并行计算

ch1 并行计算与并行计算机结构模型

1.3 多核处理器及线程级并行

- 何谓多核处理器及意义 -P10
- 微处理器中的并行方式 PC0 66-67
  - ILP：指令级并行
    - 当指令中不存在相关时，它们在流水线上是可以重叠执行的，我们重叠执行指令以提高性能。而这种指令序列存在的潜在并行性称为指令级并行。
    - 如何开发实现ILP？静态开发（依赖软件）以及动态开发（依赖硬件）
  - TLP：线程级并行，线程是最小的运行单元，一个进程中可以并发多个线程（MIMD）。 线程级并行是比指令级并行更高的层次，可以显式地将几个线程并行执行，共享执行单元
  - 多任务OS：早期的UNIX操作系统的多任务是靠分时（time sharing）机制实现的，现在有些UNIX除了具有分时机制外，还加入了实时（Real Time）多任务能力，用于像实时控制、数据采集等实时性要求较高的场合。系统在执行多任务时，CPU在某一时刻只能执行一个任务，但UNIX操作系统将CPU时间分片，并把这些时间片分别安排给多个进程。因为CPU运行很快，在操作者看来，所有程序(进程)都在同时运行。现在的多任务实时操作系统，即RTOS，包括：uc/os,linux,wince
  - SMT：同时多线程，也称同步多线程
    - 同步多线程（SMT）是一种在一个CPU 的时钟周期内能够执行来自多个线程的指令的硬件多线程技术。

      本质上，同步多线程是一种将线程级并行处理（多CPU）转化为指令级并行处理（同一CPU）的方法。

      同步多线程是单个物理处理器从多个硬件线程上下文同时分派指令的能力。同步多线程用于在商用环境中及为周期/指令（CPI）计数较高的工作负载创造性能优势。

      处理器采用超标量结构，最适于以并行方式读取及运行指令。

      同步多线程使您可在同一处理器上同时调度两个应用程序，从而利用处理器的超标量结构性质。
    - 是一种提高具有硬件多线程的超标量CPU整体效率的技术。同时多线程 允许多个独立的执行线程更好地利用现代处理器架构提供的资源。
    - 名称中的“多线程”有些暧昧，因为它不仅可以在一个CPU核心上同时执行多个线程，也可以是多个任务（哪怕不同分页表、不同任务状态段、不同分级保护域、不同I/O权限等等）。尽管运行在同一个核心，它们彼此间完全分离。多线程在概念上类似抢占式多任务处理，但在现代的超标量处理器中以线程级执行来实现。

      同时多线程（SMT）是多线程的两个主要实现之一，另一个是**时间多线程**（也称超线程）。在时间多线程中，同一时间只有一个线程的指令可以在任何指定流水线阶段中执行。在同时多线程中，多于一个线程的指令可以在任何指定的流水线阶段中同时执行。完成它在基本的处理器架构上没有太多改变：所需的主要添加是在一个周期中从多个线程获取指令的能力，以及一个更大的寄存器文件来保存来自多个线程的数据。并发线程的数量可以由芯片设计者决定。常见模式是每个CPU核心有两个并发线程，但一些处理器的每个核心支持最多八个并发线程。

因为该技术实则是一种效率解决方案，并且不可避免地增加了共享资源的冲突，所以测量或认同该解决方案的有效性可能有困难。但是，使用并行的本机和受管理的工作负载在传统130 nm到32 nm的Intel SMT（超线程）实现上测量SMT的能效发现，在45 nm和32 nm实现中，SMT极为节能，即使使用了inorder Atom处理器[ASPLOS'11]。在现代系统中，SMT有效使用非常少的额外动态功率达成并发。也就是说，即使在性能增益最小时，功率的节省也是可观的。[来源请求]

一些研究人员已经表明，额外的线程可用来主动为高速缓存等提供共享资源，从而提高另一个单线程的性能，并称这表明SMT不只是一个效率解决方案。SMT的其他用法还有提供冗余计算，以用于某种程度的错误检测和恢复。

但在大多数情况下，SMT是有关隐藏内存延迟、提高效率，以及增加每个所用硬件的计算吞吐量。

- 在处理器设计中，有两种方法以较少的资源需求来增加芯片并行性：一种是尝试利用指令级并行性（ILP）的超标量技术；另一种是利用线程级并行性（TLP）的多线程方式。

  超标量意味着在一个处理器芯片内以线程级并行性（TLP）同时执行多个线程的多个指令。有很多方法来支持芯片内的多个线程，即：

  - 交叉多线程：交错发出不同线程的多个指令，也称为时间多线程。它可以进一步分为细粒度多线程与粗粒度多线程，这取决于交错发出的频率。**细粒度多线程**——例如在一个桶处理器——在每个周期后发出不同线程的指令，而**粗粒度多线程**仅在当前执行线程导致一些长延迟事件（如页错误等）时才切换到从另一线程的发出指令。粗粒度多线程对于线程之间较少的上下文切换是更常见的。例如，英特尔的Montecito处理器使用粗粒度多线程，而Sun的UltraSPARC T1使用细粒度多线程。对于每个内核只有一个流水线的处理器，交叉多线程是唯一可能的方法，因为它在每个周期最多发出一条指令。
  - 同时多线程（SMT）：在一个周期中发出多个线程的多个指令。处理器必须为超标量。
  - 芯片上多处理器（CMP或多核心处理器）：将两个或多个处理器集成到一个芯片，每个处理器独立地执行线程。
  - 任何多线程/SMT/CMP的组合。
    区分它们的关键因素是查看处理器在一个周期中可以发出多少条指令以及指令来自哪些线程。例如，Sun Microsystems的UltraSPARC T1（2005年11月14日发布之前称为“Niagara”）是一个多核处理器，它结合了细粒度多线程技术而不是同时多线程，因为每个核心一次只能发出一条指令。
- 处理器的硬件多线程技术主要有两类交替多线程/时间多线程(Temporal Multi-Threading)，和同时多线程(Simultaneous Multi-Threading)。区别在于前者的多个线程是交替执行的，后者的多个线程是同时执行的。
  时间多线程根据线程切换的时机又可以分为细粒度和粗粒度，细粒度是指每执行一条指令就切换，而粗粒度是指当访存导致CPU停顿(stall)时才进行切换。
  硬件多线程技术主要解决执行部件(整数，浮点，访存等）的利用率，提高处理器吞吐量。
  执行部件的浪费分为两种:水平浪费和垂直浪费。**水平浪费**是指同一时钟周期，不是所有执行部件都在使用，而**垂直浪费**是指由于CPU停顿，导致某个时钟周期中所有执行部件都没有在使用。
  时间多线程只能解决垂直浪费，而同时多线程可以解决垂直和水平浪费。

- Intel超线程技术
  - 超线程技术把多线程处理器内部的两个逻辑内核模拟成两个物理芯片，让单个处理器就能使用线程级的并行计算，进而兼容多线程操作系统和软件。超线程技术充分利用空闲CPU资源，在相同时间内完成更多工作。
    虽然采用超线程技术能够同时执行两个线程，当两个线程同时需要某个资源时，其中一个线程必须让出资源暂时挂起，直到这些资源空闲以后才能继续。因此，超线程的性能

并不等于两个CPU的性能。而且，超线程技术的CPU需要芯片组、操作系统和应用软件的支持，才能比较理想地发挥该项技术的优势

- 运作方式：每个单位时间内，一个单运行管线的CPU只能处理一个线程（操作系统：thread），以这样的单位进行，如果想要在一单位时间内处理超过一个线程是不可能的，除非是有两个CPU的实体单元。双核心技术是将两个一样的CPU放置于一个封装内（或直接将两个CPU做成一个芯片），而英特尔的多线程技术是在CPU内部仅复制必要的资源、让两个线程可同时运行；在一单位时间内处理两个线程的工作，模拟实体双核心、双线程运作。
  Intel自Pentium开始引入超标量、乱序运行、大量的寄存器及寄存器重命名、多指令解码器、预测运行等特性；这些特性的原理是让CPU拥有大量资源，并可以预先运行及平行运行指令，以增加指令运行效率，可是在现实中这些资源经常闲置；为了有效利用这些资源，就干脆再增加一些资源来运行第二个线程，让这些闲置资源可执行另一个线程，而且CPU只要增加少数资源就可以模拟成两个线程运作。
  P4处理器需多加一个Logical CPU Pointer（逻辑处理单元）。因此P4 HT的die的面积比以往的P4增大了5%。而其余部分如ALU（整数运算单元）、FPU（浮点运算单元）、L2 Cache（二级缓存）并未增加，且是共享的。
- 优缺点：

  优点
  1.超线程技术的优势在于同时进行多任务批处理工作，尽管支持超线程技术的软件不多，也只有少数的软件可以享受到由超线程技术带来的性能提升，但是这符合今后软件等技术的发展方向，今后更多的软件将受益于超线程技术。
  2.从来看，部分客户可以发觉在运行某些特定软件时，超线程技术让系统有了30%的性能提升，为超线程技术优化的软件都能够享受到超线程技术的好处。
  3.客户同时运行两个以上的软件时候，将可以明显的感受到这两个软件的性能都得到提升相比关闭超线程技术的情况下都有很大的提升，超线程技术的效率优势只有在多任务操作时候才能得到发挥。
  4.支持超线程技术的Windows XP操作系统，其中的很多系统软件都已经针对超线程技术优化过，因此在使用Windows 操作系统的时候可以很好的享受到超线程技术带来好处。
  缺点
  1.因为超线程技术是对多任务处理有优势，因此当运行单线程运用软件时，超线程技术将会降低系统性能，尤其在多线程操作系统运行单线程软件时将容易出现此问题。
  2.在打开超线程支持后，如果处理器以双处理器模式工作，那么处理器内部缓存就会被划分成几区域，互相共享内部资源。对于不支持多处理器工作的软件在双处理器上运行时出错的概率要比单处理器上高很多。
  3.因为很多工作站软件为Windows 2000操作系统进行过优化，但是采用Windows 2000这样的操作系统的工作站无法完全利用超线程技术的优势，也带来不了高的工作效率
  4.超线程技术只能提高40%左右的性能（测评时可以看成50%，即Core i3 的执行效率为3核速率，Core i5 4核 HT与Core i7 的执行效率为6核速率）
- 与多核心的区别：超线程技术与多核体系结构的区别如下：
  - ①超线程技术是通过延迟隐藏的方法，提高了处理器的性能，本质上，就是多个线程共享一个处理单元。因此，采用超线程技术所获得的性能并不是真正意义上的并行。从而采用超线程技术获得的性能提升，将会随着应用程序以及硬件平台的不同而参差不齐。
  - ②多核处理器是将两个甚至更多的独立执行单元，嵌入到一个处理器内部。每个指令序列（线程），都具有一个完整的硬件执行环境，所以各线程之间就实现了真正意义上的并行。
- 超线程技术与多核技术相结合可以给应用程序带来更大的优化空间，进而极大地提高系统的吞吐率。

1.4 并行计算机体系结构

- 并行计算机结构模型
  - SISD, SIMD, MISD, MIMD P15
  - SIMD, PVP, SMP, MPP, COW(Cluster), DSM P14-15
  - 对比表格 P16-17
- 并行计算机访存模型
  - UMA,NUMA,COMA,CC-NUMA,NORMA
  - 不同存储结构的并行机系统(P20 图 1.11)

1.5 更多的并行计算概念

- 一个 MFLOPS (megaFLOPS) 等于每秒1百万 (=10^6) 次的浮点运算，
  一个 GFLOPS (gigaFLOPS) 等于每秒10亿 (=10^9) 次的浮点运算，
  一个 TFLOPS (teraFLOPS) 等于每秒1万亿 (=10^12) 次的浮点运算，
  一个 PFLOPS (petaFLOPS) 等于每秒1千万亿 (=10^15) 次的浮点运算。
- Rmax是实测的最高速度，Rpeak是理论上的最高速度。总会有些实际原因使超算的性能达不到理论上限，所以Rmax总是小于Rpeak

  Linpack是最流行的用于测试高性能计算机系统浮点性能的测试软件。通过用高斯消元法求解N元一次稠密线性代数方程组的测试，评价高性能计算机的浮点性能。
  而Rmax和Rpeak是linpack软件提供的两个参考值。
  它们的单位为：flop/s或者tflop/s，分别对应：每秒浮点运算次数和每秒万亿（10的12次方）浮点运算的次数
  Rmax - Maximal LINPACK performance achieved linpack实际测试中达到的最大性能
  Rpeak - Theoretical peak performance 理论最大性能。由理论推算出来的性能，不可能达到。
  Rpeak的理论计算值为：CPU主频×CPU每个时钟周期执行浮点运算的次数×系统中CPU核心数目

ch2 并行计算系统互联与基本通信操作

- 并行计算机的互联方式
  - 静态互连：LA(LC), MC, TC, HC； (P42 表 2.1 各种网络特性表)
  - 动态互连：us, Crossbar Switcher, MIN(Multistage Interconnection Networks)
  - 特别地， 标准网络互连： FDDI, 快速以太网, Myrinet, InfiniBand

ch4 并行计算性能评测

ch5 并行算法与并行计算模型

- PC5 - 6，SPMD（并行循环、并行块）、MPMD、同步与通信

ch6 并行算法基本设计策略

ch7 并行算法常用设计技术

ch8 并行算法一般设计过程

ch9 稠密矩阵运算

ch10 线性方程组的求解

ch11 快速傅里叶变换 FFT

ch13 并行程序设计基础

ch14 共享存储系统并行编程

ch15 分布存储系统并行编程

GPU 与 CUDA 编程

其他

- **SPMD**的（单一的过程中，多个数据或单个程序，多数据）计算，是实现并行的技术，它是一个子类的MIMD。任务是分裂，并同时运行在多个处理器上，以不同的输入更快地获得结果。 SPMD的是最常见的并行编程风格。