CANN hixl仓介绍及相关优秀实践分享

CANN

目录

hixl仓介绍

hixl（Huawei Xfer Library）是CANN通信库中提供点对点数据传输能力的单边通信库，面向集群场景提供简易、可靠、高效的点对点数据传输

- **链路管理**：支持多样化的传输链路管理，快速完成同构/异构集群场景动态扩缩容，构建可靠的集群能力。
- **数据传输**：支持集群间/集群内的D2D、H2D、D2H、H2H高效数据传输，构建高效的集群交互能力。
- **简易可集成**：作为PD分离系统的传输引擎，已经集成至Mooncake社区，并适配vLLM/SGLang等推理引擎。

https://gitcode.com/cann

CANN

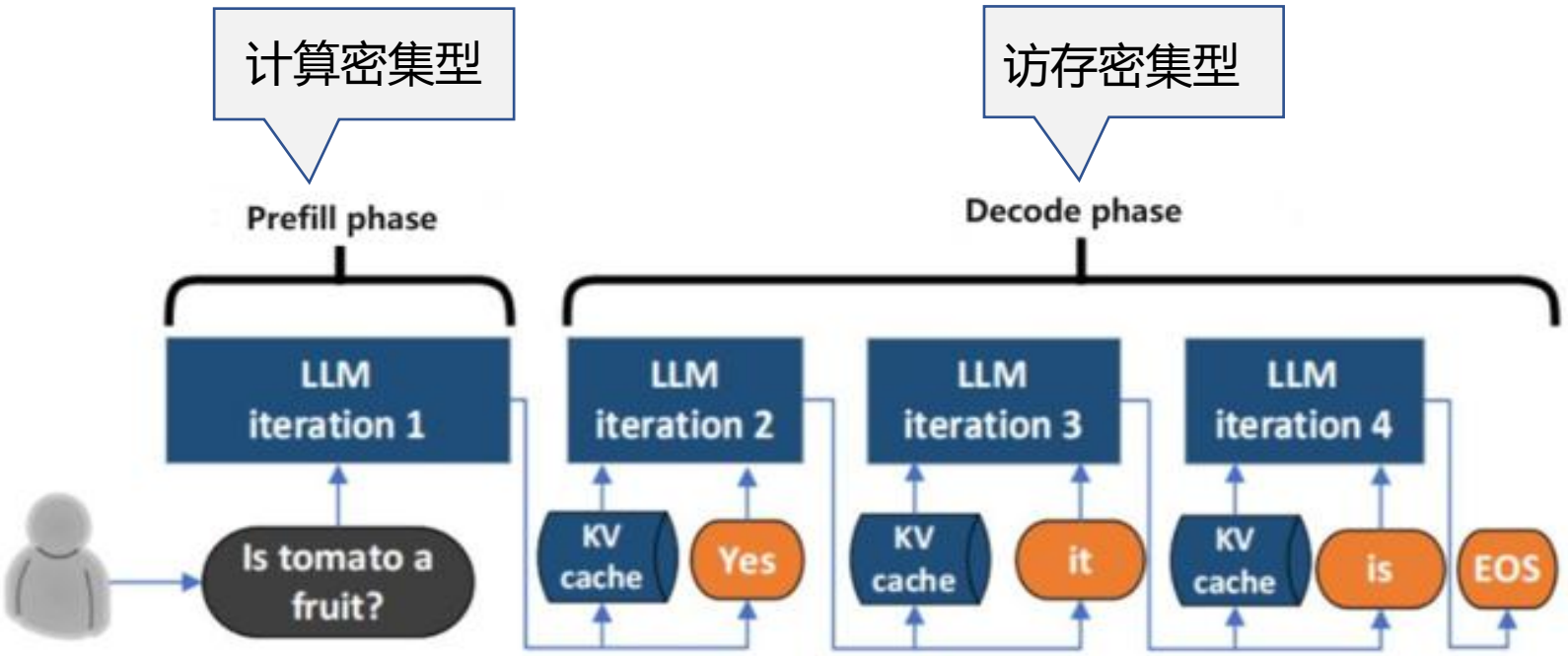
hixl仓目录结构介绍

```
├── build.sh                    # 项目工程编译脚本
├── cmake                       # 项目工程编译目录
├── CMakeLists.txt              # 项目的CMakeList
├── docs                        # 项目文档介绍
│   ├── cpp                     # C++文档
│   └── python                  # Python文档
├── examples                    # 端到端样例开发和调用示例
│   ├── cpp                     # C++样例
│   ├── python                  # Python样例
├── include                     # 头文件
│   ├── hixl
│   └── llm_datadist
├── README.md
├── scripts                     # 脚本路径
│   └── package
├── src                         # 源码路径
│   ├── CMakeLists.txt
│   ├── hixl
│   ├── llm_datadist
│   └── python
└── tests                       # 测试工程目录
```

- **docs**：包含python和c++接口介绍等md文档；
- **examples**：端到端示例；
- **include**：c++层的对外接口；
- **src**：c++以及python层单边通信的接口实现；
- **tests**：内部实现的测试用例。

CANN

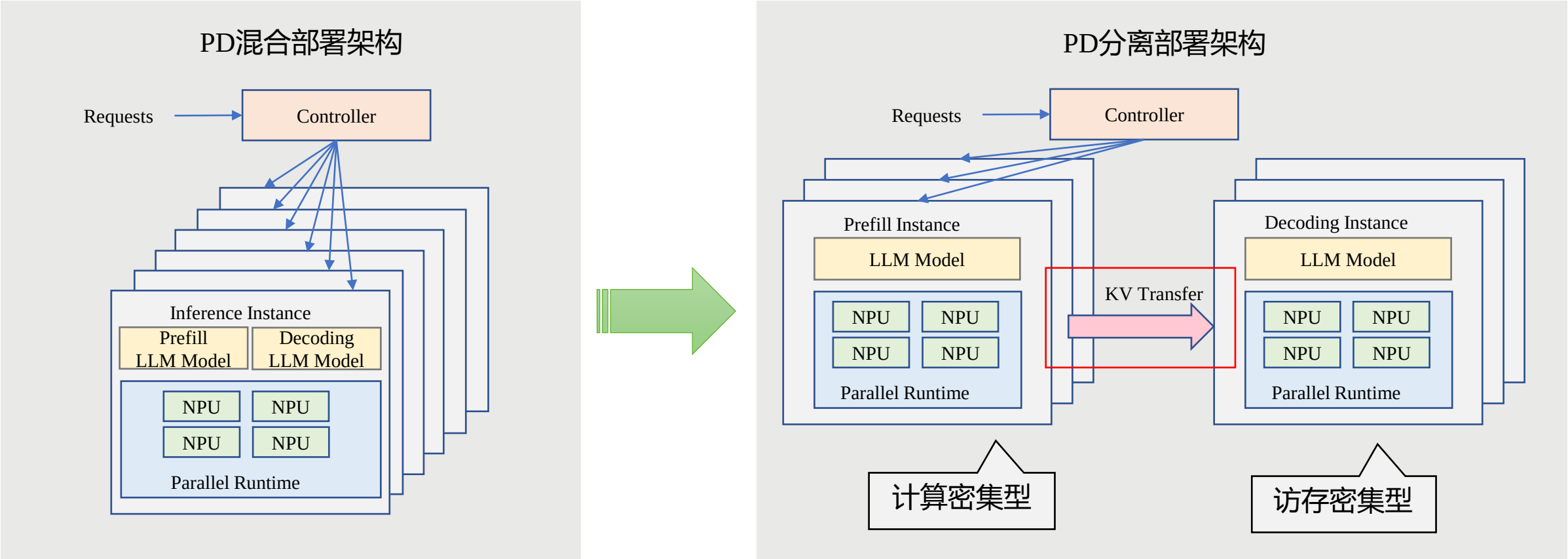
背景小知识: 大模型推理基础



关键指标

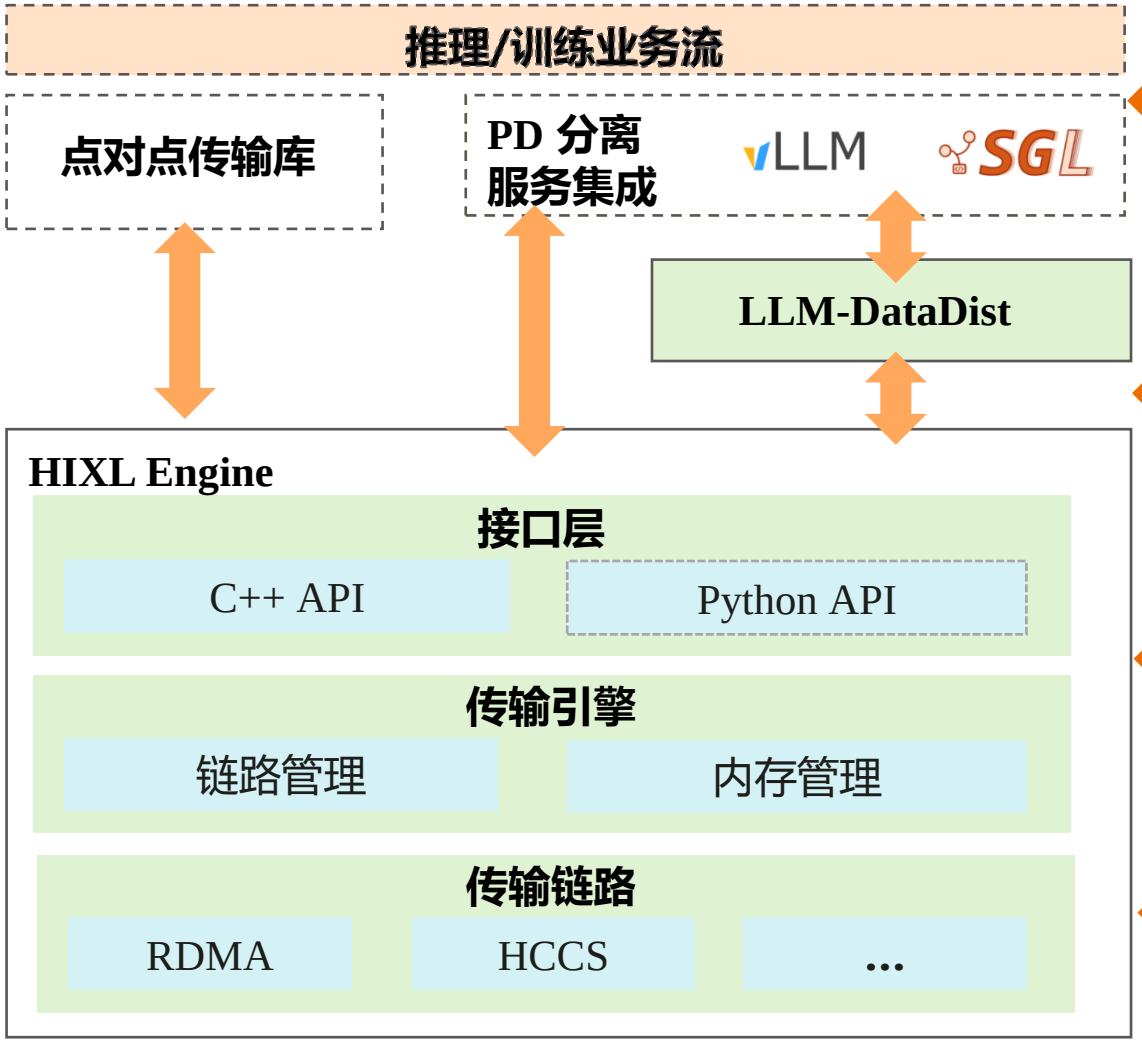
- **TTFT (Time To First Token)** ：模型返回第一个输出 Token 的时间。
- **TPOT (Time Per Output Token)** ：模型生成后续每个 Token 的平均时间。

背景小知识: HIXL的前世今生



- LLM由PD混合部署架构演变为PD分离部署架构，独立优化Prefill/Decoding。
  同时引入了PD之间高效KV Cache传输的诉求，因此HIXL应运而生。
  可用于构建大模型PD分离、RL后训练参数切换、模型参数缓存等多种业务场景。

HIXL架构



多种应用范式：
支持PD分离推理，checkpoint传输等多种传输场景；

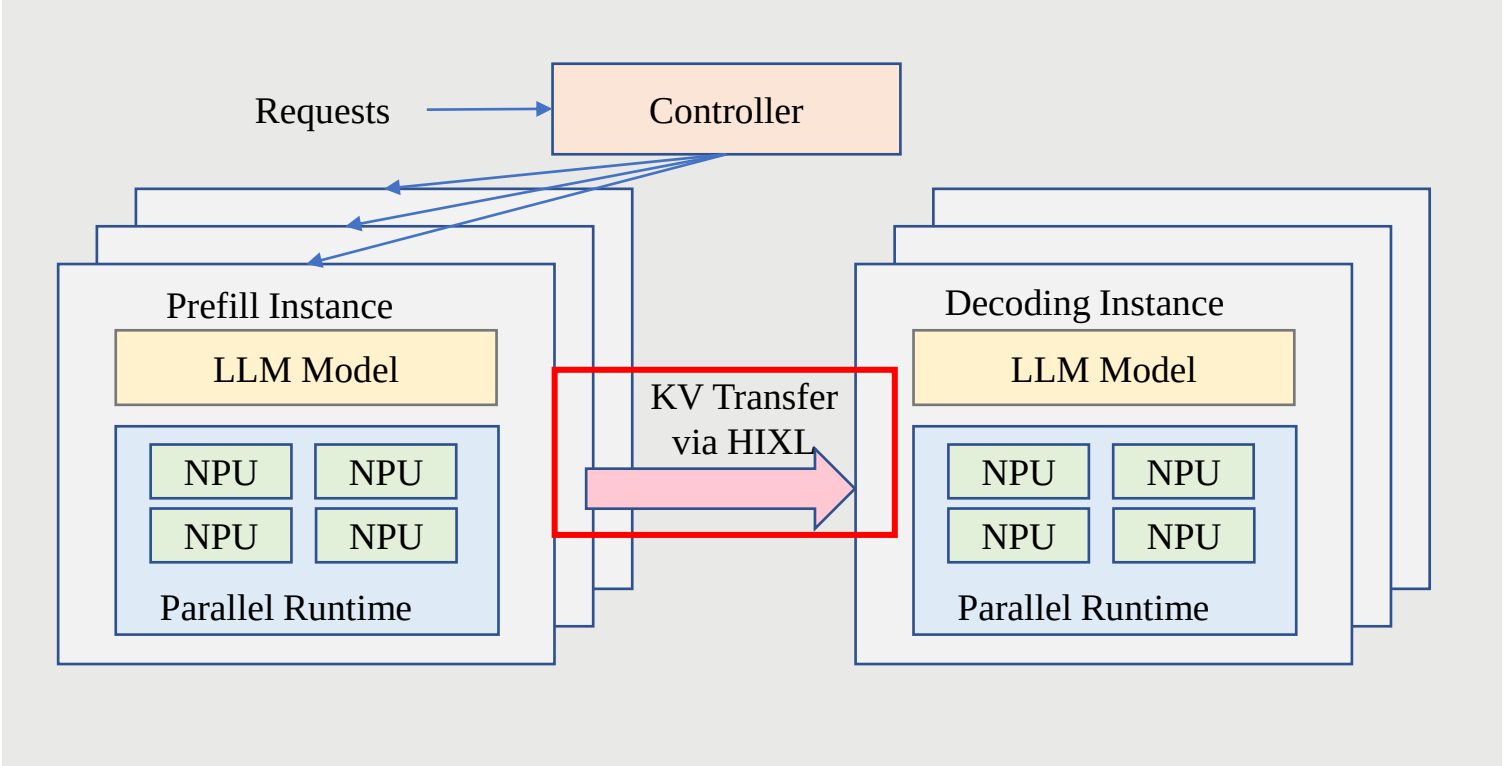
极简 API，深度适配开源生态框架：
接口设计简易高效，目前已实现与Mooncake、DeepLink等开源框架的深度集成；

灵活高效的传输引擎：
支持灵活的链路管理，以及高效的数据传输能力，为集群的动态扩缩容以及各种传输诉求提供可靠保障；

屏蔽硬件差异，高效单边零拷贝机制：
屏蔽多种传输协议以及芯片差异，单边操作完成传输，无需远端参与，零拷贝避免冗余数据搬运；

CANN

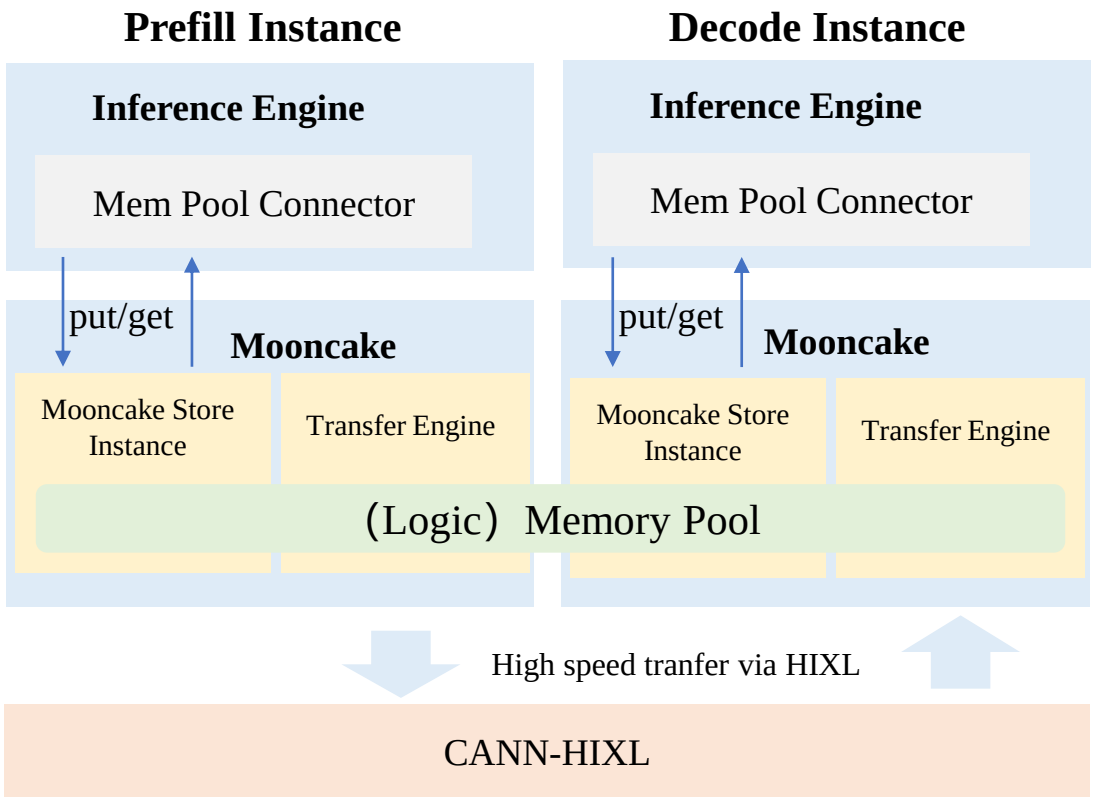
典型应用场景: PD分离模式下KV Cache点对点直传



★HIXL需要提供高效可靠的D2D直传能力

CANN

典型应用场景: KV Cache池化管理

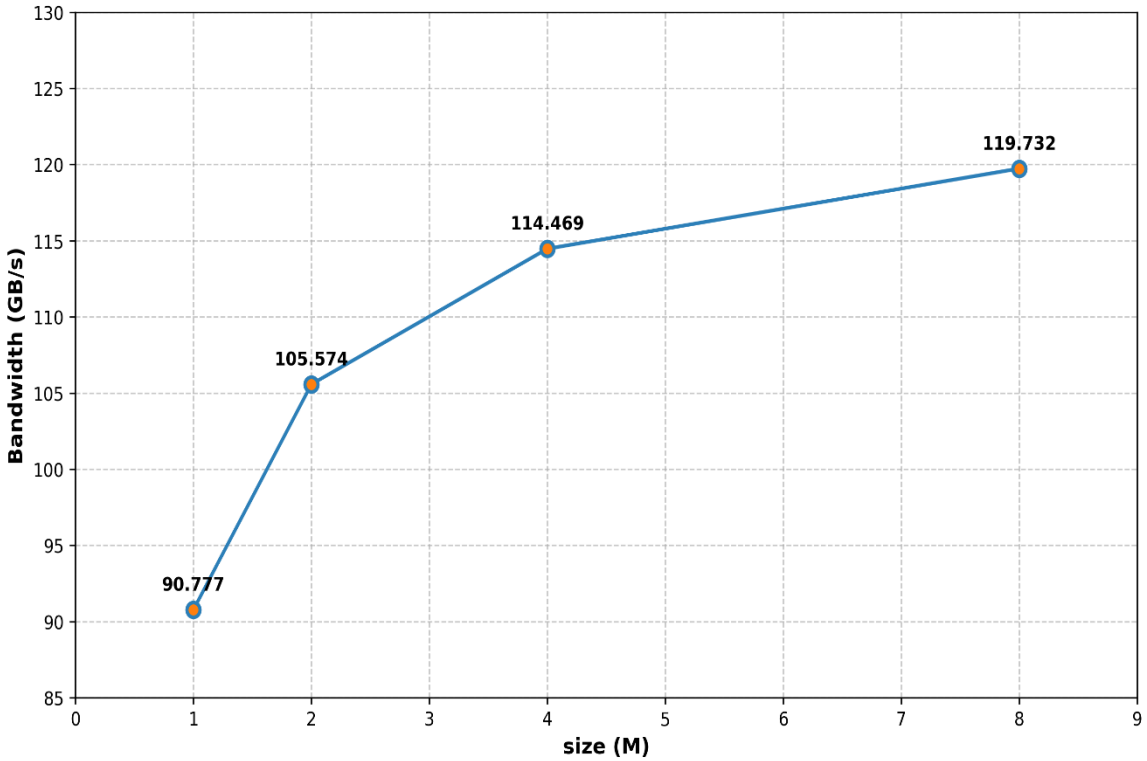


- **零拷贝、高带宽利用**：由 Transfer Engine 提供支持, 数据链路零拷贝，实现高速数据读写；
- **动态资源伸缩**：支持动态添加和删除节点，灵活应对系统负载变化，实现资源的弹性管理；
- **分级缓存**：支持将 HBM 中缓存数据卸载到 DDR 中，以进一步实现成本与性能间的平衡，提升存储系统的效率；

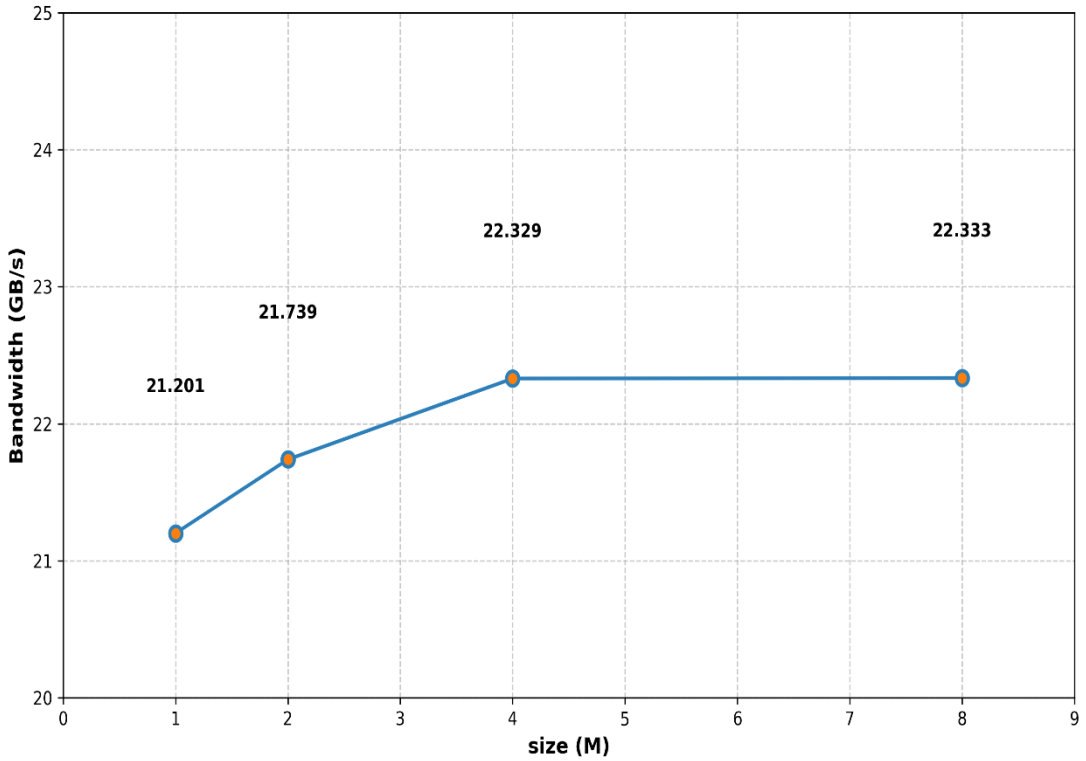
★ HIXL需要支持数据在NPU HBM、CPU DRAM之间的高效流转

HIXL性能表现

总数据量128M场景下不同buffer大小的传输性能



HCCS传输性能



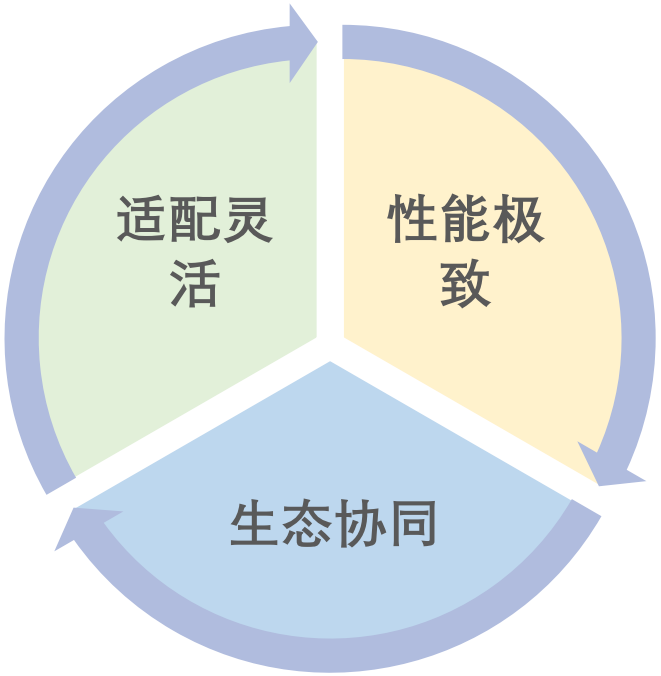
RDMA传输性能

HIXL路标规划

- ☑ 支持多链路D2D/D2H/H2D/H2H高效单边通信
- ☑ 支持小数据批量传输时合并传输优化
- ☑ 支持接入Mooncake/DeepLink开源社区
- ☐ 提供异步数据传输及状态查询API
- ☐ 持续完善异构场景
- ☐ 扩展支持Python接口

目录

HIXL开源生态对接



- 封装统一API，降低底层适配门槛，为大模型推理引擎提供传输层基础设施
- 深度挖掘昇腾硬件潜力，最大化释放硬件性能
- 与Mooncake Store等开源生态深度集成，适配vLLM/SGLang等大模型推理引擎

HIXL并非孤立的传输工具，而是深度融入昇腾生态，为不同用户场景提供端到端支撑

CANN

适配灵活：提供极简API

引擎初始化/去初始化
Initialize/Finalize

- 自动获取NPU通信资源
- 仅需配置host ip/port作为引擎唯一标识

链路管理
Connect/Disconnect

- 仅需指定对端引擎唯一标识
- 链路异常时支持自动清理
- 动态扩缩容

本地内存注册/注销
RegisterMem/DeregisterMem

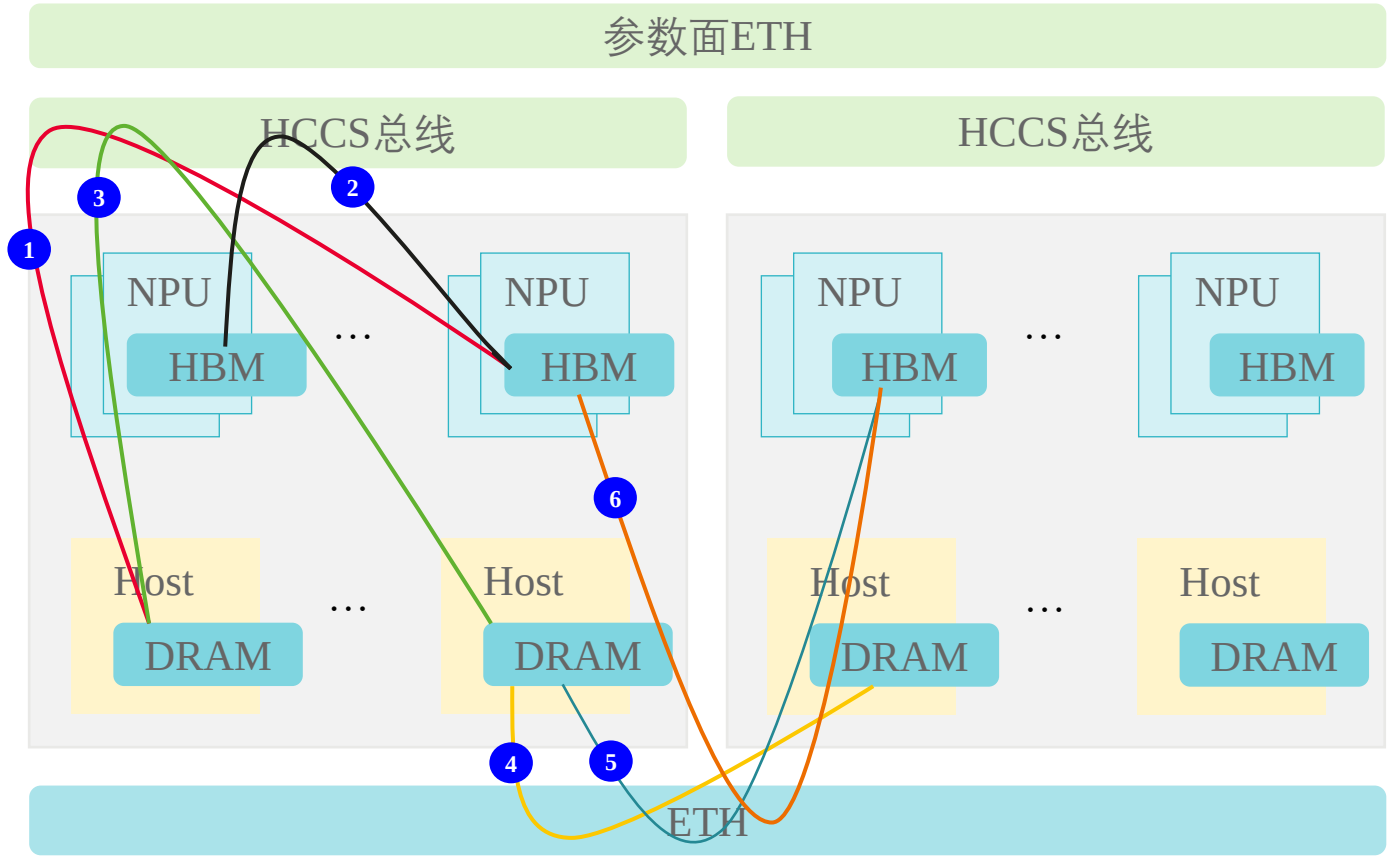
远端内存直访
TransferSync
TransferAsync/GetTransferStatus

- 需指定对端引擎唯一标识，以及本地、远端读写地址信息

屏蔽底层芯片差异，可选多样化传输链路

CANN

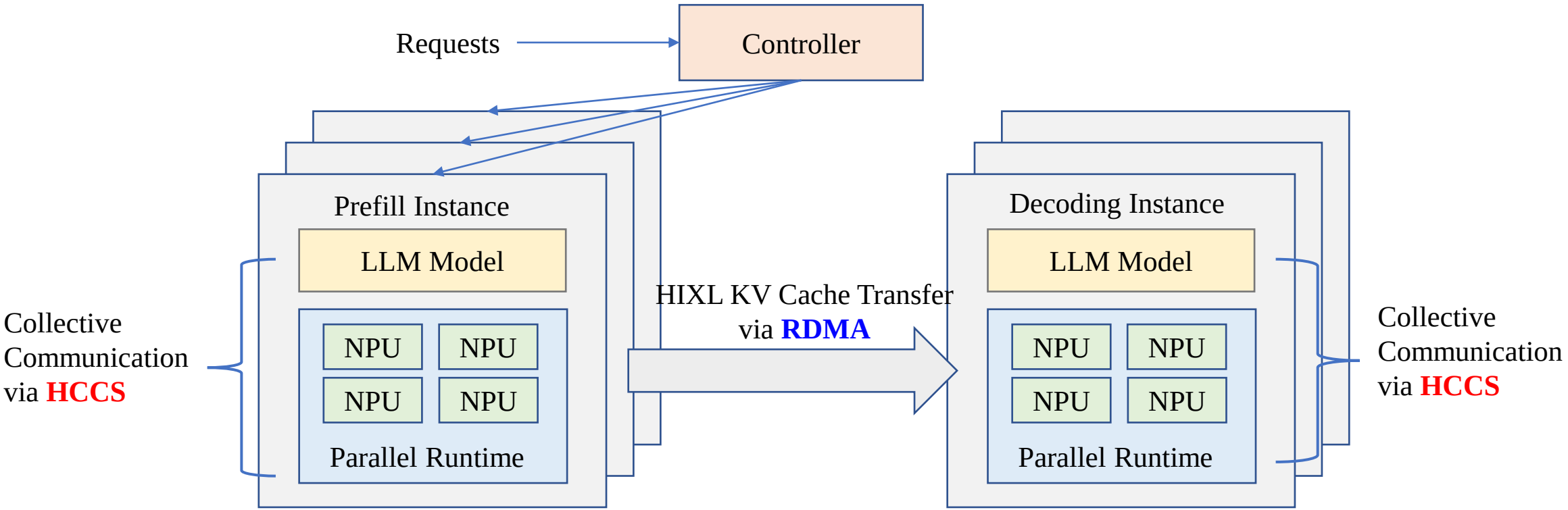
性能极致：充分发挥硬件能力



1. rH2D/D2rH via SDMA
2. D2D via SDMA
3. H2H via SDMA
4. H2H via RoCE
5. rH2D/D2rH via RoCE
6. D2D via RoCE

★ 利用昇腾芯片高速互联（HCCS/RDMA），充分释放硬件能力

性能极致：端到端优化



优化场景：模型集合通信与KV Cache传输存在流量冲突，影响TPOT

优化手段：

- 集合通信与HIXL KV Cache传输可采用不同通信协议，流量做到互不干扰
- 支持流量优先级规划，可为不同业务配置不同流量优先级

CANN

性能极致：批量小Buffer传输优化

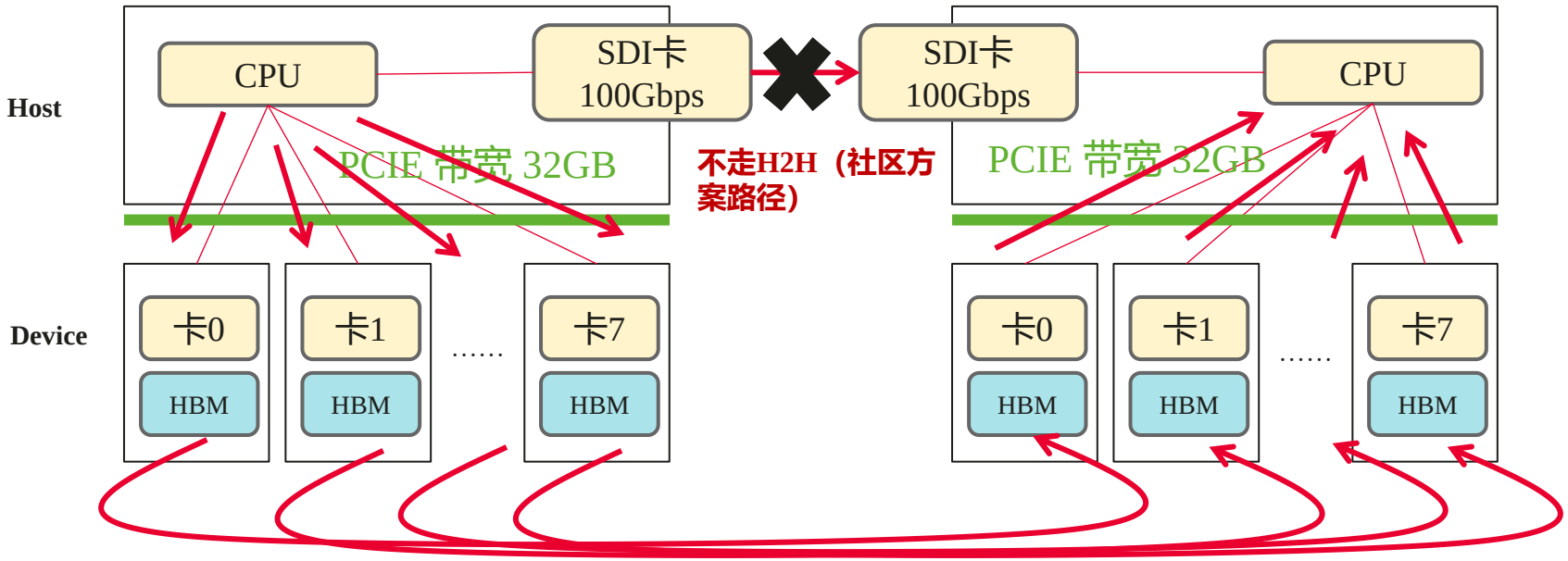
优化措施：支持小Buffer传输任务批量下发，基本消除控制面开销

性能收益：批量128KB Buffer传输测试中，带宽利用率提升**300%**

26 * 64个128KB小Buffer，D2D耗时79ms->19ms

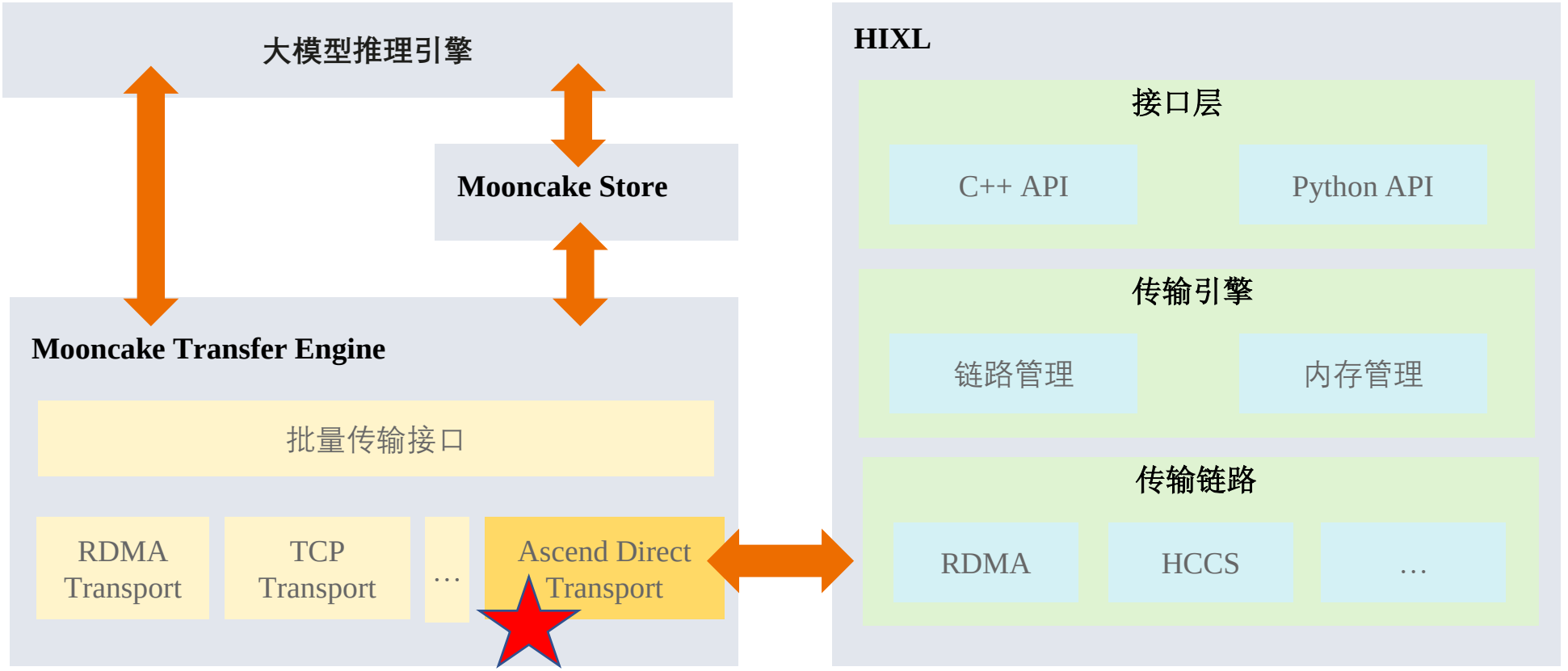
优化措施：支持批量小Buffer聚合传输，利用多级流水实现高性能传输

性能收益：批量128KB Buffer传输测试中，带宽利用率提升超**100%**



CANN

生态协同：与业界主流分布式KV Cache存储引擎Mooncake集成



Mooncake社区贡献：Ascend Direct Transport，开放昇腾多样化、高性能、高可靠的传输能力

CANN

生态协同：Mooncake开源社区贡献

提供Ascend Direct Transport，释放昇腾多样化传输能力

- **核心优势**：低延迟、高带宽，屏蔽硬件差异，多链路跨设备高速互联
- **社区价值**：破解大模型推理/训练数据传输瓶颈，提升端到端效率

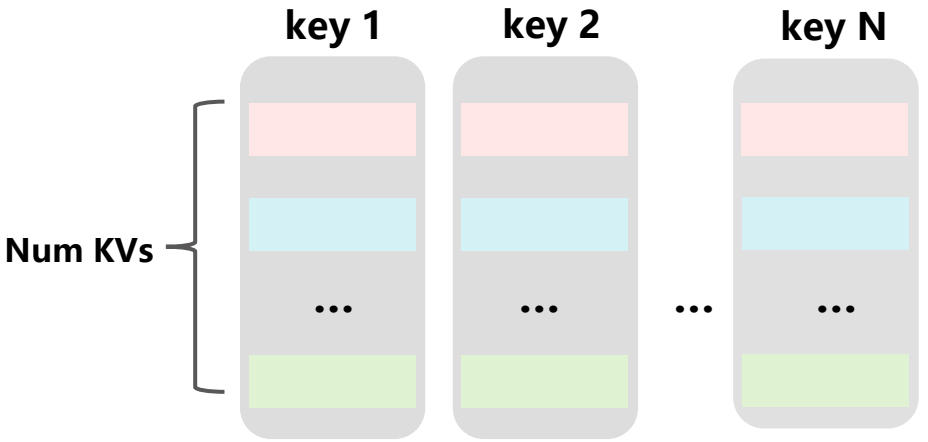
单进程内多种传输任务场景下，Transfer Engine支持单实例

- **核心优势**：替代多实例冗余设计
- **社区价值**：减少内存占用与资源竞争

提供昇腾亲和的batch_put/batch_get API

- **核心优势**：支持多缓冲区批量读写，支持单key索引多缓冲区buffer
- **社区价值**：提升大量小buffer数据读写性能，亲和PagedAttention
  batch_put_from_multi_buffers/batch_get_into_multi_buffers

聚焦昇腾优化
赋能Mooncake社区生态



https://gitcode.com/cann

CANN

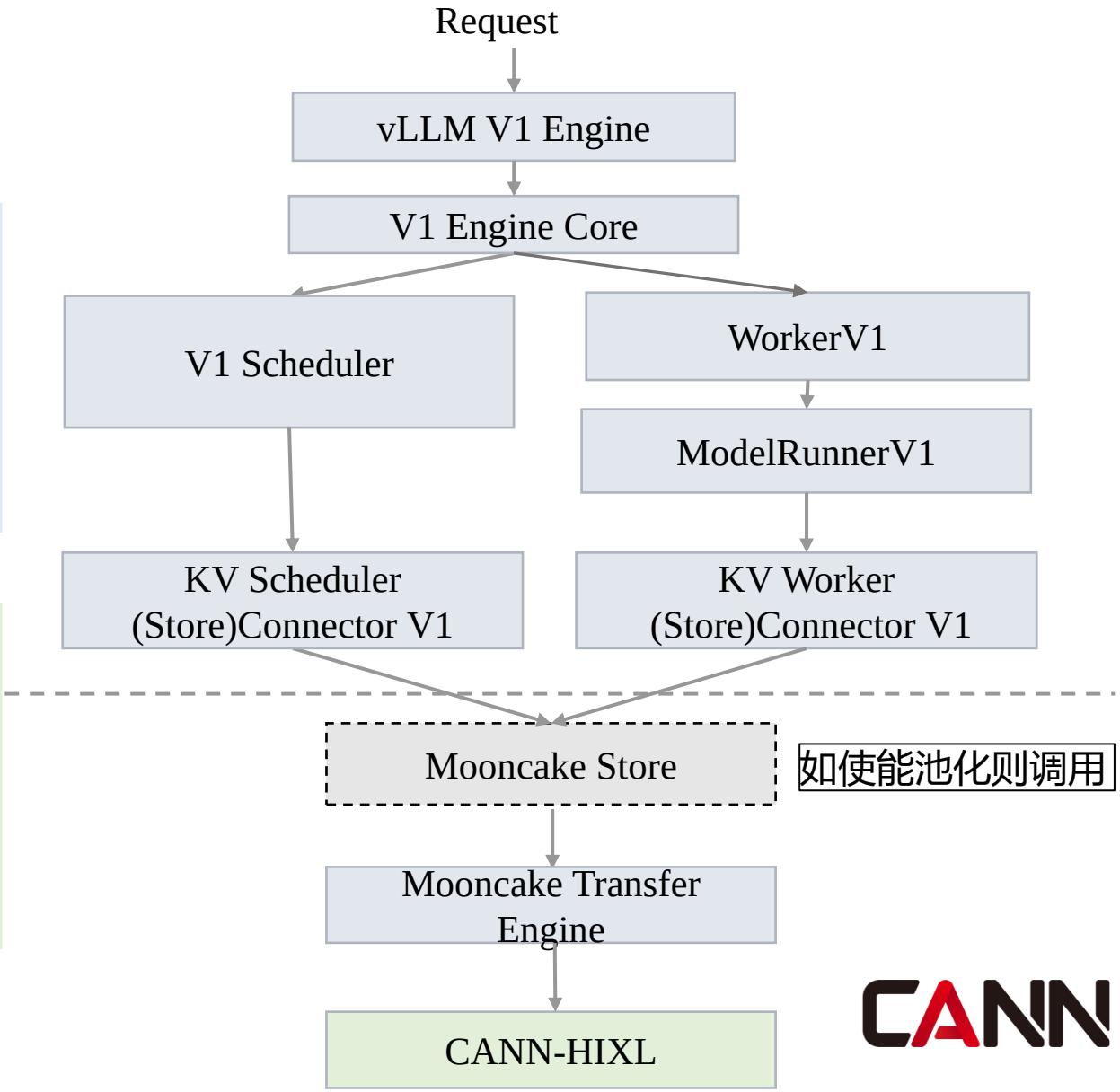
生态协同：与业界主流大模型推理引擎vLLM/SGLang集成

低延迟传输场景：
vLLM的KV Connector直接调用Mooncake Transfer Engine，将推理生成的KV Cache通过P2P通道实时传输，减少中间转发开销

一般仅需D2D

池化复用场景：
大模型推理引擎对接Mooncake Store，配置KV Cache池化策略，将非实时复用的KV数据写入Store进行集中管理，实现跨节点缓存共享

D2H/H2D/H2H



CANN

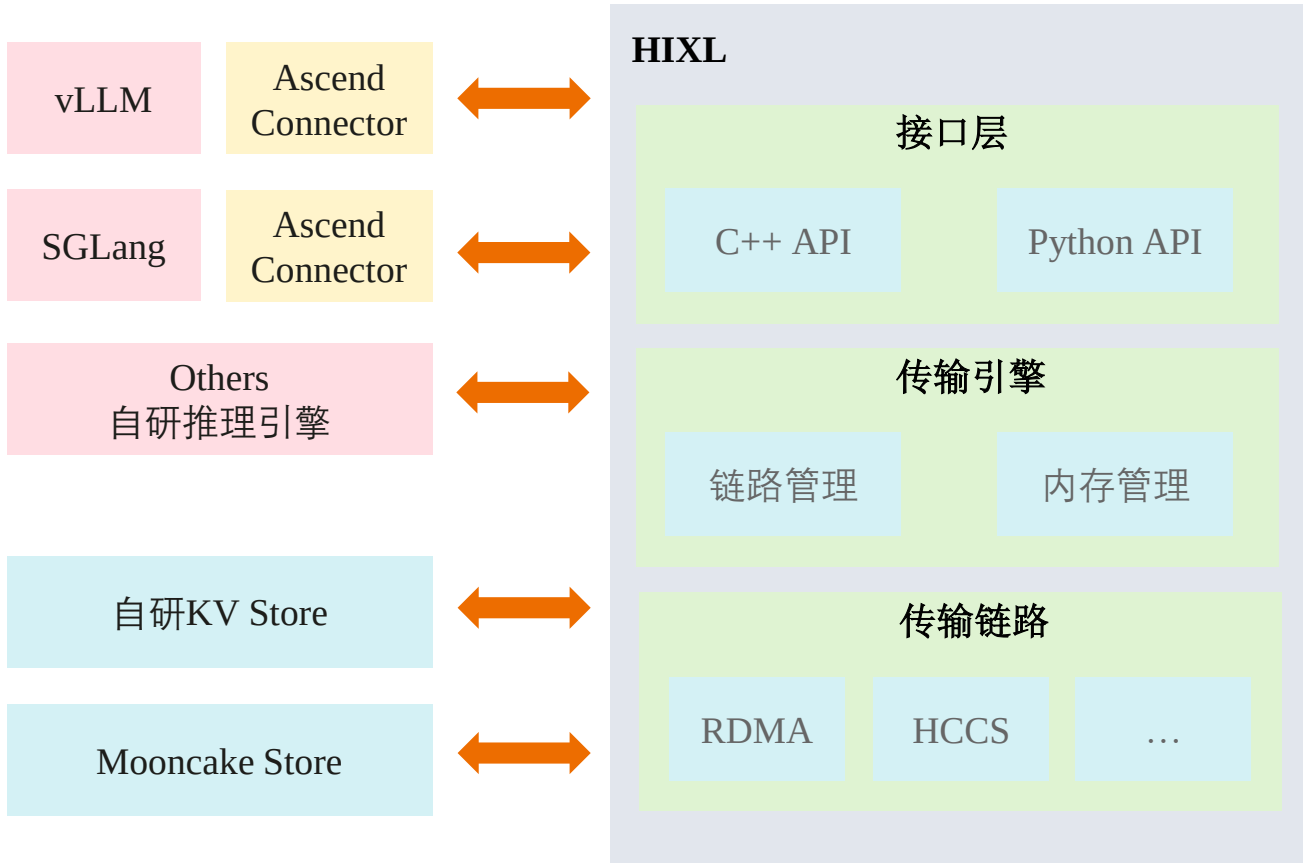
生态协同：与多客户自研推理引擎集成

自研推理引擎的PD分离架构优化：

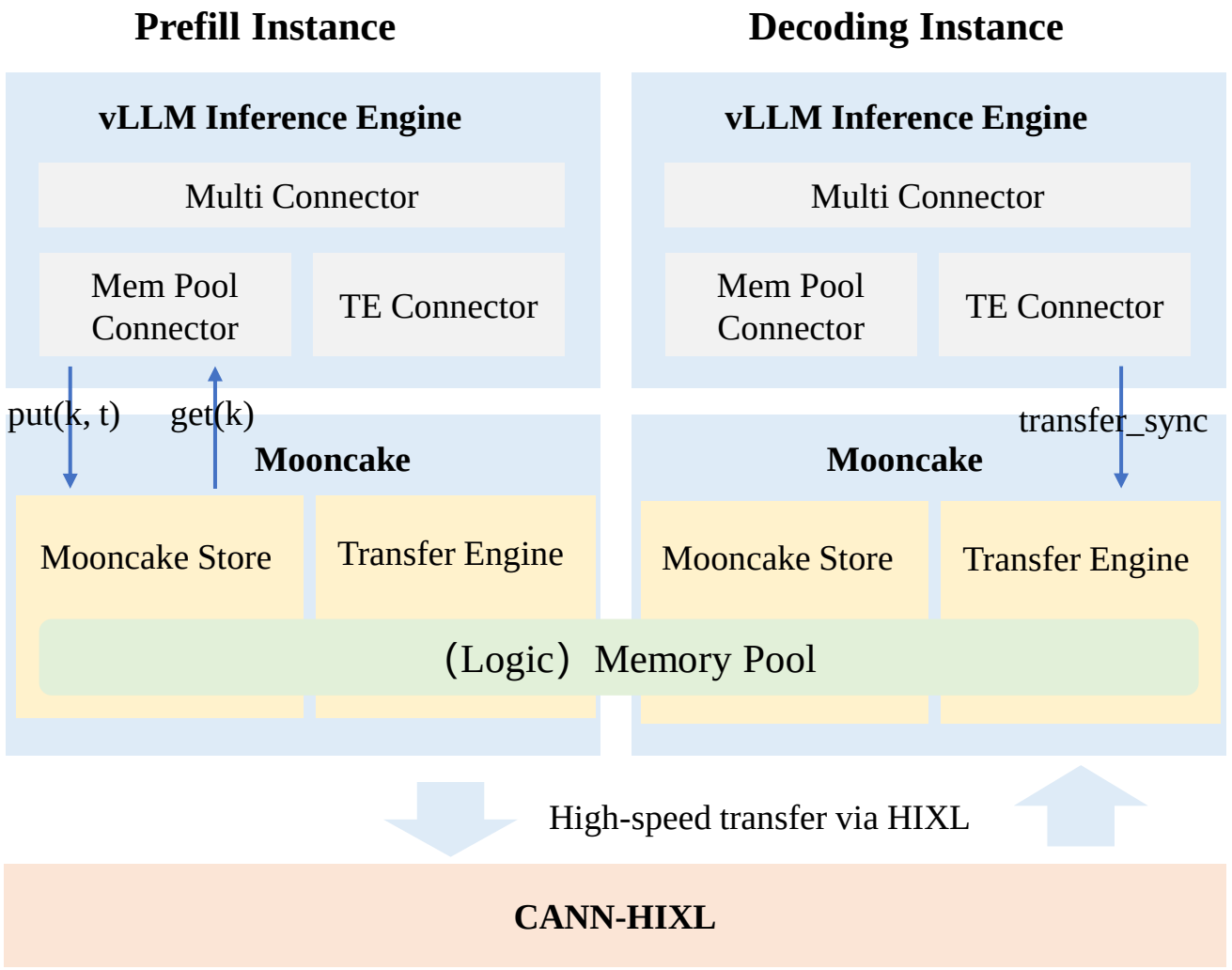
HIXL承担核心数据枢纽角色，将Prefill生成的KV Cache以点对点方式直传至Decode，避免转发导致的冗余开销

定制化KV Cache池化管理引擎优化：

Local KV Cache面临**内存墙**与**共享难**问题，需将空闲CPU内存聚合为全局缓存池
HIXL为此提供跨介质传输能力：支持KV Cache数据在NPU HBM、CPU DRAM之间的高效流转



生态协同：客户联创



架构优化：

- P2P直传：提升PD之间KV Cache传输效率
- 池化管理：提升Prefix缓存复用率，优化TTFT

业务场景：4k输入1.5k输出，prefix 50%匹配

性能收益：TTFT收益40%+

https://gitcode.com/cann

Thank you.

社区愿景：打造开放易用、技术领先的AI算力新生态

社区使命：使能开发者基于CANN社区自主研究创新，构筑根深叶茂、跨产业协同共享共赢的CANN生态

Vision: Building an Open, Easy-to-Use, and Technology-leading AI Computing Ecosystem

Mission: Enable developers to independently research and innovate based on the CANN community and build a win-win CANN ecosystem with deep roots and cross-industry collaboration and sharing.

上CANN社区获取干货

关注CANN公众号获取资讯

https://gitcode.com/cann