通用在网计算系统架构及协议设计

System Architecture and Protocol Design for Generic In-Network Computing

姚柯翰/YAO Kehan，陆璐/LU Lu，徐世萍/XU Shiping
（中国移动通信有限公司研究院，中国 北京 100053）
(China Mobile Research Institute, Beijing 100053, China)

DOI：10.12142/ZTETJ.202304009
网络出版地址：http://kns.cnki.net/kcms/detail/34.1228.TN.20230718.1826.002.html
网络出版日期：2023-07-19
收稿日期：2023-05-20

摘要：生成式人工智能大模型训练以及海量的大数据实时处理对任务处理性能提出了更高要求。在网计算在数据完成尽力而为转发的基础上，进一步将计算相关的操作卸载至转发节点，可以有效提升系统计算效率。针对当前在网计算系统设计碎片化问题开展研究，提出了满足多种在网计算场景的通用系统架构，并进行相关协议设计。通用在网计算架构兼顾了系统实现的灵活性以及应用开发友好性，为进一步推进在网计算的规模化应用提供了新思路。

关键词：在网计算；算力网络；网络架构；网络协议

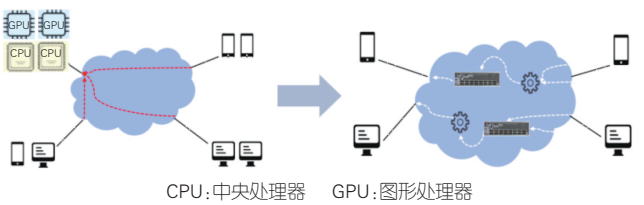
Abstract: The large model training of generative artificial intelligence and the real-time processing of big data need higher requirements for task processing performance. On the basis of completing the best-effort forwarding of data, in-network computing (INC) further offloads computing-related operations to network forwarding nodes, which can effectively improve the system computing efficiency. Aiming at solving the problem of design fragmentation of INC systems, a generic system architecture is proposed to meet different requirements of various INC scenarios, and related protocols are designed. The generic INC architecture takes into account the flexibility of system implementation and the friendliness of application development, and puts forward a new idea for further improving the scalability of INC.

Keywords: in-network computing; computing force network*; network architecture; network protocol

引用格式：姚柯翰, 陆璐, 徐世萍. 通用在网计算系统架构及协议设计 [J]. 中兴通讯技术, 2023, 29(4): 43-48. DOI: 10.12142/ZTETJ.202304009
Citation: YAO K H, LU L, XU S P. System architecture and protocol design for generic in-network computing [J]. ZTE technology journal, 2023, 29(4): 43-48. DOI: 10.12142/ZTETJ.202304009

1 在网计算的发展

在网计算（INC）指将部分计算任务卸载至网络，让数据在完成转发的同时实现数据处理，从而提升数据计算效率。如图1所示，传统的计算模式是数据由终端产生，全部送往集中的数据处理节点（如数据中心）来完成运算。



▲图1 端侧计算向在网计算演进

在网计算则可实现数据边转发边处理，大大降低数据处理节点的负载。

1.1 在网计算演进历程

在网计算的技术理念首次出现在1995年由美国国防部高级研究计划局（DARPA）提出的主动网络中[1]。在主动网络中，网络数据包不仅携带数据，还携带了数据的操作信息或程序。在网计算可以给主动网络中的转发节点使能主动计算属性，基于数据包中的程序指令对数据包进行操作，从而实现应用相关的功能，比如防火墙或网页代理等。但主动网络并未形成主流技术体系，主要原因在于其实现依赖于中央处理器（CPU）的处理能力。而网络设备的核心要务是进行

* 作者确认算力网络译为computing force network

线速数据包转发，这对转发芯片能力有严格要求。在当时，网络设备的转发芯片并不支持可编程能力，因此能够在网络设备做的计算也比较有限。

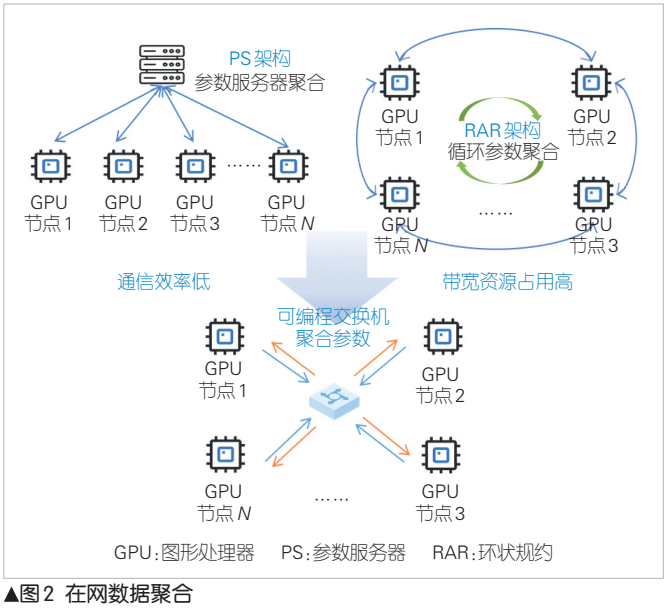
近年来，随着可编程网络硬件的发展以及软件定义网络架构的逐渐成熟，在网计算技术不断发展。斯坦福大学N. MCKEOWN教授团队在2014年发表的论文中首次提出了协议无关的包处理编程语言P4[2]，用于对网络数据平面的算法和处理逻辑进行自定义编程，从而实现更加灵活丰富的功能。目前，大量的学术研究聚焦在如何发挥可编程网络的灵活性和高性能，可编程网络成为在网计算发展的关键使能技术。

1.2 在网计算主要应用场景

在网计算[3]已广泛用于各种分布式系统。在网计算将应用相关的功能卸载至网络节点，实现分布式应用处理性能的有效提升以及网络带宽资源的合理优化。本节中，针对目前在网计算在应用加速方面的主要研究，我们进行了总结，内容主要包括在网数据聚合、在网数据推理、在网缓存以及在网共识。

1）在网数据聚合

分布式机器学习模型训练可以基于在网数据聚合进行加速。目前，主流分布式机器学习系统架构为环状规约（RAR）和参数服务器（PS），如图2所示。RAR架构对网络带宽资源占用高，完成一次完整的分布式机器学习模型训练任务需要传递约模型总参数量2倍的通信量，极易引起网络拥塞；而PS架构则由于集中式服务器节点的吞吐瓶颈问题，面临较大的聚合延迟，限制了分布式机器学习模型训练的效率，扩展性较差。



▲图2 在网数据聚合

在网计算可由网络交换节点实现参数聚合功能，既克服了聚合节点的吞吐瓶颈问题，也避免了RAR架构高额的带宽资源占用，实现了训练性能和带宽资源的有效平衡，极大地提升系统的扩展性。

2）在网数据推理

在网数据推理可实现网络流量分类和控制。业界相关的研究包括在网络转发设备实现决策树、支持向量机（SVM）、朴素贝叶斯等各种分类算法[8]，以及通过神经网络实现联邦学习，支撑网络设备在网络路径上就近返回处理结果，从而提升集群计算能力。

与基于分析服务器的推理方式相比，中间层交换机推理提前终止了终端设备发往分析服务器的原始数据流量，节省了更高层核心网络的带宽，同时利用网络设备的高速处理来减少推理时间，加速数据实时分析和控制指令响应。

3）在网数据缓存

高性能的分布式数据存储和索引需要依赖于高性能的Key-Value存储。在社交网络等高并发应用中，慢速的Key-Value存储可能导致较大的系统尾延迟，进而影响系统性能。通过设计层次化的缓存系统，在边缘网络节点部署Key-Value缓存服务，在网络设备中完成高频内容缓存以及快速查询和响应[4]。在网数据缓存机制是高性能存储系统以及高性能流式处理系统加速的关键。

4）在网数据共识

在分布式系统中，可以通过共识协议来实现对某个数据值或操作序列的一致性，比如锁定管理系统、组播通信、一致性协调。卸载共识功能的部分或全部功能卸载到网元，可以减少协调延迟，提升分布式系统的可用性。文献[5]利用可编程交换机实现了一致性算法的网内卸载，实验也证明了在网数据共识对分布式系统性能的优化。

2 通用在网计算架构

在网计算对系统的性能优化已被广泛地论证，但是在网计算在架构设计层面还面临碎片化的问题。目前，在网计算主要根据应用场景进行定制化设计，满足相关应用的个性化需求，但是这样的设计方法扩展性较差，不利于在网计算的规模化应用。

同时，对应用开发者而言，想要基于网络设备的计算能力进行系统设计，既需要了解上层系统的逻辑架构，还要了解底层物理网络的属性，包括网络设备的编程能力以及网络的规划能力。这也进一步提升了在网计算系统的设计门槛，阻碍了在网计算的应用。

为解决上述问题，我们在架构设计层面，从在网计算的通用性和应用设计的友好性出发，设计了S（任务调度层）、C（在网计算控制层）、I（基础设施层）3层通用在网计算架构，系统架构如图3所示。

2.1 基础设施层

基础设施层包含执行计算任务的端侧主机节点以及在网计算节点。由于异构网络设备在硬件架构方面存在较大差异，这些在网计算节点能够提供的计算能力也不同。这意味着同一个在网计算原语在异构网络设备内部可以有不同的实现方式。针对不同场景下在网计算原语，很多研究进行了分类和总结[7]。本文在这些研究的基础上，进一步设计了面向异构在网计算节点的统一北向接口，在网计算节点通过北向接口上报在网计算原语信息，对外提供统一的服务接口。这使得在网计算更具通用性。

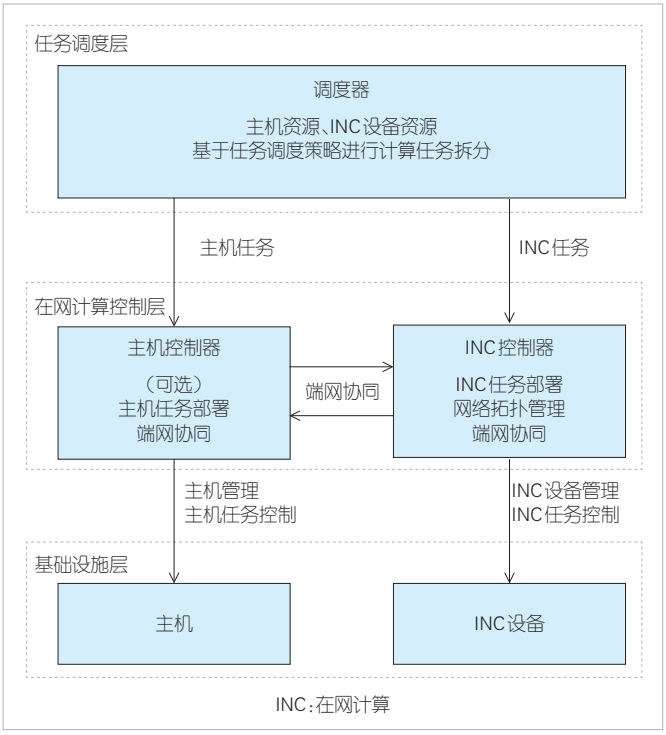
在端主机侧，应用程序也需要进行相应适配。网络无法保证大部分在网计算应用独立完成计算任务，因此需要通过端网协同机制来完成。端主机侧应用程序需要感知在网计算任务，这样可以保证计算的完整性，同时可以提高网络传输的可靠性。端主机侧通过北向接口连接端侧任务控制器，实现端侧计算任务分配。

2.2 在网计算控制层

在网计算控制层是实现端网协同在网计算的关键。控制层包含主机控制器和在网计算控制器。主机控制器根据应用场景按需部署，主要负责主机任务部署以及端到端的可靠性保障。在网计算控制器主要负责通用的网络管理以及在网计算任务部署和控制等。在网计算控制器通过南向接口实现网络管理功能，包含网络设备管理以及网络拓扑管理。网络设备管理包括网络设备状态、网络设备负载、网络设备计算能力、网络设备计算资源管理等，其中网络设备的计算能力是在网计算控制器和传统网络控制器最大的不同。网络设备的计算能力通常通过在网计算原语、在网计算数据结构来表示。网络拓扑管理包括网络拓扑更新、链路状态监控等。主机控制器和在网计算控制器共同实现端网协同控制，并根据网络资源状态综合选路，为在网计算和转发选择一条最优路径。

2.3 任务调度层

任务调度层实现在网计算系统和应用的对接。应用将任务需求提交给统一的任务调度器。任务调度器通过南向接口对接端侧控制器以及在网计算控制器，收集端侧和网侧的当前计算资源状态。任务调度器结合应用任务请求及计算资源状态，基于特定的算法进行计算图设计，生成计算节点之间的逻辑依赖关系，进而产生具体的任务分配策略。恰当的任务调度策略可以实现合理的在网计算资源分配，从而在保证任务处理性能的同时优化网络管理。



▲图3 通用在网计算架构

3 基于SRv6协议的通用在网计算的实现

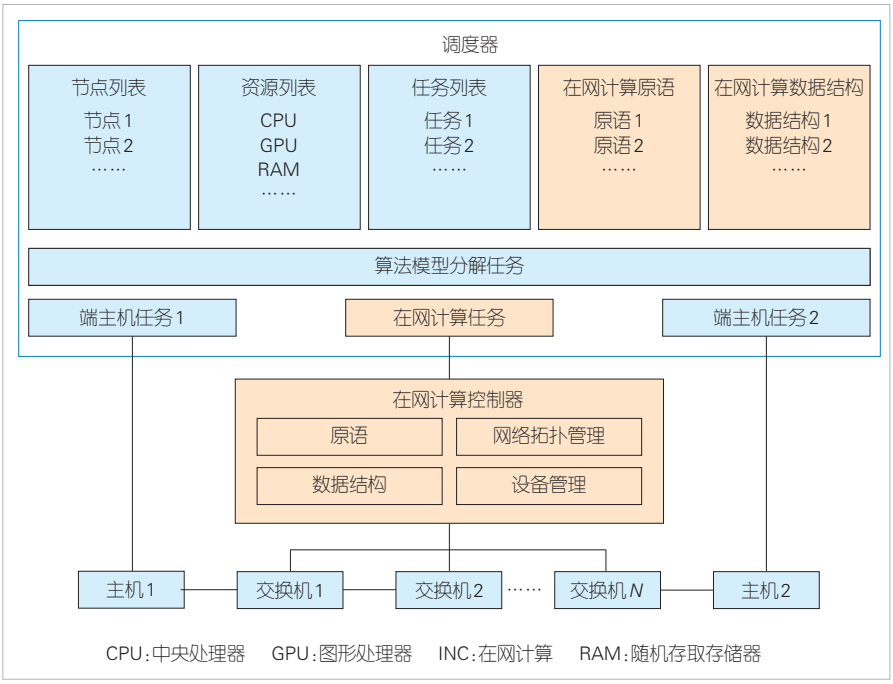
通用在网计算框架不约束数据面转发协议。本节中，我们以数据面运行基于IPv6的段路由（SRv6）协议为例，说明通用在网计算框架的工作机制。

3.1 基于SRv6协议的通用在网计算实现流程

基于SRv6的集中式通用在网计算架构如图4所示。该架构未部署主机控制器，任务调度器直接对接服务器，在网计算控制器负责承接在网计算任务，数据面运行SRv6协议，SRv6包头在接入交换机上封装。

1）管理员配置在网计算控制器，将在网计算原语和在网计算数据结构模型配置生成模板库。

2）网络设备初始化时，上报自身在网计算能力，实现标准的在网计算原语和在网计算数据结构。设备自身的实现可能有计算精度、数据范围等差异。网络设备平稳运行后，周期上报自身负载和在网计算能力变化。



▲图4 集中式通用在网计算框架

3）调度器根据任务分解策略将计算任务拆解为主机任务和在网计算任务，拆解时需要考虑主机和在网计算的能力和资源，然后告知在网计算控制器该任务具体要执行哪些在网计算原语。

4）当主机节点有数据要执行在网计算时，首先向在网计算控制器发送请求，说明在网计算任务ID、源节点、目的节点、要执行的在网计算原语，然后由UniqueID对业务分配标识。

5）在网计算控制器根据网络拓扑、在网计算能力、网络负载等情况进行综合选路，将选路结果反馈给主机侧，并在网络设备上做在网计算资源预留。

6）源服务器发送数据包，接入交换机封装SRv6头，各网络节点根据协议包头在网计算指示信息执行在网计算。

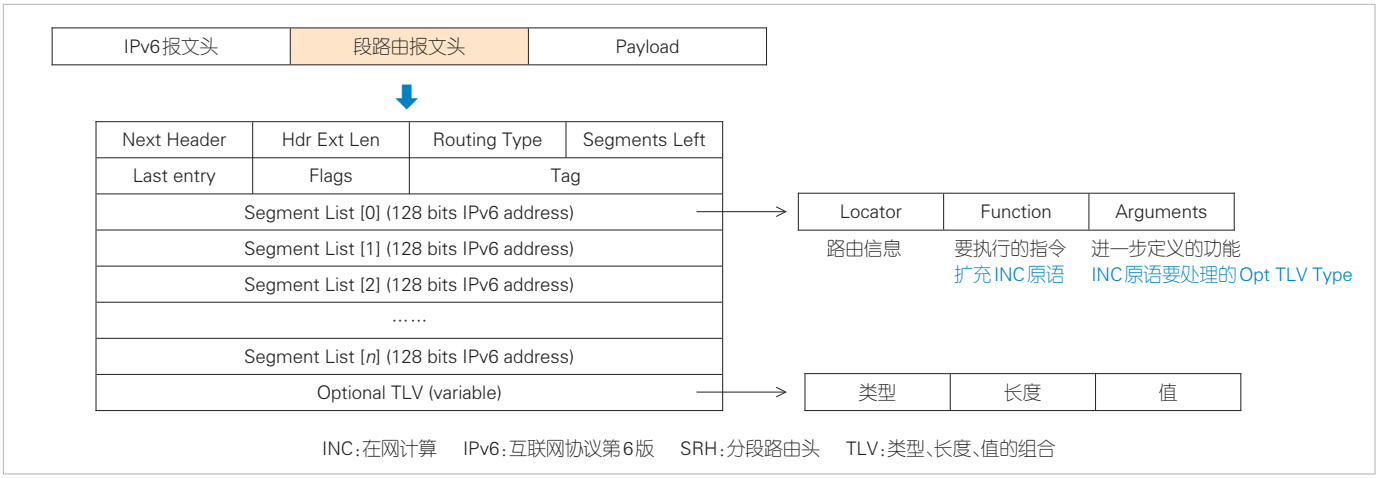
3.2 基于SRv6协议报文头扩展

国际互联网工程任务组（IETF）定义了SRv6段标识符（SID）中的Function字段的通用转发行为[9]。如图5所示，我们新增了INC Segment定义在网计算行为。其中，Locator与其他segment保持一致，表示路由位置信息；Function字段表示具体要做的在网计算原语；Arguments指示对应Optional TLV（类型、长度、值的组合）的类型，可以由应用自行定义；Optional TLV可以用来携带在网计算原语所需要的信息，例如需要处理的数据偏移、计算需要的参数等。

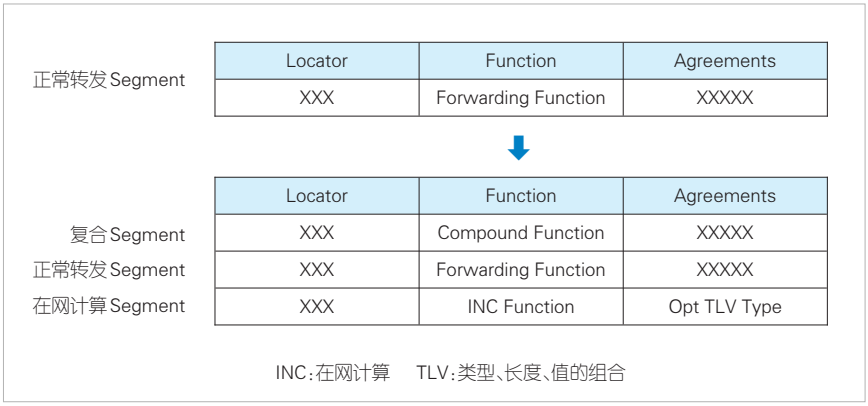
节点在执行在网计算时，不能影响正常转发，因此需要增加复合Segment，使在网计算交换机同时支持正常转发和在网计算能力，具体如图6所示。复合Segment用来指示后续两个连续的Segment都需要在本节点处理，第1个为正常转发Segment（Forwarding Segment），第2个为在网计算Segment。

3.3 交换机在网计算原语能力传播

交换机的在网计算原语可以由控制器统一管理，并基于路由协议进行信息扩展，再传递到相应的在网计算执行节点。例如，自治域内源路由使用内部网关协议（IGP）来传



▲图5 SRH协议支持在网计算协议扩充



▲图6 复合Segment使交换机同时支持转发和在网计算

播SID和对应的Function等信息。以中间系统–中间系统协议（ISIS）为例，传递在网计算可以使用两种方式。

1）扩展ISIS SRv6协议中Sub–TLV字段

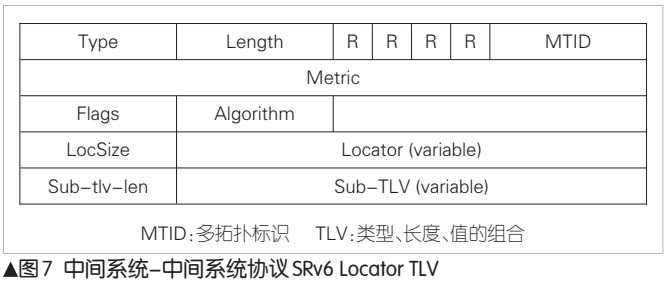
SRv6 Locator TLV用于发布SRv6 Locator以及该Locator相关的Endpoint SID。Locator具有定位功能，一般要在段路由域内唯一标识，Endpoint SID用于标识网络中的某个目的节点。

ISIS的SRv6 Locator TLV格式如图7所示。其中，Locator（variable）表示发布的SRv6 Locator，长度可变；Sub–TLVs（variable）可以根据类型不同，携带不同信息，长度可变。

因此，可以有方案1：扩展Sub–TLVs，新增一种描述在网计算原语信息的报文结构。其中，Type字段表示在网计算原语类型，Length字段表示Value长度。Value为交换机支持的在网计算原语补充信息，如果Value等于0则无补充信息。

2）扩展一种新类型的SRv6 Sub–TLV

ISIS本身有多种Sub–TLV协议报文格式，分别用来传递不同信息。其中，SRv6 Capabilities Sub–TLV用于通告SRv6



▲图7 中间系统–中间系统协议SRv6 Locator TLV

▲图8 中间系统–中间系统协议SRv6 Capabilities Sub–TLV报文格式

能力。SRv6 Capabilities Sub–TLV的格式如图8所示。

因此，可以有方案2：新定义一种用于传递在网计算原语能力的Sub–TLV类型SRv6 INC Sub–TLV。其中，Type字段为网计算原语能力，Optional Sub–sub–TLVs为交换机支持的在网计算原语。

4 在网计算发展的挑战

1）网络设备硬件资源受限

目前，可编程网络设备尚不能支持大规模或泛在的在网计算，主要原因在于可编程硬件片上资源受限。例如Tofino交换芯片，其片上的静态随机存取存储器（SRAM）、三态内容可寻址存储器（TCAM）存储空间约数十兆字节[6]，只能存储少量的带状态数据。另外，分布式机器学习及高性能计算需要在网计算具备高精度浮点数处理能力，但目前可编程交换芯片只能支持整型数据处理。

2）跨设备资源管理和任务协同

分布式系统中高并发、大数据量的处理任务对在网计算资源提出挑战，这导致在网计算的加速性能有限，因此需要设计跨交换资源的管理机制以及任务跨设备的分解调度机制，以实现在网计算的规模扩展。交换机资源如何池化以及如何利用控制器进行资源和任务协同，还有待进一步研究。

3）计算可靠性挑战

在网计算在转发的同时要实现对数据的处理，这给传统的可靠性机制带来了挑战。网络尽力而为的转发机制可能会造成在网计算结果错误。例如，在网数据在聚合过程中会丢弃已聚合的数据包，只保留最后的聚合结果，传统的可靠性机制会将这一行为判断为丢包。再如，在网计算设备可能由于资源不足或其他原因导致无法完成在网计算任务，可靠性机制需要能够灵活判断和计算。不同的场景对于可靠性的要求不同，这给在网计算的发展带来了很大的挑战。

4）安全性挑战

在网计算需要在网络转发节点终结一部分数据流并进行数据操作，这在一定程度上为网络引入了安全风险。目前，在网计算的主要应用和设计聚焦在安全可控的网络场景中。未来，面向通用泛在的在网计算应用场景，如何提升系统安全性，降低数据计算结果被篡改的风险成为挑战。

5 结束语

本文分析了在网计算在多种应用场景下的共性能力，并

针对在网计算系统碎片化问题进行架构设计，提出了S、C、I的3层通用在网计算系统架构。异构在网计算节点通过统一的北向接口向在网计算控制器上报计算能力，为不同应用场景提供共享的网络基础设施。同时架构简化了在网计算应用开发，应用只需要向任务调度器提出需求，再由任务调度器综合决策，有效避免了应用开发者对底层物理网络复杂逻辑的理解，降低了应用开发门槛。本文以SRv6数据面协议为例，设计了通用在网计算的实现机制，同时针对在网计算的通用性和扩展性的提升提出了一些需要关注的问题。

参考文献

[1] TENNENHOUSE D L, SMITH J M, SINCOSKIE W D, et al. A survey of active network research [J]. IEEE communications magazine, 1997, 35(1): 80–86. DOI: 10.1109/35.568214

[2] BOSSHART P, DALY D, GIBB G, et al. P4: programming protocol–independent packet processors [J]. ACM SIGCOMM computer communication review, 2014, 44(3): 87–95. DOI: 10.1145/2656877.2656890

[3] LAO C L, LE Y F, MAHAJAN K, et al. ATP: in–network aggregation for multi–tenant learning [C]//18th USENIX Symposium on Networked Systems Design and Implementation (NSDI 21). NSDI, 2021: 741–761

[4] SEEMAKHUPT K, LIU S H, SENEVIRATHNE Y, et al. PMNet: in–network data persistence [C]//Proceedings of 2021 ACM/IEEE 48th Annual International Symposium on Computer Architecture (ISCA). IEEE, 2021: 804–817. DOI: 10.1109/ISCA52012.2021.00068

[5] JIN X, LI X Z, ZHANG H Y, et al. Netchain: scale–free sub–RTT coordination [C]//Proceedings of the 15th USENIX Conference on Networked Systems Design and Implementation. ACM, 2018: 35–49. DOI: 10.5555/3307441.3307445

[6] CHOLE S, FINGERHUT A, MA S, et al. dRMT: disaggregated programmable switching [C]//Proceedings of the Conference of the ACM Special Interest Group on Data Communication. ACM, 2017: 1–14. DOI: 10.1145/3098822.3098823

[7] ZHAO B, WU W, XU W. NetRPC: enabling In–network computation in remote procedure calls [EB/OL]. [2023–05–20]. https://arxiv. org/abs/2212.08362

[8] ZHENG C, XIONG Z, BUI T T, et al. IIsy: practical in–network classification [EB/OL]. [2023–05–27]. https://arxiv.org/abs/2205.08243

[9] FILSFILS C, CAMARILLO P, LEDDY J, et al. Segment routing over IPv6 (SRv6) network programming: RFC 8986 [S]. 2021

作　者　简　介

姚柯翰，中国移动通信有限公司研究院研究员；研究方向包括未来网络架构、在网计算、可编程网络等。

陆璐，中国移动通信有限公司研究院基础网络技术研究所副所长、中国通信标准化协会TC5核心网组组长；长期从事移动核心网策略、演进、标准和技术研究工作，主要涉及未来网络架构、智能管道、边缘计算、算力网络等领域。

徐世萍，中国移动通信有限公司研究院研究员；研究方向包括未来IP网络、算力网络、在网计算等。