

更多 GPU ， 怎样才能更快？

第一期 · 扩展概念与数据路径

第一期

01 从通信域开始

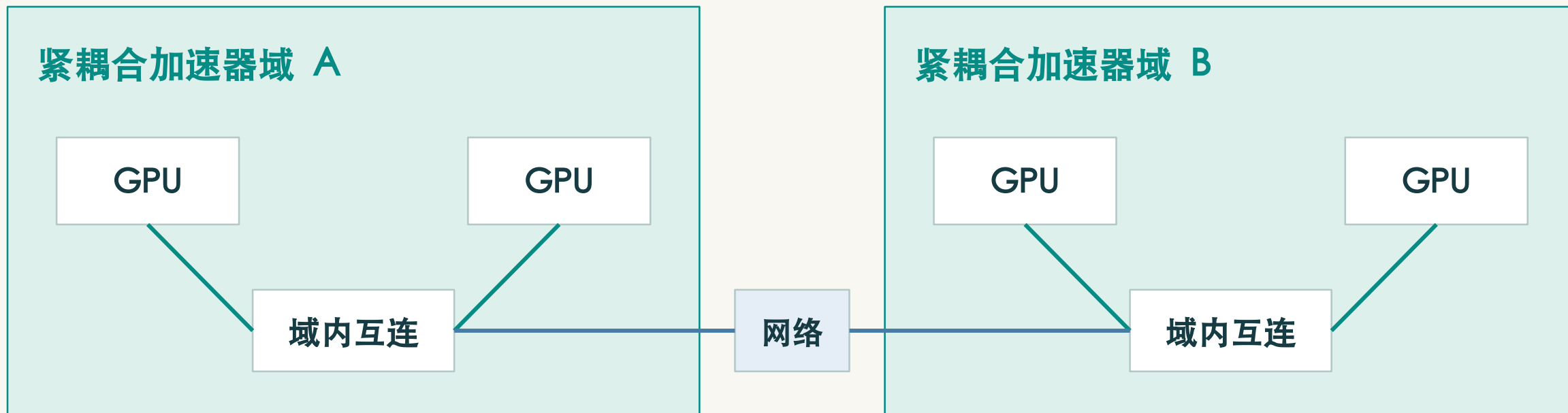
02 沿一笔数据走

03 把带宽还原成时间

先建立模型，再评价方案

先说语境，再说 scale-up / out

本系列采用 AI 加速系统的通信域语境



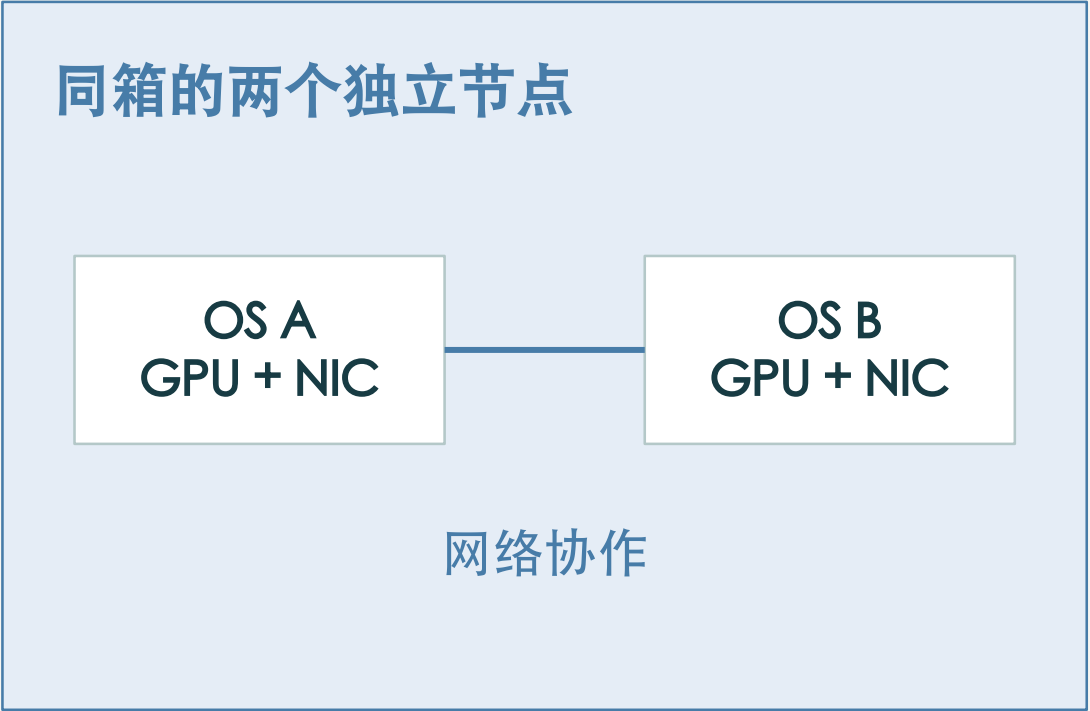
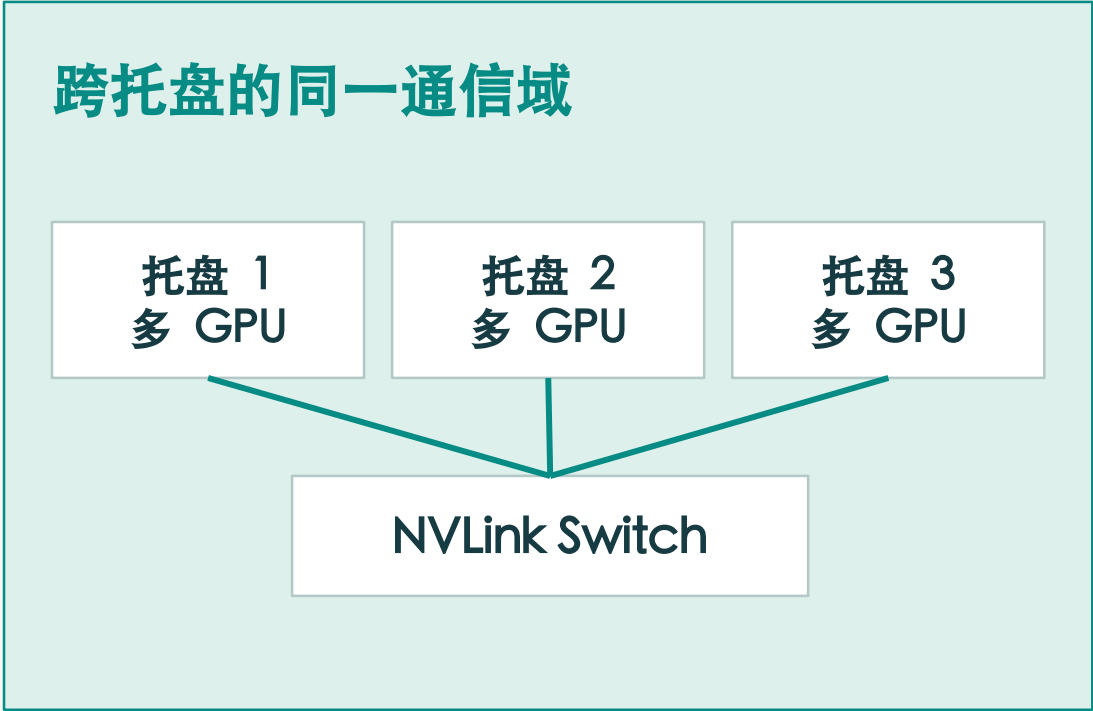
Scale-up : 扩大紧耦合域

Scale-out : 连接多个域

机内 / 机间不是普遍定义

机箱边界，不等于通信域边界

物理封装、操作系统与互连是三个不同维度



先画通信域，再画机械边界

工作负载决定互连压力

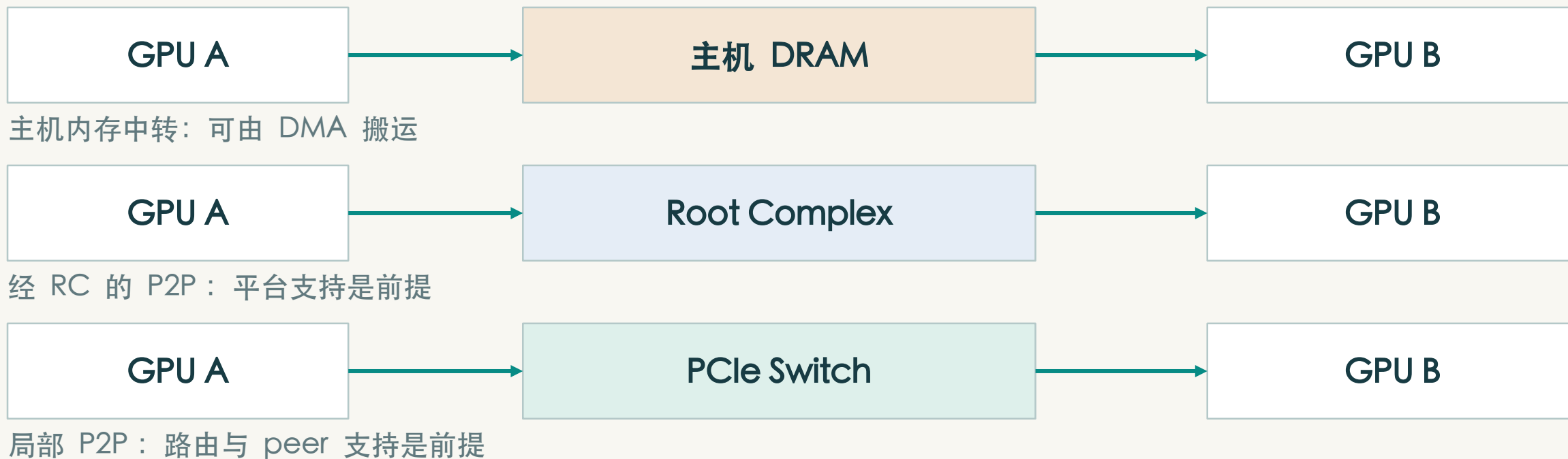
同样是八张卡，通信图可以完全不同

并行方式	典型交换	首先看什么
张量并行 TP	层内结果与归约	频次、小消息延迟
流水线并行 PP	阶段间激活与梯度	阶段均衡、空泡
数据并行 DP	梯度同步	吞吐、计算重叠
专家并行 EP	token 分发与回收	all-to-all 、尾延迟

通信频率 × 消息大小 × 暴露时间

一笔数据，可能走三条路径

“不经过 CPU” 至少有三种不同含义



控制平面存在，不妨碍数据平面旁路

PCIe、NVLink、RDMA：比较层次

物理互连、交换设备与通信机制需要分开

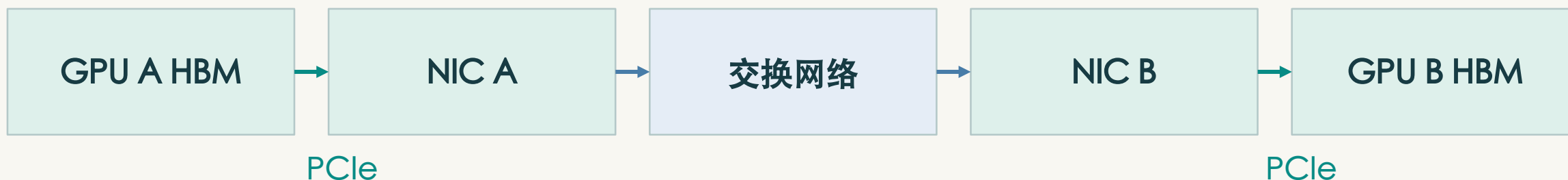
对象	它主要是什么	不能自动推导
PCIe / PCIe Switch	I/O 事务互连 / 转发设备	任意 GPU 可 P2P
NVLink / NVSwitch	GPU 互连 / 域内交换设备	任意 CPU 获得一致性
RDMA / GPUDirect RDMA	远程直接访存 / GPU 内存路径	没有 CPU 控制参与
CCL（例如 NCCL）	集合通信库及算法实现	每次都走同一硬件路径

同一条通信路径可以同时包含 PCIe 与 RDMA

跨域路径： GPU → NIC → 网络

GPUDirect RDMA 的简化数据路径

CPU 控制：资源建立 · 内存注册 · 工作提交 · 异常处理



数据平面可绕过主机 DRAM；主机仍然参与控制

数据直达 GPU，仍需资源管理与同步

带宽先统一口径

下面均为每个接口的单向口径；实际载荷还要扣除开销

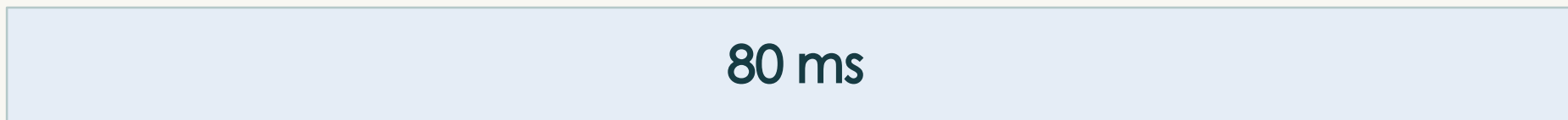
接口示例	单向速率口径	解读方式
PCIe 4.0 ×16	约 31.5 GB/s	扣 128/130 编码，未扣 TLP 开销
PCIe 5.0 ×16	约 63.0 GB/s	扣 128/130 编码，未扣 TLP 开销
PCIe 6.0 ×16	128 GB/s 原始量级	FLIT / FEC 机制，另扣开销
400 Gb/s 网络端口	50 GB/s 线速量级	8 bit = 1 byte ， 另扣协议开销

方向、聚合范围、协议开销、实际配置

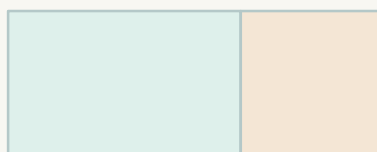
真正的目标：缩短一次任务时间

教学假设：单卡 80 ms，八卡计算理想均分

单卡



八卡



总计 16 ms

计算 10 ms + 暴露通信 6 ms（条长按比例）

$$80 \div 16 = 5\times$$

$$5 \div 8 = 62.5\%$$

只把暴露在关键路径上的通信计入等待

现在，可以带着五个问题读原文

从作者观察走向可重复的系统判断

- 1 任务：哪个模型、哪种并行、什么消息分布？

- 2 域：哪些 GPU 属于同一个通信域？

- 3 路径：显存、交换芯片、RC、NIC 各在哪里？

- 4 能力：设备与驱动真正支持什么？

- 5 证据：版本、配置、正确性与测量能否复现？

下一期：交换芯片能解决什么，不能承诺什么