利用 GPU 进行高性能数据并行计算

■文 / 丁艺明 刘 波

数据库技术的成熟、数据挖掘应用、生物基因技术的发展、历史数据的几何级膨胀等使高性能计算(High Performance Computing，HPC)成为必要。虽然通过创建分布式系统可以解决部分大型计算的问题，但是分布式系统有通信开销大，故障率高；数据的存取结构复杂，开销大；数据的安全性和保密性较难控制等弱点。随着计算机处理器，特别是GPU (Graphical Processing Unit)计算能力的飞速提高，高性能计算正在逐步进入桌面（低端）领域，我们开始探讨并行编程模型与并行编程等软件技术 。

GPU 强大计算能力

早期的3D游戏，显卡只是为屏幕上显示像素提供一个缓存，所有的图形处理都是由CPU单独完成。图形渲染适合并行处理，擅长于执行串行工作的CPU实际上难以胜任这项任务。直到1995年，PC机领域第一款GPU 3dfx Voodoo出来以后，游戏的速度、画质才取得了一个飞跃。GPU的功能更新很迅速，平均每一年多便有新一代的GPU诞生，运算速度也越来越快。以下表1表明2006年度GPU与CPU价格相当的情况下，GPU的计算能力已经远远高于CPU的计算能力。注：GFLOPS为每秒浮点运算能力。

表1 CPU/GPU计算能力比较

Intel Core2Due Woodcrest	GeForce 8800 GT	运算能力比较
24 GFLOPS	520 GFLOPS	GPU快21.6倍

为什么GPU跑得快？

GPU具有两点主要特征：超长流水线与并行计算。

如果装配一台汽车需要10个时间单元，将它分成10个流水线阶段，每个阶段分配一个时间单元，那么一条装配线每一个时间单元就可以生产一辆汽车。显然流水线模式的生产在理想状况下要比串行方式快了十倍。

GPU通过单指令多数据(SIMD)指令类型来支持数据并行计算。参见图1，在单指令多数据流的结构中，单一控制部件向每条流水线分派指令，同样的指令被所有处理部件同时执行。例如NVIDIA 8800GT显卡中包含有14组多处理器(Multiprocessor)，每组处理器有8个处理单元(Processor)，但每组多处理器只包含一个指令单元(Instruction Unit)。

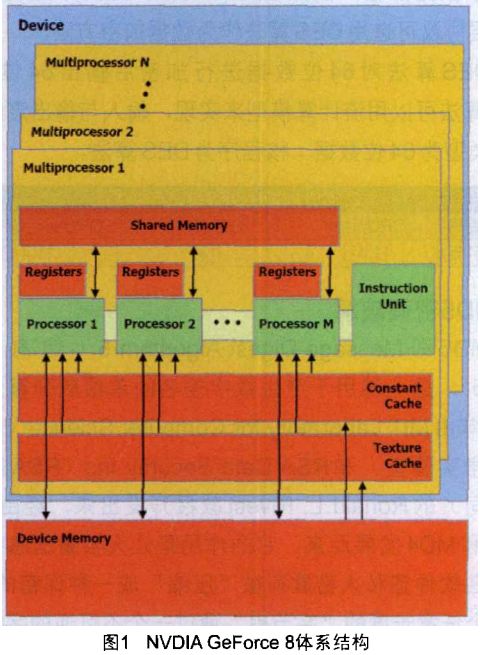

图1 NVDIA GeForce 8体系结构

GPU流式编程模型

GPU编程以流式编程模型为基础，它以允许高效计算和通信的方式构造程序。在流式编程模型中，所有数据都表现为流。我们把流定义为具有相同数据类型的数据的有序集。数据类型可以是简单的(整数或浮点数流)或复杂的(点或三角形或变换矩阵流)。流可以是任意长度，如果

流很长(流中有上百或更多的元素)，那么流上的操作并行度将很高。流上允许的操作包括复制它们，从它们导出子流，用一个单独的索引流索引入它们，以及用核在它们上执行计算。GPU程序称为核，核操作整个流，获取一个或多个流作为输入并产生一个或多个流作为输出。核的特征是它操作多个流上的所有元素而不是独立的元素。

CPU程序以异步的方式调用GPU核程序。GPU作为CPU的协处理器(Coprocessor)提供服务。

实验

我们的实验基于CUDA的SDK以及C语言编译器在8800GT显卡上开发运行的。CPU版程序为双线程，用VC++6.0开发，运行于Intel Core2Duo主频为2.6G赫兹。实验结果中，GPU版程序运行时间包括输入数据流和输出数据流上传和下载到显卡的I/O时间。

1 DES 编解码

对称密钥加密算法DES(Data Encryption Standard)是由IBM公司在70年代发展起来的，并经过政府的加密标准筛选后，于1976年11月被美国政府采用，DES随后被美国国家标准局和 美国国家标准协会(American National Standard Institute，ANSI) 承认。DES算法广泛应用于数据加密。例如SSL (Secure Socket Layer)安全套接层协议可选用DES算法作为数据加密方法。

DES算法对64位数据进行加密后输出64位数据。DES算法可以用流计算模型来实现，输入与输出流的基本数据类型为64位数据。核程序为DES算法。

表2 CPU/GPU DES编码实验结果

DES 编码	CPU 时间	GPU时间	GPU版比CPU版快
64 M字节	11.4秒	1秒	11.4 倍

2 MD5密码破解

MD5 即 Message-Digest Algorithm 5（信息-摘要算法5)，是一种用于产生数字签名的单项散列算法，在1991年由MIT Laboratory for Computer Science（IT计算机科学实验室）和RSA Data Security Inc（RSA数据安全公司）的Ronald L. Rivest教授开发出来，经由MD2、MD3和MD4发展而来。它的作用是让大容量信息在用数字签名软件签私人密匙前被“压缩”成一种保密的格式。将一个任意长度的“字节串”通过一个不可逆的字符串变换算法变换成一个128比特的大整数，换句话说就是，即使你 看到源程序和算法描述，也无法将一个MD5的值变换回原始的字符串。

MD5典型应用是对一段消息产生一摘要，以防止被篡改。

MD5还广泛用于加密和解密技术上。例如在UNIX系统中用户的密码就是以MD5（或其它类似算法）经加密后存储在文件系统中。当用户登录时，系统把用户输入的密码计算成MD5值，然后再去和保存在文件系统中的MD5值进行比较，进而确定输入的密码是否正确。

在我们的程序中，允许用户输入一长度为五的密码的MD5值，每位密码变化范围是A~Za~z[]\^_`{}|~，共64种字符。穷举所有的密码并用MD5算法得到所有的MD5值，与用户输入的MD5值比较，若枚举的密码MD5值与用户输入匹配，输出该密码。

MD5 破解可以用流计算模型来实现，输入流基本数据为长度为5个字符的密码，可以枚举出来。所有基于密码产生的128比特 MD5值可看为中间结果流。核程序为MD5算法。最后，把中间结果和输入的MD5值比较的布尔值组成最终结果流。

表3 CPU/GPU MD5 破解实验结果

MD5 破解	CPU 时间	GPU时间	GPU版比CPU版快
穷举1G种可能	201秒	15.3秒	13.1 倍

3 字符串匹配

字符串匹配问题可以理解为从给定的符号序列中找出具有某种属性的模式。字符串匹配广泛应用于信息检索和计算生物学领域。人们对字符串匹配问题重视程度与日俱增，之所以有这些现象，不仅因为需要处理的文本规模越来越大，而且由于需要在文本中进行越来越复杂的搜索。

本实验随机产生64M字节的文本和64个长度为8的关键字，找出在输入的文本中出现的关键字。本实验的程序采用的是Boyer-Moore-Horspool-Sunday（BMHS）字符串匹配算法。

字符串匹配问题用流计算模型来实现，输入流为64M字节文本。核程序为分别对64个关键字进行字符串匹配的算法。把64个关键字字符串匹配结果的布尔值组成结果流。

值得一提的是，对每个关键词的搜索在窗口内进行，窗口的大小于关键词的长度相等，窗口沿着文本向右滑动。BMHS算法将窗口内文本的最后一个字符(L)和关键字的最后一个字符进行比较。如果相等，则需要在搜索窗口中从后向前对文本和关键字进行比较，直到完全相等或者在某个字符处不匹配。然后，都将根据L在关键字的下一个出现的位置将 窗口向右移动。对每个关键词移动的距离，也就是下次读取字符的位置，是不一样的。参见图2 NVDIA GeForce 8体系结构，每次从GPU设备存储器(Device Memory)读取数据需要耗费400~600个时钟周期[1]。本实验把输入文本和一两维图像(纹理)进行绑定，这样也就利用了纹理缓存(Texture Cache)来提高设备存储器的访问速度，减少大量的I/O时间。

表4 CPU/GPU字符串匹配实验结果

字符串匹配	CPU 时间	GPU时间	GPU版比CPU版快
64 M字节文本	14.5秒	1.4秒	10 倍

4 实验结果小结

吞吐量可由输入数据大小比上处理器运行时间。从图2 CPU/GPU吞吐量实验结果表明，GPU在通用计算方面的性能能够比CPU快10倍以上。MD5密码破解程序的I/O最小，DES编码程序次之，字符串匹配程序I/O最大。相对于CPU版程序吞吐量，GPU版MD5密码破解相对性能最高，DES编码程序次之，虽然字符串匹配程序相对性能最低，但GPU版程序也能比CPU版程序快一个数量级。

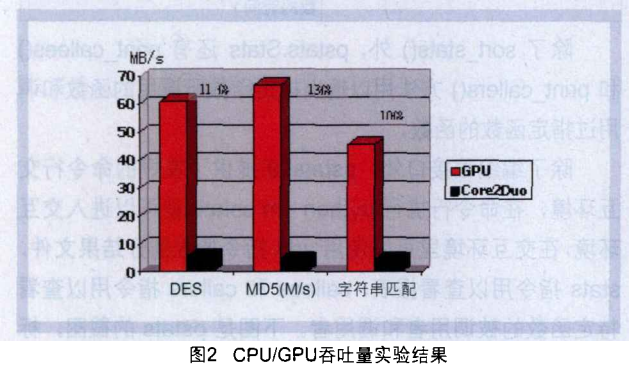

图2 CPU/GPU吞吐量实验结果

GPU能取代CPU吗？

GPU在运算能力的远远超越CPU，GPU是否能取代CPU呢？答案是否定的。GPU具有CPU所没有的局限性。GPU只提供单指令多数据类型处理，适合于数据并行计算。GPU在条件控制能力方面非常弱，若程序使用条件控制语句会极大影响GPU程序的执行效率。当然，有部分条件控制语句可以用计算来代替，例如，判断两个整数是否相等可以用两个整数异或后再映射成0和1来代替。本文中的实验中，利用了这些技巧来避免使用条件控制语句。另外现在的GPU与主机(host)数据交换只能通过总线来实现，对于需要大量I/O的应用，通信就会成为GPU性能瓶颈。

以通用计算为目的GPU发展趋势

NVIDIA发布了Tesla通用计算架构方案，Tesla GPU运算处理器不是一图形处理专业卡，可以看作之前的NVIDIA图形处理专业卡的通用计算版本。

可以看出，以通用计算为目的GPU发展趋势是GPU和CPU的整合，适合于大量数据并行计算的任务由GPU来承担，GPU定位为CPU的协处理器，需要复杂条件控制的，只能串行处理的任务则由CPU来承担。■

作者简介

丁艺明，男，趋势科技高级工程师，从事网络安全产品开发。可以通过 ding.yiming@gmail.com 与他联系。

刘波，男，趋势科技高级研发经理，从事网络安全产品开发。

■ 责任编辑:赵健平（zhaojp@csdn.net）

Python 性能优化经验谈

本文介绍了如何使用工具解决Python编程的性能调优问题。

■文 / 赖勇浩

引言

随着多核架构成为主流，CPU性能持续攀升，现代计算机的性能已经强大得令普通人难以想象。很多软件开发者为此乐得大喊：“写代码不用再考虑性能啦！”对于这个观点，我是同意一半反对一半的。同意的是硬件性能提升的确在一定程度上把开发人员从性能泥潭中解放出来，可以使用开发效率更高的开发工具（如脚本语言）来快速有效地完成工作；反对的是随着 Web2.0、网络游戏、在线视频等新的产业兴起，对性能也提出了新的要求，所以性能优化仍然是专业程序员必修的一课。

近几年来，因为能够提升开发效率降低开发成本，脚本语言逐渐占有了大份额的开发市场。其中开源的 Python 更成为众多 Web 应用和游戏开发者的首选，在2008年1月 Python 获得TOIBE 2007年度编程语言称号，一举超越了老牌劲旅 Perl，占据了排行榜的第6位。随着 Python 越来越流行，应用 Python 的项目越来越多，用 Python 完成的项目越来越大，对性能的要求开始变得非常敏感。

Python 性能剖分工具

眼看着项目即将完成，却被测试人员告知没有通过性能测试，这种情况在开发中屡见不鲜。接下来的工作就是加班