

XMFs：跨节点池化共享 文件系统

华为操作系统部
乔一凡



目

CONTENTS

录

01

Background

02

Motivation

03

Design

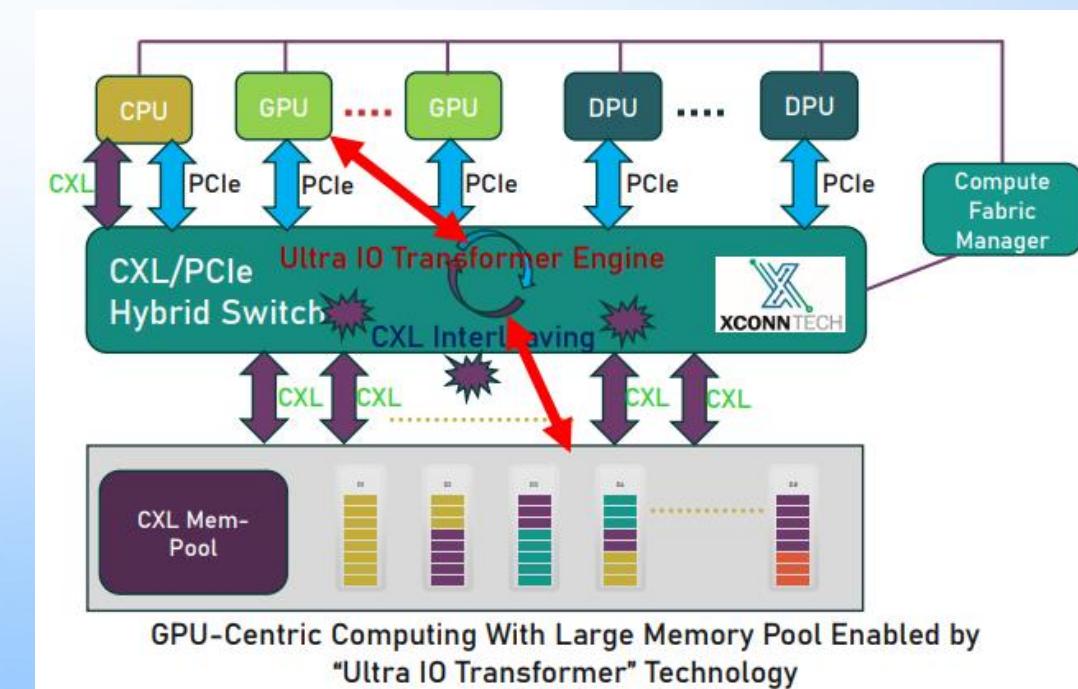
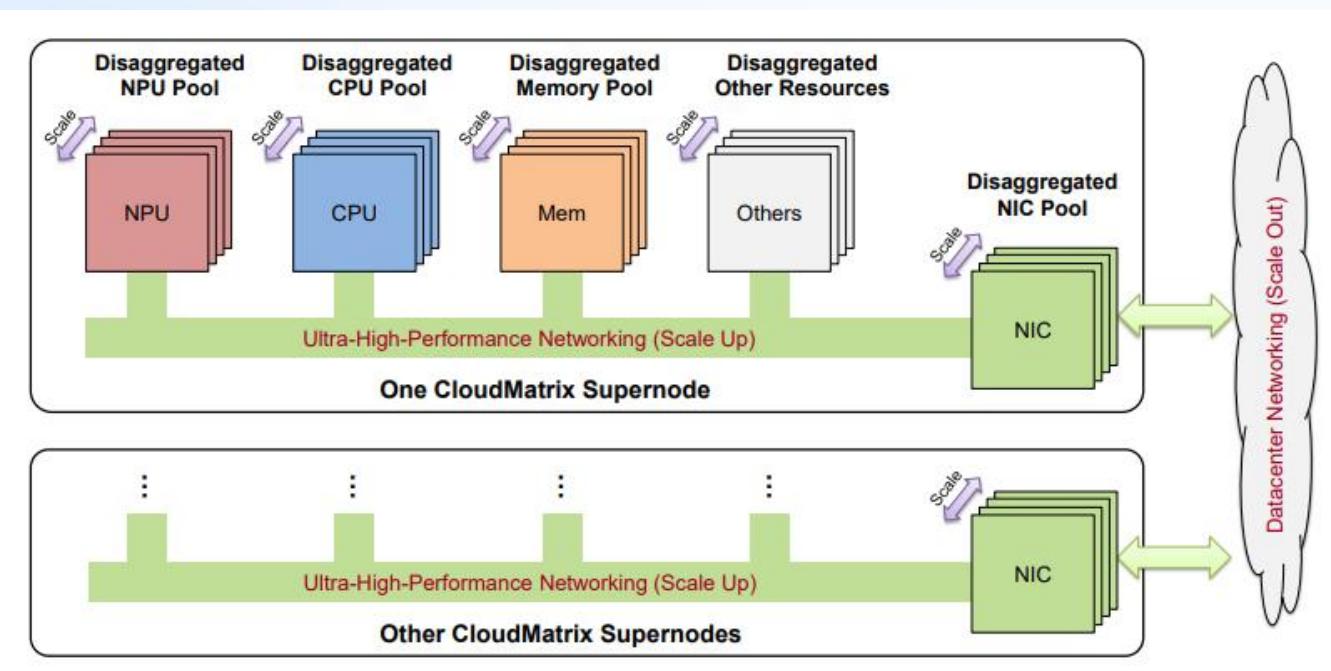
04

Experiments

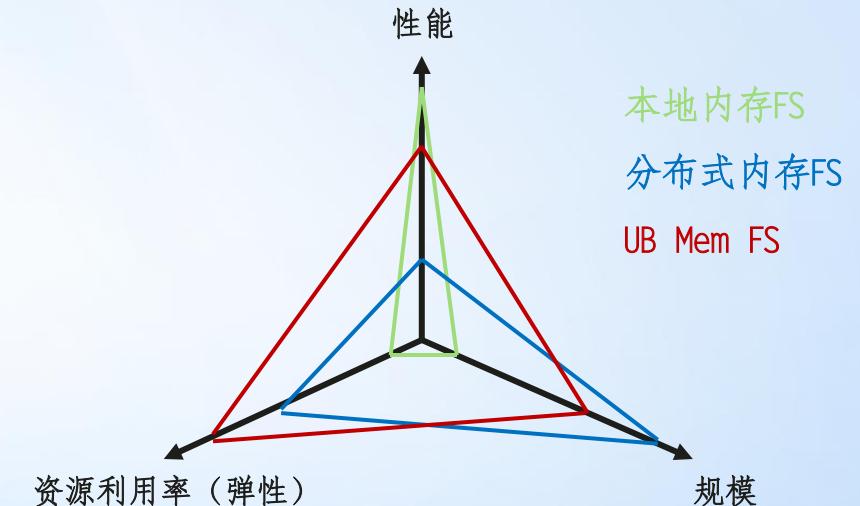
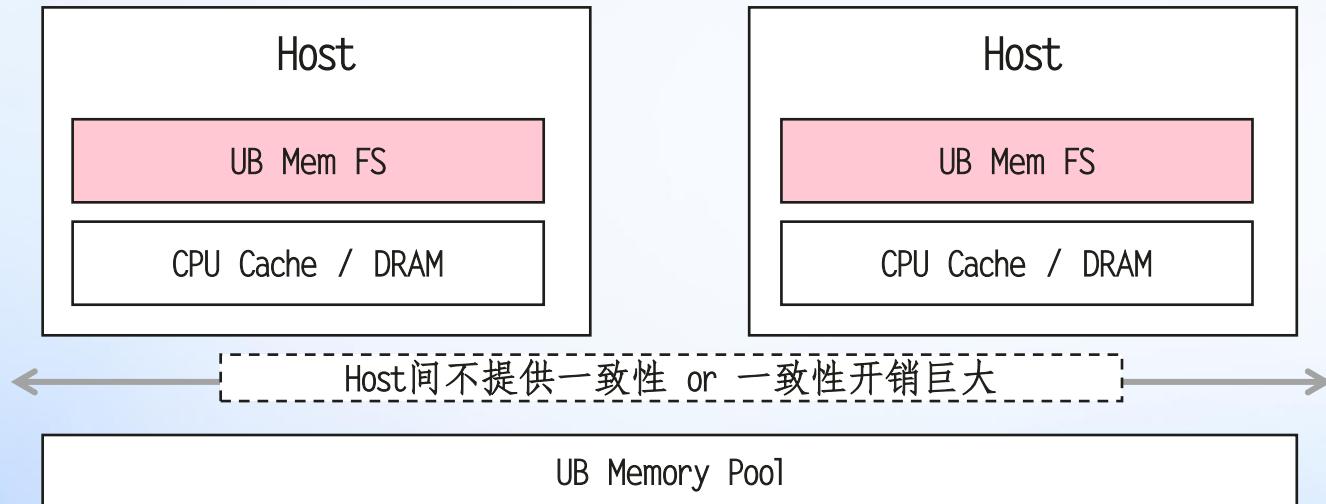


背景：新总线

共享内存友好，均支持ld/st语义直访



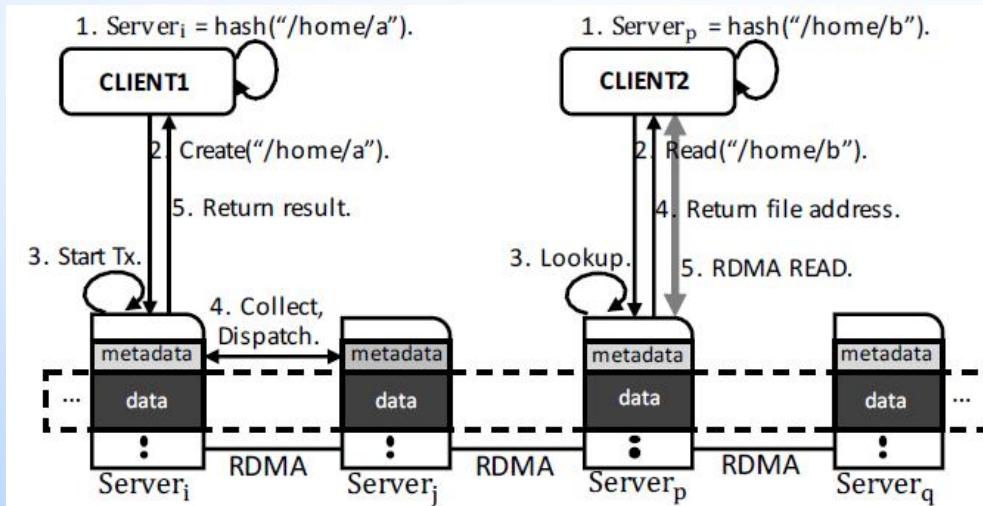
背景：生态



- 跨节点IPC: 作为sockfs, 性能超网络Socket
- 大数据处理缓存: 中间结果缓存, Shuffle加速
- 容器冷启动加速: 减少冷启频率, 降低内存底噪
- 大模型推理加速: 多模态前后处理(编解码), KV Cache
- HPC中间数据缓存: WRF, Openfoam

相关工作局限性

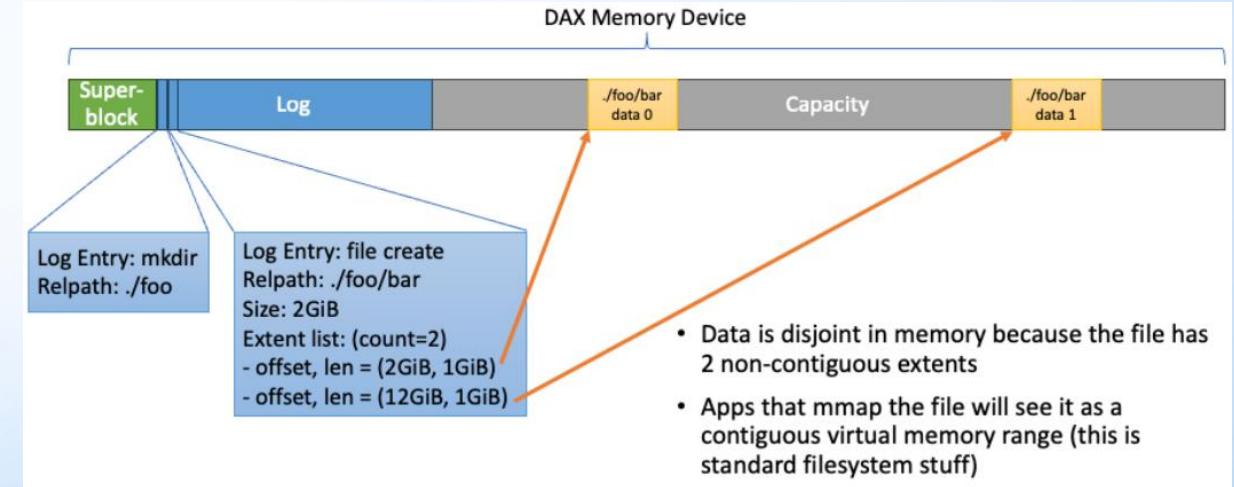
基于网络通信的FS



不足：

- ① 网络：缓存失效通知、依赖网络性能（不丢包）
- ② 过程复杂：多次数据拷贝
- ③ 不支持cc

FamFS

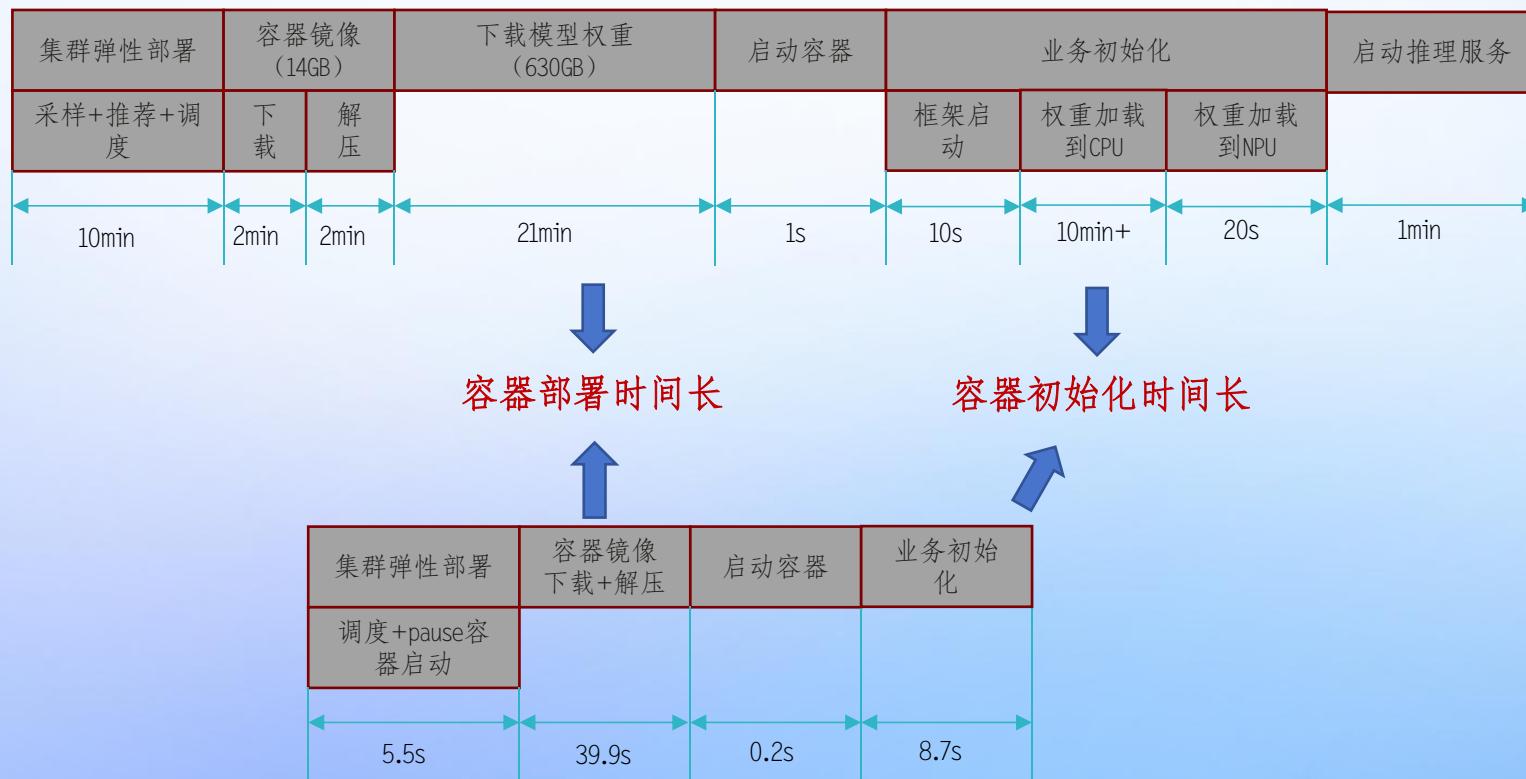


Famfs-fuse:

- ① 元数据开销大
- ② 用户态可靠性无法保证
- ③ 无跨Rack能力
- ④ 使用不友好，功能不完善

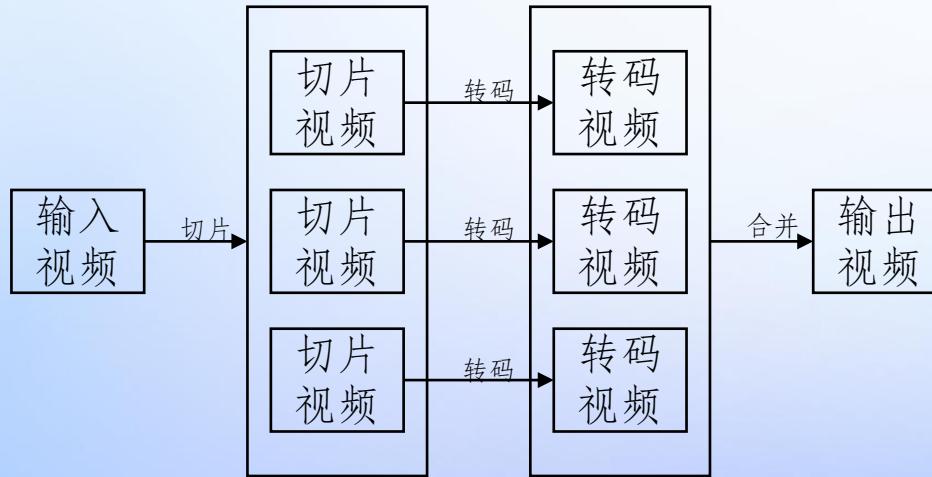
背景：容器弹性场景

容器弹性生命周期

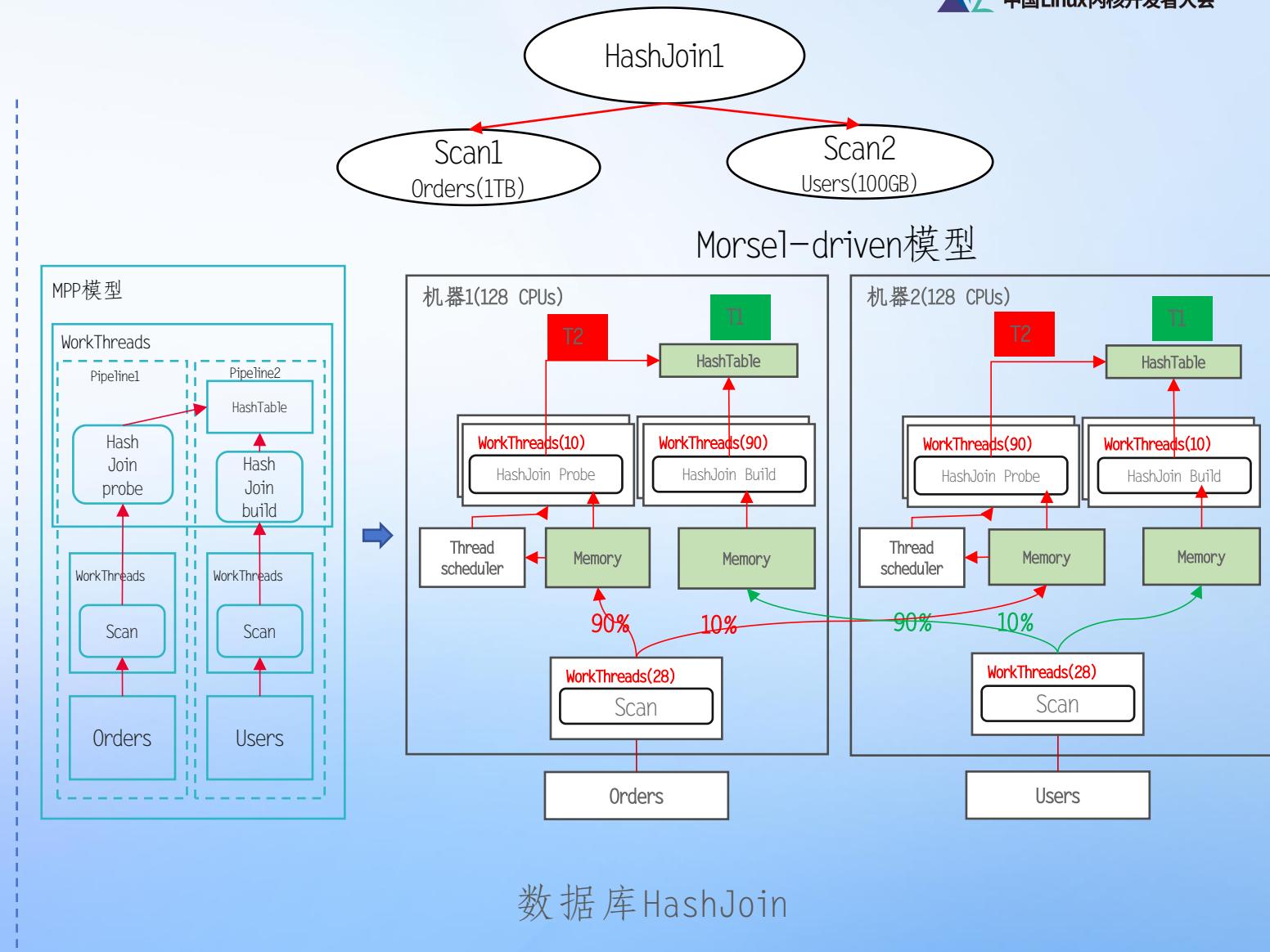


背景：场景驱动

算力开销： N 次编码 + N 次解码



视频编解码服务



Motivation

问题1：传统共享方案开销大

- 传统共享方案需要考虑序列化的问题
- 网络分布式方案的时延远大于内存语义的时延，且涉及多次数据拷贝
- 业务进行大量适配



设计1：基于低时延的内存语义，减少共享过程中的数据拷贝，消除序列化过程，减少数据共享开销。最小化业务更改

问题2：可靠性保障及负载均衡

- 节点故障不能扩散
- 非集中式场景，需要考虑负载均衡，读性能受并发数影响严重



设计2：在Rack内使用多副本部署，将读负载分部到多节点上提高读带宽和读性能，并通过多副本实现可靠性保障

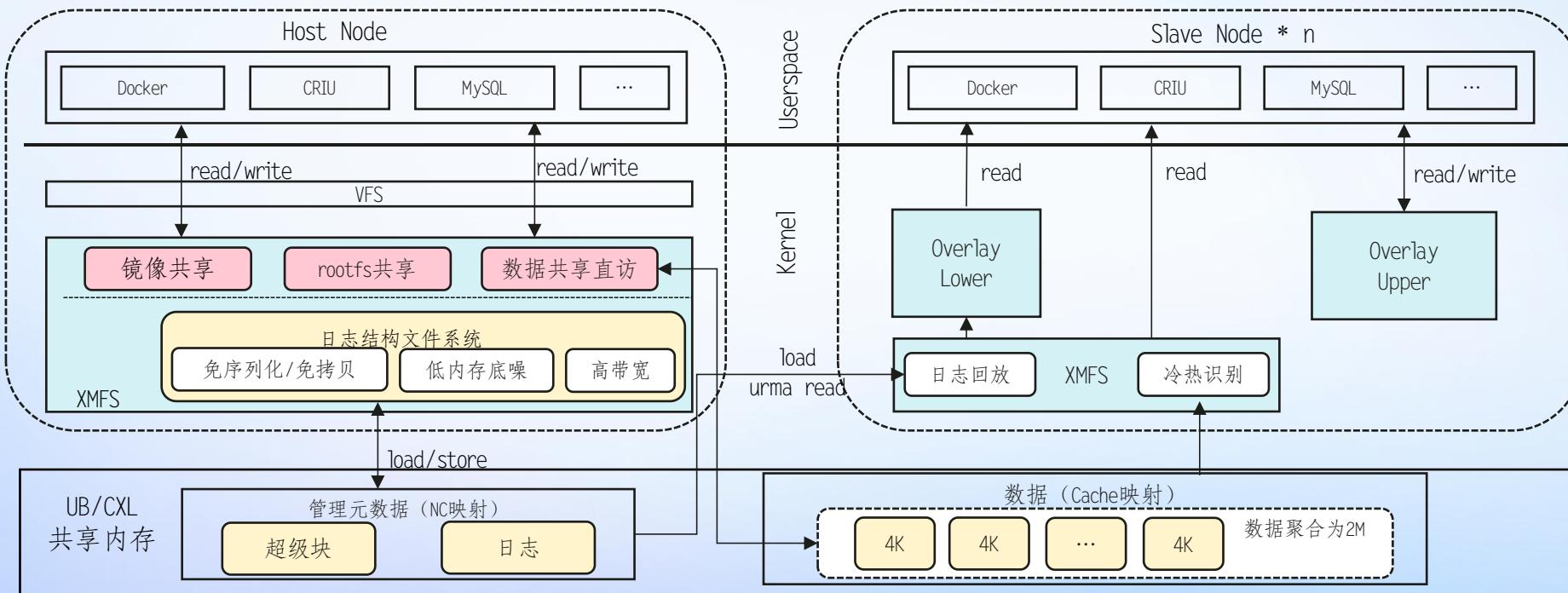
问题3：不提供全局硬件cc能力/硬件cc不高效

- 当前新的互联总线没有高效的广泛的硬件cc能力，通过网络消息广播读写请求做clflush开销过大



设计3：针对只读共享、一写多读的场景，实现基本的软件cc能力

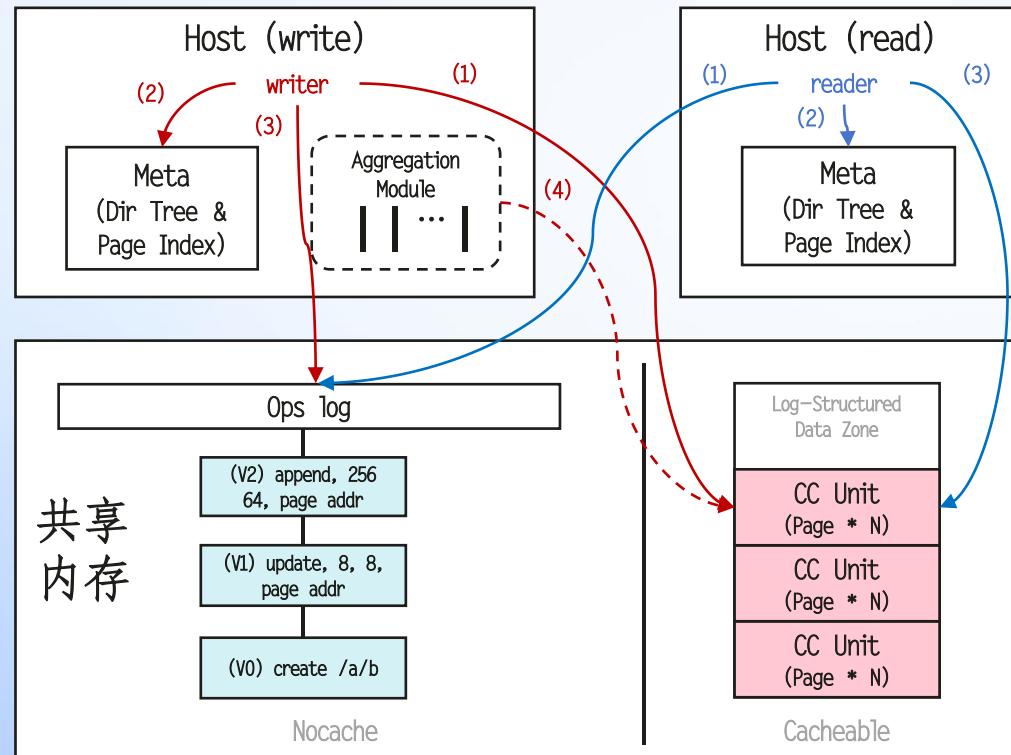
设计



关键技术：

1. 日志结构文件系统及复制协议：通过日志结构管理文件系统，通过回放在节点间快速同步，仅追加写。
2. 级联共享：通过多节点多副本部署，获取full mesh共享能力。
3. 软件cc：元数据面通过noncache模式进行全局一致性维护；数据采用Cache模式。
4. 可靠性设计：safe_copy + 多副本
5. 读节点性能加速：通过冷热识别算法，将热数据加载到本地，提高后续热数据的读性能。

关键设计

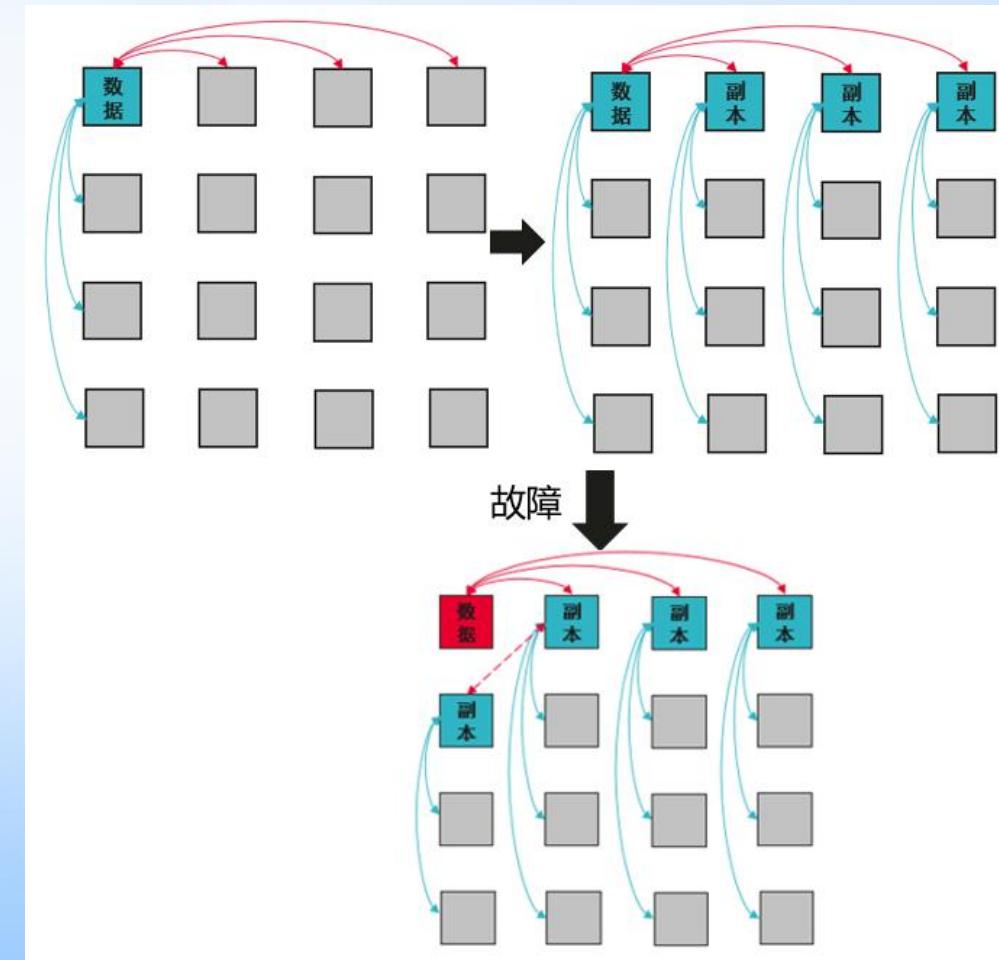


写流程：

- ① 新数据append到数据区
- ② 修改本地元数据副本
- ③ 元数据操作附加到对应的log
- ④ 满一个Unit, clflush, 完成后更新版本号

读流程：

- ① 读前比较元数据版本
- ② 若版本落后, apply元数据操作到本地
- ③ 根据最新元数据访问数据, 由于log-structured新的/修改过的数据在新地址, 无需刷缓存



级联共享，缓解带宽压力，解决单点故障可靠性问题

实验

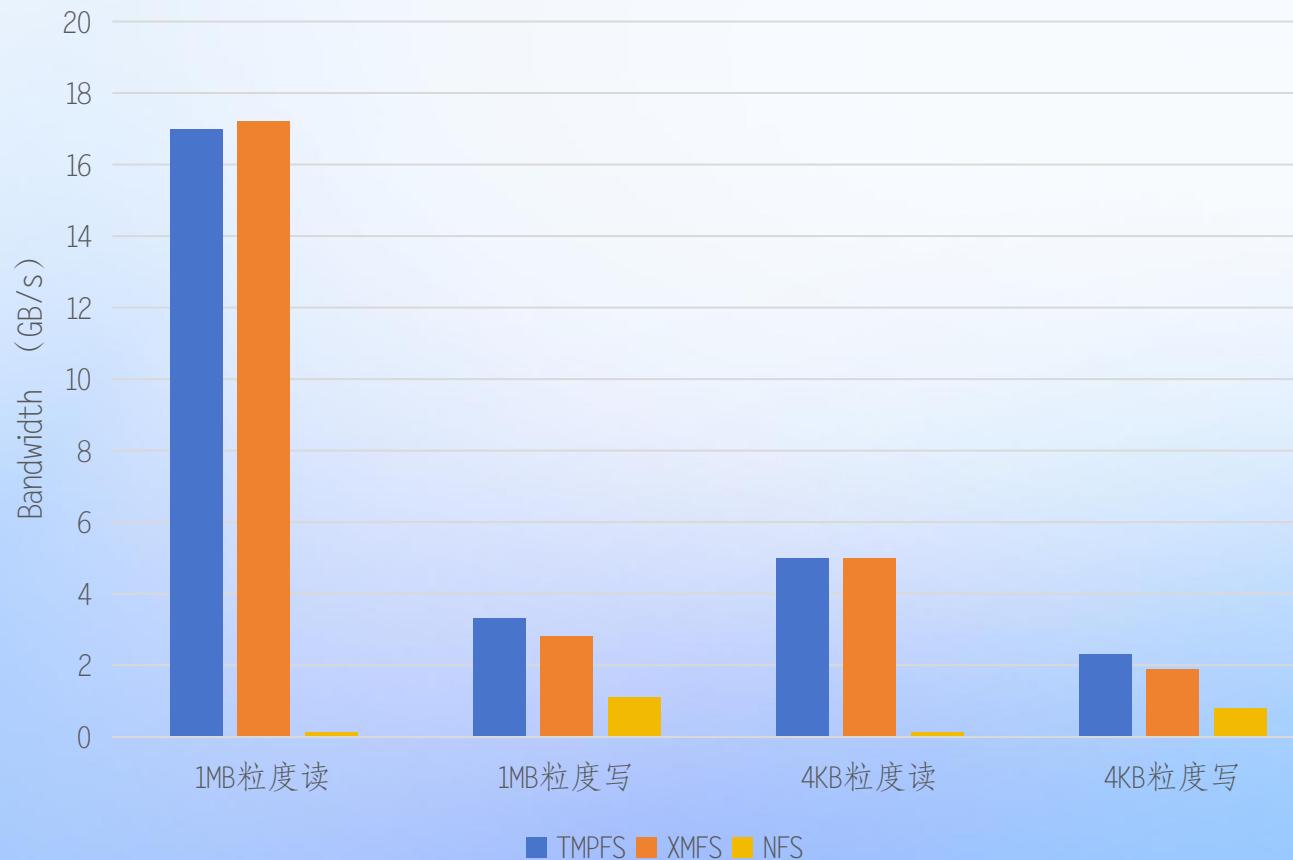
- 鲲鹏+UB互联环境：

- 4P2节点，每节点320个鲲鹏920（2.9GHz），1TB内存

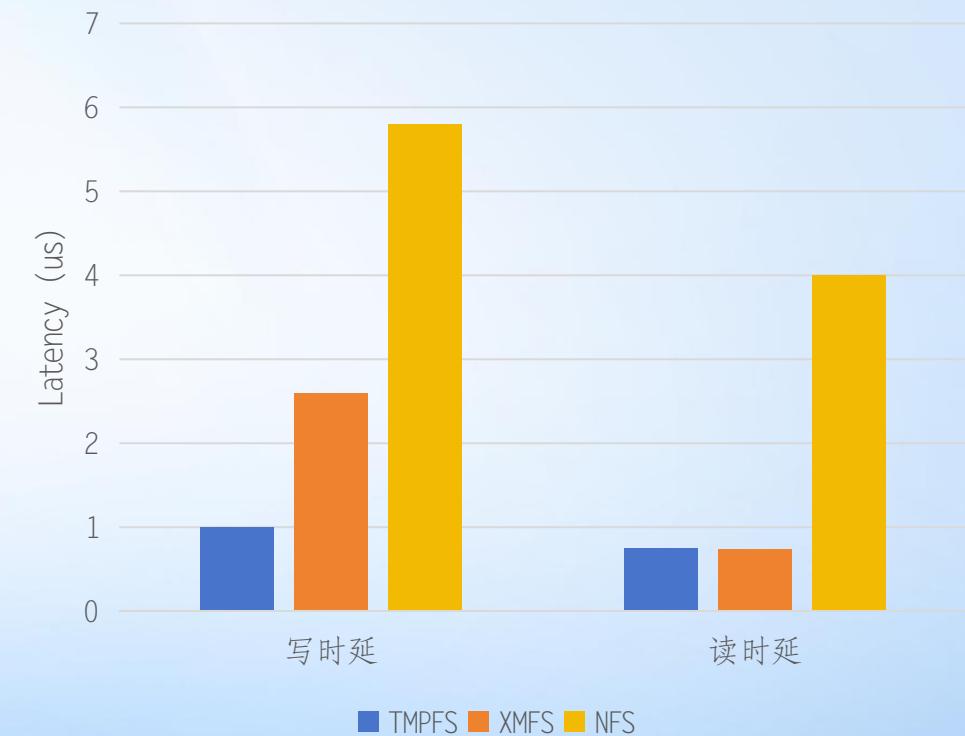
- OS版本：

- OpenEuler 24.03-LTS

基本性能对比



XMFS性能接近本地tmpfs性能，写时延由于日志转换有开销

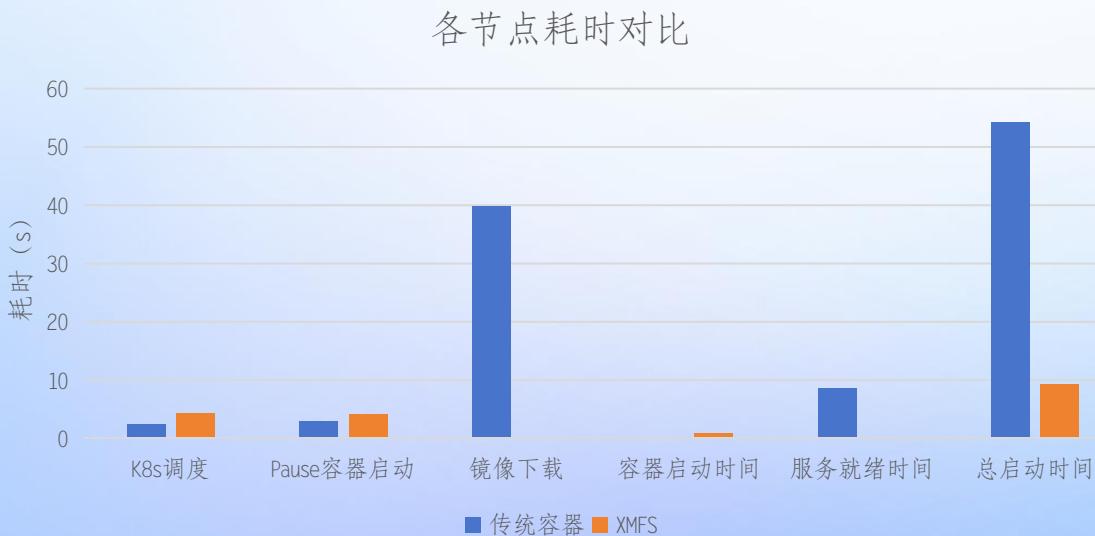


	元数据性能
TMPFS	0.094s
XMFS	0.097s
NFS	2.306s
FAMFS	4.051s

关键业务场景性能

容器冷启动加速：

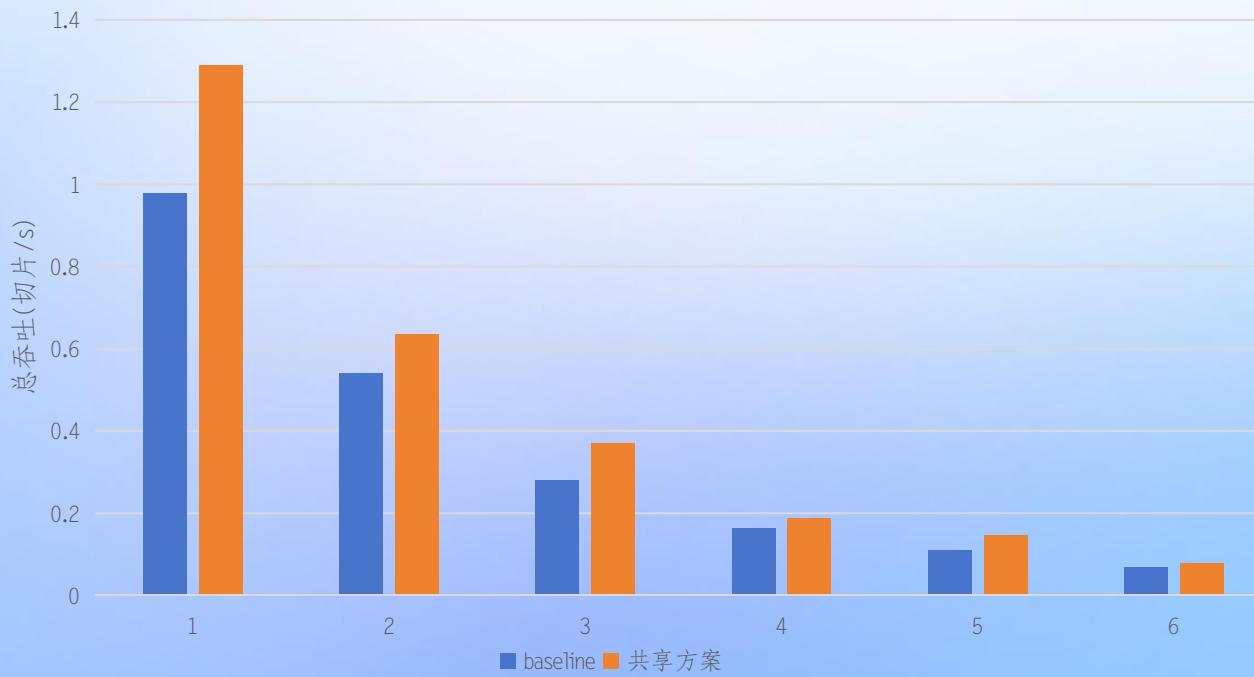
共享elasticsearch镜像855MB + 755MB快照



通过共享省去镜像下载开销，降低服务准备时间；
内存底噪在拉起多容器时，收益更大

关键业务场景性能

- 视频编解码加速：
 - 4P2节点，每节点192核
 - 将10s/30s/60s 切片（H265）编码为多种格式多种分辨率



测试方式：

ú 1/3/5: H265+H264

ú 2/4/6: H265+H264+AV1

Baseline:

ú 切片标准化 worker: 转码 worker =
15:33 / 10:38

共享方案：

ú 每节点1个解码worker，47个编码worker

和baseline相比，1/3/5有30%+提升，2/4/6有15%+的提升

将baseline修改为1个解码+47个编码后，吞吐提升10.26%

未来规划

- 开源整理
- 数据库场景收益
- LMCache 共享后端
 - 比 disk 后端更优的 key 操作性能，读性能和 CPU 相当

Thanks!

关于我们

华为OS内核实验室 HULK团队(杭州): 致力于面向新型存储与互联技术打造高性能IO栈, 成果应用于大模型训推、大规模容器、大数据处理等场景。多个相关工作成果及特性已推向社区。