中国科学: 信息科学 2020 年 第 50 卷 第 9 期: 1303–1326
SCIENTIA SINICA Informationis
《中国科学》杂志社
SCIENCE CHINA PRESS
纪念《中国科学》创刊 70 周年专刊 · 评述

E 级计算的几个问题

钱德沛*, 王锐

北京航空航天大学计算机学院, 北京 100191
* 通信作者. E-mail: depeiq@buaa.edu.cn

收稿日期: 2020–04–21; 接受日期: 2020–07–31; 网络出版日期: 2020–09–23

国家科技重点研发计划 (批准号: 2016YFB0200100) 和国家自然科学基金 (批准号: 61732002) 资助项目

摘要 过去 20 余年, 在国家科技计划持续支持下, 中国的高性能计算事业得到长足发展, 目前, 正在向 EFlops 级 (百亿亿次级, 简称 E 级) 高性能计算机的目标冲刺. 本文简要回顾了我国高性能计算发展的历史, 针对当前 E 级计算所遇到的困难, 从体系结构、处理器、互连网络、并行操作系统、并行编程、算法和可靠性等 7 个方面, 探讨了需要重点研究和解决的技术问题.

关键词 E 级计算机, 异构体系结构, 众核处理器, 互连网, 并行编程

1 引言

理论与分析、实验与观察、计算与模拟是人类认识客观世界规律, 产生重大科学发现的 3 个重要手段. 高性能计算为计算与模拟提供强大的计算能力, 因此成为发达国家激烈竞争的战略性、前沿性技术. 高性能计算不仅是解决国家面临的重大挑战性问题的利器, 同时也推动了计算技术的不断进步, 促进了计算机产业的发展.

过去 20 余年, 在国家科技计划持续支持下, 中国的高性能计算事业得到长足发展, 自主研发的高性能计算机连续排名世界第一, 国家高性能计算环境资源能力和服务水平位居世界前列, 自主研发的高性能计算应用软件正在各行业发挥作用. 目前, 正在向 EFlops 级 (百亿亿次级, 简称 E 级) 高性能计算机的目标冲刺. 但是, 应该看到, 在通向百亿亿次机的道路上充满荆棘, 有许多重大技术障碍需要克服.

本文针对 E 级计算机研制中的困难, 探讨需要重点研究的问题和可能的解决方案. 第 2 节对我国高性能计算的发展历史作简要回顾. 第 3 节提出 E 级计算机研制所面临的主要技术挑战. 第 4 节探讨在 E 级计算机研制过程中需要重点研究和解决的技术问题, 包括体系结构、处理器、互连网络、并行操作系统、并行编程、算法和可靠性等. 最后, 第 5 节给出本文的结语.

引用格式: 钱德沛, 王锐. E 级计算的几个问题. 中国科学: 信息科学, 2020, 50: 1303–1326, doi: 10.1360/SSI-2020-0099
Qian D P, Wang R. Key issues in exascale computing (in Chinese). Sci Sin Inform, 2020, 50: 1303–1326, doi: 10.1360/SSI-2020-0099

2 发展简要回顾

我国高性能计算机的早期代表性成果是国防科技大学在 1983 年完成的 “银河一号” 超级计算机 [1,2] 和中国科学院在同年完成的 757 大型向量计算机[1], “银河一号” 采用双向量阵列并行全流水体系结构, 64 位字长, 用射级耦合逻辑 (emitter coupled logic, ECL) 电路实现, 系统峰值计算速度超过每秒 1 亿次. “银河一号” 的诞生使我国跨入了世界研制巨型机的行列. 757 计算机采用中国科学院计算技术研究所提出的向量纵横加工和向量累加器的概念, 向量运算速度达到每秒 1000 万次, 标量运算速度达到每秒 280 万次, 该系统的成功推动了我国计算机研制水平的提高. 1986 年启动的国家 863 计划设立了智能计算机主题, 该主题最初的目标是研究面向人工智能的智能计算机系统. 1990 年, 该主题根据应用需求和技术发展趋势, 把主要研究方向从智能计算机转向并行计算机, 开始了 863 计划在高性能计算方向长达 30 年的研究历程. 1993 年, 国家智能机研究中心研制成功曙光一号 [3], 系统采用全对称紧耦合共享存储多处理器 (symmetric multi-processing, SMP) 体系结构, 包含 16 颗 Motorola 88100 处理器, 系统运算速度为每秒 6.4 亿次定点运算. 1995 年, 大规模并行处理 (massively parallel processing, MPP) 结构的曙光 1000 [4] 研制成功, 系统包含 32 个基于 Intel i860 的节点, 采用自研的虫孔路由的 Mesh 网将节点互连, 系统峰值速度每秒 25 亿次浮点运算. 1996 年, 采用集群结构的曙光 1000A 推出, 这是曙光第一款集群结构的计算系统, 此后的曙光 2000、曙光 3000 均采用集群结构. 进入 21 世纪, 研制万亿次计算机, 进一步缩小与世界最先进计算机的差距成为 863 计划的目标. 2001 年, 联想公司研制成功我国首台万亿次高性能计算机联想深腾 1800 [2], 此后在 863 计划支持下, 联想在 2003 年研制成功 5.3 万亿次的联想深腾 6800 [3]. 曙光公司在 2004 年推出 11.2 万亿次的曙光 4000A [5], 该系统在世界超级计算机 TOP500 排行榜中位居第 10. 从 2006 年开始的 “十一五” 863 计划根据国家中长期科技发展计划纲要, 把目标设定为研制千万亿次高性能计算机, 这在我国计算机研发史上是一次大的飞跃. 作为第 1 步, 首先研制了两台百万亿次级系统, 即 156 万亿次的联想深腾 7000 [6] 和 230 万亿次的曙光 5000A [7]. 从千万亿次机开始, 系统功耗已成为不可忽视的重要指标. 为了达到千万亿次机功耗不超过 2 MW 这个指标, 提出了异构混合体系结构, 即使用通用处理器和高性能的加速部件构成系统. 2009 年 10 月, 天河一号 [8] 研制成功. 系统采用通用 Intel 处理器和 AMD 的 GPU 实现, 峰值性能首次突破每秒千万亿次浮点运算. 2010 年 4 月, 曙光 6000 [9] 研制成功, 系统采用 Intel Xeon 处理器和 Nvidia 的 Fermi GPU 实现的异构结构, 峰值性能每秒 3000 万亿次浮点运算, Linpack 性能每秒 1270 万亿次浮点运算, 在世界 TOP500 排名第二. 2010 年 10 月, 异构加速结构的天河 -1A [10] 研制成功, 系统也采用 Intel Xeon 处理器和 Nvidia 的 Fermi GPU 实现, 峰值性能每秒 4700 万亿次浮点运算, Linpack 性能每秒 2560 万亿次浮点运算, 首次位居世界 TOP500 的榜首. 2011 年, 神威 · 蓝光 [11] 问世, 系统采用国产申威 16 核处理器实现, 是我国第一台全部采用国产处理器实现的千万亿次计算机.

早在 2008 年, 国际高性能计算界就提出了 E 级计算机的议题, 并开始了研讨和预研工作. 在攻克千万亿次机的难关后, 我国的 863 计划也将 E 级计算机列为下一步的目标. 作为研制 E 级机的第 1 步, “十二五” 期间研制十亿亿次 (100 PFlops) 高性能计算机. 经过两年努力, 中国人民解放军国防科技大学团队在 2013 年研制成功天河二号 (一期). 系统采用 Intel Xeon 处理器加 Intel

1) High performance vector computer. CCF China Computer History. [757 大型向量计算机组件. CCF 中国计算机历史记忆. https://www.ccf.org.cn/c/2018-09-12/652327.shtml].

2) DeepComp 1800. https://www.top500.org/system/173175.

3) DeepComp 6800 world's 14th fastest supercomputer. China Daily, Nov. 20, 2003. http://www.chinadaily.com.cn/en/doc/2003-11/20/content_283107.htm.

表 1 我国近 30 年研制的主要高性能计算机

Table 1 Major supercomputers developed in China in the recent 30 years

System	Year	Architecture	Peak performance	Linpack performance	Power
Dawning I	1993	SMP	640 MIPS	–	–
Dawning 1000	1995	MPP	2.5 GFlops	–	–
Dawning 2000	1999	Cluster	111.7 GFlops	–	–
Dawning 3000	2000	Cluster	403.2 GFlops	–	–
DeepComp 6800	2003	Hybrid cluster	5.3248 TFlops	4.183 TFlops	–
Dawning 4000A	2004	Cluster	11.264 TFlops	8.061 TFlops	–
DeepComp 7000	2008	Hybrid cluster	146 TFlops	102.8 TFlops	750 kW
Dawning 5000A	2008	Cluster	233.5 TFlops	180.6 TFlops	992 kW
Dawning 6000	2010	Heterogeneous accelerated	2984.3 TFlops	1271 TFlops	2.58 MW
TH-1A	2010	Heterogeneous accelerated	4701 TFlops	2566 TFlops	4.04 MW
Sunway BlueLight	2011	Multicore-based	1070.2 TFlops	795.9 TFlops	1.074 MW
TH-2/TH-2A	2013/2017	Heterogeneous accelerated	54.9 PFlops /100.68 PFlops	33.86 PFlops /61.44 PFlops	17.80 MW /18.48 MW
Sunway TaihuLight	2016	Heterogeneous manycore based	125.43 PFlops	93.01 PFlops	15.37 MW

Xeon Phi 加速器的异构加速结构, 峰值速度每秒 5.49 亿亿次浮点运算 (54.9 PFlops), Linpack 性能 3.39 亿亿次 (33.9 PFlops), 在 2013~2015 年连续 6 次位居世界 TOP500 榜首. 无锡江南计算技术研究所在 2016 年初研制成功神威 · 太湖之光系统. 系统基于片内异构的 260 核申威处理器实现, 峰值速度每秒 12.5 亿亿次浮点运算 (125 PFlops), Linpack 性能每秒 9.3 亿亿次浮点运算 (93 PFlops), 超越了天河二号, 在 2016~2017 年连续 4 次位居 TOP500 榜首. 天河二号 (二期) 本来计划通过简单更换新的 Intel Xeon Phi 加速器达到每秒 10 亿亿次的目标, 但是由于中国高性能计算机的进步引起美国的关注, 2015 年春, 美国商务部 (United States Department of Commerce) 把中国人民解放军国防科技大学及相关的国家超算中心列入实体名单, 禁止美国公司向中国人民解放军国防科技大学出售新的处理器, 使中国人民解放军国防科技大学不得不改变原有方案, 通过自行研制加速器升级天河二号. 为此, 天河二号 (二期) 推迟两年完成. 2017 年底, 天河二号 (二期) 即天河 -2A 完成研制, 系统采用新研制的加速器 Matrix 2000 替代了 Intel 的加速器, 峰值速度 100 PFlops, Linpack 性能 61 PFlops, 达到了预定的指标.

表 1 列出了近 30 年来我国自主研发的主要高性能计算机的基本情况.

与高性能计算机发展同步, 过去 20 余年, 我国的高性能计算基础设施也得到快速发展, 高性能计算应用的水平有了巨大提高. 基于自主的系统软件, 聚合互联网上的计算资源建成了国家高性能计算环境 “中国国家网格服务环境 (CNGrid)”, 目前总计算资源 460 PFlops, 存储容量 230 PB, 部署了数百个软件, 有数万用户, 支持了数千项国家重要的科研和工程项目. 我国高性能计算的应用领域从过去的气象、油气勘探等少数领域, 拓展到经济建设和百姓生活的方方面面, 已经从阳春白雪走向千家万户. 并行计算的水平从几十个处理器提高到千万核以上, 两次获得国际高性能计算应用最高奖 “戈登 · 贝尔奖”[4].

进入 21 世纪第 2 个 10 年, 美国、日本、欧盟的 E 级计算计划均进入具体实施阶段. 2015 年, 美国

4) ACM Gordon Bell Prize. https://awards.acm.org/bell/award-winners.

启动国家战略计算规划 NSCI[5]. 该规划提出要创造一整套可持续、多机构参与的国家战略规划及联邦政府投资战略, 让高性能计算为美国带来最大化的效益, 维持并提升美国在高性能计算研究、开发与部署领域的科学、技术与经济领导地位. 作为 NSCI 的一部分, 美国能源部正在执行 ECP 计划[6]. ECP 为研制 3 台 E 级计算机投资 18 亿美元, 第 1 台 E 级机 Aurora 将在 2021 年上半年完成, 持续性能将达到 1 EFlops (每秒百亿亿次). 第 2 台 E 级机 Frontier 将在 2021~2022 年问世. 第 3 台 El Capitan 的峰值性能将达 4~5 EFlops, 持续性能 2~3 EFlops, 将在 2023 年完成. 与超级计算机研制配套, ECP 为软件与应用研发另外投入 18 亿美元. 日本的 E 级机计划已经实施数年, 其第一台 E 级计算机 "富岳 (Fugaku)" 基于 ARM 处理器实现, 以保护软件资产. 为此研制了新一代 ARM 处理器 A64FX 48C. 2019 年下半年, Fugaku 的初阶系统安装到位, 其能效在 Green500 中排名第一 (16.9 GF/W)[7], 证明基于众核处理器的系统能效有可能超过基于 GPU 的异构加速系统. 2020 年 6 月, "富岳" 全系统完成并荣登世界超级计算机 TOP500 榜首, 其峰值速度 513.85 PFlops/s, Linpack 性能 415.53 PFlops, Linpack 效率达到 80.8%[8]. 包含 10 个超算中心的日本的超级计算基础设施 HPCI 也同步发展, 日本的计算资源将得到大大提升. 欧盟认识到由于欧洲缺少高性能计算机硬件厂商, 在 E 级计算的全球竞赛中已经被动. 为此, 在 2017 年启动了 E 级计算的 Euro HPC 计划, 欧盟通过 "地平线 2020" 框架计划在 2020 年年底前给 Euro HPC 投资 10 亿欧元, 在下一期框架计划 (2021~2028) 再投入 27 亿欧元, 参与 Euro HPC 的欧盟成员国将提供更多的配套经费. 欧盟将基于目前的 PRACE 基础设施, 在 2023 年前打造欧洲的 E 级计算基础设施 (3 台左右 E 级机)[9]. 欧洲高性能计算基础研究和应用的基础好, 在新的计算模型、语言、算法和大规模数值模拟技术等方面有深厚积累.

中国正在实施 "十三五" 重点研发专项 "高性能计算", 其目标是突破 E 级计算机核心技术, 依托自主可控技术, 研制满足应用需求的 E 级高性能计算机系统, 研发一批关键领域/行业的高性能计算应用软件, 构建高性能计算应用生态环境, 建立具有世界一流资源能力和服务水平的国家高性能计算环境. 目前专项正在为实现这一目标而努力 [12].

3 面临的挑战

世界超级计算机排行榜 TOP500 自 1993 年发布以来, 榜上世界最快的计算机的性能大约每 10~11 年会提高 1000 倍, 高于摩尔定律 (Moore's law) 所预测的性能提高速度. 这既得益于处理器速度的提高, 也来源于系统规模的不断扩大. 但是, 从 2013 年开始, TOP500 第 1 名的性能曲线变得平缓了 (见图 1[10]). 而且, 2019 年 11 月发布的 TOP500 的前 10 名与 2019 年 6 月的前 10 名相比较, 系统和性能完全没有变化, 这在 TOP500 的历史上是很罕见的. 按照目前的发展速度, 世界最快计算机的性能今后每 10 年大概只能提高 100 倍甚至更低.

TOP500 的数据说明超级计算机的发展的确遇到瓶颈, 其主要原因是: (1) 能效指标的约束. 根据美国能源部 (United States Department of Energy) 的指标, E 级机的功耗不得超过 20 MW, 这意味着不能单纯靠扩大系统规模提高系统的性能, E 级机可采用的技术手段受到很大限制. (2) 摩尔定律 [13] 接

5) Website of NSCI project. 2019. https://nsf.gov/cise/nsci/.

6) Website of ECP of DOE. https://exascaleproject.org/.

7) Prototype of Fugaku Supercomputer reaches Number One on Green500. https://insidehpc.com/2019/11/prototype-of-fugaku-supercomputer-reaches-number-one-on-green500/.

8) Japan Captures TOP500 Crown with Arm-Powered Supercomputer. https://www.top500.org/news/japan-captures-top500-crown-arm-powered-supercomputer/.

9) EU launches 1B Euro project to build fastest supercomputer in the world by 2023. https://sciencebusiness.net/news/eu-launches-eu1b-project-build-fastest-supercomputer-world-2023.

10) TOP500 website. https://www.top500.org/.

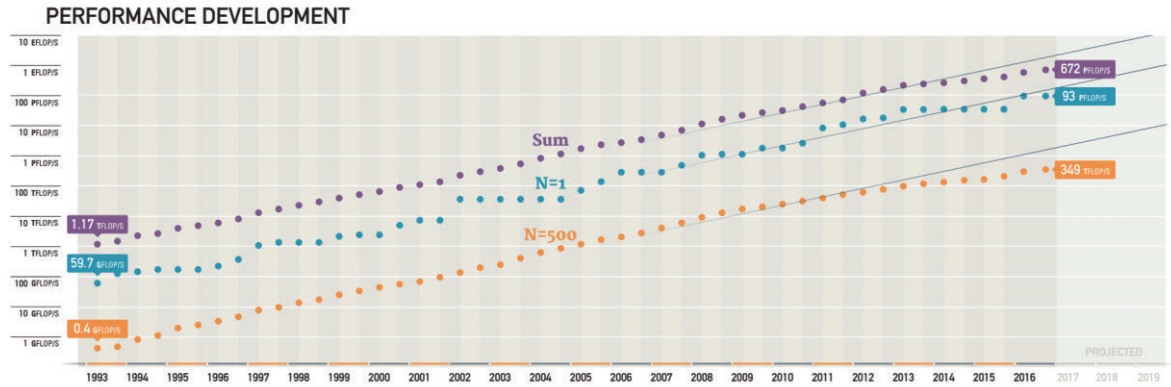

图 1 (网络版彩图) 超级计算机 TOP500 性能变化

Figure 1 (Color online) The performance increase of TOP500 supercomputers

表 2 中国 "十三五" 期间 E 级机指标

Table 2 Major performance index of China's exascale computer in the 13th 5-year plan

Index term	Expected performance/features
Peak performance	1 EFlops
Linpack efficiency	>60%
Memory capacity	⩾ 10 PB
Storage capacity	Exabytes supported
System energy efficiency	>30 GFlops/W
Performance of high-speed interconnect network	>400 Gbps, good scalability
Software	Node and system parallel operating system, large-scale resource management and scheduling system, common programming languages and compilers, easy-to-use parallel programming models and program development environment, system resource monitoring and management, fault-tolerant mechanism, support to reliable and scalable operation of large-scale applications

近失效, 处理器的性能已经难以按照摩尔定律持续增长. 而描述集成电路能耗的登纳德缩比 (Dennard scaling) 定律 [14] 已经失效多年, 处理器的能耗无法伴随半导体工艺的换代而成比例地降低. (3) 计算机体系结构变化缓慢. 由于多年来系统性能的提高依赖于芯片技术的进步, 并行计算机体系结构变化不大, 对性能改善贡献不足. (4) 没有颠覆性技术出现. 尽管新的计算技术, 例如量子计算机、超导计算机、DNA 计算机已经讨论多年, 但是在近期内, 还看不到实用的前景. (5) 新原理器件缺少突破. 新型的计算、存储、传输器件没有取得大的进展, 还难以替代当前的 CMOS 工艺器件.

我国 "十三五" 重点专项立项时设定的 E 级计算机系统的指标如表 2 所示.

要依托自主可控技术实现上述目标仍面临重大的技术挑战, 主要包括以下 4 方面.

低功耗 (power). 美国能源部设定的 E 级机能效指标是 1 EFlops 系统的功耗不得超过 20 MW, 也就是 50 GFlops/W 的能效. 到目前为止, 还没有找到能够确保达到该能效指标的有效技术途径. 我国 E 级机能效指标设定为 30 GFlops/W, 这是从实际出发设定的指标, 即使这样, 要达到也很困难.

高应用性能 (performance). E 级机不仅要追求高的峰值性能和 Linpack 性能, 更要追求应用可获得的实际性能. 然而, 在实践中, 很多机器的 Linpack 性能尽管可以达到峰值性能的 60% 以上, 但是运行实际应用, 性能经常在峰值性能的 10% 甚至 5% 以下.

良好的可编程性 (programmability). 大规模并行程序的编写本来就很困难, 面临着编程难、调试难、性能不确定等难题, 而异构加速体系结构的采用给并行编程带来新的问题, 严重影响系统的可编程性.

弹性容错性或韧性 (Resilience). E 级计算机的巨大规模使系统平均无故障时间 (mean time between failures, MTBF) 大为缩短, 预计只有几小时甚至 1 小时以下. 如何在这样的系统上保证应用的长时间不间断运行是严重的挑战.

应对这些 E 级计算的挑战只能依靠体系结构的创新、关键技术的突破和软件硬件的协同. 在第 4 节我们将讨论为应对这些技术挑战需要重视的几个技术问题.

4 E 级计算的几个问题

E 级计算涉及硬件技术和软件技术的方方面面, 其成功只能是多方面努力的结果. 本节针对 E 级计算面临的重大技术挑战, 提出并分析若干重要的研究问题.

4.1 体系结构

随着登纳德缩比定律和摩尔定律的失效或接近失效, 单靠半导体工艺的改善、芯片主频的提高就能获得系统性能增益的路快要走到尽头了. 登纳德缩比定律的结束意味着芯片功耗会急剧上升, 而摩尔定律的放缓, 使片内晶体管的数量无法再按照确定的性能 – 成本规律持续增加. 因此, 必须通过探索新的体系结构, 开辟提升系统性能的新途径. 图灵奖获得者 John Hennessy 和 David Patterson 在其获奖演说中回顾了几十年来计算机的发展历史, 提出要用体系结构创新改善计算机的性能、成本和能效. 他们预见将会出现像 20 世纪 80 年代那样的体系结构研究的黄金年代, 国际上有学者将此比作计算机体系结构的 “寒武纪爆发”. 在 20 世纪 80 年代, 出现了 RISC、超标量处理器、多层次缓存、预测执行、编译优化等一大批体系结构创新, 使计算机性能每年提升约 60%. 今天能否再次出现类似的体系结构 “百花齐放、百家争鸣” 的局面? 能否从以规模取胜的庞大 “恐龙” 式系统, 向灵巧、节能、应用高效的 “哺乳动物” 式系统转变? 这是全世界体系结构设计者, 也是 E 级计算机设计者必须回答的问题.

体系结构要解决的基本问题是: (1) 机器的体系结构与应用的计算模型相匹配. 虽然应用问题的内在并行性会制约可达到的性能 [15, 16], 但好的计算模型能够挖掘问题的并行性. 体系结构要适应计算模型所能挖掘的并行性, 在问题的分解、激进投机执行、数据竞争与冲突消解、通信与同步、存储一致性等方面予以支持. (2) 计算与访存相匹配. 冯 · 诺伊曼体系结构的基本特征是存储程序执行, 程序和数据都放在内存中, 程序的执行离不开访存, 存储器是关键通路. 而内存性能的进步一直落后于处理器性能的进步, 由此造成 “存储墙” 问题 [17]. 而且, 访存操作的能耗也比浮点运算高得多, 成为系统能耗的重要组成部分 [18]. 高性能低功耗的存储器是体系结构必须解决的问题.

4.1.1 异构体系结构

事实上, 没有哪一种体系结构能够完美支持各类不同的计算模型, 满足所有问题高效求解的需求. 通用还是专用一直是体系结构设计中争论的问题. 采用专门设计的硬件实现的专用计算机的确能在求解某些问题上达到很高的性能, 能效指标也好, 但是这样的专用系统往往只对特定问题有效, 且伴之而来的是编程困难、使用不灵活等问题. 另一个极端, 全部采用通用处理器实现的通用计算机具有编程性好、适应面宽的优点, 但是性能和功耗都不可能很高. 因此, 通专结合, 在性能、能效和可编程性间

求得平衡, 是研制 E 级计算机的一个基本思路. 2008 年后出现的异构体系结构体现了这样一种思路. 它力求在性能、能效和可编程性之间求得平衡, 以通用 CPU 适应大范围应用, 获得好的可编程性, 以高计算能力、高能效的加速部件完成程序的计算密集部分, 二者协调实现系统的高性能和低功耗.

异构计算系统大致可分为两类, 一类称之为节点内异构, 美国的 “TITAN”, 中国的 “天河一号”、“天河二号”, 以及美国最新的 “Summit” [19] 都是这类系统. 这类系统的节点由通用 CPU 和若干加速处理器构成, 加速器以协处理器方式工作. 加速器与 CPU 数量之比一般为 2:1 到 4:1. 加速器多采用 Nvidia 的 GPU, 也有 Intel 的 Xeon Phi. CPU 与加速器之间用 PCIE 总线连接. 经过多年发展, 这类系统的软件已经相当丰富, 适用于多类型的应用, 可以达到很高的性能和能效. 这类系统的缺点是 CPU 和加速器之间的数据交换的瓶颈, 因为 CPU 和加速器工作于不同的物理内存空间, 待加工的数据和加工后的结果必须在 CPU 和加速器之间传输. 此外, 不同节点的加速器之间不能直接通信. “Summit” 采用 NVLink 实现节点内互连, 使得节点内数据交换性能更好, 也更加灵活, 但并未解决不同节点的加速器之间的直接通信. 而且, 该结构并不适用于所有问题. 例如, 有些问题仅使用 CPU, 使得节点上的加速器空闲, 造成资源的浪费. 另外, 尽管这类系统的编程已有较好的工具支持, 但是要调试出好的性能仍很不容易. 另一类异构系统是片内异构, 美国的 “走鹃” 系统采用片内异构的 Cell 处理器[11], 是第一个峰值性能达到千万亿次的片内异构系统. “神威 · 太湖之光” 采用 260 核的申威 26010 众核处理器实现 [20], 也是这类系统的代表. 申威 26010 内含 4 组计算核心, 每组由一个主核加 64 个从核构成, 使用统一的内存空间, 具有较好的可编程性. 由于计算核心数量多, 计算能力很强, 一个处理器就达到 3 TFlops 的性能, 功耗不超过 300 W. 对于计算密集型的应用, “神威 · 太湖之光” 可以达到很高的性能. 这类系统的缺点是, 相对于计算核心的数量而言, 内存容量较小, 每个从核分到的访存带宽不够. 程序员需要精细地调整片内寄存器和局部存储器的使用才能达到程序性能的优化, 这需要相当高的技巧, 有相当大的难度. 针对这两类异构系统的缺点, 我们提出了系统级异构 (或称分区异构, partitioned heterogeneity) 的概念 [21]. 在这样的系统中, CPU 和加速器处于同等地位, 由同一张互连网连接, 通过软件配置系统的形态. 系统的不同部分可以配置成纯 CPU、纯加速器, 或者 CPU + 加速器等不同形态, 根据应用的性质而定. 这样做的好处是, 资源根据应用特点按需使用, 不会造成浪费. 加速器之间可以通过互连网直接通信, CPU 和加速器之间的数据传输瓶颈也可以得到缓解. 当然, 完成系统配置任务的软件更为复杂. 这类系统是否合理, 还需要实践的检验.

4.1.2 同构体系结构

虽然异构体系结构已经成为世界 TOP10 计算机中的主流, 但基于通用众核处理器的同构体系结构仍然是实现最高性能计算机的选项之一. 日本的 “京” 计算机、“富岳” 计算机以及中国的 “神威 · 蓝光” 计算机都属于同构结构的计算机. “富岳” 登顶 TOP500 且实现了很高的 Linpack 性能和效率, 说明同构体系结构仍有其生命力. 基于通用处理器的同构结构的机器最大的优点是易于使用, 能够适应宽广的应用面. 编程人员面对的是同一种处理器, 在问题分解时不必考虑 CPU 和加速器的不同特点和性能, 传统的问题分解和负载平衡策略都可以适用. 异构系统中因加速器的使用而引入不同编程语言的问题也不复存在, 因此系统可编程性好. 在异构系统中 CPU 和加速器使用不同的物理内存空间, 由此产生 CPU 和加速器之间数据交换的瓶颈问题, 而在同构系统中, 地址空间是一致的, 没有额外的数据交换的需要, 因此也不存在异构系统中 CPU 和加速器间数据交换的瓶颈. 此外, 基于通用处理器的同构计算系统也有利于软件的移植. 一般而言, 只要重新编译源代码, 现有的自研软件或商业软件就可以在系统上执行, 而在异构系统中, 软件移植意味着至少加速器的代码部分要重新编写, 而

11) ROADRUNNER. https://www.top500.org/resources/top-systems/roadrunner-los-alamos-national-laboratory/.

且往往涉及整个程序结构的变化, 甚至影响到高层的编程模型. 为了更好地支持传统的数值模拟应用, A64FX 48C 处理器中增加了 512 位的可扩展向量部件 SVE. 针对深度学习等人工智能应用的需求, A64FX 48C 支持 8 位整数运算和 16 位、32 位、64 位等多种字长的浮点运算. 再加上处理器内置互连网络接口、HBM2 实现的高性能的内存, 以及高达 0.4 的访存带宽与计算速度之比等特性, “富岳” 达到了很高的应用性能. 基于通用处理器的同构系统的最大缺点是功耗较大. 尽管 “富岳” 计算机采用了最新设计的 A64FX 48C 高能效处理器, 但是系统的功耗仍高达 28.33 MW, 对于一台峰值性能略微超过 500 PFlops 的机器而言, 这个功耗还是偏高了. 当然, 通过降低计算核的复杂性, 在片内放置更多的计算核, 采用更先进的集成电路工艺实现, 都有可能进一步降低众核处理器的功耗, 进而改善系统的能效指标. 但是改进的空间究竟有多大, 仍需今后的实践来证明.

4.1.3 高效低耗的存储器

存储器的设计关乎性能和能耗, 是体系结构设计的重要组成部分. 存储墙的存在、访存和数据传输操作的高能耗, 都要求从体系结构角度考虑如何尽量减少存储器访问和数据的传输, 缩短数据传输的距离.

流 (streaming) 式结构 [22] 是减少访存的一种方式, 其出发点是减少内存数据的存取. 数据一旦从内存取出, 就在其流动中完成处理, 中间结果不需要存回内存, 再重新取出. 在信号处理、图像处理等领域, 流处理已经得到很好的应用. 数据流机是 20 世纪 70 年代就提出的体系结构概念, 其原理是发掘和利用数据的内在并行性, 依靠数据的到来激活相关的操作, 而不是靠显式的机制控制操作的并发执行. 一个操作当其所需要的数据都就绪时就被激活执行, 处理数据, 并将结果发往目的地, 去激活后续操作. 如果所处理的数据具有内在并行性, 操作就可以并行地执行. 但是, 由于对应的微体系结构难以具体实现, 数据流机一直没有成为主流. 英国的 Maxeller 机的设计者声称 Maxeller 机是静态的数据流机 [23], 由软件把高层业务逻辑转换成数据流逻辑, 然后按照数据流配置 FPGA 实现的硬件. 输入数据流入系统, 经过配置好的数据流硬件处理, 流出的就是结果. 中间不涉及访存, 效率高、能耗低. 该系统在金融、油气勘探中得到应用. 近年来, 随着大数据和人工智能应用的兴起, 数据流机概念又重新活跃起来, 特别是以软件实现数据流处理的研究工作值得关注.

提高数据的重用是减少访存的传统途径, 几十年来, 高速缓存 (Cache) 的组织结构及其一致性协议设计都是基于和利用了数据的局部性, 以达到数据的重用 [24,25]. 异构体系结构给 Cache 设计引入新的问题, 例如 CPU 和加速器的不同存储空间如何建立 Cache、提高重用、减少数据传输 [26]. 在众核条件下, 如何改进 Cache 结构、拓展 Cache 协议、支持代码块的原子性执行, 都是需要研究的问题 [27]. 在降低 Cache 一致性协议开销方面仍有空间, 例如, 区分程序中的私有数据和共享数据, 只有共享数据才需要维持一致性, 而私有数据可以旁路一致性协议操作, 这样可以降低一致性协议的开销, 提高程序执行性能 [28].

让存储器和处理部件尽可能靠近, 是缩短数据传输距离、减少数据传输能耗、降低数据访问时延的手段. 一种做法是把 DRAM 内存放进处理器芯片内部, 以此提高访存带宽、降低访存时延 [29,30]. 当然, 受芯片面积限制, 片内的内存不可能太大, 而且片内内存成为存储体系结构中的额外一层, 要和片外内存协调管理. 另一种做法以存内处理 (processing in memory, PIM) [31] 为代表, 是把一些处理功能放到存储芯片里面去, 这样这些操作在存储器内部就可以完成了, 不需要把数据取出来再处理. 要解决的问题是哪些操作可以在存储器内完成, 要权衡硬件的开销和获得的收益. 基于硅穿孔 (through silicon via, TSV) 技术的 3D 封装存储器 (high-bandwidth memory/high-bandwidth memory 2, HBM/HBM2) 是目前最实际的缩短存储器和处理器之间距离的手段, 由于互连线短且数量多, 存储器的带宽和时延

表 3 几种非易失性存储器的性能对比, 数据来自文献 [33]

Table 3 The comparison of typical non-volatile memories, data from [33]

	SRAM	DRAM	Flash (NOR)	Flash (NAND)	FeRAM	MRAM	PRAM	STT-RAM
Non-volatile	No	No	Yes	Yes	Yes	Yes	Yes	Yes
Cell size (F^2)	50∼120	6∼10	10	5	15∼34	16∼40	6∼12	6∼20
Read time (ns)	1∼100	30	10	50	20∼80	3∼20	20∼50	2∼20
Write/Erase time (ns)	1∼100	50/50	1 μs/10 ms	1 ms/0.1 ms	50/50	3∼20	50/120	2∼20
Endurance	10^{16}	10^{16}	10^5	10^5	10^{12}	$>10^{15}$	10^{10}	$>10^{15}$
Write power	Low	Low	Very high	Very high	Low	High	Low	Low
Other power consumption	Current leakage	Refresh current	None	None	None	None	None	None
High voltage required (V)	None	2	6∼8	16∼20	2∼3	3	1.5∼3	<1.5

都可以得到大大改善. 需要解决的问题是如何降低功耗减少发热, 如何设计有效的散热机制, 以及如何管控存储器物理空间的访问, 避免局部过热 [32].

随着非易失存储 (NVM) 器件的逐渐成熟, 存储器的实现方案有新的选择. 非易失存储器件存储密度高, 有利于实现大存储容量, 读出的速度快、能耗低, 而且不需要刷新, 不会带来刷新的能耗, 也不存在刷新带来的性能开销. 但最大的缺点是写入的速度慢、能耗高, 而且有限的写入次数对其工作寿命有很大影响, 不适合用作反复读写的内存. 表 3 [33] 给出了几种非易失存储器件的性能参数. 将 DRAM 和 NVM 结合的混合内存是兼有二者优点的一种尝试. 其思路是只读数据放在 NVM 中, 读写数据放在 DRAM 中, 根据数据使用的性质动态地将数据在 DRAM 和 NVM 之间切换. 希望达到既提高容量和能效, 又避免写入开销大和寿命短的问题. 和传统的 Cache 设计基于数据访问的时间和空间局部性类似, 混合内存建立在数据在多数时间是只读的这一假设之上, 如果这点不成立, 就失去了前提. 在混合内存的组织结构和访问协议上还有很多工作要做, 其合理性需要实践的检验.

4.1.4 创新体系结构

领域特定体系结构是 Patterson 和 Hennessy 教授 [34] 对体系结构发展趋势的判断. 他们在图灵奖获奖演说中指出, 体系结构研发黄金时代的特点是敏捷的体系结构设计和实现能力. 云上提供 FPGA, 使每个人都可以创新, 可以贡献, 都可以设计和部署自己定制的 “硬件”. 通过抽象领域应用的特征, 以领域特定语言描述, 发展最适合的领域特定体系结构, 用免费的开放体系结构和开源硬件实现, 这是研发领域特定体系结构的必由之路.

美国的 Chien 教授 [35] 在讨论处理器体系结构发展时曾以瑞士军刀和专用工具箱做比方. 瑞士军刀包含的小工具种类繁多, 但哪一个都不好用, 真正好用的是包含众多专一用途工具的工具箱.

我们认为未来创新的体系结构一定是异构的, 只有贴切应用特征, 用最合适的硬件完成特定的功能, 才可能高效, 这是基本原则. 未来的异构一定会更加多样化, 因为应用特征是多样化的. 未来的异构一定会进一步深入芯片内部, 把半导体工艺进步所带来的可用面积转变为支持不同操作的高效部件. 我们今天已经看到这种趋势, 例如支持张量计算的张量处理器, 支持图计算的图处理器等, 都是面向特定功能的高效部件. 未来的系统一定是可动态重构的. 这种动态重构不一定是基于 FPGA 的现场可编程, 而是根据需要, 动态组织系统内或片内各种各样加速部件协同工作. 可以想象, 众多完成特定功能的部件各尽所能, 按需使用, 这样的系统一定是高能低耗的.

与其预测某一种体系结构是未来的主流, 还不如建立敏捷的体系结构研发及其硬件实现的能力, 这样, 适应应用特征的各种各样的体系结构就会不断涌现, 而它们的硬件实现又能够像制造一个晶体

管那样容易. 只有达到这个境界, 创新的体系结构才能源源不绝, 才能有生命力.

4.2 处理器

4.2.1 提高处理器的能效

E 级计算机对处理器的能效提出了十分苛刻的要求, 降低功耗是处理器设计面临的最大挑战. 几十年来, 处理器的发展一直遵循登纳德缩比定律. 根据该定律, 芯片功率密度不随工艺换代而改变. 芯片面积 A 中的器件消耗的动态功耗可以表示为

$$P_{\mathrm{dyn}} = C \times V_{\mathrm{dd}}^2 \times f, \tag{1}$$

其中, C 是电容, V_{dd} 是工作电压, f 是工作频率. 那么 A 的功率密度为

$$(C \times V_{\mathrm{dd}}^2 \times f)/A. \tag{2}$$

每一次工艺换代都使器件的各维度的尺寸、电容和 V_{dd} 均降低为原来的 0.7, 而频率提高为 $f/0.7$, 这样, 容纳同等数量晶体管的面积缩小为 $0.7^2 \times A$. 新一代工艺下的功率密度

$$(0.7C \times (0.7V_{\mathrm{dd}})^2 \times (f/0.7))/(0.7^2 \times A) = (C \times V_{\mathrm{dd}}^2 \times f)/A \tag{3}$$

保持不变. 但是当工艺达到 130 nm 时, 这个规律不再成立. 首先, V_{dd} 不可能再按 0.7 比例下降. 其次, 静态功耗所占比重越来越大. 结果使芯片功率密度快速增长, 供电和散热成为大问题, 其结果是处理器的主频不能随工艺换代而成比例提高, 性能的提升有限. 要提高处理器性能只能走多核/众核的路.

降低 V_{dd} 是降低处理器功耗的主要手段, 因为降低 V_{dd} 可以按平方关系降低动态功耗, 以超线性关系降低静态功耗. 例如, V_{dd} 为 500 mV 的芯片要比 V_{dd} 为 0.9 V 的节能 40 倍 [36,37]. 但是, 芯片 V_{dd} 的降低受到制造工艺偏差的制约, 因为工艺的偏差使不同区域的器件参数离散, 为保证芯片可靠工作, 芯片 V_{dd} 和工作频率的选择只能照顾片内最差的器件, 造成很大的浪费.

应对参数离散性的主要手段如下所述.

分区电压频率控制. 工艺变化造成的参数离散与物理位置相关, 即相邻器件的参数较为一致. 因此, 可以把芯片分区, 为不同的区域选择最适宜的 V_{dd} 和工作频率. 分区供电需要高效率的电压调节器, 一种做法是两级电压调节 [38], 芯片包含 1 个或几个一级调节器, 每个一级调节器下接多个二级调节器, 每个二级调节器为一个或几个区域供电. 这样设计的电压调节机制损耗低, 输出纹波小. 片上网络对于参数的离散性更为敏感, 如果采用单一 V_{dd} 供电, 就只能照顾最差的器件, 选择较高电压. 可以将片上网络分区, 每个区包含若干参数一致的路由器. 对每个区的 V_{dd} 动态调节, 直至达到传输错误率满足要求的最低电压为止 [39].

分簇众核结构. 尽管降低 V_{dd} 会降低功耗, 但同时也使电路工作的速度降低, 只能靠增加并行工作的核来弥补性能损失. 一个众核处理器内包含大量简单的计算核, 考虑到器件参数的离散性, 可将核按簇组织, 因为一个簇内器件的物理位置相邻, 参数差不多, 可选择相同的 V_{dd} 和主频. 簇内采用异构结构, 包含一个主核和众多从核, 主核高压 V_{dd} 供电, 用于快速执行串行或临界区里的代码, 而从核低压 V_{dd} 供电, 有效降低功耗 [40]. 片内异构的申威 26010 众核处理器 [20] 是这类众核处理器的代表. 它内含 4 组计算核心, 每组由一个主核加 64 个计算从核构成, 一共 260 个计算核心. 它以 300 W 左右的功耗达到 3 TFlops 的性能, 能效是很高的.

细粒度电源门控. 电源门控技术通过关闭当前或未来一段时间不会使用的功能部件的供电, 降低芯片的静态功耗. 控制的区域可以精细到片上存储器的某个区域或片上网络的某些路由器. 电源门控

的粒度越精细, 节能的效果越好, 但是需要对功能部件的当前及未来的工作状态作细粒度的动态监视和预测, 粒度越精细, 实现的复杂度也越高.

减少数据传送是降低功耗的另一个重要手段. 随着芯片规模的扩大, 片内数据传送的能耗在芯片整体能耗中的比重越来越大 [41]. 一种方案是将计算核心及其所带的存储器按照层次结构的簇来组织, 这样系统软件就可以把相互通信的线程及其数据安排到同一个簇中, 减少计算过程中数据传输的总量, 同时缩短传输距离. 另一种技术是片内采用单一地址空间, 由软件显式地管理数据在 Cache 层次结构中的移动, 而不是依靠硬件的 Cache 一致性机制 [42]. 许多 E 级系统的应用具有相对简单的控制结构与数据结构, 例如, 在规范的循环中访问阵列数据. 可以利用现代编译技术精准地分析数据访问模式, 生成目标代码, 使片内存储层次结构中的数据传输尽可能地少. 此外, 单一地址空间的另一个优点是通信时不需要内存拷贝, 进一步减少数据的搬运. 申威 26010 众核处理器采用了类似的实现方式. 它的内存地址空间统一编址, 仅在 4 个主核之间保持高速缓存一致性, 计算从核不使用数据 Cache, 而是使用独立编址的局部存储器, 由软件管理数据在内存、局存和高速寄存器组之间的移动. 这样做虽然增加了编程的难度, 但是有效降低了功耗. 高效的通信和同步原语也是降低通信开销的手段, 例如, 动态的层次结构的硬件栅障、硬件实现的 “占用 – 空闲” 位支持的核间点对点同步等 [43].

4.2.2 新型处理器

深度神经网络计算逐渐成为高性能计算机的重要负载. 用于处理图像和语音的深度神经网络算法动辄包含上百个隐含神经元层次和数以亿计的权重参数, 如果采用通用处理器, 一次推理计算需要耗费几百亿到几千亿个处理器周期. 相比之下, GPU 更加适合神经网络的推理和训练计算. GPU 利用大量简单处理器核组成流处理器 (streaming processor), 以单指令多线程的方式, 同步式地调度线程组 (warp). 这种方式能以较高的效率执行大量的矩阵乘、向量乘等运算. 但是由于 GPU 的体系结构并非专门为神经网络计算定制, 而且深度神经网络结构多样, 每种算法的可并行规模和访存特征并不一定能完全匹配 GPU 的资源, 会带来很高的能耗代价. 例如英伟达 Tesla V100 GPU 处理器尽管理论上最高可以达到半精度 125 TFlops 的计算性能, 而功耗也高达 300 W. 因此, 针对深度神经网络的计算特征设计专用的计算加速部件从而提升计算效率具有重要意义. 神经网络加速器的实现途径主要有两种, 一种是以硬件模仿生物神经网络, 以电脉冲的方式激活神经元执行计算, 一般称为神经拟态处理器; 另一种是以处理器指令模拟神经网络的计算过程, 从存储器访问和指令设计等方面进行针对性优化, 一般被称为人工神经网络 (artificial neural networks) 处理器.

中国科学院计算技术研究所研制的 DIANNAO 系列深度学习处理器 [44, 45] 就是一种人工神经网络处理器, 为了适应算法特点并降低功耗, 在微体系结构上对神经网络进行分块处理, 将不同类型的数据块存放在不同的片上 RAM 中, 尽可能地减少处理器与片外存储器之间的数据搬运. 在此后的寒武纪系列处理器中, 还针对深度学习算法提出了专用的深度学习处理器指令集 Cambricon [46], 采用专门的函数指令集设计, 在运行时通过微编程动态调整电路的结构以适应不同深度学习算法的特性, 高效地完成深度神经网络函数.

Google 的 TPU [47] 也是一种人工神经网络处理器. 它由 65536 个 8 比特的乘加器组成阵列, 计算峰值可以达到 92 TOPS. 其体系结构最主要的优化在于采用了确定性执行模型, 使其省去了 CPU 和 GPU 的乱序执行、分支预测等复杂判断和控制逻辑, 能够达到较高的能效比.

IBM 公司的 TrueNorth [48] 是一种模仿脑结构的脉冲神经网络处理器, 它仿照生物神经网络用 54 亿个晶体管搭建了 1 百万个可编程的脉冲神经元和 2.56 亿个神经突触, 从根本上改变了计算和存储分离的传统计算机体系结构. 由于采用了脉冲电路, 神经元只有在有信号通过时才被激活, 因此芯

片功耗仅有 63 mW. 美国劳伦斯利沃莫国家实验室 (Lawrence Livermore National Laboratory, USA) 部署了一台用 16 个 TrueNorth 芯片构建的计算机[12], 可以以 2.5 W 的功耗执行计算.

Intel 的 Loihi [49] 也是一种非冯 · 诺依曼 (von Neumann) 结构的类脑神经网络处理器, 它在之前的神经拟态处理器基础上加入了神经元分层连接和突触学习规则可编程等特点. Loihi [50] 在嗅觉学习上的研究以多个气体传感器同时采集的嗅觉感应数据为输入, 模拟了大脑对嗅觉的反应机制, 可以用极小的样本达到学习的目标.

从体系结构设计的角度看, 在灵活性和执行效率之间取得平衡是人工智能处理器面临的最主要问题. 由于人工智能算法的迭代速度很快, 处理器必须能够提供必要的灵活性, 在未来一段时间内支持算法的更新, 这不可避免地会牺牲执行效率和优化空间. 以指令集方式模拟神经网络执行的处理器对算法的适应能力较强, 尽管能效显著高于通用处理器, 但仍然无法达到神经拟态处理器的超低功耗水平. 神经拟态处理器在神经元节点上实现了计算和存储的融合, 但是由于具有独特的连接结构, 因此无法移植从其他不同类型处理器中训练得到的算法. 另外, 拟态芯片大多采用模拟电路构建, 生产工艺带来的神经元和突触的个体差异给误差控制带来了很大的挑战.

图数据结构可以天然地表示数据之间的关联属性, 图计算也因此成为了主要的大数据分析手段. 图算法一般需要在大范围的图遍历过程中执行原子操作, 因此更加倾向于将数据尽可能多地装载到内存中甚至处理器芯片内, 这给面向传统负载设计的多层存储体系带来了巨大挑战. 由于图计算主要关心的是节点或边之间的关系, 而不是对节点和边本身执行复杂计算, 因此在图遍历过程中的计算量很小, 计算访存比很低, 这与传统计算机体系结构以计算为中心的设计不能很好地适配. 对图计算来说, 如何将数据与计算更加紧密地结合起来是主要的挑战.

GraphPIM [51] 是一种将近存计算思想与近年来的三维堆叠存储器 (hybrid memory cube/high-bandwidth memory, HMC/HBM) 结合起来的图加速技术. 在三维堆叠存储器中, 一般在最底层设计一个 CMOS 逻辑层, 可以部署低复杂度的计算逻辑 (出于散热的考虑, 一般不会直接将高性能处理器直接放置到最底层), 如位操作、比较、布尔操作、简单算术操作. 在逻辑层之上堆叠若干层 DRAM 芯片, 层与层之间用硅穿孔或其他导线技术互连. GraphPIM 将一些针对数据的原子操作 (例如比较与置换操作 CAS) 编译成 HMC 存储器支持的操作组合, 然后将其发送到 HMC 底层逻辑层, 将数据在 HMC 内部处理, 省去数据的读出、比较和再写入. 文献 [52, 53] 也是利用三维 DRAM 存储器设计的存储计算加速图计算方法. Graphicionado [54] 是一种以近存储计算方式实现的图计算加速处理器. 针对通用处理器在图数据处理上的不足, 例如 Cache 装载与访问粒度不匹配导致片外访问利用率低、缓存无法容纳足够的可重用数据以及指令执行的数据类型与图数据结构不匹配等, 重新设计了数据类型和存储子系统, 在预处理阶段分别将图结构中的边和节点建立列表和索引, 在处理阶段, 用软件管理的 Scrachpad 作为片上缓存, 定义了顺序节点写、随机节点读、随机节点写等原语, 充分发掘对边和节点访问的并行性, 加速预处理阶段数据结构的执行. 总的来说, 图计算加速的核心在于改变传统处理器中计算和访存两者的主/辅地位, 以一种数据访问为核心的思想设计整个系统.

随着工艺的进步, 芯片集成度还会进一步提高, 如何利用好数量不断增长的晶体管, 发展更高性能、更低功耗的处理器始终是国际计算机界努力的方向. 随着处理器内核数量的上升, 核的异构性也会越来越显著, 未来的处理器中, 可能会集成各种各样能高效完成特定计算的部件, 达到更好的能效水平. 而新的集成电路技术, 例如小芯片 (chiplet) 技术和晶圆级集成 (wafer scale integration) 技术, 将支持和推动这一发展趋势.

12) Lawrence Livermore National Laboratory and IBM Collaborate to Build Brain-Inspired Supercomputer. https://www-03.ibm.com/press/us/en/pressrelease/49424.wss.

4.3 互连网络

互连网络是并行计算机的核心部件, 它连接并行工作的节点, 提供数据传输和同步的通路. 互连网络的性能在很大程度上影响系统性能, 一张高性能的互连网是大规模并行计算机高效稳定工作的基础.

E 级计算机对互连提出了前所未有的要求. (1) 它必须具有很高的带宽, 特别是对分带宽, 因为它代表了网络的最低性能. 对于 E 级计算, TB/s 级或至少数百 GB/s 级的互连带宽是必须的. (2) 互连的最大点对点时延必须要低, 与电信级通信不同, 高性能计算机中的互连网既追求高带宽, 又追求低时延, 因为处理器之间传递大量同步短消息, 互连网必须以非常低的时延传递这些短消息, 这对网络时延提出很高要求, 应该在亚微秒级别. (3) 网络的可扩展性要好, 对于 E 级系统, 应该能有效地连接 10 万个以上的节点, 未来的 E 级系统可能达到这样的并行规模. (4) 互连网的能耗要低, 在系统总能耗中互连所占的比例要降低或至少不能提高. (5) 互连网络要能稳定工作, 具有很高的长期可靠性.

上述对 E 级系统互连的要求涉及方方面面的网络技术. 比如, 链路技术, 即传输是同步的还是异步的, 是并行的还是串行的, 是电信号传输还是光信号传输, 采用何种编码方案等. 在拓扑结构方面, 重要的问题是在保证性能的前提下, 降低信道数量从而降低成本, 同时尽可能缩短网络的直径, 降低网络的时延. 在工程实现上, 要用标准的交换机实现不同的网络结构, 如胖树、蝶形网、高维的环网 (torus) 或超立方网等. 在网络协议方面, 是电路交换还是分组交换, 是集中控制还是分布控制, 都是要做出的选择, 还要发展高效率和高度稳定的路由协议. 流量控制涉及虚拟通道、缓冲等技术, 主要目的是避免死锁和活锁, 提高网络信道的利用率, 改善网络性能, 保证服务质量.

一方面, 互连网络是一个已有长期研究积累的领域, 已经有大量成熟的技术和产品, 形成了系列的国际标准. 我国在互连网络方面也已取得长足的进步, 例如, 天河 -1A 与天河二号的自主胖树互连网络 [10], 天河 E 级原型机的自主蝶形网络, 曙光 E 级原型机的 6-D Torus 网络等. 但另一方面, 互连网络性能和能效的进一步改善遇到技术瓶颈, 还不能满足 E 级计算机的需要.

互连网络发展的两个趋势值得重视: (1) 随着计算机规模的不断扩大, 互连网络能耗在系统总体能耗中的比重越来越大. (2) 随着片内计算核数的不断增长, 连接计算核的片上网络日益重要. 无论是系统域网络, 还是片上网络, 都需要新的颠覆性技术的支持. 光互连网是有发展前途的技术, 因为不论是带宽还是功耗, 光网络都有其独特优势, 特别是伴随硅光子 (SiP) 技术的进步, 以硅光子技术实现的稠密波分复用 (dense wavelength division multiplexing, DWDM) 技术可能成为未来系统互连和片内互连的主角. 目前的超级计算机中电路板间和机柜间已经使用光纤作为传输介质, 由于互连网络结构复杂, 光纤数量巨大. DWDM 允许在单一光纤 (波导) 上传输多个不同波长信道的数据, 因而可以大大降低系统中互连光纤的数量. 此外, 光信号的传输能耗很低, 在系统互连的范围基本与传输距离无关, 可以在提高互连性能的同时降低其能耗. SiP 技术以相同的微电子工艺在同一硅片上实现光通信与传统计算的功能, 以 SiP 实现的 DWDM 是实现芯片间和芯片内光通信的理想技术. 近期, 以 SiP 实现的 DWDM 光通信芯片, 可紧挨着别的芯片放置, 或者以多模块封装技术与别的芯片封装在一起, 实现电路板上芯片间通信. 更远一点, 可实现众核处理器内的片上光网络. 文献 [55] 对基于 SiP 技术的 DWDM 芯片的带宽密度和功耗作了精细的建模和分析, 其结果显示, 基于微环的 SiP 技术能够在仅占用 1.5 mm^2 硅片面积的单条光链路上提供 1.8 Tbps 的带宽. 光芯片的能效可达 0.9 pJ/bit (每信道 12.5 Gbps 带宽) 和 1.5 pJ/bit (每信道 25 Gbps 带宽). 加上入出光芯片的电信号数据传输, 能效可达 1 pJ/bit (每信道 12.5 Gbps 带宽) 和 2 pJ/bit (每信道 25 Gbps 带宽). 因此, 配合多光纤连接的光芯片及多链路封装模块技术, 可以为处理器 – 内存、处理器 – 处理器之间的数据传输提供足够的带宽.

在基于 SiP 技术的片上 DWDM 通信方面有许多基础性问题需要解决, 相关研究工作包括: 根据传输状态控制光源强度, 从而降低激光源功耗 [56,57], 降低微环周边温度调节所需的功耗, 减少波长寻址 [58] 中波导和波长数量的预约寻址方法 [59,60], 以 3D 封装技术实现片上光网的物理布局, 以及用重传、校错码、出错信道波长动态重配置等措施提高光网可靠性等.

研究者已经提出许多片上光网的方案, 例如, 波长寻址的片上全光网 λ-Router [61], Snake 和 Folded Crossbar [62], 基于预约寻址的片上全光网 QuT [63] 和 Amon [64], 基于时分复用的片上光网 Corona [65], FlexiShare [66], Channel Borrowing [67] 和 SUOR [68], 兼有电网络和光网络优点的光电混合网络 Meteor [69], Atac [59] 和 Firefly [70] 等. 文献 [71] 对基于 SiP 的 DWDM 片上光网做了相当全面的介绍, 有兴趣的读者可以参阅.

基于硅光子技术的稠密波分复用为高性能互连网提供了新的技术手段, 但该技术的成熟和实用化需要计算机、微电子、光电子领域的科学家的紧密合作.

4.4 并行操作系统

操作系统发展早期面对的是单个处理器、简单 I/O 设备和几个进程的场景, 担负进程、作业、文件、I/O 和用户管理的职责. 随着计算机硬件和运行环境的日益复杂, 现代操作系统已经发展成具有千万行代码, 功能复杂的软件系统. 即便如此, 面对未来 E 级计算机的复杂运行环境, 操作系统仍面临诸多技术问题. 首先, 未来 E 级机的并行规模庞大. 系统是由 5~10 万个节点构成的大规模并行系统, 每个节点本身就是一个包含多个众核处理器的并行系统, 而一个众核处理器内部又包含了几百到几千个计算核, 整个系统会包含上亿个并行工作的计算核, 同时运行着几亿个线程. 其次, E 级机是结构异常复杂的异构系统, 如前所述, 其异构性体现在片内、节点内和节点间等多个层次, 随着面向应用的体系结构及专用加速器的发展, 异构性将更加突出, 而且其存储系统也是多层次、异质的. 如何让海量、多样的计算核心并行协调工作是巨大的挑战. 再次, E 级机上运行的应用是多样的, 除了传统的科学计算外, 大数据和人工智能也日益成为 E 级机必须支持的重要应用. 最后, 超级计算机的使用模式在发生变化, 越来越多的超算中心将以云模式运行, 支撑多种业务.

面对变化的硬件和应用环境, 操作系统的发展必须与时俱进, 特别要关注以下几点.

(1) 节点操作系统要充分考虑众核特点. 操作系统本身代码必须紧凑精简, 去掉不必要的功能, 使代码占用尽可能小的内存空间. 要提高用户线程的加载和切换的速度, 努力降低操作系统的操作对应用程序性能带来的抖动. 要精细管理片内的局存/缓存资源, 减少片外存储器访问. 要提供有效的能耗管理能力. 例如, 操作系统必须了解芯片工艺偏差对处理器性能参数的影响, 为分区电压和频率控制提供支持, 持续监控硬件工作状态、能耗、温度、资源的使用水平等, 根据这些信息调节区域的电压和主频. 日本研究人员配合 A64FX 48C 处理器和 “富岳” 系统的研发, 从几年前就开始研制轻量级的操作系统 McKernel, 并在 “富岳” 系统中得到实际应用 [72], 是众核操作系统的一个成功范例.

(2) 系统级并行操作系统要适应异构体系结构的特征. 资源适配、任务调度要综合考虑异构资源的合理使用、共享资源的竞争以及应用对资源的需求模式等因素. 例如, 根据程序的资源需求和使用模式, 把资源使用行为互补的程序调度到一个节点执行, 避免资源竞争对共同运行的作业造成的性能干扰 [73]. 特别是系统级异构的系统, 操作系统要根据应用对资源使用的模式, 合理配置系统组态, 为程序执行提供最合理的资源环境, 使通用处理器、加速器都得到充分利用, 达到资源利用率和应用性能的最优.

(3) 要适应应用多样化的趋势, 有效支持不同类型的应用. 针对大数据和人工智能等新兴高性能计算应用的特点, 在面向传统的科学计算的软件栈的基础上, 增加新的系统软件功能, 形成能高效支

持大数据和人工智能等不同类型应用的软件栈 [74].

(4) 支持以云模式使用高性能计算机. 通过资源虚拟化的手段提供对超级计算机云模式运行的支持, 同时注重降低虚拟化的开销, 保证应用性能不受影响. 天河二号上的星光超算云平台[13)]是这方面的一个尝试.

4.5 并行编程

并行编程是传统的难题. 首先, 并行程序编程难, 正确性难保证. 程序员要考虑任务的分解、数据集的合理划分、数据依赖关系和数据竞争的表示、共享变量的互斥访问、进程/线程间的同步与通信、并行执行语义的表达等等, 现有的并行编程模型对程序并行性的表达并不直观, 与硬件结构相关的优化更难驾驭. 基于编译技术的串行程序自动并行化 [75~78] 无法完全理解程序的行为, 不能充分发挥并行潜力, 局限性很大. 多数研究者认为自动化并行技术只适用于细粒度的局部并行化或中小规模的简单应用, 不能满足大规模复杂应用并行化的需求. 众核处理器和异构体系结构的出现给并行编程带来新的困难. 众核使每个程序员都必须面对并行编程的任务, 而不像过去仅限于并行计算机的软件开发者. 而异构结构要求程序员了解硬件结构的特点, 发现和定位程序中可被加速器执行的代码部分, 还要处理通用处理器和加速器之间的数据交换和同步.

第二, 并行程序调试难, 错误难以捕捉再现. 由于并行程序执行的状态空间巨大, 且存在不确定性. 多个并行任务 (线程) 被动态地调度执行, 并行任务间的交错执行顺序和任务执行步调都是动态的, 这一过程很难被再次重现. 不确定性还可能源于互斥/同步设置不当所造成的数据竞争, 对于相同的输入, 程序多次重复执行的结果可能不同. 由于程序的运行过程难以重现和追溯, 使错误无法采用串行程序调试中的断点设置、单步执行等调试手段进行定位. 记录/重放 [79] 是并行程序调试的重要手段, 但是众核的引入使得执行状态爆炸式增长, 需要记录的执行轨迹信息量太大, 难以分析. 另外, 异构系统中不同处理器具有不同的异常处理机制, 例如, 在传统同构环境下可以通过操作系统捕获并反馈的页错误, 在异构环境下, 由于不同处理器对页错误的表达和处理机制不同, 操作系统服务难以直接处理该异常.

第三, 并行程序执行行为不稳定, 性能不确定. 在众核系统中通常运行多个应用程序, 应用程序内还包含多个线程, 程序之间、线程之间的相互影响或干扰会对程序的性能产生影响. 例如, 多个线程对共享数据的互斥访问以及线程间同步操作会对程序性能产生影响. 不同程序对缓存/主存、I/O 及网络等资源的访问可能相互干扰, 导致程序的性能难以预测. 不同程序访问不同的数据集合, 可能导致共享缓存中的数据频繁换入/换出, 产生缓存 “颠簸” 效应, 降低 Cache 命中率, 导致性能下降.

不能指望应用开发者都是熟悉体系结构和硬件细节的专家, 要通过并行编程支持系统改善并行编程. 一种思路是以系统的方法支持众核编程, 通过编程模型、语言扩展、并行调试、运行时优化和体系结构支持几方面的综合努力, 改善并行众核编程的效率和性能. 面向 GPU 的软件事务内存 (software transactional memory) 是解决在 GPU 上开发数据动态共享型应用的一种编程模型 [80], 它通过层次结构的访问冲突验证和锁请求时刻的锁排序机制, 获得大规模线程需要的可扩展性, 消除了死锁和活锁, 提高了编程效率. 而且无需修改硬件, 可直接用于商用 GPU. 在商用 GPU 上的实验结果表明, 其执行性能比粗粒度锁机制提高 20 倍. 顺序一致性 (sequential consistency) 是保证程序正确性的基本特性. 传统的以插入同步/栅障指令保证程序顺序一致性的方法对于运行大量线程的众核处理器并不

13) National Supercomputer Center in Guangzhou. Tianhe Star cloud supercomputing platform. http://en.nscc-gz.cn/Product/HighPerformanceComputingService/ServiceCharacteristics.html. [国家超级计算广州中心. 天河星光云超算平台. http://www.nscc-gz.cn/Product/THStarLightMichaelChaIsPlatform/THStarLightPlatformDeadline.html?ColumnId=3160].

适用, 不仅难以保证正确性, 而且影响程序执行的性能. 在众核处理器上以原子代码块作为基本执行单元可以高效实现顺序一致性 [81], 在原子代码块内部, 指令可以乱序执行获得高性能, 而代码块的原子性保证了块的全局全序执行, 维持了顺序一致性, 这种编程模型大大降低了维持顺序一致性的开销, 提高了代码的执行性能. 在体系结构支持的并行调试方面, 适应任意弱一致性模型的硬件冲突记录器是对可确定性记录/重放的基本支持, 难点在于检测并记录弱一致性模型执行中的违反顺序一致性的事件, 并在重放时正确地重现记录的事件. 一种冲突的硬件记录机制 Volition [82] 在原有一致性协议实现中附着一致性检测消息, 可精确检测违反顺序一致性的事件, 在此基础上, 实现了面向总线协议的记录/重放可扩展谱方法 Rainbow [83] 和面向分布式目录协议的记录/重放方法 Pacifier [84], 有效降低了日志的大小和记录硬件的开销, 提高了重放执行的并行度与性能.

并行编程框架是支持并行编程的另一种思路. 并行编程框架的基本理念是程序开发人员不需要了解并行计算机硬件的细节, 也不需要是并行编程和异构编程的专家, 只需专注于按串行方式编写数值计算程序就可快速研制并行应用软件. 所编写的软件通过编程框架可以自动平滑移植到亿亿次、十亿亿次和 E 级高性能计算机, 实现 “一份源码, 多平台高效运行”. 这就是所谓 “并行思考 – 串行编程” 的软件研发模式. 国家 “高性能计算” 重点专项针对结构网格、非结构网格、无网格组合几何计算、有限元计算、非数值图计算等 5 类应用, 分别研制了 JASMIN, JAUMIN, JCOGIN, PHG 和 GRAPHINE 5 个子框架 [85~87], 形成并行编程框架的体系. 框架自底向上分为运行时性能优化层、网格数据模型层、并行算法层、数值算法层和应用编程接口 5 个层次, 提供了领域数据模型、并行算法模板、网格划分、数据通信、负载平衡和运行时工具箱等功能. 框架在神威 · 太湖之光、天河二号和 E 级计算机验证原型系统上部署, 支持了飞机优化设计、大型力学工程、材料科学等领域应用软件的开发.

跨平台的应用软件开发平台是适应异构性的另一种途径. SCIDE 是一个面向国家超级计算基础设施中天河、神威、曙光等多种高性能计算机的应用集成开发环境. 环境与各种计算机之间设置适配层, 以便与不同机器的作业提交系统、任务调度系统、文件系统和运行时系统对接. 在适配层之上提供基础算法库、应用模块库、程序模板库、优化工具库、拖拽式的工作流编排器、适配多种国产处理器的跨异构结点编译环境等. 软件开发人员可以基于模板库的开发向导, 自动生成程序代码框架, 重用基本算法和模块库中的代码, 快速构建应用程序, 并在国家超级计算基础设施的不同机器中运行. 基于 SCIDE, 成功研发了大型飞机流场数值模拟与多学科优化、高能物理格点量子色动力学与分波分析等一批具有自主知识产权的大规模并行应用软件, 并实际应用.

4.6 算法

大规模并行算法是高性能计算机能够发挥作用的关键. 多个国家科技计划都对大规模并行算法研究予以了支持. 国家自然科学基金重大研究计划 “高性能科学计算的基础算法及可计算建模” 历时近 10 年, 在基础算法、可计算建模方法及应用示范方面开展了大量研究工作. 江南计算技术研究所等完成的面向 E 级计算机的共性算法重点项目深入分析了国产 100PF 机和未来 E 级机体系结构特点, 提出在应用问题、数值模拟、计算机体系结构之间建立合理映射的 “关联图及其耦合理论”. 该理论将共性求解算法、物理模型、并行计算机体系结构分别归结为计算网格关联图、数理特性关联图、体系结构关联图, 用于描述它们的分层耦合特性和空间分布的非均衡特性, 从而使物理模型、计算机体系结构的特性传递到共性求解算法. 用计算机体系结构的特性指导系数矩阵分块、计算网格区域分解和算法高效实现. 在关联图理论指导下研究了非线性问题求解共性算法、特征值问题求解共性算法、快速多极子方法等 3 类共性算法的可扩展实现方法, 使 3 类算法在神威 · 太湖之光超级计算机上的可扩展性提升了一个数量级, 运行规模已超过百万核 [88~90].

国家重大基础研究计划 (973 计划) 连续支持了 3 个大规模科学计算的项目. 创新计算方法的基础理论、大规模并行计算、可扩展基础并行算法等方面持续开展基础研究. 研发了自适应结构网格并行应用支撑软件框架和自适应非结构网格并行软件平台. 针对典型领域应用问题, 在复杂流动问题的高性能算法、材料物性的多物理多尺度计算、气候系统模式的高性能算法与应用等方面开展了研究和应用示范.

国家 “高性能计算” 重点专项支持了可计算建模和优化算法的研究. 针对超高精度医学影像重构、核聚变中磁流体不稳定性控制数值模拟、血栓形成机理数值模拟、千米级建筑在随机地震作用下的失效模式识别等 4 个高计算复杂度的应用问题, 从可计算物理模型中提炼出具有共性的二维等谱问题 (PDE 特征值问题的反问题), 总结并诠释了二维等谱问题的内在计算数学性质与规律, 为高性能基础算法的研究提供了新的思路, 为大规模特征值问题的求解开辟了新的研究方向. 在此基础上, 发展了新型并行优化技术和高精度高可扩展的计算方法, 形成了支撑新型可计算物理模型并具有较强泛化能力的基础算法集 [91~93].

普通的软件开发者通过使用并行算法库可降低软件开发的工作量和难度, 提高软件性能. 为使软件开发广泛受益, “高性能计算” 重点专项针对大规模网格生成与区域划分、超大规模稠密/稀疏线性系统直接求解、可扩展并行预条件迭代方法、粒子类问题的可扩展并行算法等问题, 研发了适应体系结构特点的高效并行算法库, 并在国内三台 E 级原型机上部署. 大规模线性系统直接求解算法的并行性突破 60 万核, 达到 50% 的并行效率. 粒子输运程序 OpenSn 实现 100 万核可扩展计算, 并行效率 80% 以上.

算法是学科高度交叉的研究领域, 需要数学家和领域科学家专注于新的计算模型, 设计数值计算方法, 需要计算机科学家和软件工程师在并行计算机上高效地实现这些算法. 一个优秀的算法可能比昂贵的硬件加速器更为有效. 算法的设计和优化与计算机硬件结构密切相关, 因此, 算法的设计与实现者必须要和计算机硬件的开发者密切配合, 理解并合理利用特定硬件结构的特点, 提高算法的性能. 大规模并行算法的研究将始终伴随我国高端计算系统的研发, 发展共性的数学建模方法, 实现适应体系结构特点的高性能算法, 形成国产超级计算机上的基础算法库, 使国产超级计算机在应用中体现更大价值.

4.7 可靠性

可靠性决定 E 级计算的成败. 未来的 E 级计算机将包含 5~10 万个节点, 按目前的估计, 功耗将超过 30 MW, 大电流的浪涌和高组装密度系统的散热问题十分突出. 根据可靠性理论计算, 其平均无故障时间 (MTBF) 甚至可能低于一个小时. 而在 E 级机上运行的大规模数值模拟有些要连续运行几十个小时甚至几十天. 如何保证长时间不间断运行的应用在 E 级规模的计算系统上的稳定可靠运行是迫切需要解决的问题.

提高系统的可靠性要依靠软硬结合的手段, 从多个侧面、多个层次入手. 基本的思路是如何让系统少出错, 出了错能继续工作, 能自动从错误中恢复.

少出错首先依赖于硬件的可靠, 从可靠性设计、元器件筛选、制造工艺控制、质量管控到系统的测试, 每个阶段都要把关. 例如, 内存和数据通路均采用误差校验. 其次, 要保证良好的系统工况, 包括高稳定度低纹波电源、高效的系统冷却等. 为了使系统工作在理想的温度下, 曙光 E 级验证原型机采用了沉浸式液冷技术[14], 电路板浸泡在密封的冷媒之中, 低温蒸发的冷媒高效地带走电路板散发的

14) Sugon X86 supercomputer prototype: liquid cooling, peak performance. [3.18PFLOPS. 曙光国产 X86 超算原型曝光: 液冷散热 3180 万亿次性能. https://www.cnbeta.com/articles/tech/865797.htm].

热量, 使芯片结温保持在很低的水平, 有效提升了系统长时间运行的可靠性.

要让系统出了错能继续工作可以依靠冗余容错机制, 但是对于 E 级规模的系统, 全面冗余的代价不可接受, 最多只能是核心部件冗余. 更为现实的做法是在软件配合下, 使系统具有降级运行的能力. 出了故障只是使性能降低, 但系统运行不停顿. 为此, 需要精准及时的错误检测、定位和隔离能力.

系统从错误状态的自动恢复涉及硬件和软件两方面的措施. 例如, 隔离故障节点, 将原来在故障节点的任务迁移到备份节点, 然后将备份节点加入系统, 恢复系统的正常功能和性能. 检查点技术是维持程序长时间运行的另一种技术 [94]. 系统周期性地在检查点保存程序执行的现场, 一旦检测到程序出错, 就回滚到前一个检查点, 恢复保存的现场并从该检查点重新执行. 但是在 E 级规模的系统中, 系统级的检查点现场保护开销太大, 更为实际的是由应用程序自己设置检查点以及需要保护的现场内容, 这样能够控制开销.

此外, E 级系统还会面临一些在小规模系统中不会出现的可靠性问题. 例如, 当程序在调试中出现错误时, 需要区分究竟是由于程序代码编写的错误, 还是系统硬件的瞬时故障造成的. 小规模系统的 MTBF 足够长, 一般不会在调试程序时出现硬件故障, 而 E 级系统很短的 MTBF 和较长的程序调试时间, 就可能带来这个新问题.

总之, 必须对 E 级规模系统的可靠性有清醒的认识, 从硬件、系统软件、算法和应用软件多个层面解决可靠性问题.

5 结语

本文针对 E 级计算机研制所面临的重大技术挑战, 提出了 7 个方面的研究问题, 希望这些问题的研究有助于应对技术挑战, 促进 E 级或后 E 级计算系统的研发. E 级计算机的成功不仅将为重大应用提供利器, 还将推动计算技术的进步, 这是过去几十年来一直在发生的事情. 当前, 我国 E 级计算机的研制工作正处于关键时刻, 然而, 从历史的角度看, 这只不过是计算机发展过程中的一个阶段. 人类追求新的科学发现和技术进步的脚步永远不会停歇, 因此, 超级计算机的发展也将随之持续下去.

在解决不断产生的技术问题的同时, 要特别关注高性能计算应用的生态环境问题, 特别是在我国高性能计算的发展受到外部封锁和遏制的情况下, 如何通过加强基础研究, 提高我国在该领域技术积累的深度和广度, 实现我国超级计算的可持续发展, 这是必须回答的问题.

最后, 大数据和人工智能的兴起既给高性能计算带来挑战, 也带来新的难得的机遇. 需要以科学的态度和求实的精神, 分析新需求, 研究新问题, 使我国的高性能计算迈上新台阶.

参考文献

1 “Yinhe” 100 Mega-scale super computer system is sucessfully developed. Comput Eng Sci, 1984, 1: 137 [“银河” 亿次巨型计算机系统在我校研制成功并通过国家鉴定. 计算机工程与科学. 1984 , 1: 137]

2 Si H W, Feng L S. The development of the first supercomputer YH-1 in China and its inspiration. Studies History Natural Sci, 2017, 36: 563–580 [司宏伟, 冯立昇. 中国第一台亿次巨型计算机 “银河 -I” 研制历程及启示. 自然科学史研究, 2017, 36: 563–580]

3 Li G J, Chen H A, Fan J P, et al. Dawning-1 parallel computer. Chin J Comput, 1994, 17: 882–889 [李国杰, 陈鸿安, 樊建平, 等. 曙光一号并行计算机. 计算机学报, 1994, 17: 882–889]

4 Sun N H, Liu H, Liu W Z, et al. The design of system software of dawning-1000 massively parallel processing system. Chin J Comput, 1997, 20: 259–268 [孙凝晖, 刘宏, 刘文卓, 等. 曙光 1000 大规模并行计算机系统软件的设计. 计算机学报, 1997, 20: 259–268]

5 Sun N H, Meng D. Dawning4000A high performance computer. Front Comput Sci, 2007, 1: 20–25

6 Zhu M F, Xiao L M, Ruan L, et al. DeepComp: towards a balanced system design for high performance computer systems. Front Comput Sci China, 2010, 4: 475–479

7 Yu Y, Zhang Y Q, Wang T, et al. Early performance evaluation of Dawning 5000A and DeepComp 7000. In: Proceedings of the 15th IEEE International Conference on Parallel and Distributed Systems, Shenzhen, 2009. 578–585

8 Yang X, Liao X, Xu W, et al. TH-1: China's first petaflop supercomputer. Front Comput Sci China, 2010, 4: 445–455

9 Li Q, Li B, Huo Z, et al. Design and implementation of communication system of the Dawning 6000 supercomputer. Front Comput Sci China, 2010, 4: 466–474

10 Yang X J, Liao X K, Lu K, et al. The TianHe-1A supercomputer: its hardware and software. J Comput Sci Technol, 2011, 26: 344–351

11 Calamia J. China's homegrown supercomputers. IEEE Spectrum, 2012, 49: 60–62

12 Qian D P, Luan Z Z. High performance computing development in china: a brief review and perspectives. Comput Sci Eng, 2019, 21: 6–16

13 Moore G E. Cramming more components onto integrated circuits. Electronics, 1965, 38: 114–117

14 Dennard R H, Gaensslen F H, Yu H N, et al. Design of ion-implanted MOSFET's with very small physical dimensions. IEEE J Solid-State Circ, 1974, 9: 256–268

15 Amdahl G M. Validity of the single-processor approach to achieving large-scale computing capabilities. In: Proceedings of the AFIPS '67 Spring Joint Computer Conference, Atlantic City, 1967. 483–485

16 Gustafson J L. Reevaluating Amdahl's law. Commun ACM, 1988, 31: 532–533

17 Wulf W A, McKee S A. Hitting the memory wall: implications of the obvious. SIGARCH Comput Architect News, 1995, 23: 20–24

18 Horowitz M. Computing's energy problem. In: Proceedings of IEEE International Solid-State Circuits Conference (ISSCC), 2014. 57: 10–14

19 Vazhkudai S S, de Supinski B R, Bland A S, et al. The design, deployment, and evaluation of the CORAL pre-exascale systems. In: Proceedings of the International Conference for High Performance Computing, Networking, Storage, and Analysis (SC'18), Dallas, 2018. 52: 1–12

20 Fu H H, Liao J F, Yang J, et al. The Sunway TaihuLight supercomputer: system and applications. Sci China Inf Sci, 2016, 59: 072001

21 Qian D P. China's effort on exascale computing: current status and perspectives. In: Proceedings of International Conference for High Performance Computing, Networking, Storage, and Analysis (SC'18), Dallas, 2018

22 Cugola G, Margara A. Processing flows of information: from data stream to complex event processing. ACM Comput Surv, 2012, 44: 1–62

23 Becker T, Burovskiy P, Nestorov A M, et al. From exaflop to exaflow. In: Proceedings of the Conference on Design, Automation & Test in Europe Conference & Exhibition (DATE'17), Lausanne, 2017. 404–409

24 Kaplan K R, Winder R O. Cache-based computer systems. IEEE Comput, 1973, 6: 30–36

25 Liptay J S. Structural aspects of the system/360 model 85: II the cache. IBM Syst J, 1968, 7: 15–21

26 Power J, Basu A, Gu J L, et al. Heterogeneous system coherence for integrated CPU-GPU systems. In: Proceedings of the 46th Annual IEEE/ACM International Symposium on Microarchitecture, Davis, 2013. 457–467

27 Martin M K, Hill M D, Wood D A. Token coherence: decoupling performance and correctness. In: Proceedings of the 30th International Symposium on Computer Architecture (ISCA'03), San Diego, 2003. 182–193

28 Wang H, Wang R, Luan Z Z, et al. Improving multiprocessor performance with fine-grain coherence bypass. Sci China Inf Sci, 2015, 58: 012104

29 Iyer S S, Kalter H L. Embedded DRAM technology: opportunities and challenges. IEEE Spectr, 1999, 36: 56–64

30 Iyer S S, Barth J E, Parries P C, et al. Embedded DRAM: technology platform for the blue Gene/L chip. IBM J Res Dev, 2005, 49: 333–350
31 Ghose S, Hsieh K, Boroumand A, et al. Enabling the adoption of processing-in-memory: challenges, mechanisms, future research directions. 2018. ArXiv: 1802.00320
32 Zhou M, Prodromou A, Wang R, et al. Temperature-aware DRAM cache management -relaxing thermal constraints in 3D systems. IEEE Trans Comput-Aided Des Integr Circ Syst, 2019. doi: 10.1109/TCAD.2019.2927528
33 Wolf S A, Lu J, Stan M R, et al. The promise of nanomagnetics and spintronics for future logic and universal memory. Proc IEEE, 2010, 98: 2155–2168
34 Hennessy J L, Patterson D A. A new golden age for computer architecture. Commun ACM, 2019, 62: 48–60
35 Chien A. Technology scaling and the future of microprocessors: the 10×10 approach. 2012. http://i2pc.cs.illinois.edu/seminars.html
36 Chang L, Frank D J, Montoye R K, et al. Practical strategies for power-efficient computing technologies. Proc IEEE, 2010, 98: 215–236
37 Dreslinski R G, Wieckowski M, Blaauw D, et al. Near-threshold computing: reclaiming Moore's law through energy efficient integrated circuits. Proc IEEE, 2010, 98: 253–266
38 Ghasemi H R, Sinkar A, Schulte M, et al. Cost-effective power delivery to support per-core voltage domains for power-constrained processors. In: Proceedings of the 49th Annual Design Automation Conference (DAC'12), San Francisco, 2012. 56–61
39 Ansari A, Mishra A, Xu J, et al. Tangle: route-oriented dynamic voltage minimization for variation-afflicted, energy-efficient on-chip networks. In: Proceedings of the 20th IEEE International Symposium on High Performance Computer Architecture (HPCA'14), Orlando, 2014. 440–451
40 Torrellas J. Extreme-scale computer architecture. National Sci Rev, 2016, 3: 19–23
41 Kogge P, Borkar S, Campbell D, et al. ExaScale Computing Study: Technology Challenges in Achieving Exascale Systems. DARPA-IPTO Sponsored Study, 2008
42 Feautrier P. Some efficient solutions to the affine scheduling problem. I. one-dimensional time. Int J Parallel Prog, 1992, 21: 313–347
43 Smith B. Architecture and applications of the HEP multiprocessor computer system. In: Proceedings of SPIE, 1982. 241–248
44 Chen T S, Du Z D, Sun N H, et al. DianNao: a small-footprint high-throughput accelerator for ubiquitous machine-learning. In: Proceedings of the 19th International Conference on Architectural Support for Programming Languages and Operating Systems (ASPLOS'14), Salt Lake City, 2014. 269–284
45 Chen Y J, Chen T S, Xu Z, et al. DianNao family: energy-efficient hardware accelerators for machine learning. Commun ACM, 2016, 59: 105–112
46 Liu S L, Du Z D, Tao J H, et al. Cambricon: an instruction set architecture for neural networks. In: Proceedings of the 43rd ACM/IEEE Annual International Symposium on Computer Architecture (ISCA 2016), Seoul, 2016. 393–405
47 Jouppi N P, Young C, Patil N, et al. In-datacenter performance analysis of a tensor processing unit. In: Proceedings of the 44th Annual International Symposium on Computer Architecture (ISCA'17), Toronto, 2017. 1–12
48 Merolla P A, Arthur J V, Alvarez-Icaza R, et al. A million spiking-neuron integrated circuit with a scalable communication network and interface. Science, 2014, 345: 668–673
49 Davies M, Srinivasa N, Lin T H, et al. Loihi: a neuromorphic manycore processor with on-chip learning. IEEE Micro, 2018, 38: 82–99
50 Imam N, Cleland T A. Rapid online learning and robust recall in a neuromorphic olfactory circuit. Nat Mach Intell, 2020, 2: 181–191
51 Nai L F, Hadidi R, Sim J, et al. GraphPIM: enabling instruction-level PIM offloading in graph computing frameworks.

In: Proceedings of the IEEE International Symposium on High Performance Computer Architecture (HPCA'17), Austin, 2017. 457–468

52 Ahn J W, Hong S P, Yoo S, et al. A scalable processing-in-memory accelerator for parallel graph processing. In: Proceedings of the 42nd Annual International Symposium on Computer Architecture (ISCA'15), Portland, 2015. 105–117

53 Zhuo Y W, Wang C, Zhang M X, et al. GraphQ: scalable PIM-based graph processing. In: Proceedings of the 52nd Annual IEEE/ACM International Symposium on Microarchitecture (MICRO'19), Columbus, 2019. 712–725

54 Ham T J, Wu L, Sundaram N, et al. Graphicionado: a high-performance and energyecient accelerator for graph analytics. In: Proceedings of the 49th Annual IEEE/ACM International Symposium on Microarchitecture, Taipei, 2016. 1–13

55 Ophir N, Mineo C, Mountain D, et al. Silicon photonic microring links for high-bandwidth-density, low-power chip I/O. IEEE Micro, 2013, 33: 54–67

56 Kurian G, Sun C, Chen C H O, et al. Cross-layer energy and performance evaluation of a nanophotonic manycore processor system using real application workloads. In: Proceedings of the 26th International Parallel and Distributed Processing Symposium (IPDPS'12), Shanghai, 2012. 1117–1130

57 Thakkar I G, Chittamuru S V R, Pasricha S. Run-time laser power management in photonic NoCs with on-chip semiconductor optical amplifiers. In: Proceedings of the 10th IEEE/ACM International Symposium on Networks-on-Chip (NoCS'16), Nara, 2016. 1–4

58 Haurylau M, Chen G Q, Chen H, et al. On-chip optical interconnect roadmap: challenges and critical directions. IEEE J Sel Top Quantum Electron, 2006, 12: 1699–1705

59 Anders M A. High-performance energy-efficient NoC fabrics: evolution and future challenges. In: Proceedings of the 8th IEEE/ACM International Symposium on Networks-on-Chip (NoCS'14), Ferrara, 2014

60 Werner S, Navaridas J, Luján M. Efficient sharing of optical resources in low-power optical networks-on-chip. J Opt Commun Netw, 2017, 9: 364–374

61 Li H, Fourmigue A, Le Beux S, et al. Towards maximum energy efficiency in nanophotonic interconnects with thermal-aware on-chip laser tuning. IEEE Trans Emerg Top Comput, 2018, 6: 343–356

62 Ramini L, Grani P, Bartolini S, et al. Contrasting wavelength-routed optical NoC topologies for power-efficient 3D-stacked multicore processors using physical-layer analysis. In: Proceedings of the Conference on Design, Automation and Test in Europe, Grenoble, 2013. 1589–1594

63 Cao R, Wang K, Gu H, et al. A crosstalk-aware wavelength assignment method for optical network-on-chip. IEICE Electron Express, 2016, 13: 20160821

64 Werner S, Navaridas J, Lujan M. Amon: an advanced mesh-like optical NoC. In: Proceedings of the 23rd IEEE Annual Symposium on High-Performance Interconnects (HOTI'15), Santa Clara, 2015. 52–59

65 Vantrease D, Schreiber R, Monchiero M, et al. Corona: system implications of emerging nanophotonic technology. In: Proceedings of the 35th International Symposium on Computer Architecture (ISCA'08), Beijing, 2008. 153–164

66 Pan Y, Kim J, Memik G. Flexishare: channel sharing for an energy-efficient nanophotonic crossbar. In: Proceedings of the 16th International Conference on High-Performance Computer Architecture (HPCA'10), Bangalore, 2010. 1–12

67 Xu Y, Yang J, Melhem R. Channel borrowing: an energy-efficient nanophotonic crossbar architecture with light-weight arbitration. In: Proceedings of the International Conference on Supercomputing(ICS'12), Venice, 2012. 133–142

68 Wu X Q, Xu J, Ye Y Y, et al. SUOR: sectioned undirectional optical ring for chip multiprocessor. J Emerg Technol Comput Syst, 2014, 10: 1–25

69 Kirman N, Kirman M, Dokania R K, et al. Leveraging optical technology in future bus-based chip multiprocessors. In: Proceedings of the 39th Annual IEEE/ACM International Symposium on Microarchitecture, Orlando, 2006. 492–503

70 Pan Y, Kumar P, Kim J, et al. Firefly: illuminating future network-on-chip with nanophotonics. In: Proceedings of the 36th International Symposium on Computer Architecture (ISCA'09), Austin, 2009. 429–440

71 Werner S, Navaridas J, Luján M. A survey on optical network-on-chip architectures. ACM Comput Surv, 2018, 50: 1–37

72 Gerofi B, Takagi M, Hori A, et al. On the scalability, performance isolation and device driver transparency of the IHK/McKernel hybrid lightweight kernel. In: Proceedings of the IEEE International Parallel and Distributed Processing Symposium (IPDPS'16), Chicago, 2016. 1041–1050

73 Zhang L D, Liu Y, Wang R, et al. Lightweight dynamic partitioning for last-level cache of multicore processor on real system. J Supercomput, 2014, 69: 547–560

74 Reed D A, Dongarra J. Exascale computing and big data. Commun ACM, 2015, 58: 56–68

75 Kulkarni M, Pingali K, Walter B, et al. Optimistic parallelism requires abstractions. In: Proceedings of the ACM SIGPLAN 2007 Conference on Programming Language Design and Implementation, San Diego, 2007. 211–222

76 Kulkarni M, Pingali K, Ramanarayanan G, et al. Optimistic parallelism benefits from data partitioning. In: Proceedings of the 13th International Conference on Architectural Support for Programming Languages and Operating Systems (ASPLOS'08), Seattle, 2008. 233–243

77 Kulkarni M, Burtscher M, Inkulu R, et al. How much parallelism is there in irregular applications? In: Proceedings of the 14th ACM SIGPLAN Symposium on Principles and Practice of Parallel Programming (PPoPP'09), Raleigh, 2009. 3–14

78 Bauer M, Clark J, Schkufza E, et al. Programming the memory hierarchy revisited: supporting irregular parallelism in sequoia. In: Proceedings of the 16th ACM SIGPLAN Symposium on Principles and Practice of Parallel Programming (PPoPP'11), San Antonio, 2011. 13–24

79 Gao L, Wang R, Qian D P. Deterministic replay for parallel programs in multi-core processors. J Softw, 2013, 24: 1390–1402 [高岚, 王锐, 钱德沛. 多核处理器并行程序的确定性重放研究. 软件学报, 2013, 24: 1390–1402]

80 Xu Y L, Wang R, Goswami N, et al. Software transactional memory for GPU architectures. In: Proceedings of the 12th Annual IEEE/ACM International Symposium on Code Generation and Optimization (CGO'14), Orlando, 2014. 1–10

81 Qian X H, Torrellas J, Sahelices B, et al. BulkCommit: scalable and fast commit of atomic blocks in a lazy multiprocessor environment. In: Proceedings of the 46th Annual IEEE/ACM International Symposium on Microarchitecture, Davis, 2013. 371–382

82 Qian X H, Sahelices B, Torrellas J, et al. Volition: precise and scalable sequential consistency violation detection. In: Proceedings of the 18th ACM International Conference on Architectural Support for Programming Languages and Operating Systems (ASPLOS'13), Houston, 2013. 535–548

83 Qian X H, Huang H, Sahelices B, et al. Rainbow: efficient memory dependence recording with high replay parallelism for relaxed memory model. In: Proceedings of the 19th IEEE International Symposium on High-Performance Computer Architecture (HPCA'13), Shenzhen, 2013. 554–565

84 Qian X H, Sahelices B, Qian D P. Pacifier: record and replay for relaxed-consistency multiprocessors with distributed directory protocol. In: Proceedings of the 41st ACM/IEEE International Symposium on Computer Architecture(ISCA'14), Minneapolis, 2014. 433–444

85 Mo Z Y, Zhang A Q, Cao X L, et al. JASMIN: a parallel software infrastructure for scientific computing. Front Comput Sci China, 2010, 4: 480–488

86 Liu Q K, Zhao W B, Cheng J, et al. A programming framework for large scale numerical simulations on unstructured mesh. In: Proceedings of the IEEE International Conference on High Performance and Smart Computing (HPSC'16), New York, 2016. 310–315

87 Liu Q K, Mo Z Y, Zhang A Q, et al. JAUMIN: a programming framework for large-scale numerical simulation on

unstructured meshes. CCF Trans HPC, 2019, 1: 35–48

88 Wang W, Wang S Y, Jiang J R, et al. Implementation and optimization of fast multipole method on Sunway manycore processors. Computer Engineering & Science, 2019, 41: 1161–1167 [王武, 王舒扬, 姜金荣, 等. 快速多极子方法在申威众核处理器上的实现和优化. 计算机工程与科学, 2019, 41: 1161–1167]

89 Zhang H R, Cao J W. A combinational efficient algorithm for elliptic eigenvalue problems. J Algor Comput Technol, 2016, 10: 1–12

90 Yu T Y, Zhao Y H, Zhao L. Optimize a preconditioned block iterative eigensolver on sunway MAC. J Numerical Methods Comput Appl, 2019, 40: 291–309 [于天禹, 赵永华, 赵莲. 基于神威太湖之光架构的 LOBPCG 并行算法研究. 数值计算与计算机应用, 2019, 40: 291–309]

91 Jiang Y, Li S, Xu Y S. A higher-order polynomial method for SPECT reconstruction. IEEE Trans Med Imag, 2019, 38: 1271–1283

92 Wu K, Tang H. On physical-constraints-preserving schemes for special relativistic magnetohydrodynamics with a general equation of state. Z Angew Math Phys, 2018, 69: 84

93 Tang T, Wang L L, Yuan H F, et al. Rational spectral methods for PDEs involving fractional laplacian in unbounded domains. SIAM J Sci Comput, 2020, 42: 585–611

94 Boito F Z, Inacio E C, Bez J L, et al. A checkpoint of research on parallel I/O for high-performance computing. ACM Comput Surv, 2018, 51: 1–35

Key issues in exascale computing

Depei QIAN* & Rui WANG

School of Computer Science and Engineering, Beihang University, Beijing 100191*, China*
* Corresponding author. E-mail: depeiq@buaa.edu.cn

Abstract Over the past several decades, high performance computing (HPC) in China has undergone tremendous growth under the continuous support of national research programs. The development of exascale computers is the current goal set by the National Key R&D Project on HPC. Starting with a brief historical review of China's HPC development, this article analyzes the major challenges encountered in developing exascale computers. Thereafter, some important issues in realizing exascale computing are discussed, including architecture, processor, interconnect, parallel system software, parallel programming, algorithm, and resilience.

Keywords exascale computer, heterogeneous architecture, many-core processor, interconnect, parallel programming

Depei QIAN is a professor at Beihang University. He has served as a member of the expert group and committee of the National High-Tech Research & Development Program (the 863 Program) in information technology since 1996. Currently, he is the chief scientist of the National Key R&D Project on high performance computing. He has been working on computer architecture and networks for many years. His current research interests include high performance computer architecture and implementation technologies, distributed systems, network computing, and multicore/manycore programming.

Rui WANG is an assistant professor in the School of Computer Science and Engineering, Beihang University. He received his B.S. and M.S. degrees in Computer Science from Xi'an Jiaotong University and his Ph.D. degree in Computer Science from Beihang University. His research interests involve domain-specific computer systems and architecture.