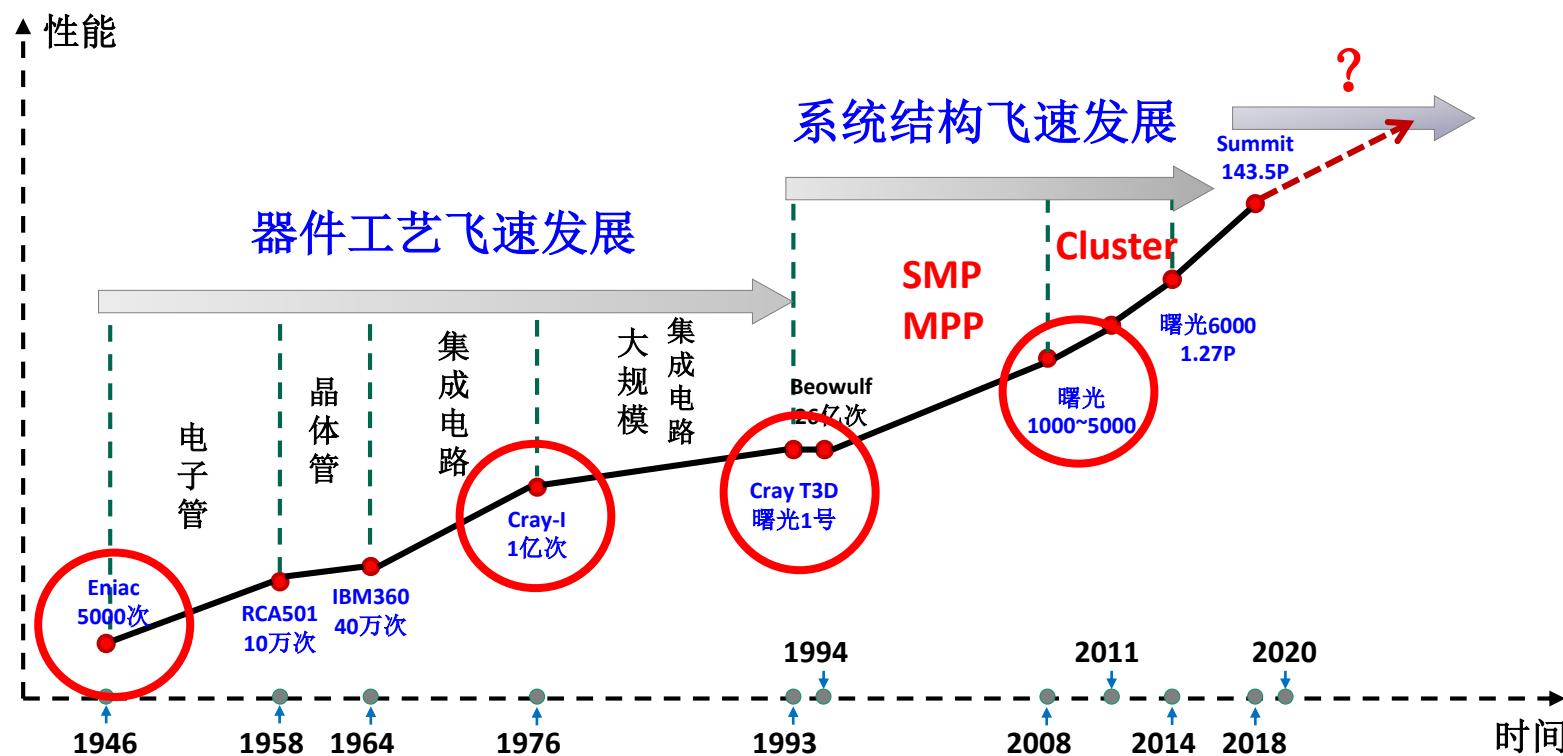
- 列出世界上最强大的 500 台计算机
- 衡量标准：Linpack的Rmax
  - Solve  $Ax=b$ , dense problem, matrix is random
  - Dominated by dense matrix-matrix multiply
- 每年更新两次：
  - ISC'xy in June in Germany
  - SCxy in November in the U.S.
- 相关信息访问网站: [www.top500.org](http://www.top500.org)

# 题外话：TOP500评价是否合理？



# 高性能计算机发展的时间线

- “史前时代”（大规模集成电路出现之前）、单机能力型时代（SMP）、多机能力型时代（MPP）、多级容量型时代（Cluster）



# 史前—基于晶体管电路的高性能计算机

## Seymour Roger Cray

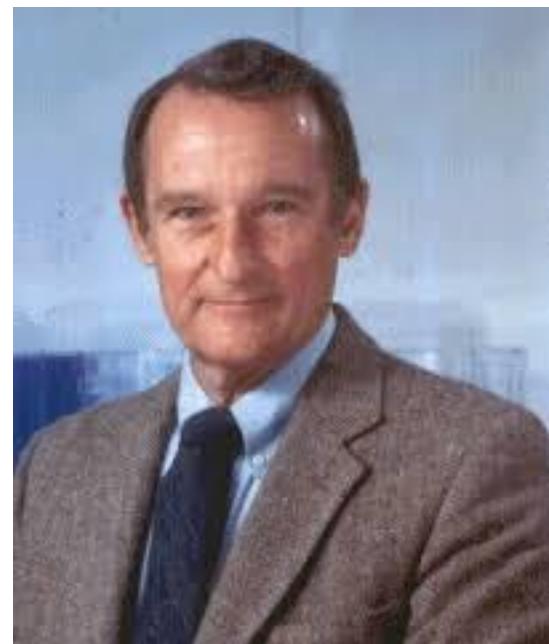
- The father of supercomputing

代表机器:

- CDC 6600/7600/8600
- Cray-1/2/3/4

创办的公司:

- Control Data Corporation
- Cray Research/Computer Corporation
- SRC Computers



September 28, 1925– October 5, 1996

*“Anyone can build a fast CPU. The trick is to build a fast system.”*

# 第一台超级计算机CDC 6600 [1964]

- 3 Mflops

- 1CPU, 40MHz

- 10 并行功能单元，每个单元执行不同的任务：浮点加法, 浮点除法, 布尔逻辑, etc.
- 60 位字长和 60 位寄存器
- 第一台RISC系统

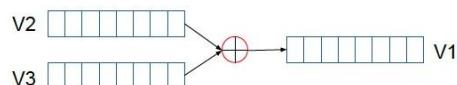
- \$13M



- 比当时最快的计算机 (IBM 7030 Strech) 快 10 倍
- 用在四个机柜周围的管道中循环的氟利昂冷却
- 10 个外设处理器，每个处理器都专门用于管理 I/O 和保持 CPU 任务队列满

# 第一台向量机Cray-I [1976]

- 136 Mflops
- 向量处理器



- 8个64-element x 64-bit vector (V) 寄存器, 以及1个向量长度(VL) and 1个向量掩码vector mask (VM)
- 12条独立的流水线执行单元
- 80MHz (个人电脑20年后才达到)
- 内存总线宽度24位, 内存容量8MB
- 1662块印刷电路板, 每块印刷电路板最多有 144 个 IC
  - 只使用了四种集成电路 (两种不同速度的5路、4路或非门、1Kbit SRAM、16x4位寄存器)
- \$8M



完成后安装在美国洛斯•阿拉莫斯国家实验室

用Verilog再现的Cray-I硬件: [Homebrew.Cray-1A-chrisfenton.com](http://Homebrew.Cray-1A-chrisfenton.com)

# 30 Years of SC上的Cray-I和我国的银河-I



Conference Series

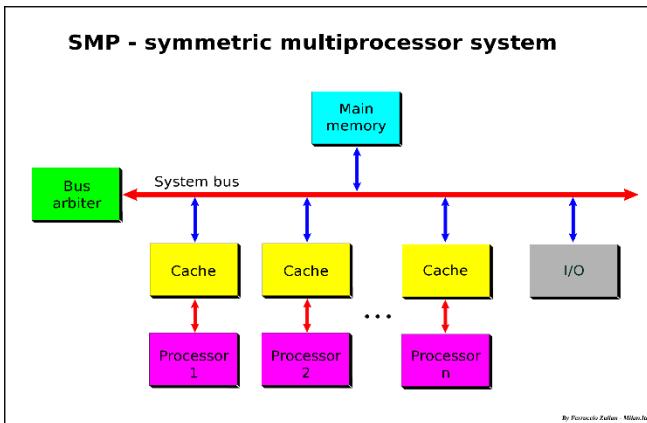
30TH ANNIVERSARY  
1988–2018

- 1983年，银河-I中国第一台每秒钟运算达1亿次以上的计算机研制成功
- 银河-I采用了类似Cray-I的向量机架构
- 中国成为继美国、日本之后，第三个能独立设计和制造巨型计算机的国家



# 单机能力型时代：SMP架构高性能计算机

- Symmetric Multi-Processing, Shared-memory Processing, **SMP**
  - 所有的CPU共享全部资源，如总线，内存和I/O系统等
  - 每个CPU访问内存中的任何地址所需时间是相同的
  - 由于资源争用，扩展能力有限



- **Cray-XMP[1982]**
  - 800 Mflops
  - 4 CPU, 105MHz
  - 16 M 64-bit words of SRAM memory
  - 32 disk storage units
  - 1.2GB, 10MB/s
- **\$15M**



- **曙光1号[1993]**
  - 640 Mflops
  - 单板4 M88100 CPU, 4块板
  - 内存容量768 MB
- **曙光1号诞生的第3天，美国便宣布解除10亿次计算机对中国的禁运**
- **折价2000万人民币成立曙光公司**



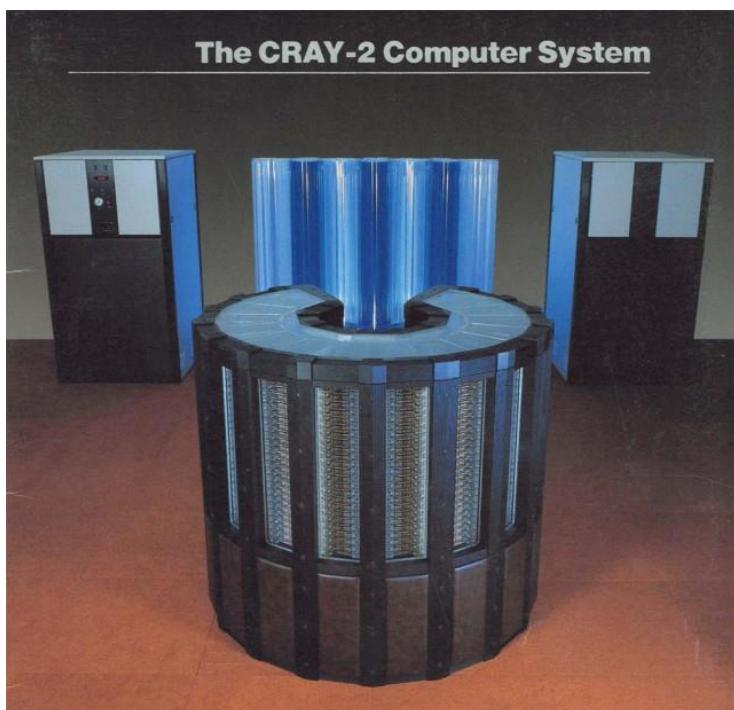
# 基于SMP的商用超算

## ■ Cray-2 [1985]

- 1.9 Gflops
- 256M 64bit内存
- 前台处理器（foreground processors）通过非常快的每秒千兆总线将数据从主内存加载到本地内存（类似于缓存，但不完全是），然后将指令传递给实际执行计算的后台处理器（background processors）
- 使用主流软件，主要是UniCOS, Unix System V 的具有一些BSD功能的衍生品
- 被许多大学和公司使用

## ■ 曙光1号同样使用了Unix操作系统，才顺利走上了产业化道路

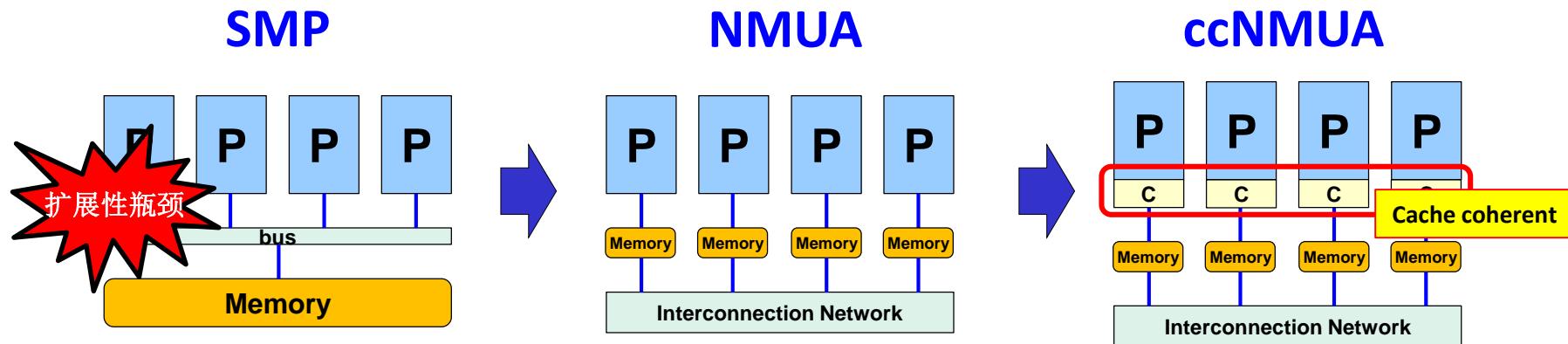
- 曙光公司成立前，曙光1号已在教育行业（中国科技大学、武汉大学）、信息服务（1994年开通国内第一个BBS网站，北京信息中心）、军队（总后油库管理）、政府（国家科委办公自动化）、援外项目（埃及穆巴拉克科学园）等项目上进行了成功的销售，累计生产与销售20多套。



Cray-2的别称是“泡泡”，历史上第一台浸没式液冷计算机，通过玻璃可以看到沸腾的氟化物（全氟三丁胺），**流浪地球中的超级计算机550W**

# 如何进一步提升单机能力：ccNUMA

- **NUMA: Non-uniform memory access**
  - 消除SMP总线竞争，提高系统扩展性
  - 访问本地和访问远程内存延迟不一致，影响编程
- **ccNUMA: Cache Coherent Non-uniform memory access**
  - Cache是提高处理器计算效率的重要手段
  - NUMA会破坏程序员使用Cache的方式（远程内存内容在本地缓存时）
  - ccNUMA从硬件层面向程序员屏蔽NUMA带来的本地缓存和远程内存不一致问题，保持程序员使用Cache的方式（透明无感）



\*SMP有没有cache coherent问题？

在ccNUMA系统中，多个处理器之间通过专门的硬件保持Cache中的数据和共享内存中的数据的一致性，不需要软件来保持多个数据副本之间的一致性。

# 过去和现在的ccNUMA

## ■ SGI Origin 2000 [1996]

- 支持128~1024 CPU
- 使用NUMAlink实现CPU之间互连，并维护CPU之间的Cache一致性



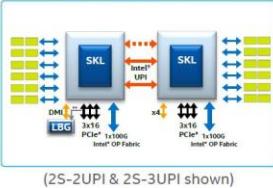
于 2001 年安装在美国洛斯•阿拉莫斯国家实验室的Origin 2000，被称为 ASCI Blue Mountain

SMP

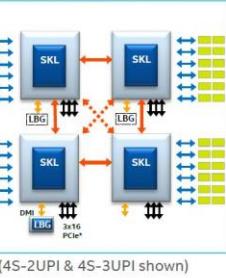


Platform Topologies

2S Configurations

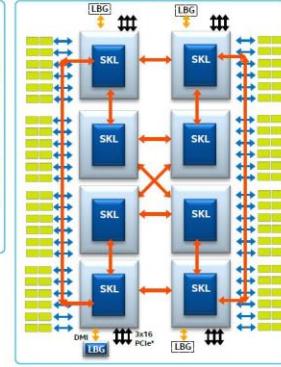


4S Configurations



ccNUMA

8S Configuration



INTEL® XEON® SCALABLE PROCESSOR SUPPORTS CONFIGURATIONS RANGING FROM 2S-2UPI TO 8S

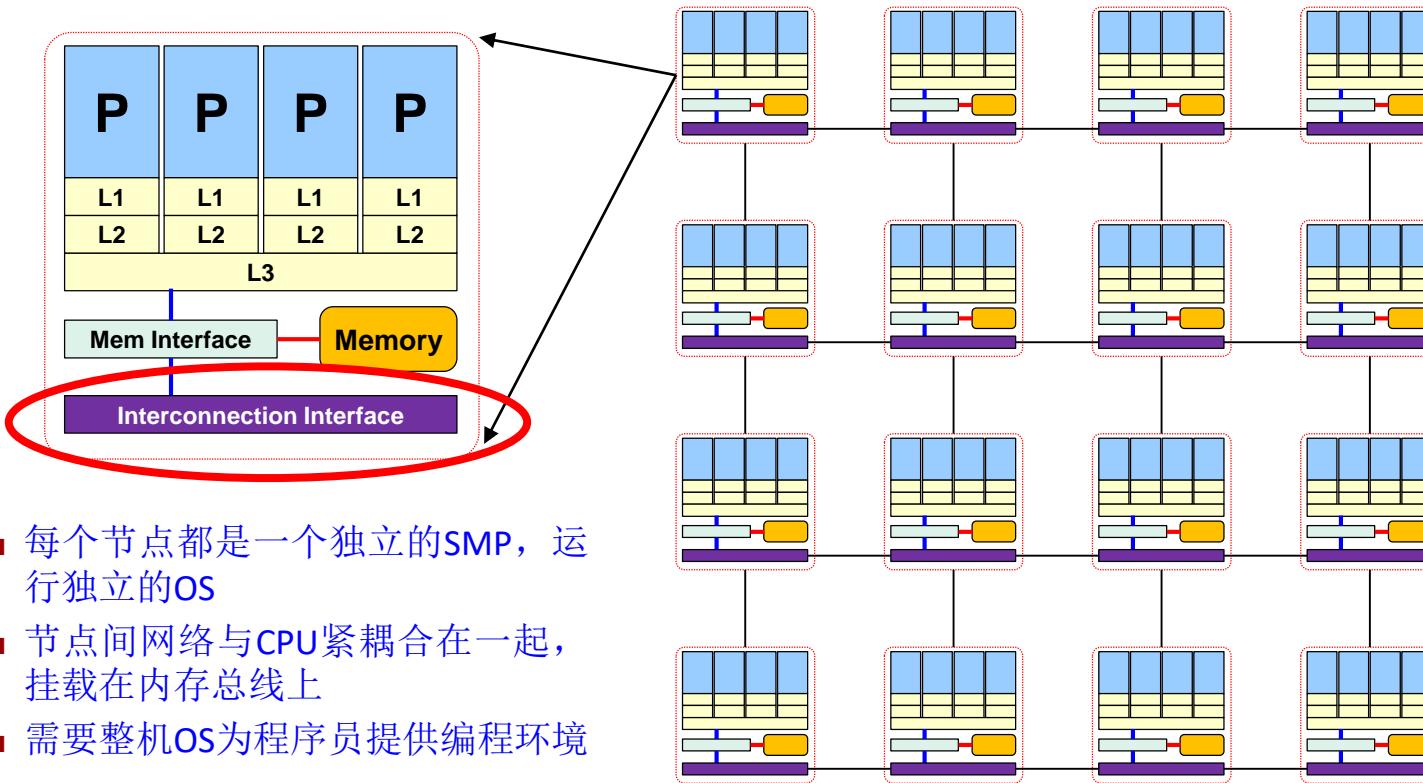


# 多机能力型时代：MPP高性能计算机

## ■ 单机和多机的区别，主要是程序员编程的区别

- 单机情况下，程序员面向的是一个内存空间，不需要显式处理数据通信
- 多机情况下，程序员面向的是多个内存空间，需要显式处理不同空间之间的数据交换

## Massively Parallel Processing (MPP)



# MPP: from Hitachi SR2201 to Fugaku

## ■ Hitachi SR2201

- 600 **Gflops**
- HARP-1E CPU(RISC, vector)
- 2048 CPU@150MHz
- 2D and 3D networks
- 2.4Gb/s over each link



## ■ Fugaku

- 442 **Pflops** (6个数量级)
- A64FX CPU(ARM v8, vector)
- 158976 CPU
- 6D networks
- 28Gb/s over each link

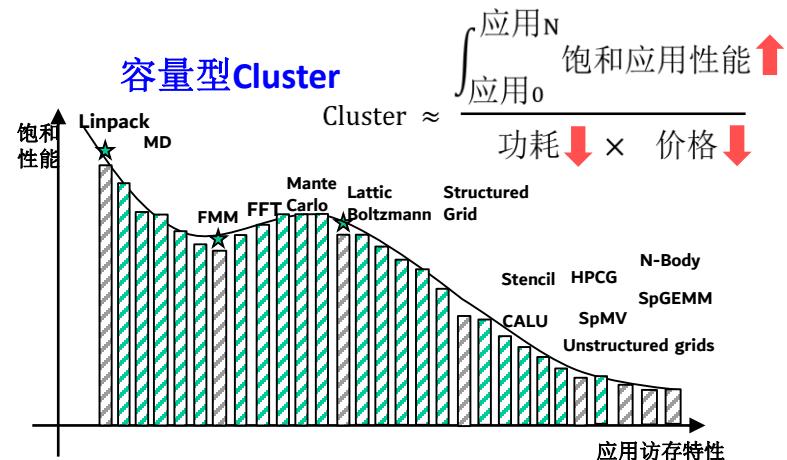
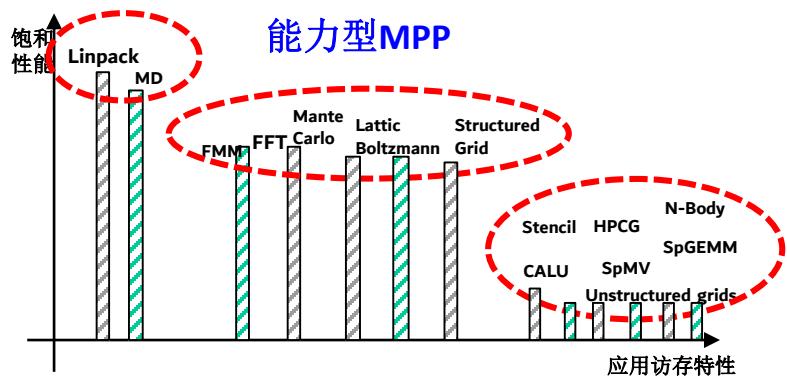


(C) RIKEN

# 多机容量型时代：Cluster高性能计算机

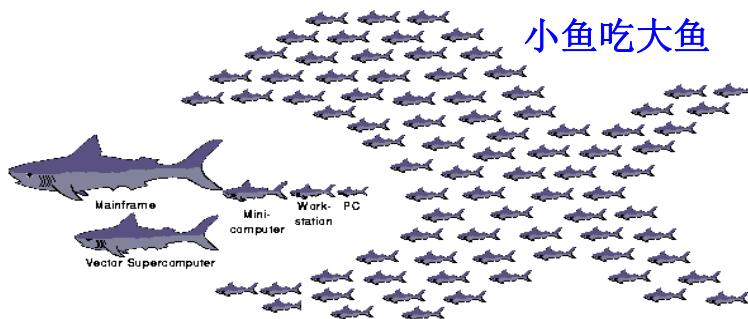
## ■ 能力和容量的区别，主要是应用生态的区别

- 能力型高性能计算机通常面向特定领域定制，追求该领域应用的极限性能
- 容量型高性能计算机主要为了满足各个领域应用对性能的要求，追求费效比



## 如何提高费效比？

- 采用通用的、商用的主处理部件
- 为不同领域应用定制加速部件
- 定制高性能互连网络（挂载在IO总线）
- 兼容商用或开源操作系统
- 打造平台式的机群软件.....等等

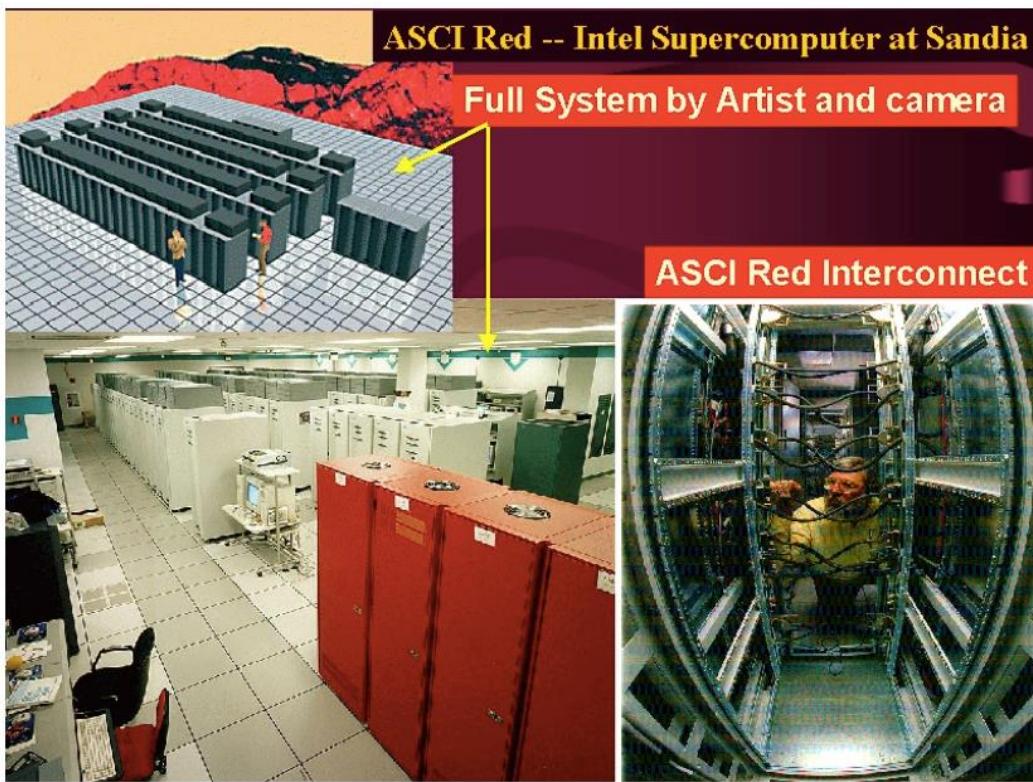


# Off-the-shelf CPUs Supercomputer [1996]

第一台达到 Teraflops 的超级计算机，由现成的 CPU（Pentium Pros，然后是 Pentium II Xeons）和其他现成的商业部件制成

## ASCI Red

- 1.3Tflops
- 6000 200MHz Intel Pentium CPUs
- distributed message-passing
- \$46 million



安装于美国桑迪亚国家实验室

# The REAL Commodity Supercomputer [1998]

## ■ Beowulf cluster

- Cluster of Workstations (COW)
- MPI
- Parallel Virtual Machine(PVM), 允许用户像使用单机一样使用机群
- Linux



# 基于机群架构的曙光系列超级计算机

Dawning 2000



First Cluster in China

Dawning 3000



First Industry Standard Cluster

Dawning 4000

11TFlops  
Ranked at 10<sup>th</sup> inTOP500

Dawning 5000

First Blade Cluster  
230TFlops  
Ranked at 10<sup>th</sup> inTOP500

Dawning 6000 (Nabulea)

Linpack:1.27PFlops  
3PFlops  
Ranked at 2<sup>nd</sup> inTOP500

Dawning Si-Cube



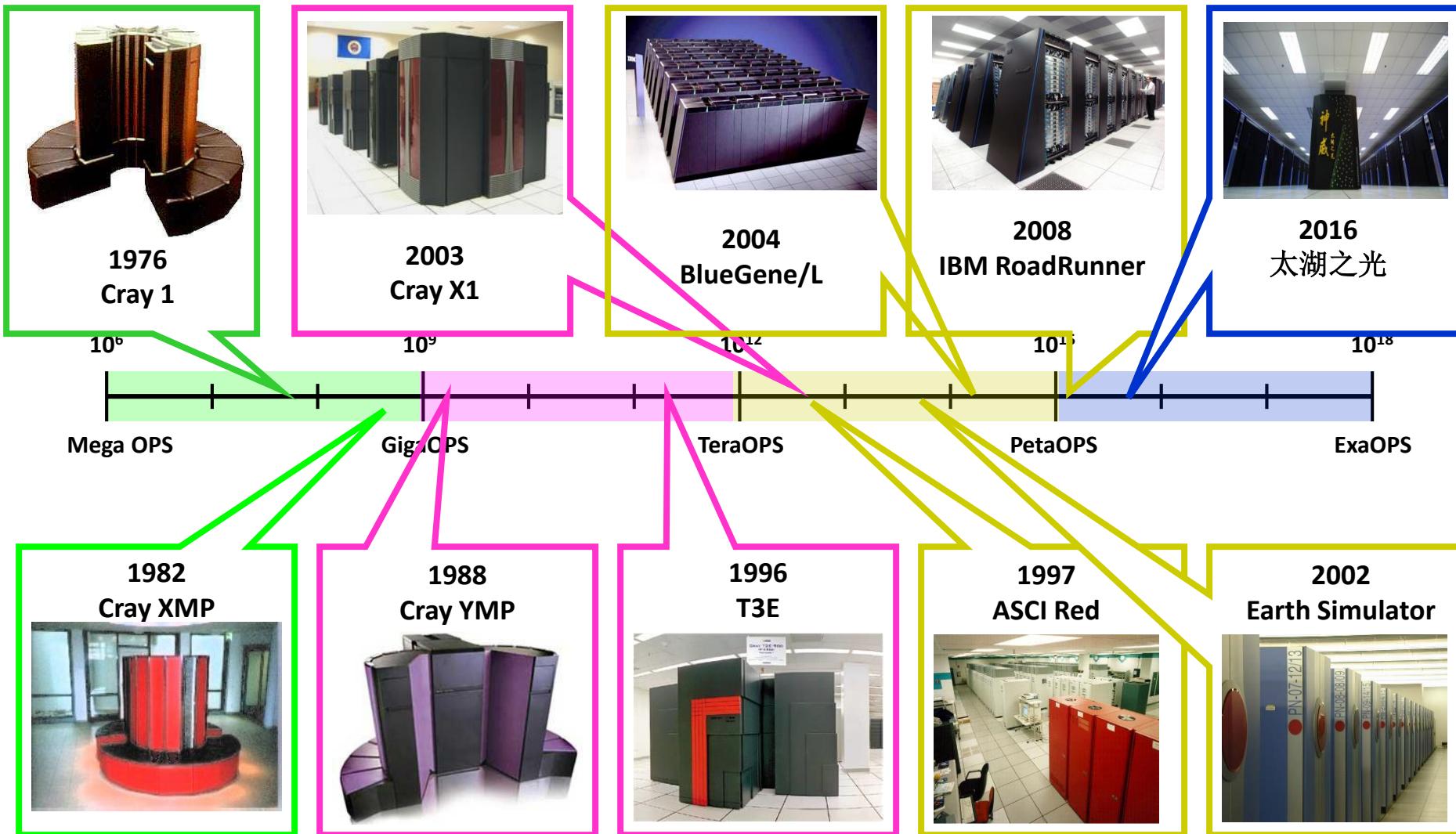
Exascale Computer Prototype

Dawning 7000



Exceeded Summit in 2018

# 从M到E，追逐性能的时间线



# 中国高性能计算在西方封锁之下发展

1949年

冷战时期西方成立了“巴统组织”，长期对中国实施禁运。

70年代

引入第一批Cray HPC产品，美国开始对HPC进行出口管制

。

80年代

美国实行“Supercomputer Safeguard Plan”，与美国HPC制造商形成协议，控制出口。

90年代

冷战后西方又成立“瓦森纳协议”，继续联合封锁中国。

1993年，美国宣布算力不高于**194 MTOPS**的计算解除出口管制，中国、东欧等除外。

1995年，美国政府“放开”HPC出口限制，将出口分为四个类别，中国为了第三类国家，HPC出口算力限制为军用**2000 MTOPS**，民用**7000 MTOPS**。

2022年

美国政府要求Nvidia停止向中国（含中国香港）出口两种用于人工智能工作的顶级计算芯片，此次管制涉及英伟达 A100 和即将出货的 H100 两款芯片，以及英伟达未来推出的峰值性能等同或超过 A100 的其他芯片

# 银河天河系列超级计算机的发展



1983年



2010年



2013年



# 神威系列超级计算机的发展



神威一号

384GFLOPS, TOP500.  
No.48

1999年



神威蓝光计算

首个全部采用国产  
CPU神威1600和系统  
软件构建的PFLOPS  
级超算

2011年

2016年

神威太湖之光



2016.06, TOP500. No.1  
采用自主研发的申威  
26010众核处理器

# 天河六连冠

2010年11月，“天河一号”在第36届Top500排行榜位居世界第一，计算能力每秒4.7 Pflops，这是中国超算首次站到世界超算之巅。

2013年06月，“天河二号”大型超级计算机在第41届Top500排行榜以每秒33.86 Pflops的速度成为全球最快的超级计算机。

2013年11月，“天河二号”在第42届世界超算Top500获得“二连冠”。

2014年06月，“天河二号”在第43届世界超算Top500获得“三连冠”。

2014年11月，“天河二号”在第44届世界超算Top500获得“四连冠”。

2015年06月，“天河二号”在第45届世界超算Top500获得“五连冠”。

2015年11月，“天河二号”在第46届世界超算Top500获得“六连冠”。



“天河一号”超级计算机



“天河二号”超级计算机

# 神威的Gordon Bell Prize



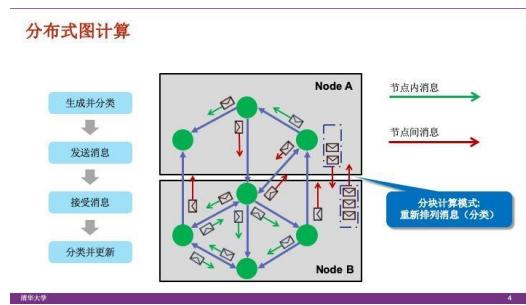
## ACM Gordon Bell Prize Winner

2016年，基于神威·太湖之光的应用成果“大气动力框架的高可扩展全隐式求解器”



## ACM Gordon Bell Prize Winner

2017年，基于神威·太湖之光的应用成果“非线性地震模拟”



## ACM Gordon Bell Prize 提名

2018年，基于神威·太湖之光的应用成果“超大规模图计算框架——神图”

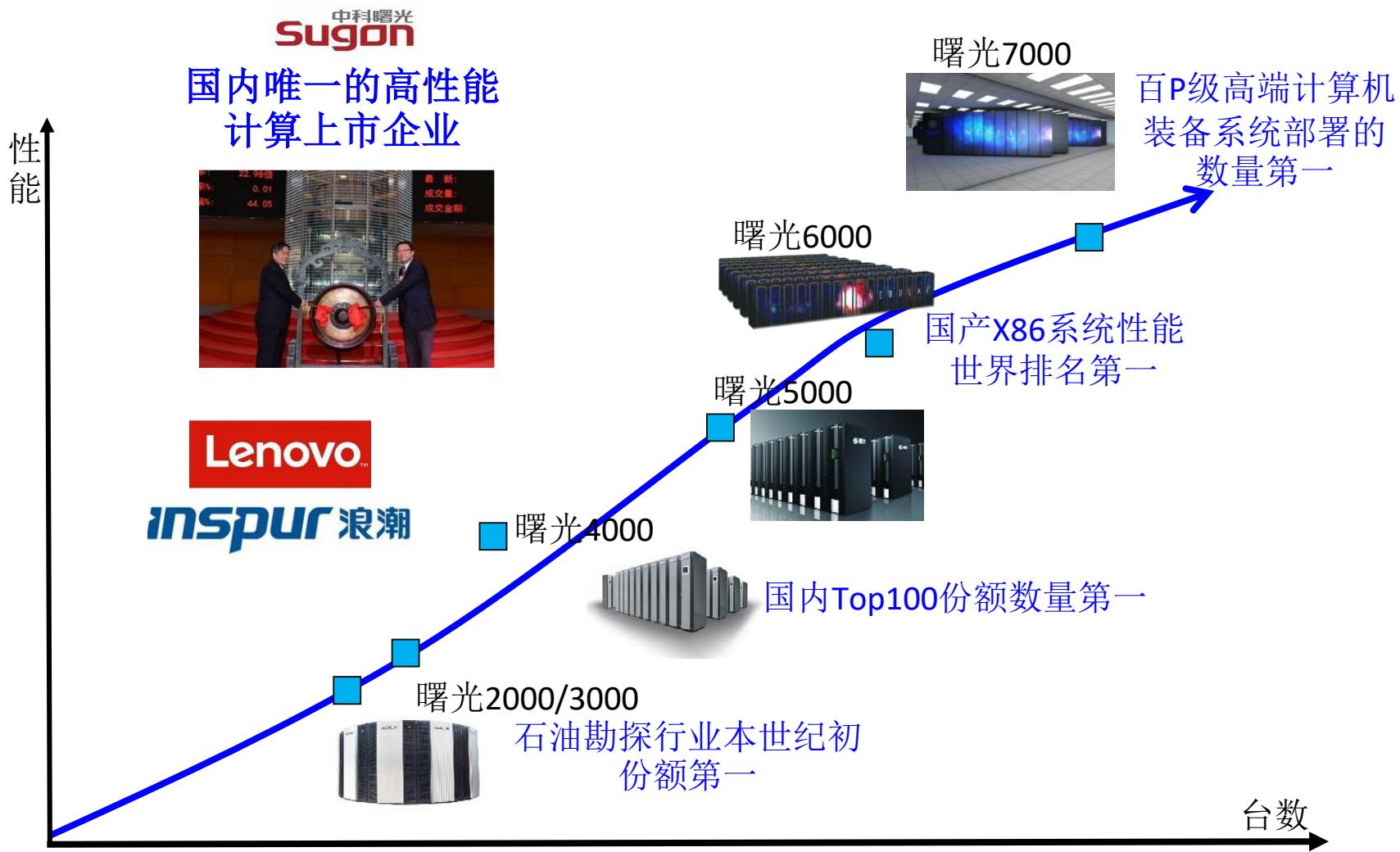


# 国产商用高性能计算机的产业化之路

- 高性能计算系统研制仅依靠若干科研单位和军方机构，仅能满足少数国家战略部门需求
- 国内普通用户只能高价购买落后产品
- 国产系统一旦研制成功，就会遭受西方巨头恶意压价打击，进一步生存和发展困难重重



# 国产商用高性能计算机的产业化之路



# 不仅单台性能突破，台数也实现超越



Rank	Site	Computer	Century	Cores	Rmax (PFlop/s)	Power (kW)
1	Oak Ridge National Laboratory	<b>Frontier</b> HPE Cray EX235a, AMD Optimized 3rd Generation EPYC 64C 2GHz, AMD Instinct MI250X, Slingshot-11, HPE	USA	8,699,904	1,194.00	22,703
2	Argonne National Laboratory	<b>Aurora</b> HPE Cray EX - Intel Exascale Compute Blade, Xeon CPU Max 9470 52C 2.4GHz, Intel Data Center GPU Max, Slingshot-11, Intel	USA	4,742,808	585.34	24,687
3	Microsoft Azure	<b>Eagle</b> Microsoft NDv5, Xeon Platinum 8480C 48C 2GHz, NVIDIA H100, NVIDIA Infiniband NDR, Microsoft	USA	1,123,200	561.20	
4	RIKEN Center for Computational Science	<b>Fugaku</b> A64FX 48C 2.2GHz, Tofu interconnect D, Fujitsu	Japan	7,630,848	442.01	29,899
5	EuroHPC/CSC	<b>LUMI</b> HPE Cray EX235a, AMD Optimized 3rd Generation EPYC 64C 2GHz, AMD Instinct MI250X, Slingshot-11, HPE	Finland	2,752,704	379.7	7,107
6	EuroHPC / CINECA	<b>Leonardo</b> BullSequana XH2000, Xeon Platinum 8358 32C 2.6GHz, NVIDIA A100 SXM4 64 GB, Quad-rail NVIDIA HDR100 Infiniband, EVIDEN	Italy	1,824,768	238.7	7,404
7	Oak Ridge National Laboratory	<b>Summit</b> IBM Power System AC922, IBM POWER9 22C 3.07GHz, NVIDIA Volta GV100, Dual-rail Mellanox EDR Infiniband, IBM	USA	2,414,592	148.6	10,096
8	EuroHPC/BSC	<b>MareNostrum 5 ACC</b> BullSequana XH3000, Xeon Platinum 8460Y+ 40C 2.3GHz, NVIDIA H100 64GB, Infiniband NDR200, EVIDEN	Spain	680,960	138.2	2,560
9	NVIDIA Corporation	<b>Eos NVIDIA DGX SuperPOD</b> NVIDIA DGX H100, Xeon Platinum 8480C 56C 3.8GHz, NVIDIA H100, Infiniband NDR400, Nvidia	USA	485,888	121.4	
10	Lawrence Livermore National Laboratory	<b>Sierra</b> IBM Power System AC922, IBM POWER9 22C 3.1GHz, NVIDIA Volta GV100, Dual-rail Mellanox EDR Infiniband, IBM / NVIDIA / Mellanox	USA	1,572,480	94.64	7,438

# 美国已开始全面重启对高性能计算技术的封锁

## 断供国防部门

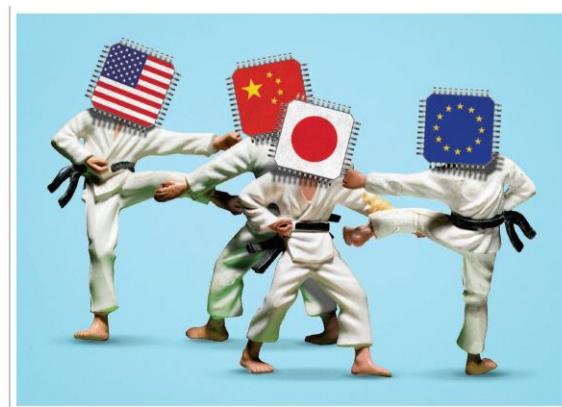
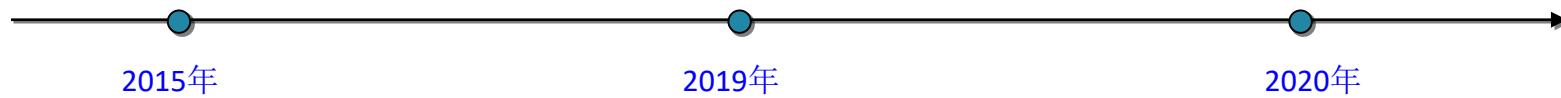
- 国家超级计算长沙中心
- 国家超级计算广州中心
- 国家超级计算天津中心
- 国防科学技术大学

## 断供民营企业

- 中兴
- 华为
- 中科曙光/海光
- 江南计算技术研究所

## 断供技术交流

- 参加超算大会受限
- 中国主要超算企业不参加TOP500排名
- HPC公派留学生出国受阻
- 将科研院所列入实体清单



---

# 第四部分： 如何在高性能计算机上 编程（为什么难）

# 程序员对性能的看法

---

问题:你如何让你的程序运行得更快?

2004年之前的答案:

- 只需等待 6 个月，然后购买一台新机器!
- (或者，如果你真的沉迷技术，你可以了解并行性.)

2004年后的答案:

- 你需要编写并行程序.

# 并行编程

- 要使用可扩展的并行计算机（多数是高性能计算机），你必须能够编写并行程序
- 你必须了解编程模型、以及用于实现它的编程语言、并行编程库、系统软件
- 所以，并行编程并不简单，或者说比较难



# 从问题出发的并行编程模型分类

## ■ 并行程序的两种通用模型

- 任务并行 (Task parallel)
  - 问题被分解为要多个任务来执行
  - 不同任务之间通过通信相互协调执行
- 数据并行 (Data parallel)
  - 问题被视为可以并行操作的数据
  - 数据被分配到各个进程进行本地计算

## ■ 可伸缩 (Scalable) 并行程序的特点

- 可以通过数据域分解改善数据的局部性
- 通信在整个程序中的运行时间占比不会随规模增大而显著增加

# 从并行计算机系统结构角度的编程模型分类

## 共享内存编程模型：

- 机器提供共享内存地址空间，比如SMP、ccNUMA
- 通常比较容易编程
  - 通过（共享）数据进行隐式通信
  - 但需要显式同步对同一地址数据的访问
- 编程方法
  - 手动方法
    - 使用标准线程库的多线程进行编程
  - 自动方法
    - 并行编译器
    - OpenMP并行制导指令
  - 使用显式线程(e.g. POSIX threads)

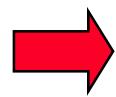
# 从并行计算机系统结构角度的编程模型分类

## 分布式内存编程模型

- 机器提供分布式内存地址空间，比如MPP、Cluster
- 相对更难编程
  - 需要显式的为不同地址空间分配数据
  - 需要显式的通过消息实现通信
  - 需要显式的通过消息实现同步
- 编程方法
  - 使用消息传递编程框架
    - 有很多框架可以使用 (但绝大多数时候都用MPI)
    - 支持的核心操作包括： send-receive, one-sided (RDMA Read/Write) , active messages
  - 一般来说都是数据并行 (不同的节点计算不同的数据分块)

# 一些并行编程的原则

- 一. 要尽可能挖掘程序的并行度 (**Amdahl's Law**)
- 二. 考虑好并行的**粒度**—每个并行任务应该有多大
- 三. 要充分考虑**局部性**—因为通常移动数据比计算代价更高
- 四. 考虑**负载均衡**—不要让一千个处理器都在等待一个较慢的处理器 (**straggler**)
- 五. 要结合**体系结构**做反复的性能建模/调试/调优



所有这些使得并行编程比顺序编程更难。

# 计算机中的“自动并行”

- 比特位级并行
  - 例如CPU中多个运算部件.
- 指令集并行 (ILP)
  - 多发射，每个时钟周期执行多条指令
- 访存并行
  - 将访存操作与计算操作重叠
- 操作系统提供的并行
  - 在商用SMP计算机上实现多个作业并行执行

但这些还不够：为了获得更高的性能，需要用户自己来识别程序中的可并行部分，并合理的安排协调它们并行执行

# 一、尽可能挖掘程序中的可并行部分

## ■ Amdahl's law 阿姆达尔定律

- 假设 $s$ 表示一个程序中只能串行执行部分的占比，那么 $(1-s)$ 就是可并行化部分的占比
- $P$  = 用于并行程序执行的处理器数量

$$\text{Speedup}(P) = \text{Time}(1)/\text{Time}(P)$$

$$<= 1/(s + (1-s)/P)$$

$$<= 1/s$$

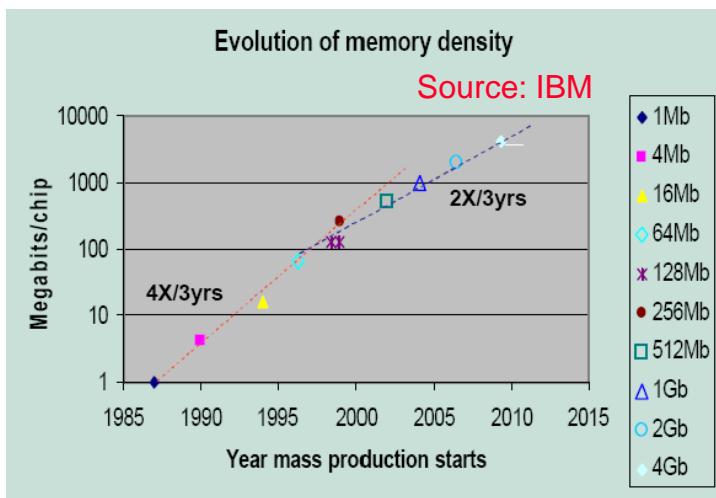
- 即使并行部分实现了完美加速，性能也受到串行部分限制

## 二、要考虑好并行的粒度

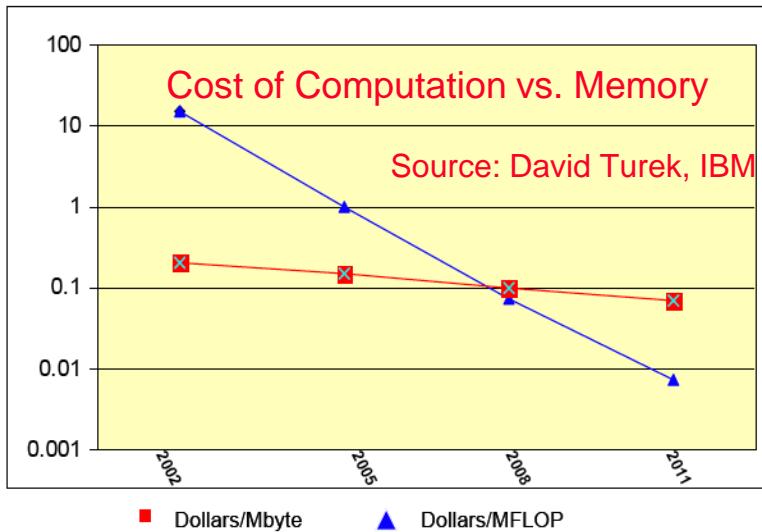
---

- 为什么要考虑并行粒度？因为并行有开销！
- 并行开销是我们想获得期望加速比最大的障碍
- 并行开销包括：
  - 启动线程或者进程的开销
  - 交换共享数据带来的通信开销
  - 同步开销
  - 额外的计算开销（例如分子动力学模拟中的截断半径）
- 每一个的时间消耗都可能达到毫秒级  
某些系统中： **milliseconds (=millions of flops)**
- 因此我们需要考虑**tradeoff**: 算法需要足够大的并行粒度来降低并行开销，但又不能大到没有足够的并行

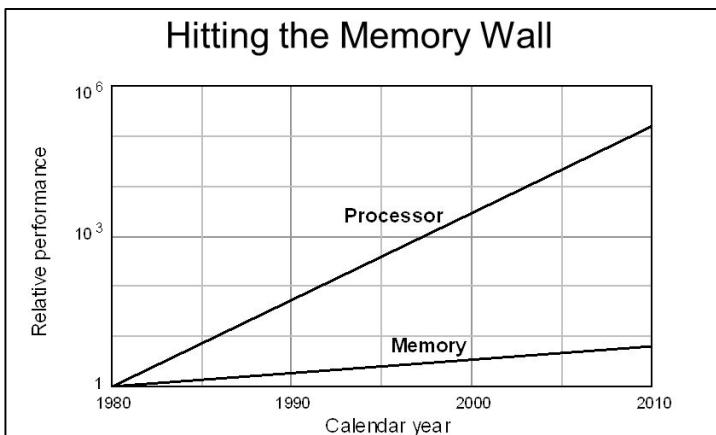
# 三、要充分考虑局部性



内存密度每三年翻一番，处理器逻辑器件的密度是每两年



逻辑成本的下降速度大于存储成本



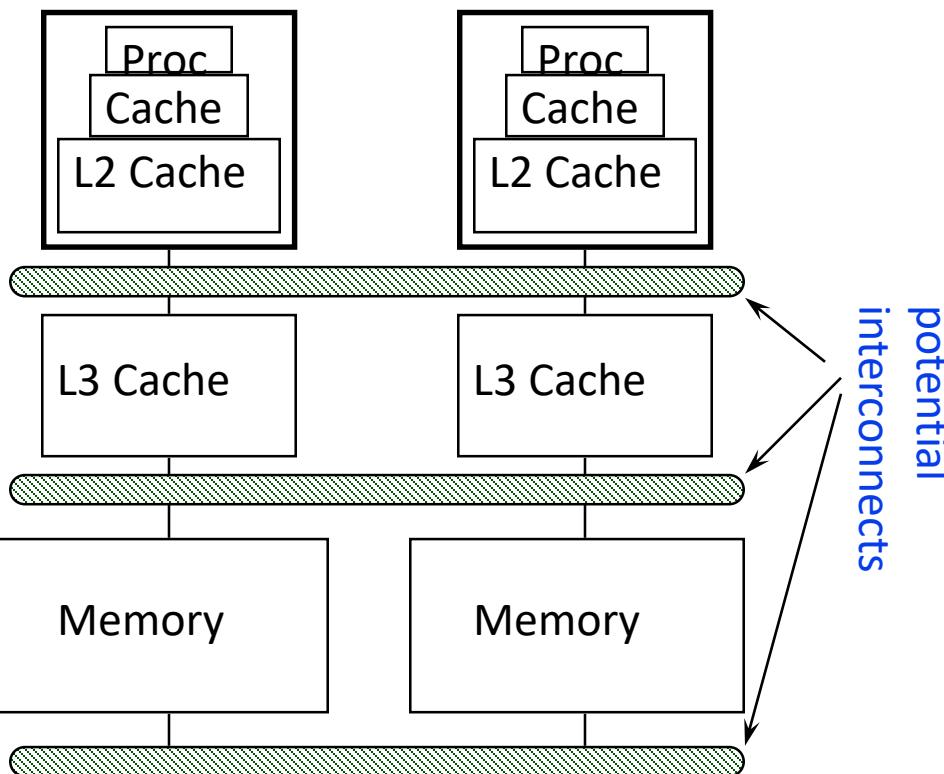
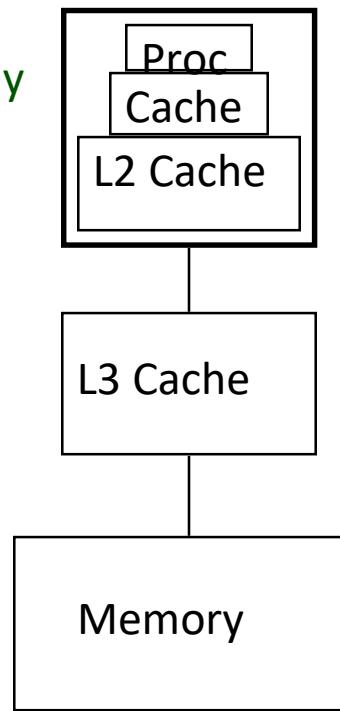
内存带宽与处理器性能之间的鸿沟越来越大

与计算部件相比

- 存储越来越小
- 存储越来越贵
- 存储越来越慢

# 局部性和并行

Conventional  
Storage  
Hierarchy

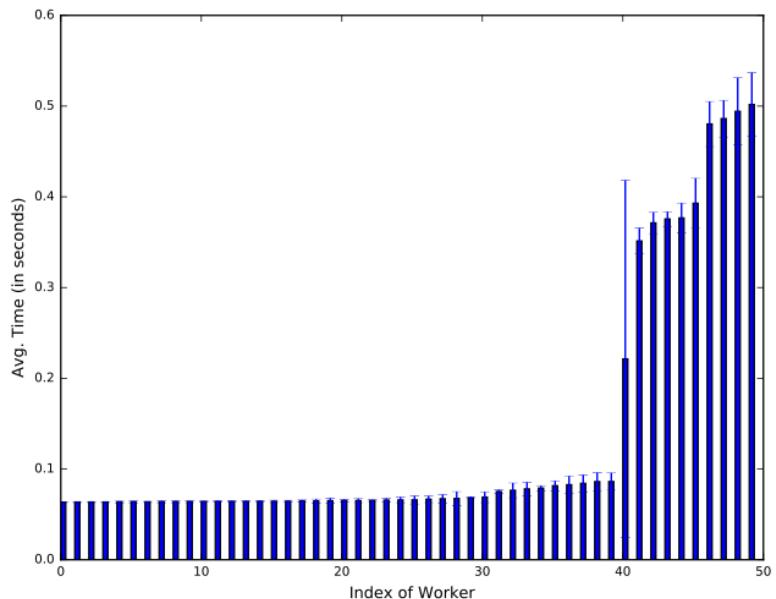


- 慢速存储器件容量大（DRAM），快速存储器件容量小（SRAM）
- 尽可能提前让数据从存储层次中流入快速存储器件以释放算力
- 在并行系统中，尽可能避免通信，不管是在哪个存储层次

# 四、要充分考虑负载均衡

## ■ 解决负载均衡问题的两种方法

- 在任务开始之前更加均衡的划分每个节点的工作量
  - “Static Load Balancing”
- 在任务执行过程中动态调整每个节点的工作量
  - “Dynamic Load Balancing,” eg work-stealing



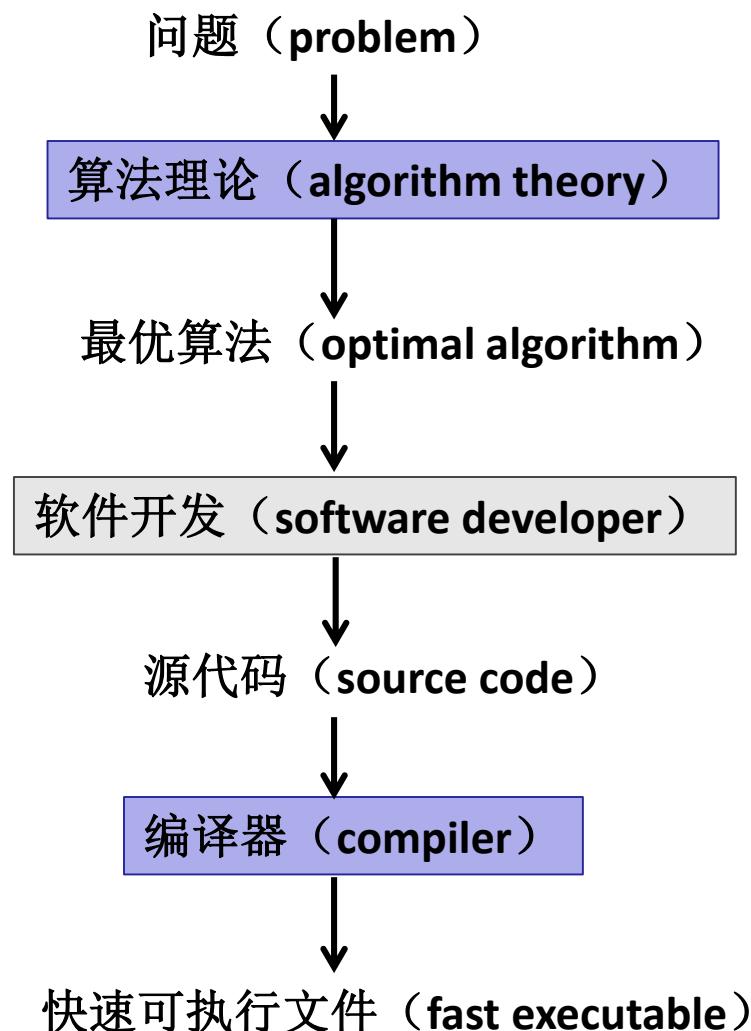
在参数服务器下典型的 straggler 场景

在分布式机器学习训练的参数同步中，部分节点显著比其他节点慢，所有节点都需要等待最慢的节点完成才可以进行参数同步，这个现象称为掉队现象（straggler），会将单个步骤拖慢数10倍以上。

# 五、结合体系结构的性能调优

- 两种类型的编程人员 → 两个软件层次
- 效率层 **Efficiency Layer (20% of programmers)**
  - 程序专家，主要工作是构建library, kernels, Frameworks, OS, ....
  - 要尽可能挖掘这个层次的峰值性能
- 生产力层 **Productivity Layer (80% of programmers)**
  - 领域专家/非专家程序员，利用现成的库或框架写应用
  - 他们不需要了解计算机系统实现的细节，并行是尽力而为
  - 愿意为高效编程牺牲一些性能
- 大家以后可能会工作在不同的层次
  - 尽管如此，我们还是希望所有程序员都对效率层有足够的了解，进而有效地利用并行性，提高程序性能

# 性能层实现快速代码到底有多难？



“Compute Fourier Transform”

算法

“Fast Fourier Transform (FFT)”  
 $O(n \log(n))$  or  $4n \log(n) + 3n$

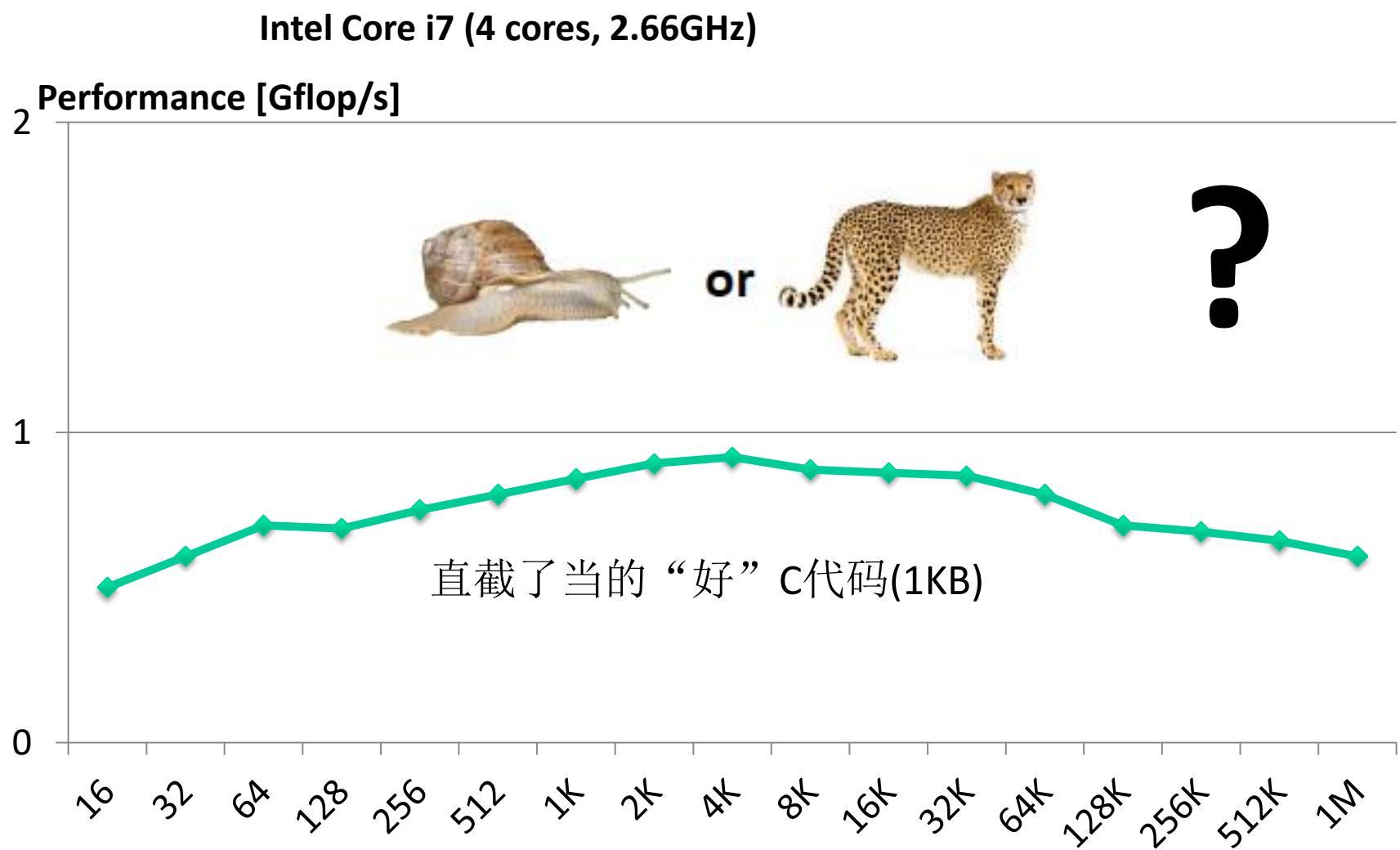
软件

A C/Fortran Function

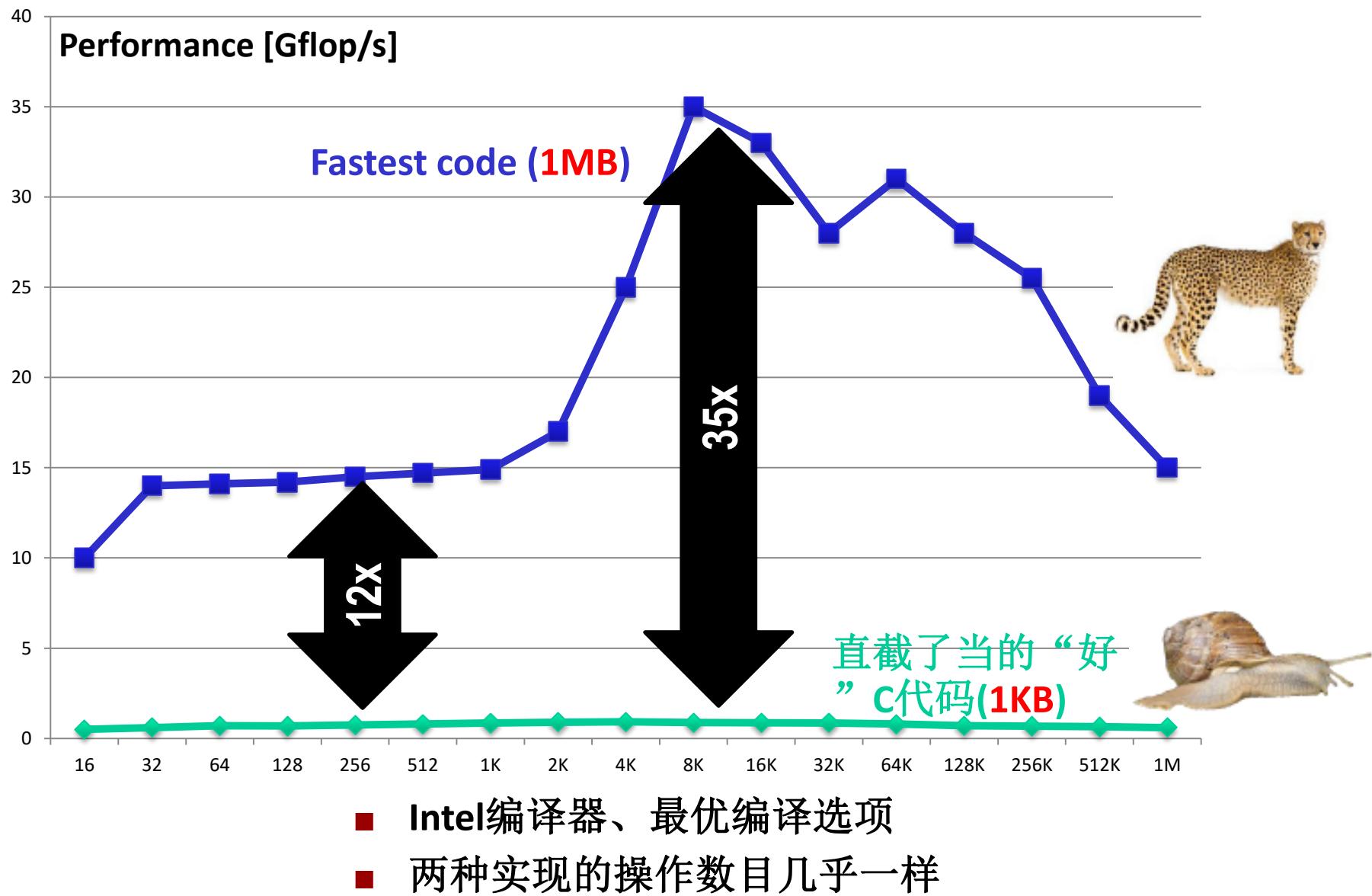
体系结构

*How well does this work?*

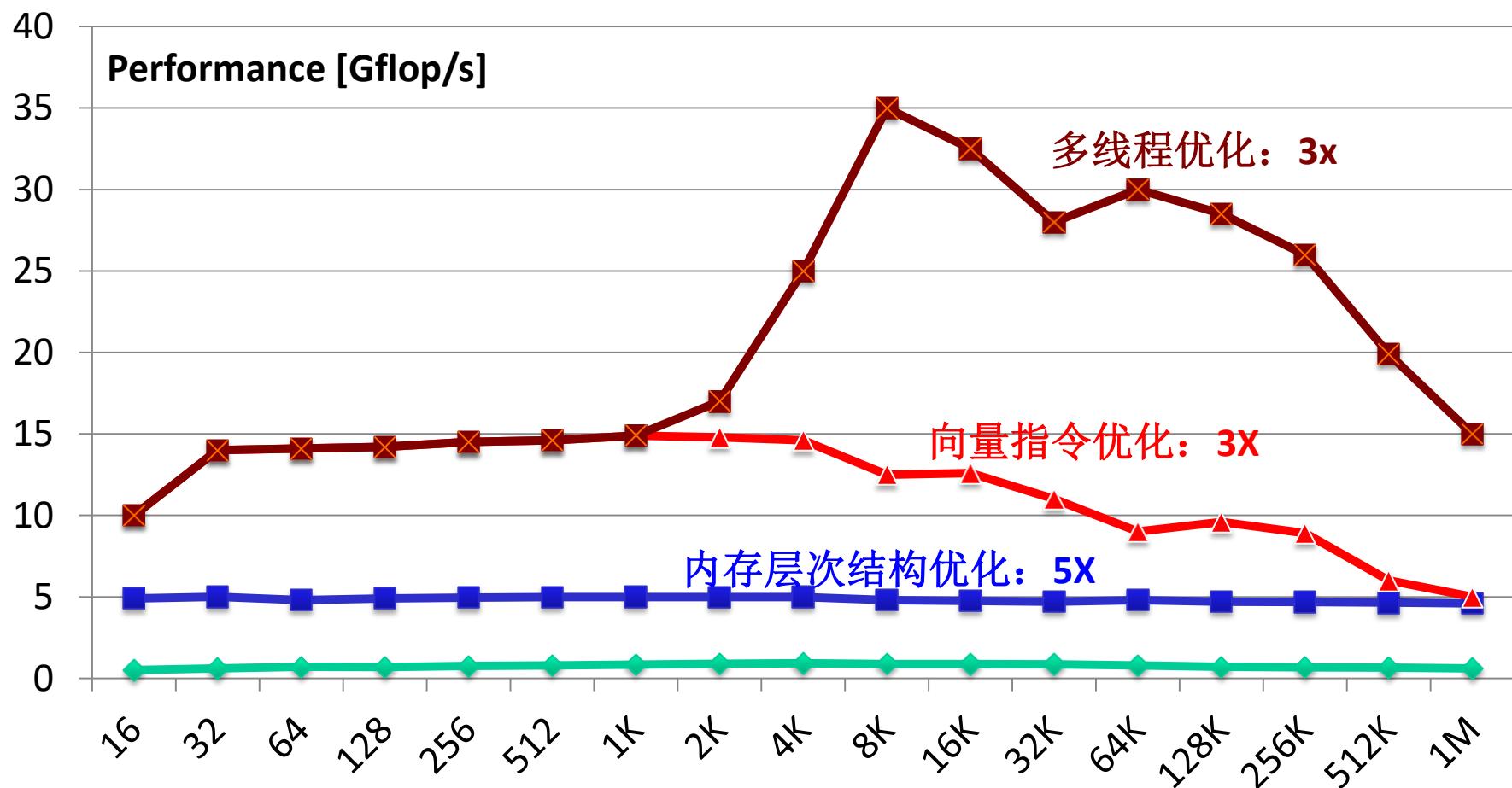
# 案例: DFT (single-precision)



# 案例: DFT (single-precision)



# DFT性能优化 (Intel Core i7, 4-core)



- 编译器很难自动实现这样的优化
- 手工优化任务繁重！

# 规律总结—1

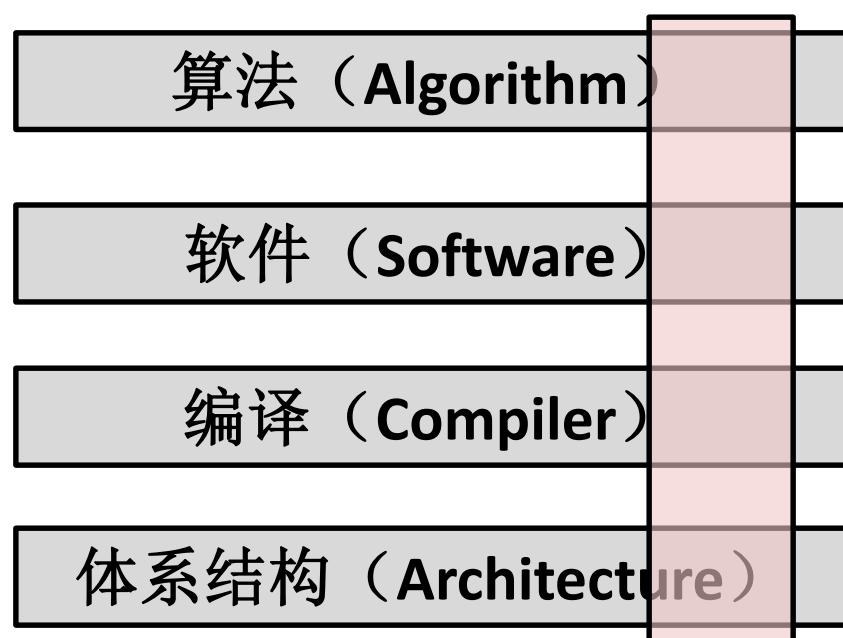
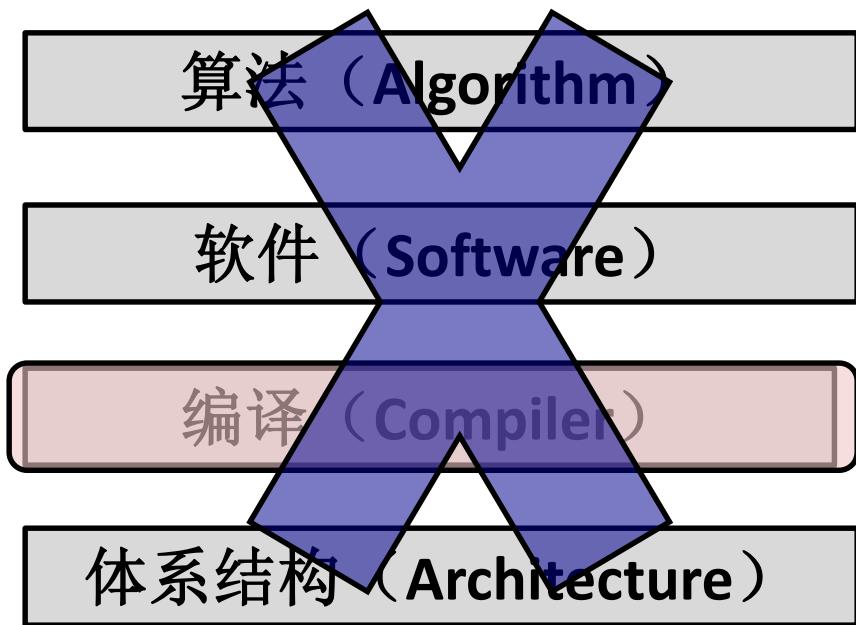
---

- 相同操作数的不同实现的性能差异显著，可达100倍
  - Cache不命中开销是计算操作开销的100倍
  - 是否利用到向量指令
  - 多核、众核
- 操作数的最小化  $\neq$  性能的最大化
- 对遗产代码免费获得加速比的时代已经结束
  - 需要开发更多并行性来提高性能

# 规律总结—2

- 写快速运行的代码是件不容易的工作
  - 要重点细致考虑内存层次结构
  - 熟悉向量指令
  - 开发高效的并行
  - 需要算法、编程和体系结构三个方面的专业知识
- 并行代码通常比较大
  - 会违反好的软件工程规范
- 编译器经常爱莫能助
  - 通常需要算法上内在的变化
  - 自动并行、向量化依然是NP问题
- 高性能通常不具备可移植性

# 获得性能需要多个层次的努力



# 总结

---

- 比较粗粒度的介绍了计算机中的并行
- 介绍了工程与科学应用和高性能计算机的紧密联系
- 介绍了高性能计算机的发展历史（SMP、MPP、Cluster）
- 介绍了并行编程为什么难以及一些初步的结论

# 并行计算的黄金时代—李国杰院士

- 获得2017年图灵奖的计算机体系结构大师John Hennessy和David Patterson指出，“无论是指令级处理器技术还是多核技术，通用处理器固有的低效率加上Dennard缩放比例定律和Moore定律的终结，使得处理器架构师和设计者已经不太可能在通用处理器中保持显著的性能改进速度”
- “下一个十年将出现一个全新计算机架构的“寒武纪”大爆发，学术界和工业界计算机架构师将迎来一个激动人心的时代。”
- 计算机体系结构的改进、创新必须和**并行算法、并行软件**的改进创新同步进行，越是高层的改进得到的性能和效率的提高越大。不管是突破摩尔定律还是冯•诺依曼瓶颈，都必须跨层次进行。**未来几十年是并行计算的黄金时代！**

# 我们这门课的主题1:

## Designing and writing parallel programs . . . that scale!

### ■ 并行思维

- 如何将任务合理分解以符合安全并行执行条件
- 将任务分配到每个处理器
- 管理好处理器之间的通信/同步，使其尽可能成为并行加速瓶颈

### ■ 如何将上述思考落实

- 使用主流的并行编程语言编写代码

## 课程主题2:

# Parallel computer hardware implementation: how parallel computers work

- 理解用于高效实现各种并行抽象的底层机制
  - 各种架构实现的性能特征
  - 设计权衡: 性能 vs. 实现的难易程度 vs. 成本
- 为什么需要了解硬件?
  - 因为机器的特性真的很重要
  - 因为你关心效率和性能（毕竟你在写并行程序！）

## 课程主题3:

# Thinking about efficiency

- **FAST != EFFICIENT**
- 因为你的程序在并行计算机上运行得更快了，但并不意味着它有效地使用了硬件
  - 在有 10 个处理器的机器上获得 2 倍加速是否是一个好的结果？
- 程序员视角: 利用机器提供的功能
- 机器设计人员视角: 选择正确的功能放入系统  
*(performance/cost, cost = silicon area?, power?, etc.)*

---

■ 谢谢！