并行计算对比分布式计算

- 并行计算:
  - 使用两个或更多的处理器（计算机），在一个系统内，同时工作以解决一个问题。

- 分布式计算:
  - 涉及多台计算机的计算，这些计算机相互远离，在计算问题或信息处理中各自发挥作用.

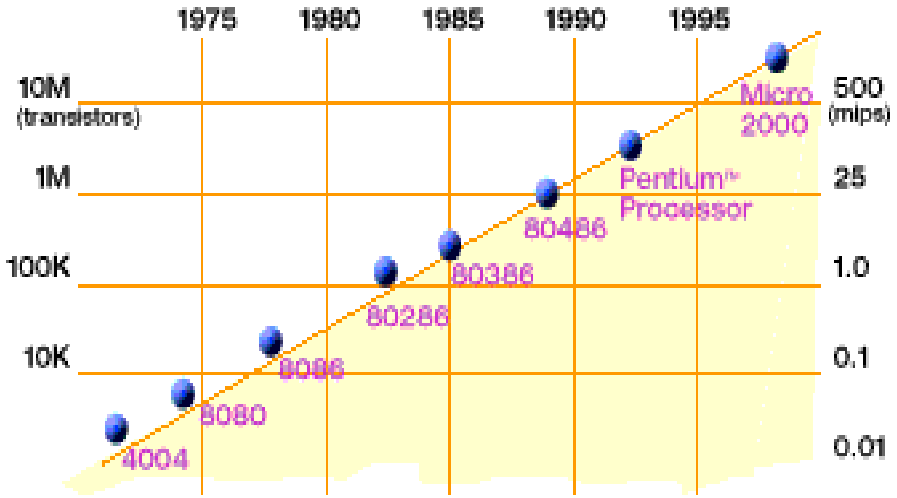
什么是并行计算机

- 并行计算机是一种支持并行编程的多处理器计算机系统。
- 两类并行计算机
  - **Multicomputer:** 并行计算机是由多台计算机和一个互连网络(信息传递)构成的。
  - **Centralized multiprocessor(**对称型多处理机**SMP):** 一种更高度集成的系统，所有**CPU**共享一个内存。(通过共享内存进行通信和同步)

PARALLEL V.S. DISTRIBUTED SYSTEMS

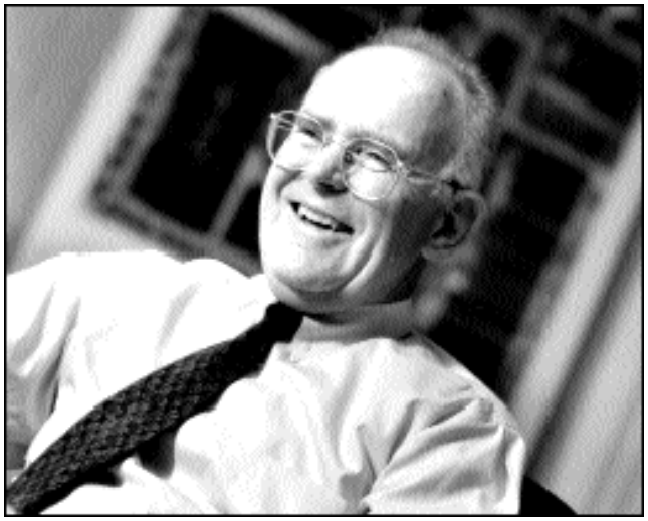
	并行系统	分布式系统
Memory	**Tightly coupled shared** memory UMA, NUMA 分布式内存 Message passing	分布式内存 Message passing, RPC, and/or used of distributed shared memory
Control	**Global clock** control SIMD, MIMD	**No global clock** control 需要同步算法
处理器连接速度和方式	Order of **Tbps** Bus, mesh, tree, mesh of tree, and hypercube (-related) network	Order of **Gbps** Ethernet(bus), token ring and SCI (ring), myrinet(switching network)
颗粒度	**Fine**	粗略的
可靠性	认为可靠	不认为可靠
Main focus	**Performance**(时间和规模) 科学计算	**Performance**(成本和可扩展性) 可靠性 信息/资源共享

TECHNOLOGY TRENDS: MICROPROCESSOR CAPACITY



2X transistors/Chip Every 1.5 years

Called "摩尔定律"

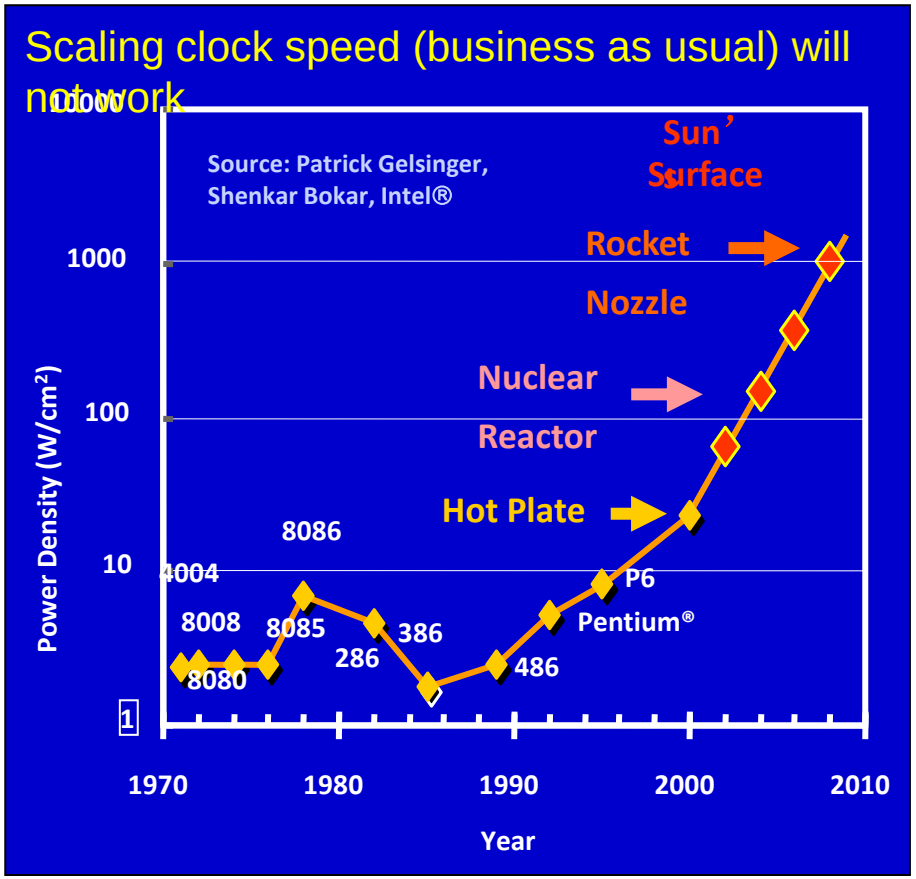


Gordon Moore (co-founder of Intel) predicted in 1965 that 半导体芯片的晶体管密度将大约每18个月翻一番。

Microprocessors have become smaller, denser, and more powerful.

LIMIT #1: POWER DENSITY

- 并行系统的效率更高
  - 动态功率和 V^2fC成正比
  - 增加频率 (f) 也会增加电压 (V) →cubic effect
  - Increasing cores increases capacitance (C) but only linearly
  - Save power by lowering clock speed

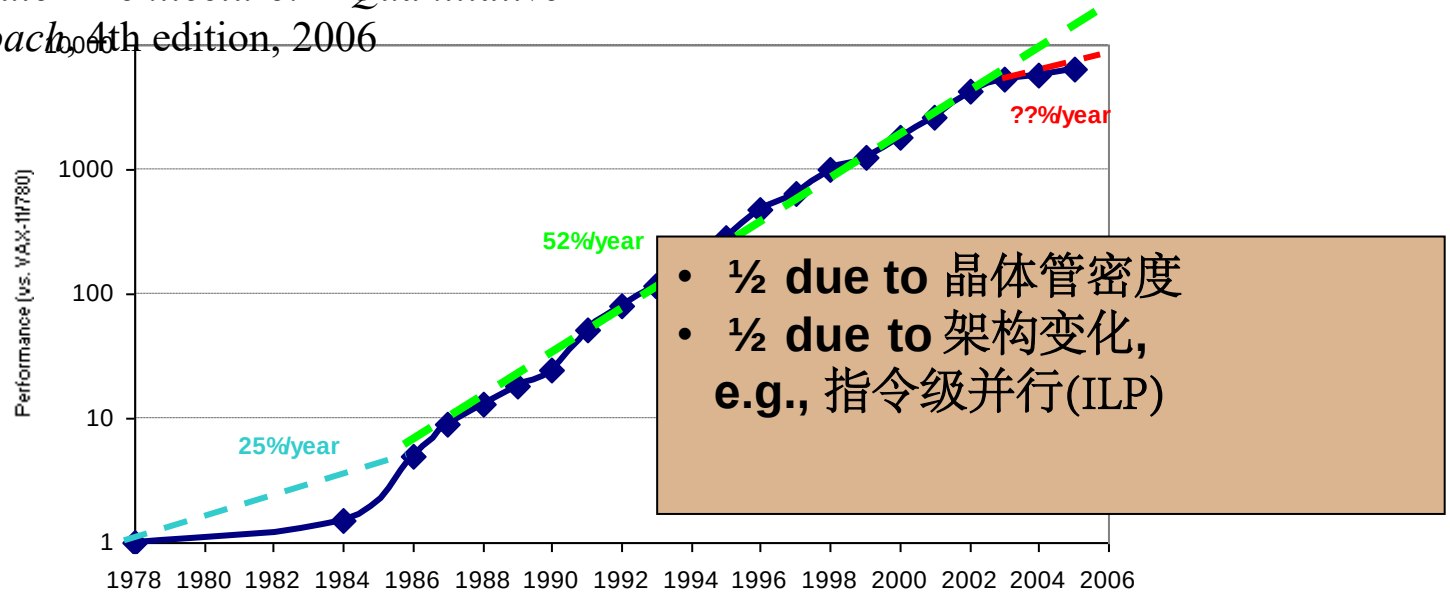


- High performance serial processors waste power
  - 预测, 动态依赖检查, etc. burn power
  - 隐含的并行性的发现
- 更多的晶体管，但不是更快的串行处理器

LIMIT #2: ILP TAPPED OUT

根据这里的SpecInt基准测试，应用性能每年增加52%。

From Hennessy and Patterson,
Computer Architecture: A Quantitative Approach, 4th edition, 2006



- ½ due to 晶体管密度
- ½ due to 架构变化,
  e.g., 指令级并行(ILP)

- VAX : 25%/year 1978 to 1986
- RISC + x86: 52%/year 1986 to 2002

LIMIT #2: ILP TAPPED OUT

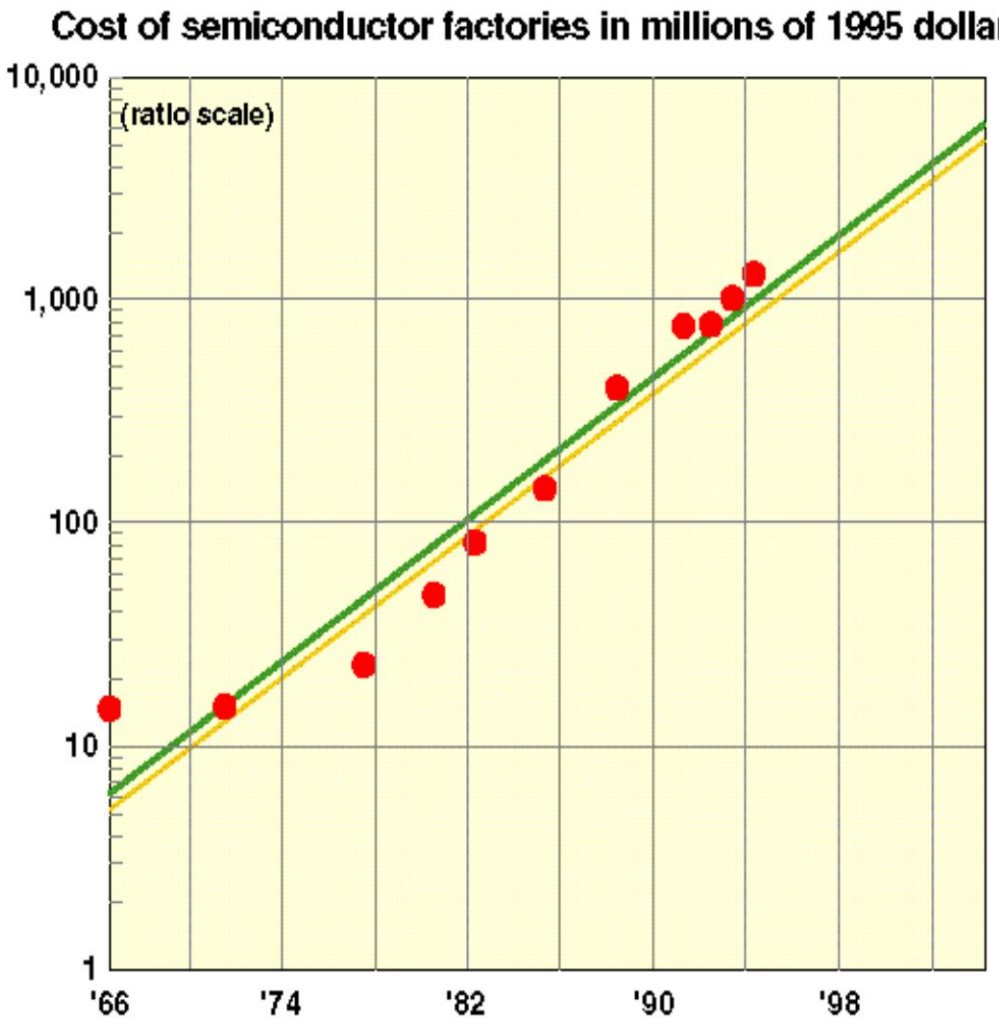
- 超标量 (SS) 设计是最先进的; 许多形式的并行性对程序员来说是不可见的
  - 多指令问题
  - 动态调度：硬件发现指令之间的并行性
  - 推测性执行：查看之前预测的分支
  - 非阻塞式缓存：允许多个未完成的内存操作

- **You may have heard of these before, but you haven't needed to know about them to write software**

- **Unfortunately, these sources have been used up**

LIMIT #3: 芯片产量

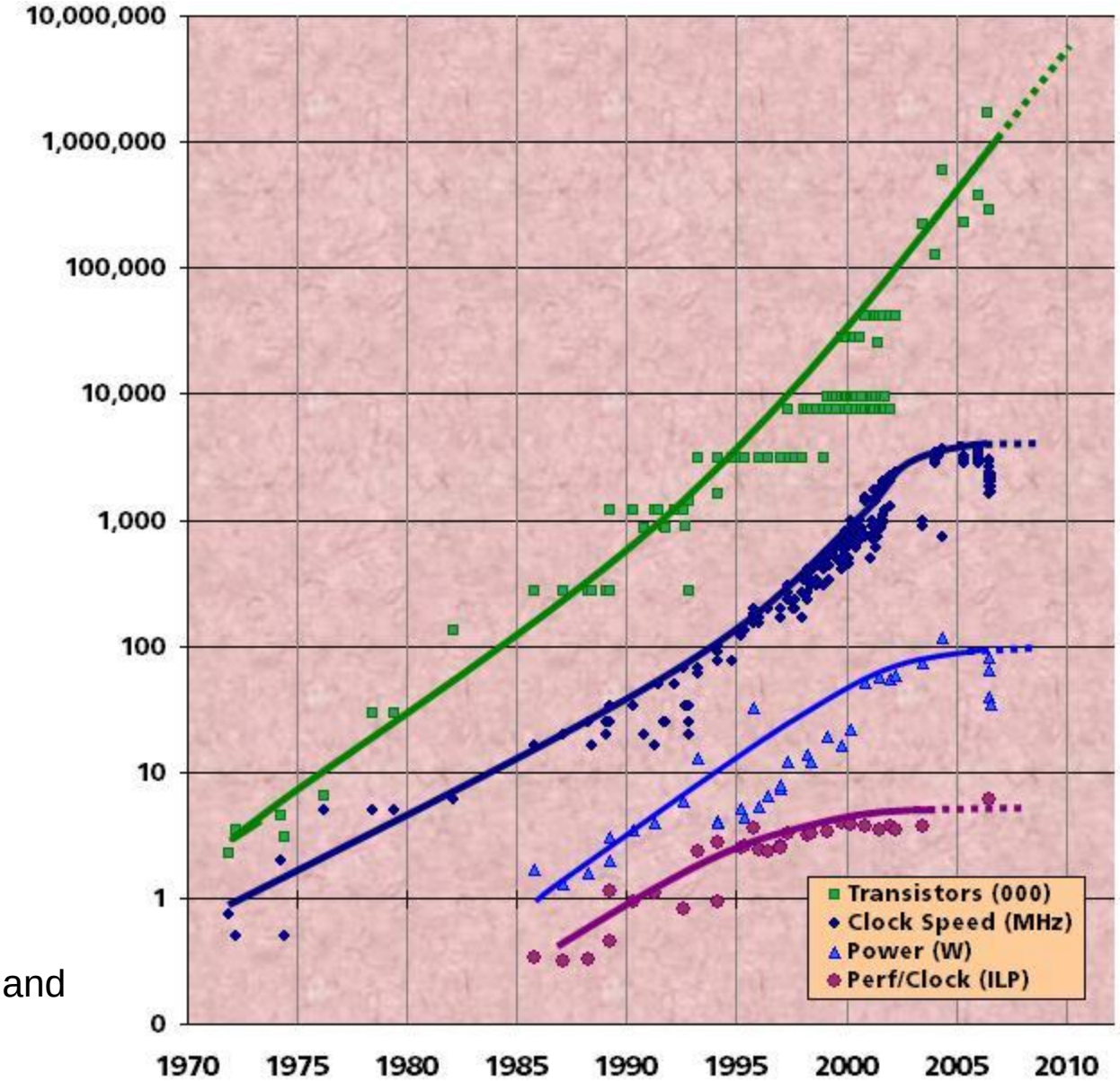
制造成本和产量问题限制了晶体管密度的使用



- Moore’s (Rock’s) 2nd law: 制造成本上升
- 产量下降
- Parallelism can help
  - 小而简单的处理器容易设计和验证
  - 可以使用部分工作的芯片
  - E.g., 出售屏蔽了损坏部件的CPU

CURRENT SITUATION

- 芯片密度正在持续增加
  - 时钟速度没有增加
  - 处理器内核的数量可能翻倍
- 几乎没有隐藏的并行性（ILP）可寻。
- 并行性必须暴露在软件中并由软件管理



Source: Intel, Microsoft (Sutter) and Stanford (Olukotun, Hammond)

PARALLELISM IN 2021?

- 这些论点不再是理论上的了
- 所有主要的处理器供应商都在生产多核芯片
  - 每台机器都将很快成为并行机器
  - 要保持性能翻番，并行性必须加倍
- 哪些（商业）应用可以使用这种并行性？
  - 它们是否必须从头开始重写？
- 所有的程序员都必须是并行程序员吗？
  - 需要新的软件模型
  - 尝试对大多数程序员隐藏复杂性--最终会的
  - 与此同时，需要了解它
- 计算机行业押注于这一重大变革，但并不具备所有的答案

摩尔定律重新解释

- 每块芯片的内核数量每两年可翻一番
- 时钟速度不会增加（可能减少）
- 需要处理具有数百万并发线程的系统
- 需要处理芯片间的并行性以及芯片内的并行性

HOW DO WE WRITE PARALLEL PROGRAMS?

- 任务并行化
  - 将解决问题所进行的各种任务在核心之间进行划分。
- 数据并行化
  - 将用于解决问题的数据在各核心之间进行分割。
  - 每个核心对它的那部分数据进行类似的操作。

DIVISION OF WORK-THE GLOBAL SUM

- 全局和的第一部分是数据并行的一个例子。

  – 每个处理器对其分配的元素进行大致相同的操作：它通过调用**Compute_next_value**计算所需的值，并将它们加在一起。

- 全局和的第二部分是一个任务并行的例子。

  – 有两项任务：接收和增加处理器的部分和，这是由主处理器执行的；将部分和交给主处理器，这是由其他处理器执行的。

COORDINATION

- 处理器通常需要协调他们的工作。
- 通信 – 一个或多个处理器将其当前的部分总和发送给另一个处理器。
- 负载均衡 – 在各处理器之间均匀地分担工作，使一个处理器不至于负荷过重.
- 同步 – 因为每个处理器都以自己的速度工作，所以要确保处理器不会比其他处理器领先太远。