软件学报 ISSN 1000-9825, CODEN RUXUEW
Journal of Software, [doi: 10.13328/j.cnki.jos.006080]
©中国科学院软件研究所版权所有.
E-mail: jos@iscas.ac.cn
http://www.jos.org.cn
Tel: +86-10-62562563

面向异构融合处理器的性能分析、优化及应用综述*

张 峰[1,2], 翟季冬[3], 陈 政[1,2], 林甲灶[4], 杜小勇[1,2]

[1](数据工程与知识工程教育部重点实验室(中国人民大学),北京 100872)

[2](中国人民大学 信息学院,北京 100872)

[3](清华大学 计算机科学与技术系,北京 100084)

[4](北京大学 信息管理系,北京 100871)

通讯作者: 杜小勇, E-mail: duyong@ruc.edu.cn

摘 要: 随着异构计算技术的不断进步,CPU 和 GPU 等设备相集成的异构融合处理器在近些年得到了充分的发展,并引起了学术界和工业界的关注.将多种设备相集成带来了许多好处,例如,多种设备可以访问同样的内存,可以进行细粒度的交互.然而,这也带来了系统编程和优化方面的巨大挑战.充分发挥异构融合处理器的性能,需要充分利用集成体系结构中共享内存等特性;同时,还需结合具体应用特征对异构融合处理器上的不同设备进行优化.本文首先对目前涉及异构融合处理器的研究工作进行了分析,之后介绍了异构融合处理器的性能分析工作,并进一步介绍了相关优化技术,随后对异构融合处理器的应用进行了总结.最后,对异构融合处理器未来的研究方向进行了展望,并进行了总结.

关键词: CPU;GPU;异构融合处理器;性能分析;性能优化

中图法分类号: TP311

中文引用格式: 张峰,翟季冬,陈政,林甲灶,杜小勇. 面向异构融合处理器的性能分析、优化及应用综述.软件学报,2020,31. http://www.jos.org.cn/1000-9825/6080.htm

英文引用格式: Zhang F, Zhai JD, Lin JZ, Du XY. Survey on performance analysis, oOptimization, and applications of heterogeneous fusion processors. Ruan Jian Xue Bao/Journal of Software, 2020,31 (in Chinese). http://www.jos.org.cn/1000-9825/6080.htm

Survey on Performance Analysis, Optimization, and Applications of Heterogeneous Fusion Processors

ZHANG Feng[1,2], ZHAI Ji-Dong[3], CHEN Zheng[1,2], LIN Jia-Zao[4], DU Xiao-Yong[1,2]

[1](Key Laboratory of Data Engineering and Knowledge Engineering (Renmin University of China) , MOE, Beijing 100872, China)

[2](School of Information, Renmin University of China, Beijing 100872, China)

[3](Department of Computer Science and Technology, Tsinghua University, Beijing 100084, China)

[4](Department of Information Management, Peking University, Beijing 100871, China)

Abstract: With the development of heterogeneous computing technology, heterogeneous fusion processors, such as CPU-GPU integrated processors, have been fully developed in recent years, and arouse attention from both academia and industry. The fusion of different devices has several advantages. For example, all devices share the same memory and can have fine-grained cooperation. However, many system programming challenges and optimization challenges have emerged. To take full advantage of the capacity of heterogeneous fusion processors, we need to utilize features of heterogeneous fusion processors such as shared memory, and perform architecture optimizations to different devices according to different applications. We first analyze and summarize the research work related to heterogeneous fusion

* 基金项目: 国家重点研发计划(2016YFB0200100);国家自然科学基金项目(61732014, 61722208, 61802412)

Foundation item: National Key R&D Program of China (2016YFB0200100); National Natural Science Foundation of China (61732014, 61722208, 61802412)

收稿时间: 2019-01-31; 修改时间: 2020-04-09; 采用时间: 2020-05-07; jos 在线出版时间: 2020-05-26

processors. Second, we introduce the related work about performance analysis. Third, we summarize the optimizations on heterogeneous fusion processors. We also provide a summarization for the applications that utilize heterogeneous fusion processors. At last, we provide the future directions on heterogeneous fusion processors and give conclusion.
Key words: CPU; GPU; heterogeneous fusion processors; performance optimization; performance analysis

近些年,GPU 由于具有高并行性的特点,在高性能计算等领域得到了充分的应用.然而,由于 GPU 特有的以线程组 SIMT 的执行方式执行指令,对访存方式有较高的要求,通常越规则、密集的计算任务越容易发挥 GPU 的硬件性能,反之,性能较弱;同时,GPU 的内存和 CPU 的内存是分离的,对于大数据应用,有限的内存容量和经过 PCIe 的数据传输也是影响 GPU 可用性的一大挑战.针对这一问题,硬件设计厂商推出了将 CPU 和 GPU 集成在一个芯片的异构融合处理器,兼顾两者优点.例如,AMD 公司在 2011 年推出了 APU 处理器[1],Intel 公司在 2012 年推出了带有集成 GPU 的处理器[2],而 Nvidia 公司也在 2014 年推出了 Tegra 处理器[3].这种异构融合架构能够同时结合 CPU 和 GPU 的体系结构特征,对异构计算进行了扩展,为高性能计算等领域带来了新机会[52,53].

和普通的离散 GPU 设计相比,异构融合处理器具有诸多优势.第一,CPU 和 GPU 能够共享内存,这种共享内存的设计可以避免 CPU 和 GPU 间通过 PCIe 进行数据传输.Intel 公司还推出了 CPU、GPU 共享缓存的异构融合处理器[3],使不同设备间的通信更高效.第二,CPU 和 GPU 可以访问同样的数据,这样 CPU 和 GPU 可以有更多的交互.这些优势使得异构融合处理器具有更广泛的应用前景,但如何利用好异构融合处理器仍充满挑战.首先,异构融合处理器还处于发展阶段,许多设计并不完善.其次,充分发挥异构融合处理器的性能需要对 CPU 和 GPU 进行不同的体系结构优化,并且需要考虑额外的多设备任务分配,这增加了异构融合处理器的编程难度.再次,异构融合处理器上相集成的设备具有多样性,不仅局限于 GPU、CPU 设备的集成;例如,目前也出现了 FPGA 和 CPU 相集成的异构融合处理器.FPGA 著名生产厂商 Altera 被 Intel 公司收购后也推出了 FPGA-CPU 相融合的处理器[4],将 FPGA 和 CPU 集成在一起,能够使 FPGA 和 CPU 更好地交互.

面对这些挑战,为了更充分地利用异构融合处理器,首先需要对其进行性能分析,充分理解异构融合处理器的行为.目前对异构融合处理器的利用方式,可以分为单程序独占异构融合处理器利用不同设备特点运行,以及多程序在异构融合处理器上混合运行.其中,对于单程序的运行,并不是在所有情况下同时使用多种设备混合运行可以达到最优的性能结果;实验表明,对于 CPU、GPU 相集成的异构融合处理器,根据程序的性能表现大体可分为四类:利用异构融合处理器上的多种设备混合执行能够带来加速效果的程序, GPU 主导型程序、CPU 主导型程序、和无设备倾向型程序[74].每种程序类型都具有不同的性能特征[21],例如,多种设备混合执行能够带来加速效果的程序不需要较高的带宽;而只使用 GPU 设备达到最佳性能的 GPU 主导型程序有高并行度等特点,这样可更充分地利用 GPU 资源,包括计算处理单元和可控制的局部存储等;而 CPU 主导型程序具有低并行度等特征[74].而对于多程序在异构融合处理器上的混合运行,存在性能降级,影响因素也更复杂.研究表明,操作系统的上下文切换在多程序混合运行中起了关键作用[24,25],目前,不同程序多设备混合运行性能降级的主要原因在于 CPU 内的竞争,而不是对内存带宽的竞争;操作系统中的上下文切换策略对混合运行中的 GPU 性能影响较大,CPU 主机端部分的 GPU 控制线程是混合运行性能降级的主要原因.此外,功耗也是处理器设计的一个重要指标,目前异构融合处理器通常具有低功耗的特性.

基于对异构融合处理器性能特征的理解,编程时可从异构融合处理器所引入的新特性、设备体系结构特征等方面进行优化,具体可分为针对不同设备进行数据访问方式调整、并行粒度优化、使用局部存储的方式调整等[6]. 同时, 将负载向异构融合处理器的具体设备分配时, 有多种优化策略进行选择. Zhang 等人[6]提出在异构融合处理器上基于建模的方法混合运行 CPU、GPU 设备. 首先, 使 CPU 处理一部分负载, 记录时间, 并预测 CPU 的处理速度; 之后, 用同样的方法预测 GPU 的处理速度. 在获得 CPU、GPU 的测试性能后, 对剩余负载依据 CPU 和 GPU 的性能对比进行划分. Pandit 等人[12]研究不需要采样就可以同时混合运行离散 GPU 和 CPU 的方法, 这一方案对任务的分配是通过 OpenCL 编程中的 workgroup[7]在 CPU 和 GPU 端的分配完成的.目前也有关于异构融合处理器功耗优化的研究工作,Zhu 等人[42]研究在异构融合处理器上如何混合运行多个程序使处理器运行的功耗较低.这项研究从功耗的角度进行了分析,并提出了相应混合运行调整策略. Garzón 等人[43]针对迭代计算提出了异构融合处理器上的功耗优化方案 E-ADITHE,该方案包含启发式的方法,基于功耗预测选择合

适的处理设备,同时也会考虑异构处理器内部的负载均衡特性. 此外, 异构融合处理器正被用于越来越多的领域, 结合具体的应用, 异构融合处理器往往具有更多的优化空间.

通过对以往研究工作的总结,本文对异构融合处理器研究的未来发展方向进行了展望.首先,结合异构融合处理器的不断演变的发展过程,未来的异构融合处理器会集成更多的计算核心,性能和稳定性等方面也会增强,可用于高性能计算机的设计.其次,对于异构融合处理器,目前需要针对不同设备进行程序优化才能使其达到较优的性能,编程要求较高,未来的异构融合处理器可能会集成更多种类的设备,编程情况会更复杂,而这些情况会促使新的编程语言设计和辅助工具出现.再次,对于大数据类数据密集型应用,由于异构融合处理器具有共享内存等特征,会有越来越多的大数据应用利用异构融合处理器进行处理.而对于机器学习等新型计算密集型应用,异构融合处理器集成了不同的设备特性,对于特定机器学习应用会有新突破.

本文首先对异构融合处理器的体系结构特征和编程语言进行了介绍(第 1 节),并从研究方向、成果表现等方面对以往相关工作进行了分类与汇总(第 2 节).之后,对异构融合处理器上的性能分析类工作进行了总结(第 3 节),并进一步对涉及异构融合处理器的优化工作进行了汇总(第 4 节),对异构融合处理器的具体应用也进行了介绍(第 5 节).随后,本文对异构融合处理器未来的研究方向进行了展望(第 6 节).最后对全文进行了总结(第 7 节).

1 异构融合处理器概述

本节主要对异构加速器件和异构融合处理器进行基本情况介绍,包括异构加速器件简介、体系结构特征和编程语言.

1.1 异构加速器件

随着信息技术的发展,需要处理的数据量增大,来自科学计算等领域的应用需求的复杂性也进一步增大,传统 CPU 无法满足大量的计算需求.目前,常见的解决方案是在系统中增加 GPU 等异构加速器件,这样可以在对系统改动较小的情况下极大地增加系统的处理能力.配有异构加速器件的计算系统通常由两部分构成:主机端和异构加速器件端.通常主机端由通用 CPU 构成,异构加速器件端体现系统的主要计算能力.异构加速器件具有多种选择,如 GPU、Xeon Phi、FPGA 等,目前最为流行的加速器件是 GPU.GPU 是由大量并行处理单元构成的,这些处理单元包括了上千个轻量级的计算核心.这些轻量级的计算核心可以访问共同的全局内存,但个并行处理单元包含的计算核心具有各自独立的局部存储.异构加速器件常见的编程语言包括 OpenCL[7]和 CUDA[8],在编写程序过程中需要对异构加速器件的体系结构有所了解;目前也存在更高层的编程工具,如 Matlab 等,但这些工具通常无法充分发挥异构加速器件性能.

对于离散的异构加速器件,需要将数据从主机端传输到异构加速器件的内存中,数据计算完成再将结果传输回主机端.数据传输会引起时间开销,这种主机端和异构加速器件之间的交互对于计算系统性能影响很大,因此,在离散架构存在的同时,出现异构融合架构,将异构加速器件与主机端集成在一个芯片上,构成异构融合处理器,共享内存等资源,避免了数据传输开销.但异构融合处理器的实现具有诸多挑战,例如如何有效利用芯片面积,如何解决芯片功耗密度上升问题等.

1.2 体系结构

尽管异构融合处理器的设计具有诸多挑战,目前仍出现了一些异构融合架构,如 AMD 公司的 Carrizo[56]、Intel 公司的 Skylake[57]、和 Nvidia 公司的 Denver[58]等.图 1 以 CPU、GPU 相集成的异构融合处理器为例进行了展示,图 1 中的异构融合处理器分为 CPU 部分和 GPU 部分,这两部分可访问同样的内存.CPU 部分具有 L1 和 L2 等多级缓存结构,不同型号的 CPU 略有不同,例如,AMD 公司生产的 A10-7850K 异构融合处理器的 CPU 部分具有共享的 L2 缓存,而 Intel 公司生产的 i7-4770R 异构融合处理器则采用私有 L2 缓存设计.对于 GPU 部分,不同硬件厂商生产的 GPU 存在较大的体系结构差异,包括局部存储和缓存设计等方面.例如,相比于 A10-7850K 异构融合处理器,i7-4770R 异构融合处理器上存在共享 L3 缓存.此外,部分异构融合处理器上出现了

多设备共享缓存和嵌入式存储(embedded DRAM, EDRAM)等结构.对于共享内存访问,目前对 CPU 和 GPU 等设备提供的带宽不一样[5].

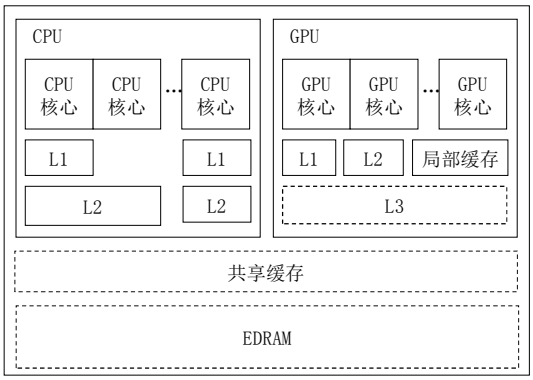


Fig.1 CPU and GPU integrated heterogeneous fusion processor[6]

图 1 CPU、GPU 相集成的异构融合处理器

目前针对异构融合处理器的研究工作都是围绕上述类型体系结构展开的.研究者围绕多设备相集成的特点结合应用在异构融合处理器上进行各种性能分析与优化.由于异构融合处理器将不同设备集成在一起共享内存,和离散架构相比需要考虑更多因素,如多设备间如何保证内存一致性等问题.异构融合处理器共享内存的特性避免了通过 PCIe 的数据传输,但也增加了不同设备维护共享内存一致性的开销[55].

关于异构融合处理器的共享缓存,Yang 等人[47]提出了在 GPU 运行时使 CPU 进行辅助预取的工作.首先,从 GPU 需要运行的程序中抽取出和内存访问相关的地址计算指令,增加 CPU 循环操作准备为不同的线程预取数据.其次,当 GPU 程序开始运行时,CPU 提前运行预先抽取出的内存操作指令,使 GPU 需要访问的数据提前存入缓存.为了保证 CPU 预取的有效性,通过观察缓存缺失率改变循环迭代的步长. 此外,对于 CPU、GPU 异构融合处理器,GPU 的高吞吐率会使 CPU 的一致性维护开销较大.Power 等人[46]提出了基于目录结构的针对 CPU、GPU 异构融合处理器的一致性协议(HSC,heterogeneous system coherence),为 CPU 和 GPU 的 L2 缓存开设局部存储区,用来记录被 CPU 或 GPU 控制的区域.这种设计的空间开销小,同时也比传统的块级目录方法简单.由于维护 CPU 和 GPU 间的缓存一致性存在开销,Agarwal 等人[54]提出了选择性缓存策略(selective caching),使 GPU 不缓存需要维护一致性的数据,降低共享缓存所带来的负担.

继 CPU、GPU 相集成到一个芯片后,各处理器生产厂商也相继推出了 CPU、FPGA 相集成的异构融合处理器.例如,Intel 公司设计了使用快速通路(QPI,quick path interconnect)替换 PCIe 连接的 CPU-FPGA 集成方案.Choi 等人[48]分析、对比了通过 PCIe 与 CPU 相连接的离散 FPGA 和通过 QPI 与 CPU 连接的 FPGA 性能,指出通过快速通路相集成的 CPU-FPGA 方案在细粒度通信等方面具有更低的延迟,而 FPGA_DRAM 的重用率决定这两个平台的最大有效带宽.通过高效的设计,使用 PCIe 连接的 CPU-FPGA 平台能够在很多方面达到和使用 QPI 平台类似的效果.同时,该研究指出目前通过 PCIe 连接和通过 QPI 连接的方式都有较大的性能提升空间,这一研究可以为 CPU-FPGA 异构融合处理器的未来设计提供参考.对于大数据应用,Cong 等人[49]进一步提出了数据流执行模型和 FPGA、CPU 间的调度算法,该方案能够使 CPU 和 FPGA 间更好地进行交互,提升系统整体资源利用率.

1.3 编程语言

异构融合处理器的编程语言主要选用 OpenCL 并行编程语言[7].OpenCL 可控制异构融合处理器上的多种设备.相比于 Pthreads[10]、OpenMP[9]、CUDA[8]等语言,OpenCL 的主要优点在于所支持的硬件设备的广泛性.自 Khronos 小组 2009 年公布 OpenCL 编程标准后,各大硬件设备生产厂商纷纷支持 OpenCL 的编程标准,开发

了各自硬件产品相应的 OpenCL 编程库.因此,可以用 OpenCL 控制绝大多数设备,如 CPU、FPGA 和 GPU 等设备,这是其他的并行编程语言所无法做到的.目前异构融合处理器上的相关编程工作也主要以 OpenCL 的编程为基础进行.

OpenCL 的编程流程较为复杂,首先需要建立程序所需要的上下文环境(context),并在上下文环境中创建设备(device),如异构融合处理器中的 CPU 和 GPU 等.其次,需要为不同的计算设备建立相应的命令队列(command queue),用于运行所需的计算核心(kernel).再次,需要分配资源,指定具体的线程组织方式.OpenCL 中以线程组(workgroup)的形式对线程进行组织,并将线程视为工作项(workitem).最后,通过 OpenCL 的 clEnqueueNDRangeKernel()函数发射计算核心到相应设备的命令队列中进行运算,获得结果,并显式释放资源.在 OpenCL 的计算核心运行过程中,一个线程组只可在同一设备所提供的计算单元(computing unit)中执行,而线程组内的线程可通过局部存储(local memory)和屏障操作(barrier)进行组内交互和同步.OpenCL 中不存在所有线程的屏障同步操作.此外,OpenCL 提供 clFush()函数,可确保设备命令队列中的指令发射到了相应设备,而 clWaitForEvents()函数可用于确保完成全部命令.

2 研究概况

本节对过往针对异构融合处理器的研究进行了总结. 通过对所有涉及异构融合处理器的研究进行整理,表 1 列出了近年来关于异构融合处理器的研究汇总,概括地描述了这些研究工作的方向和研究内容.研究方向分为异构融合处理器的性能分析、异构融合处理器的性能优化、和异构融合处理器的具体应用实例.其中,性能分析是研究的基础,在充分了解异构融合处理器特性的基础上,进一步探索各类优化技术,最后,将异构融合处理器使用在各领域的应用实例中.表 1 中具体成果表现又可进一步细分:性能分析类研究可进一步细分为单程序运行性能分析、多程序运行性能分析、和涉及处理器功耗的相关分析研究;涉及性能优化的研究目前可分为体系结构优化、负载划分与调度优化、以及针对处理器功耗的优化;而异构融合处理器的具体应用目前主要涉及科学计算和数据科学领域.表 1 同时列出了论文标题、发表年份、作者信息、研究内容概括、和在本文中的引用编号.

图 2 描述了表 1 中各方向的论文分布情况.性能分析类论文有 16 篇;性能优化类论文有 14 篇,主要集中在不同设备间的负载划分与调度优化.涉及具体应用实例的论文有 23 篇.目前基于异构融合处理器的应用多集中在数据科学领域和科学计算领域,但随着近年来机器学习技术的发展,研究者开始考虑如何将异构融合处理器应用于机器学习领域,如深层神经网络训练的加速等环节.虽然目前这个领域的研究较少,但具有较大的发展潜力.此外,也有研究者提出了利用异构融合处理器对网络数据包进行处理的解决方案,这也是一个异构融合处理器潜在的应用方向.

图 3 描述了各研究方向的论文随时间线的增长图:从图中可以看出,关于异构融合处理器的研究从 2010 年开始快速增加,这与计算机技术的发展密切相关.正是在 2010 年前后,AMD 宣布推出集成 CPU 和 GPU 的异构融合处理器,以此为契机,关于融合处理器的研究开始被大量研究者作为研究课题.到目前为止,关于异构融合处理器的性能分析优化和基于异构融合处理器的应用是这个领域的热点方向.

Table 1 Summary of surveyed papers

表 1 论文总结

题	发表年份	作者	研究内容
efficacy of a fused CPU+ GPU processor (or APU) for parallel computing	2011	Daga M 等	设计基准程序评测成熟测试异构融合
leoffs of fused memory hierarchies in heterogeneous computing architectures	2012	Spafford KL 等	异构融合处理器的缓存和内存使用
terizing the performance benefits of fused CPU/GPU systems using FusionSim	2013	Zakharenko V 等	Rodinia 基准测试程序集在异构融
ance characterization of data-intensive kernels on AMD fusion architectures	2013	Lee K 等	AMD 体系架构下数据密集内核的性
run, or not to co-run: A performance study on integrated architectures	2014	Zhang F 等	比较研究 Rodinia 程序集分别在融
ng memory management methods on integrated CPU-GPU systems	2017	Dashti M 等	集成 CPU&GPU 的内存管理方法
: Irregularity-aware fine-grained workload partitioning on integrated architectures	2017	Zhang F 等	针对不规则负载程序在异构融合处
ance evaluation and analysis of sparse matrix and graph kernels on heterogeneous processors	2019	Zhang F 等	异构融合处理器上稀疏矩阵和稀疏图
ing shared last-level cache in a heterogeneous multicore processor	2013	Mekkat V 等	异构融合处理器上不可预测的缓存
anding co-run degradations on integrated heterogeneous processors	2014	Zhu Q 等	异构融合处理器上多个独立的共享
anding co-run performance on CPU-GPU integrated processors	2017	Zhu Q 等	分析在异构融合处理器上程序协同运
run, or not to co-run: A performance study on integrated architectures	2014	Zhang F 等	比较研究 Rodinia 程序集分别在融
n Scheduling with Power Cap on Integrated CPU-GPU Systems	2017	Zhu Q 等	在考虑功率上限的异构融合处理器
ing the accelerated processing units for seismic imaging: A performance and power efficiency comparison against CPUs and GPUs	2017	Said, I 等	比较三维声学逆时在异构融合处理
of Workload Distribution on Energy Consumption, Performance, and Reliability of Heterogeneous Devices	2019	Dávila G P 等	负载分配对异构融合处理器功耗性
mance, energy consumption and reliability evaluation of workload distribution on heterogeneous devices	2019	Dávila G P 等	异构融合处理器在性能、功耗等方
tm apping of irregular C++ applications to integrated GPU	2014	Benk R 等	将不规则 C++应用有效映射到异构
anding co-running behaviors on integrated CPU/GPU architectures	2017	Zhang F 等	异构融合处理器的性能优化探究
e Heterogeneous Scheduling for Integrated GPUs	2014	Kaleem R 等	集成 CPU&GPU 的自适应调度算法
kernels: Cooperative execution of OpenCL programs on multiple heterogeneous devices	2014	Pandit P 等	FluidiCL:在多个异构设备上协同执
multi-resource fairness: balancing fairness and efficiency in coupled CPU-GPU architectures	2016	Tang S 等	融合 CPU&GPU 架构下公平资源的
enting directed acyclic graphs with the heterogeneous system architecture	2016	Puthoor S 等	针对异构系统架构给出细粒度地
hregularity-aware fine-grained workload partitioning on integrated architectures	2017	Zhang F 等	针对不规则负载的多设备混合运行
anding co-running behaviors on integrated CPU/GPU architectures	2017	Zhang F 等	基于概要分析的负载分配模型
r-Fly Workload Partitioning for Integrated CPU/GPU Architectures	2018	Younghwan Go 等	异构融合处理器上针对不规则应用
utic Irregularity-Aware Fine-Grained Workload Partitioning on Integrated Architectures	2019	Zhang F 等	异构融合处理器对不规则负载的多
ng AMD's Kaveri APU for heterogeneous computing	2014	Bouvier D 等	基于 APU 的异构计算机提供更强的
entation and evaluation of deep neural networks (DNN) on a mainstream heterogeneous system	2014	Gu J 等	在异构融合处理器上加速多层感知
oach to optimise the energy efficiency of iterative computation on integrated GPU-CPU systems	2017	Garzón EM 等	一种优化集成 GPU-CPU 系统迭代计
n Scheduling with Power Cap on Integrated CPU-GPU Systems	2017	Zhu Q 等	在异构融合处理器上调度多程序协
erizing and evaluating a key-value store application on heterogeneous CPU-GPU systems	2012	Hetherington TH 等	键值储存存在异构融合处理器上的性
ng coarse-grained parallelism in B+ tree searches on an APU	2012	Daga M 等	在异构融合处理器上实现 B+树的负
nting MapReduce on a coupled CPU-GPU architecture	2012	Chen L 等	在异构融合处理器上加速 MapReduc
ng co-processing for hash joins on the coupled CPU-GPU architecture	2013	He J 等	利用异构融合处理器实现数据库查
e query co-processing on coupled CPU-GPU architectures	2014	He J 等	异构融合处理器上基于缓存的数据
efficient MapReduce workload acceleration using integrated-GPU	2015	Kim S 等	使用异构融合处理器实现高效的 M
)ynamic pipelines for in-memory key-value stores on coupled CPU-GPU architectures	2017	Zhang K 等	异构融合处理器上的内存数据库的流
eam: Fine-Grained Window-Based Stream Processing on CPU-GPU Integrated Architectures	2020	Zhang F 等	面向异构融合处理器的流处理系统
ing APU oriented scientific computing applications in OpenCL	2011	Doerksen M 等	如何在 APU 上优化 OpenCL 程序
arison of the FDTD algorithm implemented on an integrated GPU versus a GPU configured as a coprocessor	2012	Tagner RG 等	基于共享内存的有限差分时域算法
and k sort on the AMD fusion accelerated processing unit	2013	Delorme MC 等	利用异构融合处理器实现并行基数
up: An efficient heap data structure for asymmetric multicore processors	2014	Liu W 等	异构融合处理器上的高效堆数据结
t breadth-first search on a heterogeneous processor	2014	Daga M 等	在异构融合处理器上实现高效的广
strategy for stencil computations on the APU	2014	Eberhart P 等	异构融合处理器上的 Stencil 计算
ctive segmented sum for sparse matrix-vector multiplication on heterogeneous processors	2015	Liu W 等	在异构融合处理器上实现稀疏矩阵
work for general sparse matrix matrix multiplication on GPUs and heterogeneous processors	2015	Liu, W 等	在异构融合处理器上实现高效的稀

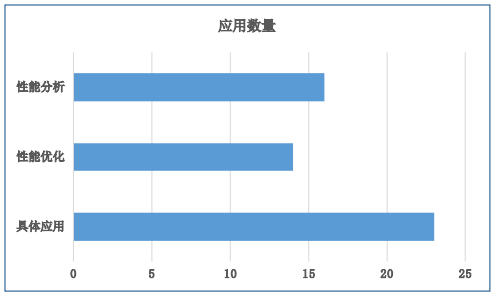


Fig.2 The amount of papers in each category

图 2 各领域的论文数量

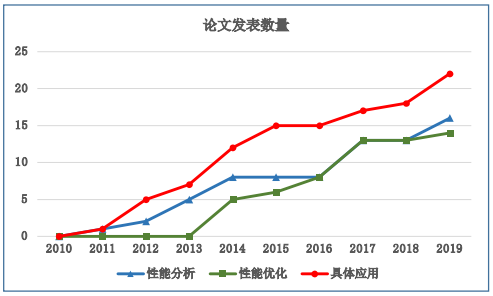


Fig.3 Paper pulication increasing trend

图 3 论文数量增长

对于表 1 中的应用,为了进一步分析这些应用和优化技术间的关系,本文将各应用对体系结构优化、负载划分与调度优化和功耗优化的利用情况总结如表 2 所示,发现所有应用都会结合异构融合处理器中的异构设备进行体系结构的优化.23 个应用中有 10 个应用针对不同设备进行了负载划分与调度优化,超过半数的应用仅使用了异构融合处理器上的加速设备,没有考虑对 CPU 的利用情况;这些研究证明仅使用异构融合处理器上的加速设备也可以带来性能加速效果,性能提升的主要因素在于避免了 CPU 和异构设备间的数据传输开销.此外,表 2 还说明目前绝大多数应用主要考虑系统的计算能力,考虑功耗方面的优化较少.随着异构融合处理器的性能不断提升,以及由异构融合处理器搭建的低功耗集群越来越常见,预计未来会有更多应用研究将功耗作为优化指标出现.

Table 2 Summary of the surveyed applications in terms of optimizations

表 2 各应用所用优化方法汇总

具体应用	论文标题	论文编号	体系结构优化	负载划分与调度优化	功耗优化
数据密集型应用	Characterizing and evaluating a key-value store application on heterogeneous CPU-GPU systems	37	✓		
	Exploiting coarse-grained parallelism in B+ tree searches on an APU	29	✓		
	Accelerating MapReduce on a coupled CPU-GPU architecture	17	✓	✓	
	Revisiting co-processing for hash joins on the coupled CPU-GPU architecture	18	✓	✓	
	In-cache query co-processing on coupled CPU-GPU architectures	38	✓	✓	
	Power efficient MapReduce workload acceleration using integrated-GPU	44	✓		✓
	Dido: Dynamic pipelines for in-memory key-value stores on coupled CPU-GPU architectures	19	✓	✓	
	FineStream: Fine-Grained Window-Based Stream Processing on CPU-GPU Integrated Architectures	73	✓	✓	
计算密集型应用	Designing APU oriented scientific computing applications in OpenCL	28	✓		
	A comparison of the FDTD algorithm implemented on an integrated GPU versus a GPU configured as a co-processor	33	✓		
	Parallel radix sort on the AMD fusion accelerated processing unit	30	✓	✓	
	Ad-heap: An efficient heap data structure for asymmetric multicore processors	36	✓		
	Efficient breadth-first search on a heterogeneous processor	32	✓	✓	
	Hybrid strategy for stencil computations on the APU	31	✓		
	Speculative segmented sum for sparse matrix-vector multiplication on heterogeneous processors	35	✓		
	A framework for general sparse matrix-matrix multiplication on GPUs and heterogeneous processors	61	✓		
	An adaptive breadth-first search algorithm on integrated architectures	59	✓	✓	✓
	Asw: accelerating Smith–Waterman algorithm on coupled CPU–GPU architecture	65	✓	✓	
	Non-uniform domain decomposition for heterogeneous accelerated processing units	66	✓	✓	

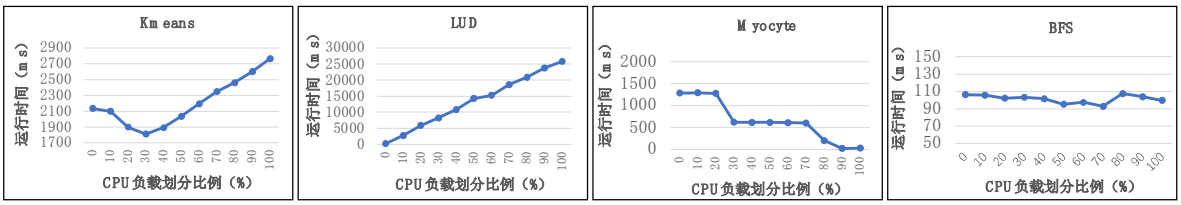
其他领域	Implementation and evaluation of deep neural networks (DNN) on mainstream heterogeneous systems	41	✓		✓
	APUNet: Revitalizing GPU as Packet Processing Accelerator	51	✓		
	Hash-based OpenFlow Packet Classification on Heterogeneous System Architecture	68	✓		✓
	Parallel implementations of frame rate up-conversion algorithm using OpenCL on heterogeneous computing devices	69	✓		

3 异构融合处理器的性能分析

性能分析是体系结构研究的一个重要方面, 可以帮助理解计算机的行为表现. 本节主要介绍异构融合处理器的性能分析工作, 分为单程序在异构融合处理器上利用多设备运行的性能分析研究, 多程序同时在异构融合处理器上混合运行的分析研究, 以及异构融合处理器上的功耗分析研究.

3.1 单程序运行性能分析

单程序运行的性能分析指分析一个程序在异构融合处理器的不同设备混合运行时的性能表现. Zakharenko 等人[20]利用模拟器对 CPU、GPU 相融合的处理器进行分析,模拟出 Rodinia 基准测试程序集在异构融合处理器上的性能收益.Zhang 等人[21]对 CPU-GPU 异构融合处理器上的程序运行结果进行了分析,将程序分为四种类型：(1)多种设备混合运行能够带来性能收益的程序(co-run friendly programs),(2)仅运行 GPU 可获得最佳性能的 GPU 主导型程序(GPU-dominant programs),(3) 仅运行 CPU 可获得最佳性能的 CPU 主导型程序(CPU-dominant programs),(4)对不同设备进行负载划分程序性能差异不大的无倾向型程序(ratio-oblivious programs).这四种程序类型的性能表现如图 4 所示.



(a) Co-run friendly programs (b) GPU-dominant programs (c) CPU-dominant programs (d) Ratio-oblivious programs

Fig.4 Different performance types[21]

图 4 不同种类的性能表现

其中,多种设备混合运行获得性能收益的程序具有相对长的运行时间(计算核心的开启时间开销可忽略),且对带宽的要求不高;而 GPU 主导型程序常包含关于局部存储(local memory)的使用与优化,且有高度并行性;CPU 主导型程序的并行度不高,运行时间可能不长,尽管 CPU 的计算性能远低于 GPU,但在这些情况下 GPU 无法充分发挥其强大的计算能力;无倾向性程序的性能瓶颈常在于内存访问带宽,或所选择的计算核心执行时间占比相对低,这样对不同设备进行负载划分,异构融合处理器的性能表现差异不显著[74].

对于不规则负载在 CPU-GPU 异构融合处理器上的处理,Zhang 等人[14]进一步分析,指出对于不规则负载运行时需要考虑更多的细节,因为 GPU 和 CPU 对不规则负载的敏感程度不同.以稀疏矩阵为例,令稀疏矩阵中每行中的非零元素的数目变化的方差表示负载的不规则程度,对于大量具有不同不规则程度的矩阵,随着不规则程度的增加,GPU 性能表现出了明显的下降趋势;而相应的 CPU 性能没有类似趋势.因此,CPU 对负载的不规则变化不如 GPU 敏感,在异构融合处理器上对不规则负载进行程序优化时需要结合不规则负载特性. Zhang 等人[67]在异构融合处理器上对稀疏矩阵计算程序和图算法程序进行了对比,指出图程序通常计算模式更为复杂,计算过程可包含多伦迭代,且每次涉及不同的子图的部分,具有动态特征,需要进一步优化.

Spafford 等人[22]对异构融合处理器的缓存和内存使用情况进行了研究,分析吞吐率和延迟,还在 CPU 对内存不断访问的情况下对 GPU 的程序运行进行分析,指出目前异构融合处理器在资源配置管理等方面存在提升

空间.异构融合处理器具有共享内存的特性,Daga 等人[23]设计基准测试程序集进行测试,包括数据传输程序、计算核心函数、真实应用场景等,实验结果表明异构融合处理器和离散的 GPU 相比在数据传输方面有明显的优势. Dashti 等人[55]指出异构融合处理器中不同设备维护共享内存一致性存在开销.Lee 等人[5]分析了 AMD Llano 异构融合处理器上的不同设备访问内存的读写带宽情况.OpenCL 语言中可以使用不同关键字声明数组的类型,如主机端(host buffer)类型、设备端(device buffer)类型,这一研究结果表明 CPU、GPU 读写主机端和设备端数据时具有不同的性能表现.在异构融合处理器程序设计时要充分考虑这些因素.

3.2 多程序的性能分析

研究者也关注如何在异构融合处理器的不同设备上运行不同程序.Zhu 等人[24,25]研究让两个程序分别运行在异构融合处理器的 CPU 和 GPU 设备上,分析和单个程序运行在处理器上相比的性能下降情况.这一研究首次指出操作系统的上下文切换在多程序混合运行中起了关键作用.研究的主要发现包括:(1)不同程序多设备混合运行性能降级的主要原因在于 CPU 内的竞争,而不是内存的竞争;(2)操作系统中上下文切换策略对混合运行的 GPU 性能影响较大;(3)CPU 主机端部分的 GPU 控制线程是混合运行性能降级的主要原因;(4)数据拷贝对性能有轻微影响;(5)操作系统的功耗管理对 GPU 的性能也有潜在的影响.最后,这一研究提出了异构融合处理器设计新的管理方案.

部分异构融合处理器存在 CPU、GPU 共享的缓存(shared last-level cache),Mekkat 等人[26]研究不同类型程序在 CPU、GPU 设备上混合运行对缓存的影响,研究指出由于 GPU 具有远多于 CPU 的硬件线程,在目前的缓存管理策略中,如果存在多程序运行,运行在 CPU 上的应用对共享缓存的利用会被运行在 GPU 上的应用干扰,对于缓存敏感型程序性能降级明显.而对于 GPU,大部分应用具有足够的并行度,能够容忍由于共享缓存访问缺失所带来的访存延迟.该研究进一步提出了异构缓存管理策略(HeLM,Heterogeneous LLC Management),能够利用 GPU 对访存延迟不敏感的特性,阻止 GPU 对共享缓存的访问,以腾出更多的缓存空间供 CPU 使用.

3.3 功耗分析

功耗是评估异构融合处理器的另一重要指标,许多硬件厂商如 AMD 和 Intel 将异构融合处理器视为性能和功耗的折中方案.Zhang 等人[21]对异构融合处理器进行了功耗分析,并和离散的 GPU 进行了比较.研究指出,对于运行同样的程序,并非在所有情况异构融合处理器均可能耗最少.因为目前同一时期的异构融合处理器计算能力往往低于离散 GPU,尽管在程序运行过程中异构融合处理器具有较低的功率,但运行时间也更长,耗能也可能更高.Said 等人[62]以三维声学处理程序为例,在单机和集群环境中对 CPU-GPU 异构融合处理器和离散的 CPU、GPU 设备进行对比,实验表明异构融合处理器的性能介于 CPU 和 GPU 两者之间,但功耗更低.普通 CPU 处理器中存在功耗控制,用于使处理器具有低功耗开销和更高可靠性;Zhu 等人[42]对异构融合处理器的功耗控制机制进行了分析,首次指出异构融合处理器需要考虑两种及以上类型的设备,处理器对功耗控制更敏感.异构融合处理器多种设备紧密结合的设计会带来更多的内存竞争,也会影响到功耗限制. Da´vila 等人[70,71]研究了负载划分对功耗和性能等的影响,并指出两者间的相关性.

3.4 小结

性能分析是进一步优化程序的基础.对程序进行性能分析可探明程序在不同情况下的性能表现,分析异构融合处理器对于不同类型程序的性能瓶颈,如多设备混合运行是否能够获得性能提升等,从而进一步优化程序,提升程序在异构融合处理器上的性能.本节从单程序的性能分析、多程序的性能分析、以及功耗角度对异构融合处理器的运行进行了介绍.对于单程序的性能分析,当内存带宽利用情况不紧张时,可以考虑增加计算设备更充分利用系统资源;对于多程序混合运行,在目前操作系统层缺少相应优化设计,多程序混合运行效果不理想.在功耗方面,异构融合处理器整体功耗低于离散 GPU 功耗,但由于性能等方面限制,优势不明显.研究表明,目前的异构融合处理器仍不成熟,具有很大的发展空间.

4 异构融合处理器上的优化工作

在充分理解异构融合处理器的性能表现的基础上, 本节主要介绍针对异构融合处理器的优化工作.

4.1 体系结构优化

Barik 等人[15]首次考虑在异构融合处理器运行不规则负载时如何将负载从 CPU 端迁移到 GPU 端,实验结果表明大部分的程序都能获得性能受益,但这一研究主要使用了异构融合处理器的 GPU 设备,CPU 端存在资源浪费.Zhang 等人[6]以 GPU-CPU 异构融合处理器为例,详细介绍了异构融合处理器上不同设备混合运行的性能优化技巧,具体包括不同设备体系结构的相关优化,内存带宽的利用,以及不同设备运行时计算核心的重叠情况等. 其中体系结构的优化可以分为针对异构融合处理器的不同设备进行数据访问方式调整、并行粒度优化、使用局部存储的方式调整等[74]. 图 5 是局部存储优化和访存优化的程序示例.Streamcluster 来自于 Rodinia[27]基准测试程序集.图 5(a)表示的计算核心更适合 GPU 运行, 图 5(b)表示的计算核心更适合 CPU 运行.首先,GPU 存在局部存储,但 CPU 没有,因此对于使用局部存储的数组 coord_s,CPU 计算核心需要直接对全局内存中的相应数据进行操作,避免与局部存储相关的访问,如在图 5(b)中,没有出现关键字__local.其次,CPU 和 GPU 的最优访存方式不同,GPU 适合同一线程组内的多个线程同时访问相邻的数据,如图 5(a)中,线程组中的线程访问的是相邻的数据;而 CPU 适合各独立线程访问的数据是相邻的,这样缓存局部性更好.因此,在图 5(b)中,添加了 coord_d 所表示的矩阵的转置数组 coord_d_cpu.CPU 单线程对 coord_d_cpu 数组的访问连续,具有较好的局部性.通过这些优化技术,不同设备均可实现高效的内存数据访问.

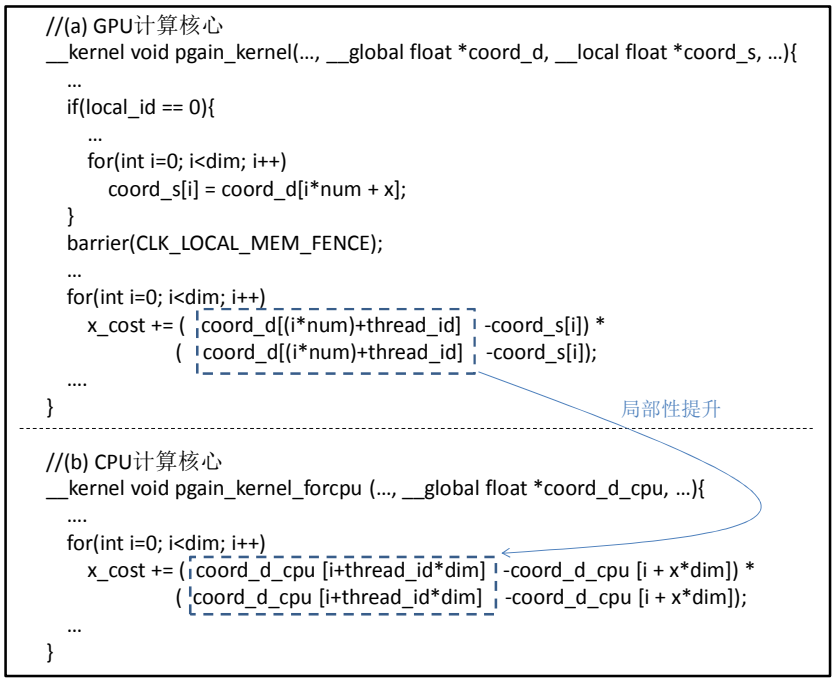


Fig.5 Optimizations of local memory and memory access[6]

图 5 局部存储优化和访存优化

图 6 是对并行度和循环优化的例子,代码来源于 Rodinia 程序集高斯消元程序,通过调整并行粒度和循环优化来增加局部性.图 6(a)表示的计算核心适合 GPU 运行, 图 6(b)表示的计算核心适合 CPU 运行.图 6(b)中的计算核心相比于图 6(a)中的计算核心降低了线程维度,减少了线程数量,且保证对于 a_dev 数组,CPU 中每个线程能够实现连续访问,具有较好的数据局部性,这有助于 CPU 缓存发挥作用.

对于内存带宽的因素, 由于多种设备访问同样的内存, 因此需要减少设备对内存带宽的影响. 例如, 尽可能把数据缓存在 GPU 的局部存储. 同时, CPU 和 GPU 设备运行计算核心的时间重叠情况也会影响混合运行的性

能,运行时间长的计算核心更容易重叠;对于运行时间短的计算核心,多设备混合运行获得的性能提升有限.许多相关研究是通过对上述因素进行优化,提升系统性能.

4.2 负载划分与调度优化

本节主要介绍针对异构融合处理器中不同设备的负载划分与调度优化工作.异构融合处理器存在多种设备,如何使多种设备能够混合运行一直是融合处理器的研究热点.Kallem 等人[11]研发了面向异构融合处理器的调度方案,该方案首先抽取一部分负载交由异构融合处理器上的 GPU 处理,CPU 继续处理后续负载.在 GPU 处理完成所分配的负载后,计算 CPU 和 GPU 所处理负载量的比例,进而获得 CPU 和 GPU 处理能力上的差异.之后,根据 CPU 和 GPU 处理数据能力的不同,对剩余负载进行重新划分.假设 GPU 所分得的负载比重为 ag(0≤ag≤1),则剩下 1-ag 的负载将分配到 CPU.在理想情况下,异构融合处理器上的不同设备能够同时结束对各自负载的处理,实现负载均衡.

```
//(a) GPU计算核心
__kernel void Fan2(…){
  int globalIdx = get_global_id(0);
  int globalIdy = get_global_id(1);
  if(globalIdx < size-1-t && globalIdy < size-t){
    a_dev[size*(globalIdx+1+t)+(globalIdy+t)] -=
      m_dev[size*(globalIdx+1+t)+t] * a_dev[size*t+(globalIdy+t)];
    …
  }
  …
}
-----------------------------------------------------------------------
//(b) CPU计算核心
__kernel void Fan2_forcpu(…){
  int globalIdx = get_global_id(0);
  int globalIdy = get_global_id(1);
  int globalIdysize = get_global_size(1);
  int globalIdystart = globalIdy/globalIdysize*size;
  int globalIdyend = (globalIdy+1)/globalIdysize*size;
  for(globalIdy = globalIdystart; globalIdy<globalIdyend; globalIdy++)
    if(globalIdx < size-1-t && globalIdy < size-t){
      a_dev[size*(globalIdx+1+t)+(globalIdy+t)] -=
        m_dev[size*(globalIdx+1+t)+t] * a_dev[size*t+(globalIdy+t)] ;
    …
  }
  …
}
```

循环优化

Fig.6 Optimizations of parallelism and loop[6]

图 6 并行度和循环优化

Zhang 等人[6]提出在异构融合处理器上基于建模的方法混合运行 CPU、GPU 设备.首先,使 CPU 处理一部分负载,记录时间,并预测 CPU 的处理速度;之后,用同样的方法预测 GPU 的处理速度.在获得 CPU、GPU 的测试性能后,对剩余负载依据 CPU 和 GPU 的性能对比进行划分.这一方法的缺点在于提升预测性能的准确性需要增加 CPU、GPU 处理负载的采样量,但这样真正用于 CPU、GPU 混合处理的负载量就减少了,并且对于不规则负载很难通过简单采样直接预测整体性能.同时,对于动态负载,Zhang 等人[72]进一步提出了实时细粒度负载划分算法,充分利用系统资源.

Pandit 等人[12]研究不需要采样就可以同时混合运行离散 GPU 和 CPU 的方法,这一方案也适用于异构融合

处理器.该方案对任务的分配是通过 workgroup(OpenCL 术语)在 CPU 和 GPU 端的分配完成的.CPU 和 GPU 端同时运行计算核心,首先,通过数据传输命令将数据部署到 CPU 和 GPU 设备上.在 GPU 端,开启所有的线程组,每组内设置一个线程接收 CPU 端的状态,CPU 端的运行划分为许多子计算核心.该方案对负载进行编号,根据编号使 GPU 从负载最前端向后处理,CPU 从后向前处理,当 CPU 处理完成每个子计算核心后就将数据和状态发送给 GPU.当 GPU 执行到某时刻发现将要处理的数据 CPU 已经处理完成时,GPU 停止计算.之后 GPU 进行数据整合,并将处理结果传回 CPU.Pandit 等人[12]在异构融合处理器上实现了这一方案,实验表明存在一定开销.Tang 等人[13]在异构融合处理器的资源分配方面设计了多种资源公平分配策略,这一策略引入了博弈论中的公平性法则,在异构融合处理器资源分配中同时考虑公平性和效率,并在两者间进行平衡.

并行环境中异步的任务可以描述为有向无环图,Puthoor 等人[60]针对异构系统架构研究这一算法,可以细粒度地进行任务管理.FinePar[14]是首个针对不规则负载的多设备混合运行细粒度负载划分框架,能够同时利用多种设备.以往针对异构融合处理器的划分算法没有考虑 GPU 和 CPU 的体系结构特点以及负载内部的不规则性,不能完全适应不规则负载.FinePar 使用离线训练的方式建立性能模型,能够依据给定负载的不规则程度模拟出 CPU 和 GPU 的性能.之后,对于用户提供的程序和不规则负载,能够自动修改代码,把负载中相对规则的部分分配给 GPU,剩下部分分配给 CPU.因为 GPU 对负载的不规则性较为敏感,这种方法能够最大化 GPU 性能,进而提升异构融合处理器的整体性能.Cho 等人[50]进一步实现了实时的不规则负载划分策略.

4.3 功耗优化方法

在高性能计算领域,功耗是评价计算机系统的一项重要指标. AMD 和 Intel 等硬件厂商将异构融合处理器作为平衡性能和能耗的设计,可以提供更好的性能功耗比[39,40]. Gu 等人[41]在异构融合处理器上对多层感知神经网络进行加速,并从性能功耗比的角度展示异构融合处理器的优势. Zhu 等人[42]研究在异构融合处理器上如何混合运行多个程序使处理器运行的能耗较低.这项研究从能耗的角度进行了分析,并提出了相应混合运行调度策略.首先,通过不同频率下的模型判断程序混合运行是否可以从混合运行中受益,记录受益最大的结果;之后,将混合运行结果和最高频率下的单设备运行结果进行比较,并选择最优策略.Garzón 等人[43]针对迭代计算提出了异构融合处理器上的功耗优化方案 E-ADITHE,该方案包含启发式的方法,基于功耗预测选择合适的处理设备,同时也会考虑异构处理器内部的负载均衡特性.

4.4 小结

异构融合处理器将多种设备集成到一个芯片上,目前优化工作主要依靠编程人员实现.本节对异构融合处理器上的优化工作进行了总结. 在内存带宽允许的情况下,可进行多设备混合运行充分利用系统资源;此外,当应用可拆解为不同部分时,可考虑各部分在不同设备的性能表现,将各部分运行在在异构融合处理器上最适合的设备.在程序优化的过程中,由于不同设备间的体系结构差异巨大,需要对不同设备的代码分别进行优化;同时,异构融合处理器也具有以往离散架构所没有的特性,如共享内存、多设备异步执行等,因此在优化的过程中还需考虑这些新特性. 接下来将对异构融合处理器在其他领域的具体应用进行介绍.

5 异构融合处理器的应用

多设备融合设计和以往离散的设计相比具有更多的优化机会,目前已被应用于越来越广泛的应用之中.本节主要对异构融合处理器上的具体应用场景进行介绍.

5.1 数据密集型应用

异构融合处理器可以在数据库等领域进行性能加速,目前有学者研究异构融合处理器在大数据管理中的使用.Hetherington 等人[37]评估了键值存储在异构融合处理器上的性能表现,和使用离散的 GPU、CPU 架构相比,异构融合处理器能够带来较大的性能提升.针对在线分析处理,He 等人[38]提出了在异构融合处理器上基于缓存的查询算法,使 CPU 辅助 GPU 缓存数据,提升 GPU 查询的缓存命中率.在大数据管理领域,He 等人[18]首先提出了利用异构融合处理器完成数据库查询操作中的哈希合并(Hash Join)操作.由于合并操作可以分为建立、探测等步骤,CPU 和 GPU 设备在不同步骤中有不一样的性能表现.这一研究在各步骤中考虑异构融合处理器中

的 CPU、GPU 特性,合理分配负载混合处理.DIDO[19]是异构融合处理器上的内存键值存储系统,系统在 GPU、CPU 设备间实现了 pipeline 动态执行模型,能够自适应地将不同负载动态地在运行时分配给 GPU 和 CPU 设备.Daga 等人[29]在异构融合处理器上研究 B+树的实现.B+树是数据库领域中索引的基础操作,这一工作针对其访问的不规则特性和 CPU、GPU 间的数据传输进行优化.FineStream[73]是首个利用异构融合处理器实现的 SQL 流处理系统,查询语句可分解为多个算子,不同算子对异构融合处理器的设备有不同的性能偏好.FineStream 建模将不同的算子分配在异构融合处理器的不同设备上,并考虑了共享内存带宽等特性,实现了细粒度地流处理多设备混合运行.

MapReduce[16]是一个用来处理大规模数据集的并行编程模型,将对数据的处理抽象成 Map(映射)和 Reduce(归纳)两个阶段.用户只需要按照给定的 Map 和 Reduce 接口编写程序,MapReduce 编程模型就会自动对数据进行划分和调度,并行处理数据,用户不需要对其进行控制.这样的编程方式极大地降低了用户的编程难度.异构融合处理器将 CPU 与多种设备相集成,不同设备体系结构差异较大,需要有针对性地进行细粒度复杂优化以发挥不同设备的潜在性能,这无形中增加了异构融合处理器的使用门槛.Chen 等人[17]首先在异构融合处理器上实现了 MapReduce 框架,能够对用户屏蔽异构融合处理器的底层细节,用户只需按照给定的接口编写程序,系统就能自动在不同设备上分配、处理负载.在该框架的划分策略中,调度器将输入任务分配到各 worker,所有的 worker(MapReduce 术语)都执行同样的步骤.每个 worker 都是一个 workgroup(OpenCL 术语),由一组线程构成,可存在于主机端也可存在于异构设备端,分配到 worker 的任务会均匀分配到其中的所有线程一起处理.MapReduce 是迄今应用最广泛的数据并行编程框架之一,尽管 MapReduce 框架存在扩展性和容错方面的实现开销,仍有大量关于提升 MapReduce 计算性能和降低功耗的研究,这些研究涉及异构器件通常会使用离散 GPU. 功耗是大规模集群计算考虑的重要指标,由于异构融合处理器具有低功耗的特性,Kim 等人[44]首先从功耗角度研究在由异构融合处理器搭建的集群上运行 MapReduce 框架. 这一研究使用 Intel 异构融合处理器,通过修改 Hadoop 的通用机器学习库 Apache Mahout[45]使其支持运行. 通过进一步优化异构融合处理器的新特性,如对共享内存、缓存等方面进行优化,证明了异构融合处理器在集群计算功耗方面的巨大优势.

5.2 计算密集型应用

研究者将异构融合处理器用于科学计算领域,结合具体应用进行优化.本节选择其中代表性的应用进行介绍.Doerksen 等人[28]研究在异构融合处理器上如何进一步提升科学计算程序性能,并以高斯消元程序和 0-1 背包问题求解程序为例进行研究.该研究工作将 GPU 用于计算,而 CPU 只用于检查结束条件和同步等操作.Delorme 等人[30]在异构融合处理器上实现了并行快速排序算法,将算法分为了局部排序、局部分析、排名、分发等四个阶段,在各阶段分别考虑 CPU 和 GPU 处理负载的不同部分,共同执行程序.Eberhart 等人[31]研究异构融合处理器上的 Stencil 计算.Stencil 计算负载大部分较为规则,适合 GPU 并行处理,但仍存在一些边缘的不规则部分.这一研究利用 CPU、GPU 共享内存的特点,令 CPU 负责处理 Stencil 中的边缘部分,GPU 处理规则部分,使程序达到了较高性能.Daga 等人[32]研究广度优先搜索算法在异构融合处理器上的实现.广度优先搜索算法常见的遍历包括自上而下的遍历和自下而上的遍历两种,其中自上而下的遍历算法适合 CPU,自下而上的算法适合 GPU.这一工作利用共享内存的特征在算法执行的过程中选择合适算法,并在不同设备间进行切换.Zhang 等人[59]进一步提出了异构融合处理器上不同策略的自适应模型,能够有进一步的性能提升.Ilgner 等人[33]设计了异构融合处理器上的有限差分时域算法,能够利用异构融合处理器的共享内存特性对算法进行加速.Liu 等人设计了异构融合处理器上可以高效执行的稀疏矩阵-稀疏矩阵乘算法[34,61]、稀疏矩阵向量乘算法[35],并针对异构融合处理器的特性设计了堆数据结构[36].Zou 等人[65]在异构融合处理器上实现了 Smith-Waterman 算法,为了充分利用各设备,设计了动态负载划分策略.Freytag 等人[66]在异构融合处理器上研究非均匀域分解,利用共享内存特性避免了设备间数据传输开销.

5.3 其他领域

目前有研究工作使用 GPU 等异构设备加速网络应用中的数据包处理,对于其中一些计算密集型的算

法,GPU 比 CPU 处理能力强;然而,由于 CPU 和离散 GPU 间的 PCIe 传输,GPU 的加速效果大打折扣.为了避免不同设备间因数据传输所带来的时间开销,Go 等人[51]设计了利用异构融合处理器进行加速的网络数据包处理器 APUNet,能够对诸多网络应用实现高效网络数据包处理.APUNet 在所有阶段均可避免 CPU 和 GPU 间的数据传输,更好地利用带宽,释放 GPU 性能;为了完成 CPU 和 GPU 间的低延迟通信,APUNet 使多个 GPU 线程并行处理输入数据包流. Chang 等人[68]基于异构融合处理器共享内存和低功耗等特性,在异构融合处理器上探究了网络数据包分类应用.

对于机器学习领域,Gu 等人[41]探索了在异构融合处理器上实现了深度神经网络模型,并和离散 GPU 进行了比较.实验表明,异构融合处理器具有更高的性能功耗比,在未来可搭建低功耗的深度神经网络计算平台.此外,在视频处理领域,Zhu 等人[69]在异构融合处理器上实现了高效帧频转换算法.

5.4 小结

尽管异构融合处理器目前处于起步阶段,在性能方面和高端 GPU 等加速器存在差距,但异构融合处理器的出现使不同领域的研究人员看到了新的机会,同时也被用在越来越多的领域中.例如,在网络领域,处理延迟至关重要,而异构融合处理器中的 GPU 等设备能够直接高效对内存数据进行操作,相对离散 GPU 具有一定优势.这些研究工作表明异构融合处理器具有强劲的应用潜力,在未来会有更多的展现机会.

6 异构融合处理器研究未来发展展望

结合异构融合处理器不断演变的发展历程,未来的异构融合处理器会集成更多的计算核心,性能也会有进一步提升,甚至可以应用于高性能服务器中.本节从新型体系结构研究、数据密集型研究、计算密集型研究等方面展望异构融合处理器的未来发展.

6.1 新型体系结构研究

高性能计算指利用超级计算和并行处理技术解决复杂的计算问题.高性能计算技术通常设计并行处理算法和系统,利用计算机仿真、计算机模拟和分析解决科学计算问题,应用的领域包括生命科学、地理信息数据处理、石油勘探模拟、电力系统设计、气象模拟等方面.高性能计算涉及到的这些问题关乎人们的日常生活,然而,解决更大、更复杂的问题也需要更强大的计算能力.特别是在大数据时代,需要处理的数据量过于庞大,设计新型的 E 级超级计算机(exaflops)来缓解新的技术需求变得越来越迫切.但由于计算机扩展性方面的限制,简单将计算设备进行堆叠受限于功耗、内存和网络带宽,来自于数据传输等方面的开销过大,难以达到所需要的计算能力.

异构融合处理器具有将传统处理器与加速设备相融合的特点,可以从多方面满足设计 E 级超级计算机的需求.第一,将高吞吐率设备如 GPU 等和 CPU 处理器相结合的设计可以兼顾不同类型的计算任务需求,提高高性能计算机处理器的计算适应性;第二,异构融合处理器中的不同设备集成在一起可以共享相同的内存和存储设备,紧密结合的设计可以避免加速设备和处理器分离所带来的数据传输开销,扩展性更好;第三,不同类型处理器集成到一个芯片的设计可以采用更高效的功耗控制技术,同时也可拥有更多的组件优化设计,降低功耗和成本;第四,异构融合处理器可设计多级存储体系结构,分层的存储设计可更充分地发挥异构融合处理器中不同设备的计算特性.目前的异构融合处理器仍处于发展阶段,性能较弱.但如表 1 所示,有越来越多的研究者利用异构融合处理器解决科学计算相关的一些列计算任务,和相同价位的 GPU 等异构计算设备相比,异构融合处理器表现出了较强的计算能力和性能功耗比.同时,尽管异构融合处理带来了诸多好处,集成芯片设计复杂,同时各设备类型不同,如果没有对不同的处理器有针对性地进行优化则可能出现较大的性能降级.如何设计高效的不同设备间的缓存一致性也是一个所面临的挑战.未来的高性能异构融合处理器设计需要考虑这些问题,设计出性能更强的新型处理器.

编程语言是一种形式化语言,包含了一组用于产生各种输出的计算机指令集.随着计算机体系结构的不断发展,相应的编程语言也在不断发展、进化.从最原始的机器语言,发展至现代的编程语言,从可编程性、程序开发效率、维护效率等方面无不发生着巨大的变化,异构融合处理器的出现进一步加剧了能够适应新型处理设备

的编程语言的需求.目前的研究表明,需要针对不同设备进行程序优化才能使异构融合处理器达到较优的性能,需要在编写程序时充分了解体系结构特性,对编程人员要求较高.未来的异构融合处理器可能会集成更多种类的设备,包含多层异构存储体系结构,并提供动态能耗弹性管理特性,编程所需考虑的情况会更多样化,这些考虑因素可能会令程序员无法专注于应用本身,无法高效地开发程序.设计可编程性高、性能强的编程语言需要对硬件特性和编程模型进行充分探索,对底层硬件从软件开发人员角度进行抽象,运行时能够利用异构融合处理器的各种特性.

目前异构融合处理器最具代表性的编程语言是 OpenCL,被各大处理器生产厂商所支持.然而,对于类似 GPU 等处理单元,需要编程人员在代码中显式写明各线程如何并行处理数据、如何利用共享缓存以充分利用处理器资源,而对于 CPU 处理器则不需要考虑这些操作.未来针对异构融合处理器的编程语言,需要降低不同处理单元间的差异性以降低编程要求,但同时应保证程序编译后仍具有较高的性能,这要求编译器能够自动针对不同设备进行优化.例如,对于一个循环操作,编译器和运行时库能够自动为 CPU 和异构单元(如 GPU)分配合适的循环并执行相关优化操作.与此同时,由于开发通用目的编程语言(general-purpose language)充满挑战,未来会出现越来越多的针对特定领域的编程语言(domain specific langulage),能够针对某些领域提供特定程序接口,高效利用异构融合处理器,满足不同领域编程人员的开发需求.此外,未来的编程语言设计还需配套相关编程辅助工具.具体包括:(1)简单、易于使用的代码调试工具,能够对运行在异构融合处理器上的程序进行代码调试;(2)针对异构融合处理器的代码分析工具,能够在可接受的时间范围内分析出程序的性能瓶颈位置以及计算、访存情况等信息;(3)自动调优工具,能够辅助编程人员针对异构融合处理器优化数据存放位置、数据访问方式、计算模式等.

6.2 数据密集型应用研究

以大数据为代表的数据密集型应用未来会在异构融合处理器的研究中扮演重要的角色.大数据指无法在一定时间内通过常规数据处理工具进行处理的大规模数据集合,因为数据量过大,传统的数据处理技术无法直接进行处理[63].大数据所带来的挑战不仅是数据存储,还包括分析、搜索、传输、查询、更新、可视化等一系列挑战.大数据处理技术受益于计算机体系结构的发展,各类计算处理单元为大数据系统提供底层支持,在新型计算设备之上设计合适的数据结构和算法可以搭建出性能更强的大数据处理系统.近些年,随着以 GPU 为代表的异构计算的兴起,越来越多的大数据处理系统利用异构计算技术应对挑战.相对于 CPU,GPU 拥有更多的计算核心,即更高的并行度.然而,使用异构加速器件也存在着一些问题和挑战,例如,利用 GPU 等异构设备需要首先从 CPU 端传输数据存在时间开销,其次,异构设备通常使用独立的内存,容量有限,在大数据环境下难以一次性将全部数据装入设备内存,再次,对于图数据库等新型数据密集型应用[64],由于应用数据访问的不规则性,GPU 等异构设备无法充分发挥其内在性能.而未来大数据处理对异构融合处理器的使用则可以解决这些问题,多种异构设备集成到一个芯片集成了不同设备的特性,并可以共享相同的物理内存,避免了不同设备间的数据传输开销,可以更高效地对大数据进行处理.

大数据处理等数据密集型应用使用异构融合处理器高效处理数据,需要面对一系列新的挑战.第一,异构融合处理器集成了多种设备,对于大数据计算任务如何向不同的设备进行任务分配,是否需要在大数据处理系统内部设计不同处理器的调度策略.第二,多种设备相集成的特性使得不同设备共享有限的内存带宽,对于数据密集型应用,如何设计数据处理策略使不同设备的数据处理过程不互相干扰也是一个亟待解决的挑战.第三,针对异构融合处理器的数据结构和大数据处理算法,充分发挥新型硬件的计算能力,以往适合传统处理器的数据结构和并行算法可能不适用于多设备融合的特点,因此需要考虑不同设备差异进行设计.第四,大数据处理系统的操作人员可能不是计算机从业者,有可能是统计等领域的从业人员,如何保证异构融合处理器在大数据处理技术中的易用性也是一个需要面对的挑战.

6.3 计算密集型应用研究

传统的高性能计算领域中计算密集型应用可受益于异构融合处理器的发展,此外,在计算机领域,异构融合

处理器有可能成为其中机器学习任务中计算密集型应用的新突破口.人工智能利用计算机程序实现人类智能技术,而机器学习作为实现人工智能的一种方法,大量使用 GPU 等异构加速设备.特别是对于深度学习中的深度神经网络训练,GPU 等高并发加速设备使其能够在较短时间内完成训练.然而,目前机器学习领域对异构设备的使用仍处于探索阶段,存在诸多挑战.首先,为了进一步提高训练模型的准确性,灵活应对复杂的机器学习任务和应用场景,需要用高维模型海量数据进行训练,而目前流行的 GPU 等异构设备中有限的内存无法一次性装入大量数据,需要数据和参数的传入、传出,这会带来开销.其次,GPU 等加速设备往往对于规则的密集型计算任务较为适合,而大规模机器学习中有可能遇到非规则计算型任务,例如,有可能会遇到稀疏特征等问题,GPU 等加速效果有限,需要进一步采取优化操作.再次,当涉及多个设备混合运算时,每个设备都有独立的内存架构,当需要跨设备进行数据访问时开销较大,且当数据同时存在不同设备时较难维护数据的一致性.而异构融合处理器将不同加速设备相融合的特性为机器学习等人工智能领域应用加速带来了新的机遇.

机器学习应用与异构融合处理器相结合,有如下考虑因素.第一,对于同样的机器学习应用,可能存在多种机器学习训练模型结合不同的算法,且模型间计算模式、数据依赖、访存模式等均不相同,因此需要对机器学习模型和不同计算设备体系结构具有充分了解才能开发出针对异构融合处理器合适的模型.第二,面对大规模机器学习任务,如何设计合理的数据结构充分考虑设备间体系结构差异,提供高维模型海量数据的训练能力.第三,针对异构融合处理器机器学习系统的易用性和高效性等考虑因素,需要为用户提供统一的编程接口,同时机器学习系统内部能够充分考虑设备间的结构差异以及异构融合处理器的特性.总之,机器学习技术是目前学术界和工业界的研究热点,对于异构融合处理器生产厂商也在不断尝试如何设计体系结构能够更贴合机器学习应用特性,相信在不远的将来,可以看到异构融合处理器与机器学习技术相结合的新突破.

7 总结

异构融合处理器将不同的设备集成到一个芯片为科学计算、大数据处理等领域带来了新的研究机会.但由于其共享内存、编程时需要考虑不同设备体系结构差异等特点,异构融合处理器的研究与发展也存在不小的挑战.本文从异构融合处理器的性能分析、优化、以及具体应用等角度对以往研究工作进行了分类与总结,使用异构融合处理器编程需要充分考虑不同设备硬件特性、异构融合处理器特有特征、以及负载特点;在程序优化方面,现有研究从性能和功耗角度对异构融合处理器进行了探索,考虑了不同设备的特点与优化方式.同时,异构融合处理器正被用于越来越广泛领域,这可为进一步的处理器设计提供参考.最后,本文从高性能计算、编程模型、大数据处理、机器学习这四方面对异构融合处理器的未来发展趋势进行了展望.

References:

[1] Foley D, Steinman M, Branover A, Smaus G, Asaro A, Punyamurtula S, Bajic L. AMD's 'Llano' Fusion APU. InHot Chips 2011 Aug 17 (Vol. 23, pp. 1-38).

[2] Intel, The Compute Architecture of Intel Processor Graphics Gen7.5 [OL]. [2017-12-01] https://software.intel.com/sites/default/files/managed.

[3] Nikolskiy, Vsevolod P., Vladimir V. Stegailov, and Vyacheslav S. Vecher. "Efficiency of the Tegra K1 and X1 systems-on-chip for classical molecular dynamics." High Performance Computing & Simulation (HPCS), 2016 International Conference on. IEEE, 2016.

[4] Colangelo P, Luebbers E, Huang R, et al. Application of convolutional neural networks on Intel Xeon processor with integrated FPGA[C]// 2017 IEEE High Performance Extreme Computing Conference (HPEC). IEEE, 2017: 1-7.

[5] Lee K, Lin H, Feng WC. Performance characterization of data-intensive kernels on AMD fusion architectures. Computer Science-Research and Development. 2013 May 1;28(2-3):175-84.

[6] Zhang F, Zhai J, He B, Zhang S, Chen W. Understanding co-running behaviors on integrated CPU/GPU architectures. IEEE Transactions on Parallel and Distributed Systems. 2017 Mar 1;28(3):905-18.

[7] Stone JE, Gohara D, Shi G. OpenCL: A parallel programming standard for heterogeneous computing systems. Computing in science & engineering. 2010 May;12(3):66-73.

[8] Sanders J, Kandrot E. CUDA by example: an introduction to general-purpose GPU programming. Addison-Wesley Professional; 2010 Jul 29.

[9] Dagum L, Menon R. OpenMP: an industry standard API for shared-memory programming. IEEE computational science and engineering. 1998 Jan;5(1):46-55.

[10] Nichols B, Buttlar D, Farrell J. Pthreads programming: A POSIX standard for better multiprocessing. " O'Reilly Media, Inc."; 1996.

[11] Kaleem R, Barik R, Shpeisman T, Lewis BT, Hu C, Pingali K. Adaptive heterogeneous scheduling for integrated GPUs. In Proceedings of the 23rd international conference on Parallel architectures and compilation 2014 Aug 24 (pp. 151-162). ACM.

[12] Pandit P, Govindarajan R. Fluidic kernels: Cooperative execution of OpenCL programs on multiple heterogeneous devices. In Proceedings of Annual IEEE/ACM International Symposium on Code Generation and Optimization 2014 Feb 15 (p. 273). ACM.

[13] Tang S, He B, Zhang S, Niu Z. Elastic multi-resource fairness: balancing fairness and efficiency in coupled CPU-GPU architectures. In Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis 2016 Nov 13 (p. 75). IEEE Press.

[14] Zhang F, Wu B, Zhai J, He B, Chen W. FinePar: Irregularity-aware fine-grained workload partitioning on integrated architectures. In Code Generation and Optimization (CGO), 2017 IEEE/ACM International Symposium on 2017 Feb 4 (pp. 27-38). IEEE.

[15] Barik R, Kaleem R, Majeti D, Lewis BT, Shpeisman T, Hu C, Ni Y, Adl-Tabatabai AR. Efficient mapping of irregular C++ applications to integrated GPUs. In Proceedings of Annual IEEE/ACM International Symposium on Code Generation and Optimization 2014 Feb 15 (p. 33). ACM.

[16] Dean J, Ghemawat S. MapReduce: simplified data processing on large clusters. Communications of the ACM. 2008 Jan 1;51(1):107-13.

[17] Chen L, Huo X, Agrawal G. Accelerating MapReduce on a coupled CPU-GPU architecture. In Proceedings of the International Conference on High Performance Computing, Networking, Storage and Analysis 2012 Nov 10 (p. 25). IEEE Computer Society Press.

[18] He J, Lu M, He B. Revisiting co-processing for hash joins on the coupled CPU-GPU architecture. Proceedings of the VLDB Endowment. 2013 Aug 26;6(10):889-900.

[19] Zhang K, Hu J, He B, Hua B. Dido: Dynamic pipelines for in-memory key-value stores on coupled CPU-GPU architectures. In Data Engineering (ICDE), 2017 IEEE 33rd International Conference on 2017 Apr 19 (pp. 671-682). IEEE.

[20] Zakharenko V, Aamodt T, Moshovos A. Characterizing the performance benefits of fused CPU/GPU systems using FusionSim. InDesign, Automation & Test in Europe Conference & Exhibition (DATE), 2013 2013 Mar 18 (pp. 685-688). IEEE.

[21] Zhang F, Zhai J, Chen W, He B, Zhang S. To co-run, or not to co-run: A performance study on integrated architectures. In Modeling, Analysis and Simulation of Computer and Telecommunication Systems (MASCOTS), 2015 IEEE 23rd International Symposium on 2015 Oct 5 (pp. 89-92). IEEE.

[22] Spafford KL, Meredith JS, Lee S, Li D, Roth PC, Vetter JS. The tradeoffs of fused memory hierarchies in heterogeneous computing architectures. InProceedings of the 9th conference on Computing Frontiers 2012 May 15 (pp. 103-112). ACM.

[23] Daga M, Aji AM, Feng WC. On the efficacy of a fused CPU+ GPU processor (or APU) for parallel computing. In Application Accelerators in High-Performance Computing (SAAHPC), 2011 Symposium on 2011 Jul 19 (pp. 141-149). IEEE.

[24] Zhu Q, Wu B, Shen X, Shen L, Wang Z. Understanding co-run degradations on integrated heterogeneous processors. In International Workshop on Languages and Compilers for Parallel Computing 2014 Sep 15 (pp. 82-97). Springer, Cham.

[25] Zhu Q, Wu B, Shen X, Shen K, Shen L, Wang Z. Understanding co-run performance on CPU-GPU integrated processors: observations, insights, directions. Frontiers of Computer Science. 2017 Feb 1;11(1):130-46.

[26] Mekkat V, Holey A, Yew PC, Zhai A. Managing shared last-level cache in a heterogeneous multicore processor. In Proceedings of the 22nd international conference on Parallel architectures and compilation techniques 2013 Oct 7 (pp. 225-234). IEEE Press.

[27] Che S, Boyer M, Meng J, Tarjan D, Sheaffer JW, Lee SH, Skadron K. Rodinia: A benchmark suite for heterogeneous computing. In Workload Characterization, 2009. IISWC 2009. IEEE International Symposium on 2009 Oct 4 (pp. 44-54). IEEE.

[28] Doerksen M, Solomon S, Thulasiraman P. Designing APU oriented scientific computing applications in OpenCL. In High Performance Computing and Communications (HPCC), 2011 IEEE 13th International Conference on 2011 Sep 2 (pp. 587-592). IEEE.

[29] Daga M, Nutter M. Exploiting coarse-grained parallelism in B+ tree searches on an APU. In High Performance Computing, Networking, Storage and Analysis (SCC), 2012 SC Companion: 2012 Nov 10 (pp. 240-247). IEEE.

[30] Delorme MC, Abdelrahman TS, Zhao C. Parallel radix sort on the AMD fusion accelerated processing unit. In Parallel Processing (ICPP), 2013 42nd International Conference on 2013 Oct 1 (pp. 339-348). IEEE.

[31] Eberhart P, Said I, Fortin P, Calandra H. Hybrid strategy for stencil computations on the APU. In Proceedings of the 1st international workshop on high-performance stencil computations, Vienna 2014 Jan 21 (pp. 43-49).

[32] Daga M, Nutter M, Meswani M. Efficient breadth-first search on a heterogeneous processor. In Big Data (Big Data), 2014 IEEE International Conference on 2014 Oct 27 (pp. 373-382). IEEE.

[33] Ilgner RG, Davidson DB. A comparison of the FDTD algorithm implemented on an integrated GPU versus a GPU configured as a co-processor. In Electromagnetics in Advanced Applications (ICEAA), 2012 International Conference on 2012 Sep 2 (pp. 1046-1049). IEEE.

[34] Liu W, Vinter B. An efficient GPU general sparse matrix-matrix multiplication for irregular data. In Parallel and Distributed Processing Symposium, 2014 IEEE 28th International 2014 May 19 (pp. 370-381). IEEE.

[35] Liu W, Vinter B. Speculative segmented sum for sparse matrix-vector multiplication on heterogeneous processors. Parallel Computing. 2015 Nov 1;49:179-93.

[36] Liu W, Vinter B. Ad-heap: An efficient heap data structure for asymmetric multicore processors. In Proceedings of Workshop on General Purpose Processing Using GPUs 2014 Mar 1 (p. 54). ACM.

[37] Hetherington TH, Rogers TG, Hsu L, O'Connor M, Aamodt TM. Characterizing and evaluating a key-value store application on heterogeneous CPU-GPU systems. In Performance Analysis of Systems and Software (ISPASS), 2012 IEEE International Symposium on 2012 Apr 1 (pp. 88-98). IEEE.

[38] He J, Zhang S, He B. In-cache query co-processing on coupled CPU-GPU architectures. Proceedings of the VLDB Endowment. 2014 Dec 1;8(4):329-40.

[39] Bouvier D, Sander B. Applying AMDs Kaveri APU for heterogeneous computing. In Hot Chips: A Symposium on High Performance Chips (HC26) 2014 Aug 10.

[40] Intel Graphics Technology, [OL]. [2020] https://en.wikipedia.org/wiki/Intel_Graphics_Technology.

[41] Gu J, Zhu M, Zhou Z, Zhang F, Lin Z, Zhang Q, Breternitz M. Implementation and evaluation of deep neural networks (DNN) on mainstream heterogeneous systems. In Proceedings of 5th Asia-Pacific Workshop on Systems 2014 Jun 25 (p. 12). ACM.

[42] Zhu Q, Wu B, Shen X, Shen L, Wang Z. Co-Run Scheduling with Power Cap on Integrated CPU-GPU Systems. In Parallel and Distributed Processing Symposium (IPDPS), 2017 IEEE International 2017 May 29 (pp. 967-977). IEEE.

[43] Garzón EM, Moreno JJ, Martínez JA. An approach to optimise the energy efficiency of iterative computation on integrated GPU–CPU systems. The Journal of Supercomputing. 2017 Jan 1;73(1):114-25.

[44] Kim S, Bottleson J, Jin J, Bindu P, Sakhare SC, Spisak JS. Power efficient MapReduce workload acceleration using integrated-GPU. In Big Data Computing Service and Applications (Big Data Service), 2015 IEEE First International Conference on 2015 Mar 30 (pp. 162-169). IEEE.

[45] The Apache Mahout Project, Apache Mahout, [OL]. [2012] http://mahout.apache.org/.

[46] Power J, Basu A, Gu J, Puthoor S, Beckmann BM, Hill MD, Reinhardt SK, Wood DA. Heterogeneous system coherence for integrated CPU-GPU systems. In Proceedings of the 46th Annual IEEE/ACM International Symposium on Microarchitecture 2013 Dec 7 (pp. 457-467). ACM.

[47] Yang Y, Xiang P, Mantor M, Zhou H. CPU-assisted GPGPU on fused CPU-GPU architectures. InHigh Performance Computer Architecture (HPCA), 2012 IEEE 18th International Symposium on 2012 Feb 25 (pp. 1-12). IEEE.

[48] Choi YK, Cong J, Fang Z, Hao Y, Reinman G, Wei P. A quantitative analysis on microarchitectures of modern CPU-FPGA platforms. In Proceedings of the 53rd Annual Design Automation Conference 2016 Jun 5 (p. 109). ACM.

[49] Cong J, Fang Z, Huang M, Wang L, Wu D. CPU-FPGA Coscheduling for Big Data Applications. IEEE Design & Test. 2018 Feb;35(1):16-22.

[50] Cho, Y., Negele, F., Park, S., Egger, B., & Gross, T. R. (2018, November). On-the-fly workload partitioning for integrated CPU/GPU architectures. In Proceedings of the 27th International Conference on Parallel Architectures and Compilation Techniques (p. 21). ACM.

[51] Go, Y., Jamshed, M. A., Moon, Y., Hwang, C., & Park, K. (2017, March). APUNet: Revitalizing GPU as Packet Processing Accelerator. In NSDI (pp. 83-96).

[52] Vijayaraghavany, T., Eckert, Y., Loh, G. H., Schulte, M. J., Ignatowski, M., Beckmann, B. M., ... & Kayiran, O. (2017, February). Design and Analysis of an APU for Exascale Computing. In High Performance Computer Architecture (HPCA), 2017 IEEE International Symposium on (pp. 85-96). IEEE.

[53] Schulte, M. J., Ignatowski, M., Loh, G. H., Beckmann, B. M., Brantley, W. C., Gurumurthi, S., ... & Rodgers, G. (2015). Achieving exascale capabilities through heterogeneous computing. IEEE Micro, 35(4), 26-36.

[54] Agarwal, N., Nellans, D., Ebrahimi, E., Wenisch, T. F., Danskin, J., & Keckler, S. W. (2016, March). Selective GPU caches to eliminate CPU-GPU HW cache coherence. In 2016 IEEE International Symposium on High Performance Computer Architecture (HPCA) (pp. 494-506). IEEE.

[55] Dashti, M., & Fedorova, A. (2017, June). Analyzing memory management methods on integrated CPU-GPU systems. In ACM SIGPLAN Notices (Vol. 52, No. 9, pp. 59-69). ACM.

[56] Krishnan, G., Bouvier, D., & Naffziger, S. (2016). Energy-Efficient Graphics and Multimedia in 28-nm Carrizo Accelerated Processing Unit. IEEE Micro, 36(2), 22-33.

[57] Doweck, J., Kao, W. F., Lu, A. K. Y., Mandelblat, J., Rahatekar, A., Rappoport, L., ... & Yoaz, A. (2017). Inside 6th-generation Intel Core: new microarchitecture code-named Skylake. IEEE Micro, (2), 52-62.

[58] Boggs, D., Brown, G., Tuck, N., & Venkatraman, K. S. (2015). Denver: Nvidia's first 64-bit ARM processor. IEEE Micro, 35(2), 46-55.

[59] Zhang, F., Lin, H., Zhai, J., Cheng, J., Xiang, D., Li, J., ... & Du, X. (2018). An adaptive breadth-first search algorithm on integrated architectures. The Journal of Supercomputing, 74(11), 6135-6155.

[60] Puthoor, S., Aji, A. M., Che, S., Daga, M., Wu, W., Beckmann, B. M., & Rodgers, G. (2016, March). Implementing directed acyclic graphs with the heterogeneous system architecture. In Proceedings of the 9th Annual Workshop on General Purpose Processing using Graphics Processing Unit (pp. 53-62). ACM.

[61] Liu, W., & Vinter, B. (2015). A framework for general sparse matrix–matrix multiplication on GPUs and heterogeneous processors. Journal of Parallel and Distributed Computing, 85, 47-61.

[62] Said, I., Fortin, P., Lamotte, J. L., & Calandra, H. (2017). Leveraging the accelerated processing units for seismic imaging: A performance and power efficiency comparison against CPUs and GPUs. The International Journal of High Performance Computing Applications, 1094342017696562.

[63] Pan W, Li Z, Zhang Y, et al. The New Hardware Development Trend and the Challenges in Data Management and Analysis[J]. Data Science and Engineering, 2018, 3(3): 263-276.

[64] Lin X, Yu J X. Special Issue on Graph Processing: Techniques and Applications[J]. Data Science and Engineering, 2017, 2(1): 1-1.

[65] Zou H, Tang S, Yu C, et al. Asw: accelerating Smith–Waterman algorithm on coupled CPU–GPU architecture[J]. International Journal of Parallel Programming, 2019, 47(3): 388-402.

[66] Freytag G, Navaux P O A, Lima J V F, et al. Non-uniform domain decomposition for heterogeneous accelerated processing units[C]//International Conference on Vector and Parallel Processing. Springer, Cham, 2018: 105-118.

[67] Zhang F, Liu W, Feng N, et al. Performance evaluation and analysis of sparse matrix and graph kernels on heterogeneous processors[J]. CCF Transactions on High Performance Computing, 2019, 1(2): 131-143.

[68] Chang Y K, Chi T Y. Hash-based OpenFlow Packet Classification on Heterogeneous System Architecture[C]//2019 Eleventh International Conference on Ubiquitous and Future Networks (ICUFN). IEEE, 2019: 300-305.

[69] Zhu H, Wang D, Zhang P, et al. Parallel implementations of frame rate up-conversion algorithm using OpenCL on heterogeneous computing devices[J]. Multimedia Tools and Applications, 2019, 78(7): 9311-9334.

[70] Dávila G P, Oliveira D, Navaux P, et al. Impact of Workload Distribution on Energy Consumption, Performance, and Reliability of Heterogeneous Devices[C]//2019 27th Euromicro International Conference on Parallel, Distributed and Network-Based Processing (PDP). IEEE, 2019: 166-173.

[71] Dávila G P. A performance, energy consumption and reliability evaluation of workload distribution on heterogeneous devices[J]. 2019.

[72] Zhang F, Zhai J, Wu B, et al. Automatic Irregularity-Aware Fine-Grained Workload Partitioning on Integrated Architectures[J]. IEEE Transactions on Knowledge and Data Engineering, 2019.

[73] Zhang F, Yang L, Zhang S, et al. FineStream: Fine-Grained Window-Based Stream Processing on CPU-GPU Integrated Architectures. USENIX Annual Technical Conference (USENIX ATC), 2020.

[74] 张峰, 面向集成异构平台的负载分析与优化关键技术研究[D], 清华大学, 2017.