并行與分布式計算通訊

地　址：武汉市华中科技大学东五楼二楼
邮　编：430074
电　话：（027）87541924 或 87543529
传　真：（027）87557354
E-mail：liuqin19@hust.edu.cn
网　址：http://grid.hust.edu.cn

SCTS 服务计算技术与系统教育部重点实验室

CGCL 集群与网格计算湖北省重点实验室

并行與分布式計算通訊

BING XING YU FEN BU SHI JI SUAN TONG XUN

2019年第3期　总第38期　2019年09月

封面人物：杜亚娟——不忘初心，砥砺奋进

2018-2019年度实验室各研究方向成果展示

源代码漏洞分析数据处理

科研就像与大师有约

http://grid.hust.edu.cn

2019 暑期年会照片

2018-2019年度
实验室各研究方向成果展示

金秋时节，硕果飘香！华中科技大学“大数据技术与系统国家地方联合工程研究中心”、“服务计算技术与系统教育部重点实验室”暨“集群与网格计算湖北省重点实验室”、“湖北省大数据安全工程技术研究中心”在老师和同学们的齐心协力下，百尺竿头、更进一步，以喜人的成绩踏入新学年的征程。值此丰收之季，本季刊对实验室过去一年所获得的成果进行回顾与总结。

在分布式系统方面，长期致力于虚拟化、数据中心、边缘计算与云端融合等领域研究。研究成果包括：1）在分布式系统软件层，设计实现了容器网络流量控制系统、容器在线迁移系统和容器启动加速系统，解决容器网络恶性竞争、快速迁移、冷启动性能瓶颈。2）在分布式基础架构层，实现了基于DPDK和mTCP用户态协议栈的动态按需组包卸载与批处理系统，快速改善Memcached在突发流量下的性能。在考虑端到端延迟和各种成本的情况下，提出了不依赖数学模型的优化方法来协调VNF并调度网络流以最大化整体网络效用。另外，设计了独辟蹊径的高效能数据中心混合液冷系统，通过温水冷却协同廉价普适的半导体制冷器件、以及软硬结合的细粒度控制方法，将主要耗能部件的局部能效比指标PUE降至国际先进水平。3）在分布式应用层，探索为不同区块链系统上的数据建立高效可信连接；利用无线感知技术实现价格低、易扩展、易部署的交通状态测量；提出了基于深度卷积神经网络的检测模型，从医学图像中提取特征实现对微小出血病灶的精准检测。

在系统软件与体系结构方面，主要致力于利用新型处理器和存储器件为上层新型应用提供有效支撑平台。研究成果包括：1）在内存计算方面，研发了异构内存大页处理系统，降低大内存地址转换开销并提升系统性能。2）在图计算体系结构方面，研发了基于订制缓存的高性能图计算系统，在FPGA-DRAM平台上采用两级定制cache以及传输-计算重叠策略，同时设计了基于3D-ReRAM的图计算加速器，充分利用其高并行性加快图处理。3）在运行时系统方面，提出了依赖关联性感知的图处理执行机制，加快GPU上图顶点状态传递速度和减少数据访问开销；采用基于无效更新过滤的图计算加速策略减少数据处理量；同时研发了面向图计算的高层综合系统，简化图分析任务的编程。4）在应用支撑方面，提出了内存重用及优化方法和分层细粒度深度神经网络并行优化方法，加快深度学习过程；提出了异构内存感知的并行图计算系统，通过划分图数据和任务来避免多个处理器的数据竞争、原子更新带来的额外开销、以及多核更新数据对预取效果的影响。

在网络空间安全方面，围绕云计算安全、软件安全、数据与隐私安全三方面做出了研究成果：1）在云计算安全方面，设计了自动感知容器云中更新事件的框架，以快速评估主动防御技术的有效性变化。针对传统粗粒度、上下文不敏感的控制流完整性方法可能被攻击的问题，设计了上下文敏感的增量式控制流完整性方法，在引入较低运行时开销的情况下抵御常见的控制流劫持攻击。2）在软件安全方面，提出了结合程序静态分析和模糊测试的轻量级并发漏洞检测框架，增大了漏洞被触发的概率。3）在数据与隐私安全方面，设计了基于区块链的安全数据交易生态系统。针对传感器设备采集的海量数据，提出了适用于云辅助无线传感器网络的轻量级可搜索公钥加密方案，用于保护和高效访问云中的传感器数据。此外，针对云上以压缩格式存放的多媒体数据，提出了同态加密FLAC音频密文下的解压缩技术，提高了解压缩效率和降噪效率。

在大数据方面，基于大数据基础理论，研究大数据的管理、处理、分析等技术，为大数据的广泛高效应用提供支撑。研究成果包括：1）在大数据理论方面，提出了基于博弈论的准流图划分方法，同时也关注算法体系结构协同设计方面，提出了图流中针对节点的链路预测算法。2）在大数据处理方面，研究了高时效大数据流连接处理系统，研发了基于动态负载监测的高性能实时分布式流连接系统，以及面向二路等值连接的高效可扩展实时分布式流连接系统，同时提出了基于生命周期的内存计算垃圾回收优化器，并开展了基于动态预取策略的容器写时复制机制优化研究。3）在大数据管理方面，针对大数据加密抗隐私泄露，提出了面向加密数据的隐私保护多关键字Top-k相似性搜索算法。4）在大数据分析方面，提出了基于社交距离感知贝叶斯个性化排序推荐算法，以及基于语义匹配的知识问答系统。

立足当下，展望未来！更高的平台和起点，也意味着更重的担当和更大的动力，需要我们不忘初心、砥砺前行。伴随新学年而注入实验室的新鲜血液，也为实验室的发展带来了新的活力。在此，衷心祝愿实验室全体师生在新学年取得新突破！

刘方明

二〇一九年九月

主　编：金　海
本期执行主编：刘方明
编　委：陈汉华、代炜琦、丁晓锋、耿　聪、顾　琳、胡　侃、华强胜、黄　宏、蒋文斌、廖小飞、刘方明、刘海坤、刘英书、陆　枫、马晓静、羌卫中、邵志远、石宣化、王多强、吴　松、肖　江、谢　夏、徐　鹏、叶晨成、余　辰、袁　斌、袁平鹏、张　腾、张　宇、章　勤、赵　峰、郑　龙、郑　然、邹德清
（按姓氏拼音排序）
责任编辑：刘　芹

地　址：武汉市华中科技大学东五楼二楼
邮　编：430074
电　话：（027）87541924 或 87543529
传　真：（027）87557354
E-mail：liuqin19@hust.edu.cn
Homepage：http://grid.hust.edu.cn

（此刊仅供内部交流学习）

卷首语
…… 1

热点
…… 3

封面人物
不忘初心，砥砺奋进……杜亚娟　7

专栏
分布式系统组典型成果介绍
……周　放，戴小海，郝威峰，陈彩娟
顾　琳，陆　枫，余　辰，肖　江，刘方明，吴　松　9
系统软件与体系结构组典型成果介绍
……刘海坤，李若时
张　宇，郑　龙，马　阳，张杨松，蒋文斌，廖小飞　18
网络空间安全组典型成果介绍
……邹德清，马晓静，徐　鹏，羌卫中，代炜琦，袁　斌　30
大数据组典型成果介绍
华强胜，赵　峰，陈汉华，石宣化，丁晓锋，袁平鹏，谢　夏　38

声音
OSv 中的线程调度模型……何　轲　49
浅谈混合内存页面管理机制……彭甘为　50
深入浅出解析 RDMA……谭　杰　52
源代码漏洞分析数据处理……唐　静　55

动态
实验室 2019 年暑期年会召开……刘　芹　57

推荐
vNIDS: Towards Elastic Security with Safe and Efficient Virtualization of Network Intrusion Detection Systems
……陈浩宇　推荐　58
TEEv: Virtualizing Trusted Execution Environments on Mobile Platforms……罗瑞坤　推荐　60
ScatterCache: Thwarting Cache Attacks viaCache Set Randomization……张　盼　推荐　62

交流
科研就像与大师有约……费新财　64

迎接5G数据爆发时代

（姜志浩 整理）

随着信息技术和人类生产生活不断融合，互联网快速普及，全球数据呈现爆发增长，这对经济发展、社会治理、国家管理、人民生活都产生了重大的影响。当前全球大数据正进入加速发展的时期，技术产业应用创新不断迈向新高度。

近日，国家统计局与联合国经济和社会事务部将在杭州建立大数据研究所，这是联合国机构首次与相关国家合作建立大数据研究所。这一项目能够充分发挥中国地方政府和企业在大数据领域的先发和引领优势，重点聚焦新技术、新产业、新业态，更好服务于经济和社会发展。

权威国际研究机构预测，到2020年，中国数据总量将会超过8万亿GB，占全球数据总量的20%左右。我国正在成为数据量最大、数据类型最丰富的国家之一，这样规模化的优势为创新发展提供了关键要素。由此可见，我国大数据发展也正在快车道上不断前行。

数据是最有价值的战略资源

众所周知，大数据具有容量大、类型多、存取速度快、应用价值高等主要特征。大数据的集合过程，就是对来源分散、数量巨大、格式多样的数据进行采集、存储，并进行关联分析，从而发现新知识、创造新价值、提升新能力。大数据时代，正是从“人人互联”逐步延展至“人机交互”“物物相联”，发展空间无限广大。

工业和信息化部总经济师王新哲在2019大数据产业峰会上指出，大数据是新时代的“数字宝矿”，是当今世界最有价值的战略资源。大数据与5G、云计算、人工智能、工业互联网等新一代信息通信技术结合，从根本上改变了经济发展的模式，重塑了全球产业链分工。近年来，我国大数据产业蓬勃发展，产业规模日益壮大，融合应用不断深化，数字经济量质齐升，对经济社会的创新驱动、融合带动作用显著增强。

作为产业发展主管部门，工业和信息化部将扎实推进大数据战略，深入推动大数据与实体经济融合发展，加快发展壮大以数据为关键要素的数字经济，为促进我国经济实现高质量发展提供重要支撑和保障。具体而言，工信部将重点做好6个方面的工作：一是完善网络设施，夯实发展基础；二是推动协同创新，增强发展活力；三是深化融合应用，拓宽发展领域；四是构建支撑体系，优化发展环境；五是增强安全保障，促进健康发展；六是加强国际合作，推动开放发展。

大力开发数据资源，赋能各行各业

随着大数据与实体经济的融合更加深入，精确营销、智能推荐、应用信用评价等不断普及，个性化定制趋势明显，智慧医疗、智能交通等新模式、新业态不断涌现，为培育新兴产业提供了广阔的空间。而跨行业、跨领域的大数据资源开发，唤醒了越来越多“沉睡”的数据，开发出层出不穷的新数据。

我国大数据产业的发展离不开政产学研用各方通力配合，一方面，大数据服务于企业，海量的数据已经成为企业最具价值的财富，发展“数字经济”已成共识，大数据技术的应用场景也越来越广泛。比如电商巨头阿里巴巴，从市场营销到平台设计，从市场预测到决策支持，从效能提升到运营管理，从云计算到人工智能，都在使用大数据发挥重要的支撑作用。另一方面，大数据开始蔓延到社会的各个领域，让城市越来越智慧，协助解决交通、消防、警务、医疗、城管等群众最关切的“难点”“痛点”。

数据的开发只是第一步，如何妥善保存和利用数据才是建设数字中国的关键所在。目前，社会各界对数据应用背后的管理还不够重视，数据质量低、数据“孤岛”普遍存在，数据安全管理不到位，数据流通共享不畅的问题成为困扰大数据应用往前发展的障碍。为此，中国通信标准化协会理事长奚国华强调，各方

都需要像重视实物资产一样重视数据资产的管理，为实现数据价值的持续释放打好基础。还要着力深化大数据行业应用，当前大数据行业中的应用主要集中在数据处理、用户画像以及企业管理效率优化等方面，下一步应重点推动大数据在更深层次与实体经济的融合，这需要鼓励大数据技术企业不断提升大数据平台和应用的可用性和操作便捷程度，优先支持面向各应用行业的产品、服务和解决方案的开发，简化大数据底层繁琐的技术，并方便大数据行业的部署。

在5G时代，新的应用场景定会催生很多新业态，包括无人机投递、VR游戏、工业机器人、视频安防、远程医疗等。当5G大规模商用以后，业界会开创出更多数字内容、数字形态，促进大数据产业的发展。

形式化验证的并发文件系统

（袁　杰　整理）

文件系统是存储和检索永久数据的基石，然而并发文件系统的设计和实现的复杂性使得其容易存在缺陷，从而不能在所有可能的并发执行下保证一致性。形式化验证是目前已知唯一的保证系统没有编程错误的方法，目前在形式化验证领域有一些杰出的工作（FSCQ，COGENT，Yggdrasil，CIO-FSCQ等），这些工作展示了用形式化验证的方法编写复杂系统的可行性，但是这些工作都无法验证在多核上运行的并发文件系统。

上海交通大学陈海波教授团队在SOSP 2019上发表的论文AtomFS对并发程序精化验证的相关理论成果进行了扩展，使用CRL-H框架进行形式化验证，并在定理证明工具Coq中设计并实现了支持帮助机制的并发精化验证框架，成功地验证了AtomFS，第一个形式化验证的并发文件系统。该工作证明了AtomFS接口（rename，mkdir等）在任意并发环境下的功能正确性以及原子性。测试表明，与ext4等成熟文件系统相比，AtomFS具有良好的多核扩展性，在运行实际的应用程序时能取得合理的性能。

谷歌正式发布数据可视化工具，提供超 500 个数据源

（王海瑞　整理）

大数据可视化是大数据产业链中的基础支撑产业，成为大数据行业热点。在大数据行业中，数据可视化起到的作用是更好的分享和传达数据信息。

Google宣布在2016年发布测试版本的数据可视化工具Data Studio，2017年发布的数据预先处理服务Cloud Dataprep。经过2年的发展，目前已提供了正式版。Data Studio与Google云平台紧密整合，使用者能以简单的操作，读取Google Analytics、Google Ad和Google BigQuery共超过500个Google和非Google的数据资源，进行可视化并探索数据。

探测器让使用者只要简单的鼠标点击操作，就能完成数据合并的麻烦工作，这加快了分析师的工作流程，也对非技术用户更加友好。数据混合则能让企业以单一画面，检查多个跨数据来源的可视化分析结果。Google Marketing Platform报告库则能创建特别的报告，这个功能提供了一系列的报告样板，让使用者可以快速整合数据，并生产信息丰富的专业报告。

Data Studio持续加入更新后，宣布成为正式版，Google还开发社区连接器，让使用者可以连接不同的数据来源以及定制化报告模板，除了能够生产企业专属的报告外，也能轻松分享解决方案。

Cloud Dataprep是一个无服务器的可视化数据处理服务，能够自动识别上传数据的Schema、形态等，供机器学习或进一步分析之用。Cloud Dataprep推出正式版，提供更好的协作功能，团队可以共同进行数据处理工作，透

过分享工作流程，重新使用数据样本，甚至能够检查和审查成员的工作进度。对于商业分析师，Cloud Dataprep也提供了计算表中的Pivot和Union功能，用来改进数据来源。

由于Cloud Dataprep是无服务器的服务，因此所有新功能都是现在立即可用，新用户可以通过操作指南开始进行探索和清理数据。

华云发布：首款国产通用型云操作系统安超OS™

（张士强 整理）

2019年8月8日，在工业和信息化部及多个国家级行业协会单位的支持下，首款国产通用型云操作系统安超OS™在北京雁栖湖国际会展中心隆重发布。

国产通用型云操作系统安超OS™依托中国云计算领军企业华云数据集团的雄厚资源，是一款具有应用创新特性的轻量级云创新平台，能够让政府和企业客户通过简单便捷的操作实现云部署和数字化转型，推动经济快速发展。

安超OS™拥有全栈、安全、创新、无厂商锁定的特性，依托国内雄厚的资源和全球领先技术，支持国内外众多品牌服务器，全面适配国内外芯片、操作系统和中间件，加速产品技术提升以及产业生态健康发展。

安超OS™目前已经顺利完成各项国家级行业认证，包括“可信云开源云解决方案认证”，“国家标准云计算解决方案标准符合性测试”，同时也通过了国际认可的CNAS实验室测试；安全方面通过了可信云的云服务用户数据保护能力的认证。在适配兼容方面完成了银河麒麟基于飞腾芯片的互操作认证、中标麒麟操作系统的互操作认证以及与海光CPU服务器的兼容测试认证。

安超OS™国产通用型云操作系统分为标准版、企业版、高级版三个版本，是一个具有全面功能、可灵活扩展，且能长期演进的全栈云平台，从最底层的芯片开始，直到硬件、软件、解决方案等各个层面都可以全面提供支撑，为不同的应用设计和优化，是一种横向扩展的软件集成基础架构，具有软硬件解耦、应用优化、支持混合业务负载等特点，面向政府、制造、金融、医疗、能源、电力、教育、交通以及互联网等多行业客户，为企业客户提供完整的SDDC（软件定义数据中心）解决方案，让业务更灵活的创新，数据更安全的流转；运维更轻松的实现。

SD-WAN带来了新的安全挑战

（孙胜尧 整理）

SD-WAN的现状如何？随着数字技术的发展，很明显，传统的分支机构之间的联系已经不能满足当今企业所需要的复杂连接。就像一条分开的隧道一样简单，一个分支机构与公司总部之间有专门的连接，而与互联网的实时连接可能会破坏整个组织的安全。SD-WAN提供类似支持先进的业务应用，移动的能力对延迟敏感的数据，如声音或视频到可靠，高速链接，并结合多个连接在一起，如核心网络的链接，连接到多重云网络和服务，和生活连接到互联网和移动设备——成为一个完整的软件包。我们看到组织所面临的最大挑战是试图将一致的安全框架应用到这个新环境中。它不仅需要保护主要的SD-WAN连接，而且还需要集成到部署在其他地方的任何安全解决方案中，比如在云中或远程网络中。这允许组织实现单一的安全策略，包括应用程序保护、web过滤、沙箱、网络访问控制、SSL检查，以及诸如NGFW、IPS和VPN等解决方案，从而保护应用程序、工作流和动态数据。

既然安全是SD-WAN的核心要求，那么又带来了哪些新的挑战呢？最大的挑战是传统的安全解决方案已经不再适用。传统安全解决方案不具备SD-WAN连接所需的性能、灵活性或互连性。更有挑战性的是，他们经常

看不到边缘连接。这就是为什么我们一直在开发基于意图的细分。这种策略可以基于许多参数隔离用户、应用程序、工作流或数据，从而在整个事务路径上提供安全性。流量可以被强制遵循特定的行为，或者被隔离到特定的用户或目的地，以确保始终一致的策略应用程序和执行。

安全团队面临的挑战之一是：他们已经被太多的安全工具所淹没，这不会加剧这个问题吗？真正的问题是尝试使用从未为此设计的工具来保护分布式网络。往往会发生的是，安全性仅应用于网关，这会降低对网络的深入可见性，或者为网络的不同部分选择和部署不同的工具。IT团队很快就会被安全蔓延所淹没，因此，工具不会得到更新或优化，或者执行不一致。

我们需要的是一个单一的安全平台，无论在何处部署安全解决方案，都可以提供一致的策略实施，然后使用统一的管理和编排控制台进行管理。核心，云端和分支机构的安全性需要像单个整体系统一样进行部署，实施，管理和优化。当然，这说起来容易做起来难。例如，不同云环境中的本机控件可能会有很大差异。安全解决方案需要根据其应用和管理的能力进行仔细选择，无论其部署位置如何。

2019年数据泄露事件激增，怎样防止数据泄露？

（杨　林　整理）

根据Risk Based Security公布的数据，2019年的数据泄露事件达到了一个高峰，已经发现超过3800多起企业或者机构数据泄露事件，在过去四年中增加了50%甚至更多。

报告指出，在2015年至2018年期间，数据泄露事件数量变化了不到200起。但在2019年的前六个月，数据泄露事件的数量与去年同期相比增加了54%。相比之下，根据该报告，2019年上半年曝光的记录数量与2017年同期相比下降了30%，尽管这个数据可能会在下半年发生变化。最近的报告详细介绍了Paige A. Thompson所披露的数据范围，被称为Capital One的黑客据称拥有来自30多家其他公司、教育机构和其他实体的数TB被盗数据。

尽管网络安全社区对内部威胁提出了担忧，但有89%的漏洞是外部攻击造成的。但报告指出，越来越多的泄露事件源于内部人士无法妥善处理敏感数据或安全信息，其中错误配置数据库和服务共有149起，导致超过32亿条记录被泄露。

Risk Based Security还指出，将敏感数据交给第三方也会发生数据泄露，例如黑客侵入了AMCA的网络，窃取了逾2200万债务人的记录，包括姓名、地址、出生日期、社交号码和财务细节。该报告指出，这些违规行为对AMCA造成了严重后果，因为该公司仅在违规消息成为头条新闻的两周后，就被迫申请破产保护。

根据Risk Based Security公布的数据，医疗保健服务是受影响最大的行业，零售、金融和保险、公共管理和IT占据前五位。那么，该怎样保护重要的数据不被泄露呢？公有云在未来可能是个不错的选择。全球云入侵防护软件市场预计将增长18亿美元，复合增长率为24.8%。以往人们非常担心公共云在网络数据安全方面的风险，但在过去几年中，由于安全技术提供商云端研发支出的增长，这种情况可能会发生改变。

换句话说，如果您正在寻找能够与最新安全功能保持同步的技术，那么公有云可能是您最好的选择，甚至不需要考虑成本和敏捷性优势，因为这些优势会融入到托管选择方案中。然而，在不久的将来这种方案可能也会出现一些基于云的漏洞，但是，这些漏洞并不是由于缺乏可靠的安全技术，而是由于缺乏安全人才。所以，想要保护企业数据不受侵害，可靠的安全技术与优秀的安全人才缺一不可。

不忘初心，砥砺奋进

题为“LaLDPC: Latency aware LDPC for Read Performance Improvement of Solid State Drives”的论文发表在Massive Storage Systems and Technologies (MSST) 2017会议上。MSST是计算机存储领域享有盛誉的顶级国际会议之一。论文针对现有闪存中的低密度奇偶校验码读数据过程进行改进，提出延迟感知的读机制，以实现闪存读性能的显著提升。

前几天收到实验室老师的邀请，分享当时发表这篇MSST论文时的心路历程。很高兴有这么个机会，在时隔两年后的今天，再去回顾那段奋斗的岁月，希望给实验室的研究生同学们带来一些启发和鼓励。

在读博的几年里，科研的心态经历了几个不同的阶段：从国际顶级会议论文（下文简称“顶会论文”）念头的萌芽，到自身科研知识和能力的逐步提升，再到困境时的坚持，到最后论文的发表。这些阶段都离不开老师们的培养，尤其是邹德清教授和金海教授，以及在香港城大的导师薛春教授。接下来，按照这几个阶段中所涉及的重要事件来讲述当时的一些经历。

1. 萌芽

从一入学了解实验室的规章制度开始，TOP80顶会论文列表就开始慢慢融入到科研的学习生活中，成为读论文的目标。导师给了课题研究方向后，就在这些顶级会议里找相关论文。虽然一开始读不懂一些专业术语，那时候也觉得可望而不可及。但是，心底里有那么一种念头，如果有一天也有机会发顶级会议论文，该有多好！之后在香港学习期间，仍然离不开TOP80会议列表，它犹如一盏明灯，照亮科研学习之路，也照亮前行的目标。甚至在走上工作岗位后，仍然时不时翻看实验室的最新TOP80列表，看那些会议的截稿日期，看那些会议的召开和发布论文时间，真想每一个会议都去投稿试试！

2. 逐步提升

随着顶会论文阅读量的增加，在知识学习，科研能力和英文写作这三个方面都慢慢有了提高。在知识学习方面，对于从数学转到计算机专业的学生而言，一开始学习最大的苦难就是计算机系统结构的知识有所欠缺。于是从基本的书籍资料开始，把论文中遇到的任何不懂的知识点都弄清楚。想分享的经验是，查资料要以论文为核心，而不是以书籍基础知识为核心。

在科研能力方面，经过一个显微镜图像超分辨项目的训练，慢慢地开始领悟什么是科研，该如何针对一个不熟悉的方向开展研究工作。而且，也开始懂得科研能力在不同项目上是相通的，这一点在后面系统结构的研究中受益匪浅。研究闪存存储器相关问题是从2016年初开始的，因为导师给闪存中的低密度奇偶校验码这个研究点，找论文的时候问题很聚焦。从顶级会议中找到相关论文后，就开始精读。以introduction作为了解研究内容和问题的入口，以background作为学习基础知识的入口。在读了大约10篇左右论文之后，对其中一篇论文印象极为深刻，并且会反复思考其中的设计。为什么低密度奇偶校验读过程要从最低

等级开始试？针对闪存后期，错误率很高的情况会不会有很大的叠加读延迟？随着这些问题的推进，延迟感知的思想开始进入脑海并形成一个完整的思路。idea的产生是有一定的运气存在的，但是必须建立在大量阅读和对问题的充分理解基础之上。确定基本Idea思路后，接着安装了闪存模拟器并动手实现基本思路。导师建议先通过简单的实验去验证方法的动机和有效性，于是大约2周就做出了基本实验结果。最后，在完善实验结果后选择投稿FAST顶级会议，虽然知道距离顶会还很远，但是能有投稿的机会去听一下审稿人的意见也是很不错的。

在英文写作方面，第一篇论文是独立完成写作和修改润色工作的。一开始完全不知道怎么写，就只能去模仿读过的论文中每个章节和段落的写作方式。针对每一段内容的写作，都要反复翻看几篇论文的相似章节。为了写好每一个章节和段落，清楚表述一个问题，经常反复在脑海里思考很久才开始动笔。等写完几个章节之后，又回过头来读，竟然发现自己写的完全都不知道在说些什么。于是，重新开始再写。这样的重写大概发生了四五次，才终于有一版自己能看得懂的论文。接着给师姐看了一下，她虽然不是做这个方向的，但是却提出了十分中肯的建议。改下来又重新写了几遍，最后终于投稿。这个经历对之后的写作大有裨益。每个人都应该严格要求自己，如果连自己都读不懂的内容，别人怎么可能懂呢？一个好的习惯是这样的，写好后过几天再来读，以一个阅读者的心态去读，这时候你才能发现自己哪里没有说清楚。基于前面的写作经验，MSST这篇论文的写作倒是少了几个重写的来回。

3. 坚持

上面说到MSST2017这篇论文先是投稿FAST2017会议，但当时投的版本实在是拙劣，方法和实验都过于单调，所以reviewers给的评审意见都不乐观。各位老师们给的建议都是修改一下投稿trans期刊，但是太想投一篇TOP80的顶会了！那个曾经萌芽的初心，这时候愈发强烈。一直在思考，这篇工作到底要怎样才能实现延迟感知思想的真实价值？某个晚上想到它在应用时缓存管理会有一些问题，于是进一步提出了延迟感知的缓存管理算法，和延迟感知的读过程结合，这样工作就更完整了。虽然那时候一篇可以算作毕业的论文都没有，冒着继续被延毕的危险，还是决定再次尝试顶会。也许因为这份坚持，把这个工作做得非常扎实，包括实验细节和论文的展示。最后，很幸运收到了被录用的好消息。

上述回顾的那段经历，恍如昨日。带着一定要发一篇TOP80顶会论文的初心，在科研道路上不断提升自己，要求自己做到最好。也是这份初心，使我在当前工作岗位上，能够继续保持顶会论文阅读习惯，保持对顶会的敬畏之心，保持砥砺奋进的精神！在此，再次感谢实验室营造良好的科研环境和老师们的精心培养！

杜亚娟
2012级博士研究生
研究方向：闪存存储器性能优化
Email：dyjxtu5@126.com

分布式系统组典型成果介绍

周　放，戴小海，郝威峰，陈彩娟，顾　琳，陆　枫，余　辰，肖　江，刘方明，吴　松

关键词：容器技术，分布式网络，数据中心冷却，区块链，智能交通，医疗大数据

1. 介绍

分布式系统组长期致力于虚拟化、数据中心、云端融合等领域研究。基于本组所承担的“容器虚拟化及新型ICT系统结构”、“高效能云计算数据中心资源调度关键技术研究”、“面向人机物融合的云计算平台”、“面向国产服务器的虚拟化关键技术”、“国家高性能计算环境服务化机制与支撑体系”等国家重点研发计划项目，以及“高效能数据中心--数据中心资源的优化组织与调度”、“边缘计算基础理论与关键技术”、“基于无线网络物理层信息的细粒度手势识别方法研究”等国家自然科学基金项目和一批企业合作项目，小组在三个层面开展了主要研究工作：在分布式系统软件层针对轻量级虚拟化技术展开研究；在分布式基础架构层针对数据中心网络与冷却系统展开研究；在分布式应用层针对区块链技术、智慧交通和医疗大数据等方向展开了研究。

1）在分布式系统软件层，主要对新型的轻量级虚拟化—容器技术进行了研究。设计并实现了容器网络流量控制系统、容器在线迁移系统和容器启动加速系统。为解决容器网络恶性竞争，提出了容器数据包识别、网络资源保证和动态调整三个功能模块，实现了容器网络的有效隔离和最大化的网络资源利用率。为了便于分布式系统中负载均衡管理和故障恢复，我们利用Docker层级镜像的特点提出了容器快速在线迁移系统，以满足性能、扩展性和宕机时间三个方面的要求。为解决无服务计算等一些场景下，容器部署和冷启动存在的性能瓶颈，利用新型非易失性存储器NVM，设计了容器启动加速系统。

2）在分布式基础架构层，我们聚焦分布式网络资源管理和数据中心冷却系统。为了缓解互联网应用带来的突发流量压力，设计实现了一种基于DPDK和mTCP用户态协议栈的快速缓解突发流量的动态按需组包卸载与批处理系统PostMan，克服了传统方法中的痛点问题，显著降低应对突发流量的延迟，并大幅提升吞吐量。为实现雾计算中不受限的VNF动态部署和配置，提出了一种不依赖数学模型的优化方法，在考虑端到端延迟和各种成本的情况下，协调VNF并调度网络流以最大化整体网络效用。以可持续发展的“绿色计算”为目标，设计实现了独辟蹊径的高效能数据中心混合液冷系统，其巧妙而实用的“黑科技”：温水冷却协同廉价普适的半导体制冷器件、以及软硬结合的细粒度控制方法，以高性价比将主要耗能部件的局部能效比指标PUE降至国际先进水平，大幅降低了数据中心的能耗成本。

3）在分布式应用层，不仅针对区块链应用开展了系统研究，还在新兴的智慧交通和医疗大数据领域也进行了研究探索。在区块链技术中，跨链交互和区块链性能依然是区块链领域亟待解决的问题。为此我们探索了不同区块链系统上数据建立高效可信连接。在智慧交通方面，利用无线感知技术，实现了价格低廉、易扩展、易部署的交通状态测量。在医疗大数据方面，针对微小出血病灶的监测提出了一种基于深度卷积神经网络的检测模型，从医学图像中提取特征实现对微小出血病灶实现精准检测，辅助医生进行智能诊断。

2. 分布式系统软件

分布式系统软件方向专注于云平台中的轻量级虚拟化技术。相比于传统虚拟机，以容器为代表的轻量级虚拟化技术，具有开发效率高、方便大规模部署、启动快、服务灵活、资源消耗少等优良特性，正越来越广泛地应用于云服务中。我们针对当前云服务商在分布式系统管理中遇到的三大热点问题：容器的网络流量控制、在线迁移和快速启动进行了深入的研究。

2.1 容器网络流量控制系统

为了提供可靠稳定的运行环境，云服务提供商需要确保不同用户间的有效隔离。传统虚拟机拥有完整的操作系统和独占的内核，使用硬件抽象的方式为客户提供计算、内存、存储和网络资源的隔离与控制。容器共享主机内核以减少虚拟化开销，天然具有隔离性差的特点。Docker 作为主流的容器系统，使用CGroup 提供了计算、内存和存储资源的隔离和控制，但不提供对网络资源的隔离，使得同主机上的容器产生资源竞争，严重影响网络性能。

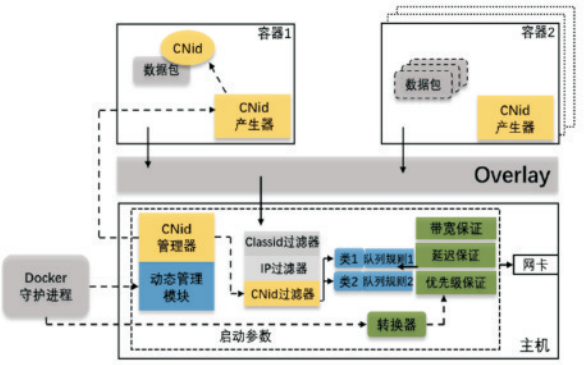


图2.1.1 网络流量控制系统架构图

基于已有Traffic Control（TC）框架，我们设计实现了一个有效、易用且适应容器特点的网络流量控制系统。TC 是一个为无虚拟化进程设计的网络流量控制框架，具有以下缺点：无法识别Overlay 网络中的容器数据包；配置过程复杂，使用难度大；使用静态资源管理，资源利用率低。

我们的系统主要使用三个模块，实现对容器网络流量的有效控制：1）识别模块，包含一种为容器设计的网络数据包标记和识别方式。该标记在数据包控制信息结构体中添加容器信息以避免Overlay 网络导致的信息隐藏和数据包跨网络命名空间导致的信息丢失；2）资源保证模块，包含一系列用于资源配置的容器启动参数和转换器。容器启动时通过启动参数设置网络资源需求，再由转换器将参数向网络资源转换，实现容器的带宽、延迟和优先级保证；3）动态调整模块，包含一个调整算法，根据容器状态调整网络流量控制需求。当存在释放的网络资源时，模块将资源重新配置给运行中容器，以最大化网络资源利用率。

实验结果表明，我们的系统能够为容器提供有效的网络资源隔离，消除容器网络间的恶性竞争。同时，本系统根据容器状态动态地对主机网络进行调整，最大化网络资源的利用率。最后，系统性能开销极小，符合轻量级虚拟化技术的要求。

2.2 容器在线迁移系统

为了保障云服务的质量，并为数据中心负载均衡和单点故障的场景提供解决方案，Docker 容器的快速迁移技术不可或缺。理想的Docker 容器在线迁移必须满足以下三个要求：1）在功能上保证核心组件（镜像，运行时和管理上下文）的完整性和一致性；2）在性能上要能适应数据中心的扩展性；3）保证服务的宕机时间尽可能短。然而现有容器迁移无法同时满足这3个要求，要么忽略Docker 容器的组件完整性，只关注容器的运行时环境的迁移；要么在保证组件完整性的同时，牺牲系统可扩展性；要么使用重载Docker Daemon的方式来恢复Docker管理上下文，明显增加宕机时间。

为了同时满足数据中心对Docker 容器在线迁移的三项需求，我们设计了面向数据中心的Docker 容器在线迁移系统，利用Docker层级镜像的特点加速镜像的迁移，并对原始管理上下文迁移的方式进行了优化。首先，提出一种轻量级

容器仓库机制，基于对本地镜像层级关系的分析，在源端构建轻量级仓库来实现端到端镜像迁移。该机制能够避免冗余镜像层传输，大幅提升性能的同时明显增强了数据中心的可扩展性。其次，提出了管理上下文的动态加载机制，基于对Docker Daemon和Docker 容器管理模式上耦合关系的分析，可以在不影响目的端Docker Daemon运行的前提下，加载迁移容器的管理上下文，能够显著降低Docker 容器服务的宕机时间。

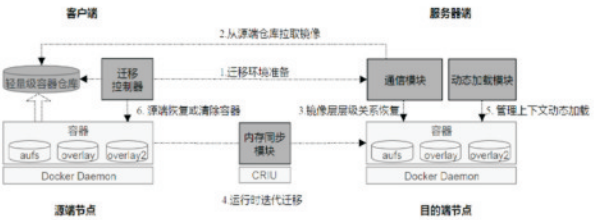


图2.1.2 Docker容器在线迁移系统框架图

实验结果表明，面向Docker 容器的在线迁移系统在大量并发的容器迁移环境中能够降低90%的迁移时间，分别减少了迁移过程中87%的镜像迁移时间和71%的容器服务宕机时间，可以同时满足数据中心对Docker 容器在线迁移在功能、性能和扩展性方面的需求。

2.3 容器启动加速系统

近年来以Docker为代表的轻量级虚拟化技术获得快速发展和广泛应用，快速部署的特性是一个重要原因。然而在无服务器计算等一些场景下，容器部署和冷启动存在明显的性能瓶颈。每当函数被调用时，容器就会启动，并且在处理完请求后会立即销毁。在无服务计算平台中，通常会使用高级语言，比如Python 和JavaScript，需要更复杂的运行时环境，导致其启动速度比同等的C 程序慢十几倍，产生大量容器冷启动，极为耗时。新兴的非易失内存NVM的出现为加速容器启动提供了新的机会。

我们对Docker Hub 上下载超过100 万次的134 个镜像进行深度的分析后发现：1. 镜像存储是容器部署过慢的原因之一；2. 容器冷启动只依赖一小部分镜像文件。基于这两点发现，我们设计了非易失内存的容器加速系统N-Docker。在容器部署阶段，N-Docker 能找到拉取较慢的镜像层，并将其放到非易失内存，加快部署过程。在容器冷启动阶段，N-Docker 通过跟踪抓取容器冷启动过程所必需的镜像文件（热镜像文件），将其存储到非易失内存，剩余部分（冷镜像文件）写回硬盘，以加速冷启动过程。

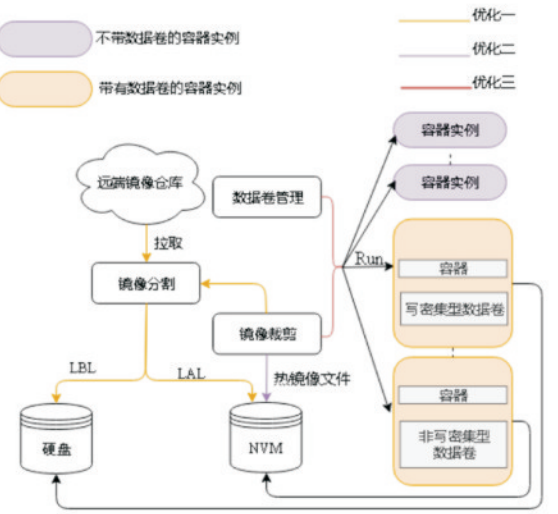


图2.1.3 N-Docker总体功能

实验结果表明，N-Docker 容器部署速度加快了1.21 倍，容器冷启动加快了2.96 倍。相比于将镜像全部存储在非易失内存的NVM-Docker，N-Docker获得了一样的性能，但是非易失内存的使用量只是NVM-Docker 的11.8%。

3. 分布式基础架构

在分布式基础架构层，我们主要关注分布式系统网络资源管理和数据中心冷却系统的优化。针对分布式系统突发流量压力，设计了数据包卸载与批处理系统，克服了传统方法中存在的痛点问题，显著降低应对突发流量的延迟并大幅提升吞吐量；在雾计算网络方面，主要研究了在云雾混合环境中考虑端到端延迟和各种成本的情况下，协调VNF（即VNF激活和停用）并调度网络流以最大化整体网络的效用问

题；为了实现可持续发展的“绿色计算”目标，我们设计了IT负载感知的高效能数据中心混合液冷系统，以高性价比将主要耗能部件的局部能效比指标PUE降至国际先进水平，大幅降低数据中心的能耗成本。

3.1 快速缓解突发流量的数据包卸载与批处理系统

互联网突发事件（如热点新闻、促销季等）会产生难以预测的激增流量，实际测量表明这类突发流量中往往包含大量的小数据包。当前的分布式系统及网络协议栈在处理这些小数据包时会产生巨大开销，给后端服务器带来严重的负载均衡问题。解决负载均衡问题的传统方法之一是进行热数据迁移，进而将流量从过载服务器分发至其他服务器。然而，这一方案耗时久、开销大，会进一步降低过载服务器的性能。

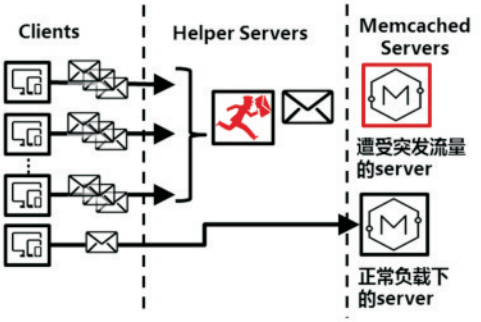


图3.1.1 PostMan基本思路与功能

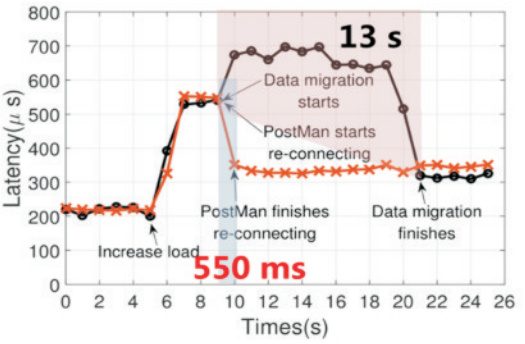


图3.1.2 PostMan快速缓解Memcached服务中的突发流量

为了解决该痛点问题，设计了快速缓解突发流量的动态按需组包卸载与批处理系统PostMan，可作为热数据迁移的应急方案。如图3.1.1所示，PostMan在客户端与服务器之间部署合理数量的中间辅助节点，并实现了基于DPDK和mTCP的用户态协议栈，以进行高效的按需组包与批处理：即当服务器过载时，辅助节点将发送给同一服务器的大量小数据包先组装成大数据包，然后再转发给该服务器，从而将小数据包的处理开销从过载服务器卸载到辅助节点上。同时，PostMan能够仅在检测到突发流量时才按需启动或关闭辅助节点，以最大限度减少额外成本。进一步，为了保障辅助节点的容错性，PostMan能够在不同辅助节点之间进行无缝流量迁移，并在迁移过程中避免数据包乱序的隐患。如图3.1.2所示的实验表明，PostMan可帮助Memcached服务在数百毫秒内缓解突发流量，而传统的热数据迁移方案则需要数十秒之久。

3.2 VNF实例部署与网络流调度

雾计算作为一种新兴的计算范式，利用网络边缘的分布式设备资源，将网络中心的云服务扩展到网络边缘，被认为是VNF部署的理想平台。在靠近终端用户的地理分布式雾计算服务器中部署VNF和处理网络流，可以快速灵活地提供多样性的网络服务，同时有效降低因特网上的传输延迟和流量拥塞，改善用户体验。但在雾计算范式中提供服务时，雾节点在资源提供和资源需求方面表现出高度动态，因此在云雾混合环境中，如何以一种动态的方式适应部署和配置VNF，对网络流进行调度，以适应时变的网络环境对于降低成本支出和最大化网络效用来说是至关重要的。而传统的基于模型的资源管理方法由于某些假设或先决条件的参与，在实际应用中受到限制。因此，设计一种无需任何先验知识、不依赖数学模型的优化方法，是构建云雾混合环境下高效网络服务的关键挑战。

我们主要研究了在云雾混合环境中，协调VNF（即VNF激活和停用）并调度网络流以最大化整体网络的效用问题。根据每个时隙中每种网络流的速率，决策每个时隙VNF实例的开启或停用，同时对网络流进行分流处理，以达到各种成本支出的减少、延迟和收益的折衷，

使收益和成本之差最大。而鉴于强化学习在已有工作上的优良表现，我们选择利用强化学习算法中的Deep Deterministic Policy Gradients（DDPG）算法来决策出每个时隙每种网络流的分流策略，其中VNF实例的开启/停用策略则通过对网络流分流的情况间接推导得出，从而提出了一个基于强化学习的VNF编排和网络流调度算法，来解决云雾混合环境中NFV的性能优化问题，以获得更大的网络效用。

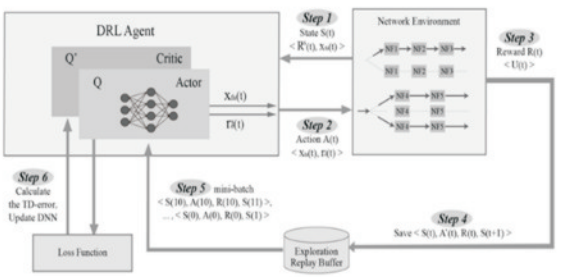


图3.2 DDPG算法架构和步骤

图3.2显示了DDPG算法的架构以及主要步骤。主要为DRL Agent获取Network Environment中的环境状态State（在我们的问题中为当前时隙每个VNF实例的激活状态以及每种服务的网络流速率），做出相应的决策Action（网络流的调度决策，并从中推导出VNF实例的激活状态），然后将该决策应用到Network Environment中，获取环境给予的反馈Reward（网络总效用）以及Network Environment的下一个状态。最后将这些信息储存在经验回放库中，并在适当的时候进行取样用来更新Agent的决策机制。Agent不断迭代重复这一过程，最终使算法达到收敛状态。

与此同时，在实验过程中观察到原始的DDPG算法收敛缓慢并且性能不佳，主要有两个原因：1）DDPG算法使用基于正态分布的随机探索方法来探索自己的决策，这导致一些可能较优的决策更少被探索和学习；2）DDPG算法从经验回放库中随机抽取样本用来更新Agent的决策机制，使得算法有时被困在低效用的坏样本中。此外，由于经验回放库的大小限制，不良的样本甚至会塞满经验回放库而踢出效用良好的样本。

为了取得更好的性能，我们针对这些问题对算法进行了改进，提出了我们的Customized DDPG (cDDPG)算法，主要有：1）偏差探索：针对在基于正态分布的探索中一些较优的决策可能会被忽略的情况，我们提出了偏差探索，通过对一些特定的状态指定相应的决策进行探索，确保这些较优的决策被更多的探索与学习。2）双重经验回放库：上面提到的第二个问题是由于在经验回放库中无法区分存储所有样本而引起的。因此，我们提出了一种由全局回放库和基线回放库组成的双重经验回放库结构。全局回放库与通用DDPG算法设计中的经验回放库相同，而基线回放库特别用于存储效用良好的样本。而且由于构造了两个经验回放库，我们还重新设计了经验取样策略，除了仅从全局回放库中随机抽取样本之外，还使用来自基线回放库的一些样本来训练神经网络，通过这种方式来确保一些潜在的效用良好的样本可以包含在抽样中对神经网络进行训练。最后，通过实验验证了所提出的cDDPG算法相比于传统基于模型的算法以及原始DDPG算法都有极大的性能优势。

3.3 基于IT负载感知的高效能数据中心混合液冷系统

通过实际测量数据中心中IT能耗与非IT设施能耗之间的关系，发现：1）数据中心在实际运行中IT能耗负载远低于数据中心的设计峰值，导致配套的制冷设施利用率低下，存在能耗浪费现象；2）当前的水冷系统制冷策略非常保守，采用的制冷水温较低，因此存在大量冗余的制冷能力，而通过提高水温可以有效减少水冷系统的能耗，但是提高水温在面临突发负载时存在制冷失败的风险；3）当前水冷系统由于技术限制，大都采用集中式、粗粒度的制冷控制方法，导致水冷系统在应对局部热点时制冷效率低下。

为此，提出使用温水制冷来减少制冷能耗，并针对温水制冷带来的制冷失败风险设计了一种与半导体制冷片协同的混合水冷系统（如图3.3.1和3.3.2所示），以及自适应数据中心负载变

化的细粒度制冷控制方法：如图3.3.3所示，当数据中心中服务器温度普遍较低时，只采用半导体制冷片应对局部热点，而仅当大量服务器温度普遍较高时，才启用制冷机降低整体水温。该方法能够为数据中心服务器提供精确制冷，避免集中粗放式的制冷能耗浪费。通过原型系统实现和真实数据中心负载的实验验证，所提出的混合水冷系统在对CPU制冷时，可将局部能效比指标PUE将至1.04~1.05 的国际先进水平，大幅降低了数据中心的能耗成本。

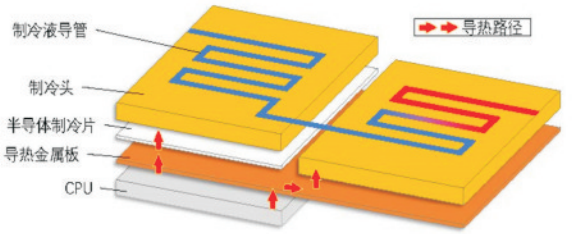


图3.3.1 混合水冷结构设计

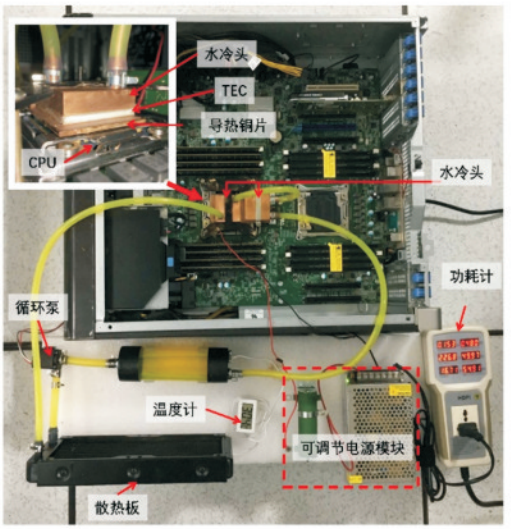


图3.3.2 混合水冷系统原型

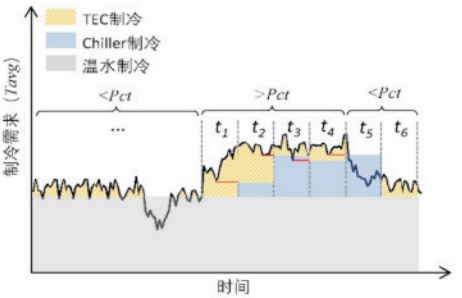


图3.3.3 混合水冷自适应细粒度控制方法

4. 分布式应用与新型应用

分布式应用方向的研究涵盖了区块链技术，以及关系国计民生的智慧交通和医疗大数据应用。

4.1 区块链技术

近年来，区块链技术经历了高速发展，越来越多的区块链系统得以部署和应用。然而，跨链交互和区块链性能依然是区块链领域亟待解决的问题。为此，我们探索了不同区块链系统上数据建立高效可信连接的问题。

4.1.1 面向跨链交互的新型区块链架构

为了解决现有区块链系统呈现数据流通性差和数据孤岛现象，我们展开针对跨链交互技术的研究。首先，我们对“跨链交互”进行了严格定义。在此基础上，我们提出了包括“原子性”、“高效性”、“安全性”、“兼容性”和“易开发”五个关键挑战。然后，我们对区块链的系统架构进行了分层设计来探讨“跨链交互”问题。如图4.1.1所示，分层包括数据层、网络层、共识层、合约层和应用层。最后我们探讨从不同层次解决这些挑战的可能思路。此外，结合多链数据，我们尝试进一步对链上的海量数据进行综合分析，从而绘制出区块链用户的人物画像。

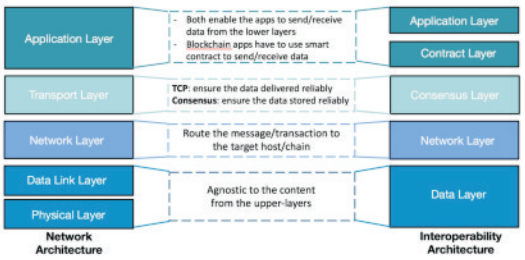


图4.1.1 区块链系统架构的分层设计

4.1.2 基于Merkle branch和bloom filter的数据缩减方案

针对比特币系统中节点存储容量急剧增大的问题，我们设计了一种基于Merkle branch和bloom filter的数据缩减方案。为保证数据缩减后系统运行的安全性，我们利用Merkle branch保证交易中的输入是存在的，并利用bloom filter来证明交易中的输入是没有被花费的。

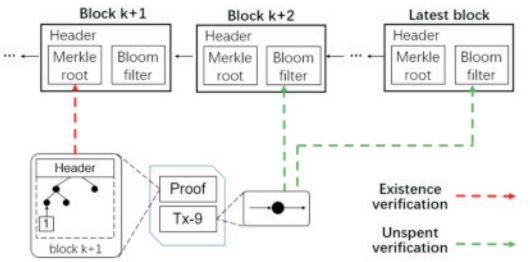


图4.1.2 数据缩减方案

4.1.3 区块链结构化网络拓扑协议

区块链通信时延是限制区块链发展的关键因素，一方面通信时延会影响数据传输；同时通信时延也会影响区块链安全性。为了降低区块链通信时延，我们提出了一种结构化网络拓扑协议，从区块链连接建立和通信广播两个角度进行优化。首先，为了提升区块链网络的可扩展性同时保证网络负载均衡，我们基于K-Means算法将网络中的节点划分为若干个分簇。其次，在网络分簇的基础上，构造高效的区块链分层结构化网络拓扑结构。最后，在上述的网络拓扑结构基础上，实现高效网络数据广播算法，包括簇内高效广播算法和跨簇高效广播算法。

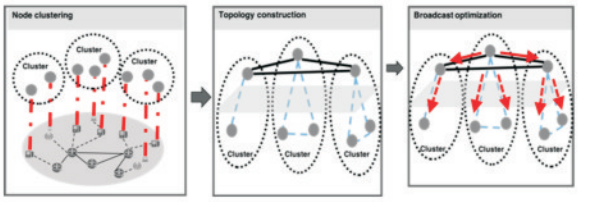


图4.1.3 BlockP2P 结构化网络拓扑协议

4.2 新型应用探索

我们对智慧交通和医疗大数据等新型应用分别进行了探索。在智慧交通方面，利用无线感知技术，实现了价格低廉、易扩展、易部署的交通状态测量。在医疗大数据方面，针对微小出血病灶的监测提出了一种基于深度卷积神经网络的检测模型，从医学图像中提取特征实现对微小出血病灶实现精准检测，辅助医生进行智能诊断。

4.2.1 基于细粒度无线网络的智能交通监测

随着城镇化的进程加快，私家车的普及和交通体量的增大，现有的交通检测系统面临着越来越多的挑战。另一方面，随着无线设备的普及和网络技术的发展，无线感知技术即利用无线网络信息如CSI感知周围环境的变化也成为一个新兴的研究方向。这启发我们将细粒度的无线网络信息CSI应用于监测交通信息的变化，来实现价格低廉、易扩展、易部署的交通状态的测量。

基于无线网络信息的交通监测的核心思想在于当车辆经过无线设备时，由于车辆的遮挡会对无线链路造成影响，通过测量无线网络信道状态的变化反过来推测交通状态的信息。交通监测系统主要由四部分组成：数据预处理，车辆检测，车辆分类，车速预估。我们利用多分支CNN的模型实现了车辆分类，并将边界检测和曲线拟合相结合的方法应用于车速估计。系统架构如图4.2.1所示。

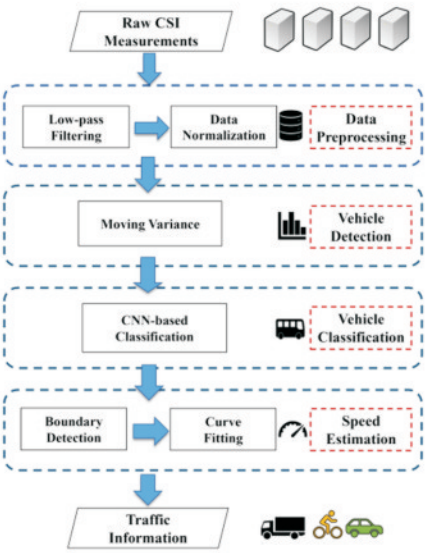


图4.2.1 基于CSI的交通信息监测的系统架构

我们用实际采集的数据对模型进行训练学习，实验表明移动方差在车辆检测时具有一定的鲁棒性，多分支CNN模型在车辆分类问题上要优于传统机器学习方法，边界检测和曲线拟合可以有效的实现车速预估。

4.2.2 基于WCE数据的微小出血病灶检测

作为一种非侵入式现代医学成像技术，无线胶囊内窥镜技术（Wireless Capsule Endoscopy, WCE）已经广泛应用于胃肠道疾病检查。然而，

产生的海量数据给医生带来了沉重的负担，基于深度学习技术进行计算机辅助医生智能诊断是当前迫切的需求。出血症状是许多胃肠道疾病常见的综合症状，而微小出血病灶对于诸如早期胃癌等疾病的早期发现尤为重要。因此，构建针对WCE图像中出血病灶，尤其是微小出血病灶点的检测方法，具有重要的理论和实际意义。

我们针对微小出血病灶的检测问题提出了一种基于深度卷积神经网络的检测模型。首先，构建基础特征提取网络，通过深度卷积神经网络从图像中自动提取特征。然后，再利用多尺度区域建议神经网络从特征金字塔中筛选出出血目标的候选区域。事先筛选出目标可能存在的区域不仅可以突出目标特征，且有助于减少无效特征的融合。之后，再通过构建自顶向下的特征融合网络，实现对目标浅层特征和深层特征的融合，通过融合不同层次的特征解决了由于微小出血病灶特征丢失导致的检测困难的问题，从而实现对微小出血病灶的精准检测，如图4.2.2所示。

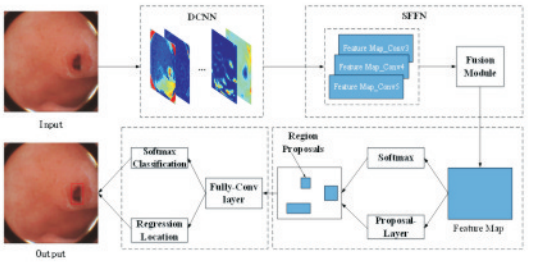


图4.2.2 基于特征融合的微小病灶检测神经网络模型

利用真实的临床数据对模型进行训练和测试，实验结果表明在包含微小出血病灶的图像数据集上，基于深度卷积神经网络构建的多尺度特征融合出血检测方法在灵敏度、准确率和F1得分等多项指标上明显优于现有方法。相较于现有出血检测方法提升了31.69%，整体性能F1得分提升了22.12%。

5. 总结

分布式系统小组的典型成果涵盖了分布式系统软件、分布式基础架构和分布式应用三个不同层面，尤其是在容器技术、分布式系统网络管理、数据中心冷却系统、区块链技术等方面有较强的研究特色，形成了一批有代表性的开源系统、论文和专利。小组除了在现有优势领域继续深入研究之外，在边缘计算、智慧交通和医疗大数据等新兴领域也开展了研究探索，并取得了初步的研究成果。

附成果列表：

论文

[1] Weixiang Jiang, Ziyang Jia, Sirui Feng, Fangming Liu, Hai Jin. Fine-grained Warm Water Cooling for Improving Datacenter Economy. ISCA 2019

[2] Panpan Jin, Jian Guo, Yikai Xiao, Rong Shi, Yipei Niu, Fangming Liu, Chen Qian, Yang Wang. PostMan: Rapidly Mitigating Bursty Traffic by Offloading Packet Processing. ATC 2019

[3] Bin Gao, Zhi Zhou, Fangming Liu, Fei Xu. Winning at the Starting Line: Joint Network Selection and Service Placement for Mobile Edge Computing. INFOCOM 2019

[4] Qixia Zhang, Fangming Liu, Chaobing Zeng. Adaptive Interference-Aware VNF Placement for Service-Customized 5G Network Slices. INFOCOM 2019

[5] Shutong Chen, Lei Jiao, Lin Wang, Fangming Liu. An Online Market Mechanism for Edge Emergency Demand Response via Cloudlet Control. INFOCOM 2019

[6] Qiong Chen, Zimu Zheng, Chuang Hu, Dan Wang and Fangming Liu. Data-driven Task Allocation for Multi-task Transfer Learning on the Edge. ICDCS 2019

[7] Xiaohai Dai, Jiang Xiao, Wenhui Yang, Chaofan Wang, Hai Jin. Jidar: A Jigsaw-like Data Reduction Approach Without Trust Assumptions for Bitcoin System. ICDCS 2019

[8] Lin Gu, Deze Zeng, Wei Li, Song Guo, Albert Zomaya, Hai Jin. Deep Reinforcement Learning based VNF Management in Geo-distributed Edge Computing. ICDCS 2019

[9] Song Wu, Shadi Ibrahim, Hai Jin, and Jiang Xiao. When FPGA-accelerator meets Stream Data Processing in the Edge. ICDCS 2019.

[10] Hang Huang, Jia Rao, Song Wu, Hai Jin, Kun Suo, Xiaofeng Wu. Adaptive Resource Views for Containers. HPDC 2019

[11] Zhuo Huang, Song Wu, Song Jiang, Hai Jin. FastBuild: Accelerating Docker Image Building for

Efficient Development and Deployment of Containers. MSST 2019
[12] Hao Fan, Song Wu, Shadi Ibrahim, Hai Jin. NCQ-Aware I/O Scheduling for Conventional Solid-State Drives. IPDPS 2019
[13] Song Wu, Fang Zhou, Xiang Gao, Hai Jin, Jinglei Ren. Dual-Page Checkpointing: An Architectural Approach to Efficient Data Persistence for In-Memory Applications. HiPEAC 2019 & TACO 2019
[14] Lin Gu, Junjian Guan, Song Wu, Hai Jin, Jia Rao, Kun Suo, Deze Zeng. CNTC: A Container Aware Network Traffic Control Framework. GPC 2019
[15] Weifeng Hao, Jiajie Zeng, Xiaohai Dai, Jiang Xiao, Qiang-Sheng Hua, Hanhua Chen, Kuan-Ching Li, Hai Jin. BlockP2P: Enabling Fast Blockchain Broadcast with Scalable Peer-to-Peer Network Topology. GPC 2019
[16] Jiang Xiao, Huichuwu Li and Yuxiang Liu. WiWrite: Wi-Fi based Handwriting Recognition like Playing Lego. ICCCN 2019
[17] Yusheng Yi, Jiang Xiao, Song Wu, Huichuwu Li, Hai Jin. Failure Order: A Missing Piece in Disk Failure Processing of Data Centers. HPCC 2019
[18] Lin Gu, Qizhi Tang, Song Wu, Hai Jin, Yingxi Zhang, Guoqiang Shi, Tingyu Lin, Jia Rao.N-Docker: a NVM-HDD Hybrid Docker Storage Framework to Improve Docker Performance. NPC 2019
[19] Caijuan Chen, Kaoru Ota, Mianxiong Dong, Chen Yu, Hai Jin. WITM: Intelligent Traffic Monitoring Using Fine-grained Wireless Signal. TETCI 2019

周　放
2014级博士
研究方向：异构内存管理
Email：flame@hust.edu.cn

戴小海
2017级博士
研究方向：区块链的可扩展性研究
Email：seafooler@hust.edu.cn

郝威峰
2016级硕士
研究方向：区块链网络优化
Email：haoweifeng@hust.edu.cn

陈彩娟
2017级硕士
研究方向：数据挖掘、普适计算
Email：chencaijuan@hust.edu.cn

顾　琳
讲师
研究方向：网络功能虚拟化、软件定义网络、云计算
Email：anheeno@mail.com

陆　枫
副教授
研究方向：高性能计算应用、移动互联网等
Email：lufeng@hust.edu.cn

余　辰
教授、博导
研究方向：普适计算、移动互联网
Email：yuchen@hust.edu.cn

肖　江
副教授、博导
研究方向：分布式计算与系统、大数据分析与处理、无线网络与移动计算
Email：jiangxiao@hust.edu.cn

刘方明
教授、博导
研究方向：分布式系统与网络
Email：fmliu@hust.edu.cn

吴　松
教授、博导，实验室副主任
研究方向：云计算、虚拟化
Email：wusong@hust.edu.cn

系统软件与体系结构组典型成果介绍

刘海坤，李若时，张　宇，郑　龙，马　阳，张杨松，蒋文斌，廖小飞

关键词： 内存计算，图计算，运行时，应用支撑

1. 介绍

随着图计算、深度学习等大数据新型应用的普及和各种加速器（GPU、FPGA）以及存储器件（MRAM、PCRAM）不断涌现，本组旨在研究新型体系结构及其系统软件关键技术，利用新型处理器和新型存储器件为上层新型应用提供有效的支撑平台。基于所承担的国家重点研发计划（“面向图计算的通用计算机技术与系统”和“新型内存和存储技术”），国家自科基金重点基金“大数据高效处理体系结构”及其它项目，在上一年度，本组主要在如下四个方面进行了研究：（1）内存计算；（2）图计算体系结构；（3）运行时系统；（4）应用支撑。下面介绍本组在这四个方面研发的一系列典型系统和技术：

1）在内存计算方面，研发了异构内存大页处理系统Rainbow，采用二维页面监测的形式进行页面信息监测，使用分离式TLB机制提升DRAM和 NVM的地址转换效率，采用高效物理地址重映射机制提升DRAM中的热页面地址转换效率，与没有大页支持的策略相比，可以显著降低大内存地址转换开销，同时提升系统性能。

2）在图计算体系结构方面，研发了基于订制缓存的高性能图计算系统FabGraph，在FPGA-DRAM平台上采用一种两级定制cache，减少图顶点信息冗余传输和流水线停顿，同时采用传输–计算重叠策略，充分利用两级cache，提高性能；同时也设计了基于3D-ReRAM图计算加速器，充分利用3D-ReRAM的计算高并行性加快图处理。

3）在运行时系统方面，提出了依赖关联性感知的图处理执行机制，利用图顶点之间依赖关系加快GPU上图顶点状态传递速度和减少数据访问开销；采用基于无效更新过滤的图计算加速策略减少数据处理量；同时研发了面向图计算的高层综合系统，简化图分析任务的编程。

4）在应用支撑方面，提出了内存重用及优化方法和分层细粒度深度神经网络并行优化方法加快深度学习过程；提出了异构内存感知的并行图计算系统NGraph，通过对图数据和任务划分，避免多个处理器处理同一个数据所带来的数据竞争，从而也避免了原子更新带来的额外开销。此外，其数据和任务划分策略也可以避免多核更新数据对数据预取效果的影响。

2. 异构内存大页处理系统

随着应用内存占用率的增加，虚实地址转换成为了系统性能的新瓶颈。当代的计算机系统普遍在用快表（Translation lookaside buffer, TLB）作为页表的缓存，通过在快表里面缓存最近使用的虚实地址转换关系，从而实现快速的地址转换查询操作，特别是在应用局部性较好的环境中，快表可以极大地提高地址转换效率，进而提升系统性能。目前有大量增加快表覆盖范围（TLB coverage）的工作。 这些工作主要是增加可以通过快表可以直接映射的内存范围。由于受限于访问速度、能耗和空间等因素，快表表项（TLB entries）的数目不能太多，所以大页（superpage）被广泛的用来增加快表覆盖范围。然而，大页有一个副作用，即

连续的虚实地址空间的要求影响了形如页面迁移等的轻量级、灵活的内存管理机制。

另一方面，由DRAM、NVM组成的异构内存系统需要依靠内存迁移来获得较好的系统性能和能效。即异构内存系统需要周期性的把较慢的非易失内存介质中的热数据（频繁访问的数据）迁移到DRAM中，同时把DRAM中的冷数据（非频繁访问的数据）迁移到非易失内存介质中，从而提高系统性能与能耗。遗憾的是，如果绝大多数的内存访问都分布在一个大页的小部分区域中，此时以大页粒度（比如2 MB）进行页面迁移，那么迁移操作将会造成难以忍受的性能开销（甚至会掩盖页面迁移带来的性能收益），例如DRAM 资源和带宽资源的巨大浪费。以上可知，异构内存中大页的使用存在一个两难的境地：轻量级页面迁移会削弱甚至抵消改进TLB涵盖范围所带来的性能收益。

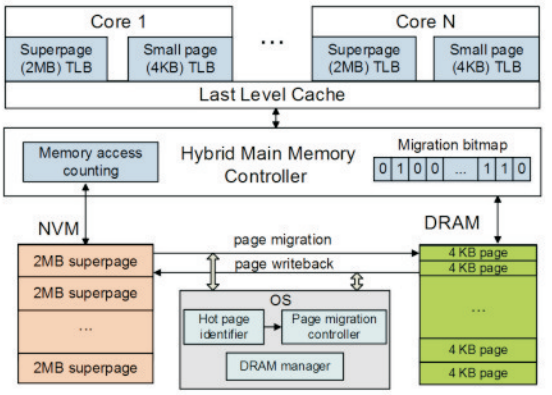


图1 Rainbow系统架构

为了解决上节所述的各项挑战，我们实现了如图1所示的异构内存大页管理系统——Rainbow，Rainbow 针对以上挑战的采取了以下创新的设计来同时支持大页机制与轻量级页面管理机制。

- 针对细粒度页面监测开销较大的问题，Rainbow 采用二维页面监测的形式进行页面信息监测。第一步，在一个给定的时间段内，Rainbow 首先以大页（2 MB）的粒度统计NVM主存页面的访问信息，接下来选择热度最高的N个页面作为被迁移对象。第二步，Rainbow进一步以小页（4 Kb）粒度监测热的大页里面的各个小页的访问信息。这种基于历史信息的策略避免了对冷的大页面冗余的细粒度监测，因此显著的降低了热页监测的开销。
- Rainbow使用分离式TLB机制提升DRAM和 NVM的地址转换效率。当一些小页面被迁移到DRAM中时，为了保证大页TLB的完整性，Rainbow通过在内存控制器中使用一个位图来甄别页面被迁移与否，故不需要拆分大页。
- Rainbow 采用了一种高效的物理地址重映射机制来提升DRAM中的热页面地址转换的效率，从而避免了昂贵的页表查询操作开销。该机制把迁移的热页的目标地址存储在迁移页面的源地址处。一旦出现某一个热页面的TLB缺失情况，系统会通过大页中的地址间接的访问这个热页。这种设计逻辑上把大页TLB 作为了4 KB 页面TLB的缓存。因为通常情况下大页TLB的命中率极高，Rainbow 可以显著的提升DRAM页面地址转换效率。
- 通过把以上设计模块进行统一与实现，我们基于Zsim 和 NVMain模拟器实现了Rainbow系统。据我所知，这是第一种研究同时支持大页机制和轻量级页面管理机制的工作。课题组使用大量的测试集测试，并使Raibow与多种相关策略进行对比从而评测Rainbow的性能。实验证明，与没有大页支持的策略相比，Rainbow可以显著的降低大内存负载的地址转换开销，同时提升多达2.9倍（平均43%）的系统性能。与此同时，Rainbow可以显著的提升系统能效。

3. 图计算体系结构

3.1 FabGraph：基于订制缓存的高性能图计算系统

在图计算中，FPGA平台提供了高效处理大量图数据的能力。在现有的FPGA-DRAM图计算平台上，每条流水线附带了对应的顶点cache用于在计算过程中缓存源和目标顶点。然而这样的单级cache结构会消耗大量的DRAM带宽，并造成流水线停顿，降低了性能。因此我们提出了一种FPGA-DRAM平台上的两级定制cache图计算架构FabGraph，来减少顶点信息的冗余传输和流水线停顿，以进一步提高性能。为了充分利用两级cache，FabGraph同时还采用了传输–计算重叠的策略。

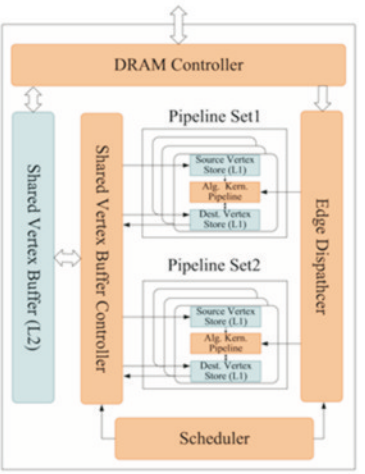


图2 FabGraph解释图

图2展示的是FabGraph的逻辑结构框图，其中图的边和顶点一开始被保存在DRAM中，在计算过程中顶点数据通过L1和L2两级cache被流水线访问，而边则以边流的形式在计算的同时传输给流水线。两级cache主要通过两种策略发挥作用：（1）分块机制。图分块就是将一个图中的顶点序列分在Q个大小相等的间隔中，然后将图上的所有边根据每条边的原点和终点分给Q2个边块中。这样每次计算都在块内进行，大大加强了计算的局部性。（2）计算–传输重叠机制。FabGraph有两组L2 cache，当其中一组作为流水线cache被访问时，另一组从L1 cache中将下一次计算需要的顶点数据读出，从而减少了流水线延迟。此外，BRAM使用倍频的方法运行在更高的频率，可以充分利用片上BRAM资源。在实验结果中，FabGraph相对于当前性能最高的ForeGraph架构在BFS和PageRank算法的四个不同数据集上的加速比在1.2~3.1x之间，同时数据传输量减少了大约一半。

3.2 基于3D–ReRAM图计算加速器

新兴的忆阻器（如ReRAM）在交叉开关阵列（Crossbar）形式下具有执行矩阵向量乘法的潜力。相比与传统基于CMOS的计算结构，它允许每个周期同时执行多个操作，具有相当大的潜在并行性，并且无需数据移动，可以显著降低能耗开销。由于具有原位计算和高并行性的能力，ReRAM是存内计算（Processing in Memory）架构的理想选择。

图算法通常可以转换为矩阵向量乘的形式，利用ReRAM加速图计算是一种有效的途径。随着单片三维（3D）集成技术的发展，3D集成成为增强ReRAM可扩展性的有效方法。如图3所示，3D-ReRAM不仅提供更高的密度，而且还增加了相对于2D ReRAM结构的模拟计算并行性。因此我们提出RAGra利用3D ReRAM的原位计算能力和高并行性加速图计算。

然而，将图算法映射到3D-ReRAM仍然存在极大的挑战。为了追求更高的密度，如图3所示，相邻层共享字线和位线，这对输出提出了新的要求，即输出结果是两个矩阵向量乘的和，而不是诸如2D结构的单个矩阵向量乘。其次，为了节省面积和能量消耗，奇数层和偶数层的所有字线分别共享解码器。这为输入向量带来了新的要求，即所有奇数和偶数层输入都是相同的。为了充分挖掘3D-ReRAM的并行

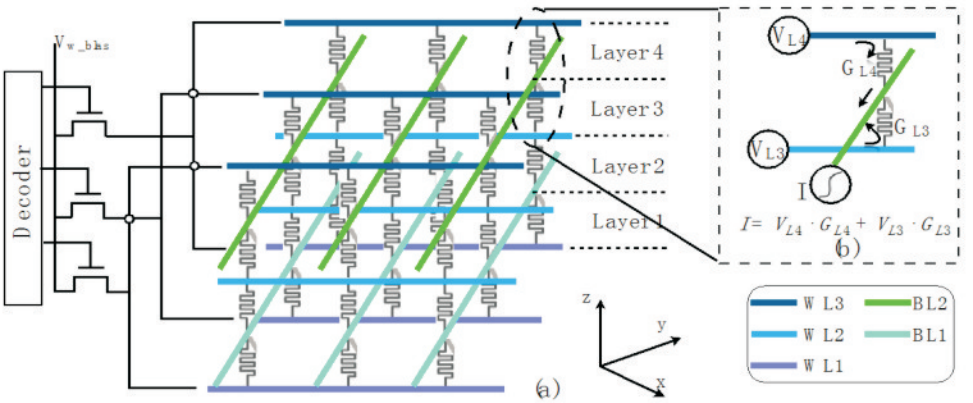


图3 单片3D-ReRAM（a）具有四层堆叠的3D-ReRAM架构（b）3D-ReRAM的计算实例

性，我们重点关注3D-ReRAM的独特输入和输出特征，以3D-ReRAM结构为中心的映射方案，将图算法映射为3D-ReRAM自适应形式，可通过独特的数据组织充分利用3D-ReRAM的计算高并行性。

首先，对于以3D-ReRAM结构为中心的映射方案，根据边之间的聚合操作将图算法分为两类：加和非加。对于加操作的算法（例如PageRank），它可以很好地映射到3D-ReRAM并最大化3D-ReRAM的并行性，称之为全层并行模式。对于非加操作算法（例如BFS和SSSP），我们使用特殊设计来最大化计算效率，我们称之为半层并行模式。更有趣的是，对于BFS判断连通性算法，我们发现它可以转换为全层并行模式算法最大化3D-ReRAM 的计算并行性。

在全层并行模式类型中，图算法边之间的聚合操作是加法操作，这提供了全层并行的可能性，因为3D-ReRAM的最小计算内核是两个乘积的总和。如图4（b），以每层2×2Crossbar的四层3D-ReRAM为例。首先，由于相邻的两个Crossbar共享位线，因此输出的结果是上下Crossbar结算结果的总和。由于边之间的减少操作是加法，可以垂直延伸图上的子图，如图4（a）的黄色部分所示。其次，由于奇数层和偶数层的字线共享输入，所以随着层数的增加，进行水平扩展，如图4（a）的黄色部分所示，例如层3和层4。

在半层并行模式类型中，图算法边之间的聚合操作不是一个加操作。单源最短路径（SSSP）是此模式的典型示例，首先，子图映射到L1，与L1共享位线的L2全部设置为1以执行边处理操作，并且输入是从相应顶点到源的距离，如图4（c）所示。其次，对于3D-ReRAM带来的输入限制，采用与全层并行模式相同的水平扩展方法，第四层Crossbar与第一层共享相同的输入。由于边之间聚合操作与Crossbar行间操作不匹配，在每个时间片处理一行，即一个点对应的边。

由于3D-ReRAM结构的局限性，属于半层并行模式的图算法难以充分利用3D-ReRAM的并行性。然而，我们发现可以将一类无加权连通算法转换为全层并行模式，以利用3D-ReRAM的并行性。广度优先搜索（BFS）是这种模式的一个很好的例子。这类算法边之间的聚合操作通常是比较操作，但是我们发现可以通过将边处理的中间结果先相加，最后在与原值比较，如图4（d）所示。

我们对比了基于2D ReRAM的图加速器GraphR，RAGra默认配置为8层ReRAM。与GraphR相比，RAGra具有改进计算并行性的特点。因此，RAGra的表现优于GraphR，平均加速平均为6.18倍。由于转换后的并行性完全暴

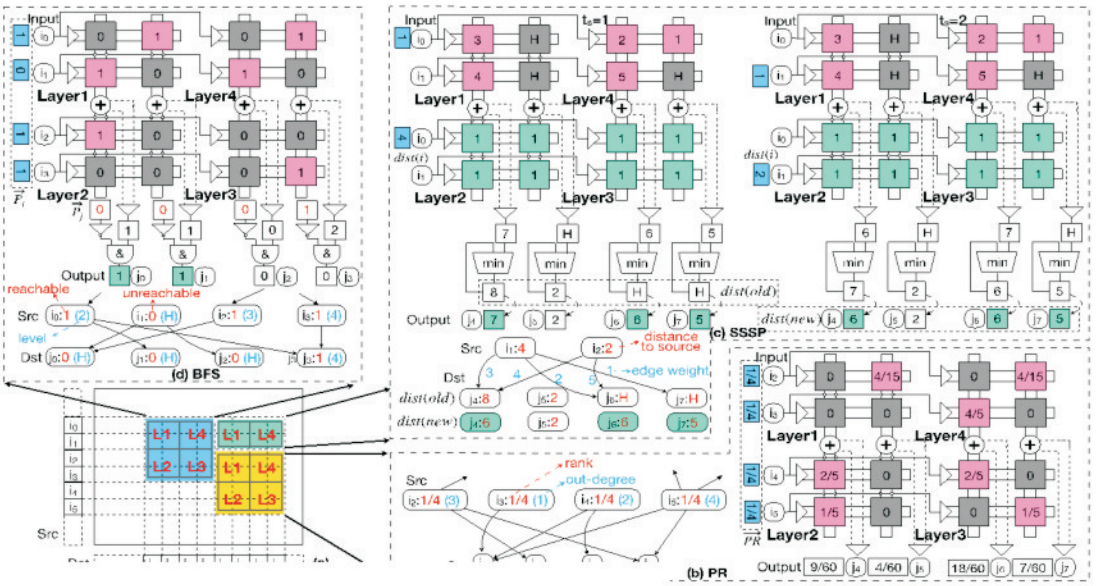


图4 图算法3D-ReRAM映射方案

露，BFS具有高达13.19倍的加速比，显着高于PR和SSSP。我们还评估RAGra相比GraphR的能耗开销，RAGra显示出比GraphR更高的能量效率，平均节省2.21倍。这主要归功于字线和位线的相邻层共用，降低了数模信号间的转换次数，从而节省了能耗开销。

4. 运行时系统

4.1 基于依赖关联性的单图处理任务执行机制

最近出现了许多系统使用GPU加速器进行大规模迭代图处理。然而对于GPU上的迭代有向图处理，现有解决方案的收敛速度慢，数据访问成本高。这是因为许多顶点无法有效地重新处理许多轮，以便根据其他活动顶点更新其状态，而忽略它们之间的依赖性。在本文中，我们在一台支持多个GPU的机器上提出了一种新颖有效的迭代有向图处理系统。与现有系统相比，我们系统的独特之处在于它以三种新颖的方式利用了顶点之间的依赖关系。首先，它表示一组不相交的热/冷定向路径的有向图，并将该路径作为基本的并行处理单元，以便有助于沿着GPU上的路径进行有效的顶点状态传播，以获得更快的收敛速度和更高的加载数据利用率。其次，它尝试根据它们的依赖图的拓扑顺序将路径分派给GPU进行并行处理。然后，许多路径在处理它们一次之后沿着这样的顺序收敛，从而获得较低的再处理开销。第三，在每个流式多处理器上进一步开发路径调度策略，以实现路径（例如，热路径）的特权执行。根据顶点依赖性，缩短收敛时间对顶点状态传播的影响更大。实验结果表明，与最先进的系统相比，我们的方法将迭代有向图处理速度提高了3.54倍。

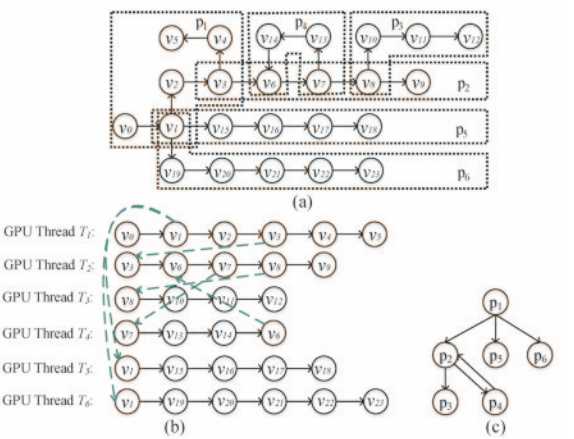


图5 以路径为中心的执行模型解释

设 $G = (V, E)$表示有向图，其中 $V = \{V_t | 0 \le l < n\}$ 是一组顶点，$E = \{< v_t, v_m > | v_t \in V \wedge v_m \in V\}$ 是一组有向边，n是顶点数量。根据我们的观察，由于顶点的更新依赖性，当每个路径的顶点以异步方式在每一轮内沿着该路径上的顺序由单个GPU线程顺序地处理时，每个有向路径的顶点可以快速地将它们的新状态传播到其他路径。为了利用更新依赖性以实现更快的收敛，在我们的模型中，图G由一组不相交的有向路径（没有交叉边）表示，即 $G = P = U_{(x \in V \wedge y \in V)} Path(x, y)$，其中 $Path(x, y)$ 是从顶点x到y的连续有向边的有序序列。我们希望对于 $\forall Path(x, y) \in P \wedge \forall Path(x', y') \in P$ s.t. $Path(x, y) \cap Path(x', y') \subseteq \{x, y, x', y'\}$ 可以确保更多的路径不相互依赖，以减少重新处理的成本。图5（a）给出了一个例子。对于具有相交顶点的任何两个路径，它们彼此依赖并需要重新处理。

然后，它将有向路径作为基本的并行处理单元，且GPU线程同时处理不同的路径。每个有向路径上的图顶点在每一轮内以异步方式按照它们在该路径上的顺序由单个GPU线程顺序处理，图顶点也按照这样的顺序进行存储以便合并访问。每个顶点的新状态立即用于在同一轮中的同一路径上更新其直接后继顶点的状态。通过这种方式，无论路径有多长，每条有向路径上的顶点状态都可以在一轮内沿着该路径到达其他路径。一个顶点状态传播到另一个顶点状态的最大图处理轮数等于该传播所经过的路径数，这通常远小于现有解决方法，比如图的直径。如图5（b）所示，我们的方法使用三个回合沿着三条路径将 的新状态传播到 ，表明收敛速度比现有解法更快。注意，路径之间的状态传播是通过顶点副本的状态同步（例如，顶点 的三个复制顶点）完成的，其中它将分区作为同步单元并且分批进行以实现低开销。然而，一些有向路径依赖于其他路径并且当它们由不同的GPU线程同时处理时可能面临重新处理问题（图5（b）），因为它们的顶点的状态可以基于其他路径的旧顶点状态来更新。为了避免重复处理某些路径，它会为路径生成有效的依赖关系图G', $G'=(V', E')$，其中$V'=\{p_i | p_i \in P\}$。图5（c）说明了如何获得有向图的路径依赖图。然后，它可以通过将此依赖图收缩来获得的DAG。DAG的每个顶点表示一组路径，DAG中的每个边描述两个图顶点之间的更新依赖性，这允许根据此DAG略图的拓扑顺序处理路径。当该图顶点的前继顶点不活动时，尝试将每个图顶点表示的路径集异步调度到GPU以进行并行处理。然后，由于没有来自其他路径的新顶点状态，许多路径不需要再次处理，从而节省了冗余的重新处理。注意，如果存在空闲SMX（例如，可调度路径的数量小于空闲GPU 核心的数量），则还允许预先处理由具有最少数量活动前继顶点的活动图顶点表示的连续路径。通过充分利用GPU的高并行性，使顶点状态更接近最终状态。

4.2 基于无效更新过滤的图计算加速策略

图处理系统大多采用pull模型来处理图迭代的计算密集型部分，具有较高的并行性。考虑到每次迭代都要处理所有的顶点和边，pull模型可能会遇到大量无效的点或边操作，这些操作不会促进图的收敛，从而会使潜在的性能下降。我们认识到可以通过利用一小部分关键信息过滤这些无效的操作。然而，大多数关键信息常常超出了正在处理的活动顶点的可见性。因此我们提出了两种新的过滤方法协同识别不可见的关键信息与边界切割启发式推测来预测图算法，并将这两种方法及其混合解决方案集成到三个最先进的图形处理系统中。

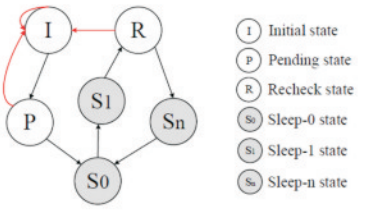


图6 图迭代预测示例

我们遵循收敛指导思想过滤无效操作，基本原理是如果一个给定的顶点已经收敛，那么它的所有剩余的关联边都不需要处理。首先，顶点收敛与图形算法的操作语义密切相关。例如，CC算法中的顶点可以通过获得一个极小的标签来收敛，而SSSP则通过找到最短路径来收敛。其次，顶点收敛性与图的固有拓扑结构密切相关，对于给定的图，如果我们去掉了顶点之间的几个特定边，每个顶点的收敛性在图修改前后可能有显著差异。这也使得在图拓扑信息有限的情况下，很难对顶点收敛做出准确的判断。许多图算法的顶点收敛性可以通过利用少数顶点的一小部分关键信息来判断，即使不是所有顶点都是收敛的，只要找到所有未收敛顶点之间的最短路径，就可以认为一个顶点是收敛的。因此，许多图算法的顶点值是单调变化的，CC算法中的所有顶点都试图找到一个更小的标签。这使得我们能够找到一个严格的边界线，它可以清晰地将顶点划分为绝对收敛集合和不确定集合。一旦一个顶点值落在边界上，就可以判断为立即收敛。同时，对于许多图数据算法来说，在过去几次迭代中没有被激活的顶点具有很高的收敛性，这允许用推测性决策来预测顶点的收敛。

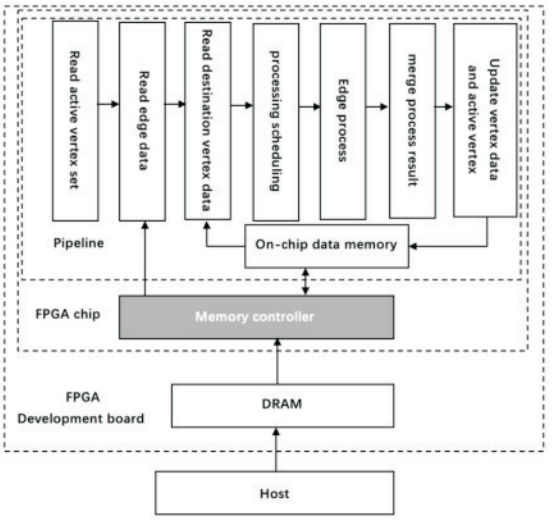


图7 硬件架构

4.3 面向图计算的高层综合系统

图数据结构分析已经成为了人们十分关注的领域，对于这类频繁执行的应用，特定的硬件加速器可以显著地提高性能和能效。然而众所周知，硬件代码的编写和验证十分耗时，所幸高层次综合(High Level Synthesis, HLS)系统的出现允许用户通过高级语言（如C语言）生成硬件结构。然而HLS系统很难从命令式语言中挖掘并行流水机会，使得开发人员难以通过在上层调整高级语言代码从而在底层实现特定硬件架构，底层低效的硬件模板既无法灵活表达上层需求，也不能达到高效性能。现有的研究主要着重于支持各种粒度的流水执行和探寻多循环嵌套中的并行机会，但是简单的片上存储模型难以支持高并行度实现。目前还没有HLS工具可以从上层高级语言为图算法直接生成高并行度的高效流水结构。一方面因为其表达能力有限，用户很难准确描述所要的架构并指定微架构参数；另一方面对图算法指定高并行度后，会产生大量数据依赖和冲突，没有足够的底层优化去实现高并行度所需的存储和计算结构，导致最后生成的硬件电路实际串行执行或者资源耗尽无法生成。

为了解决HLS系统缺乏对图计算有效支撑的问题，考虑到图算法自身的嵌套循环、大量随机访存、并行带来的严重数据冲突以及图数据幂律分布的特性，我们提出了一种面向图计算的高效HLS方法，采用了数据流架构来实现高效的并行流水执行，保证了处理单元的负载均衡。方法主要包含三个部分，分别是上层编程模型、模块化数据流IR和底层参数化硬件模版。

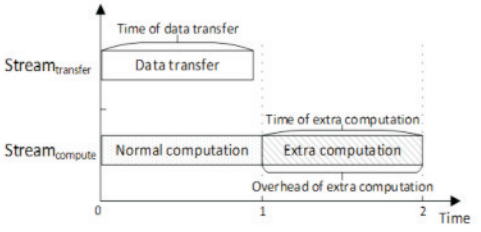
编程模型突出了七个主要的图操作，允许用户直接在每个操作上添加特定的架构和微架构参数，用户只需使用提供的编程原语和优化指令描述图操作，无需命令式地指定数据应该如何存放，何时去何处取数据，十分简洁高

效；模块化数据流IR为图算法定制了高效的数据通路和存储模型，不需要复杂的调度算法即可为不同的图算法定制高效架构；底层参数化硬件模版为重要的图操作提供高效的模块化硬件实现，保证数据流架构中各模块间的有效配合；通过我们提供的编程原语可快速定制以点为中心的图算法并转化为模块化的数据流中间表示，进而映射到参数化的硬件模版，硬件实现的架构如图7所示。我们在Xilinx Virtex UltraScale+ XCVU9P现场可编程门阵列【=Field-Programmable Gate Array，FPGA】上测试了生成的硬件代码，验证了此方法具有显著的效果。

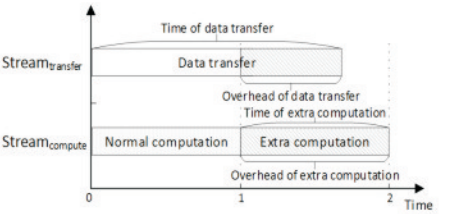
5. 应用支撑

5.1 深度学习系统内存重用及优化方法

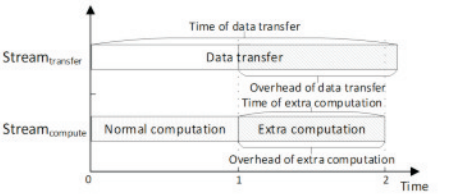
在深度学习中，GPU通常用于加速深度神经网络的训练过程，然而GPU有限的物理内存意味着它很难训练大规模的深度神经网络模型。现有的内存优化方法主要包括重新计算方法和CPU-GPU迁移方法，它们对神经网络中的所有层应用单一的优化方式，不能实现理想的训练性能。因此我们提出一种基于层的内存复用和优化方法Layup，以进一步节约GPU显存消耗。我们的方案主要基于以下两个观察：第一，不同层对于计算资源和内存空间需求并不总是保持一致，因此需要设计以层为粒度的优化方案以实现更小的性能开销；第二，在神经网络的训练过程中会产生多种类型的中间数据，为了实现更好的内存效率，需考虑中间数据内存消耗问题。



(a) Threshold ≤ 1



(b) 1 ≤ Threshold ≤ 2



(c) Threshold ≥ 2

图8 额外计算与数据传输之间的阈值分析

Layup方法包含两种策略：（1）层自适应策略选择机制。为神经网络中不同的层选择合适的内存优化方式，降低训练时的性能开销。结合GPU异步执行的特性，通过分析神经网络中不同层的迁移开销和重新计算开销，将神经网络中的层划分为计算敏感型和迁移敏感型两类，使用CPU-GPU迁移方法优化计算敏感型层的特征图，使用重新计算方法优化迁移敏感型层的特征图，同时使用流水线并行的方式将数据传输过程和计算过程重叠起来。（2）基于多种中间数据的内存复用策略。首先，使用滑动窗口的方式复用梯度图的内存空间；同时，基于神经网络逐层计算的特性，逐层复用卷积工作空间和cuDNN句柄数据，以进一步降低深度神经网络训练时的内存使用量。

实验表明，Layup方法可以显著降低深度神经网络训练时的内存消耗，同时保持较低的性能开销。训练时的内存消耗量最高减少92%左右，同时，在所测试的神经网络模型上的性能开销平均仅为12%。特别地，在内存容量为12GB的GPU上可以训练高达2504层的ResNet模型（批量大小为16），与SuperNeurons方法相比提升约30%，进一步扩展了单个GPU上神经网络模型的规模。

5.2 分层细粒度深度神经网络并行优化方法

深度学习模型通常要进行反复的迭代式训练，需要消耗大量的时间与计算资源，因此引入并行化方法对于深度学习意义很大。现有的并行化方法主要包括数据并行和模型并行。这两种模型级别的粗粒度并行方式通常是通过增加训练单元的数目来获得加速，不能充分挖掘模型自身的可并行性与及硬件资源。

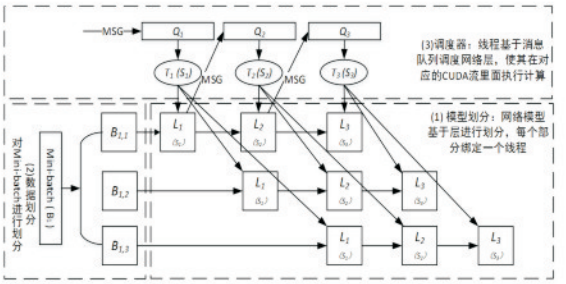


图9 层间并行数据流水线算法的描述

针对上述问题，提出了一种GPU众核平台下深度学习中基于层的细粒度并行方法FiLayer，具体包括层间并行与层内并行。前者旨在让模型中的相邻层并行处理，后者旨在并行化网络层内部的计算。为了实现层间并行，设计并实现了数据流水线算法。该算法先将网络模型按层划分成多个部分，其次将训练数据划分多个分片，最后以流水线的方式让多个网络层并行计算多个分片数据。为了实现层内并行，设计并且实现了并行化卷积算法，包括并行化卷积前向计算算法和并行化卷积后向计算算法。并行化卷积算法将卷积层的计算划分成多个部分，分发给不同的部件进行计算。CUDA流技术被应用到了FiLayer的实现中，保证了FiLayer在物理上真正的并行执行。相对于现有的基于模型的粗粒度并行方法，FiLayer充分挖掘了模型自身的可并行性和训练单元上的计算资源，在保证网络模型训练精度的前提下，提高模型的收敛速度。

上述方法实现在Caffe系统上，拓展后的系统命名为LP-Caffe。采用四个有代表性的标准数据集（MNIST, CIFAR10, CIFAR100及ImageNet）在三个不同的硬件环境上对所提出的方法进行了性能评估。实验结果表明，FiLayer在GPU上能帮助Caffe能获得1.58x-2.19x的加速。

5.3 异构内存对图计算的支撑

传统的图计算系统不区分内存的异构性，数据随机的放置在两种介质的内存中。对NVM中数据的频繁访问成为降低系统性能的关键因素，因此在异构内存系统上具有次优的性能。同时在CMP系统中并行处理图计算的时候，会有多个处理器对同一个数据进行更新，为了保证结果的正确性以及避免数据的竞争，常采用的原子操作对数据进行更新，其他处理器核处理完带来的处理器等待以及原子操作对整个缓冲行加锁操作，都会带来很大的性能开销。针对以上问题我们提出了异构内存感知的并行图计算系统NGraph。不同于此，我们通过对图数据和任务进行划分，避免多个处理器处理同一个数据所带来的数据竞争，从而也避免了原子更新带来的额外开销。此外，我们的数据和任务划分策略，也可以避免多核更新数据对数据预取效果的影响。

NGraph分析了图计算中不同类型的数据结构的访问特性，把只读，顺序访问的数据放置在NVM中，以降低NVM高延迟带来的性能下降。同时通过对图数据和任务进行划分，避免多个处理器处理同一个数据所带来的数据竞争，从而也避免了原子更新带来的额外开销。此外NGraph还采用了数据结构感知的数据预取和大页分配两个优化方式进一步隐藏NVM内存访问延迟。

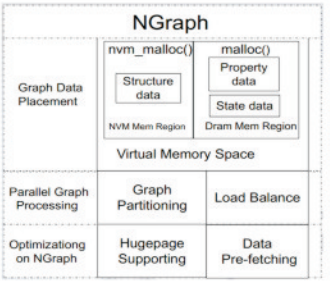


图10 Ngraph整体架构

异构内存环境下图计算数据放置：图结构在程序运行当中保持不变，因而图结构数据是只读的。自然图中，图结构数据比顶点属性以及状态数据大很多，通常占用71%到83%的比例。由于图结构数据是顺序存放于数组中，其访问也是顺序访问，具有较好的空间局部性，因此顺序只读模式的图结构数据天然适合存储于大容量NVM介质中。图计算中，通常是根据边来更新相邻顶点上属性数据，由于图的不规则性，这些更新分散在整个图上，造成了这些顶点的随机访问，因此随机读写模式的顶点属性数据适合放置在DRAM内存中。此外，运行状态数据很小，在每一轮迭代中会重新分配内存（顶点状态数组），并且频繁访问与更新，因此也应该放置在DRAM内存中。

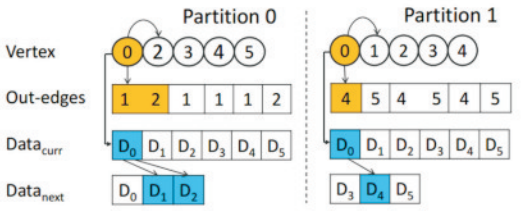


图11 按目标顶点划分子图

异构内存环境下图计算并行优化：为了避免数据竞争带来的处理器等待问题，我们采用按照目标顶点划分边到不同的子图来解决这个问题。如下图所示，目标顶点为0、1、2的7条边划分到partition 0，目标顶点为3、4、5的另外7条边划分到partition 1。每个线程依次处理图中的所有顶点，但是只更新目标顶点在这个partition中的边所对应的的邻居顶点，这样就可以有效消除不同的线程同时更新同一个顶点或者相邻顶点数据所造成的数据竞争。

边数据平均划分并不能达到很好的负载均衡，因为只有活跃顶点对应的边才真正需要进行计算，我们采用work stealing技术进一步提升多核间任务的负载均衡。我们把每个子图中的数据进行分块，每个线程从头到尾依次处理每块数据，当有线程处理完对应的子图后，检查是否有其他子图还没处理完，如果没处理完，从其最后一个数据块开始协助处理未完成处理的线程。直到所有子图全部处理完，程序进入下一轮迭代。

6. 总结

系统软件与体系结构组小组的典型成果涵盖了体系结构、运行时系统和应用支撑三个不同层面，尤其是在内存计算、图计算和深度学习等方面有较强的研究特色，形成了一批有代表性的开源系统、论文和专利。小组除了在现有优势领域继续深入研究之外，在数据流体系结构和医疗健康大数据分析等新兴领域也开展了探索性研究，并取得了初步的研究成果。

附成果列表：

主要专利：

[1] 一种基于NUMA架构的大规模流式图数据更新方法，专利申请号201910368729.7，专利申请日2019年05月05日，邵志远，金海，廖小飞，赵智慧，华中科技大学

[2] 一种异构内存虚拟机不停机的内存页面迁移系统，专利申请号201910315352.9，专利申请日2019年04月18日，刘海坤，廖小飞，金海，杨荡，华中科技大学

[3] 基于异构内存的可扩展内存对象存储系统，专利申请号201910315354.8，专利申请日2019年04月18日，刘海坤，廖小飞，金海，李志威，华中科技大学

[4] 一种硬件支持的3D堆叠NVM内存数据压缩方法及系统，专利申请号201910153736.5，专利申请日2019年02月28日，刘海坤，廖小飞，金海，叶源远，华中科技大学

[5] 一种基于极大团枚举的社交网络中社团发现方法，专利申请号201910142643.2，专利申请日2019年02月26日，邵志远，金海，廖小飞，李屹诺，华中科技大学

[6] 一种基于层宽内存分配的深度学习异构计算方法和系统，专利申请号201910136545.8，专利申请日2019年02月22日，金海，廖小飞，郑龙，

刘海坤，葛希，华中科技大学

[7] 一种基于FPGA的图数据处理方法和系统，专利申请号201910084423.9，专利申请日2019年01月29日，廖小飞，金海，郑龙，杨成波，华中科技大学

[8] 一种H.265/HEVC中8x8子宏块的并行帧内预测方法，专利号ZL201510933636.6，专利申请日2015年12月15日，授权公告日2019年06月28日，蒋文斌，金海，池也，华中科技大学

代表性论文：

[1] Yu Zhang, Xiaofei Liao, Hai Jin, Bingsheng He, Haikun Liu, Lin Gu. DiGraph: An Efficient Path-based Iterative Directed Graph Processing System on Multiple GPUs. Proceedings of the International Conference on Architectural Support for Programming Languages and Operating Systems ASPLOS. (2019): 601-614

[2] Jin Zhao, Yu Zhang, Xiaofei Liao, Ligang He, Bingsheng He, Hai Jin, Haikun Liu, Yicheng Chen. GraphM: An Efficient Storage System for High Throughput of Concurrent Graph Processing, Proceedings of 2019 International Conference for High Performance Computing, Networking, Storage, and Analysis SC. 2019, accepted.

[3] Hai Jin, Bo Liu, Wenbin Jiang, Yang Ma, Xuanhua Shi, Bingsheng He, Shaofeng Zhao. Layer-centric Memory Reuse and Data Migration for Extreme-Scale Deep Learning on Many-Core Architectures. ACM Transactions on Architecture and Code Optimization TACO. 15, 3 (2018), 37:1-37:26

[4] Xiaoyuan Wang, Haikun Liu, Xiaofei Liao, Ji Chen, Hai Jin, Yu Zhang, Long Zheng, Bingsheng He, and Song Jiang. 2018. Supporting Superpages and Lightweight Page Migration in Hybrid Memory Systems. ACM Transactions on Architecture and Code Optimization TACO. Vol. 16, No. 2, April, (2019), 11: 1-26

[5] Wenbin Jiang, Yang Ma, Bo Liu, Haikun Liu, Bingbing Zhou, Jian Zhu, Song Wu, and Hai Jin, Layup: Layer-Adaptive and Multi-Type Intermediate-Oriented Memory Optimization for GPU-Based CNNs, ACM Transactions on Architecture and Code Optimization TACO, accepted.

[6] Yu Zhang, Jin Zhao, Xiaofei Liao, Hai Jin, Lin Gu, Haikun Liu, Bingsheng He, and Ligang He. CGraph: A Distributed Storage and Processing System for Concurrent Iterative Graph Analysis Jobs. ACM Transactions on Storage TOS, 15(2): 10:1-16, 2019.

[7] Xiaofei Liao, Yu Huang, Long Zheng, Hai Jin. Efficient Time-Evolving Stream Processing at Scale. IEEE Transactions on Parallel and Distributed Systems TPDS, accepted.

[8] Long Zheng, Xianliang Li, Xi Ge, Xiaofei Liao, Zhiyuan Shao, Hai Jin, Qiang-Sheng Hua. Efficient Graph Processing with Invalid Update Filtration, IEEE Transactions on Big Data TBD, accepted.

[9] Xiaofei Liao, Zhan Zhang, Haikun Liu, Hai Jin, Improving Bank-level Parallelism for In-Memory Checkpointing in Hybrid Memory Systems, IEEE Transactions on Big Data TBD, accepted.

[10] Zhiyuan Shao, Ruoshi Li, Diqing Hu, Xiaofei Liao, Hai Jin. Improving Performance of Graph Processing on FPGA-DRAM Platform by Two-level Vertex Caching FPGA. (2019): 320-329

[11] Yu Huang, Long Zheng, Xiaofei Liao, Hai Jin, Pengcheng Yao, Chuangyi Gui. RAGra: Leveraging Monolithic 3D ReRAM for Massively-Parallel Graph Processing. Design, Automation & Test in Europe Conference DATE. (2019): 1273-1276

[12] Li Lin, Peng Li, Xiaofei Liao, Hai Jin, Yu Zhang. Echo: An Edge-Centric Code Offloading System With Quality of Service Guarantee. IEEE Access 7: 5905-5917 (2019)

[13] Wei Liu, Haikun Liu, Xiaofei Liao, Hai Jin, and Yu Zhang. NGraph: Parallel Graph Processing in Hybrid Memory Systems. IEEE Access, accepted.

[14] Chuangyi Gui, Long Zheng, Bingsheng He, Cheng Liu, Xinyu Chen, Xiaofei Liao, Hai Jin. A Survey on Graph Processing Accelerators: Challenges and Opportunities. Journal of Computer Science and Technology JCST. 34(2): 339-371 (2019)

[15] Wenbin Jiang, Yangsong Zhang, Pai Liu, Geyan Ye, and Hai Jin. FiLayer: A Novel Fine-grained Layer-wise Parallelism Strategy for Deep Neural Networks. The 27th International Conference on Artificial Neural Networks ICANN 2018

[16] Wenbin Jiang, Geyan Ye, Laurence T. Yang, Jian Zhu, Yang Ma, Xia Xie, and Hai Jin, A Novel Stochastic Gradient Descent Algorithm Based on Grouping over Heterogeneous Cluster Systems for Distributed Deep Learning, The 19th Annual IEEE/ACM International Symposium in Cluster, Cloud, and Grid Computing CCGrid2019

[17] Wenbin Jiang, Yangsong Zhang, Pai Liu, Jing Peng, Laurence T. Yang, Geyan Ye, Hai Jin, Exploiting Potential of Deep Neural Networks by Layer-Wise

Fine-Grained Parallelism, Future Generation Comp. Syst. FGCS, accepted.
[18] Xuepeng Fan, Xiaofei Liao, Hai Jin. FunctionFlow: coordinating parallel tasks. Frontiers Comput. Science FCS. 13(1): 73-85 (2019)
[19] Xinqiao Lv, Wei Xiao, Yu Zhang, Xiaofei Liao, Hai Jin, Qiang-Sheng Hua. An effective framework for asynchronous incremental graph processing. Frontiers Computer Science FCS. 13(3): 539-551 (2019)
[20] Chengbo Yang, Long Zheng, Chuangyi Gui, Hai Jin. Efficient FPGA-based Graph Processing with Hybrid Pull-Push Computational Model. Frontiers of Computer Science FCS, accepted.
[21] Chenxi Li, Xiaofei Liao, Hai Jin. Enhancing application performance via DAG-driven scheduling in task parallelism for cloud center. Peer-to-Peer Networking and Applications 12(2): 381-391 (2019)
[22] Saqib Qamar, Hai Jin, Ran Zheng, Parvez Ahmad. Multi stream 3D hyper-densely connected network for multi-modality isointense infant brain MRI segmentation, Multimedia Tools and Applications MTAP, accepted.
[23] Parvez Ahmad, Hai Jin, Saqib Qamar, Ran Zheng, Wenbin Jiang, Belal Ahmad, Mohd Usama, 3D Dense Dilated Hierarchical Architecture for Brain Tumor 3. Segmentation. 2019 4th International Conference on Big Data and Computing ICBDC. (2019)
[24] Yinuo Li, Zhiyuan Shao, Dongxiao Yu, Xiaofei Liao, Hai Jin. Fast Maximal Clique Enumeration for Real-World Graphs. Database Systems for Advanced Applications - 24th International Conference DASFAA. (2019): 641-658
[25] Xiang Shi, Xiaofei Liao, Dayang Zheng, Hai Jin, Haikun Liu. VMBKS: a shared memory cache system based on booting kernel in cloud. The Journal of Supercomputing 75(1): 4-19 (2019)
[26] 廖小飞, 陈意诚, 张宇, 金海, 刘海坤, 赵进.一种高效的面向动态有向图的增量强连通分量算法. 中国科学:信息科学, 2019, accepted.

刘海坤
副教授、博导
研究方向：虚拟化，内存计算
Email：hkliu@hust.edu.cn

李若时
2016级硕士
研究方向：系统软件与体系结构
Email：liruoshi622@qq.com

张　宇
副教授
研究方向：体系结构与系统软件、运行时系统及其优化
Email：zhyu@hust.edu.cn

郑　龙
副教授
研究方向：可重构计算机体系结构及其运行环境
Email：longzh@hust.edu.cn

马　阳
2016级硕士
研究方向：深度学习系统性能优化
Email：mayangmails@qq.com

张杨松
2015级硕士
研究方向：深度学习系统性能优化
Email：1107211209@qq.com

蒋文斌
教授
研究方向：深度学习、体系结构与系统软件
Email：wenbinjiang@hust.edu.cn

廖小飞
教授、博导，实验室副主任
研究方向：系统软件和系统结构
Email：xfliao@hust.edu.cn

网络空间安全组典型成果介绍

邹德清，马晓静，徐　鹏，羌卫中，代炜琦，袁　斌

关键词：云计算安全，软件安全，数据与隐私安全

1. 介绍

网络与信息安全作为国家高度重视的核心技术，已经成为国内外的热点领域。随着被正式获批为国家一级学科，网络空间安全在国内的工业界和学术界掀起又一波研究热潮。围绕当前热门的IT技术引入的安全挑战，本组主要做了如下三个方面的研究：（1）云计算安全；（2）软件安全；（3）数据与隐私安全。下面主要介绍本组在这三个方面研发的一系列典型系统和技术：

（1）在云计算安全方面，容器技术和微服务架构的应用增加了云环境的动态性和复杂性，这种动态性和复杂性也导致云环境中安全问题的不确定性，基于固定策略的防御机制无法有效应对容器云中的安全问题。为了解决这个问题，我们设计了一个框架Dynamic Security Evaluation and Optimization of MTD（DSEOM），该框架可以自动感知容器云中的更新事件，快速评估主动防御技术有效性的变化，并进一步根据当前场景优化其实施策略。控制流完整性（Control-Flow Integrity, CFI）保护是一种针对控制流劫持攻击的有效方法之一，传统工作主要采用粗粒度的、上下文不敏感的CFI方法，使得某些被精心设计的攻击能够乘虚而入。因此设计了一种上下文敏感的增量式CFI，充分利用了云环境的特性，可以抵御大多数常见的控制流劫持攻击，同时只引入较低的运行时性能开销。

（2）在软件安全方面，当前检测并发程序的方案主要集中在检测如数据竞争、原子违例等并发错误上，很少有方案关注到更能造成安全问题的并发漏洞的检测上。因此提出了一个结合程序静态分析和模糊测试的轻量级并发漏洞检测框架，增大了漏洞被触发的概率，成功发现了并发漏洞。

（3）在数据与隐私安全方面，为了解决数据交易平台中存在的安全问题，设计并实现了基于区块链的安全数据交易生态系统Secure Data Trading Ecosystem（SDTE），提供了一种数据安全交易机制。针对传感器设备采集的海量数据面临的严峻安全性问题，提出了一种适用于云辅助无线传感器网络的轻量级可搜索公钥加密方案LSEP，用于保护云中的传感器数据，以及实现对这些数据的安全、高效访问。云上多媒体数据一般是压缩格式存放，因此云端在处理时需先对其进行解压缩，对同态域多媒体数据进行解压缩是一个巨大的挑战。针对此挑战，提出了同态加密FLAC音频密文下解压缩方法，提高了解压缩效率和降噪效率，并且适用于其他类似编码技术的音频压缩标准。

2. 成果介绍

2.1 基于容器云的动态防御机制优化

当下，容器技术和微服务架构的结合使得容器云相较于基于虚拟机的云环境更高效与灵活。容器技术和微服务架构的广泛应用虽然极大地提升了云环境的性能，但它们带来的这些变化是否会影响云环境的安全性依旧有待研究。与基于虚拟机的云环境相比，微服务架构使得服务的扩展和更新变得更加灵活，每项服务的更新频率每天可高达数百次。遗憾的是，这种灵活性也大幅地增加云环境的动态性和复杂性，同时也为云

环境引入一些新的安全问题。具体而言，随着服务的高频更新，云环境中的安全隐患也随着发生了变化。具有固定策略的防御机制很难在这样动态环境中持续覆盖不断变化的安全隐患。因此，防御机制的有效性会随着时间的推移而波动（增加或减少），我们称之为有效性漂移问题。该问题对于主动防御机制的影响尤为显著。通常主动防御机制的有效性高度依赖于防御目标的现状，如果防御策略与假设条件不匹配，则主动防御机制可能效果不佳甚至失效。

为了解决这个问题，我们设计了一个框架DSEOM。该框架可以自动感知容器云中的更新事件，快速评估主动防御技术的有效性变化，并进一步根据当前场景优化其实施策略。考虑到云环境的复杂性和攻击行为的多样性，我们建立了多维攻击图来形式化指定容器云中的所有可行攻击事件。并且基于攻击者的行为模式，引入了介中心度的概念，与攻击图相结合帮助我们定位当前云环境中的安全薄弱点，以快速评估主动防御策略有效性漂移的问题，同时优化主动防御策略的实施规则，使其有效性最大化。

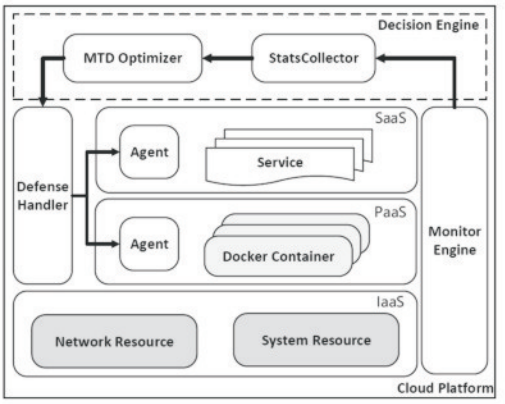


图2.1 DSEOM动态优化框架

为了有效地展示DSEOM的可用性，我们定义了一组安全性和性能指标，以量化和比较DSEOM优化后的主动防御策略与初始策略之间的有效性差异。实验结果表明DSEOM可以在基于容器的云环境中以可接受的成本快速优化和维护MTD策略的有效性。

2.2 云环境中上下文敏感的增量式控制流完整性研究

控制流完整性（Control-Flow Integrity, CFI）保护是一种针对控制流劫持攻击的有效方法之一，该类方法利用控制流图（Control-Flow Graph, CFG）检查控制流的合法性。对于常见的二进制程序，传统工作主要采用粗粒度的、上下文不敏感的CFI方法，但是，上下文不敏感的CFI方法不能生成足够精确的CFG，使得某些被精心设计的攻击能够乘虚而入。另一方面，现有的部分上下文敏感的CFI（Practical Context-Sensitive CFI）方法虽然部分地考虑了上下文，但由于采用了一些上下文不敏感的规则，导致生成的CFG仍包含一些不合法的控制流边，易被攻击者利用。

CloudCFI是一种上下文敏感的增量式CFI，系统架构如图2.2所示，它充分利用了云环境的特性，考虑同一软件的多个实例在多个虚拟机上运行的典型部署场景，来自一个软件实例的控制流检查结果可能是用于处理在其它实例上发生的控制劫持防护。在CloudCFI中，可以持续增加控制流检查的准确性以提供增量式CFI，并且，通过检查整个执行路径（而不是执行路径中的单边或部分边）来实施上下文敏感的CFI策略以确定执行路径的控制流的有效性。CloudCFI分别包括静态阶段和运行时阶段。通过静态阶段中的模拟执行来收集控制流信息和基本块信息，并且在运行时阶段跟踪执行路径以收集进程跟踪信息。接下来，它通过使用基本块信息和进程跟踪信息恢复执行路径，并使用控制流信息检查控制流的有效性。

原型系统的评估使用了RIPE和SPEC基准测试程序，以及真实应用程序Memcached和Redis。评估结果表明，CloudCFI可以抵御大多数常见的控制流劫持攻击。同时，它只引入了较低的

运行时性能开销。

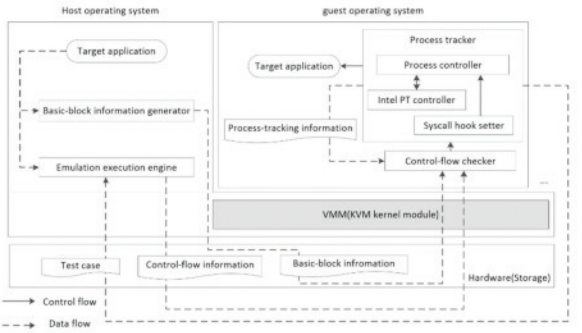


图2.2 CloudCFI的系统架构

2.3 基于模糊测试的并发漏洞挖掘

随着对并发系统算力的要求越来越大，对并发软件的需求也越来越多。因此由并发所引起的漏洞就成为了现今IT设施的不可避免的威胁之一。并发程序可以提供比串行程序更多的计算能力，并且已经应用于许多如云服务等十分关键的应用程序上。但是，并发程序也容易导致可能造成严重后果的并发漏洞，如脏牛漏洞，这是Linux内核中发现的一个典型的由并发引起的漏洞，攻击者利用这个漏洞专门针对并发程序进行攻击。然而，检测并发错误和漏洞具有较高的难度，因为在并发程序中存在太多的线程调度，穷尽一个程序可能的线程调度几乎是不可能的。现今绝大多并发问题检测都集中在并发错误上，并发错误可能造成程序崩溃，或者一些不会引发安全问题的漏洞，而很难触发并发漏洞，针对这些问题我们提出了一个针对并发漏洞的轻量级检测框架。

当前检测并发程序的方案主要集中在检测如数据竞争、原子违例等并发错误上，很少有方案关注到更能造成安全问题的并发漏洞的检测上。实验室漏洞分析小组提出了一个结合程序静态分析和模糊测试的轻量级并发漏洞检测框架。程序先由静态分析得到可能引发并发漏洞的敏感操作，然后利用模糊测试按照会引发并发漏洞的顺序去反复执行程序，提高了并发漏洞被触发的概率。最后在6个真实程序上检测出了3个并发漏洞和3个并发错误。

系统流程图如图2.3所示，我们先对程序进行静态分析得到可能并发执行的敏感操作，之后再针对这些敏感操作着重地进行模糊测试。众所周知，对于一个并发程序，一个并发漏洞的触发除了需要特定的程序输入，往往还需要执行特定的敏感操作，以及这些操作按照特定顺序去执行。在实际执行的过程中，由于并发带来的不确定性，这些条件不一定会被满足。故本文先对其进行静态检测，得到一系列能触发并发漏洞的敏感操作，以及这些敏感操作之间的执行顺序，再通过插桩使得程序每次都会按照预先设定的顺序去执行，大大提高了漏洞被触发的概率。具体的算法执行如图2.4所示，如果通过静态分析发现一对敏感操作按照某个特定顺序执行（如A必须先于B执行）会触发并发漏洞，则如果先执行到A敏感操作，则不做任何操作，如果程序先运行到B，则先将B的优先级调整为最低，只有当程序执行到先A后B时优先级才会被恢复，并当B被执行之后计数器才加一。

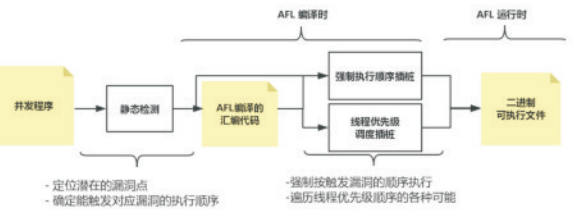


图2.3 系统流程图

其次，由于AFL缺少对线程的感知，不会对线程有特殊的操作，由经典的并发错误检测领域的工作可以得出，遍历各个线程之间的调度有助于提高并发错误被触发的概率。故本文在AFL的基础上加入了线程级别的优先级主动调度，使得AFL在模糊测试过程中尽可能多地遍历到各种线程之间的调度可能性，从而提高了并发错误被触发的可能性。

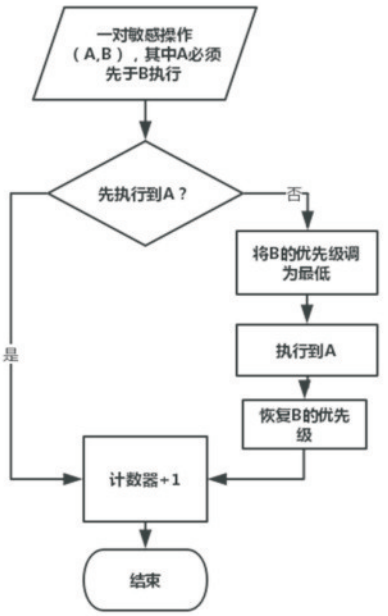


图2.4 并发漏洞模糊测试策略

本文将这个框架用Ocaml和C语言实现，其中静态程序分析部分是基于流行的开源静态程序分析框架C Intermediate Language（CIL），而动态模糊测试所采用的是灰盒模糊测试工具AFL。我们首先通过静态分析定位到了可能存在并发关系的敏感操作，并在对应的敏感并发操作上加上标注，然后我们改动了AFL使得AFL能识别这些标注，并且能按照特定的并发顺序去执行被标识的并发敏感操作，增大了漏洞被触发的概率，最终在六个真实程序中测试。经过测试得到了之前没有发现的三个并发错误和两个并发漏洞。

2.4 基于区块链的安全数据交易系统

在数据交易方面，针对数据交易过程中不诚实数据买方或第三方数据交易平台倒卖、盗卖卖方数据的问题，提出了一种基于区块链以及可信硬件的安全数据交易生态系统SDTE，确保买方仅能通过在SDTE中部署数据分析智能合约来处理数据卖方的源数据，使其只能得到数据分析结果而无法获得源数据，同时使用了SGX保护智能合约执行环境（EVM虚拟机），并且源数据和执行结果在安全环境之外都以密文的形式传输，解决了由于区块链公开透明导致交易平台泄露数据买卖双方隐私数据的问题。

数据是数据驱动型经济的一项核心资产，其日益重要的作用推动了数据交易产业的快速发展。数据交易过程依赖于数据交易平台作为数据买方和数据卖方的纽带，数据卖方的数据在数据交易平台的帮助下流向数据买方。目前数据交易平台存在两个主要问题：首先，不诚实的数据买方在获得卖方的源数据之后，可能会将数据卖方的源数据转卖给其他人；其次，当数据卖方的源数据流经数据交易平台时，不诚实的数据交易平台可能在未经数据卖方允许的情况下缓存数据卖方的源数据并对其进行转卖。

为了解决数据交易平台中存在的安全问题，设计并实现了基于区块链的安全数据交易生态系统SDTE，提供了一种数据安全交易机制，如图2.5所示。在SDTE中，买方通过在SDTE中部署数据分析智能合约来处理数据卖方的源数据，这样数据买方只能够得到数据分析结果而无法获得数据卖方的源数据，同时禁用了智能合约之间的调用功能以及对智能合约的输出结果计费，防止数据买方通过恶意智能合约窃取源数据，进一步保护了卖方源数据的安全性。由于区块链本身具有公开透明，当数据分析合约在Secure Data Trading Platform（SDTP）上处理数据卖方的源数据时，这一过程对外部来说是可见的，为了解决这个问题，SDTP使用了SGX保护智能合约执行环境（EVM虚拟机），并且源数据和执行结果在安全环境之外都以密文的形式传输，进而解决了交易平台泄露数据买卖双方隐私数据的问题，同时通过部署的数据交易管理合约记录交易信息，防止交易过程中角色的抵赖与欺诈。最后SDTE是一个区块链平台，当其中的一个或者部分节点出现故障时，并不影响整个智能合约执行过程，因此解决了

数据交易过程中单点故障问题。综上，SDTE同时解决了数据买方和数据交易平台对源数据的转卖以及数据交易平台的单点故障问题。

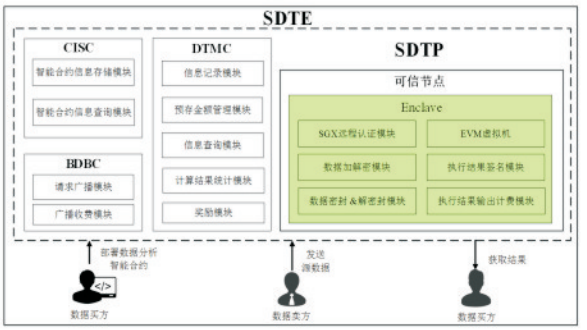


图2.5 SDTE体系结构

针对基于区块链的安全数据交易生态系统进行了安全分析和性能测试。在安全分析中，对每个角色都可能发动的DoS攻击、单个角色可能发送的攻击以及多个角色合谋时能发动的攻击进行了安全分析，表明了SDTE能够抵抗这些攻击，具有较强的安全性。在性能测试中，从密钥安全传输、可信化环境SGX内执行智能合约以及总运行时间进行性能测试和分析，测试结果表明使用SGX保护智能合约这一关键模块使用K-Nearest Neighbor (KNN)算法处理数据时，额外消耗的时间小于0.5S，同时系统带来的其他额外开销也在可接受范围之内。

2.5 轻量级可搜索公钥加密及其在云辅助的无线传感网中的应用

云辅助无线传感器网络广泛应用于各类领域，包括农业、国防、环境监测以及指挥交通等等，但是由传感器设备采集的海量数据面临着严峻的安全性问题。基于此，我们提出了一种适用于云辅助无线传感器网络的轻量级可搜索公钥加密方案LSEP（Lightweight Searchable Public-Key Encryption），用于保护云中的传感器数据，如图2.6所示，以及实现对这些数据的安全、高效访问。

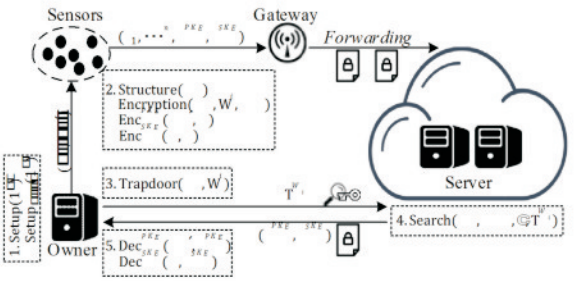


图2.6 LSPE在云辅助的无线传感器网中的应用

无线传感器网络（Wireless Sensor Networks，WSN）在云计算的帮助下快速发展，使得工业物联网正在蓬勃发展。新技术将给网络安全带来新风险，特别是云辅助无线传感器网络（Cloud-assisted WSNs，CWSN）中的数据机密性问题。可搜索的公钥加密（Searchable Public-key Encryption，SPE）是解决此问题的有效方法。理论上，它允许传感器将公钥密文上传到云，这些传感器的所有者可以安全地将关键字搜索委托给云并检索预期数据，同时保持数据机密性。但是，所有现有的和语义上安全的SPE方案在生成密文和搜索关键字方面都有昂贵的成本。因此，本文提出了一种轻量级SPE方案LSPE，该方案具有CWSN的语义安全性。LSPE减少了以前工作中采用的大量计算密集型操作，因此，LSPE的搜索性能接近于一些实用的可搜索对称加密方案。此外，LSPE节省了用于生成密文的传感器的大量时间和能量成本。最后，我们通过实验测试LSPE，并将结果与之前的一些工作进行比较，以定量地证明上述优点。

LSPE方案沿用了徐鹏等人2015年提出的结构化可搜索公钥加密的思想（相关论文Generating Searchable Public-Key Ciphertexts With Hidden Structures for Fast Keyword Search，IEEE Transactions on Information Forensics and Security，2015，在下文中简称该工作为XW15），同一个关键字的可搜索密文具有隐藏的链式结构，这

样可以达到亚线性级的检索复杂度。我们基于双线性映射pairing去构造LSPE方案，并在适应性选择关键字和结构攻击的模型SS-CKSA下，基于计算性双线性迪菲赫尔曼假设CBDH完成了方案的安全性证明。为了降低加密开销并提升检索效率，采用混合加密的思想，用轻量级的乘法和除法操作取代计算昂贵的复杂密码操作，例如pairing。在LSPE方案中，完成同一个关键字的N次加密只需执行一次pairing操作和N+1次群元素的乘法，完成一次检索只需执行一次pairing和N+1次群元素的除法，具有很高的效率。测试结果如图2.7所示。通过和XW15对比，生成相同的密文，LSPE节约了35%的时间；在检索效率方面，LSPE的检索速度大约为XW15的113倍；在能耗方面，LSPE的加密能效比EERE大约比XW15提高了62%。

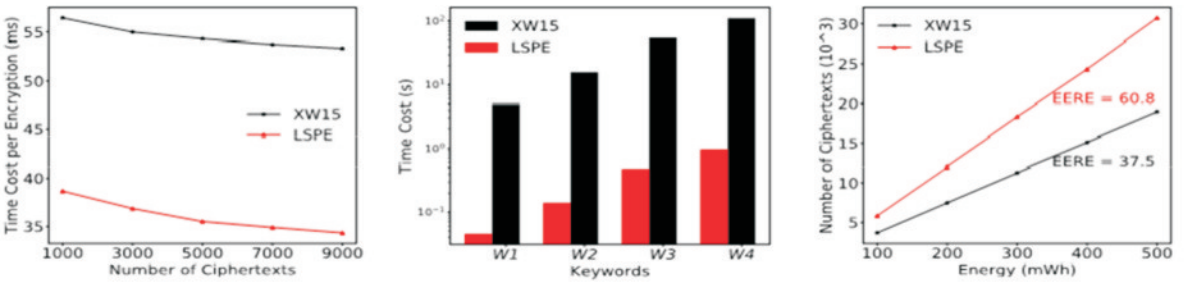


图2.7 实验结果图

2.6 同态加密FLAC音频密文下解压缩

云环境的安全问题一直受到热议，支持密文计算的同态加密算法是云上多媒体数据处理过程中保护隐私的有效手段。云上多媒体数据一般是压缩格式存放，因此云端在处理时需先对其进行解压缩。在用户数据加密情况下，云环境需在未解密情况下先对密文数据进行解压缩操作。对同态域多媒体数据进行解压缩是一个巨大的挑战，以FLAC音频为例，其挑战在于其解压缩过程需根据中间变量取值来控制程序分支，而在同态域中无法比较加密的中间变量，同态密文不支持分支程序。

为了应对上述挑战，我们提出了方案：通过计算密文匹配比特，将FLAC音频原始解码过程中的动态分支程序转化为同态密文所支持的静态顺序程序，对解压缩过程进行改写，如图2.8所示。对中间变量的每个分支计算相应的加密匹配比特并产生相应的结果，该比特表示该分支是否被匹配，所有分支中仅有一个是匹配分支，可通过密文匹配比特控制得到最终分支程序结果。计算每个分支产生的结果乘以对应的加密匹配比特并累加，累加产生的结果等同于选择匹配控制值的结果。在实现过程中，对同态密文下FLAC音频解压缩方案应用的优化方法有多线程并行、批量降噪和SIMD技术，并提出多密文连续乘优化技术。SIMD技术指将多个明文打包加密成单个密文，因此解码一个密文帧对应于解码多个明文帧。

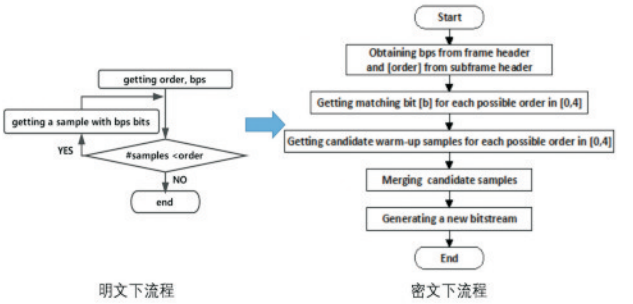


图2.8 FLAC音频解码流程

实验结果验证了同态加密域FLAC音频数据解压缩方案的正确性，优化方法提高了解压缩效率，其中SIMD技术将解压缩效率提高256倍，批量降噪将降噪效率提高16倍。同态加密域FLAC音频数据解压缩方案也适用于其他类似

编码技术的音频压缩标准，如SILK，Apple Lossless 和MPEG-4 ALS等。

3. 总 结

网络空间安全组的典型成果涵盖了云计算安全、软件安全、数据与隐私安全三个不同层面，尤其是在容器技术、并发漏洞挖掘、区块链技术、加密算法的理论与应用等方面有较强的研究特色，形成了一批优秀的会议、期刊论文和专利成果。小组除了在上述领域继续深入研究外，也在车联网、移动应用安全、侧信道攻击等领域进行了调研与探索，并取得了初步的研究成果。

附成果列表：

主要专利：

[1] 一种SDN网络中基于路径与IP地址跳变的主动防御方法，专利申请号201910390382.6，专利申请日期2019年5月10日，徐鹏，金海，张芝，袁斌

[2] 一种面向SDN网络的时序侧信道攻击的主动防御方法，专利号201910402808.5，专利申请日2019年05月15日，邹德清，袁斌，金海，徐鹏，方升泽

[3] 一种具有高效前向索引的可搜索公钥加密方法及系统，专利申请号201811529868.5，专利申请日期2018年12月14日，徐鹏，陈天阳，金海

[4] 一种边计算环境下轻量级的可搜索公钥加密方法，专利申请号201811320126.1，专利申请日期2018年11月7日，金海，徐鹏，陶冶，陈天阳

[5] 一种基于多方验证的区块链资产转移方法，专利申请号201910355006.3，专利申请日2019年04月29日，代炜琦，金海，邹德清，吴朱天

[6] 一种基于共享积分的区块链激励方法及系统，专利申请号201910345021.X，专利申请日2019年04月26日，代炜琦，代春凯，金海，邹德清

[7] 用于实现营销标签共享的区块链共识方法及区块链网络，专利申请号201910345004.6，专利申请日2019年04月26日，代炜琦，代春凯，金海，邹德清

[8] 一种基于区块链的营销标签交易方法及系统，专利申请号201910344391.1，专利申请日2019年04月26日，代炜琦，包庆华，金海，邹德清，代春凯

[9] 一种基于区块链的营销标签真实性验证方法及系统，专利申请号201910344350.2，专利申请日2019年04月26日，代炜琦，代春凯，金海，邹德清，包庆华

[10] 联盟链中实现轻钱包机制的方法、客户端、联盟链及系统，专利申请号201910319996.5，专利申请日2019年04月19日，代炜琦，金海，邹德清，王秦远

[11] 一种针对区块链的数字加密货币转移方法，专利申请号201811502165.3，专利申请日2018年12月10日，代炜琦，包庆华，金海，邹德清

[12] 一种基于区块链的发票管理方法和系统，专利申请号201811284455.5，专利申请日2018年10月31日，代炜琦，张玮炜，金海，邹德清

[13] 一种基于存储贡献值的区块链共识方法和系统，专利申请号201810997950.4，专利申请日2018年08月29日，代炜琦，吴为龙，张玮炜，金海，刘昱

[14] 一种基于存储的区块链激励方法、区块链网络及存储网络，专利申请号201810995081.1，专利申请日2018年08月29日，代炜琦，吴为龙，包庆华，金海，刘昱

代表性论文：

[1] Peng Xu, Shuanghong He, Wei Wang, Willy Susilo, Hai Jin. Lightweight Searchable Public-key Encryption for Cloud-assisted Wireless Sensor Networks. IEEE Transactions on Industrial Informatics, 2018, 14(8), pp. 3712 – 3723

[2] Bin Yuan, Deqing Zou, Shui Yu, Hai Jin, Weizhong Qiang, Jinan Shen. Defending against Flow Table Overloading Attack in Software-Defined Networks. IEEE Transactions on Services Computing, 2019, 12(2), pp. 231 – 246

[3] Deqing Zou, Zirong Huang, Bin Yuan, Haoyu Chen, Hai Jin. Solving Anomalies in NFV-SDN Based Service Function Chaining Composition for IoT Network. IEEE Access 2018, vol.6, pp. 62286 – 62295

[4] Deqing Zou, Jian Zhao, Weiming Li, Yueming Wu, Weizhong Qiang, Hai Jin, Ye Wu, Yifei Yang. A Multigranularity Forensics and Analysis Method on Privacy Leakage in Cloud Environment. IEEE Internet of Things Journal 2019, 6(2), pp. 1484 – 1494

[5] Changming Liu, Deqing Zou, Peng Luo, Bin B. Zhu, Hai Jin, A Heuristic Framework to Detect Concurrency Vulnerabilities. ACSAC '18 Proceedings of the 34th Annual Computer Security Applications Conference,

ACM, pp. 529-541
[6] Ihsan H. Abdulqadder, Deqing Zou, Israa T. Aziz, Bin Yuan. Enhanced Attack Aware Security Provisioning Scheme in SDN/NFV Enabled over 5G Network. ICCCN 2018, IEEE, pp. 1095-2055
[7] Weizhong Qiang, Zezhao Dong, Hai Jin. Se-Lambda: Securing Privacy-Sensitive Serverless Applications Using SGX Enclave. SecureComm 2018: Security and Privacy in Communication Networks, Springer, pp.451-470
[8] Deqing Zou, Ju Yang, Zhen Li, Hai Jin, Xiaojing Ma. AutoCVSS: An Approach for Automatic Assessment of Vulnerability Severity Based on Attack Process. GPC 2019: Green, Pervasive, and Cloud Computing, Springer, pp. 238-253
[9] Jinan Shen, Deqing Zou, Hai Jin, Bin Yuan, Weiqi Dai, A domain-divided configurable security model for cloud computing-based telecommunication services. The Journal of Supercomputing 75(1): 109-122 (2019)
[10] Weizhong Qiang, Weifeng Li, Hai Jin, Jayachander Surbiryala. Mpchecker: Use-After-Free Vulnerabilities Protection Based on Multi-Level Pointers. IEEE Access 7: 45961-45977 (2019)
[11] Weizhong Qiang. Performance and security in cloud computing. The Journal of Supercomputing 75(1): 1-3 (2019)
[12] Ihsan Abdulqadder, Deqing Zou, Israa Aziz, Bin Yuan, Weiqi Dai. Deployment Of Robust Security Scheme In SDN Based 5G Network Over NFV Enabled Cloud Environment. IEEE Transactions on Emerging Topics in Computing (TETC 2018)
[13] Hai Jin, Zhi Li, Deqing Zou, Bin Yuan. DSEOM: A Framework for Dynamic Security Evaluation and Optimization of MTD in Container-based Cloud. IEEE Transactions on Dependable and Secure Computing (TDSC 2019)
[14] Weiqi Dai, Chunkai Dai, Kim-Kwang Raymond Choo, Changze Cui, Deqing Zou, Hai Jin. SDTE: A Secure Blockchain-based Data Trading Ecosystem. IEEE Transactions on Information Forensics and Security（TIFS 2019）
[15] Zhen Li, Deqing Zou, Jing Tang, Zhihao Zhang, Mingqian Sun, Hai Jin. A Comparative Study of Deep Learning-Based Vulnerability Detection System. IEEE Access 2019
[16] Yuanyuan Tang, Bin Zhu, Xiaojing Ma, P. Takis Mathiopoulos, Xia Xie, Hong Huang. Decoding Homomorphically Encrypted Flac Audio without Decryption. ICASSP 2019
[17] Weizhong Qiang, Yingda Huang, Hai Jin, Laurence T. Yang, Deqing Zou. CloudCFI: Context-Sensitive and Incremental CFI in the Cloud Environment. IEEE Transactions on Cloud Computing 2019

邹德清
教授、博导
研究方向：云安全，软件安全与软件定义安全
Email：deqingzou@hust.edu.cn

马晓静
副教授、博导
研究方向：多媒体安全，密文处理
Email：lindahust@hust.edu.cn

徐　鹏
副教授、博导
研究方向：密码学、云数据安全
Email：xupeng@hust.edu.cn

羌卫中
副教授
研究方向：系统安全、软件安全
Email：wzqiang@hust.edu.cn

代炜琦
讲　师
研究方向：云安全、可信计算、虚拟化安全、可信SDN等
Email：wqdai@hust.edu.cn

袁　斌
博士后
研究方向：软件定义网络安全，物联网安全
Email：yuanbin@hust.edu.cn

大数据组典型成果介绍

华强胜，赵　峰，陈汉华，石宣化，丁晓锋，袁平鹏，谢　夏

关键词：分布式算法，流连接处理，内存计算，隐私保护，知识图谱

1. 介绍

大数据具有体量巨大、类型多样以及处理速度快等特点，给传统的算法以及系统提出了挑战，亟需新的处理模式才能合理运用大数据，以低成本创造高价值。本研究小组旨在研究面向新型体系结构的大数据系统关键技术，以大数据基础理论为基础，研究大数据系统的一系列科学问题，例如大数据管理、大数据处理、大数据分析等，并在此基础上研发了一系列大数据典型应用项目，为大数据的广泛高效应用提供技术支持。基于本组所承担的“大数据快速存储与计算环境”、“高通量计算与近似计算的理论与算法”等国家重点研发计划项目；“面向大数据的高时效并行计算机系统结构与技术”、“应用驱动的大图数据建模理论与算法研究”等国家自然科学基金项目以及一批面向电子政务、贸易运输等应用处理平台项目，对不同领域大数据问题进行深入研究，从算法以及系统等多个维度探讨了多个典型大数据领域数据处理问题的解决思路：

1）在大数据理论方面，在大数据低复杂度分布式算法方面的研究，提出了基于博弈论方法的准流图划分方法，同时也关注算法体系结构协同设计方面，提出了图流中针对节点的链路预测算法。

2）在大数据处理方面，研究了高时效大数据流连接处理系统，研发了基于动态负载监测的高性能实时分布式流连接系统FastJoin以及面向二路等值连接的高效可扩展实时分布式流连接系统Simois，同时提出了基于生命周期的内存计算垃圾回收优化器，并开展了基于动态预取策略的容器写时复制机制优化研究。

3）在大数据管理方面，针对大数据加密抗隐私泄露，提出了面向加密数据的隐私保护多关键字Top-k相似性搜索算法。同时针对需要基于有限的内存空间对大数据集合进行高效表示的大数据应用，构建了高时效大数据集合管理工具——优选Cuckoo过滤器。

4）在大数据分析和应用方面，提出了基于社交距离感知贝叶斯个性化排序推荐算法，以及基于语义匹配的知识问答系统ComQA，同时构建了一个集科研、教学、学术交流、人才培养等多功能于一体的国际平台——国际仿生科技平台。

2. 大数据理论研究

2.1 基于博弈论方法的准流图划分

大规模的图结构已经广泛融入到许多真实应用例如社交网络和网络拓扑中。由于图规模在爆炸式增长，因此分布式大图计算必不可少，而图划分是分布式大图计算的关键步骤，可以显著影响各分布式节点间通信成本和负载均衡，从而影响计算系统的性能。

2.1.1 研究问题

我们注意到大多数现实中的图都是通过以宽度优先的方式通过爬虫来收集的。因此，当图以边表的形式存储时，同一顶点上的边将相邻存储。在实际应用中，大多数图的度数服从幂律分布，其中大多数顶点度数较小。如果批大小大于大多数顶点的度数，我们可以将这些由每个批组成的子图近似地看成是不相交的。从而，我们可以并行地对这些批执行划分。我们需要指出的是，如果上述假设不成立，即如果一个

顶点的入边没有连续出现在输入中，我们可以执行一个预处理步骤使图按顶点聚集，这个过程的时间复杂度为O(n)，其中n为点的个数。

2.1.2 研究策略

针对现有图划分模型的不足，我们提出了一种准流图划分模型，并基于该模型提出了一种基于纯策略博弈的边图划分算法。具体地，通过将图的边流划分成若干个批次，在每个批次内将图划分问题转化为一个博弈过程。批次内的每条边被视为博弈的玩家，每条边的划分块选择被视为其策略。从而将原图的划分问题分解为一系列寻找纳什均衡的过程。每个批次内的边选择其最佳划分块时，只依赖该批次内其它边的划分块选择，所以各个批次内的博弈过程可以并行地寻找纳什均衡。首先通过设计适当的个体代价函数和社会福利函数构造一个博弈过程，并证明了该博弈过程是一个确切势博弈，从而一定存在纯策略纳什均衡。然后基于最佳动态响应提出了一个能快速收敛到一个纯纳什均衡的算法。

我们使用$G_b=(V_b, E_b)$来表示每个批次b的子图，用G=(V, E)表示整个图，划分数量定义为k，我们的目标是将每条边指派到不同划分中使得每个顶点的复制次数尽量的小：

$$\min \frac{1}{|V|}\sum_{v\in V}|\operatorname{Re} p(v)|$$

$$s.t. \max_{p_i\in P} L(P_i) < \lambda\frac{|E|}{k}$$

其中$Rep(v)\subseteq P$是持有顶点v副本的一个集合，$\lambda\geqslant 1$是一个小常数。|V|和|E|是图中点和边的个数。$P=\{p_1,...p_k\}$是划分的集合，以及$L(p_i)$代表了p_i中的边数等，可以得知边划分问题是一个NP难问题。

因此，我们将边划分问题建模为一个博弈过程，我们将每个边看作一个参与者，边的竞争选择等价于参与者的策略选择。在博弈过程中，当达到纳什均衡时，表明我们已经获得了全局目标的局部最优。所以我们想探索我们是否可以通过在边划分问题中的目标函数与博弈过程中的全局成本函数之间架起桥梁，找到一个较好的边划分问题的局部最优解。我们首先构建一个与边划分问题相关的博弈过程，然后证明博弈过程总是收敛于纳什均衡，最后，我们通过PoA（Price of Anarchy，无政府状态的代价，描述因自私选择带来的总体损失）来度量这些纳什均衡的质量。

2.2 图流中针对节点的链路预测算法研究

图流是一种大规模、高度动态、无止境的图结构，其节点和边以流的形式快速到达。现阶段图流应用中对节点链路预测是最重要的应用。在传统的图流节点的链路预测模型中，因为图流规模很大，主要采用了节点偏置采样的方法。虽然这种方法可以很好的解决图流的存储问题，但是存在链路预测准确率不稳定、对公共邻节点（common neighbor）指标的估计相比真实值偏差较大、链路预测算法效率有待提高等问题。因此，亟需新的算法和技术来解决图流中针对节点的链路预测问题，在不损失准确率的情况下，尽可能提高算法效率，做到实时、快速的链路预测。

针对现有的链路预测方法存在的准确率不够高、延迟高等不足之处，我们提出一种双阶段选择链路预测系统，称之为TPLP(Two-Phase Selection Link Prediction System)，旨在预测图流中最有可能与目标节点发生链接的top-k个节点。该算法的研究主要在设计优化的选择算法和设计新的算法估算图流中的节点聚集系数。

该系统所提出的的优化后的选择算法的主要设计思路如下：以求得图流中节点v的KFN(k-future-neighbors，指最有可能与节点u发生链接的top-k个节点)为例，该算法是遍历当前图流对应的节点偏置采样中的所有节点，只要$I(u)\cap I(v)$不是空集，那么就取u的所有邻接点集放入集合S中。结束遍历后最后的S就是当前u节点的

KFN的预选集合Cdd(u)。u节点与候选集合中的节点按照相似度取前k个就是当前的KFN。如果节点偏置采样中的节点个数为N，那么相比传统的选择算法，这里速度提高了 N/ | Cdd(u) | 倍。

节点聚集系数表示了某个节点的邻接点之间发生链接的概率。该系统提出了App-NCC来估计图流中的节点聚集系数。首先设置一个阈值N和一个计数器。然后，判断可能的链路数目是否小于N。如果小于N，则遍历所有可能发生链接的节点对，如果节点对之间存在链路（即节点之间有边），则将计数器加1。通过这种方式最终得到节点u的邻接点之间存在的链路数目allPair，然后使用*countallPair*得到NCC的值。否则，随机选择一个节点对 $v1, v2$），其中 $v1 \in I(u)$，$v2 \in I(u)$，判断这两个节点之间是否存在链接。如果存在，则将计数器加1。最终重复此步骤N次，计数器的值表示这N次试验当中节点u的邻接点发生链接的次数，显然，可以通过*countN*表示节点u的邻接点发生链接的概率（这也是NCC的物理意义），由此估计NCC的值。可以看到，通过这种方法估计NCC的时间复杂度是O（N · L）。假定N=100，L=100 的情况下，则与传统NCC方法相比，该方法可以将时间复杂度提高100倍。

3. 大数据处理技术

3.1 高时效大数据流连接处理系统

流连接是对实时产生的大数据流进行高效的连接处理，是获取大数据价值的基础支撑技术。流连接被广泛应用于大数据实时应用当中，例如网约车智能派单，电信欺诈监测，集群日志分析等等。与传统的数据库连接操作不同，大数据流持续快速到来的特征，给处理系统会带来处理空间复杂度高、难以扩展、时效性差、结果正确性难以保证等一系列难题。现有的流连接处理系统建立在大规模分布式集群上，利用大量的并行计算资源，并配合高效数据分发策略，来提升系统性能。然而，实际应用的大数据流普遍存在倾斜分布的特征，即少数键值在数据流中占据着显著比例，造成系统计算负载严重不均，极少比例的键值带来了系统的主要负载，严重限制了分布式流连接处理系统的处理能力，制约了系统的时效性。

针对数据流倾斜特征给连接带来的系统负载不均问题，设计了两种不同的解决思路：一种是通过预估潜在对系统带来高负载的数据键值，对不同键值数据采用高效的区分调度策略，保证系统的负载均衡；另一种是实时检测分布式流连接系统的计算负载情况，当负载不均达到一定程度时，从重载节点中选择具有部分键值的数据，向轻载节点进行迁移，并使得迁移后两个节点的计算负载相近。相比之下，前者避免负载不均的发生，后者是消除负载的不均。

基于第一种设计思路，研发了分布式流连接处理系统Simois。Simois结构如图3.1所示，系统设计了独立的高负载键值预测模块对参与连接的两条流中带来连接计算高负载键值进行实时预测，并根据预测结果，将少量高负载键值使用轮询的方法均分到所有连接计算单元上，保证系统计算负载的均衡性；将大量其他键值使用哈希的方法映射到不同的连接计算单元上，避免造成大量冗余的通信和计算开销，保证系统在存储和计算上的高可扩展性。为了实现低时空开销地预估每个数据键值可能带来的计算负载，同时有效应对流数据分布随时间高动态变化的影响，Simois 设计了一种轻量级基于概率计数的流连接键值负载预测技术，以对数级的计数空间统计统计和计算每个流数据键值的实时到来频率以及潜在会产生的连接计算次数，高效地筛选出高负载键值。相比于国际上最新的流连接系统，Simois将系统的整体吞吐率提升了52%，平均处理延时降低了64%。系统代码发布在国际开源社区https://github.com/CGCL-codes/Simois。

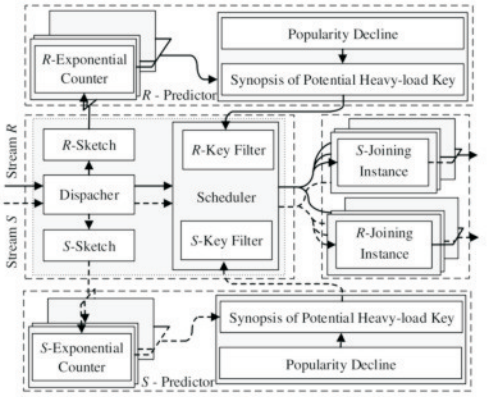


图3.1 Simois系统结构

基于第二种设计思路，研发了分布式流连接系统FastJoin。FastJoin结构如图3.2所示，系统设计了一个轻载的计算负载监测模块，触发性地同步预估每个连接计算单元的当前负载值。一旦最重载的计算单元和最轻载的计算单元间负载比值超过一定阈值，即快速从重载计算单元中筛选出部分键值，并将具有此键值的数据全部迁移向轻载计算单元中。最小化迁移后两个节点负载的差值是一个NP难的问题。对此，FastJoin设计了一种快速的近似最优键值筛选策略，并基于贪心算法和模拟退火算法进行了实现。同时基于这种思路研发了分布式流连接系统FastJoin。相比于国际上最新的流连接系统，FastJoin将系统的整体吞吐率提升了32%，平均处理延时降低了15%。系统代码发布在国际开源社区https://github.com/CGCL-codes/FastJoin。

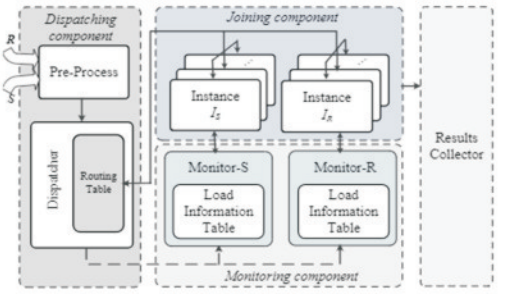


图3.2 FastJoin系统结构

3.2 内存计算垃圾回收优化器

大数据处理系统近十年来发展迅猛，比如Apache Hadoop、Apache Spark，能基于普通的硬件以横向扩展的方式处理海量的数据。与使用声明性查询语言和关系数据模型的传统数据库系统不同的是，这些大数据处理系统大多使用带有托管执行环境的高级语言（如Java、Scala、C#等）开发来降低分布式环境下的部署和调试的难度。用户可以使用系统提供的丰富算子以及用户自定义函数（UDF）和用户自定义类型（UDT）来实现复杂的应用逻辑。系统能够自动将应用程序逻辑运行在大规模的集群上，无需关心并行处理的细节。然而托管环境提供内建的高级功能，比如自动内存管理和并发模型，方便了用户的开发，却使得其对象模型的底层实现非常复杂，带来额外的内存和中央处理器（CPU）开销。

我们研究开发了Deca系统，它从用户自定义类型（UDT）和用户自定义方法（UDF）入手，将分布式数据处理系统中的数据对象的生命周期与数据容器关联，一方面剥离了对象中关于锁、状态等附加信息，完全拆解对象，解决了内存膨胀问题，另一方面通过生命周期的统一管理，将对象拆解到字节数组中，跳过JVM的内存管理，消除垃圾收集的CPU开销。

Deca首先通过RDD链和UDF的分析，确定UDT对象的生命周期，然后将相同生命周期的对象（即可放置与同一个数据容器的对象）拆解存放到MemoryPage中。MemoryPage是Deca 用来存放对象拆解后的byte序列的，其中保存了数据的bytes数组，偏移量等信息。Pages中的数据对象具有相同的生命周期，利用JVM的垃圾回收机制整块回收（此时由于Page数量远小于对象数量，垃圾回收的开销非常小）。缓存数据集和Shuffle缓冲区拆解过后的Memory布局如图3.3所示。

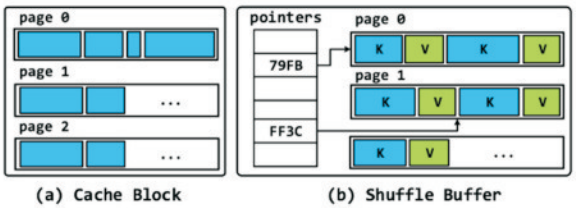


图3.3 对象拆解后的布局

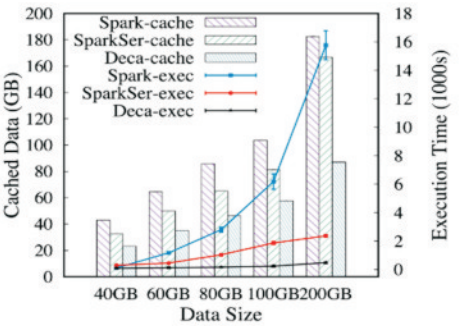


图3.4 计算时间和缓存大小对比

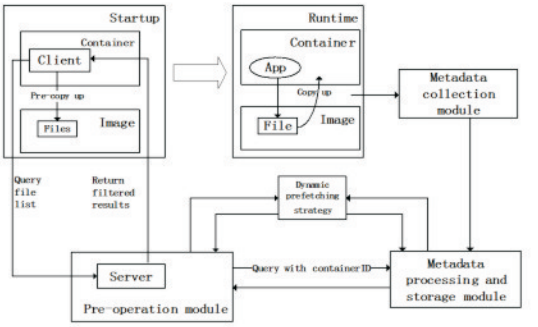


图3.5 DPSO整体架构

我们在生成和真实数据集都进行的广泛实验研究，结果表明与原生Spark相比，Deca能够将垃圾收集时间减少99.9%，内存消耗减少46.6%，缓存开销减少23.4%；在无数据溢出时，实现执行加速1.2x-22.7x，在数据溢出的情况下，实现16x-41.6x的加速。图3.4 展示了LR应用在不同优化措施下的性能对比，可以看到使用Deca的情况下，cache占用的内存和运行时间均优于Spark。

3.3 DPSO：基于动态预取策略的容器写时复制机制优化研究

容器技术的产生满足了用户对轻量级虚拟化的需求。目前Docker是最主流的容器解决方案。由于Docker是基于共享镜像文件来实现容器体量的最小化，因此镜像文件的存储需要和容器运行时产生文件的存储分离开。基于Docker对分层存储的需求，目前主流的Docker存储驱动均采用了CoW（Copy-on-Write）技术，其存储主要分为两部分，一部分是镜像文件的存储，一部分是容器运行时产生的文件存储。底层的镜像文件作为只读层存在，而由镜像文件启动的容器层则作为读写层，用来存储容器个体的私有数据。CoW机制满足了不同容器对同一镜像文件的共享需求。然而对于单个容器来说，每次在访问镜像文件时都需要先将其copy up到上层的读写层，在被访问的镜像文件较多、较大时，这将会带来无法忽视的额外开销。因此优化容器CoW机制带来的额外开销是非常必要的。

因此，我们提出了一个基于动态预取策略的CoW机制优化方法。核心思想是根据Docker镜像具有单一功能性的特点，通过同一镜像启动的不同容器的行为，分析镜像文件被访问的模式并且形成历史记录。根据不同镜像对应的历史记录，在依赖该镜像启动的容器启动初期，将部分镜像文件预先进行copy up操作。另外本系统还设计了一个动态预取策略来对历史记录进行管理和筛选，保证精准有效地对镜像文件进行预copy up操作，整体系统架构如图3.5所示。

4. 大数据管理技术

4.1 面向加密数据的隐私保护多关键字Top-k相似性搜索

云计算为个人和企业提供了巨大的计算能力和可扩展的存储容量来支持医疗保健和科学研究等领域中的各种大数据应用，因此越来越多的数据所有者将数据外包到云服务器上以方便数据管理和挖掘。但是，数据集如电子文档中的健康记录通常包含敏感信息，如果这些文档被泄露或者共享给云中部分不受信任的第三方将会造成隐私威胁。目前广泛使用的数据隐私保护技术是将外包到云服务器的数据提前进行加密处理，但是这会降低数据效用并使许多传统的数据分析方式例如基于关键字的top-k文档检索不再实用。

4.1.1 研究问题

如图4.1所示，我们考虑的系统模型包含三个部分：数据所有者，数据用户和云服务器。数据所有者上传文档集合D到云服务器，但此集

合可能包含敏感信息。为了保护数据隐私，数据所有者必须在将其外包到云服务器之前加密D。此外，为了保证云服务器在加密文档集合C上有效地处理查询，数据所有者在本地构造加密可搜索索引Ie。最后，数据所有者外包加密文档集合C和加密的可搜索索引Ie到云端，并通过安全通道分享陷门生成和文档解密的密匙给授权数据用户。当数据用户想要使用查询进行搜索时，用户首先通过查询加密为此查询生成陷门T，然后将陷门提交给云服务器用于查询处理。在收到T后，云服务器计算陷门T与索引Ie中的文档之间的相关性分数，并将具有最高分数的k个文档返回给数据用户。

4.1.2 研究策略

本研究针对大数据加密抗隐私泄露提出了一种多关键字top-k搜索理念，并尝试为此问题确定有效且安全的解决方案。具体来说，出于对查询数据的隐私考虑，我们构造了一个特殊的基于树的索引结构，并设计了一个随机遍历算法，使得即便相同的查询在索引上也能产生不同的访问路径，并且还可以在隐私保护效果更强的情形下保持查询的准确性不变。此外，为了提高查询效率，我们提出了一种基于分区思想的群组多关键字top-k搜索方案，在这里我们为所有文档构造出一组基于树的索引。最后，我们将这些方法以一种有效且安全的方式结合在一起来解决我们提出的top-k相似性搜索问题。经过在实际数据集上进行大量实验，最终实验结果证明相较于目前最新的方法，我们提出的方法可以明显提高隐私泄露的防御能力，可扩展性和查询处理的时间效率。

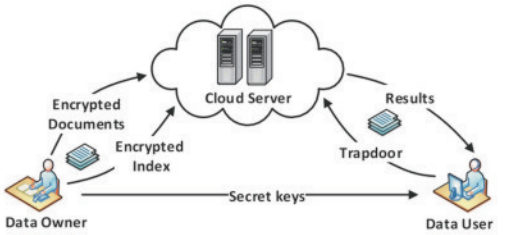


图4.1 面向外包加密数据的系统模型

4.2 高时效大数据集合管理工具——优选Cuckoo过滤器

许多大数据应用需要基于有限的内存空间对大数据集合进行高效的表示，并支撑快速的插入、查询、删除等操作。例如，云存储系统中需要利用近似集合成员判断技术，对存储的大量文件数据进行轻量快速的去重操作。在现有的大数据集合近似表示结构中，Cuckoo过滤器（CF）具有极高的空间效率，常数级的查询时间以及支持可靠数据删除等优势，在实际应用中被广泛应用。CF以指纹的形式代替原始数据进行集合成员的表示，并将每个原始数据映射到固定的两个候选存储位置。当新数据到来时，CF随机选择一个未满的候选位置对其指纹进行存储。当数据对应的候选位置均存储满了时，CF将会触发重定位操作，即随机选择一个候选位置中已存储的指纹，改将其存储到其对应的另一个候选位置中，以腾出空间存储新插入数据的指纹。被移出的指纹所对应的另一个候选位置也满了的情况下，该重定位操作会反复触发，直到所有指纹都完整的存储在结构当中。频繁的重定位操作带来的高数据插入延时是CF性能的主要瓶颈。CF中重定位操作频繁发生的根本原因在于，随机选择候选位置导致了每个存储位置间存储指纹个数的高度负载不均。部分存储位置提前存满指纹导致了重定位的过早发生。图4.2的结果显示了，在CF仅有10%的空间存储了指纹的情况下，已经开始出现重定位操作。在这种情况下，随着存储指纹数量增多，重定位发生越来越频繁，其数据插入延时也显著增长，如图4.3所示。

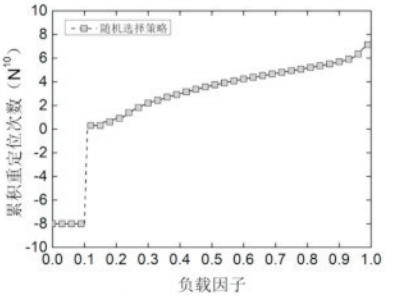


图4.2 随机选择策略导致重定位过早发生

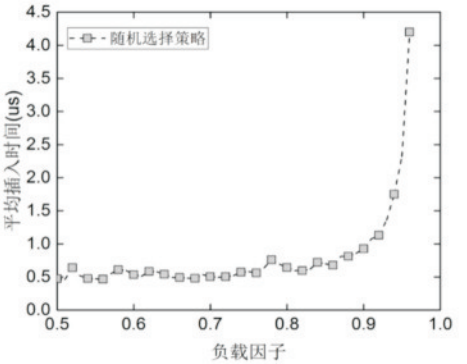


图4.3 频繁重定位带来CF的高插入延时

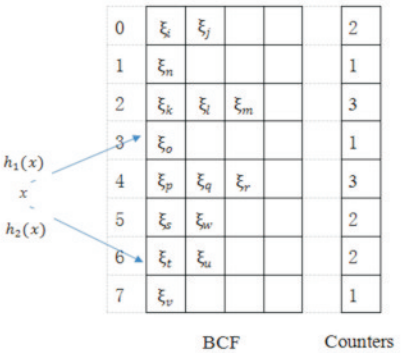


图4.4 BCF的插入过程

为了有效降低Cuckoo过滤器的重定位次数，提高大数据集合管理的时效性，研究小组设计了一种基于插入位置优选的集合表示结构—优选Cuckoo过滤器（BCF），并设计了一套高效的算法对大数据集合进行插入、查询和删除等操作。BCF基于“The Power of Two Choices”的思想，在大数据集合动态插入的过程中，尽量保证各个存储位置已插入元素个数的均衡。即在每个数据元素插入时，对比两个存储位置的已存储指纹个数，选择更少的一个存储位置进行存储。如图4.4所示，BCF为每一个存储位置设置计数器记录其存储的指纹个数，当新来的数据x插入时，其对两个候选位置3，6的计数进行对比，并选择计数更小的位置3进行插入。该插入策略有效延迟了Cuckoo过滤器中满载存储位置出现的时间，解决了原先随机选择策略下，由于哈希碰撞导致的部分存储位置过早满载，进而频繁触发重定位的问题。

结合真实大数据集的测试，证明了BCF可以有效地提升大数据集合的插入效率。如图4.5所示，相比于当前的Cuckoo过滤器，将平均重定位次数降低了35%，平均插入时间提升了18%。研究小组也将BCF集成到骚扰电话拦截原型系统中，利用BCF对用户实时标记的骚扰电话黑名单进行存储，并在新电话到来时进行查询，相比于当前的Cuckoo过滤器，BCF将骚扰电话拦截系统的平均吞吐率提升了15%。BCF的代码发布在国际开源社区https://github.com/CGCL-codes/ BCF。

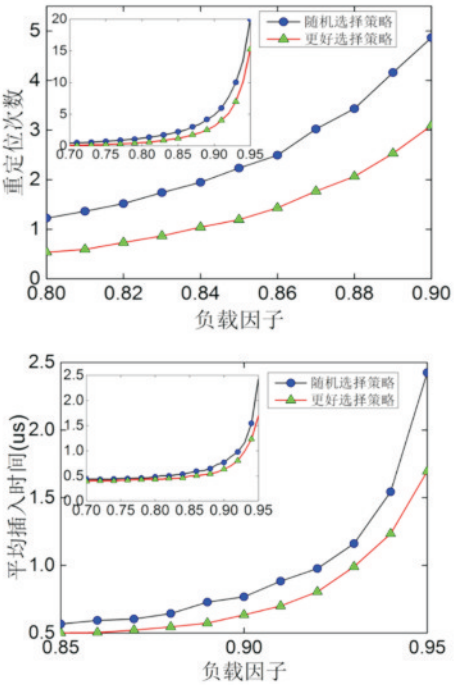


图4.5 BCF性能测试

5. 大数据分析和应用

5.1 基于语义匹配的知识问答系统ComQA

随着开放数据（如Open Government Data Initiatives、Open Access to Scientific Data、Linked Open Data等）的出现和发展，越来越多的大规模知识图谱，如DBpedia和Freebase都可以轻松获得。互联网上不断增加的知识图谱正在成为许多信息系统的知识来源，同时也催生

了一批基于知识图谱的新应用。一个典型的例子就是知识库问答（KBQA）。通过利用大规模高质量的知识图谱作为知识库，KBQA可以从过去的部分垂直专业领域扩展到开放通用领域，进而快速而准确地回答人们以自然语言形式提出的问题。知识图谱中包含大量实体及其之间关系的详细信息，对QA任务可以提供更为丰富的上下文。但知识图谱和自然语言问句之间存在的语义鸿沟却是知识库问答系统所必须要解决的一个基本挑战。

我们开发了知识库问答系统ComQA，用户可以通过系统从知识库中直接获取到复杂问题的答案。概括来讲，在ComQA中，复杂问题会被解析为若干个三元组模式；然后ComQA从知识库中检索出与该三元组模式匹配的候选子图，并且评估候选子图与三元组模式之间的语义相似度进行候选子图的排序，最后从得分最高的候选子图中抽取出答案返回给用户。我们在三个QALD的数据集上进行了实验，ComQA在精确度，召回率和F1-Score方面与当前的最新方法相当，在可回答问题数目上取得了领先。

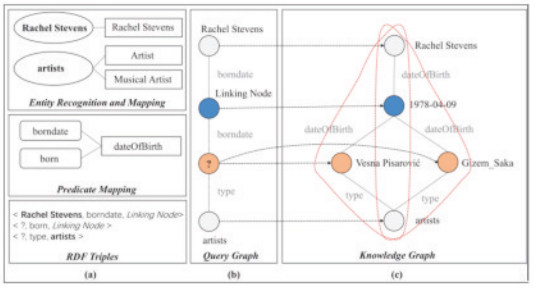


图5.1 ComQA工作流程

图5.1给出了ComQA的工作流程。整个流程分为三部分：首先，ComQA对输入问题进行解析，借助于词向量抽取并构建实体和关系的语义扩展集。接着，基于输入问题的语法依存树结合实体和关系的语义扩展集合，构建表示用户意图的查询子图集合，并将查询子图集合映射到知识图谱上获取候选子图集合。最后，结合图的同构和语义相似度，对候选子图进行过滤和排序，从得分最高的候选子图中抽取出答案返回给用户。

5.2 社交距离感知贝叶斯个性化排序推荐算法SDBPR

近年来，推荐系统在人们的日常生活中非常流行并且已经被许多互联网服务所广泛使用。这些推荐系统的存在一方面为人们的日常生活带来许多便利，另一方面也提升了企业的收益。由于推荐系统蕴含巨大的商业价值，吸引了企业及学界的大量研究人员投身于此，使得这个领域在近年来取得了飞速发展。近年来，社交网络被引入推荐系统用于提升推荐的准确率，由于社交网络中的朋友间通常具有相似的爱好，因此可以利用用户的朋友的偏好来估计用户自己的偏好。现有的社交推荐方法只考虑了用户在社交网络中直接朋友，而没有考虑用户间的影响在社交网络中的传播过程。因此现有方法无法充分利用社交网络的图结构信息，限制了其推荐的准确率。

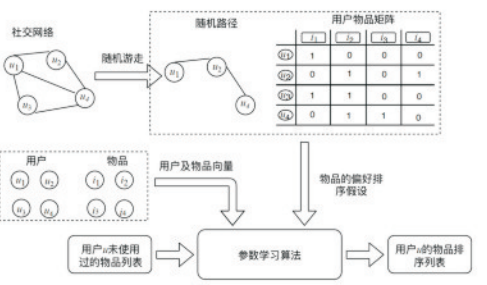


图5.2 SDBPR模型总体架构

针对现有方法的不足之处，我们提出一种社交距离感知贝叶斯个性化排序推荐算法（Social Distance-aware Bayesian Personalized Ranking, SDBPR），相比于现有的方法充分利用了社交网络的图结构信息，能够在实际的个性化推荐系统中表现出良好的性能。模型总体架构如图5.2所示，其核心思想是采用直接对用户的物品偏好排序进行建模的方式避免隐

式反馈数据的稀疏性，并通过考虑任意两个多步距离的间接朋友模拟用户影响在社交网络中的传播过程，从而充分利用了社交网络的图结构信息。

具体而言，SDBPR首先使用随机游走算法揭示了社交网络复杂的图结构，对社交网络进行遍历采样得到随机路径，并基于随机路径上的用户间距离生成用户对物品的二元排序假设；然后利用贝叶斯方法计算这些二元排序假设发生的概率并通过随机梯度下降法将每个用户和物品都被映射到一个隐式的向量空间，向量空间的每个维度代表了用户或者物品的一个特征，通过向量的点积运算得到用户对物品的偏好数值，进而为用户生成物品偏好排序列表，从而最大化这个概率得到用户及物品的特征表示；最后利用用户及物品的特征表示计算用户对物品的偏好数值，进而生成用户对物品的偏好排序列表。SDBPR考虑了社交网络的图结构对用户间的相互影响在社交网络中的传播有重大影响的这一现象，从而具有比现有模型更好的推荐效果。

5.3 国际仿生科技平台

大数据组依托吉林大学工程仿生教育部重点实验室，设计了一个集仿生百科、仿生科研、仿生教学、信息交互、仿生成果展示等功能于一体的国际仿生科技平台（http://www.bionics-online.net），9月24日正式上线。

平台为用户推送热门的仿生新闻、资源；提供丰富活跃的仿生互动国际社区；搭建授业解惑的在线教学平台；提供仿生领域权威的百科知识库，促进仿生领域的全面交流和人才汇聚。

国际仿生科技平台的硬件架构如图5.3所示。平台部署在阿里云服务器上，除了web应用服务器之外，还设置了一台数据库服务器和一台FTP服务器提供文件服务，并且支持PC、手机等多设备的访问。

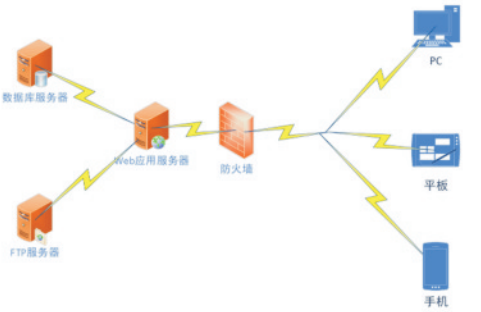


图5.3 国际仿生科技平台硬件架构

平台的软件架构设计如图5.4所示。从功能模块上划分为新闻、仿生百科、专家知识库、教学、社区、实验和设备、用户等模块。从平台逻辑角度划分为应用层、服务层、存储层、数据来源四个层次，降低了系统内部的耦合度，方便后期扩展和维护。

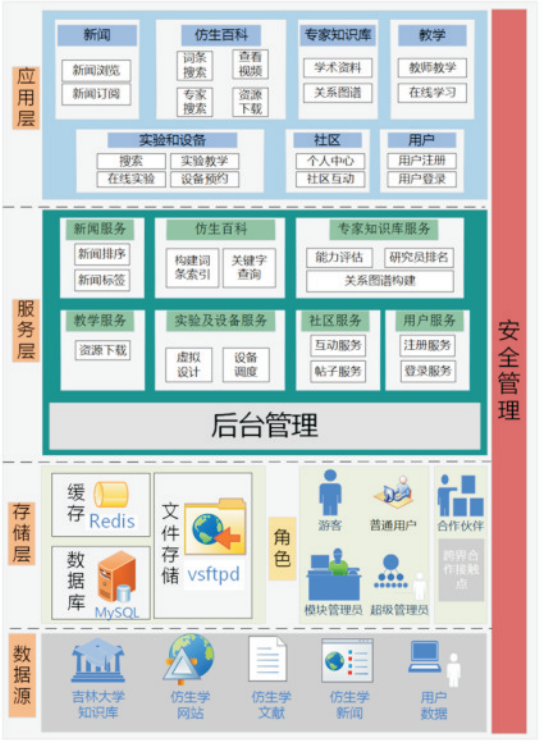


图5.4 国际仿生科技平台软件设计架构

通过国际仿生科技平台，我们希望能够向大众科普仿生科学知识，增加仿生专业人员间的学术交流和资源共享，促进仿生学术成果

（论文、专利、转化等）取得进步。

6. 总结

大数据小组的典型成果覆盖了大数据基础理论、大数据处理系统软件、大数据分析方法以及典型应用，在内存计算技术、流数据处理技术、社交数据分析方面的研究特色明显，在国际顶级学术期刊与会议上发表一批高水平论文、在国际开源社区积极推进开源，并获得一批具有领域代表性的专利成果。

附成果列表：

开源系统

[1] 分布式流连接处理系统Simois，https://github.com/CGCL-codes/Simois

[2] 分布式流连接系统FastJoin，https://github.com/CGCL-codes/FastJoin

[3] 优选Cuckoo过滤器，https://github.com/CGCL-codes/BCF

专利

[1] 基于混合着色模型的异步GPU图处理系统，专利申请号：2018SR609508，专利授权日：2018年08月02日

[2] 一种图流中针对节点的链路预测方法. 专利申请号 201910141425.7.

[3] 一种面向图流的社交网络用户间路径规划方法. 专利申请号 2019102079550.7.

[4] 基于动态负载监测的流连接系统负载均衡方法及装置, 专利申请号：201910408893.6.

[5] 基于插入位置选择的数据近似集合表示方法及系统，专利申请号：201910419541.0.

[6] 一种多应用场景下的分布式流处理系统的容错方法，专利申请号：201910423546.0.

[7] MapReduce相似连接查询的隐私保护方法及系统，专利申请号：201810919923.5.

[8] 一种面向图中三角形数据发布的隐私保护方法及系统，专利申请号：201811019771.X.

[9] 基于多GPU的通信优化的图数据处理系统，专利申请号：201711115570.5.

[10] 基于属性图的大规模关联数据划分方法及系统，专利申请号：201910425921.

论文

[1] Feng Zhao, Yu Shen, Xiangyu Gui, Hai Jin. SDBPR: Social Distance-aware Bayesian Personalized Ranking for Recommendation, FGCS 2018.

[2] Yang Xiao, Hong Huang, Feng Zhao, Hai Jin. TPLP: Two-Phase Selection Link Prediction for Vertex in Graph Streams, PAKDD 2019.

[3] Fan Zhang, Hanhua Chen, Hai Jin. Piggyback Game: Efficient Event Stream Dissemination in Online Social Network Systems. IEEE Transactions on Parallel and Distributed Systems (TPDS), 2019.

[4] Hai Jin, Changfu Lin, Hanhua Chen, Jiangchuan Liu. QuickPoint: Efficiently Identifying Densest Sub-graphs in Online Social Networks for Event Stream Dissemination. IEEE Transactions on Knowledge and Data Engineering (TKDE), 2019.

[5] Sijie Wu, Hanhua Chen, Hai Jin, Shadi Ibrahim. Shadow: Exploiting the Power of Choice for Efficient Shuffling in MapReduce. IEEE Transactions on Big Data (TBD), 2019.

[6] Hanhua Chen, Hai Jin. Efficient Keyword Searching in Large-scale Social Network Service. IEEE Transactions on Services Computing (TSC), 2018.

[7] Fan Zhang, Hanhua Chen, Hai Jin. Simois: A Scalable Distributed Stream Join System with Skewed Workloads. ICDCS 2019.

[8] Feiyue Wang, Hanhua Chen, Liangyi Liao, Fan Zhang, Hai Jin. The Power of Better Choice: Reducing Relocations in Cuckoo Filter. ICDCS 2019.

[9] Shunjie Zhou, Fan Zhang, Hanhua Chen, Hai Jin, Bing Bing Zhou. FastJoin: A Skewness-Aware Distributed Stream Join System. IPDPS 2019.

[10] Changfu Lin, Jingjing Zhan, Hanhua Chen, Jie Tan, Hai Jin. Ares: A High Performance and Fault-Tolerance Distributed Stream Processing System. ICNP 2018.

[11] Syed Muhammad Abrar Akber, Hanhua Chen, Yonghui Wang, Hai Jin. Minimizing Overheads of Checkpoints in Distributed Stream Processing Systems. CloudNet 2018.

[12] Adeel Aslam, Hanhua Chen, Jiang Xiao, Song Wu, and Hai Jin. Reasoning Based Virtual Machine Mapping Toward Physical Machine. HCC 2019.

[13] Muhammad Mudassar Qureshi, Hanhua Chen, Hai Jin. Modeling Distributed Stream Processing Systems under Heavy Workload. CW 2019.

[14] Xiaofeng Ding, Peng Liu, Hai Jin. Privacy-Preserving Multi-Keyword Top-k Similarity Search

Over Encrypted Data. IEEE Transactions on Dependable and Secure Computing (TDSC). 2019.

[15] Xiaofeng Ding, Wanlu Yang, Kim-Kwang Raymond Choo, Xiaoli Wang, Hai Jin. Privacy preserving similarity joins using MapReduce. Information Sciences. 2019.

[16] Qiang-Sheng Hua, Yangyang Li, Dongxiao Yu, Hai Jin. Quasi-Streaming Graph Partitioning: A Game-Theoretical Approach. IEEE Trans. on Parallel and Distributed Systems, 30(7):1643-1656, July 2019.

[17] Xuanhua Shi, Zhixiang Ke, Yongluan Zhou, Hai Jin, Lu Lu, Xiong Zhang, Ligang He, Zhenyu Hu, Fei Wang. Deca: a garbage collection optimizer for in-memory data processing. TOCS 2019.

[18] Yusheng Hua, Xuanhua Shi, Hai Jin, Wei Liu, Yan Jiang, Yong Chen, Ligang He. Software-defined QoS for I/O in exascale computing. CCF Transactions on High Performance Computing, 2019.

[19] Yan Jiang, Wei Liu, Xuanhua Shi, Weizhong Qiang. Optimizing CoW Mechanism of Docker by Dynamic Prefetching. NPC 2019.

[20] Ying Yang, Xuanhua Shi, Wei Liu, Hai Jin, Yusheng Hua, Yan Jiang. DDL-QoS: A Dynamic I/O Scheduling Strategy of QoS for HPC Applications. Concurrency and Computation: Practice and Experience 2019.

[21] Zhenyu Hu, Xuanhua Shi, Zhixiang Ke, Hai Jin, Fei Wang, Estimating the memory consumption of big data applications based on program analysis. SCIENTIA SINICA Informationis, 2019.

[22] PingPeng Yuan, Longlonng Lin, Zhijuan Kou, Hai Jin.Big RDF Data Storage, Computation, and Analysis: A Strawman's Arguments, ICDCS 2019.

[23] Wang T, Yuan P, Liao X. Parallel Processing SPARQL Theta Join on Large Scale RDF Graphs. GLOBECOM 2018.

[24] Hai Jin, Pingpeng Yuan, Lijian Fan. Performance RDF Updates with TripleBit+, ICDIM 2018.

[25] Haohan Pang, Peng Gan, Pingpeng Yuan, Hai Jin, Qiangsheng Hua. Partitioning Large-scale Property Graph for Efficient Query Distributed Processing. HPCC 2019.

[26] Jin Hai, Yi Luo, Chenjing Gao, Xunzhu Tang, and Pingpeng Yuan. ComQA: Question Answering over Knowledge Base via Semantic Matching. IEEE Access 2019.

华强胜
副教授，博导
研究方向：分布式算法
Email：qshua@ hust.edu.cn

赵　峰
教授，博导
研究方向：信息搜索、数据挖掘与分布式计算等
Email：zhaof@hust.edu.cn

陈汉华
教授，博导
研究方向：大数据处理系统，分布式计算系统
Email：chen@hust.edu.cn

石宣化
教授，博导
研究方向：云计算与大数处理、异构并行计算等
Email：xhshi@ hust.edu.cn

丁晓锋
副教授，博导
研究方向：大数据管理系统，隐私保护，深度学习
Email：xfding@hust.edu.cn

袁平鹏
教授，博导
研究方向：图数据库、图处理系统、分布式计算
Email：ppyuan@hust.edu.cn

谢　夏
副教授
研究方向：海量数据挖掘，并行计算
Email：shelicy@hust.edu.cn

OSv中的线程调度模型

（何 轲 https://kayhaw.github.io/2019/09/12/OSv-ScheduleModel.html）

由剑桥大学Anil，Richard等人在ASPLOS'13上发表的论文《Unikernels: Library Operating System for Cloud》首次提出了Unikernel的概念，它是一种由库操作系统而构建的单应用，单地址空间的机器镜像，Unikernel为云而生，具有体积小，速度快，攻击面少等特性，因此也被称为下一代的云操作系统。自此，有关Unikernel的研究如雨后春笋般层出不穷，其中便包括在ATC'18上发表的论文《OSv—Optimizing the Operating System for Virtual Machines》。这篇文章主要介绍在OSv中线程调度模型的设计与实现。

OSv线程调度的设计原则包含6点：lock-free，preemptive，tick-less，fair，scalable和efficient。

Lock-free

对于云环境下的对称多处理(SMP)机器，当代操作系统多采用自旋锁(spin-lock)：当一个CPU使用原子操作获得资源并对其上锁后，其他的CPU执行空循环操作直到它们获得锁。自旋锁适用于广泛的SMP机器，但对于虚拟机来说，自旋锁有着明显的缺点，也就是“lock-holder preemption”问题：当物理CPU保持运行时，虚拟CPU可能会在某个时刻“暂停”某段时间，当虚拟机因执行hyper call而退出hypervisor或者hypervisor运行其他guest时会发生这种情况。若一个虚拟CPU在持有自旋锁时暂停，其他想要获得该锁的CPU不必要地循环执行空操作(spin)，浪费CPU时间，同样地，对于使用自旋方法的互斥锁，没有获得锁的线程会不停地spin浪费CPU时间，实验表明这会带来7%至99%(极端情况)的性能损耗。

消除spin-lock影响的方法之一就是使用lock-free算法或者使用读-复制-修改(Read-Copy-Update, RCU)。但实际上lock-free算法很难开发，同时在内核中完全避免锁很难，尤其想要复用现有的内核组件如ZFS和BSD网络栈，因此，OSv的实现方法为：

1. 确保大部分的工作都在线程中完成，线程使用非自旋的互斥锁。

2. 使用lock-free算法实现互斥锁。

3. 线程调度器本身不能再线程中运行，为了保护其数据结构并不使用自旋锁，每个CPU维护一个线程运行队列并使用lock-free算法。

Preemptive

Osv完全支持抢占式的线程调度：线程可以通过waiting，yield或者唤醒其他线程自主地引起线程调度，也可以在任何时刻被其他中断抢占。线程也可以通过增加preempt-disable计数器来暂时避免被抢占。

Tick-less

大多数经典操作系统内核，甚至现代操作系统内核，使用一个周期性的计时器中断，也被称为tick。Tick周期性地引起线程调度，比如每秒100次，内核通过统计每个周期中每条线程的执行时间来确定下次的调度。

Tick技术简单方便，但也存在不足，最重要的一点，过多的Tick浪费CPU时间，尤其在中断响应明显比物理机上更慢的虚拟机上，因为虚拟机的中断需要退出hypervisor执行。由于Tick技术的不足，OSv设计实现一种tickless的方法：调度器使用高精度的时钟计算每个线程准确的执行时间，而不是通过tick大概的估计执行时

浅谈混合内存页面管理机制

（彭甘为 https://blog.csdn.net/kevlnbb/article/details/101176540）

新型存储技术和新型存储器的出现，在工业界和学术界都引起了广泛的思潮。新型存储器（NVM）相比普通的DRAM密度更高，价格更便宜。但是NVM的读写性能相比DRAM会有较大的差距，所以主流的架构方案是将NVM和DRAM混合成异构内存。虽然相比纯DRAM会带来一定的性能损失，但是能换来较大的内存容量。所以，如何管理数据提高异构内存的整体性能成为了工业界和学术界面临的主要挑战。

对于性能敏感或者高footprint的云应用，可能就不太适合完全用NVM来代替DRAM做主存，因为一个小的访存延迟都会被放大成数倍的损失。然而很多的云应用的数据访问是高度不平衡的，也就是说很大一部分数据是不频繁

接上页

间。在此基础上，配合计时器中断和调度算法来计算下次线程切换的时间。除此之外，线程调度器通过滞后(hysteresis)来避免两个繁忙线程之前频繁的切换调度。默认滞后时间为2ms，相同优先级的两个线程交替使用4ms的时间片段，这样调度器不会产生多于500次的计时器中断。

Fair

线程调度器的功能就是确定运行哪个可运行的线程，以及运行该线程多长时间，一个公平的调度器应该确保每个线程的运行时间尽量相等，或者按照优先级大小成比例。然而，使用线程总的运行时间作为指标将很快导致失衡，举个例子，如果一个线程等待10s后成为可执行状态，它将会独占CPU整整10s。作为替代，OSv调度器计算每个线程最近运行时间的指数衰变移动平均，调度器选择移动平均最低的线程作为下次运行的线程，并计算要运行的时间以确保不会超过下一个线程的运行时间。

Scalable

OSv调度器维护每个CPU运行线程队列的时间复杂度为O(lgN)：通过堆线程队列按照线程移动平均运行时间进行排序来实现。由于计算移动平均时间需要进行浮点运算，其最大的障碍就是可扩展性，调度器每次更新所有线程的移动平均运行时间是不切实际的，因此调度器每次只更新一个线程的运行时间。

Efficient

除了调度器的可扩展性，OSv还使用一些技术来提升线程调度的效率，其中包括：

1. OSv单地址空间特性意味着在进行上下文切换时不需要切换页表和flush TLB，从而减少开销。

2. 上下文切换时保存浮点运算单元(FPU)寄存器代价也很高，而大部分调度都是运行线程自发产生的，这种情况下x86_64 ABI确保FPU是由调用者保存的，因此省略FPU保存。

总结：OSv线程调度模型的设计要点都以云环境的特性为落脚点，抢占式多线程确保能够最大限度地提高云环境SMP机器的利用率，lock-free和tick-less使得线程调度的开销降至最低，同时，使用基于指数衰变的方法来计算线程调度的时间和安排运行时间，尽可能地实现线程调度公平的最大化，是一个很大的创新点。

访问的。那么对于这些访问的不频繁的数据就可以放置到访问的较慢的内存，由于对这些数据访问的不频繁，所以即使放置在NVM也不会对总体性能造成较大影响。我们把这些不常访问的数据称为冷数据。

因此，如何区分出访问较少的冷数据和访问较多的热数据就成了提高异构内存性能的关键。现有的机制是通过利用PTE中的访问位（Accessed bit），这个访问位是由硬件每次在访问该页面的时候设置。然而，实验表明仅通过访问位是不足以准确的区分出热数据和冷数据。另一方面，通过访问位来监测一个页面的访问情况，需要频繁的清除访问位和刷新TLB，这就会导致一个较大的开销问题。

另一方面，如今的64位x86架构为4KB的页面粒度提供了四级的页表结构，所以当TLB发生miss的时候，可能会引起对页表的四次访问从而引起四次访存。在虚拟化的环境下甚至可以达到24次访存。当采用2MB的大页粒度时，最坏的情况可以减少至15次，这大幅降低了虚拟化的性能开销。再者，2MB大页粒度优化了页表中间层次的可缓存性。因此，大页的使用对于虚拟化环境是非常有必要的。这也是异构内存在云应用场景需要考虑的因素。

为了区分内存中访问较多的热数据，可以考虑周期性随机选择页面的部分子集进行访问频度监测，难点在于如何处理2MB的大页并且控制在一个较小的性能开销。为了控制热度监测对性能的影响，一次只考虑对一部分页面抽样进行监测。对大页的抽样，可以考虑对随机一部分大页划分成若干4KB小页，然后对这些小页部分进行“埋雷”。然而这种做法对于大页中只有部分页面是热页的情况将不适用，因为当恰好选到的不是这些4KB热页，它会认为这个大页是个冷页。更好的解决办法是分两步来解决，首先根据硬件提供的信息（Accessed bit）对每个大页的512个小页筛选出访问过的页面，第二步对这些筛选出的页面进行监测，从而较为准确的得出大页的合计访问频率。

当前的x86架构并不支持页面粒度的访问计数，所以考虑从软件的角度去解决。BadgerTrap是可以拦截TLB缺失的工具。当某个页面被抽样作为访问计数，我们可以考虑将它设置成为不可访问，同时刷新TLB。这一点可以通过设置PTE的保留位来实现。那么下次再对这个页面访问的时候将会由于TLB缺失触发一次硬件的page walk，BadgerTrap将拦截这次错误并同时恢复该页面的访问。通过对BadgerTrap的拦截计数，我们就可以估计出该页面的访问次数。

当得到页面的访问计数的估计值，我们就可以对页面进行冷热页的划分。针对不同的场景，用户设定一个特定的可容忍的内存性能损耗slowdown值（记为x%）。对于这个值x%，可以做如下的转换：假设对较慢的内存（NVM）一秒访问A次，而NVM的访问延时记为t_s，那耗费在较慢的内存上的时间就是A t_s。因此x%的阈值就可以理解成访问的频率最高为每秒x/100* t_s，如果所有大页的某个比例（记为f）的部分被抽样，那只要某个大页的总计访问频率不超过f*x/100 t_s，我们就可以把页面放在较慢的内存。接着对所有抽样出的大页按照之前所说的方法估算出的访问频率由小到大排序，将访问较少的冷页放置到较慢的内存直到剩下的页面访问频度超过阈值。

上面所说的大页访问频率估计方法是基于大页中的一小部分4KB页面，这就可能会导致将一些热页错误的分类为冷页。为了解决这个问题，可以对冷页的访问频率进行监测，使用的方法还是之前所说。由于这些页面的访问频率比较低，所以监测开销很小。在每次抽样的周期，对NVM里的大页按照访问计数进行排序，将访问的最频繁的页面迁移至DRAM直到剩下的页面访问频度低于阈值。

深入浅出解析RDMA

（谭 杰 https://blog.csdn.net/qq_21125183/article/details/80563463）

RDMA(Remote Direct Memory Access)技术全称远程直接内存访问，就是为了解决网络传输中客户端与服务器端数据处理的延迟而产生的。它将数据直接从一台计算机的内存传输到另一台计算机，无需双方操作系统的介入。这允许高吞吐、低延迟的网络通信，尤其适合在大规模并行计算机集群中使用。RDMA通过网络把资料直接传入计算机的内存中，将数据从一个系统快速移动到远程系统内存中，而不对操作系统造成任何影响，这样就不需要用到多少计算机的处理能力。它消除了数据包在用户空间和内核空间复制和上下文切换的开销，因而能解放内存带宽和CPU 周期用于改进应用系统性能。

一、背景介绍

1.1 传统TCP/IP通信模式

传统的TCP/IP网络通信，数据需要通过用户空间发送到远程机器的用户空间。数据发送方需要将数据从用户应用空间Buffer复制到内核空间的Socket Buffer中。然后在内核空间中添加数据包头，进行数据封装。通过一系列多层网络协议的数据包处理工作，这些协议包括传输控制协议（TCP）、用户数据报协议（UDP）、互联网协议（IP）以及互联网控制消息协议（ICMP）等。数据才被发送到NIC网卡中的Buffer 进行网络传输。消息接受方接受从远程机器发送的数据包后，要将数据包从NIC Buffer中复制数据到Socket Buffer。然后经过一系列的多层网络协议进行数据包的解析工作。解析后的数据被复制到相应位置的用户空间Buffer。这个时候再进行系统上下文切换，用户应用程序才被调用。以上就是传统的TCP/IP协议层的工作。

然而如今随着社会的发展，我们希望更快和更轻量级的网络通信。

1.2 通信网络定义

计算机网络通信中最重要两个衡量指标主要是指高带宽和低延迟。通信延迟主要是指：处理延迟和网络传输延迟。处理延迟开销指的就是消息在发送和接收阶段的处理时间。网络传输延迟指的就是消息在发送和接收方的网络传输时延。如果网络通信状况很好的情况下，网络基本上可以 达到高带宽和低延迟。

1.3 当今网络现状

当今随着计算机网络的发展。消息通信主要分为两类消息，一类是Large Messages，在这类消息通信中，网络传输延迟占整个通信中的主导位置。还有一类消息是Small Messages，在这类消息通信中，消息发送端和接受端的处理开销占整个通信的主导地位。然而在现实计算机网络中的通信场景中，主要是以发送小消息为主。所有说发送消息和接受消息的处理开销占整个通信的主导的地位。具体来说，处理开销指的是buffer管理、在不同内存空间中消息复制、以及消息发送完成后的系统中断。

1.4 传统TCP/IP存在的问题

传统的TPC/IP存在的问题主要是指I/O bottleneck瓶颈问题。在高速网络条件下，与网络I/O相关的处理的高开销，限制了可以在机器之间发送的带宽。这里高额开销是数据移动操

作和复制操作。具体来讲，主要是传统的TCP/IP网络通信是通过内核发送消息。Messaging passing through kernel这种方式会导致很低的性能和很低的灵活性。其中性能低下的原因主要是由于网络通信通过内核传递，这种通信方式存在的很高的数据移动和数据复制的开销。并且现如今内存带宽性相较如CPU带宽和网络带宽有着很大的差异。其中很低的灵活性的原因主要是所有网络通信协议通过内核传递，这种方式很难去支持新的网络协议和新的消息通信协议以及发送和接收接口。

二、相关工作

高性能网络通信历史发展主要有以下四个方面：TCP Offloading Engine（TOE）、User-Net Networking（U-Net）、Virtual interface Architecture （VIA）、Remote Direct Memory Access（RDMA）。U-Net是第一个跨过内核网络通信的模式之一。VIA首次提出了标准化user-level的网络通信模式，其次它组合了U-Net 接口和远程DMA设备。RDMA就是现代化高性能网络通信技术。

2.1 TCP Offloading Engine

在主机通过网络进行通信的过程中，主机处理器需要耗费大量资源进行多层网络协议的数据包处理工作，这些协议包括传输控制协议（TCP）、用户数据报协议（UDP）、互联网协议（IP）以及互联网控制消息协议（ICMP）等。由于CPU需要进行繁重的封装网络数据包协议，为了将占用的这部分主机处理器资源解放出来专注于其他应用，人们发明了TOE（TCP/IP Offloading Engine）技术，将上述主机处理器的工作转移到网卡上。

这种技术需要特定网络接口–网卡支持这种Offloading操作。这种特定网卡能够支持封装多层网络协议的数据包，这个功能常见于高速以太网接口上，如吉比特以太网（GbE）或10吉比特以太网（10GbE）。

2.2 User–Net Networking(U–Net)

U-Net的设计目标是将协议处理部分移动到用户空间去处理。这种方式避免了用户空间将数据移动和复制到内核空间的开销。它的设计宗旨就是移动整个协议栈到用户空间中去，并且从数据通信路径中彻底删除内核。这种设计带来了高性能的提升和高灵活性的提升。

U-Net的virtual NI 为每个进程提供了一种拥有网络接口的错觉，内核接口只涉及到连接步骤。传统上的网络，内核控制整个网络通信，所有的通信都需要通过内核来传递。U-Net应用程序可以通过MUX直接访问网络，应用程序通过MUX直接访问内核，而不需要将数据移动和复制到内核空间中去。

三、RDMA详解

RDMA(Remote Direct Memory Access)技术全称远程直接内存访问，就是为了解决网络传输中服务器端数据处理的延迟而产生的。

RDMA主要有以下三个特性：Low-Latency、Low CPU Overhead、High Bandwidth。

3.1 RDMA 简介

1. Remote：数据通过网络与远程机器间进行数据传输。

2. Direct：没有内核的参与，有关发送传输的所有内容都卸载到网卡上。

3. Memory：在用户空间虚拟内存与RNIC网卡直接进行数据传输不涉及到系统内核，没有额外的数据移动和复制。

4. Access：send、receive、read、write、atomic操作。

3.2 RDMA基本概念

RDMA有两种基本操作：

1. Memory verbs：包括RDMA read、write和atomic操作。这些操作指定远程地址进行操作并且绕过接收者的CPU。

2. Messaging verbs：包括RDMA send、receive操作。这些动作涉及接收方的CPU，发送的数据被写入由接收方的CPU先前发布的接受所指定的地址。

RDMA传输分为可靠和不可靠的，并且可以连接和不连接的（数据报）。凭借可靠的传输，NIC使用确认消息来保证消息的按序传送。不可靠的传输不提供这样的保证。然而，像InfiniBand这样的现代RDMA实现使用了一个无损链路层，它可以防止使用链路层基于拥塞流量控制的损失，以及使用链路层重传的基于位错误的损失。因此，不可靠的传输会很少会丢弃数据包。

目前的RDMA硬件提供一种数据报传输：不可靠的数据报（UD），并且不支持Memory verbs。

3.3 RDMA技术

传统上的RDMA技术设计内核封装多层网络协议并且涉及内核数据传输。RDMA通过专有的RDMA网卡RNIC，绕过内核直接从用户空间访问RDMA enabled NIC网卡。RDMA提供一个专有的Verbs Interface而不是传统的TCP/IP Socket Interface。要使用RDMA首先要建立从RDMA到应用程序内存的数据路径，可以通过RDMA专有的Verbs Interface接口来建立这些数据路径，一旦数据路径建立后，就可以直接访问用户空间Buffer。

四、RDMA 技术细节

4.1 RDMA技术详解

RDMA的工作过程如下：

1. 当一个应用程序执行RDMA读或写请求时，不执行任何数据复制。在不需要任何内核内存参与的条件下，RDMA请求从运行在用户空间中的应用中发送到本地NIC（网卡NIC读取缓冲的内容），并通过网络传送到远程NIC。

2. 在网络上传输的RDMA信息包含目标虚拟地址、内存钥匙和数据本身。请求既可以完全在用户空间中处理（通过轮询用户级完成排列），又或者在应用一直睡眠到请求完成时的情况下通过系统中断处理，RDMA操作使应用可以从一个远程应用的内存中读数据或向这个内存写数据。

3. 目标NIC确认内存钥匙，直接将数据写入应用缓存中，用于操作的远程虚拟内存地址包含在RDMA信息中。

4.2 RDMA操作细节

RDMA提供了基于消息队列的点对点通信，每个应用都可以直接获取自己的消息，无需操作系统和协议栈的介入。

消息服务建立在通信双方本端和远端应用之间创建的Channel-IO连接之上。当应用需要通信时，就会创建一条Channel连接，每条Channel的首尾端点是两对Queue Pairs（QP）。每对QP由Send Queue（SQ）和Receive Queue（RQ）构成，这些队列中管理着各种类型的消息。QP会被映射到应用的虚拟地址空间，使得应用直接通过它访问RNIC网卡。除了QP描述的两种基本队列之外，RDMA还提供一种队列Complete Queue（CQ），CQ用来知会用户WQ上的消息已经被处理完。

RDMA提供了一套软件传输接口，方便用户创建传输请求Work Request（WR），WR中描述了应用希望传输到Channel对端的消息内容，WR通知QP中的某个队列Work Queue（WQ）。在WQ中，用户的WR被转化为Work Queue Element（WQE）的格式，等待RNIC的异步调度解析，并从WQE指向的Buffer中拿到真正的消息发送到Channel对端。

源代码漏洞分析数据处理

（唐　静 https://blog.csdn.net/weixin_43334908/article/details/100068971）

一、介绍

近年来安全漏洞数量逐年攀升，源代码静态分析被广泛应用于软件中的漏洞检测，但纯粹的静态数据分析耗时且费力，而利用深度学习的方法自动检测出源代码中的漏洞的方法则逐渐凸显优势。使用的源代码漏洞数据均为C/C++源程序，利用joern针对某些与漏洞相关的API库函数为关注点提取切片，并根据源程序漏洞记录文件对提取的切片做有无漏洞的标签记录。对切片做预处理生成向量之后输入到神经网络中训练并预测漏洞位置。

二、生成切片

1979年，美国Mark Weise首次在他的博士论文中提出了程序切片思想。程序切片是一种分解程序的程序分析技术，原理是根据某个关注点删除源程序中与该关注点无关的指令，保留含有数据依赖和控制依赖关系的指令。切片与源程序在一定切片准则下的语义是一致的。利用工具joern针对源程序中某些可能与漏洞相关的API库函数、表达式、指针、数组为关注点前向后向提取相关代码语句组成切片。

程序切片的类别根据概念分为可执行切片与不可执行切片，根据切片准则可分为静态切片、动态切片和有条件切片，根据关注点在切片中的位置可分为前向切片和后向切片，前向切片是提取出程序中关注点之后与关注点相关的语句，后向切片是提取出程序中关注点之前与关注点相关的语句。

三、数据处理

1、漏洞数据源

NVD和SARD，其中NVD包括软件产品中的漏洞，SARD包括production, synthetic和academic security的漏洞程序，对于NVD中的每个漏洞程序，可以根据diff文件获取漏洞程序及相应的修补程序，文件中包含漏洞程序与修补程序的差异。对于SARD中的源程序，包括无漏洞的程序、有漏洞的程序以及有漏洞的程序和其修补版本的混合程序。数据集中总共包含368个与CVE相关的开源程序，14000个SARD程序，这些程序包含126种漏洞类型。

	切片总数	无漏洞切片数	有漏洞切片数
DDCD	68353	54667	13686
DD	98262	83152	15110

126种漏洞类型对应的CWE-id如下所示：

CWE-015, CWE-020, CWE-022, CWE-023, CWE-036, CWE-078, CWE-080, CWE-088, CWE-089, CWE-090, CWE-114, CWE-119, CWE-120, CWE-121, CWE-122, CWE-123, CWE-124, CWE-126, CWE-127, CWE-129, CWE-134, CWE-170, CWE-176, CWE-188, CWE-190, CWE-191, CWE-194, CWE-195, CWE-196, CWE-197, CWE-222, CWE-223, CWE-242, CWE-244, CWE-252, CWE-253, CWE-256, CWE-259, CWE-272, CWE-284, CWE-319, CWE-321, CWE-325, CWE-327, CWE-338, CWE-345, CWE-362, CWE-363, CWE-364, CWE-366, CWE-367, CWE-369, CWE-377, CWE-398, CWE-400, CWE-401, CWE-404, CWE-412, CWE-414, CWE-415, CWE-416, CWE-426, CWE-427, CWE-457, CWE-459, CWE-464, CWE-467, CWE-468, CWE-469, CWE-475, CWE-476, CWE-479, CWE-489, CWE-506, CWE-510, CWE-526, CWE-534, CWE-535, CWE-543, CWE-562, CWE-571, CWE-587, CWE-588, CWE-590, CWE-591, CWE-605, CWE-606, CWE-609, CWE-617, CWE-620, CWE-663, CWE-665, CWE-666, CWE-672, CWE-674, CWE-675, CWE-680, CWE-681, CWE-682, CWE-685, CWE-688, CWE-690, CWE-704, CWE-758, CWE-761, CWE-762, CWE-765, CWE-771, CWE-773, CWE-774, CWE-775, CWE-780, CWE-785, CWE-789, CWE-805,

CWE-806, CWE-821, CWE-822, CWE-824, CWE-828, CWE-831, CWE-833, CWE-834, CWE-835, CWE-839, CWE-843.

具体的切片数据参见[1]。

2、生成语料库

数据处理主要是对由源代码得到的切片打标签，并对切片语句做分词处理生成语料库。打标签是根据源程序的漏洞行记录文件对提取的切片中是否含有漏洞行做0/1标记，0表示该切片无漏洞，1表示该切片有漏洞。

生成语料库的具体工作包括：首先对切片做去重处理，当切片语料相同且切片的标签相同时只保留一个切片，其余删除；当切片有相同的语料但标签不同时，删除这些切片。其二，删除语句中的注释，并按行做分词处理，之后将切片中自定义的变量名和函数名映射为统一的形式以避免其在漏洞分析过程中对语义信息造成干扰。最后，记录语料库信息、切片标签、关注点等信息用于后续的深度学习训练和测试。

3、转向量

输入神经网络模型进行训练之前，需要将切片数据转换成对应的向量，首先需要训练词向量模型，读取语料库，利用gensim训练Word2vec 模型，Word2vec模型是将所有的词向量化，从而定量地描述词与词之间的关系，一般包括CBOW (Continuous Bag-of-Words)和Skip-Gram两种模型，CBOW模型是根据上下文预测中心词，Skip-Gram则是根据中心词预测上下文。Word2vec 模型在训练过程中需要调参，主要的参数如下所示：

size：特征向量的维度，默认值为100

alpha：初始学习率（训练过程中线性下降为min_alpha），默认值为0.025

window：当前词与句中预测词的最大距离，默认为5

min_count：低频率词汇忽略阈值，默认值为5

workers：训练模型使用线程数，默认值为3

min_alpha：学习率的最低限度，默认值为0.0001

sample：高频词随机降采样阈值，默认值为0.001

max_vocab_size：在训练过程中限制RAM使用，默认值为None

seed：随机数种子，默认值为1

sg：训练算法，sg=0时使用CBOW算法，sg=1时使用skip-gram算法，默认为0

hs：hs=1时使用hiercrchical softmax算法实现模型训练，hs=0且negative为非零值时使用negative sampling，默认为0

negative：当大于0时negative sampling被使用，标识被丢弃的噪音词数量，默认值为5

cbow_mean：cbow_mean=0时使用语境词向量之和，1时使用平均值，默认值为1

hashfxn：用于随机初始权重的hash函数，默认为python自带的函数

iter：迭代次数，默认为5

trim_rule：词汇修剪规则，包括制定丢弃保留和默认规则，默认为None

sorted_vocab：sorted_vocab=1时在分配词汇序列之前梯度下降，默认值为1

batch_words：实例输入的batch大小，默认为10000

得到切片语料库中每个词对应的词向量，之后将语料中每个切片转换成一个固定维度的向量。由于切片长度不同，在输入神经网络模型之前，先将所有切片向量统一处理为相同长度的向量，对该数据集根据关注点处理，当切片长度不足设置的长度时，则在切片向量最后补0；当切片长度超过设置的长度时，则以关注点为中心向前向后做截断，超过的部分删除。

参考文献

[1] https://github.com/VulDeePecker/Comparative_Study.

[2] Zhen Li. A Comparative Study of Deep Learning-Based Vulnerability Detection System. IEEE Access 7: 103184-103197 (2019).

实验室2019年暑期年会召开

刘　芹

2019年7月31日至8月2日，实验室2019年暑期年会顺利召开，包括教师、博士后、博士生、硕士生在内的300余人参会。自实验室2001年成立至今，暑期年会每年召开一次，今年是第十九届。

实验室主任金海教授主持会议并在开幕式上致辞，他简要介绍了实验室文化与基本情况，回顾了实验室的历年暑期年会，并介绍了本次年会的会议议程。开幕式后，来自丹麦哥本哈根大学的周永銮教授作了题为“Rethinking Modern DataBase Systems”的大会特邀报告。

接着，实验室各研发组的代表老师依次作了各科研方向前瞻研究报告。张腾博士作了题为“强化学习研究前沿简介”的报告，蒋文斌教授作了题为“AI时代系统结构的挑战与机遇”的报告，羌卫中副教授作了题为“硬件设计对系统安全的影响”的报告，吴松教授作了题为“云计算及容器技术发展趋势思考、分析与展望”的报告。之后，各研发组的学生代表报告了过去一年来的项目进展和成果情况，以及下一步工作计划和方向规划等，并分别进行了系统演示。

针对各研发组的报告，师生踊跃发言，对其中的理论、方法和技术展开了热烈的讨论。金海教授从各方向的科研发展前沿提出了问题，对相关研究工作给予了指导。其他老师针对各研发组的工作也提出了宝贵的意见和建议。通过深入交流探讨，各研发组进一步明确了研究方向和工作重点、优化了研究计划，确保今后研究工作能够按计划稳步推进。

8月2日，张凡、易宇声等11名博士生分别汇报了各自的博士毕业论文撰写提纲和论文答辩计划，老师们针对汇报内容与博士生一一交流并提出了修改意见。大会最后由金海教授进行了总结，公布了经教师工作会议讨论形成的教师和研究生分组方案。

本次年会之前，7月28日至29日召开了实验室科研发展规划研讨会，实验室全体教师参加。会上，各组教师代表汇报了各组科研发展规划及个人发展规划。报告中，老师们以现有的研究工作为基础，从可持续发展的角度着重探讨未来的研究内容、论文计划、项目计划等。

实验室主任金海教授指出，各位老师要关注国家关键需求与企业重大需求，以实验室的未来发展为基点，做有价值、有影响力的科研。

刘　芹
主要负责实验室宣传等办公室事务
Email：liuqin19@hust.edu.cn

vNIDS: Towards Elastic Security with Safe and Efficient Virtualization of Network Intrusion Detection Systems

陈浩宇　推荐

“vNIDS: Towards Elastic Security with Safe and Efficient Virtualization of Network Intrusion Detection Systems”是国际顶级会议CCS’18收录的一篇论文，该论文将传统的网络入侵检测系统（Network Intrusion Detection System, NIDS）按照检测内容的差别分割成三种不同的微服务，通过按需组合实现弹性的入侵检测服务，从而满足高性能的网络流量处理需求。

传统的NIDS基于硬件进行设计和实现，使用时在网络中的固定位置部署固定数量的硬件设备，虽然可以满足处理的需求，但是不够灵活，没法进行按需调整，流量大时会出现性能瓶颈，流量低时又会出现硬件资源的浪费。而已有的一些虚拟化NIDS的方案实现了同时虚拟化多个入侵检测系统实例并运行在不同的虚拟机中，将网络流量按照一定的转发算法进行分割后，分配给不同的虚拟化NIDS实例进行处理。该方案存在两个问题：第一，不同的NIDS实例中的状态无法实现共享。例如使用计数器进行端口扫描次数的统计，然而不同实例中的计数器无法进行同步，因此可能导致统计的端口访问次数在每个实例中均低于真实扫描次数，最终漏掉了一些攻击；第二，基于转发算法进行流量分配通常在路由器/交换机中分析数据包头部，丢失了数据包载荷中的有用信息，从而使分配不合理。例如针对cookie劫持攻击的检测需要基于cookie token信息，然而该信息存在于数据包载荷中，无法在路由器/交换器中进行解析。

即使已有工作基于软件定义网络（Software-Defined Networks，SDN）和网络功能虚拟化（Network Function Virtualization，NFV）提出了弹性的安全部署方案，能够利用虚拟机或者容器docker实现网络功能的灵活部署，同时为不同的实例提供了一定的状态共享能力。但是它们依然是将网络功能看作是一个聚合的整体（即包解码、重组、头部分析、载荷分析、匹配、记录等数据包处理功能的集合），这也存在两个问题：第一，效率低下的负载均衡。首先，虚拟化的NIDS功能在负载均衡时必须暂停并且流量必须进行缓存，这就给了攻击者可乘之机，可以改变流量大小不断触发负载均衡机制，从而使服务不可用；其次，在NIDS内部也不是所有模块都被充分利用，通常性能瓶颈会在解码模块和规则匹配模块处出现，以聚合的实例整体进行负载均衡明显粒度太大。第二，效率低下的资源分配。例如，一个NIDS由四个组件构成，每个组件需要一个CPU核心来运行；如果某个物理机只有三个或者更少的CPU核心，那么就无法运行该NIDS。

为了解决上述问题，本文致力于提出一种基于微服务的更加细粒度可定制的虚拟网络入侵检测系统vNIDS，将聚合式的NIDS拆解成一些可分开部署的更小的模块，同时支持原本的

检测策略。vNIDS的框架如图1所示，最下方是vNIDS微服务，将原NIDS中的功能按照数据包的处理内容划分成三种微服务：基于头部检测的微服务、协议分析微服务和基于载荷检测的微服务，它们通过共享的数据存储来共享检测状态。中间层则是vNIDS的控制器，主要负责两件事情：（1）管理和调度不同微服务之间需要共享的状态；（2）将检测逻辑程序进行划分和部署。最上方则是管理员通过vNIDS控制器的接口部署的检测逻辑程序。该工作的主要贡献有两点，一是提出了基于微服务的NIDS划分方案，二是提出了检测逻辑程序（Detection Logic Program, DLP）的划分。下面分别对这两点贡献进行介绍。

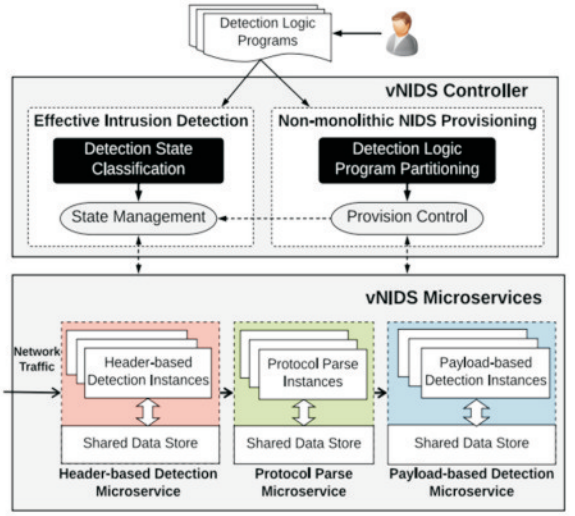


图1 vNIDS框架整体展示

基于微服务的划分是将NIDS划分成了基于数据包头检测的微服务、协议分析微服务和基于数据包载荷检测的微服务，三者需要按照给定的先后顺序进行排布，存在先后依赖关系。如图2所示，不同的检测逻辑（Detection Logics）需要不同的微服务链。例如，针对DNS隧道和木马检测的功能，需要同时启用三种微服务；而针对cookie劫持的攻击检测，则可以不需要头部的检测，只需要进行协议分析后再进行载荷分析。同时，为了实现微服务之间的数据共享，该工作对NIDS中的状态进行了分类。首先，NIDS中的状态均是存储在变量定义中，因此状态分类即对变量进行分类。论文将程序中与检测相关的变量分成三种。第一种是单数据包的相关变量，通常只使用一次后就进行回收；第二种是单条流的相关变量，即一条流中许多数据包相关的状态，通常只会在流终止之前进行回收；第三种是多条流的相关变量，通常可能会在某条流产生之前就已进行定义或者在流终止之后才进行回收。基于该分类原理，论文对NIDS程序中的变量进行自动分析，找出三种不同的状态变量，并按照处理场景的需求进行共享。

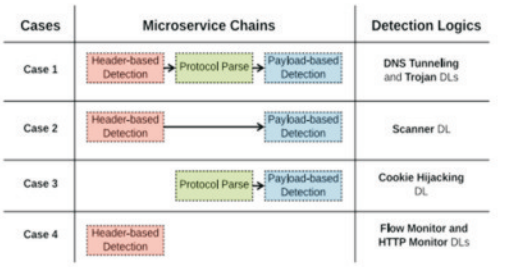


图2 微服务链使用场景示例

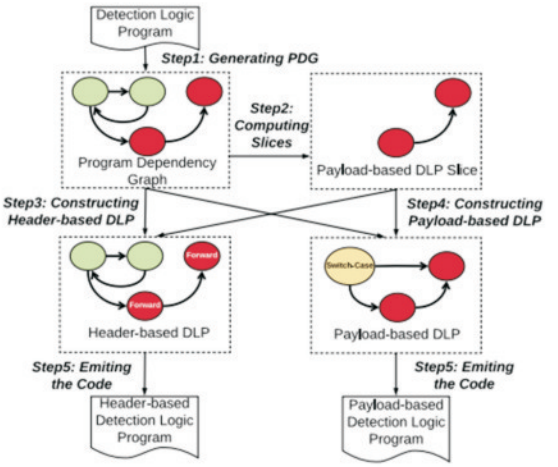


图3 逻辑检测程序DLP的划分步骤

逻辑检测程序DLP的划分则分为五步，如图3所示：

TEEv: Virtualizing Trusted Execution Environments on Mobile Platforms

罗瑞坤　推荐

《TEEv: Virtualizing Trusted Execution Environments on Mobile Platforms》是发表在操作系统与虚拟化领域的著名会议VEE 2019上的一篇Best Paper，该会议于2019年4月14日在美国普罗维登斯市召开。

近年来，随着TEE（Trusted Execution Environment，可信执行环境）在智能设备领域的广泛普及，TEE逐渐从由设备厂商控制、安全功能相对单一的封闭安全系统过渡到能够支持多应用、复杂功能的安全开放环境。然而这种趋势下TEE正在面临两大安全挑战：一是随着安全应用的不断增多，TEE的攻击面也将不断扩大，导致TEE可能存在潜在的安全问题；二是由于缺乏可信的隔离，TEE提供方、设备厂商与安全应用提供方之间难以达成互信。

针对这些问题，一种可能的思路是通过安全设计，进一步增强TEE的安全性以及TEE与安全应用的隔离性。然而实际在产业中，全球智能设备领域已经有来自不同厂商的超过10款TEE产品在不同领域都有规模不小的应用部署，而且这些产品大部分都是闭源商业化维护的，一对一地进行安全设计与增强的方案实际

接上页

（1）利用现有工具生成程序的程序依赖关系图（Program Dependency Graph，PDG）；

（2）计算分片。利用程序分片工具，从DLP的PDG中划分出只跟数据包载荷相关的处理逻辑DL；

（3）在划分后的PDG中，对基于数据包头部检测的DLP进行构造，同时定义它们与基于数据包载荷检测的DL的调用和跳转关系；

（4）在划分后的PDG中对基于数据包载荷检测的DLP进行构造，定义为第三步中构造的基于数据包头部检测的DLP提供的调用和跳转接口；

（5）部署划分后的基于数据包头部和基于数据包载荷检测的DLP代码。

更多的细节和算法在论文中有详细说明，由于空间限制不进行展开，感兴趣的读者可以去论文中查看。本文提出了基于微服务的细粒度NIDS划分方式，但是依然存在问题，例如基于处理内容的划分方式的合理性存疑，即使可以满足NIDS的处理需求，但是不便于管理员的理解，可以考虑采用事件驱动的划分方式；同时，每个微服务均需要来自于现有代码的分割，并且对代码的规范性有非常高的要求，根本原因是没有提出每个微服务的模型框架，为从头开始编写微服务提供一套标准而不是基于现有程序代码进行分割，这些都是未来可以研究的方向。

陈浩宇
2014级博士
研究方向：安全虚拟化与可视化
Email：haoyuchen@hust.edu.cn

上的实施可行性不高。

第二种思路是在同一系统中支持多个来自不同厂商、互相隔离的TEE，并且对TEE的高权限进行降权。这种方案不仅能够达到安全的目的，还能够更好地支持TEE产业应用不断丰富的需求。

TEEv就是基于第二种思路进行设计实现的。不同于传统的虚拟化方案，TEEv在设计上有几大挑战，包括：

· 设计挑战：ARM平台目前不支持TEE的硬件虚拟化；

· 安全性挑战：新系统的可信基必须足够小；

· 兼容性挑战：能够支持实际产业中不同的TEE产品，支持不同安全场景。

如图1所示，在系统设计上，TEEv在TEE之下运行一个轻量级的虚拟化层TEE-visor，TEE-visor在ARM的监控模式和内核模式中运行，和TEE基本是在同一权限级。通过剥离原有TEE中的敏感指令至TEE-visor来达到对TEE的降权，同时利用MMU软件虚拟化技术，由TEE-visor维护和控制最终的所有内存映射关系，以此达到同级隔离、管理的能力。此外，TEE-visor和TEE之间的切换逻辑需要保证原子性、唯一性和完整性，TEEv通过多个关键技术保证了即使TEE被攻破的情况下TEE-visor以及REE的隔离性依然无法被破坏。

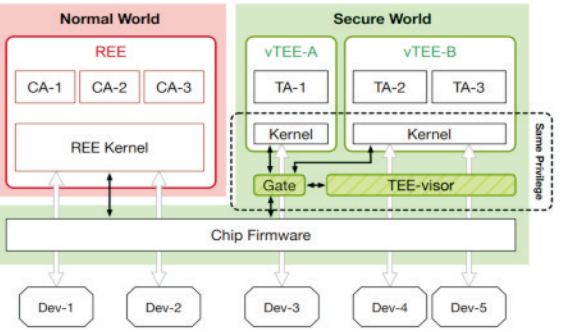


图1 TEEv架构

在实际的产业中，不同TEE的安全驱动能力具有多样化的特点，如图2所示，TEEv同时支持代理、前端后端、直通三种不同的IO设备虚拟化管理模型，满足不同场景、不同安全要求下的快速适配部署能力。

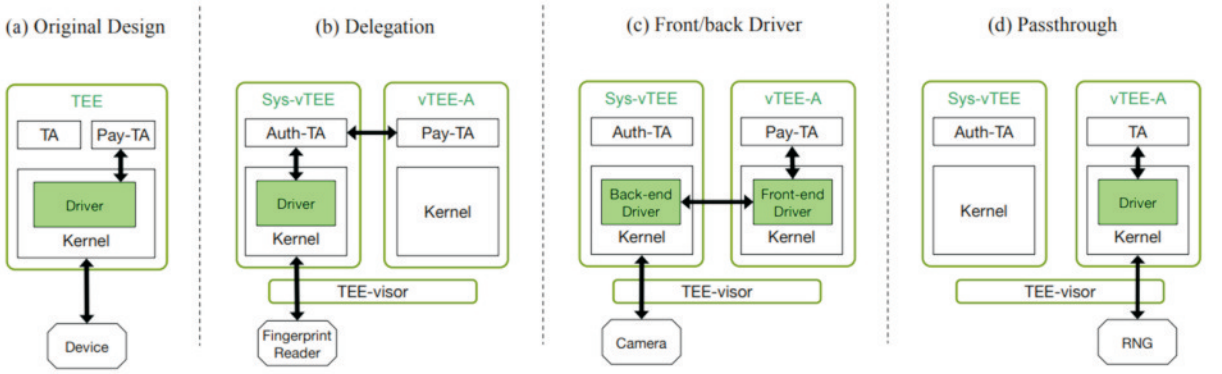


图2 TEEv对不同设备的I/O管理模式

在TEEv的实际评测中，瓶钵团队已商业化部署在数亿台智能设备的TEE产品T6和开源的OP-TEE都能够被快速部署并行运行在TEEv的平台之上，无缝支持Android KeyMaster、指纹、微信支付、支付宝支付等不同的实际安全应用。

罗瑞坤
2018级博士研究生
研究方向：云计算、边缘计算
Email：rkluo@hust.edu.cn

ScatterCache: Thwarting Cache Attacks viaCache Set Randomization

张 盼 推荐

“ScatterCache: Thwarting Cache Attacks via Cache Set Randomization”是国际顶级会议USENIX Security'19年收录的论文，该论文提出了一种全新的基于组随机化的缓存策略——SCATTERCACHE来抵御缓存攻击。

缓存桥接了CPU和主存间的巨大性能鸿沟，是当前计算架构的重要组成部分。通常情况下，若系统访问的目标数据存在缓存中，其访问所需要消耗的时间会少于数据不存在缓存内的情况。因而在最近的20年里，非常多的攻击利用缓存作为侧信道来泄露应用的敏感信息。之前，研究人员对缓存侧信道攻击的研究集中在加密算法，目的主要是泄露用户密钥。近年来，许多基于PRIME+PROBE和FLUSH+RELOAD的缓存攻击也被攻击者用于绕过程序地址空间随机化（ASLR）的保护，或泄露用户键盘输入和推测程序输入等。一种比较简单的基于缓存的侧信道攻击被称为“convert channel attacks”。在这种攻击中，“victim”在执行过程中不断的将用户的隐私数据通过一个私密的信道发送给攻击者。最近爆料的CPU漏洞，Meltdown和Spectre攻击都可以利用短暂的隐秘的缓存级的信道将私密信息发送给攻击者。为了应对缓存级的侧信道攻击，当前很多研究工作都将研究策略定位在软件层，在不修改硬件的情况下，通过重新设计软件，来避免地址的信息泄露。如，基于控制流一致的防御策略或者基于比特位划分的数据访问防御策略。然而广泛的将这些方法用于其它软件是很难的。首先，每个软件都存在不同点，不同软件存在的泄漏方式可能是不同，因而需要根据软件去对应的修改策略。因而在开发过程中除了软件功能实现，开发人员还需要关注系统架构，并针对不同平台的特性设计应用，来保护私密数据。这些策略会给软件开发人员带来很大的负担。另外一种思路是重新设计一套更有弹性的缓存架构，如，破坏内存地址和缓存索引的关联关系；或者专门提供额外的缓存空间用于关键代码。重新设计的底层缓存机制对软件透明，能很大的减轻开发的负担。然而目前基于这类方法提出的解决方案都或多或少的存在拓展性和性能方面的问题。这篇文章提出了称为SCATTERCACHE的缓存设计方案，通过给缓存添加一个简单的拓展，可以支持生成伪随机的缓存索引，从而能在拓展性和性能方面达到令人满意的结果。

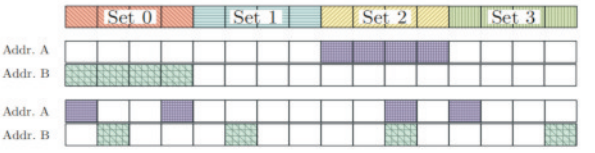


图1 扁平化的可视化表示地址到缓存组的映射关系。图中缓存为4路共16行。TOP：标准的缓存算法，使用索引位选择缓存组。Middle：伪随机的将地址映射到缓存组，但是缓存行到组的关系依然是静态的。Bottom：SCATTERCACHE伪随机的将地址映射到一个缓存行中，动态的形成缓存组。

作者认为，cache级别的侧信道攻击的根源在于传统的地址到缓存组的映射算法是可逆的，攻击者能够通过缓存组反推被访问的代码/数据地址。因而，如果能够重新设计一种基于随机化的地址到缓存组的映射实现，使得物理地址和缓存组不再有明显的映射关系，就能很好的抵御这类攻击了。

为了满足安全需求，SCATTERCACHE在设

计时从两个方面进行考虑。首先，地址映射到缓存组的映射方法应该是基于密钥的安全区域感知的，第二，构成不同缓存组的缓存行应该是不固定的。图1展现了这种想法与传统的相关工作的差异。传统的设计方法和其它提出的基于伪随机的地址映射方法都会静态的将缓存行分配给缓存组。因此，地址通过基于索引位的方法选择缓存组后，每次都仍然是使用那些相同的缓存行。这意味着PRIME-PROBE类型的攻击依然可被使用。而在SCATTERCACHE的设计中，每次访问都会通过伪随机算法从所有缓存行中进行选择。因此，通过缓存组反推访问地址会变得很困难。这里仍然使用伪随机来生成索引的原因是因为它可以抵抗“birthday-bound”的索引冲突，避免非常频繁的页抖动。图2是SCATTERCACHE设计的基本想法。IDF（Index Derivation Function）通过一个简单的加密策略进行地址映射操作。对于大小为n_{ways}组的缓存，IDF会为每次访问对应的输出n_{ways}索引。SCATTERCACHE 中，某地址的最终映射关系由被使用的密钥，SDID（Security Domain Identifier）和IDF共同决定。理论上，整个想法其实只需使用密钥就能管用，但是，SDID的引入使得安全性更加的健壮而且更不易被攻击者利用。密钥被保存在硬件中，密钥通常相比于SDID，长度更长，而且更低频率的被切换。另一方面，SDID由软件进行管理，基于被使用的策略，可以被频繁的更新。

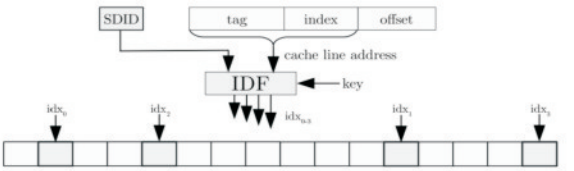


图2 想法：使用密码算法的IDF为地址输出对应的缓存组。IDF有效的将地址到缓存组的映射关系进行了随机混淆。

IDF的选择会影响到整个系统的安全和性能。从安全性方面来说，IDF应使得物理地址到索引的映射是不可预测的，即便在攻击者知道了除了秘钥以外的所有细节的情形。加密算法，如块加密，MACs和散列算法能够提供这种安全属性。从性能方面考虑，作者推荐了PRINCE，一种低延时的块加密算法；Keccak-*p*一种sponge-based的MAC算法。

对于缓存替换策略，SCATTERCACHE使用简单的随机替换策略来简化安全分析。同时对部分不可逆的IDF算法，原始的索引需要使用额外的位来进行标记，以方便将脏的缓存行写回，从而保证正确的缓存查询。在他们的测试中，这部分添加的位相比于传统的缓存设计来说仅带来了额外5%的开销。总的来说，SCATTERCACHE的设计其实非常接近于传统的组关联架构设计。明显的不同点是拥有有一个更加复杂的IDF。

秘钥在SCATTERCACHE的设计中起到了非常关键的作用。极端情况下，即便SDID被攻击者所知，它仍能够为物理地址构建随机的驱逐集，从而避免攻击者通过行碰撞信息绕过地址计算。因而秘钥的保存，对于整个设计的安全性来说，是非常关键的。SCATTERCACHE将秘钥的维护完全交给硬件去处理。每次硬件通电时，会生成一个新的秘钥交由IDF使用。秘钥在系统运行过程中也能进行更新。但是会根据不同的缓存配置，IDF会有不同的操作。对于write-through的缓存，秘钥能够在任意时间直接进行更新，不需要考虑数据一致性的问题。对于write-back的缓存，由于缓存中存在脏数据，因而在秘钥更新前，需要首先将所有数据写回。在x86指令集中，这种操作可以通过WBINVD指令直接实现。

作者在文章中通过详细的推导证明SCATTERCACHE的缓存组随机化策略的安全性。本文只对论文的大概思路进行一个简单的介绍，其中还有更多的细节由于篇幅的原因没有办法进一步介绍，感兴趣的同学可以对这篇论文进行更加详尽的阅读。

张　盼
2014级博士
研究方向：系统安全与可信执行环境
Email：panzhanghust@gmail.com

科研就像与大师有约

费新财

可能我们来自世界各地，身处五湖四海；可能你正迎着朝阳，而我却在挑灯夜战；可能我们甚至连各自的语言都不通……这些都不妨碍，只要我们研究的是相关领域中那些亟待解决的问题，我们就可以不期而遇，惺惺相惜。

如果用一句话概括我们的科研历程，就好比做了一场梦，梦中的我们徜徉在知识的海洋，利用智慧之光扬帆起航，在遥远的彼岸，与慕名的大师们一一相约，通过精神和灵感的自由交流，碰撞出绚丽的火花。

回想五年前那个懵懂的自己，多么茫然多么不知所措。面对前途的一片未知，我变得越来越不自信，怀疑自己为什么选择科研这条道路。当惊魂甫定，看到周围优秀的学长们发表了一篇又一篇优秀的论文，脸上洋溢着成功的喜悦，我便重新审视自己：为什么我不能像他们那样？当下定决心准备大干一场，我便开始学习读论文，总结论文。从一开始知其然，到后面知其所以然，甚至敢于质疑论文中的结论。而后，我知道了该读哪些论文，学会了有针对性地读学术大师的作品。当我被其中新颖的观点所吸引，被巧妙的方法所打动，被扎实的实验所折服，便感觉论文作者列表中那一串名字瞬间高大起来，像矗立在我跟前的一座高耸入云的大山，只可仰望。

在拜读了许多这样的作品后，我形成了自己的学术品味。我学会了发现问题，用知识武装自己，以高标准要求自己，书写自己的论文。虽然开始事与愿违，但经过多次修改和老师同学们的帮助，终于发表了自己的处女作。我既忐忑又喜悦，看着论文作者列表中自己的名字，一种骄傲之情油然而生。

接下来，我以演讲者的身份远赴西班牙参加所中的这个会议。可能初生牛犊不怕虎，在会上我主动结识了许多一同参会的人。那是我第一次面对面看到来自北美名校的资深教授对我言传身教，第一次面对这么多人展示自己的作品，我不但没有惧怕，反而很享受。而后我也听了其他人的报告，并在会议手册中找到当初在论文中看到过的很熟悉的名字。在他讲完他的论文回到场下，我便迫不及待地用不大熟练的英语跟他交流，告诉他我看过他的论文。他也很开心的同我握手，虽然之前我们素未谋面。那一瞬间，感觉整个会场都是我们的。

有了这次经历，我发现我爱上了科研，更加坚定了科研的决心。而后又有幸参加了一次国际会议，越来越发现曾经觉得遥不可及的大牛，瞬间亲近了许多。我终于悟出了一个道理：凡事都是相对的，譬如别人的高大；当你自己也变得高大起来，你便可以不用仰望也能看到他。

通过分享以上经历和心得，只想告诉那些即将踏入科研或还在摸索中的学弟学妹们，科研真的不可怕。反而当你发现科研之美，激发出好奇心，你会挖掘出有价值的问题，醉心于前沿的技术，结交很多有趣的灵魂。

科研就像与大师有约，其实我们并不孤单，因为我们同属一个学术圈。或许有一天，你也可以成为别人的大师。即使你处境不佳，埋头苦干却毫无头绪，你可以想象，也许在世界的某处，有人跟你思考同样的问题。蓦然回首，你会觉得：有人陪着，真好！

费新财
2014级博士研究生
研究方向：面向网络功能虚拟化的资源分配和系统优化
Email：god14fei@hust.edu.cn