在网计算（NACA）技术白皮书解读

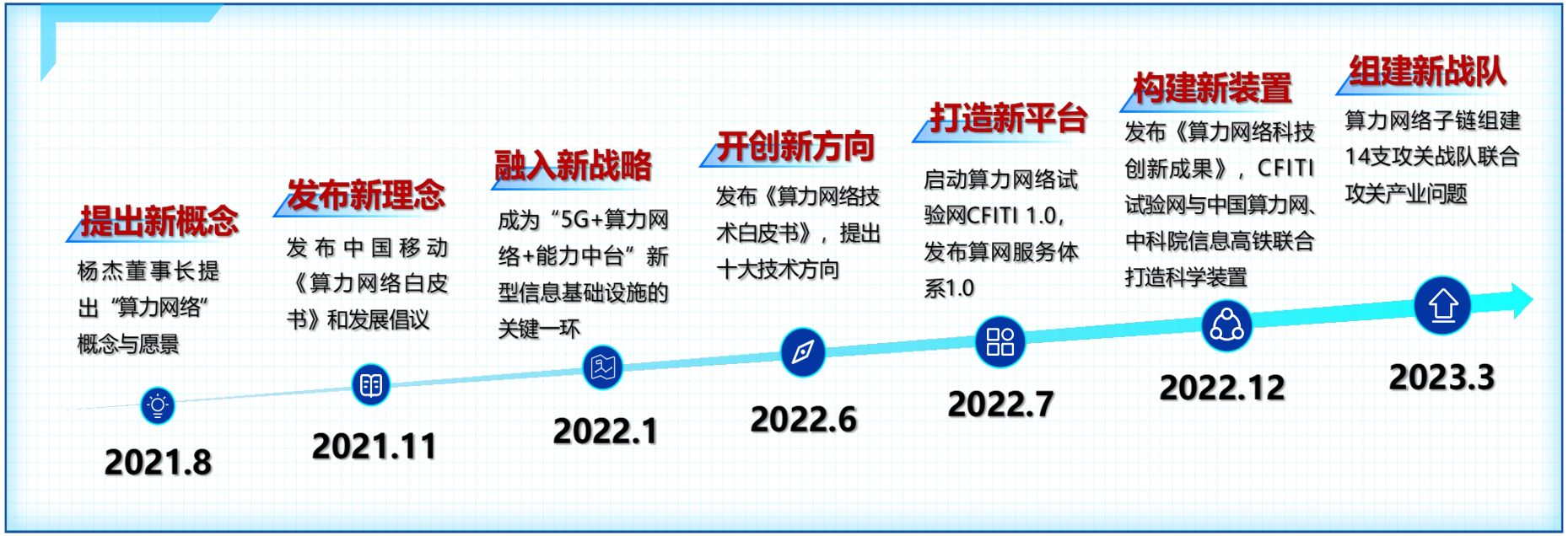
中国移动 陆璐

2023年8月

中国移动提出“算力网络”新理念

中国移动充分发把握算力时代发展脉络，以网强算提出“算力网络”全新理念，两年多来持续开拓创新，全力推进算力网络发展，形成一系列创新成果，在业界取得了广泛共识，引起了巨大反响

> 算力网络是以算为中心、网为根基，网、云、数、智、安、边、端、链（ABCDNETS）等深度融合、提供一体化服务的新型信息基础设施。
>
> ——中国移动《算力网络白皮书》

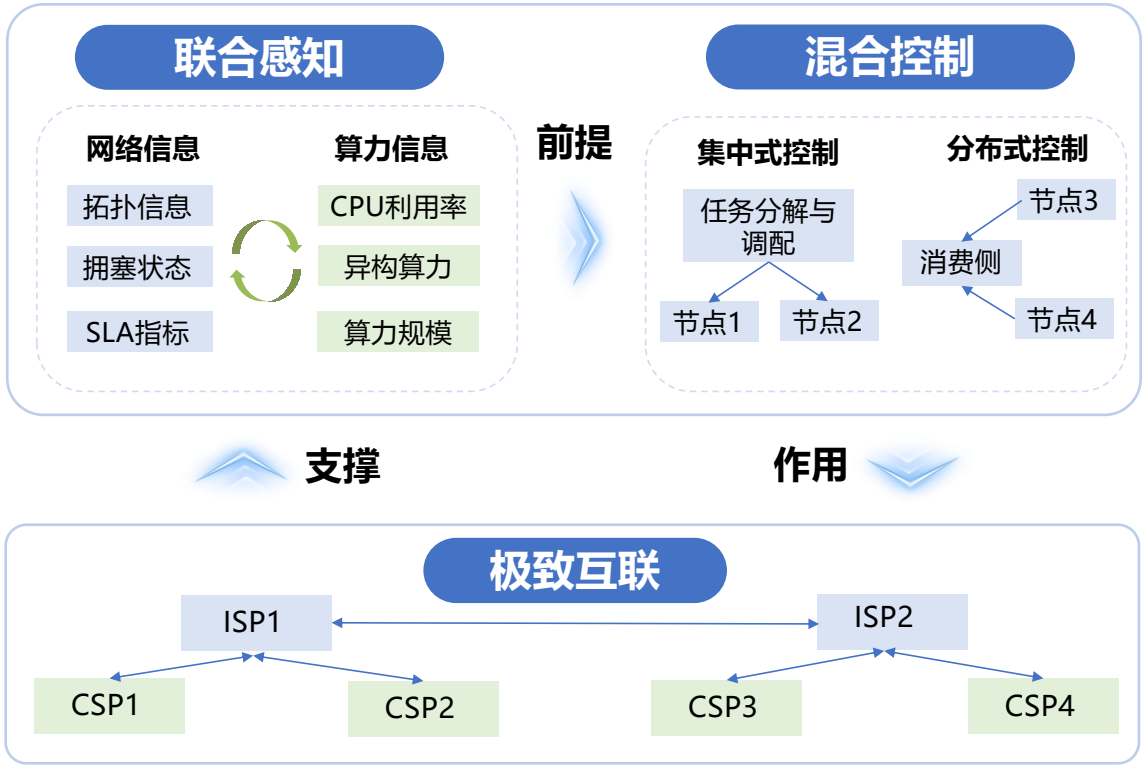


算力网络的发展经过三个阶段的发展，逐渐深化

	起步阶段：泛在协同	发展阶段：融合统一	跨越阶段：一体内生
运营服务	**一站**服务：一站开通算网服务 **协同**运营：云网运营双入口拉通	**融合**服务：产品融合、确定性服务 **统一**运营：统一入口、统一平台	**一体**服务：多层次智简无感服务 **模式**创新：多方算力可信交易
编排管理	协同编排	智能编排	智慧内生
基础设施	网随算动	算网融合	算网一体

算网一体通过“联合感知”“混合控制”“极致互联”构建面向智能化时代的数字基础设施

架构



- *ISP：网络服务提供者；　　CSP：算力服务提供者*

关键技术

算力路由

创新互联网架构协议，基于算网资源联合感知实现动态融合决策选路

在网计算

网络内生算力，基于集中式控制，实现计算任务跨云、网、边、端分布式协同

广域RDMA

突破RDMA长距传输瓶颈，实现广域高性能互联

算网数字孪生

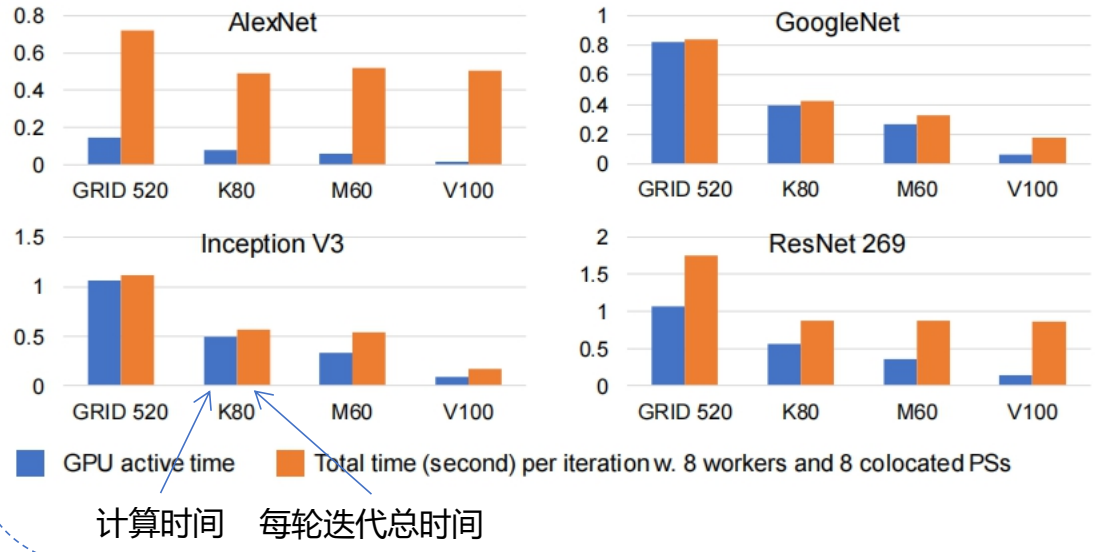
基于网络大模型的算网数字孪生构建可视、可管、可控的算网基础设施

分布式系统面临通信开销瓶颈问题

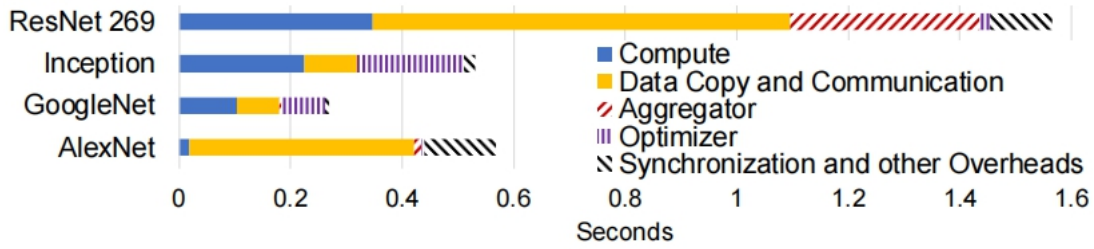
在网计算主要面向分布式应用，随着分布式系统规模不断扩大，计算节点间的通信量激增，通信模式更加复杂，通信开销已成为AI、大数据、HPC等分布式应用的性能瓶颈，严重制约系统规模扩展

分布式应用场景

模型训练时长分布[1]



分布式训练管道中模型训练时间分解[1]



不同模型在进行分布式训练时，各阶段的时间分布不同
ResNet 269的通信时间最长，且还有较长的Aggregator，
AlexNet还具有较长的Synchronization，等等

- 采用8个workers和8个PSs的网络训练AlexNet模型，网络通信时间占比可高达**80%**以上
- 面向AI场景的网络优化需要更细粒度的通信算子优化方案

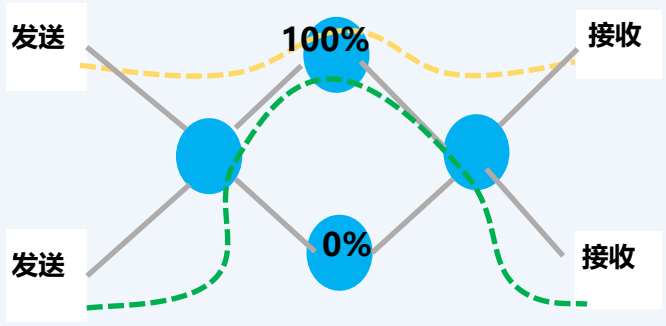
需要尽可能压缩通信的时延占比，同时结合不同类型的通信过程优化分布式系统通信性能

[1] Parameter Hub: a Rack-Scale Parameter Server for Distributed Deep Neural Network Training，https://dl.acm.org/doi/10.1145/3267809.3267840

问题主要来源

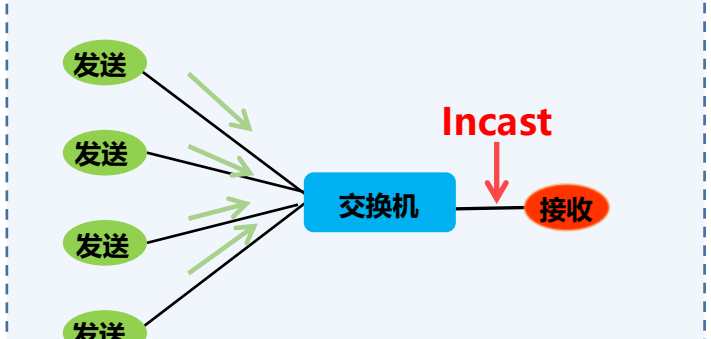
衡量分布式应用通信性能的重要指标是任务完成时间，负载均衡策略、计算节点多打一现象以及物理与逻辑通信模式不匹配等因素引发通信瓶颈问题，导致任务完成时间过长

网络负载严重不均衡



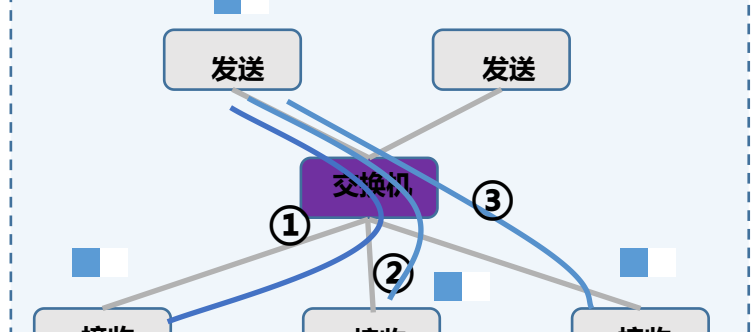
网络侧ECMP实现AI训练流量调度，AI训练以**巨型流**为主，HPC业务以**高并发小流**为主，传统网络调度方式难以满足AI、HPC等计算密集型业务场景流量调优目标。

流量需求不对等



大数据流式计算多对一的数据处理模式：训练最后一级交换机和接收方之间**Incast拥塞**，造成计算流长尾时延，计算任务完成时间过长。

通信模式不匹配



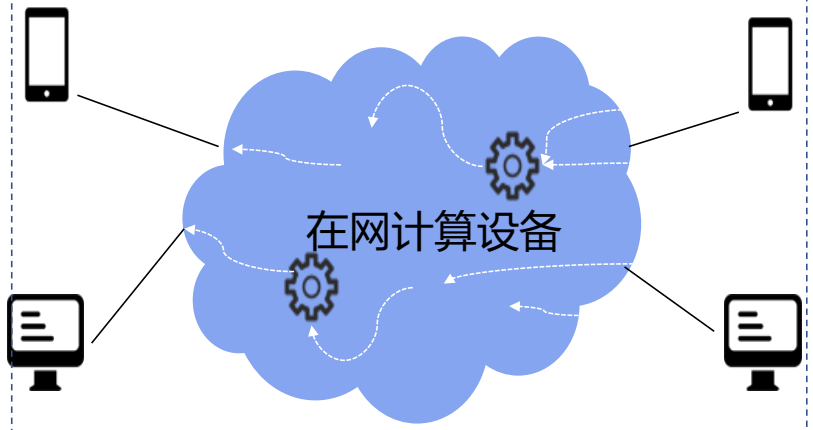
多对多逻辑通信需求与点对点物理通信实现：**进程间**MPI接口设计包含多对一、一对多及多对多的通信需求，**计算节点间**目前以单播实现MPI接口，物理网络存在大量冗余信息

通过网络与分布式应用各通信阶段紧耦合的设计方式优化分布式处理是重要发展方向

在网计算有望攻克分布式系统通信瓶颈问题

在网计算突破现有计算模式，重构应用处理逻辑，为系统算效提升带来质变

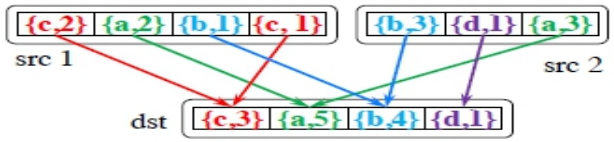
处理模式



在网计算将计算卸载至网络，实现**数据随转随算**，实现系统加速，提升算网资源利用率。

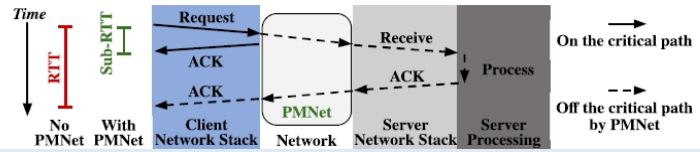
主要优势

流量压缩



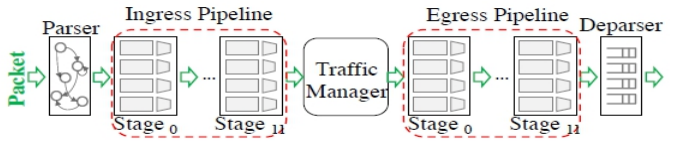
在网聚合，数据**消冗与求和**

缩短传输路径



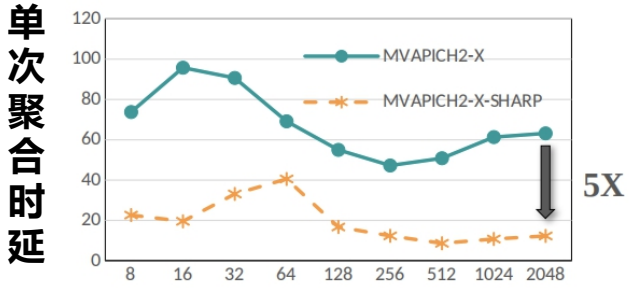
网内处理，实现**Sub-RTT**通信

线速处理

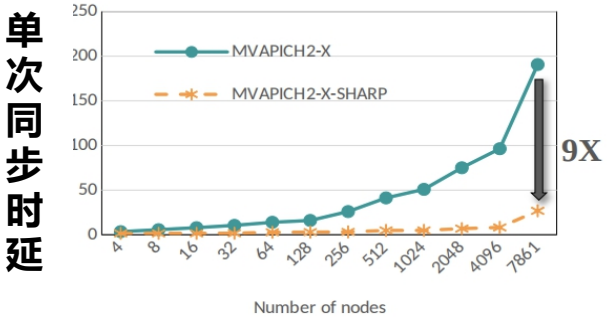


交换机**Tbps**处理能力

性能跃升



与传统软件实现**聚合操作**相比，IB SHARP方案性能提升近**5倍**



与传统软件实现**消息同步**相比，IB SHARP方案性能提升近**9倍**

在网计算发展现状

产业已逐步布局在网计算的研究和实践，中国移动积极推进试验验证和标准制定

产业与学术进展

在网聚合	在网组播	在网聚合		
SHARP	**IB based MPI-Bcast**	**ATP**	**Trio**	**NetReduce**
COMHPC 16	*IPDPS 04*	*NSDI 21*	*SIGCOMM 22*	*ASPLOS 23*
基于**IB智能网卡和IB交换机**，基于IB传输层QP，实现**参数聚合**	基于**IB交换机**的硬件组播能力，实现**MPI广播**	基于**多级可编程交换机**参与**参数聚合**，基于IP协议设计ATP 报文头	基于**NP交换机**实现**参数聚合**，基于UDP协议设计Trio-ML报文头	基于**FPGA和商用交换机**实现传输层透明的**参数聚合**

中国移动CFITI试验网创新验证

架构	AlexNet	VGG19	VGG16	VGG11	ResNet152	ResNet101	ResNet50
BytePS	330	110	120	130	110	155	250
Horovod	500	130	150	210	100	148	235
在网计算	540	155	175	215	115	165	265

测试基准：GPU型号：2080　　单位：图片数/秒

训练提速： 相比参数服务器架构BytePS，通信密集型任务最高可**提升60%**以上

带宽优化： 相比RAR架构Horovod，降低智算集群网络带宽占用**约1倍**左右

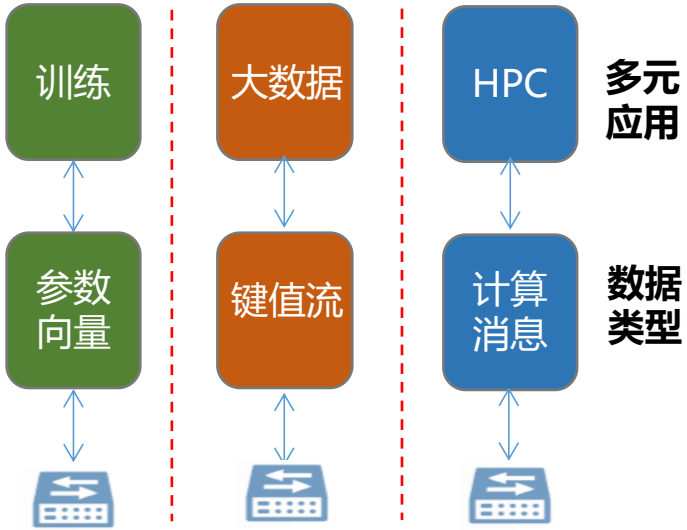
标准推进：在CCSA TC3 WG3牵头完成业界首个在网计算行标立项

在网计算方向已有一定共识，但仍面临多方面发展挑战，需要产学研协同攻关

在网计算发展挑战

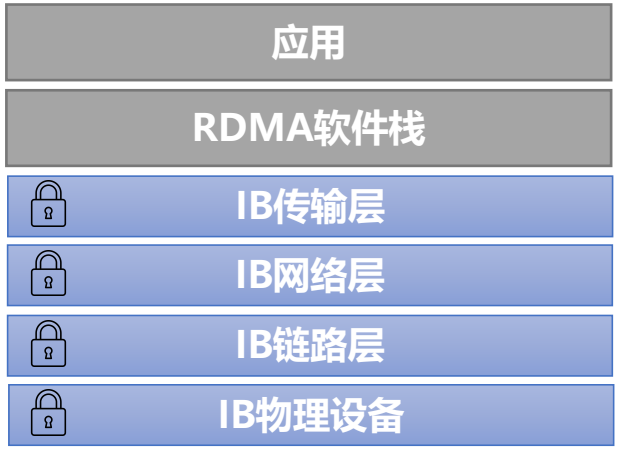
在网计算发展面临应用场景竖井式、协议实现封闭化、以及编程范式不友好等挑战

竖井式



现有在网计算方案面向单一场景竖井式设计，在**协议设计**和**硬件实现**等方面缺乏通用性

封闭化

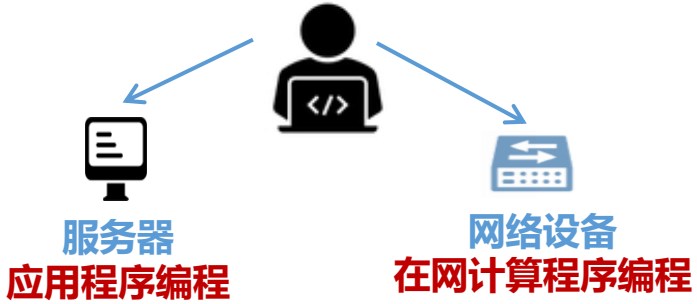


InfiniBand

基于IB协议栈及专用硬件的在网计算，性能优势明显，但**成本高**，协议栈**封闭不兼容**

编程不友好

开发模式有差异：应用程序开发模式和网络开发模式不匹配，开发者学习门槛高



运行框架不支持：Tensorflow、Spark等分布式开发框架不支持在网计算能力调用

需要从产业、生态等方面破除技术壁垒，构建统一通用的在网计算能力

在网计算NACA
Network Assisted Computing Acceleration

逻辑物理统一

通信原语统一

NACA以**提升在网计算通用性**为目标，

重构应用处理模式，构建全新的在网计算通信库，

围绕拓扑映射、编程范式、计算实现、资源管理

形成”**四个统一**”，实现**网络辅助计算加速**，

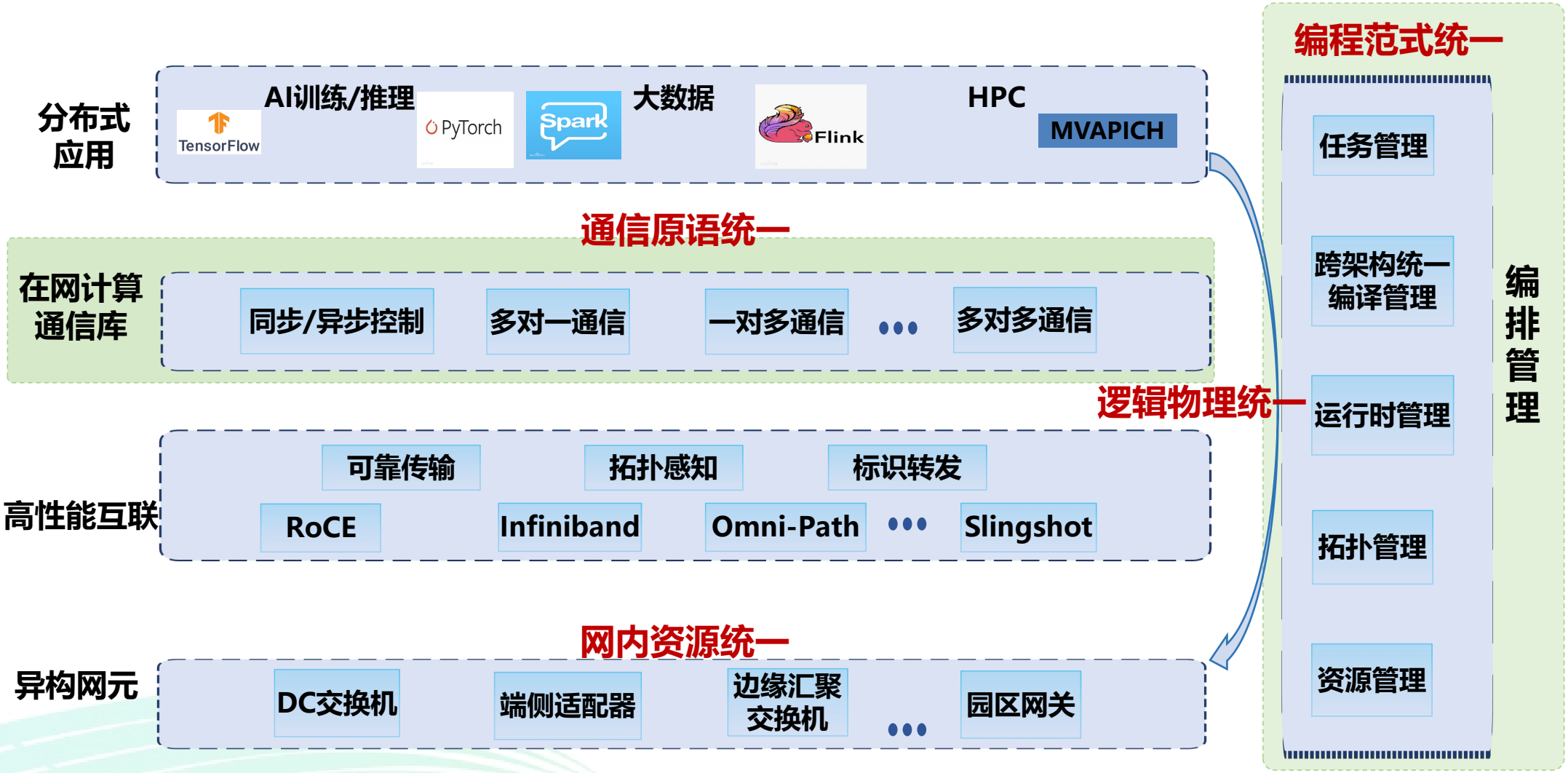
提升分布式系统算能算效

编程范式统一

网内资源统一

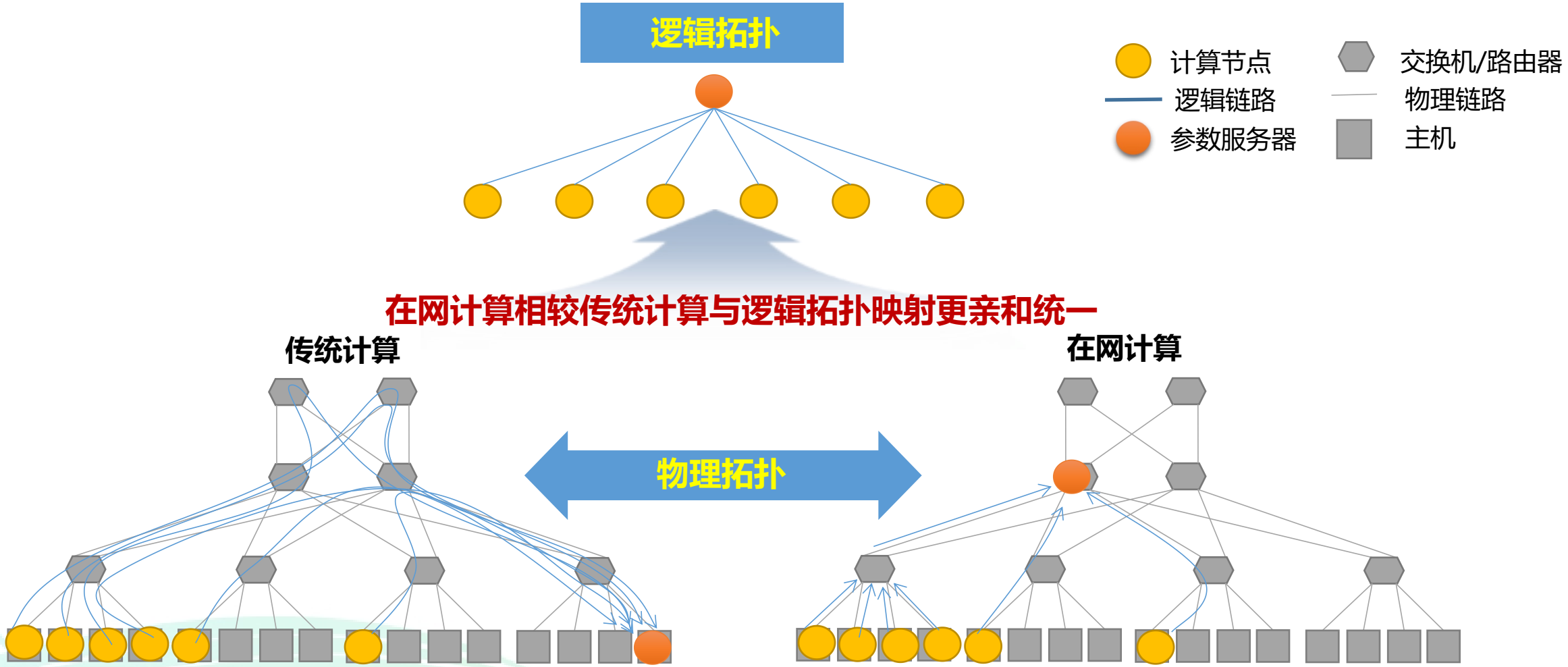
在网计算NACA技术架构

NACA架构核心在“一横一纵”，横向在网计算通信库承上启下，以异构网内算力实现统一在网计算服务，纵向编排管理全栈贯通，优化应用开发模式、协同端网任务部署、统筹网内资源管理



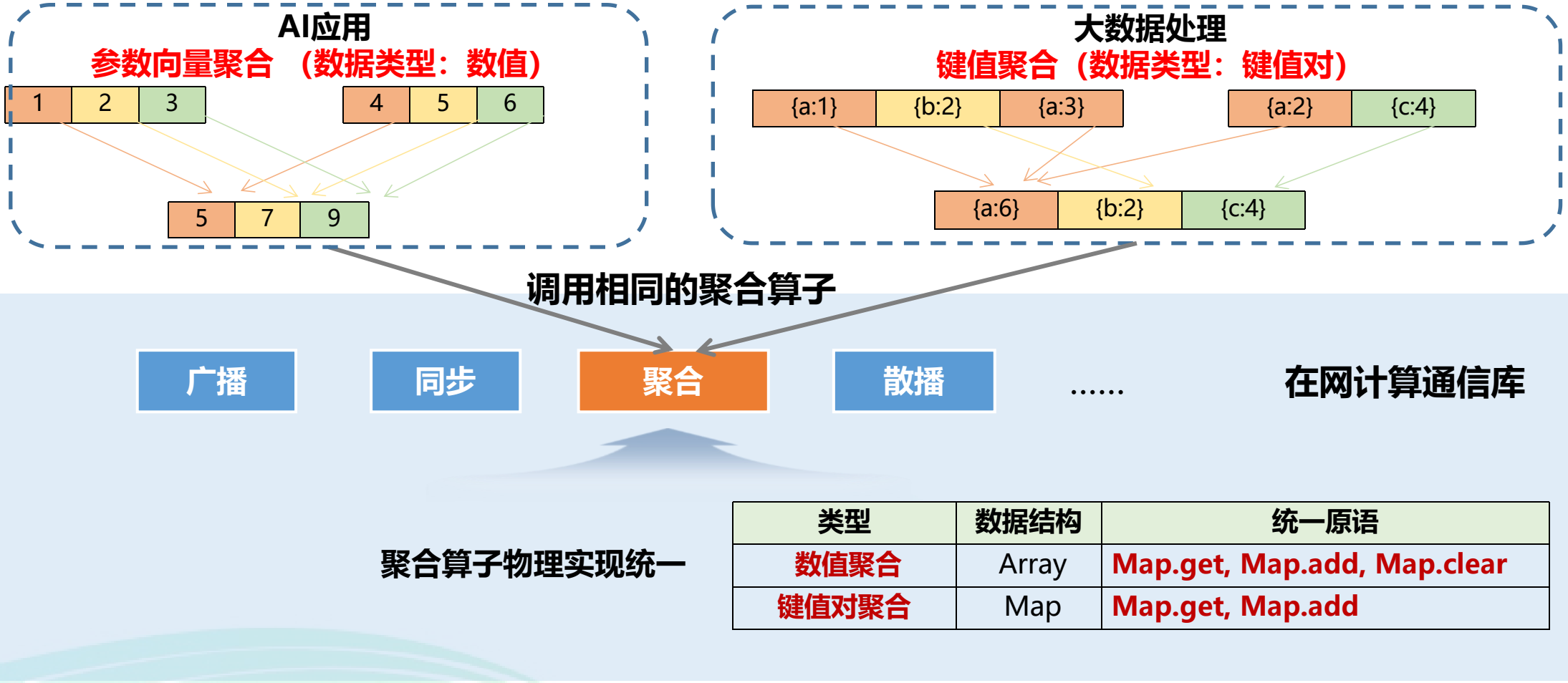
核心特征1：逻辑物理统一

NACA在网计算物理实现比传统计算实现方式更加亲和业务逻辑拓扑，网络与业务紧密耦合



核心特征2：通信算子统一

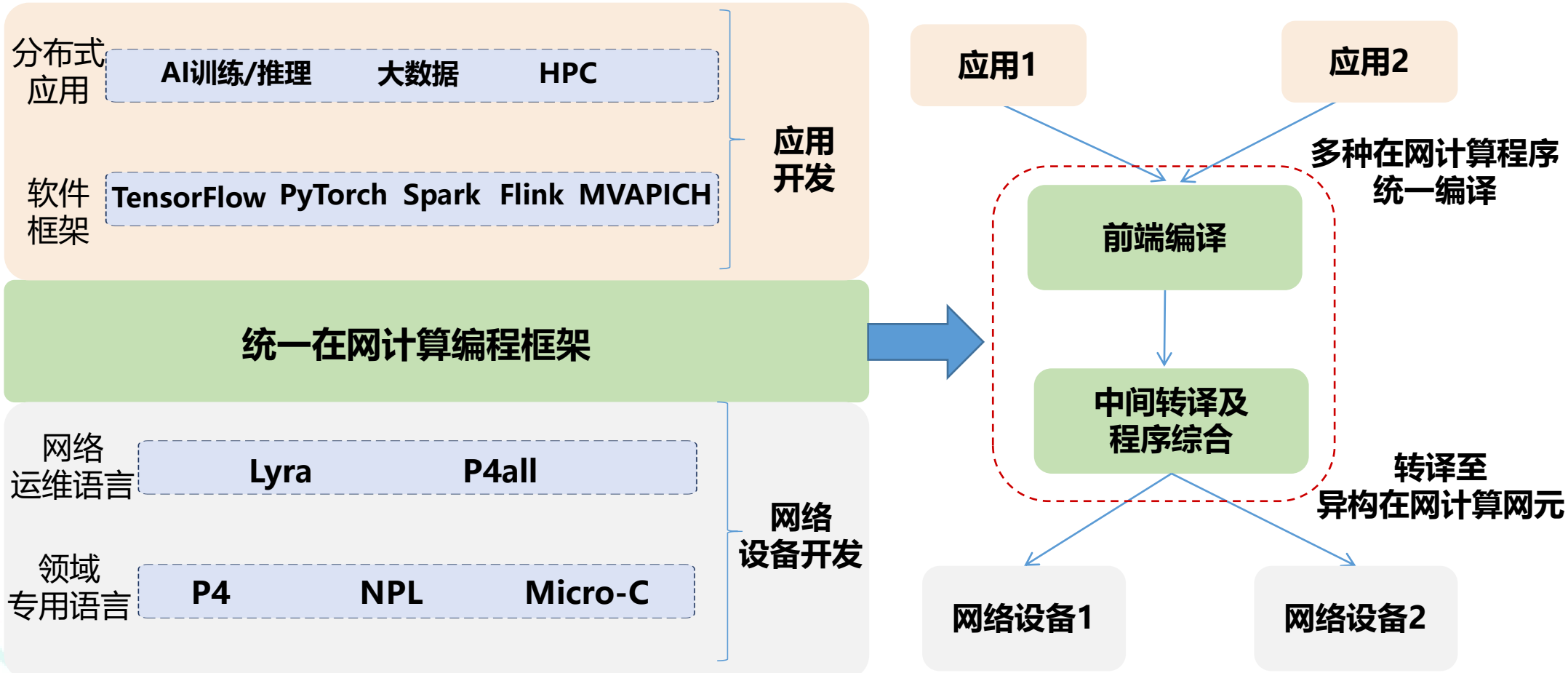
NACA面向差异化应用定义统一在网计算通信库，以统一的设备原语实现通信库，提升在网计算的通用性



类型	数据结构	统一原语
数值聚合	Array	Map.get, Map.add, Map.clear
键值对聚合	Map	Map.get, Map.add

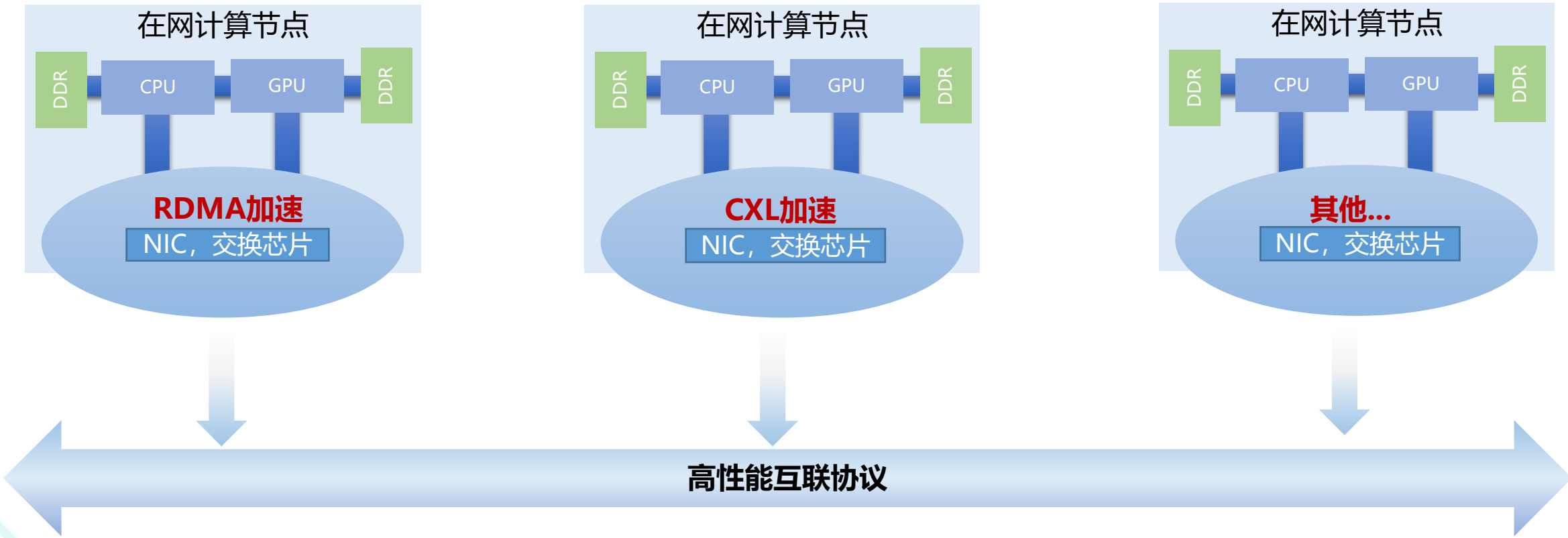
核心特征3：编程范式统一

NACA面向不同应用程序设计，提供统一编程语言及通用开发模式，简化异构设备开发入口



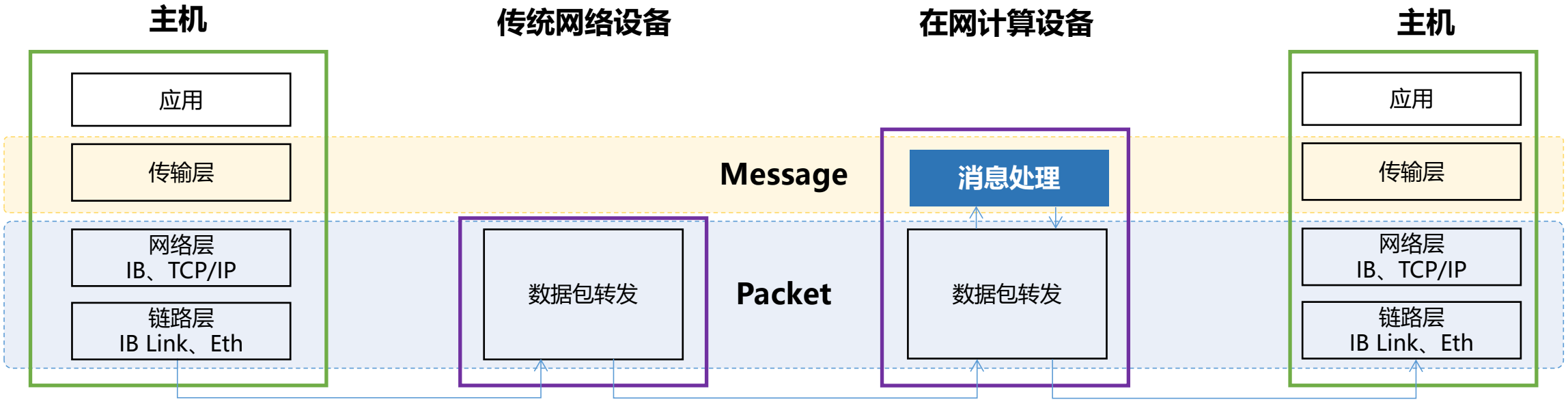
核心特征4：网内资源统一

NACA基于RDMA、CXL等高性能互联协议构建统一在网计算资源池，优化网络资源管理，提升网内资源利用率



关键技术1：计算语义映射

消息是分布式应用进程间通信的传递内容。传统网络设备基于数据包转发，在网计算设备基于消息处理，因此需要把消息和数据包的语义映射起来。

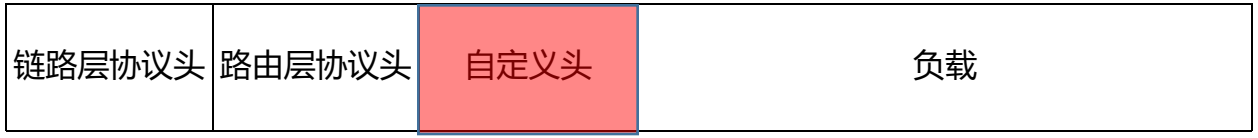


两种封装机制

① 自定义协议栈

优势：设计灵活、高效

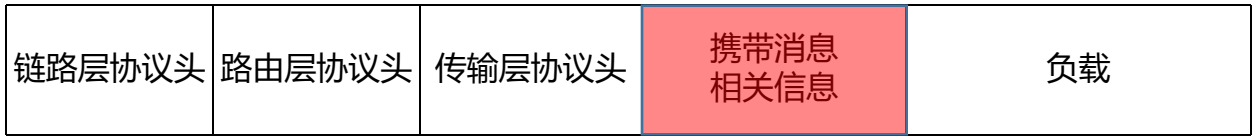
劣势：开发复杂度高，技术封闭



② 基于现有协议栈（如RDMA）

优势：兼容性高，可复用现有成熟加速技术

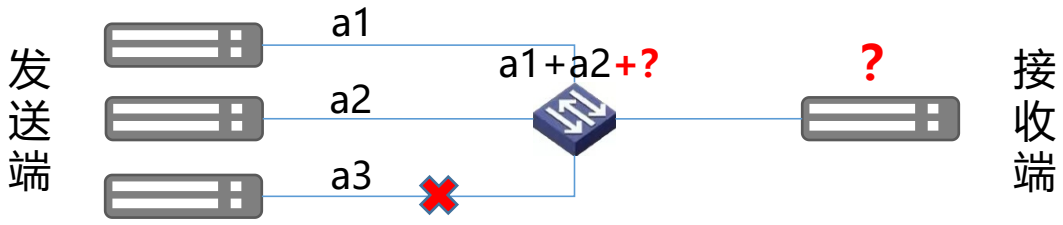
劣势：方案不灵活，传输效率相对较低



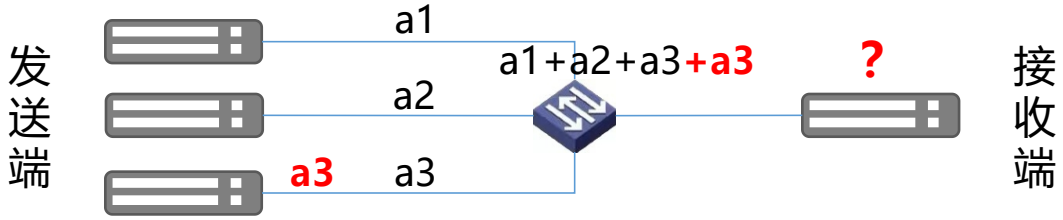
关键技术2：计算正确性保障

在网计算要保证与端侧计算的结果等价，即保证计算正确性。计算正确性还受丢包影响，网络拥塞和乱序则会加剧丢包，因此网络拥塞控制、可靠性传输是在网计算正确性和计算效率的保障。

问题：丢包、重复包影响计算正确性



聚合时发生丢包

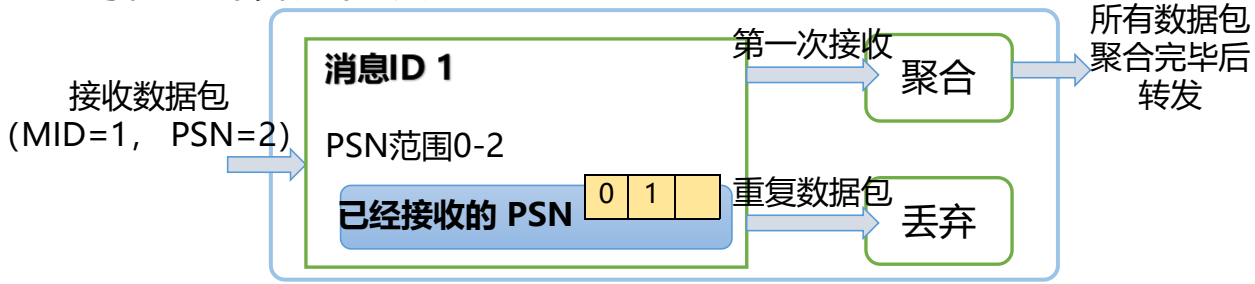


聚合时收到重复数据包

方案：依靠拥塞控制和可靠性传输降低丢包

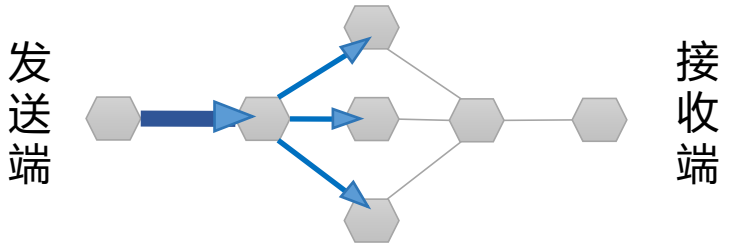
可靠性传输

- 利用bitmap高效记录已收到和已处理包的序号；
- 基于现有可靠性传输协议如Go-Back-N、选择性重传等，针对在网计算进行改进。



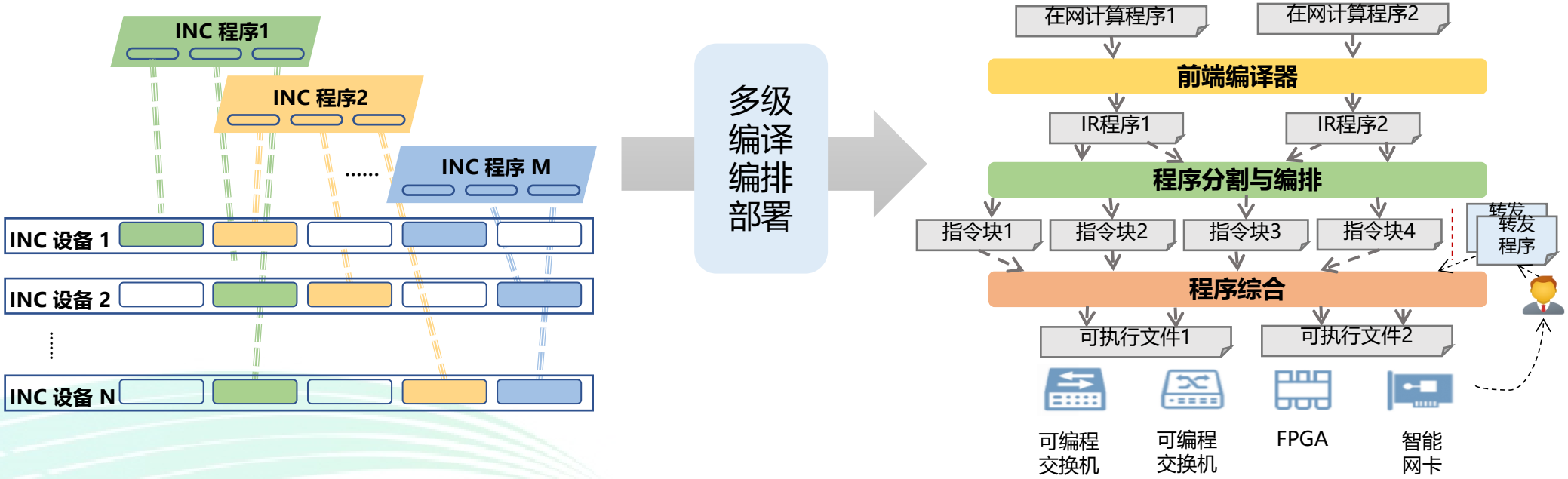
拥塞控制协议

- 优化网络负载均衡方案，避免负载不均导致的拥堵；
- 基于现有PFC、ECN、DCQCN等流量控制机制针对在网计算进行改进



在网计算多级编译编排部署架构实现程序、元素、执行一致性保证

问题	方案
硬件、拓扑、指令、能力多级依赖	构造IR块，解耦硬件、拓扑和指令依赖
多任务共享设备，程序段集成困难	动态规划算法，实现程序段高效放置
分布式分段部署，编译加载难	程序段DAG表示法，合并DAG指令除冗
异构设备多、组合多、指令冗余	异构适配程序段连接和加载

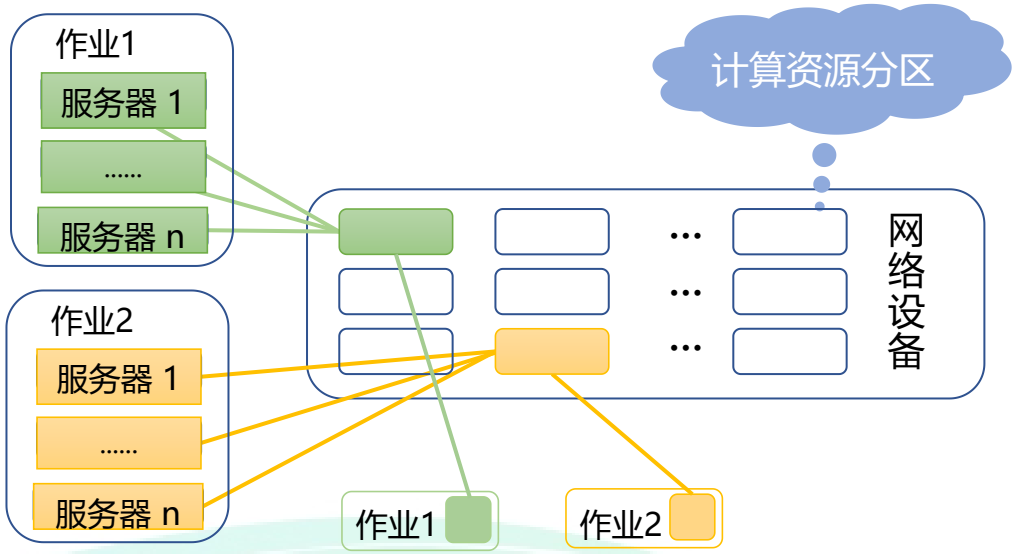


关键技术4：网内资源池化

虚拟化和池化管理技术，统一北向接口，屏蔽异构硬件差异

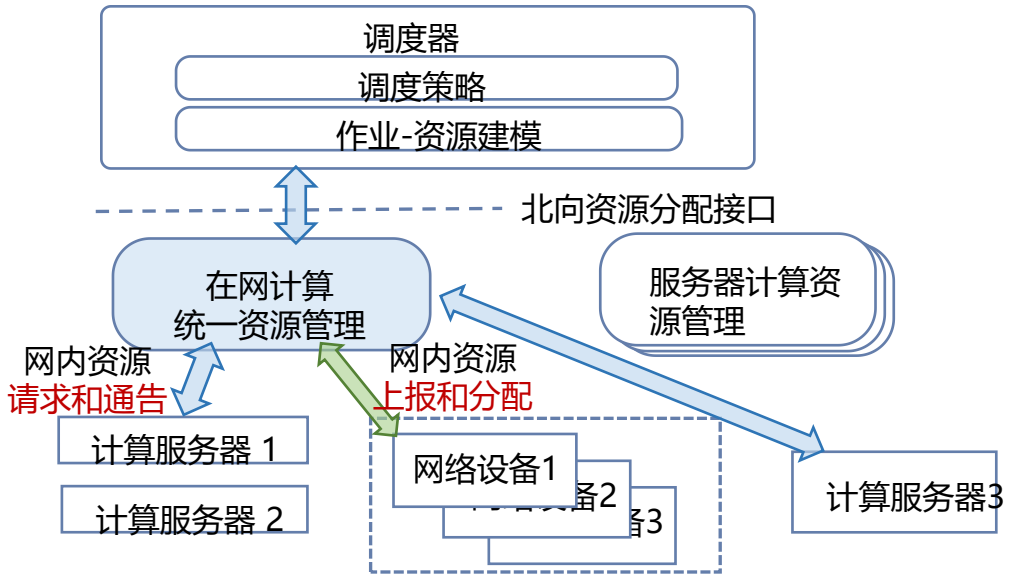
问题 计算、传输周期交替，内存利用率待提高

方案 **设备内存虚拟化**
多租户、多实例、细粒度、动态分配



问题 异构网络设备对接，端网资源一致性

方案 **跨设备资源统一池化**
北向统一对接调度、南向注册异构网络、南向通知计算服务器、一致性更新协议



关键技术5：应用友好的编程语言

在网计算编程语言满足应用开发和网络开发双重需求

问题

- 应用开发跨界、跨平台
- 大规模、异构开发、复杂度高
- 已有语言面向数据包编程，不友好

方案

- 统一在网计算编程接口
- 提供已有INC程序模板
- Python和IR组合的类汇编语言

分布式应用：AI训练/推理　大数据　HPC

软件框架：TensorFlow　PyTorch　Spark　Flink　MVAPICH

应用开发者

功能模块库

统一在网计算编程语言

网络运维语言：Lyra　P4all

领域专用语言：P4　NPL　Micro-C

应用开发者

功能模块库

统一在网计算编程接口

Operation	Explanation	Supported devices
_ram	1D-memory accessed by index	All
_cam	content-addressable memory	FPGA, NFP
_tcam	ternary-content-addressable memory	FPGA, NFP
_emt	stateless exact-match table	All
_semt	stateful exact-match table	FPGA, NFP
_tmt	stateless ternary-match table	All
_stmt	stateful ternary-match table	FPGA, NFP
_lpmt	longest-prefix-match table	All
_randint	achieve an integer random value	All
_crc	CRC series hashing calculation	All
_identity	identity-map hashing	Tofino series
_aes	AES series en(de)-crypto calculation	FPGA
_ecs	ECS series en(de)-crypto calculation	NFP
_checksum	csum16 calculation	All
_mirror	mirroring a packet	All
_multicast	multicasting packet	Tofino series, TD4

提供已有INC程序模板

Program $G ::= var=E \mid G \mid \text{if } C: G \text{ else: } G \mid \text{for } C: G$

Predicate $C ::= (E \& E) \mid (E \mid E) \mid \sim E$

Expression $E ::= V \mid var \mid const \mid F \mid E \odot E$

Function $F ::= \max() \mid \min() \mid \text{range}() \mid \text{slice}() \mid << \mid \cdots$

Field $V ::= \text{value} \mid \text{header}$

Object $O ::= \text{Table} \mid \text{Array} \mid \text{Hash} \mid \text{Seq} \mid \text{Sketch} \mid \text{Crypto}$

Primitive $P ::= \text{get}(O) \mid \text{write}(O) \mid \text{clear}(O) \mid \text{count}(O) \mid \text{del}(O) \mid \text{drop}() \mid \text{fwd}() \mid \text{copy}(O, V)$

Python和IR组合的类汇编语言

```
Prog ::== declare | operation
declare ::== header | parse | data | instance
header ::== h_type string {hBody}
hBody ::== struct {hFields}
hFields ::== type<length> string
type ::==  int | float | bit | bool
length ::== 1,2,...,1024
parse ::== cond? extract(hBody)
data ::== type string
instance ::== emt | semt| tmt | stmt | lpmt |
    cam | tcam | ram
operation ::== cond? statement | statement
statement ::== data = operation | operation
operation ::== data calc | instance action
action ::== write | get | drop | mirror |
    multicast | randint | crc | aes | ecs|
    calc
calc ::== + | - | * | / | % | bit operation |
    >>const | <<const
condition ::== state | state&&state | state||
    state
state ::== data compare
compare ::== > | >= | == | <= | <
```

- **深化在网计算技术攻关**
- 围绕在网计算关键技术挑战进行联合攻坚，共同探索解决方案，推进在网计算成为网络内生的普适能力。
- **推动在网计算技术开源及标准化**
- 逐步开展在网计算标准制定及开源工作，突破行业技术壁垒，共同构筑开放共享的在网计算发展局面。
- **开展在网计算联合试验验证**
- 基于中国移动CFITI试验平台，联合开展在网计算创新技术验证，不断推进产业成熟。

希望携手产业界推进在网计算NACA技术的开拓和研究！

中国移动研究院公众号

中移智库公众号