中国科学院软件研究所

杰出青年人才发展专项计划中期考核报告

姓名	张云泉	课题名称	并行算法与并行软件
资助金额	160 万	支持周期	2009.9-2013.9
资助类别	应用基础	所在部门	并行软件与计算科学实验室

研究工作进展总体情况

（内容包括：已完成的科研目标、成果及其在该领域的应用和所处地位）

专项计划启动 2 年多以来，本人在任务书确定的几个研究方向上，带领课题组成员开展了卓有成效的研究开发工作，总体进展情况如下：

在国际会议和期刊上发表文章 19 篇，第一作者翻译译著一本。其中在本领域著名国际会议 IEEE Cluster 2012、EuroPar 2011 和 2012，ICPP 2011 和 2012，IEEE ICPADS 2009 和 2011，IEEE ICA3PP 2012 等上各发表文章一篇。国际期刊 Frontier of Computer Science in China(SCI)发表文章一篇。拟投稿本领域著名国际会议 CGO 2013 和 PPoPP 2013 文章各一篇。2012 年 2 月，应美国 SIAM 学会邀请和全额资助，赴美国出席 SIAM PP 2012 大会，并做大会特邀报告一次。ISC 2011 和 GTC Asia 2011 大会特邀报告各一次。担任 IEEE CSE 2010 程序委员会共同主席。担任本领域著名国际会议 ICS 2010，CSE2011，SC2011, ICPADS 2012，PDCAT2012，IPDPS 2012，CCGrid 2012，e-Science 2012，Cluster 2012，CGO 2013 等的程序委员会委员。担任 ISC 国际会议 Steering Committee Member。

在《计算机研究与发展》、《数值计算与计算机应用》、《软件学报》、《计算机科学》和《计算机工程》等国内一级学报、核心期刊和国内会议共发表文章 29 篇，申请发明专利 3 项，申请软件著作权 8 项。

申请成功国家自然科学基金重点基金一项（子课题负责人），主持国家重大专项核高基课题子课题一项，主持 863 高性能重大专项课题子课题 3 项，国防军工课题 1 项，对外争取总经费额达 1600 余万元。

主持的国家 863 子课题，开展地球外核热流动的大规模可扩展并行数值模拟软件平台的研究，在国产千万亿次平台天河一号 A 上实现了超过数万处理器核的可扩展性：从 3072 至 24576 核，并行效率达到 87%，且成功尝试了 120 亿网格的运算规模，超过课题验收指标一个数量级，受到 863 重点项目专家组的好评。

在申请成功的国家重大专项核高基子课题的资助下，研制成功龙芯 3 新一代国产多核高性能扩展数学库 CLeXML V1.0 版和针对 X86 平台的国产高性能扩展数学库 CASIML V1.0 版，并取得软件著作权。基于该课题的关键技术研发的 OpenBLAS 软件包已经成为国际上比较活跃的开源项目，https://github.com/xianyi/OpenBLAS ，下载量达到 1676 次，有若干国际知名公司开始资助该项目，逐渐取代国际知名的 GotoBLAS 软件包的位置。

在 AMD 公司的资助下开发的 OpenCL 版 OpenCV 可视化算法库，已被 OpenCV 官方网站接纳并合并成为开源代码的一个分支。

本人被聘为国家自然科学基金委第十四届信息科学部专家评审组成员(2012.6 -2014.6)，中国计算机学会理事，中国软件行业协会常务理事，中国计算机学会高性能计算专业委员会秘书长，担任中国计算机学会青年计算机科技论坛学术委员会主席（2010-2011）和荣誉委员。

被中国科技大学聘为兼职教授，中国传媒大学和华南理工大学聘为兼职教授和博导。

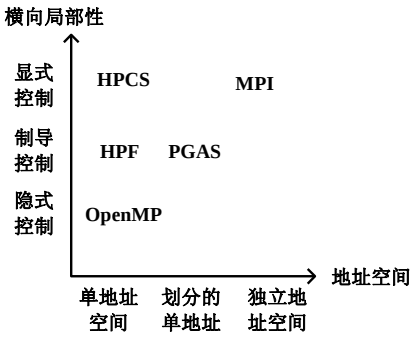
在本人的努力下，与 AMD 公司成立中科院软件所与 AMD “APU 软件联合研究开发中心”，任联合实验室主任；与美国 Argonne 国家实验室数学与计算机科学部（MCS）成立 PPCT 联合实验室（a JOINT LAB FOR Parallel PROcessing and computing techniques Research），任联合实验室执行主任；

本课题支持毕业硕士生 10 名，博士研究生一名,其中一名硕士生获得优秀毕业生，本人获得优秀导师。正指导硕士研究生八名，博士研究生六名，联合指导博士研究生两名（中科大，中国海洋大学各一名)。

详细的研究进展如下：

1. 基于层次存储和层次并行的新并行计算模型和并行编程模型研究

1) 调研国际上基于层次存储或层次并行的计算模型和编程模型研究进展，对主流并行程序设计语言的局部性设计机制进行了深入的分析和分类，总结了其优缺点，通过调研综述，把握了国内国际基于层次存储或层次并行的并行计算模型和并行编程模型的研究进展，指出了新一代并行编程语言所应具有的若干语言特点，重点提出了应综合考虑横向局部性和纵向局部性支持的设计机制的研究的观点。图 1 为主流并行编程语言和面向高生产率并行编程语言在地址空间以及横向局部性的位置图，图 2 为 Cell 编程模型的横向和纵向局部性位置图。相关研究工作整理出综述技术报告一篇。



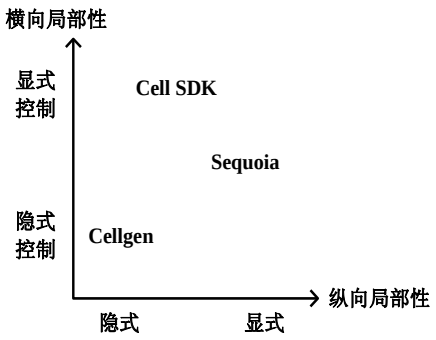


图 1 编程语言的横向局部性及地址空间　　图 2 Cell 编程模型的横向和纵向局部性

2) 提出一个新的基于延迟隐藏因子的 GPU 并行计算模型：针对 GPU 上编程和调优过程的繁琐和GPU低延迟高带宽的特性，提出了基于延迟隐藏因子的GPU计算模型，图 3 是 GPU 模型计算流程图。图 4 是 GPU 上的 Volkov 矩阵乘算法在不同规模下的预测结果和实测结果图，实验结果表明了模型的正确性。相关工作在《软件学报》发表文章一篇[11]。

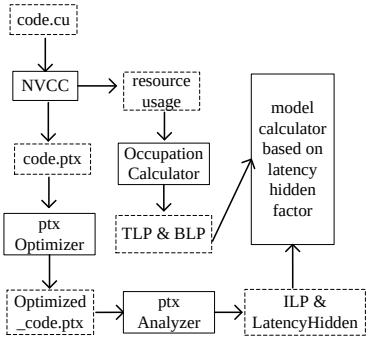


图 3 GPU 计算模型计算流程图

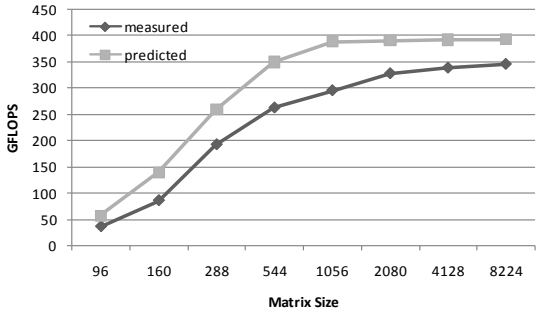


图 4 模型预测值与实测数值

3) 提出一个新的面向层次化通信的 LogGPH 模型：由于大规模集群系统网络拓扑结构复杂，通信具有层次性，在分析已有 BSP、LogP、DRAM(h)、Memory-LogP 和 HPM 等并行计算模型的基础上，根据魔方等的高效能计算系统的并行体系结构特点，提出了新的带描述通信层次性参数 H 的 LogGPH 并行通信模型，用以描述大规模集群系统的通信层次性，进一步提高并行计算模型的分析精度。用点对点和集合通信分别做了测试，图 5 是用 LogGP 和 LogGPH 模型在魔方上对 MPI_Allgather 邻居交换算法在不同配置下的预测值和实测值，结果表明引入层次性比原始模型更准确。通过考虑通信层次，点对点的平均预测误差由原始 LogGP 模型的 30%降低到 13%。通过考虑通信层次，Allgather 的平均预测误差由原始 LogGP 模型的 36%下降到 12%。相关工作发表在 IEEE CSE 2010 国际会议上[6]。

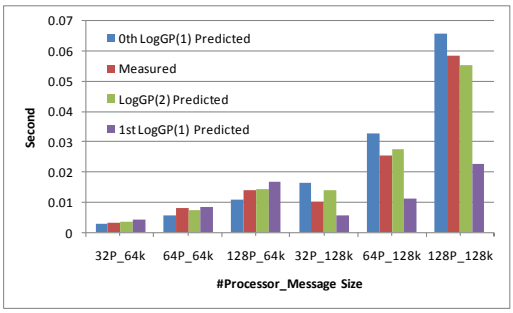


图 5 魔方上对邻居交换算法采用 LogGP 和 LogGPH 模型的预测和实测值对比

4) 提出一个新的基于横向局部性的多核计算模型 MRAM(h)

RAM(h)模型扩展 RAM 中单层存储层次为多层，但是其存储复杂度只描述了存储层次间的数据重用，并没有考虑多核共享缓存所带来的线程间共享存储的数据共享，我们在多核上对 RAM(h)进行横向扩展，同时考虑数据重用和数据共享。RAM(h)和 MRAM(h)模型的主要目的都是在实现算法时要充分考虑数据所在存储层次，尽量考虑数据重用，以提高算法的性能，MRAM(h)还要求考虑并行计算任务之间的数据共享率，尽量使得共享数据较多的任务分配的更近，也就是在更高的存储层次共享数据，此外，还要保证共享数据的访问时间相离的较近。相关工作发表在 HPC China2011 和计算机科学上。

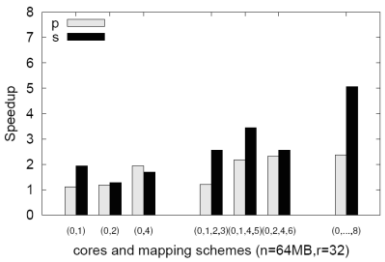


图 6. 平台 1 上共享 LLC 横向局部性

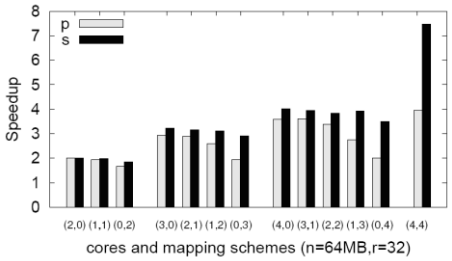
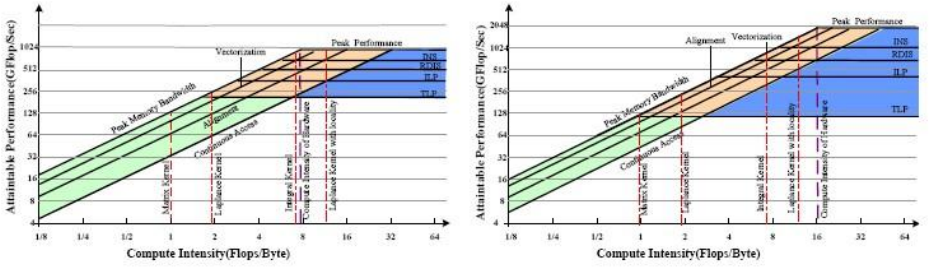


图 7. 平台 2 上共享 LLC 横向局部性

5) GPURoofline—一个 GPU 性能优化指导模型

GPU 程序性能优化的本质是将算法特性向底层硬件架构特征映射的过程。因此，GPU 程序的优化不仅要考虑具体算法的算法特性，还需要考虑底层硬件平台的架构特征。GPU 性能优化的这种复杂性以及底层硬件平台的日益多样性，大大提高了 GPU 编程，特别是实现跨平台性能移植的难度。为此，我们将 Roofline 模型引入到 GPU 编程中，并结合 GPU 编程的特点，为不同的 GPU 硬件平台分别建立和完善了其性能优化指导模型---- GPURoofline。

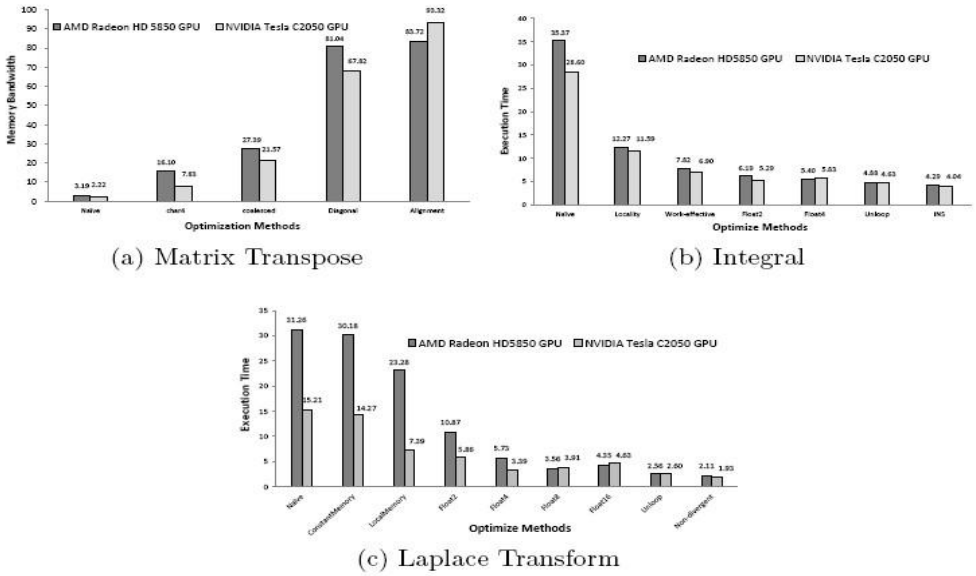
GPURoofline 性能指导模型的主要功能是通过提供性能信息确定具体 GPU 程序的性能瓶颈，并给出性能优化方案：性能优化方法以及其应用次序。为此，我们引入计算密度的概念，提出了硬件计算密度和 GPU kernel 计算密度的计算方法，并通过这两个计算密度的比较，将 kernel 分为访存密集型和计算密集型两大类。不同类型的 kernel，优化的重点不同。同时提出，提高片访存带宽和计算资源的利用率以及数据本地化是提高 GPU 程序性能的主要途径。 并结合具体的硬件平台，对影响性能的主要因素进行建模， 并通过 Micro-Benchmark 的性能测试，确定了不同硬件平台的优化空间以及不同优化方法在特定硬件平台上对性能的影响。 最后，将这些性能信息以可视化的形式表现出来，如图 1 所示，我们为 NVIDIA C2050 GPU 和 AMD HD5850 GPU 分别建立了 GPURoofline 模型。



(a) AMD HD5850 GPU (b) NVIDIA C2050 GPU

图 8 GPURoofline 模型

为验证模型的有效性，我们优化了具有不同计算密度的三个算法：矩阵转置、图形拉普拉斯变换以及图像积分函数。实现结果表明，与其初级实现版本相比，在 GPURoofline 模型的指导下，这些函数的性能无论是在 NVIDIA C2050 GPU 上还是在 AMD HD5850GPU 上都达到了 7.8~42.4 的性能加速比。图 2 展示了这三个算法在 GPURoofline 模型指导下，性能优化过程及加速比。相关成果已经在本领域的著名国际会议 EuroPar 2012 和 ICA3PP 2012 上发表。



(a) Matrix Transpose (b) Integral

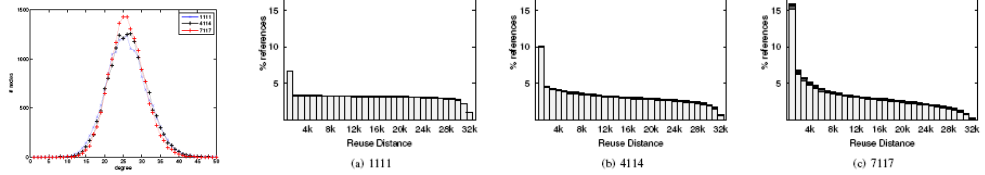
(c) Laplace Transform

图 9 不同算法在不同平台上的性能计算比

6) 图遍历中的局部性模型

图算法是非规则计算的典型模式，其中图遍历是大量图论算法的基础。由于图遍历访存模式与图结构密切相关，因此传统的，面向规则计算的重用距离预测方法并不能有效到扩展到图算法中，我们通过实验发现，对于同样大小但不同结构的图，图遍历所产生的重用距离签名有很大不同，如图所示，左边显示 3 个大小相同，结点度数分布相同的图，然而其图遍历的重用距离签名（右侧 3 个子图）却有很大不同。这就促使我们将图的结构引入到预测模型中去。

a	b	c	d	input ef	n	e	real ef
1	1	1	1	13	16384	425656	25.9
4	1	1	4	14	16384	423024	25.8
7	1	1	7	17	16384	411046	25.0



(a) 1111 (b) 4114 (c) 7117

图 10 三种大小相同但结构不同的图以及图遍历的重用距离

我们发现，在图遍历算法中，可以提取一个我们称为结点生产序列的概念，在此序列中每个结点都只出现一次，但出现排序根据图的不同结构而不同。例如在广度优先遍历中，结点生产序列的顺序即为按每个结点第一次被访的时间排序。基于结点生产序列，我们提出一种结点距离的概念，为每个内存访问对应一个结点生成序列中的元素，为了获得每个内存访问的重用距离，利用经典的球箱模型将结点序列中的结点距离转化为实际内存访问的重用距离，此转换过程中将图的结构转化为球箱模型的参数，从而将图结构融入重用距离的计算中。如下所示，式 1 是将结点距离 VD 通过球箱模型 P 转化为重用距离 RD，而在球箱模型的计算中用到图的参数，例如式 2 和式 3 中的图边数 e，结点数 n。

$$RD(d) = \sum_{i=1}^{n} VD(i) \cdot \mathrm{P}(i,d) \qquad (1)$$

$$\mathrm{P}(m,k) = \frac{T(m,k)}{(\mathrm{C}_n^e)^m} \quad (2)$$

$$T(m,k) = \sum_{i=0}^{r} T(m-1,k-i) \cdot \mathrm{C}_{k-i}^{r-i} \cdot \mathrm{C}_{n-k+i}^{i} \quad (3)$$

实验结果如图所示，对不同规模的图，预测的重用距离与实际测试结果相近，说明模型很好的描述了图的结构。

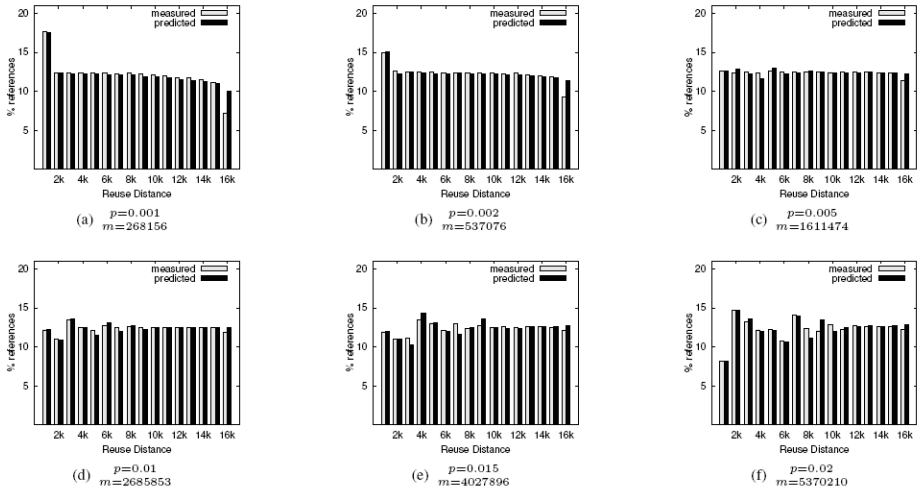


图 11 模型预测是实测结果

此外，我们还研究了若干扩展模型，例如结点距离分布预测，结果如图 12 所示；预测图遍历层数，结果如图 13 所示。

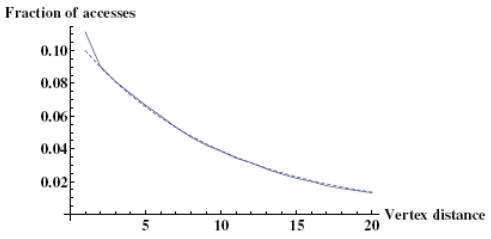


图 12 结点距离分布预测和实测结果

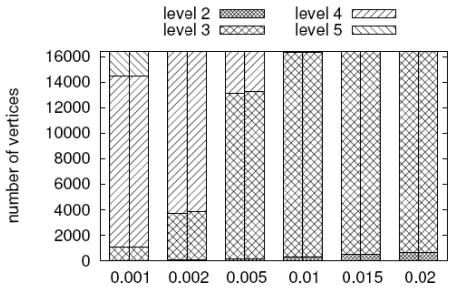


图 13 图遍历层数预测和实测结果

相关工作发表在本领域的著名国际会议 ICPP2012 上。

7) 基于局部性的并行计算模型

针对近几年在编程模型，算法优化，数据结构以及体系结构设计中出现的“load-and-compute”的计算模式，提出了一种局部性函数的概念，亦算法在一部分输入上能进行的最大计算次数，例如矩阵乘法的局部性函数为 L(n) = 2(n/3)3/2 ，基于局部性函数的计算模型如下：

$$T_P(i) = \begin{cases} \frac{W(n)}{L(C_k)}(T_P(k-1) + \frac{C_k}{B_{k+1}}) & \text{if } i = k,\ k\text{th level is shared} \\ \frac{W(n)}{P \cdot L(C_k)}(T_P(k-1) + \frac{C_k}{B_{k+1}}) & \text{if } i = k,\ k\text{th level is private} \\ \frac{L(C_{i+1})}{L(C_i)}(T_P(i-1) + \frac{C_i}{B_{i+1}}) & \text{if } i\text{th and } (i+1)\text{th level are both private} \\ \frac{L(C_{i+1})}{P \cdot L(C_i)}(T_P(i-1) + \frac{C_i}{B_{i+1}}) & \text{if } i\text{th level is private, } (i+1)\text{th is shared} \\ T_1(L(C_1)) & \text{if } i = 1 \end{cases}$$

表 1 并行计算模型以及基于模型的设计所用到的参数

parameter	description
F	FLOPS
P	number of processors
G	processor-memory gap
$B(B_m)$	memory bandwidth
B_i	cache bandwidth of ith level
C	total cache size
C_i	cache size of ith level
B_s	shared cache bandwidth
B_p	private cache bandwidth
C_s	shared cache size
C_p	private cache size
R_s	resource for the shared cache
R_p	resource for each private cache
R_c	resource for each core

基于此计算模型，我们研究了经典的双缓冲优化方法，证明了若干定理获得了最优缓冲划分方法。我们发现传统的等分双缓冲设计在 cache 较小时并不能获得最优性能，而最优的划分大小应为满足 L(C/k)/(P*F)=(C-C/k)/B 的 k，其中 k 为 cache 划分比例。

此外，面向多核体系架构中复杂 cache 结构，基于并行计算模型定量地研究了共享 cache 和私有 cache 的优势，得出 $\frac{C}{P\cdot L(C/P)}-\frac{C}{L(C)}=\frac{B_m}{B_s}-\frac{B_m}{P\cdot B_p}$，结果如图 1 所示。通常多核均包含多层次 cache，为了研究两层 cache 设计，将硬件架构映射到并行计算模型得到如公式（1）所示，利用拉格朗日乘数法获得若干等式（2），表示在基于局部性的计算模型下 cache 体系结构的最优平衡设计。

$$T=\frac{W(n)}{P\sqrt{R_p}}+\frac{W(n)}{L(R_s)}\frac{R_s}{B_m}+\frac{W(n)}{L(R_p)}\frac{R_p}{B_s}+\frac{W(n)}{PB_p} \quad (1)$$

$$P^2R_c^{3/2}=B_sPR_p^{3/2}=B_mR_s^{3/2}=\frac{2}{P(R_c+R_p)}\left(\frac{1}{\sqrt{R_c}}+\frac{1}{B_P}\right) \quad (2)$$

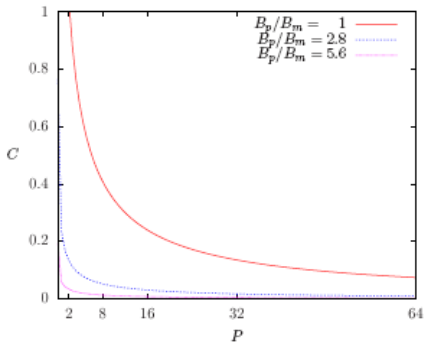


图 14 私有 cache 和共享 cache 比较

相关工作发表在本领域的著名国际会议 IEEE CLUSTER 2012 上。

2. 百万亿次高效能计算平台高可扩展基础并行算法研究;

1) 地球外核热流动的大规模可扩展并行数值模拟软件平台研究

主持的国家 863 子课题，开展地球外核热流动的大规模可扩展并行数值模拟软件平台的研究，在国产千万亿次平台天河一号 A 上实现了超过数万处理器核的可扩展性：从 3072 至 24576 核，并行效率达到 87%，且成功尝试了 120 亿网格的运算规模，超过

课题验收指标一个数量级，受到 863 重点项目专家组的好评。

2) **基于 GPU 的 Scan 算法的研究。**

Scan（也称 Prefix sum）算法是一种简单但非常常用的基础并行算法，是并行压缩，并行基数排序，并行快速排序，并行内存分配，并行 SpMV，并行 BFS 等其他基础算法中的重要组成部分。现有的基于 GPU 的大规模 Scan 算法，均是分三阶段进行，各个阶段之间需要进行全局同步。

创新性的设计了一种基于 GPU 的无全局同步的大规模 Scan 算法，相较于现有算法，对于规模为 N 的数组，我们将 Scan 算法的全局内存读写从 3N 降到了 2N。

我们设计了一种基于 GPU 的新型 Stream-Scan 算法，首先对需要 Scan 的数组进行分段，然后启动固定数量的 Block（Work-Group）依次对每一段进行 Scan，各 Block 内部分三阶段进行，第一阶段 Reduce，第二阶段等待前一个 Block 的 Reduce 结果，第三阶段 Propagate。该算法无需全局同步，仅启动一次 Kernel，不同 Block 之间 Reduce 和 Scan 可以同时进行，能充分利用计算单元。同时各段数据在同一个 Block 中处理的时候，均存储在 Shared Memory（Local Memory）和寄存器中，无需重复从全局内存读取，将全局内存读写从 3N 降到了 2N。我们将算法（OpenCL 实现）在 NVIDIA 和 AMD GPU 上分别进行了测试，在 NVIDIA Tesla C2050 上，相较于开源库 Cudpp（CUDA 实现）最高获得了 1.74 倍加速比，相较于开源库 B40C（CUDA 实现）最高获得了 1.26 倍加速比，相较于 Matrix Scan（ICS'08）（OpenCL 实现）最高获得了 1.54 倍加速比。在 AMD HD 5850 上，相较于 Matrix Scan 最高获得了 1.67 倍 kernel 加速比，1.9 倍 program（kernel time + launch time）加速比。在 AMD HD 7970 上，相较于 Matrix Scan 最高获得了 1.51 倍 kernel 加速比，2.38 倍 program 加速比。相关工作正在撰写高水平国际会议文章，拟投稿本领域的著名国际会议 PPOPP 2013.

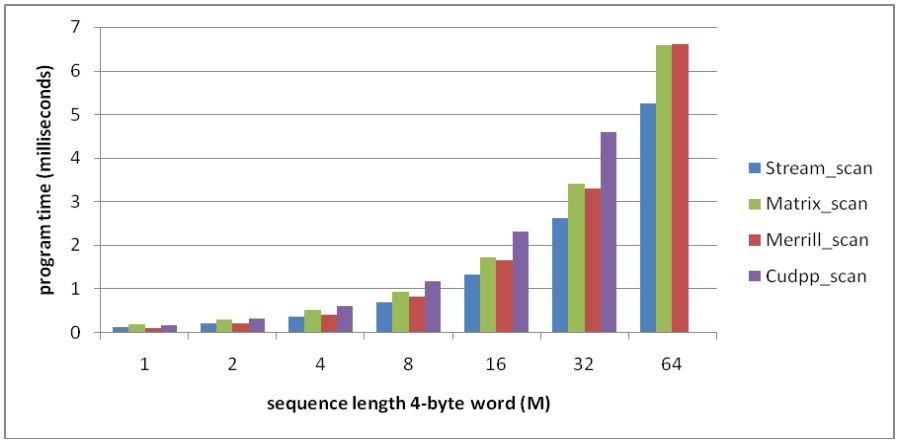


图 15 程序在 NVIDIA Tesla C2050 上运行时间（关闭 ECC）

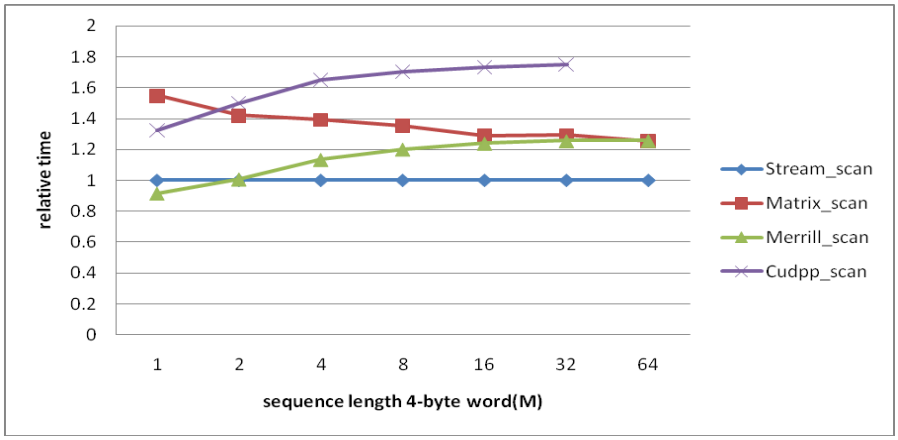


图 16 程序在 NVIDIA Tesla C2050 上加速比（关闭 ECC）

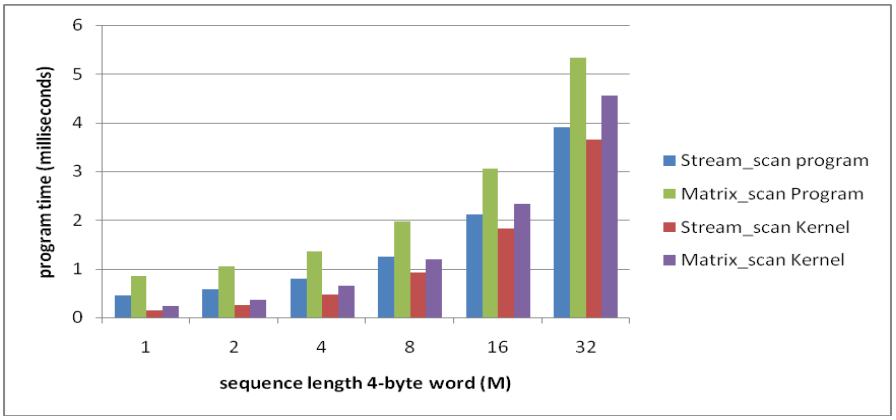


图 17 程序在 AMD HD 5850 上运行时间

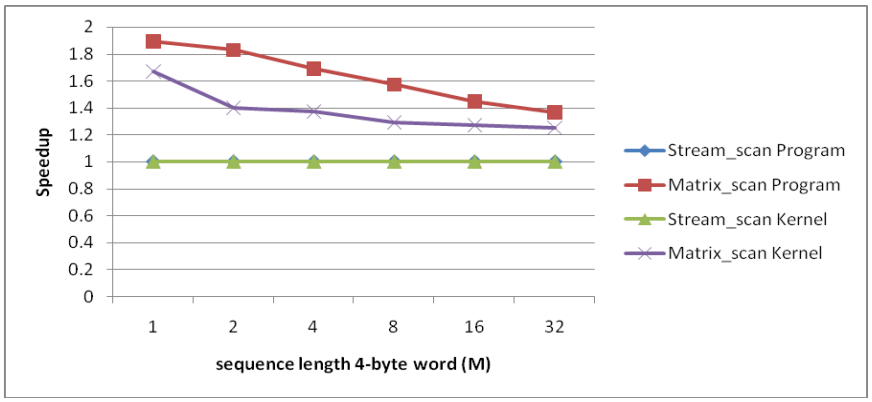


图 18 程序在 AMD HD 5850 上加速比

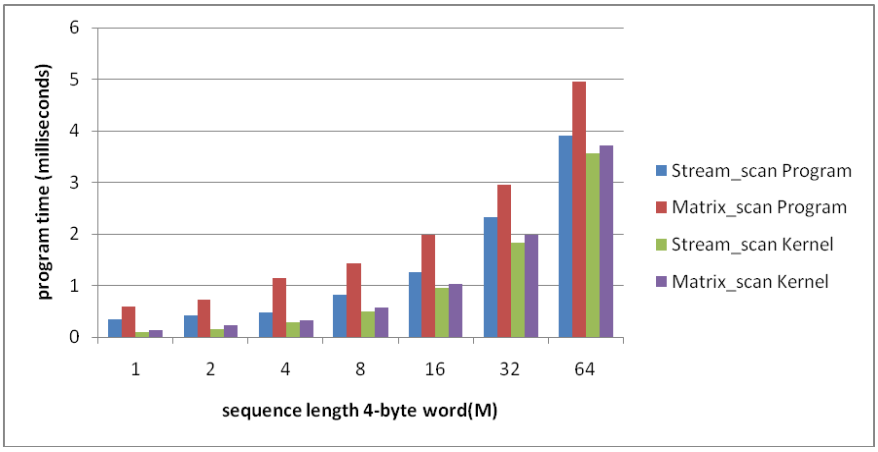


图 19 程序在 AMD HD 7970 上运行时间

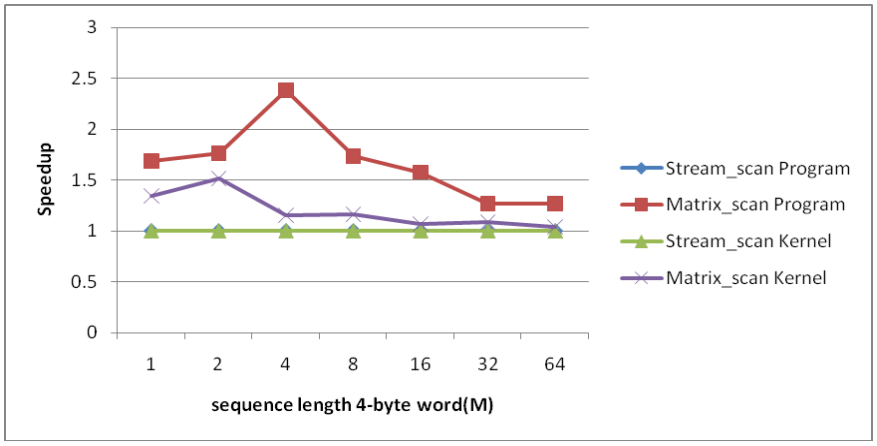


图 20 程序在 AMD HD7970 上加速比

3) OpenCV 算法库优化

OpenCV 于 1999 年由 Intel 公司建立，2006 年以后由著名的机器人研究实验室和技术孵化器 Willow Garage 提供支持。它是一个基于 BSD 许可证授权发行的跨平台计算机视觉库,包括图像处理和计算机视觉方面的很多通用算法的实现。OpenCV 有 C、C++和 Python 三个版本，支持 Windows、 Linux、 Android 和 Mac 系统，源代码中包含大概 500 个函数以及一系列使用这些函数实现的应用。从 2.2 版本开始 OpenCV 库中增添了 gpu 部分，该部分基于 CUDA 实现了对部分函数的 GPU 加速。从 2010 年 12 月 OpenCV2.2 版本发布至今，GPU(CUDA)部分不断有新的函数加入，目前为止大概有 90 个函数。

我们的工作是使用 OpenCL 将 OpenCV(Open Source Computer Vision Library)库进行加速。目前为止我们已经优化超过 60 个基础函数，两个典型应用 Face detection 和 FFT，相对于 CUDA 版本 OpenCV，我们开发的 OpenCV 版本（OpenCVCL）最高达到了 6.25 倍，平均 1.2 倍 Kernel 加速比。相较于 CPU 版本 OpenCV，最高达到了 410 倍，平均 74 倍 Kernel 加速比。

我们总体上将 OpenCV 库中的函数划分为三类：1）基本的矩阵运算函数，如四则运算、逻辑运算、最大值最小值、矩阵求和等；2）基本图像处理函数，如图像滤波、图像缩放、图像分析等；3）复杂的图像处理算法，如运动分析与对象跟踪、模式识别等。OpenCV 中基本图像处理操作包括去噪、边缘检测、角点检测、采样与插值、色彩变换、形态学处理、直方图和图像金字塔结构等。目前已经实现的函数中第一类 34 个，第二类 26 个，第三类 2 个。其中基本图像处理函数，我们也将其分为三类：数据无关、

数据共享和数据相关。表 2 展示了 17 个典型图像处理函数的分类。

表 2 典型图像处理函数分类（17 个）

类别	函数名	功能
数据共享	blur	图像模糊
	Sobel	使用 Sobel 算子计算图像差分
	Scharr	使用 Scharr 算子进行图像差分
	GaussianBlur	高斯卷积
	Laplacian	拉普拉斯变换
	erode	腐蚀
	dilate	膨胀
	morphologyEX	基于 erode 和 dilate 的高级形态变换
	cornerHarris	Harris 角点检测
	cornerMinEigenVal	计算梯度矩阵的最小特征值进行角点检测
数据无关	Threshold	对图像元素进行固定阈值操作
	copyMakeBorder	复制图像并制作边界
	warpAffine	对图像做仿射变换
	warpPerspective	对图像进行透视变换
	Resize	图像大小变换
数据相关	integral	计算积分图像
	equalizeHist	灰度图像直方图均衡化

数据无关函数 Threshold，Resize 优化：Threshold 函数对单通道图像应用固定阈值操作，将源图像中(x, y)位置的像素点与固定阈值进行比较，根据不同的阈值类型，目标图像(x, y)位置的像素点分别取相应的值。函数的典型应用是对灰度图像进行阈值操作得到二值图像，或者是去掉噪声，例如过滤很小或很大象素值的图像点。优化主要通过访存对齐和向量化写回来进行，下图展示了优化结果。

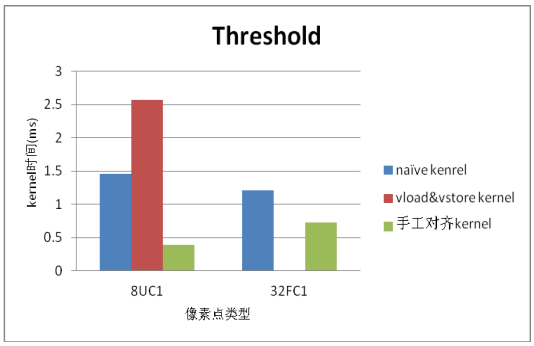
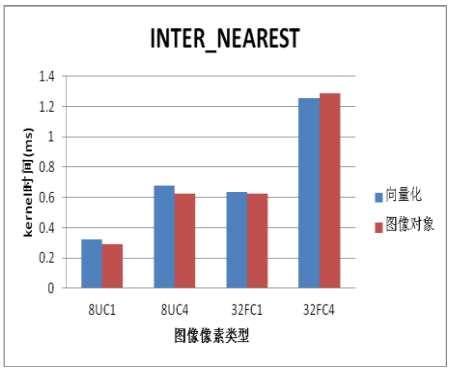
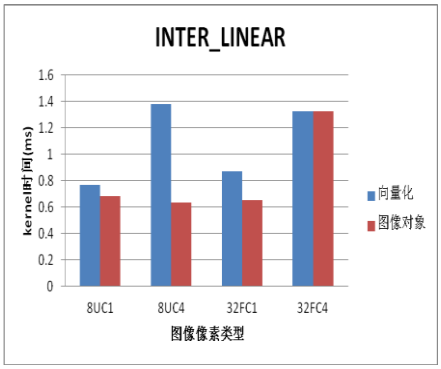


图 21 Threshold 优化结果

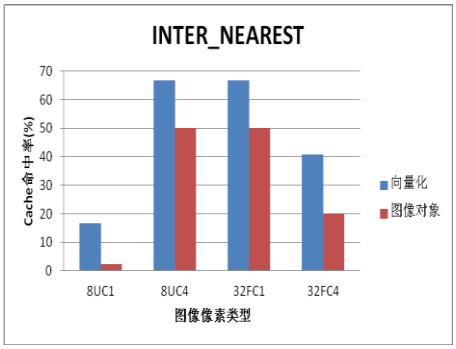
Resize 函数对图像进行大小变化，即进行图像缩放操作。缩放操作常用的插值方法有三种：最近邻插值(INTER_NEAREST)、双线性插值(INTER_LINEAR)和立方插值(INTER_CUBIC)。主要通过向量化插值和图像对象插值两种方法进行优化。图 22 和图 23 展示了优化结果。

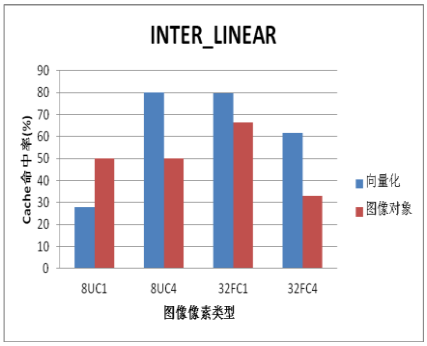




(a)最近邻插值两种方法性能对比　　(b)双线性插值两种方法性能对比

图22　向量化和图像对象两种方法性能比较

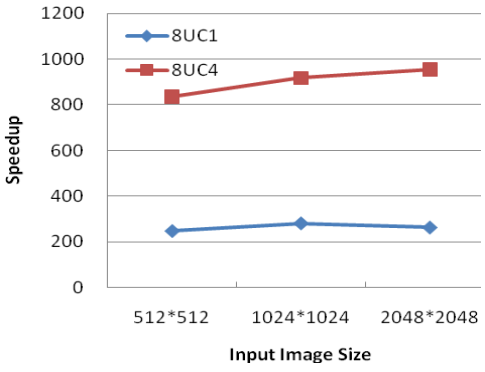


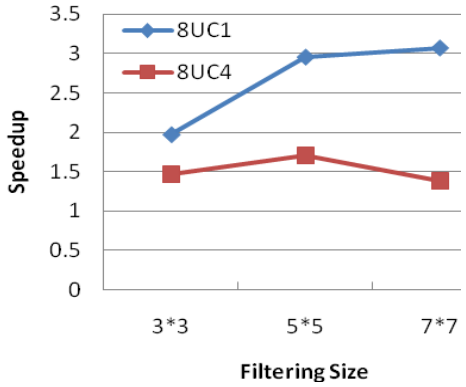


(a)最近邻插值两种方法Cache命中率对比　　(b)双线性插值两种方法Cache命中率对比

图23　向量化和图像对象两种方法Cache命中率比较

数据共享函数优化：Blur 算法是图像滤波中最基本的一种,其作用是对图像进行模糊处理，算法的基本原理是把图像中某点的值用该点周围一块区域的平均值来替代。主要优化方法为利用 local memory 实现数据重用。





(a)相对 CPU 版 blur 的加速比　　(b) 相对 NVIDIA NPP 版 blur 函数的加速比

图 24 Blur 函数加速比

数据相关函数 Integral 优化：Integral 函数用于计算图像积分，我们主要采取了三个方法对该函数进行了优化：（1）对列实施前缀和操作，按行写回存储。（2）一个线程组计算一整列。（3）一个线程组同时计算两列。通过这些方法的优化，我们取得了较好的加速比，相对于 CPU 版本函数最多获得了 11.6 倍加速，相对于 CUDA 函数，最佳获得了 1.5 倍加速。

4)　Viola-Jones 人脸检测算法的研究。

该算法是当前最高效和应用最为广泛的人脸检测算法。其算法本质就是首先由Haar特征值组成弱分类器，然后由一系列的弱分类器构成一个级联分类器，探测窗口利用这个级联分类器对图片中的人脸进行检测，通过每一级分类器的检测，将图片中最可能不是人脸的部分去除，而将主要计算都集中在最有可能是人脸的地方，从而大大减少了算法的总计算量。图 25 说明了这个算法的过程：

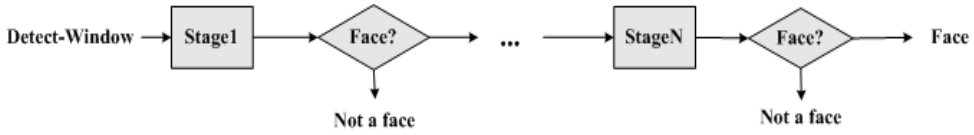


图25 Viola-Jones 人脸检测算法

从算法上看，Viola-Jones 算法非常适合运行在 GPU 上，因为这个算法体现了良好的并行性。因为没有数据相关性，每一个 Haar 特征值的计算以及每一个探测窗口的检测都可以并行执行。另外，为了能检测到图片中不同大小的人脸，图片会按照一定的比例进行放缩，而对这些放缩图片的检测也可以并行执行。然而，仅仅根据这些并行性将人脸检测算法移植到 GPU，并不能达到期望的加速比。因为这个算法有一个非常不适合在 GPU 运行的特征---线程间的负载不均衡。

如图 2 所示，我们假设有 9 个探测窗口，每一个 GPU 线程负责处理一个窗口， 经过 4 级级联分类器的检测，最终只有一个窗口检测到人脸。如图所示：在进行第一级检测时，9 个线程都处于忙碌状态。然而，在完成第一级检测后，探测窗口(0,0), (1,1)检测为不是人脸。那么在进行第二级检测时，负责这两个窗口的线程将处于空闲状态，其他七个线程依然是忙碌状态。这就造成了线程间的负载不均衡。最为严重的是，在最后一级检测中，只有一个线程处于忙碌状态，其他 8 个线程都空闲。这样就大大降低了GPU 上计算资源的利用率，从而大大影响了该算法在 GPU 上性能。

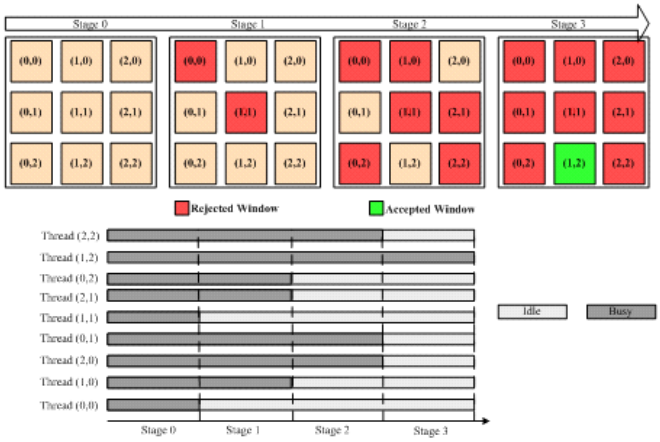


图26 人脸检测中线程间的负载不均衡

我们对这种会引起线程间负载不均衡的算法称之为不规则算法。而对不规则算法在GPU 上的优化，我们提出了 5 种优化策略：

(1)粗工作粒度（warp/wave-front）

传统 GPU 编程模型采用 work-item 作为最小并行执行单元，在本算法中我们采用warp/wavefront 作为最小并行执行单元。其实现方法就是每一个 work-group 只包含一个warp/wavefront。采用这种方法的好处就是可以消除本地同步，从而减少本地同步操作的开销。同时这也是后面即将介绍得本地队列的基础。

(2) UberKernel

因为要检测图片中不同大小的人脸，我们需要对图片按照一定的比例进行放缩，然

后再对这些放缩图片进行一一检测。然而，这么做的一个最大的缺点就是增加了全局同步开销。因此，我们引入了 UberKernel 的概念。其原理就是将这些放缩图片放在同一块全局内存上，由同一个 kernel 对这些图片进行统一处理，如图 27 所示：

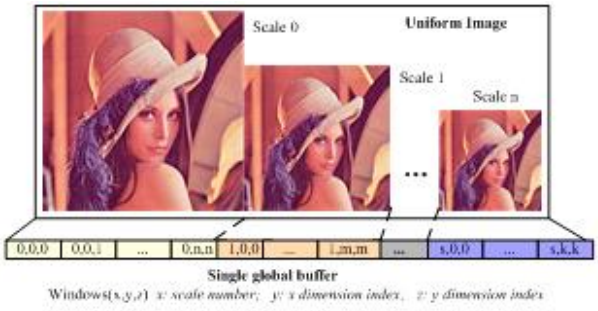


图27 UberKernel

采用 UberKernel，主要有 3 点好处：一是消除了全局同步开销；二是减少了 kernel 的启动时间；三是减少了计算资源的浪费。

(1)Persistent thread

图 28 说明了在 GPU 程序中，普通线程和 Persistent thread 的区别：对于普通线程，其生命周期为启动；获取数据、计算并返回计算结果；然后退出。而对于 Persistent thread 来说，当计算完第一次数据后，并没有退出，而是判断当前程序中，是否还有未处理的数据，如有，则继续执行，没有则退出。这也是后面介绍得本地队列的基础。

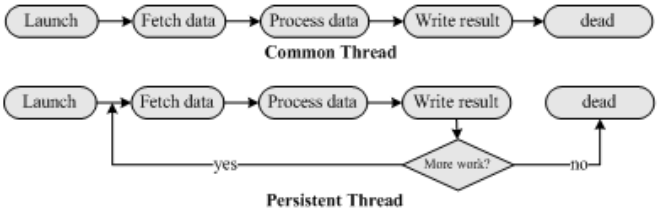


图 28 普通线程和Persistent thread的区别

(2)本地队列

在该算法中，我们采用本地队列，消除 warp/wavefront 中线程的负载不均衡，如图 29 所示：

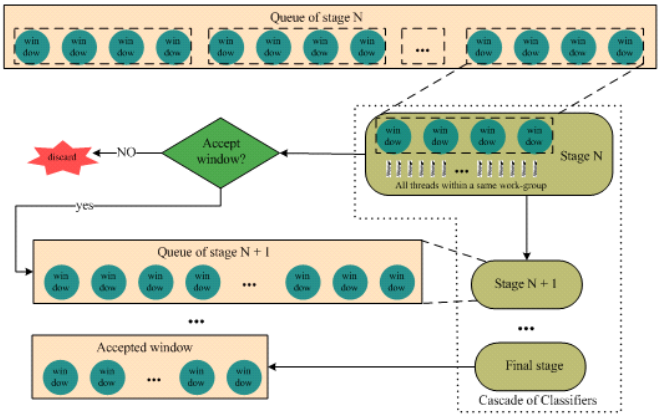


图29 本地队列

我们为每一个 warp/wavefront 在本地内存中建立一个队列，用于存储该 warp/wavefront 中线程所要处理的探测窗口，并将这些窗口按照一定的粒度（2,4,8 或 16）进行分组。每次从队列中取出一组，由 warp/wavefront 内的线程统一处理。如果该窗口检测为不是人脸，直接丢弃；否则进入队列，等待下一级分类器的处理。这样，经过 N

级分类器处理后，队列中剩余的窗口几位人脸。

(1)全局队列

本地队列仅仅解决了 warp/wavefront 内部的线程负载不平衡问题。我们引入全局队列解决 warp/wavefront 间的负载不平衡问题。目前全局队列的实现仅仅是一个初级的实现，即：首先将所有的探测窗口放入一个全局队列中，并按照 32 或者 64 进行分组；其次将这些分组平均分配给所有的 warp/wavefront; 最后，每个 warp/wavefront 独立处理所分配的探测窗口。全局队列的初级实现并没有采用负载均衡的方法，采用作业偷取（task stealing ）或者作业给予(task donation)方法实现最终的负载平衡将是下一步的工作。通过采用这五种优化方法，我们在 3 个 GPU 平台上都达到了较高的加速比。图 6 说明了人脸检测函数在不同平台上对不同图片的加速比。我们选取了 9 张不同大小、不同背景、不同人脸数目的图片作为我们的测试库。

如图 30 所示，我们选用 Intel(R) Xeon(R) X5550 4 核 CPU 和 AMD Phenom Ⅱ X4 9404 核 CPU 作为我们的 CPU 测试平台。选用 AMD HD5850，AMD HD7970 和 NVIDIA C2050 作为我们的 GPU 测试平台。实现结果表明，我们在 AMD HD5850 平台上达到了 5.193 ~35.08 times (平均 16.91) 的加速比；在 AMD HD7970 平台上，性能比 AMD HD5850 提高了 17.62%~51.35%；在 NVIDIA C2050 平台上达到了 5.85 ~32.641 times (平均 17.535)的加速比。该算法成果已经发布在 IEEE HPCC 2012 国际会议上了。

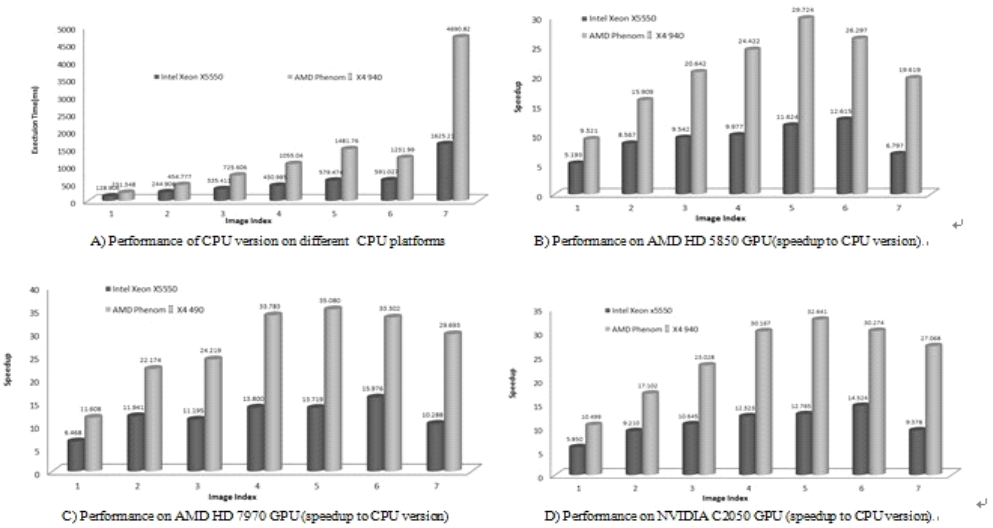


Figure 7. Performance comparison of GPU version and CPU version.

图 30 face detection 函数在不同平台上的加速比

3. 大规模可扩展非阻塞集合消息传递算法的研究；

1) **基于新的LogGPH模型设计了新的MPI_Allgather映射算法**，平衡了 MPI_Allgather 邻居交换映射算法中偶数步通信层次，在魔方单节点上测试效果平均加速比为 6%。图为魔方上新旧映射方法对比的加速比。

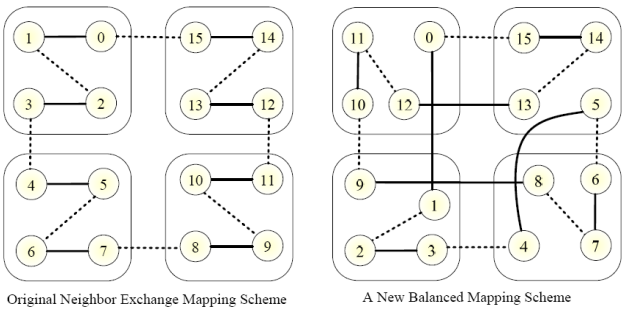


图 31 邻居交换算法新进程映射模式

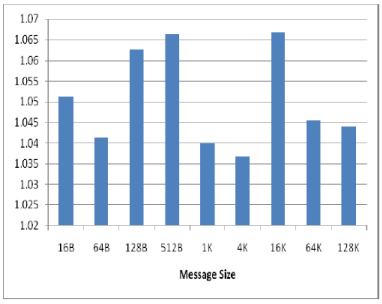


图 32 魔方上新映射方法加速比

2) 在魔方上，采用 MPICH 进程绑定工具 hydra 测试分析了 6 种不同进程绑定策略对点集合通信（IMB3.2）性能的影响，并分析了 Broadcast,Gather/Scatter, Allgather,Alltoall,Allreduce 算法在不同消息长度的实际运行程序[性能可以提高的空间。图 33 显示，长消息 ring 算法的 Allgather 性能至少可以提升 27%（性能最优的 CASE5 相对于性能最坏的 CASE3）。图 34,中短消息 Bcast 性能至少可以提升 31%。

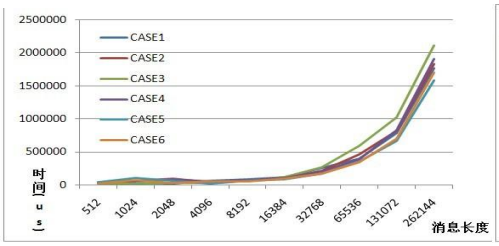


图 33 1024 进程不同绑定策略的 Allgather

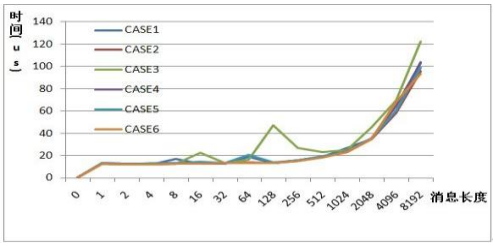


图 34 1024 进程不同绑定策略的 Bcast

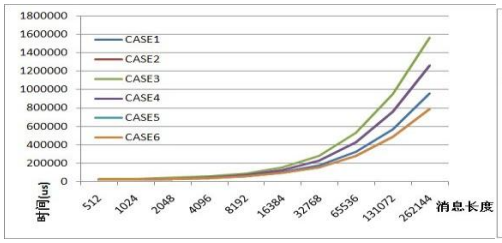


图 35 1024 进程不同绑定策略的 Allgather 模拟

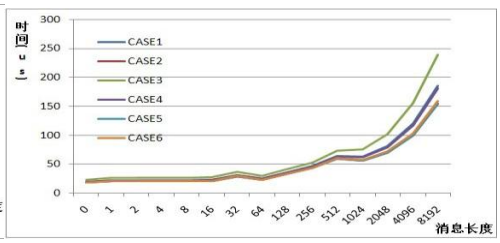


图 36 1024 进程不同绑定策略的 Bcast 模拟

通过分析不同进程绑定策略下 CPU 内、CPU 间、结点间通信的次数和消息大小，计算模拟了集合通信的实际运行时间，图 35 为 1024 进程 6 种不同进程映射策略下长消息的 ring 算法的 Allgathe 模拟时间，图 36 为短消息的 Bcast 的模拟时间。 相关工作的初步成果在国内核心期刊发表文章一篇，进一步的工作正在撰写国际会议文章。

3) 利用节点内的通信延迟差异优化 MPI 集合通信的性能得研究。

我们设计了一个节点内 NUMA-aware 的集合通信性能分析模型，分别在 Intel 和 AMD 两种多路多核平台上评估了不同进程映射对 MPI 集合通信的影响，依据该分析模型，为各集合通信算法设计了较优的通信模式。试验表明新通信模式可明显提高 MPI 集合通信的性能。如短消息的 *MPI_Bcast* 在 Intel 和 AMD 平台上可分别提高 76.5% 和 14%。我们通过穷举分析所有进程-核映射模式的方法证实理论分析结果，中短消息 Bcast 性能至少可以提升 31%。相关研究成果与美国阿岗实验室 Pavan Balaji 博士联合撰写文章一篇，并投稿至 IEEE/ACM International Symposium on Cluster, Cloud, and

Grid Computing(CCGrid2012)。

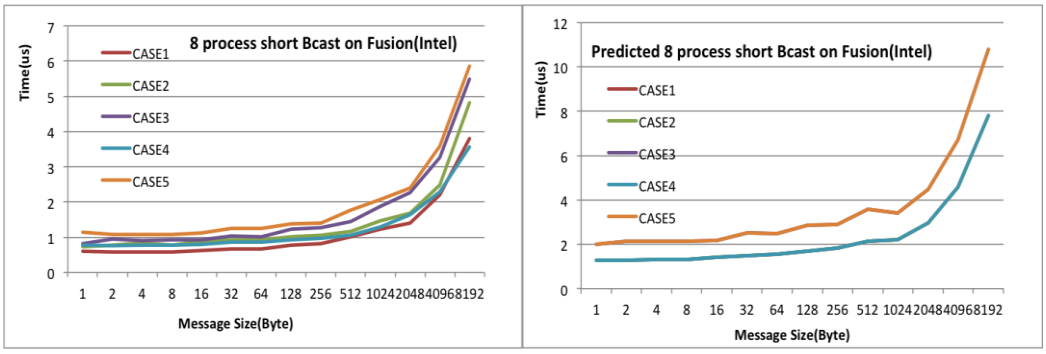


图 37 Intel 平台不同进程映射下 MPI_Bcast()性能 实测（左图）模拟（右图）

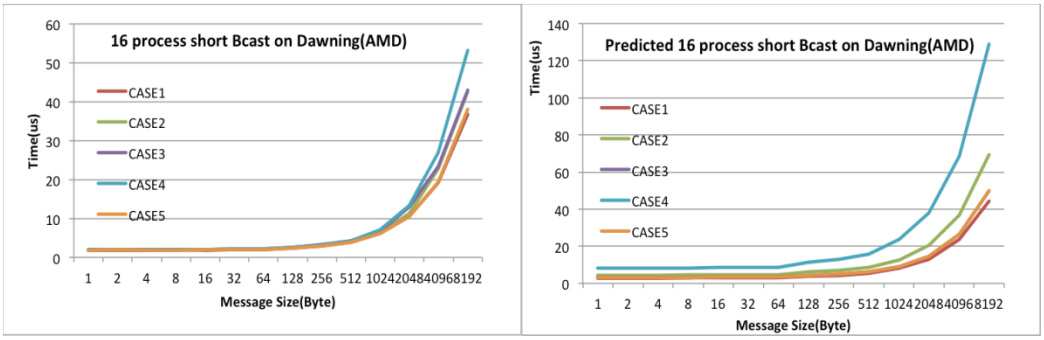


图 38 AMD 平台不同进程映射下 MPI_Bcast()性能实测（左图）模拟（右图）

假设一个双路四核共 8 个核的节点，进程-核映射就有 8! = 40320 种,其中每个 CPU 内 4!种映射是一样的,因为 CPU 内的 4 个核是一样的,两个 CPU 交换位置对性能无影响,因此 8!中映射被分为 8! / (4! * 4! * 2) = 35 个不同的组,每组以其中一个 CPU 的 4 个进程号表示如[0123]表示进程 0,1,2,3 在一个 CPU,4,5,6,7 在另一个 CPU 的 4! * 4! * 2 种同构情况，表 2 中较小的标准差最小值百分比证实了每组中同构情况的性能几乎相同.同时,性能最优的三组和最劣的三组证实了模型分析的最优/最劣映射,CASE1 是[0123]中的一个, CASE5 是[0576]中的一个.

表 3 消息长度为 4Byte 的 MPI_Bcast 进程映射穷举测试结果

组名	最小值	最大值	平均值	方差	标准差/最小值
Top 3 best group					
[0123]	0.65us	0.85us	0.70us	0.00884	4.57%
[0127]	0.66us	0.89us	0.75us	0.01657	6.17%
[0123]	0.69us	1.01us	0.80us	0.02884	7.78%
Top 3 worst group					
[0256]	1.05us	1.49us	1.21us	0.008065	8.55%
[0576]	1.01us	1.42us	1.22us	0.006397	7.92%
[0256]	1.05us	1.47us	1.24us	0.007441	8.63%

4. 面向多核国产处理器性能优化国产高性能数学库研究开发

1) 研制成功龙芯 3 多核 CPU 的新一代国产高性能扩展数学库 CLeXML V1.0 版和针对 X86 平台的国产高性能扩展数学库 CASIML V1.0 版。

在核高基国家重大专项（软件类）的资助下,作为课题负责人，带领团队研制成功针对龙芯 3 多核 CPU 的新一代国产高性能扩展数学库 CLeXML V1.0 版 和针对 X86

平台的国产高性能扩展数学库 CASIML V1.0 版，并取得软件著作权。两个软件包都包含 BLAS、LAPACK、FFT、直接解法器和迭代解法器等 5 个模块，采用了自适应性能优化、地址交错和多核并行化等技术进行性能优化。已经完成计算机软件著作权登记：X86 CPU 多核高性能数学库软件（登记号：2011SR092896）并与 Intel 公司开发的 MKL 10.1 版做了对比测试。在 X86 平台，CASIML 库 5 个模块单线程和 8 线程相对于 Intel 公司开发的 MKL 10.1 版本的平均加速比达到了 1.08 和 1.46。CLeXML 是我国自主开发的一款面向国产处理器的多核版高性能数学库，拥有自主知识产权、掌握源代码，且与 Intel 公司开发的 MKL 数学库性能相当。该数学库的开发，打破国外高性能计算机厂商在高性能数学库方面的长期垄断，填补国内无国产高性能数学库的空白，也大大降低了我国国产 CPU 的普及和应用难度。

2) 基于 RAM(h)层次存储计算模型的理论分析结果,创新性的提出 CRSD 新稀疏矩阵存储格式。该工作的论文已在本领域著名的国际会议 Euro-Par 2011 上发表。

SpMV 是求解 PDE 的一个核心算法，根据应用矩阵非零元的对角线式的分布特点创新性地提出了 CRSD 矩阵存储格式。该存储格式针对同一应用对角线格式在不同问题规模下，对角线分布具有相似性进行优化。通过对角线格式表示对角线分布。并通过自适应优化方法基于具体运行平台选择最优的 SpMV 实现。测试结果表明，所提出的新存储格式针对不同问题规模的矩阵，在不同线程下均优于 MKL 中基于 CSR 的 SpMV 实现，测试的性能与加速比结果如下图 6 所示。可以看出 CRSD 在单线程下，相对经典存储格式 CSR 运行效率，加速比平均为 3.05 倍，最高可以到达 4.38 倍；相对于 MKL 的 DIA 存储格式，对于不适合 DIA 存储格式的矩阵，CRSD 同样有很好的加速效果，如图中编号为 4 和 5 的矩阵，加速比可以达到 90 倍以上，除此之外，其余最高加速比为 2.02，平均加速比为 1.74。在多核情况下，CRSD 的加速效果在相同处理器下，是 CSR 存储格式的 2.16 倍，最高可以达到 4.61 倍。

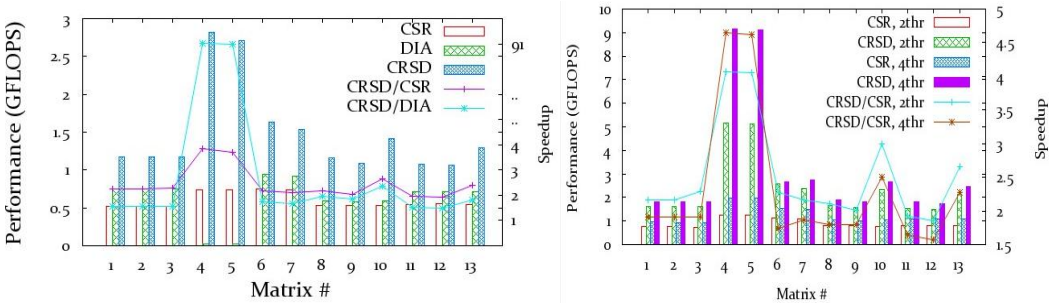
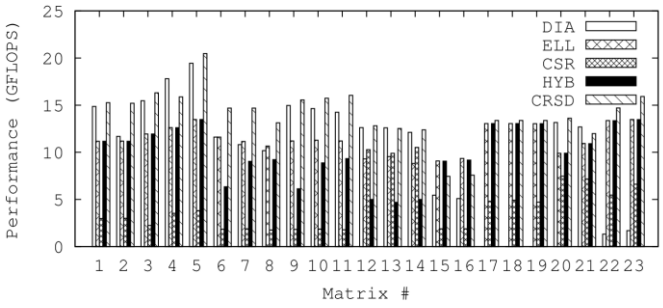


图 39　不同矩阵存储格式的性能比较

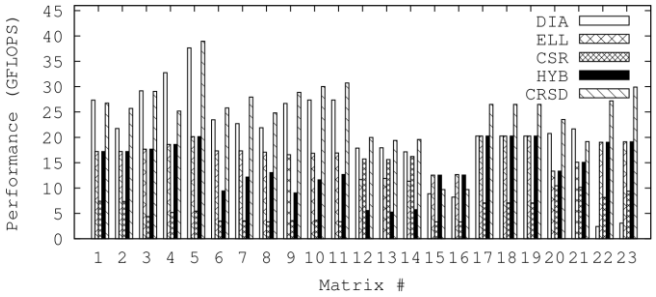
3) 基于 CRSD 格式的 SpMV 算法在 GPU 上的性能优化方法和技术研究，相关文章已在本领域著名的国际会议 IEEE ICPP 2011 上发表。

在将基于 CRSD 存储结构的 SpMV 算法实现移植到 GPU 上的过程中，根据 GPU 硬件结构特点开展了如下优化技术研究：a)有效利用局部存储器。CRSD 存储结构能够将 SpMV 处理相邻对角线时的公共 x 元素存放到局部存储器中，减少对延迟较大的全局存储器的访问次数。b)将离散点的存储方式由 CPU 上的 CSR 存储格式修改为 ELL 存储格式。由于 GPU 处理 CSR 存储格式的跳转操作时效率很低。c)CRSD 存储格式对矩阵进行分割时，分段大小是 wavefront 的整数倍，每个 work-group 处理一个对角线格式，避免 thread divergence 以及在访问全局内存时能够进行访问合并的优化方法。d)利用代码生成器，在运行时根据矩阵的对角线格式自动生成可执行代码，并采用全局展开等优化策略。与经典 GPU SpMV 实现(Bell and Garland,

SC2009) 的四种矩阵存储格式（DIA、ELL、CSR 及 HYB 四种存储格）的单双精度进行对比，实验结果如图 6 所示。相对经典实现四种存储格式的最优实现，CRSD 在处理双精度浮点的运行效率加速比为 1.52X，单精度为 1.94。



a) 双精度



b) 单精度

图 40 CRSD 在 GPU 上的性能加速

4) 龙芯 3 上 BLAS 的性能优化工作，相关工作被推荐到《软件学报》发表。所研制的 OpenBLAS 软件包已经成为国际开源项目，https://github.com/xianyi/OpenBLAS ，下载量达到 1676 次，有若干国际知名公司开始资助该项目，逐渐取代国际知名的 GotoBLAS 软件包的位置。

BLAS(Basic Linear Algebra Subprograms)是一个基本线性代数核心子程序集,主要包括向量和矩阵的基本操作。OpenBLAS 是我们在 GotoBLAS 2-1.13 BSD 版基础上发展起来的 BLAS 库开源项目，当前主要工作是为龙芯 3A CPU 提供高性能的 BLAS 库实现。该项目针对龙芯 3A CPU 的结构特点，通过分块选择，手工优化汇编，应用 128bit 访存指令，指令重排等技术对 BLAS 三级函数 GEMM（矩阵乘法）函数进行实现与优化。

```
gsLQC1(R8,F5,F4,2)          龙芯3 128bit访存指令
MADD    t11,t11,a0,b0
MADD    t21,t21,a1,b0

gsLQC1(R9,F13,F12,2)
MADD    t12,t12,a0,b1       浮点乘加计算指令
MADD    t22,t22,a1,b1
```

图 2 GEMM 核心汇编代码片断

在双精度和双精度复数情况下，OpenBLAS 优势明显，平均性能超过 ATLAS 39.6%和 27.7%，相对于 GotoBLAS 平均性能提升分别为 109.5%和 107.3%。在单精度和单精度复数下， OpenBLAS 相对于 ATLAS 的优势并不明显，分别只高出 5.0%和 2.0%；相对于 GotoBLAS，性能分别提升 52.0% 和 51.1%。因此，OpenBLAS 在单精度和单精度 GEMM 还具有明显的改进空间，我们推测是在 OpenBLAS 中没有使用单精度浮点向量（ps）指令，造成性能稍低。

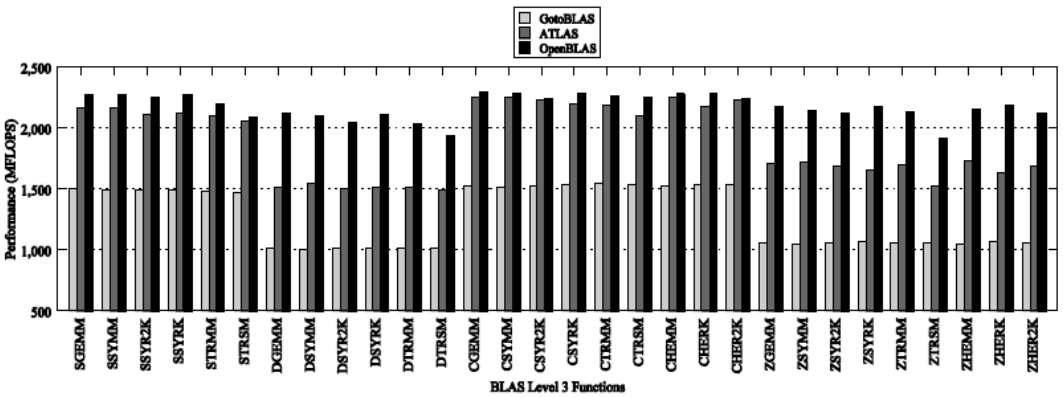


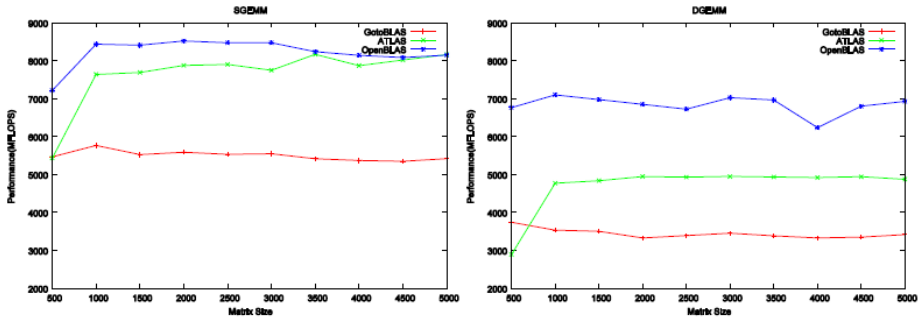
图 3 单线程 BLAS 3 级函数平均性能对比图

在多核情况中，各线程所需计算的子矩阵的总空间约为 3.6M，由于龙芯 L2cache 大小为 4M,采用四路组相连随机替换策略会导致各线程间对共享的 L2 Cache 的争。因此，我们一方面减小多线程下子矩阵的大小来减小对 L2 Cache 的消耗，另一方面，将各个线程使用的数据起址进行交错的排列布局，可尽量避免在 L2 Cache 中各个线程间的相互冲突和替换。

T0 打包后 子矩阵A	T1 打包后 子矩阵A	T2 打包后 子矩阵A	T3 打包后 子矩阵A	T0 打包后子矩阵B	T1 打包后子矩阵B	T2 打包后子矩阵B	T3 打包后子矩阵B

图 41 多线程子矩阵 A 与 B 交错布局示意图

OpenBLAS BLAS 3 级函数的 4 线程并行加速比达到 3.47。4 线程 BLAS 3 级函数平均性能高于 GotoBLAS 和 ATLAS 69%和 34%。。



(a)单精度 GEMM　　(b)双精度 GEMM

图 42 多线程性能对比效果图

5) **基于 x86 架构的稠密矩阵乘法汇编代码自动调优**

针对现有技术中手工编写稠密矩阵乘法函数的汇编代码开发效率低、可移植性差；和已有的自动调优矩阵乘法函数技术依赖底层编译器优化技术、性能低的缺点，提出了一种基于 x86 架构的稠密矩阵乘法汇编代码生成方法。该方法针对不同 x86 处理器平台硬件资源的种类和数量（如物理寄存器个数，高速缓存空间大小），利用较为成熟的源到源的翻译转换工具，先对稠密矩阵乘法实施通用优化技术，包括循环分块，循环展开，标量替换，冗余代码消除。然后结合编译优化知识，定义代码模板，将优化的稠密矩阵乘法函数的 C 代码一一映射为向量化的汇编代码。通过细粒度优化函数，如寄存器分配和指令调度，对汇编化的稠密矩阵乘法代码进行优化。最后对整套方法中出现的参数

（如矩阵分块大小，循环展开因子）进行自动搜索，找到最优稠密矩阵乘法函数。相关工作申请发明专利一项（申请号：2012101997066）。相关工作拟投稿本领域著名国际会议 CGO 2013。

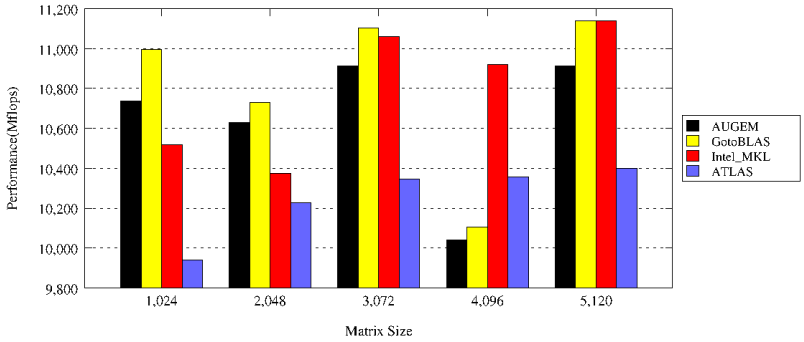


图 43 Intel Penryn 处理器上本实现方法(AUGEM)与主流 BLAS 库加速比效果图

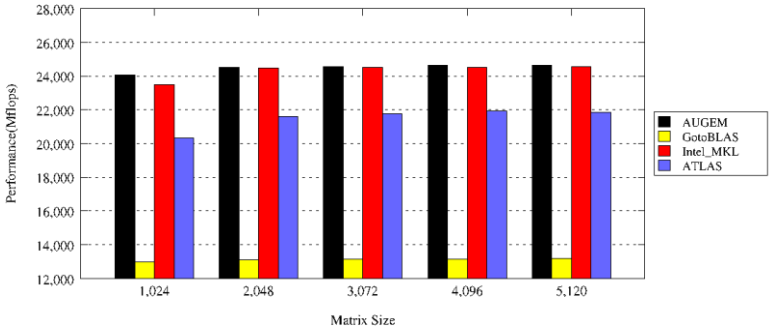


图 44 Intel Sandy Bridge 处理器上本实现方法(AUGEM)

6) 基于对角线数据结构的 SpMV 性能优化：针对对角线模式的稀疏矩阵存储方法最主要为 DIAG，这种方法会多存储矩阵范围外的非零元，对于远离主对角线的对角线，增加了存储矩阵外虚位的负担，提出了一种新的 DDD-SPLIT 存储方法。该方法只存储矩阵范围内的非零元及相应对角线上的零元，根据过对角线与矩阵两侧交点的水平线对矩阵进行切分，使得每个子稀疏矩阵行块内所需要的对角线相同，因此计算模式一致，可以重用 y。图 7 为 DDD-SPLIT 对 CSR 的加速比。相关工作申请发明专利一项（申请号：201010594672.1），在 IEEE HPCC 2010 国际会议上发表文章一篇。

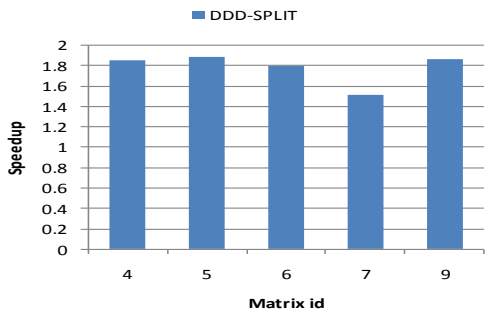


图 45 DDD-SPLIT 加速比

7) 通过在计算模型、大型稀疏线性方程组高效并行解法器、稀疏矩阵向量乘稀疏矩阵存储格式、MPI 与 OpenMP 的混合并行和可扩展消息传递集合通信算法等多个方面开展关键技术的研究工作，在国产百万亿次平台曙光 5000A（魔方）和深腾 7000 上，使地球外核热流动的数值模拟程序的强可扩展性达到了 2048 处理器核以上。

相关工作在 IEEE HPCC 2010 和 HPCA2010 国际会议上发表文章各一篇，国内一级学报《数值计算与计算机应用》发表文章一篇。

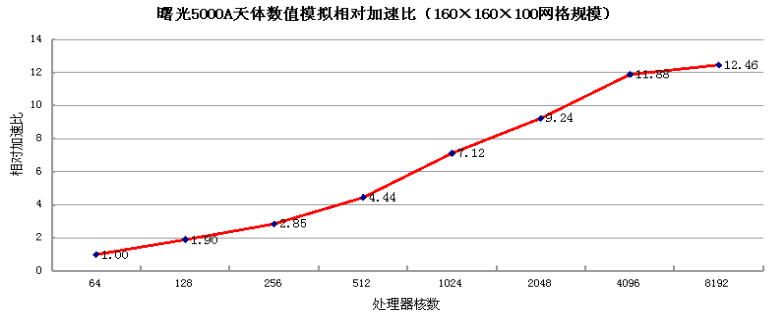


图 46 曙光 5000A 天体数值模拟加速比随处理器核数的变化（网格规模 160×160×100）

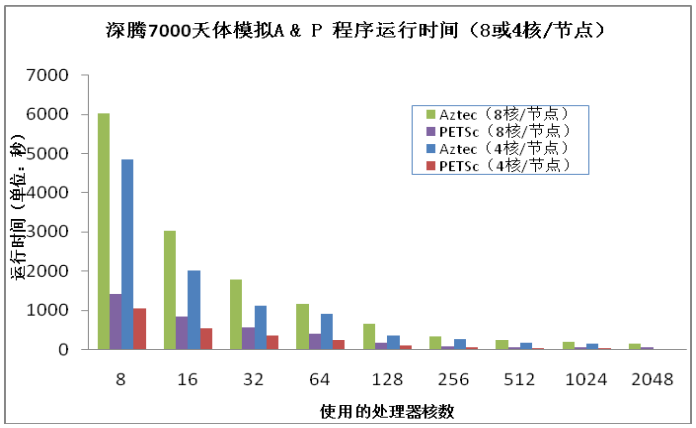
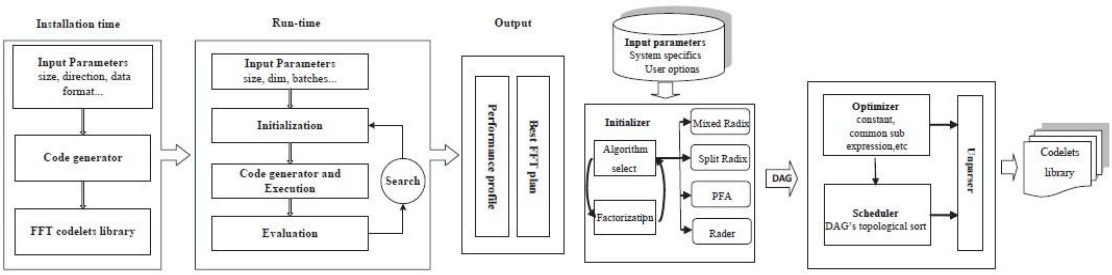


图 47 深腾 7000 天体模拟 Aztec 与 PETSc 程序运行时间（8 或 4 核/节点）

8) GPU 上国产自适应 FFT 软件包 MPFFT 的研制，相关工作发表在本领域著名的国际会议 IEEE ICPADS 2011 上。

快速傅里叶变换(Fast Fourier Transform，FFT)是离散傅里叶变换(Discrete Fourier Transform，DFT)的快速算法，具有广泛的应用，常被用于数字滤波、求解偏微分方程和信号分解等。事实上，FFT 已被评选为二十世纪科学和工程界最具影响力的十大算法之一。我们针对目前主流的 GPU 提出了一个新的 FFT 自适应框架，且基于该框架实现了自主知识产权的 FFT 库 MPFFT，如下图所示:



(a) MPFFT整体框架　　(b) 代码产生模块

图 48 MPFFT 的整理框架和代码生成模块示意图

首先在程序安装时，只需要借助一个简单的脚本文件，代码生成模块生成可以被 GPU kernel 调用的小因子 FFT 集合(codelets Library)；在程序运行时，获取输入问题的相

关参数(DFT 的维度，长度以及数量 batches)，并根据这些参数来初始化对性能影响较大的相关硬件参数譬如寄存器大小、LDS 容量和其 bank 数等。其次，运行时模块考虑这些参数和 FFT 算法的特点来对输入问题规模按照不同策略分解，从而生成多个 plan，然后在 Search 模块的监督下进行执行和性能评估。最终，Search 模块输出所有分解策略对应 plan 的性能数据，同时得到最佳 DFT 分解策略，并且生成的 profile 数据对其后问题规模的分解和构造能够提供向导。

我们在不同的 GPU 上评估了 MPFFT 库的性能，为了便于对比，CPU 版本我们选取 FFTW3.2.2 作为性能基准，而 AMD GPU 和 NVIDIA GPU 上的 FFT 分别选取 clAmdFft 1.4 和 CUFFT 4.0 作为对比。其中一、二、三维的测试结果依次如下图所示：

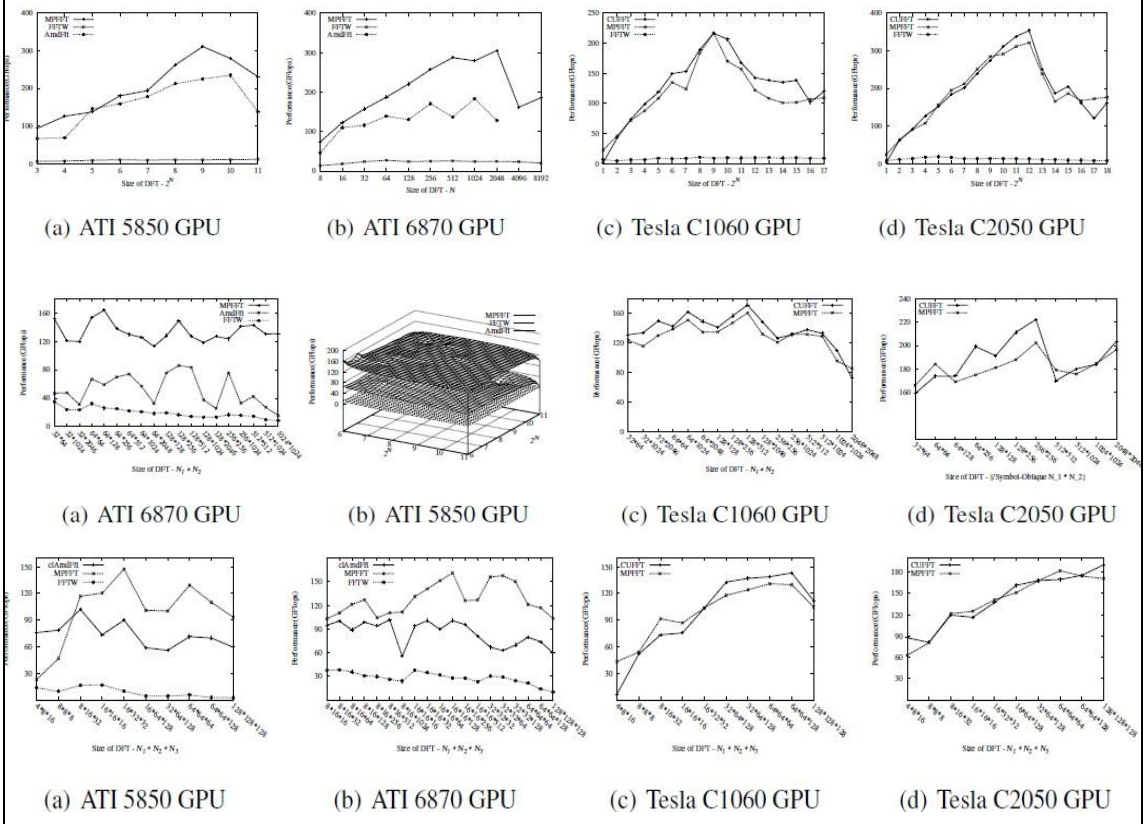
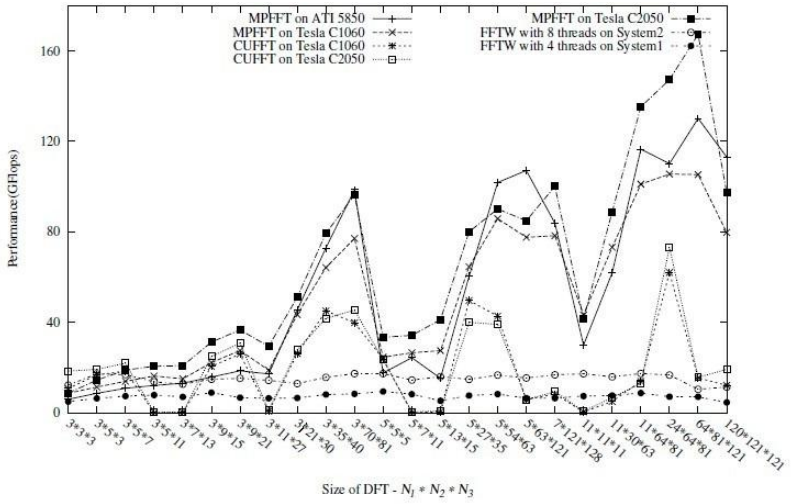


图 49 MPFFT 在不同测试平台上的性能比较

实验结果表明：一维 MPFFT 的性能相对于 clAmdFft 库，取得的加速比为 1.5x - 4x，接近于 CUFFT，是四线程 FFTW 的 6x -18x。二维 FFT 在 ATI 5850 上相对于 clAmdFft 取得的加速比为 1.5x -36.81x，平均加速比为 2.6x。此外，在 ATI 6870 上相对 clAmdFft 平均加速比 2.4x，最大 8.33x，是八线程 FFTW 的 6.52x，在 NVIDIA GPU 上平性能均达到了 CUFFT 的 90%。而三维 FFT 的性能，在 ATI 5850 和 ATI 6870 上相对 clAmdFft 的平均加速比分别为 1.81x 和 1.37x，同时，在 NVIDIA GPU 上性能也达到了 CUFFT 性能的 90%。此外，MPFFT 支持输入规模为非 2 的幂，相对 Tesla C2050 和 Tesla C1060 以及四线程 FFTW 的加速比分别为 1.5 x- 28x、20.01x、31.87x，由于 clAmdFft 对非 2 的幂支持不太完善，只支持 2i× 3j × 5k，因此在性能曲线图中没有列出。



非2的幂三维FFT性能

图 50 非 2 的幂三维 FFT 性能

发表学术论文论著情况

专利：

1. 专利发明人：孙相征，张云泉，王婷，刘芳芳，袁良；专利申请名称：针对稀疏矩阵的数据存储方法及基于该方法的 SpMV 实现方法;申请号：201010594672.1。
2. 专利发明人：袁良，张云泉，孙相征，王婷，刘芳芳；专利申请名称：稀疏矩阵的对角线数据存储方法及基于该方法的 SpMV 实现方法;申请号：201110004075.3。
3. 专利发明人：王茜，张先轶，张云泉；专利申请名称：基于 x86 架构的稠密矩阵乘法汇编代码自动生成方法；申请号：2012101997066。

软件著作权：

1. 计算机软件著作权登记，龙芯 CPU 多核高性能数学库软件（简称：CLeXML），登记号:2010SR060528。
2. 计算机软件著作权登记，行星流体力学大规模数值模拟并行软件 V1.0，登记号：2011SR008456。
3. 计算机软件著作权登记，X86 CPU 多核高性能数学库软件 CASIML V1.0 版，登记号：2011SR092896。
4. 计算机软件著作权登记，基于质谱的蛋白质非标记定量软件(MPI+CUDA 并行版) V1.0，登记号：2012SR067229。
5. 计算机软件著作权登记，基于质谱的蛋白质非标记定量软件(MPI 并行版) V1.0，登记号：2012SR067234。
6. 计算机软件著作权登记，基于质谱的蛋白质定量软件 QuantWiz V1.0，登记号：2012SR064275。
7. 计算机软件著作权登记，基于质谱的蛋白质非标记定量软件 V1.0，登记号：2012SR062946。
8. 计算机软件著作权登记，质谱可视化辅助分析软件 V1.0，登记号：2012SR062948。

译著：

《OpenCL 异构计算》，译者：张云泉，张先轶，龙国平，姚继锋。作者：Benedict R.Gaster、Lee Howes、David、R.Kaeli、 Perhaad Mistry、Dana Schaa。清华大学出版社，ISBN：9787302286851，277 页。2012 年 6 月 1 日， 第 1 版。

国际会议和期刊发表的代表性文章：

1. Haipeng jia, Yunquan Zhang, Guoping Long, Shengen Yan, and Jianliang Xu. GPURoofline: A Model for Guiding Performance Optimizations on GPUs. EuroPar 2012. To appear.
2. Haipeng jia, Yunquan Zhang, Guoping Long, Shengen Yan. An Insightful Program Performance Tuning Chain for GPU Computing. ICA3PP 2012. To appear.
3. Haipeng Jia, Yunquan Zhang, Weiyan Wang and Jianliang Xu. Accelerating Viola-jones Face Detection Algorithm on GPUs. In Proceeding of IEEE HPCC 2012. To appear.
4. Liang Yuan, Yunquan Zhang. A Locality-based Performance Model for Load-and-compute Style Computation. IEEE CLUSTER 2012. To appear.
5. Liang Yuan, Chen Ding, Daniel Stefankovic, Yunquan Zhang. Modeling the Locality in Graph Traversals. ICPP 2012. To appear.
6. Xiangzheng Sun, Yunquan Zhang, Ting Wang, Guoping Long, Xianyi Zhang, Yan Li: CRSD: Application Specific Auto-tuning of SpMV for Diagonal Sparse Matrices. Euro-Par (2) 2011: 316-327.
7. Yan Li, Yunquan Zhang, Haipeng Jia, Guoping Long, Ke Wang: Automatic FFT Performance Tuning on OpenCL GPUs. ICPADS 2011: 228-235.
8. Xiangzheng Sun, Yunquan Zhang, Ting Wang, Xianyi Zhang, Liang Yuan, Li Rao: Optimizing SpMV for Diagonal Sparse Matrices on GPU. ICPP 2011: 492-501.
9. Jing Wang, Yunquan Zhang, Xianyi Zhang, Xiangzheng Sun, Quanhu Sheng: QuantWiz: A scalable parallel software package for label-free protein quantification. BIC-TA 2010: 976-980.
10. Lei Wang, Yunquan Zhang, Xianyi Zhang, Fangfang Liu: Accelerating Linpack Performance with Mixed Precision Algorithm on CPU+GPGPU Heterogeneous Cluster. CIT 2010: 1169-1174.
11. Liang Yuan, Yunquan Zhang, Yuxin Tang, Li Rao, Xiangzheng Sun: LogGPH: A Parallel Computational Model with Hierarchical Communication Awareness. CSE 2010: 268-274.
12. Chao Yang, Yunquan Zhang, Ligang Li: Numerical Simulation of the Thermal Convection in the Earth's Outer Core. HPCC 2010: 552-555.
13. Liang Yuan, Yunquan Zhang, Xiangzheng Sun, Ting Wang: Optimizing Sparse Matrix Vector Multiplication Using Diagonal Storage Matrix Format. HPCC 2010: 585-590.
14. Yan Li, Yunquan Zhang, Ke Wang, Wenhua Guan: Heterogeneous Multi-core Parallel SGEMM Performance Testing and Analysis on Cell/B.E Processor. NAS 2010: 202-207.
15. Yunquan Zhang, Jiachang Sun, Guoxing Yuan, Linbo Zhang: Perspectives of China's HPC system development: a view from the 2009 China HPC TOP100 list. Frontiers of Computer Science in China 4(4): 437-444 (2010)
16. Chao Yang, Ligang Li, Yunquan Zhang: Development of a Scalable Solver for the Earth's Core Convection. HPCA (China) 2009: 497-502
17. Shengfei Liu, Yunquan Zhang, Xiangzheng Sun, RongRong Qiu: Performance Evaluation of Multithreaded Sparse Matrix-Vector Multiplication Using OpenMP. HPCC 2009: 659-665

18. Jing Wang, Yunquan Zhang, Xianyi Zhang, Xiangzheng Sun, Zelin Hu, Sujun Li, Rong Zeng: QuantWiz: A Parallel Software Package for LC-MS-based Label-Free Protein Quantification. HPCC 2009: 683-687
19. Yuan Yu, Yunquan Zhang, Ting Wang, Jiachang Sun, Xianyi Zhang, Yuxin Tang, Li Rao: Early Performance Evaluation of Dawning 5000A and DeepComp 7000. ICPADS 2009: 578-585

国内会议和期刊发表的代表性文章：

1. 孙相征，张云泉，王宣强，王磊。数值软件自适应性能优化搜索过程评价技术研究。计算机研究与发展，2010，Vol. 47，(4):679-686。
2. 刘胜飞，张云泉，孙相征。一种改进的 OpenMP Guided 调度策略研究。计算机研究与发展，2010，Vol. 47，(4):687-694。
3. 余元，张云泉，李会元。一类非张量积区域快速傅立叶变换算法在国产并行机上的可扩展性测试，《数值计算与计算机应用》，2010, 31(2)：123-130。
4. 陈少虎,张云泉,张先轶,程豪。BLAS 库在多核处理器上的性能测试与分析，软件学报，2010 年增刊
5. 袁良,张云泉,龙国平,王可,张先轶，基于延迟隐藏因子的 GPU 计算模型，软件学报，2010 年增刊
6. 孙相征，张云泉，王婷，杨超，李力刚。天体大规模数值模拟软件性能优化。华中科技大学学报(自然科学版)。2010 年 6 月，Vol. 38，增刊 I：69-76。
7. 程豪，张云泉，张先轶，李玉成。CPU-GPU 混合并行 DGEMM 算法实现与性能分析。《计算机工程》，Vol. 36，No.13，pp.24-26，2010 年 7 月。
8. 王磊，张云泉，刘芳芳，张先轶，基于混合精度算法的改进 HPL 软件包，《计算机工程》，Vol. 36，No.19，pp.47-49，2010 年 10 月。
9. 饶立，张云泉，李玉成。国产百万亿次机群系统 Alltoall 性能测试与分析。《计算机科学》，Vol. 8，2010 年。
10. 袁良，张云泉，并行程序设计语言中局部性机制的研究，《高性能计算发展与应用》，2010 年第一期:34-44。
11. 王婷，张云泉，孙相征，杨超，李力刚。天体大规模数值模拟软件性能优化初探。第二届中国国家网格学术年会（CNGRID2010）论文集, 2010 年 1 月 15-17，北京：87-92。
12. 王婷，张云泉，孙相征，杨超。行星流体动力学大规模计算的性能测试与分析。HPC China 2010，2010 年 10 月 28-30 日，北京。
13. 费辉，张云泉，王可。分子动力学模拟在 OpenCL 框架的 GPU 上的实现。HPC China 2010，2010 年 10 月 28-30 日，北京。
14. 李焱，张云泉,王可，赵美超。异构平台上基于 OpenCL 的 FFT 测试与分析。HPC China 2010，2010 年 10 月 28-30 日，北京。
15. 解庆春，张云泉,王可，李焱。SIMD 技术与向量数学库研究。HPC China 2010，2010 年 10 月 28-30 日，北京。
16. 费辉 张云泉 王可,许亚武. 基于 GPU 的分子动力学模拟并行化及实现. 计算机科学, 2011, (09): 275-279
17. 解庆春，张云泉， 王可，李焱，许亚武. SIMD 技术与向量数学库研究. 计算机科学, 2011, (07): 298-302
18. 李焱 张云泉 王可,赵美超. 异构平台上基于 OpenCL 的 FFT 实现与优化. 计算机科学, 2011, (08)

19. 颜深根，张云泉,龙国平,李焱. 基于 OpenCL 的归约算法优化研究. 软件学报，2011 年增刊
20. 李超,张云泉,郑昌文,胡晓慧, Intensity model with blur effect on GPUs applied to large-scale star simulators,软件学报, 2011 年增刊
21. 张先轶,王茜,张云泉, OpenBLAS:龙芯 3A CPU 的高性能 BLAS 库,软件学报, 2011 年增刊
22. 费辉，张云泉，王靖. 基于 GPU 的非标记定量软件 QuantWiz 并行化实现. HPCChina2011.
23. 袁良,张云泉. 基于横向局部性的多核计算模型 . HPC China 2011
24. 张樱,张云泉,龙国平. 基于 OpenCL 的图像模糊化算法优化研究. HPC China 2011
25. 吕渐春，张云泉，王婷，肖玄基， PLASMA 自适应调优与性能优化的设计与实现， HPC China 2011
26. 贾海鹏，张云泉，龙国平，徐建良，李焱，基于 OpenCL 的拉普拉斯图像增强算法优化研究， HPC China 2011
27. 颜深根，张云泉,龙国平,李焱. 基于 OpenCL 的归约算法优化研究. HPC China 2011 （大会优秀论文提名）
28. 李超,张云泉,郑昌文,胡晓慧, Intensity model with blur effect on GPUs applied to large-scale star simulators, HPC China 2011 （大会优秀论文提名）
29. 张先轶,王茜,张云泉, OpenBLAS:龙芯 3A CPU 的高性能 BLAS 库, HPC China 2011（大会优秀论文）

学术活动和社会兼职：

- 国家自然科学基金委第十四届信息科学部专家评审组成员(2012.6 -2014.6)
- IEEE CSE 2010 程序委员会共同主席
- SC2011 SotP, ICPADS 2012, PDCAT2012，IPDPS 2012, CCGrid 2012,e-Science 2012，CGO 2013 程序委员会委员。
- ISC 2012 Steering Committee Member
- 2010 年-2012 年全国高性能计算学术年会程序委员会副主席
- 第六届到第八届中国 CAE 工程分析技术年会专家委员会副主席
- 第一届和第二届中国科学院超级计算应用大会专家委员会委员
- 核心期刊《计算机科学》和《计算机工程与科学》编委
- 中国科学院《科研信息化技术与应用》编委
- 上海超算《高性能计算机发展与应用》编委
- 中国计算机学会《计算机学会通讯》编委
- 中科院超级计算中心用户委员会委员
- 中国计算机学会理事
- 中国软件行业协会常务理事
- 中国计算机学会高性能计算专委会秘书长
- 中国软件行业协会数学软件分会秘书长
- 中国传媒大学和华南理工大学兼职教授和博士生导师
- 中国科技大学兼职教授

大会特邀报告:

- Zhang Yunquan, SIAM PP 2012，Savanah, Georgia, USA, 2/17/2012。大会特邀报告。
- Zhang Yunquan, ISC 2011, Hamburg, Germany, June 19-23, 2011, Invited Speaker.
- Zhang Yunquan, GPU Technology Conference Asia For 2011，China National Convention Center，Beijing，China, Dec. 14-15，2011。Invited Speaker.
- 张云泉, 2011 年 7 月 25-27 日，第七届中国 CAE 工程分析技术年会，昆明,云南（大会特邀报告）
- 张云泉, 2011 年 10 月 26-10 月 29 日，2011 全国高性能计算学术年会(济南，大会特邀报告）
- 张云泉, 2011 年 12 月 21-12 月 23 日，2011 第七届全国高性能算法软件研究开发研讨会（北京，大会特邀报告)。
- 张云泉, 2011 年 11 月 6-11 月 9 日，第一届中国科学院超级计算应用大会（SCA2011），青岛,山东.大会特邀报告
- 张云泉, 2011 年 12 月 15-16 日，第二届中国科研信息化研讨会（北京，大会特邀报告)。
- Yunquan Zhang, ATIP 4th Workshop on HPC in China: Specialized Hardware & Software Development, SC2010，New Orleans, LA，Nov. 16-19, 2010。大会特邀报告。
- Yunquan Zhang, IDC 40th HPC User Forum, Beijing, Oct. 30, 2010. 大会特邀报告
- 2010 年 7 月 28-29 日，第六届中国 CAE 工程分析技术年会，哈尔滨，张云泉，大会特邀报告。
- 2010 年 10 月 27-10 月 30 日，2010 全国高性能计算学术年会，北京，张云泉，大会特邀报告。

外部经费争取情况

- 国家自然基金重点项目，《大规模异构并行系统的高效能调度理论与方法》（No.61133005），2012.1-2016.12，张云泉，子课题组长，87.64 万（总经费 280 万）。
- 国家 863 信息技术领域主题项目“新型多核/众核处理器编程与运行支撑环境”《面向国产处理器的并行程序综合优化技术与系统》（No. 2012AA010304)子课题《面向国产申威、飞腾处理器的高性能数学库》，2012.1-2015.12，90 万；
- 国家 863 信息技术领域主题项目“新型多核/众核处理器编程与运行支撑环境”《面向多核/众核处理器的并行程序编程技术、框架和语言支持》子课题《高效算法库和高效代码调优框架》，2012.1-2015.12，110 万；
- 国际企业 AMD 横向合作项目，特定 HPC 应用及 OPENCV 的移植和优化（AMD），2011.1-2013.12 ，张云泉,课题负责人，190 万。
- 国家测绘局测绘卫星应用中心横向合作项目，基于光线追踪的高精度成像并行数值计算软件，2011.7-2013.12 ，张云泉,课题负责人，80 万.
- “十一五”863 计划“高效能计算机及网格服务环境”重大项目“高性能计算与网格应用”课题，《天体大规模并行数值计算软件平台的研制》(No.2006AA01A125), 2007.1-2010.12,课题组副组长； 顺利通过结题验收，获得好评，80 万。
- 国防军工课题,《多核版高性能扩展数学库合作研发》，2009.1-2011.6，张云泉，课题组组长，顺利通过结题验收，420 万；
- 核高基重大专项（软件类)《支持国产 CPU 的编译系统及工具链》子课题《龙芯 CPU 多核并行国产高性能数学库研究开发》(2009ZX01036-001-002-3)， 2009.1-2011.6，张云泉，子课题组组长，正在结题； 391 万。
- 国家 863“高效能计算机及网格服务环境”重大专项课题,《曙光 6000 千万亿次高效能计算机系统研制》子课题《面向数万个以上处理器的新型基础算法研究》(No. 2009AA01A129), 2009.1-2010.12，张云泉，子课题组组长，顺利结题；200 万。
- 国家 863 “高效能计算机及网格服务环境”重大项目课题,《面向千万亿次计算机的并行算法库研制》（No. 2009AA01A134）《面向国产算法库的千万亿次大规模并行算法和性能优化方法研究》，2009.1-2010.12，张云泉，课题组副组长，顺利通过结题验收；35 万。
- 与 AMD 公司成立中科院软件所与 AMD “APU 软件联合研究开发中心”，任联合实验室主任；
- 与美国 Argonne 国家实验室数学与计算机科学部（MCS）成立 PPCT 联合实验室（a JOINT LAB FOR Parallel PROcessing and computing techniques Research），任联合实验室执行主任；

获表彰奖励情况

获得 2009 年，2010 年和 2011 年中国软件行业协会先进个人。
获得 2010 年中科院软件所优秀导师称号。

经费使用情况

2010 年本项目到账 45 万元。支出劳务费约 15 万元，用于学生及临时聘用人员劳务费支出；设备费约 5 万元，用于 PC 机、耗材购买；国际交流与合作费约为 3 万元，用于参加国际会议出访及接待来访交流外宾；其他业务费约为 12 万元，用于差旅费、会议费、版面费等；房屋水电费约 7.5 万元，用于项目人员使用的房屋水电费用；直接管理人员工资约为 2.5 万元。

2011 本项目到账 45 万元。支出劳务费约 15 万元，用于学生及临时聘用人员劳务费支出；设备费约 5 万元，用于 PC 机、耗材购买；国际交流与合作费约为 3 万元，用于参加国际会议出访及接待来访交流外宾；其他业务费约为 12 万元，用于差旅费、会议费、版面费等；房屋水电费约 7.5 万元，用于项目人员使用的房屋水电费用；直接管理人员工资约为 2.5 万元。

存在的问题、建议及其他需要说明的情况

无

备注：1、杰出青年人才入选者应实事求是地填写此报告，禁止弄虚作假；
2、此报告作为专项计划跟踪、管理的主要依据，报送人力资源处；
3、此报告将在所网站对外发布。