大模型系列 - 集合通信

ZOMI

集合通信概览

大模型业务全流程

数据 & 模型算法

2. 数据处理

- 开源数据
- 数据预处理
- 向量数据库

3. 模型算法

- LLM模型架构
- 多模态一切皆Tokens

模型训练 & 微调

4. 模型训练

- 混合精度
- 梯度检查
- 梯度累积
- …
- 5. 分布式并行
- 训练集群稳定性

6. 模型微调

- 全参微调
- 低参微调
- 指令微调

模型验证 & 推理部署

7. 模型验证

- 下游任务
- 测评标准
- bench mark

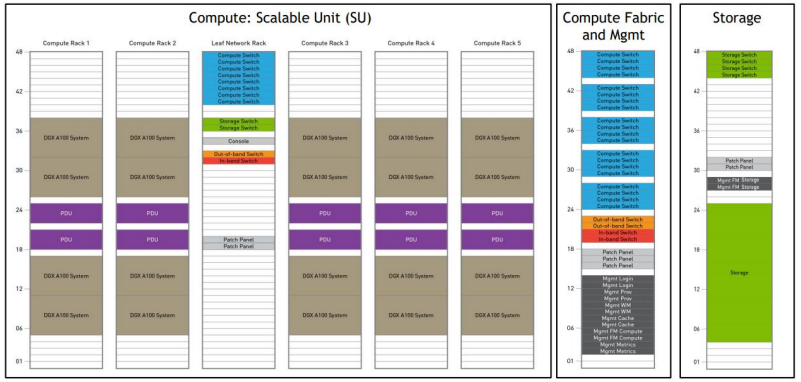
8. 推理与智能体

- 量化压缩
- 推理加速
- 9. Agent 智能体

1. AI 集群建设

- 计算、存储、网络
- AI 集群机房建设
- AI 集群上线与运维

集群算力准备



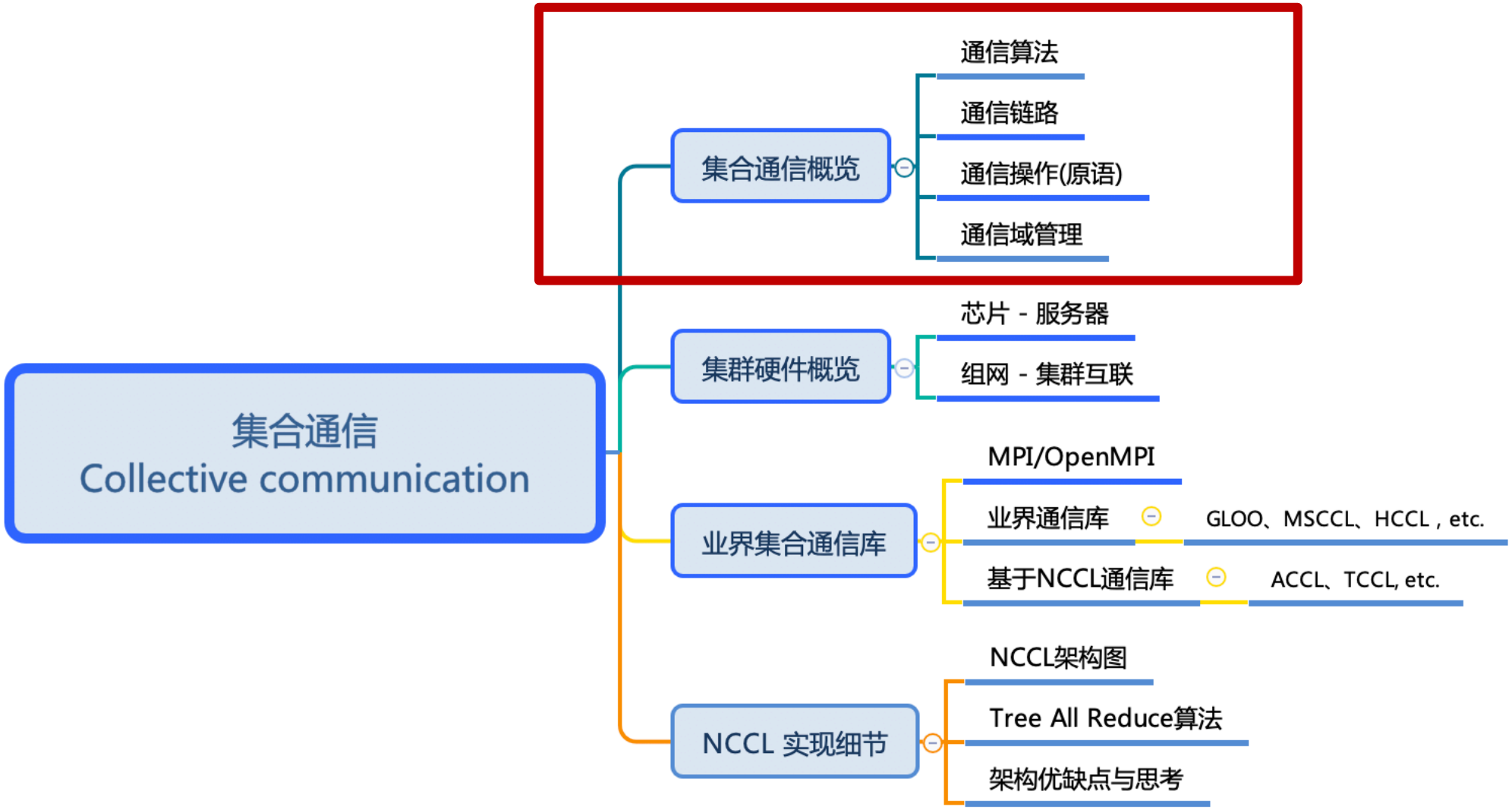
谁应该了解集合通信？

- 系统工程师：to understand how design choices will affect the performance of AI Training:
  1. PCI Topology
  2. Switch Support
  3. Network technology and topology

- 开发者：need inter-NPU or AI Cluster communication for their algorithms or application.

- 使用者：to know what performance and scalability to expect from a given platform

思维导图 XMind


集合通信
Collective communication
集合通信概览
通信算法
通信链路
通信操作(原语)
通信域管理
集群硬件概览
芯片 - 服务器
组网 - 集群互联
业界集合通信库
MPI/OpenMPI
业界通信库
GLOO、MSCCL、HCCL，etc.
基于NCCL通信库
ACCL、TCCL, etc.
NCCL 实现细节
NCCL架构图
Tree All Reduce算法
架构优缺点与思考

集合通信概览

1. AI 与通信关系（AI 基础知识、训练推理、分布式并行）
2. XCCL 基本架构（HPC 通信架构 to XCCL 通信架构）
3. 集合通信原语（All Reduce, etc.）
4. 集合网络拓扑（Hypercube 、 Ring 、 Torus 、 Fat-Tree 、 Dragonfly & Dragonfly+）
5. PyTorch 集合通信与计算并行

XCCL: XXXX Collective Communication Library

集合通信概览：基本概念

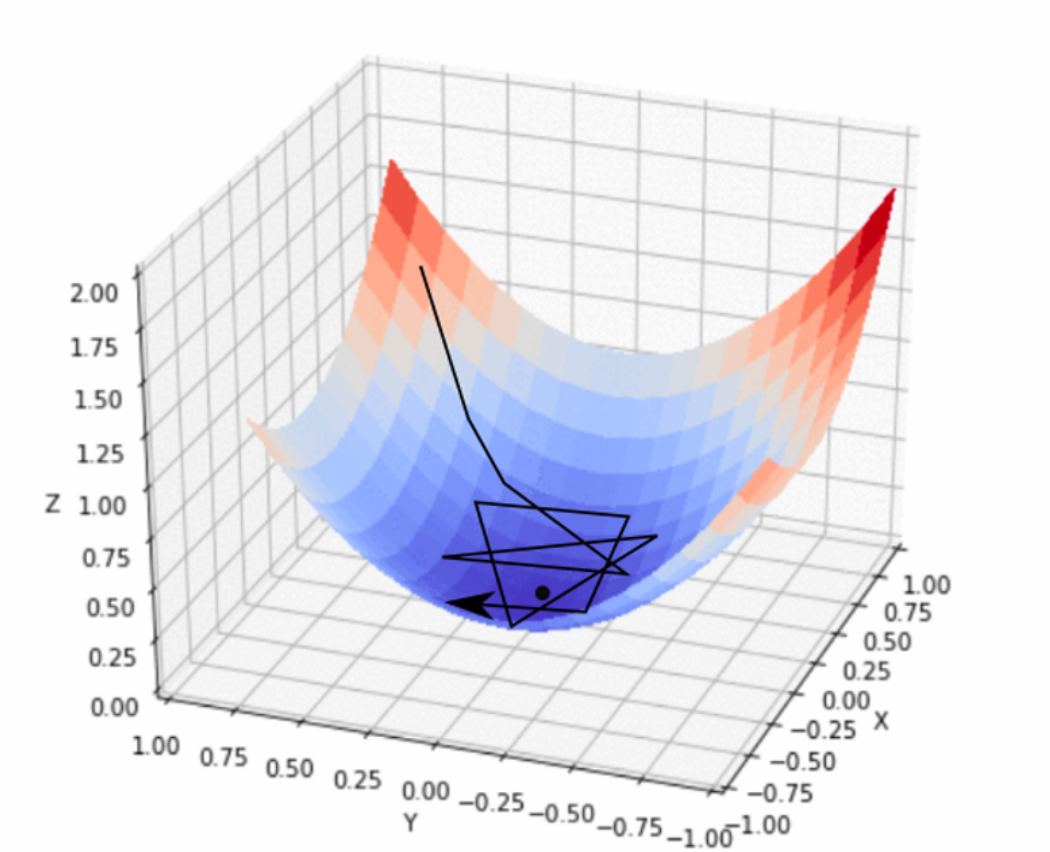
通信特性	HCCL	NCCL
通信算法	ring/mesh + ring/Hav-Doub/Pair-Wise，etc.	ring + Tree ring，etc.
通信链路	NIC / 灵渠总线 / PCIE	NIC / NVLink / NVSwitch / GPU-Direct / PCIE / CMC
通信操作	allreduce、broadcast、reduce、reduce scatter、allgather、all2all、send、recv	allreduce、broadcast、reduce、reduce scatter、allgather、all2all、send、recv
通信域管理	全局通信域、子通信域、基于全局/子通信域配置算法	全局通信域、子通信域、自定义通信域配置算法

01 基础 AI 知识

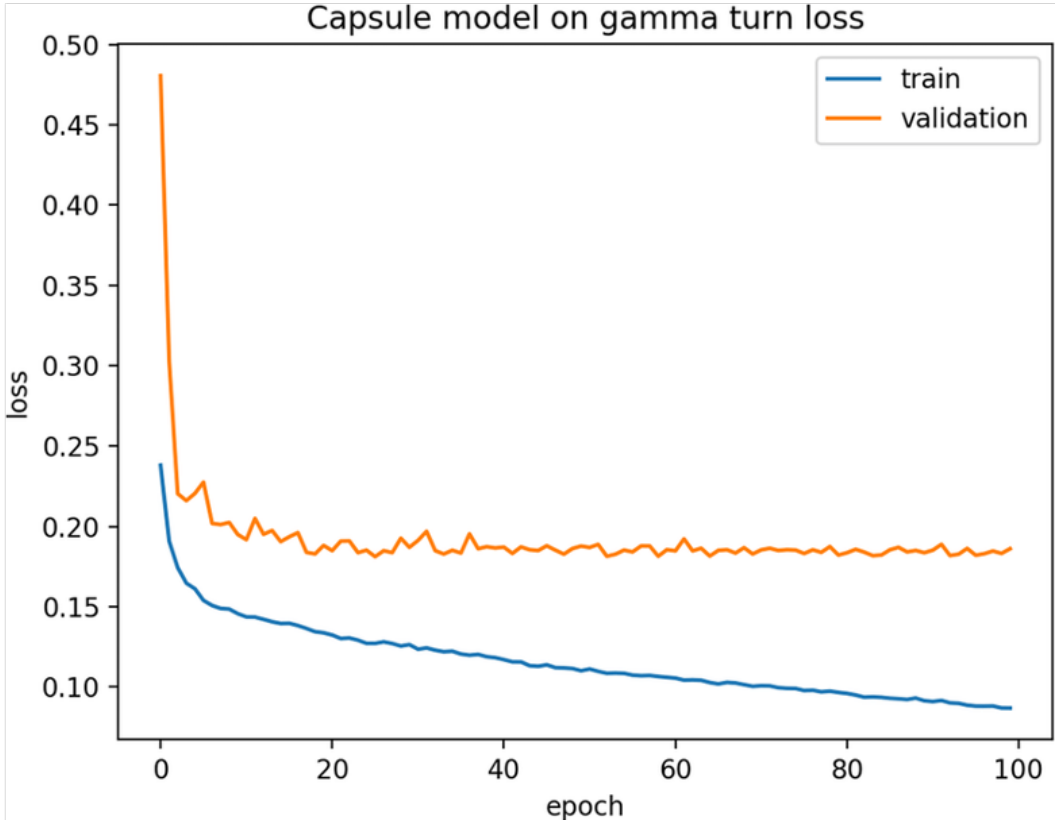
why need XCCL

单卡训练神经网络模型

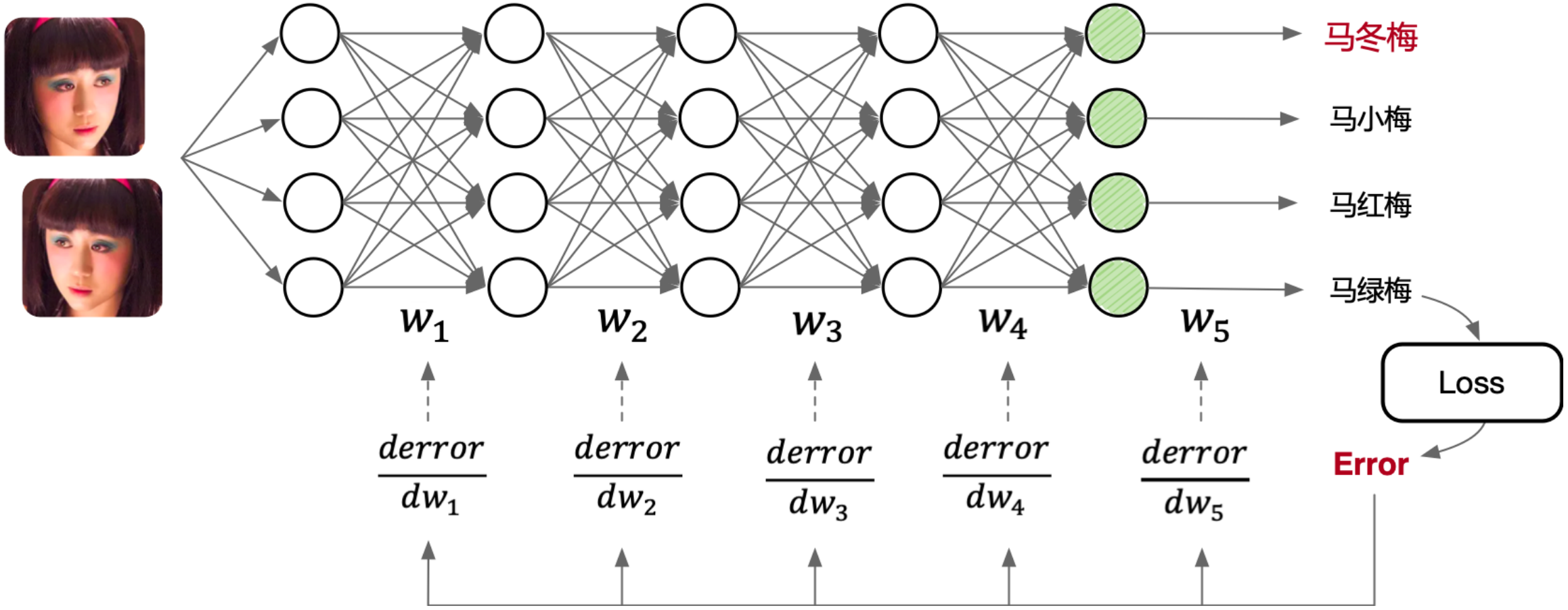
梯度下降算法寻找数据鞍点



损失值在下降

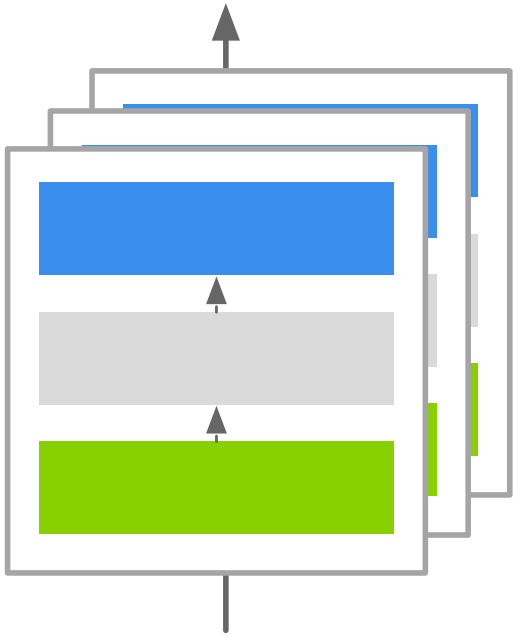


单卡训练神经网络模型


马冬梅
马小梅
马红梅
马绿梅
w_1
w_2
w_3
w_4
w_5
Loss
Error
$\frac{derror}{dw_1}$
$\frac{derror}{dw_2}$
$\frac{derror}{dw_3}$
$\frac{derror}{dw_4}$
$\frac{derror}{dw_5}$

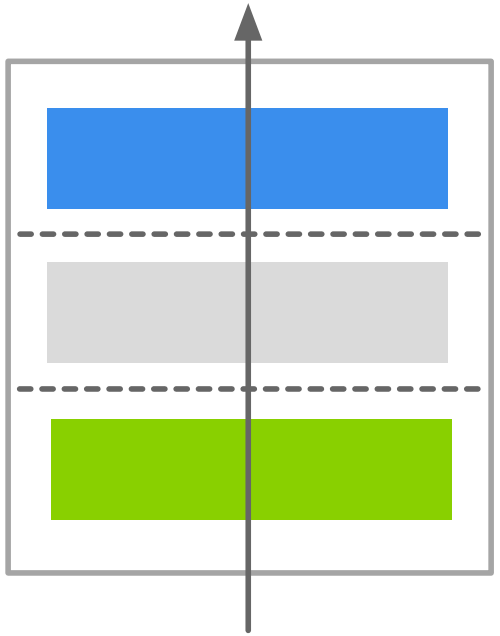
分布式训练

Data Parallelism



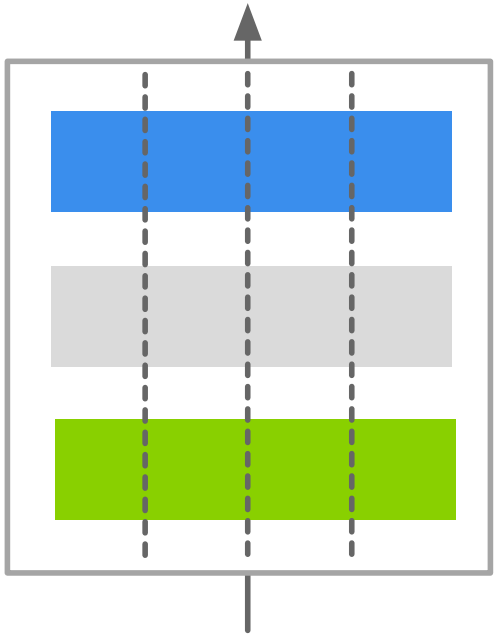
将数据分为若干份，分别映射到不同的 NPU

Pipeline Parallelism



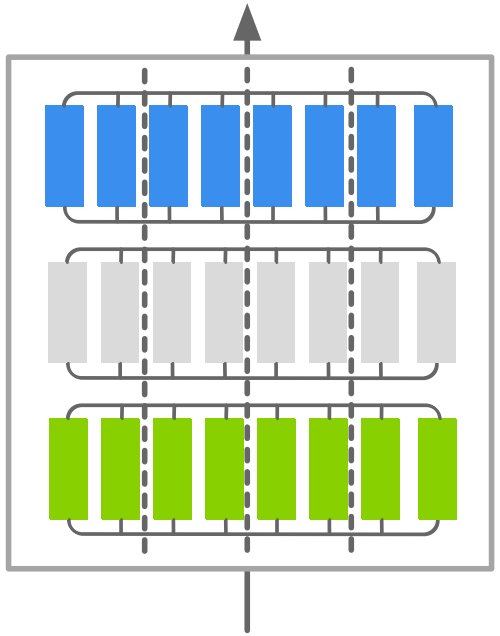
将一个网络拆分为多个流水 Stage 在不同 NPU

Tensor Parallelism



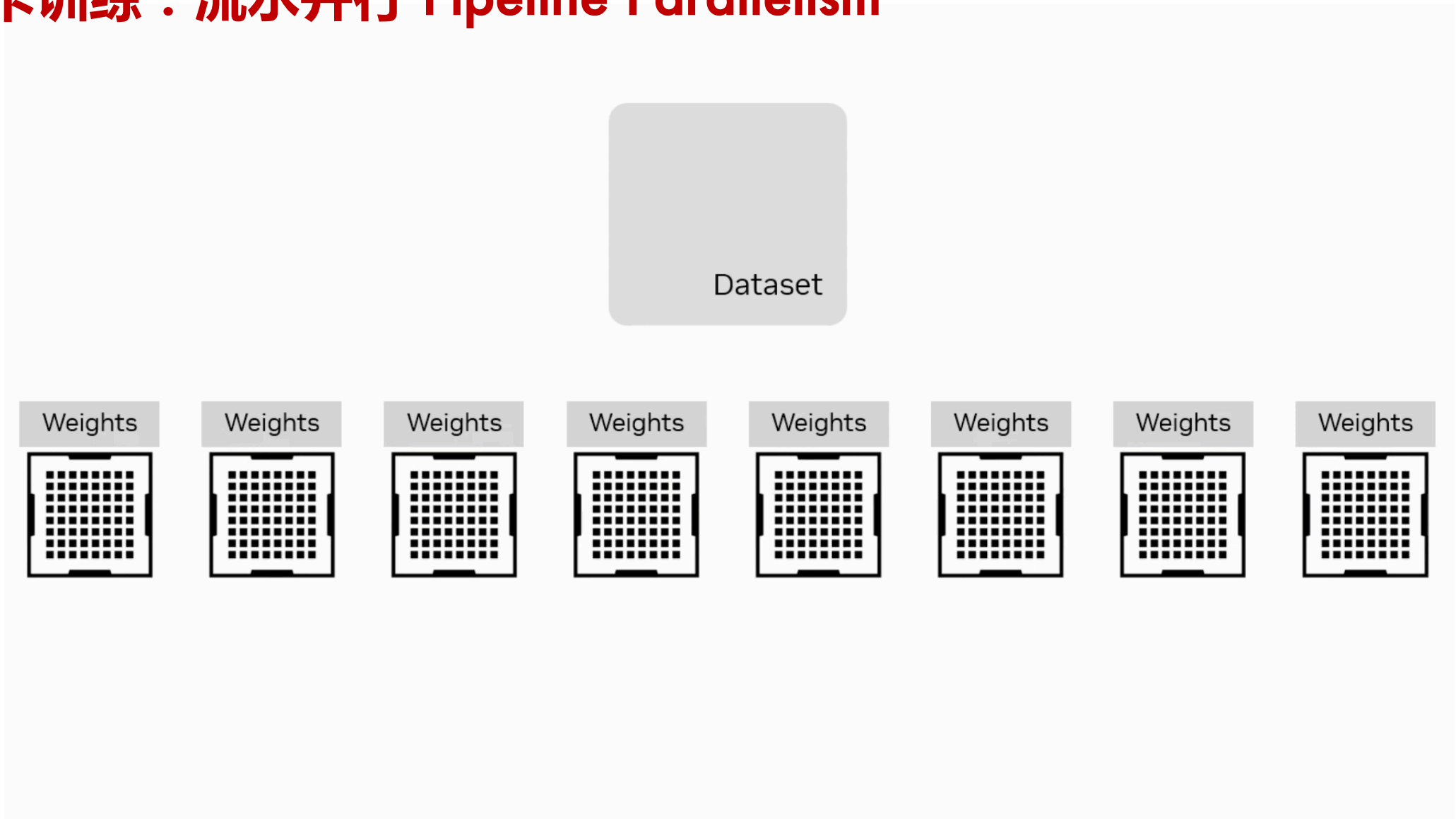
将模型层内的参数切分到不同 NPU

Expert Parallelism

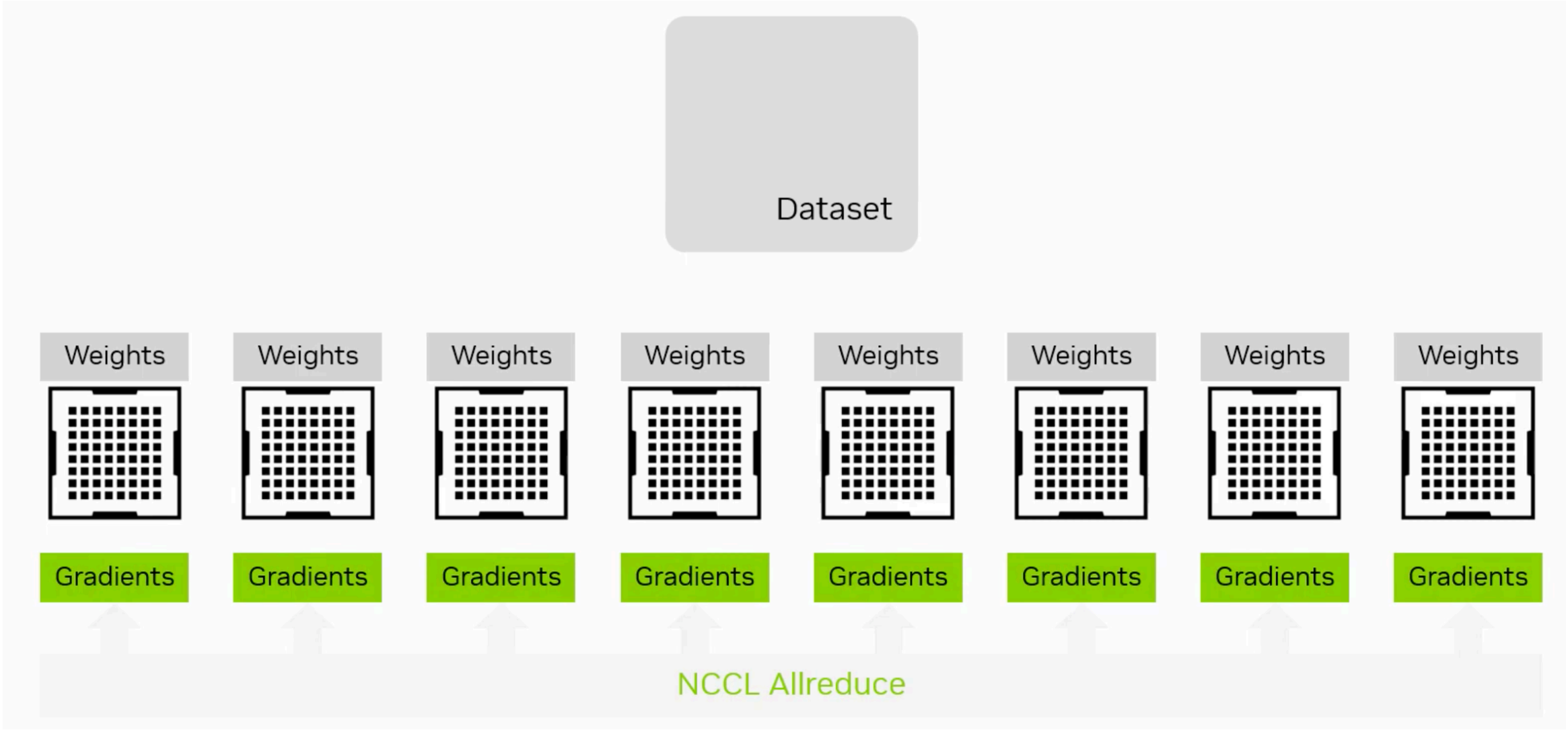


专家放置在不同 NPU，处理不同 Batch 样本

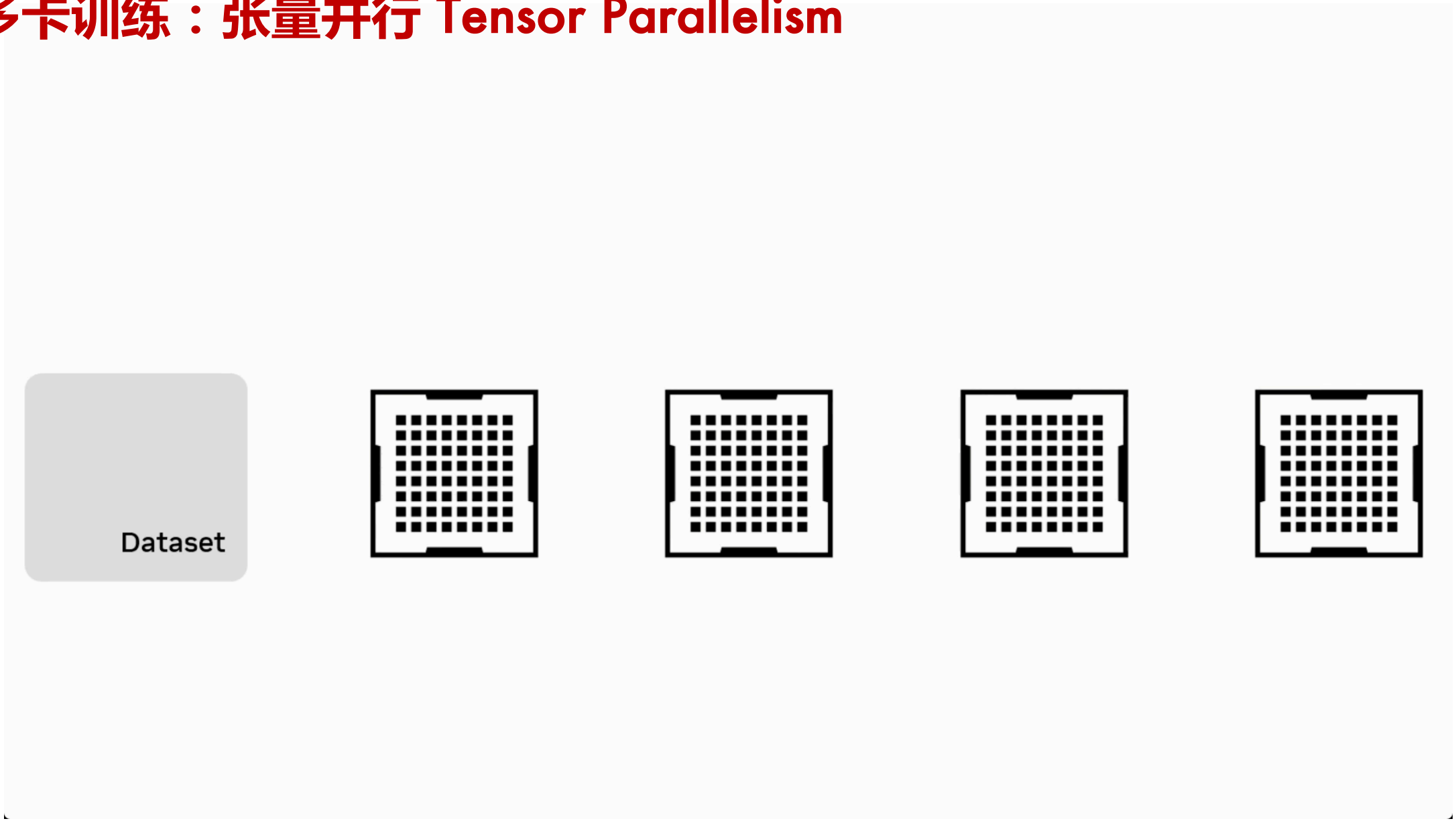
多卡训练：流水并行 Pipeline Parallelism



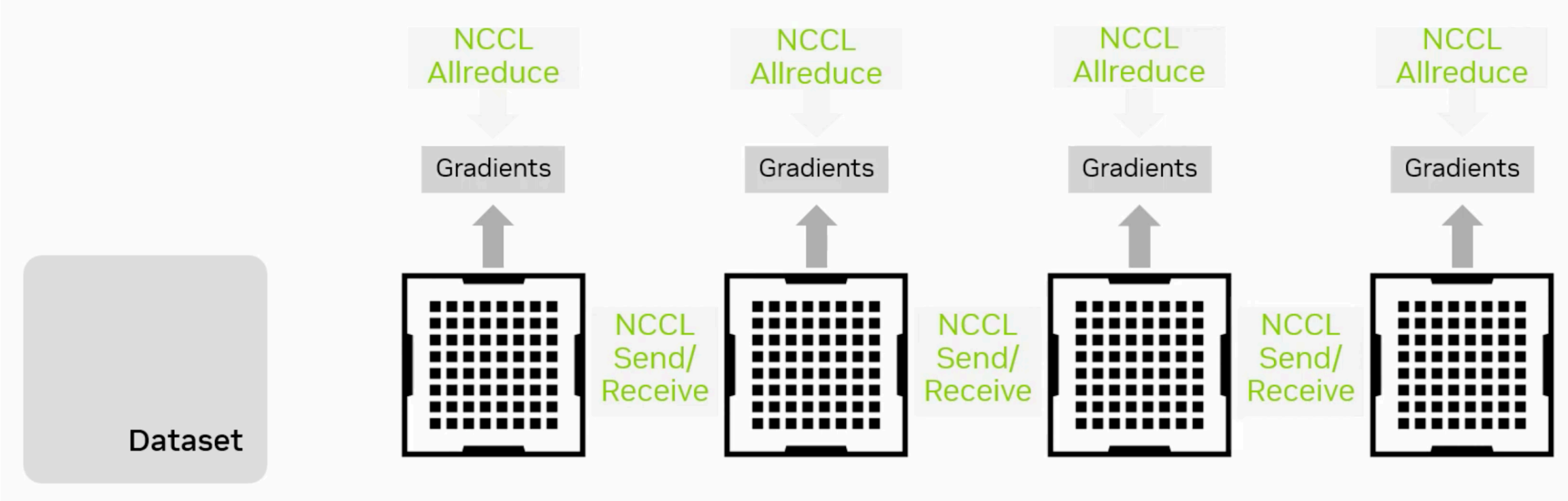
多卡训练：数据并行 Data Parallelism



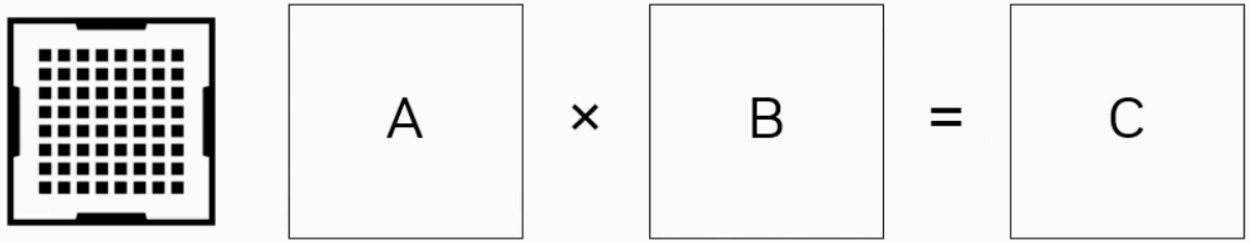
多卡训练：张量并行 Tensor Parallelism



多卡训练：流水并行 Pipeline Parallelism

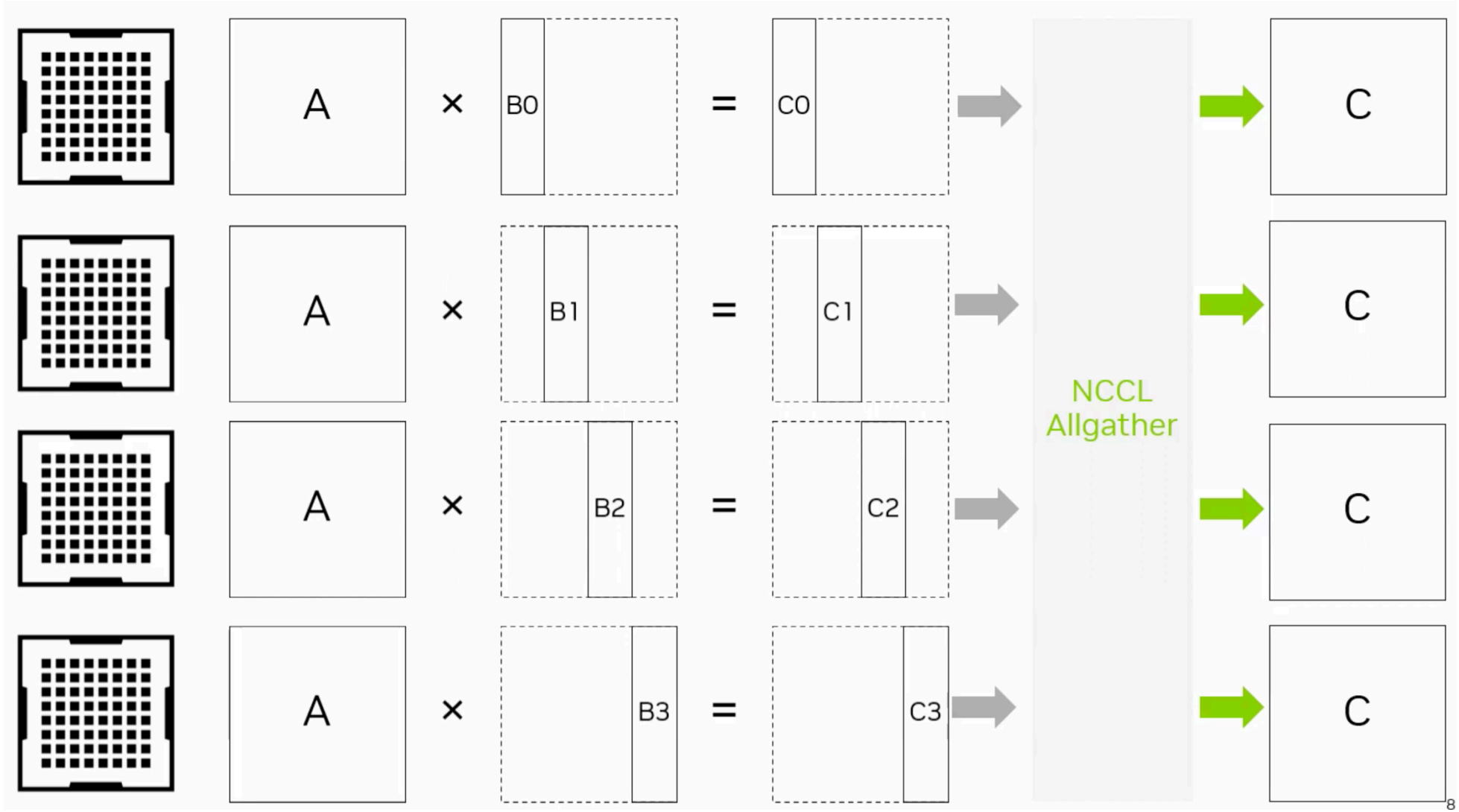


多卡训练：张量并行 Tensor Parallelism



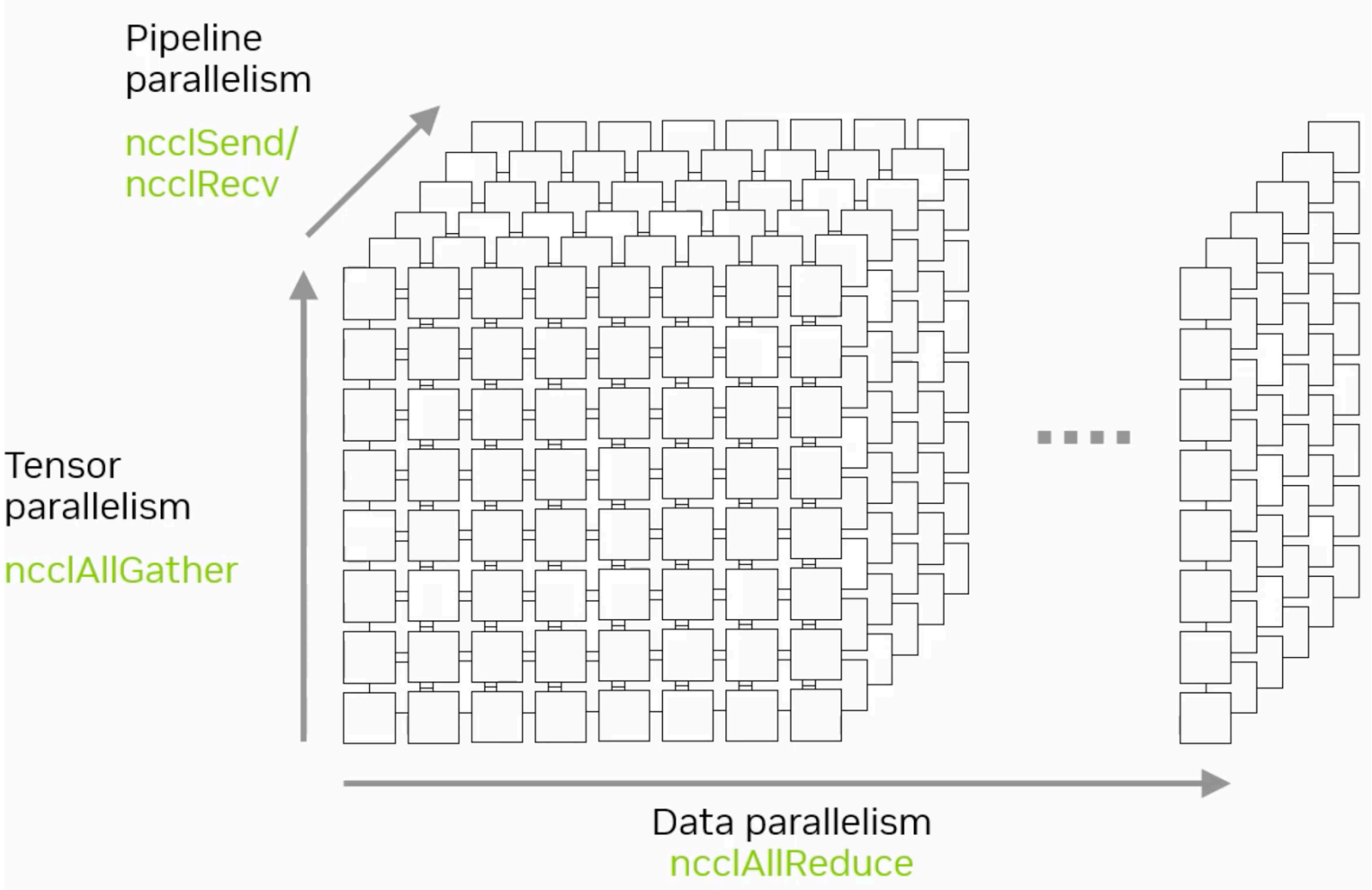
8

多卡训练：张量并行 Tensor Parallelism



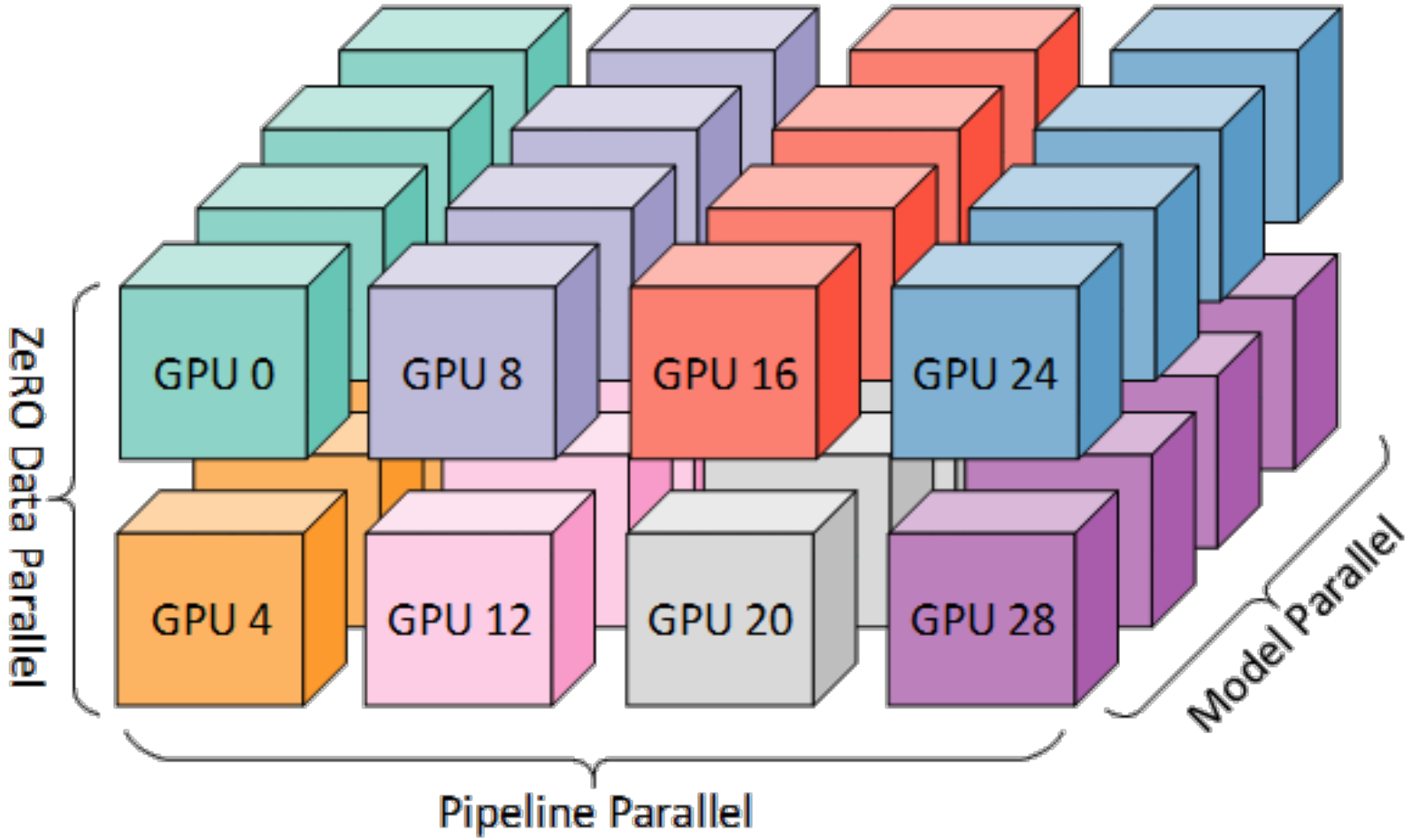
多卡训练：多维并行 Multi Parallelism

多卡训练：多维并行 Multi Parallelism

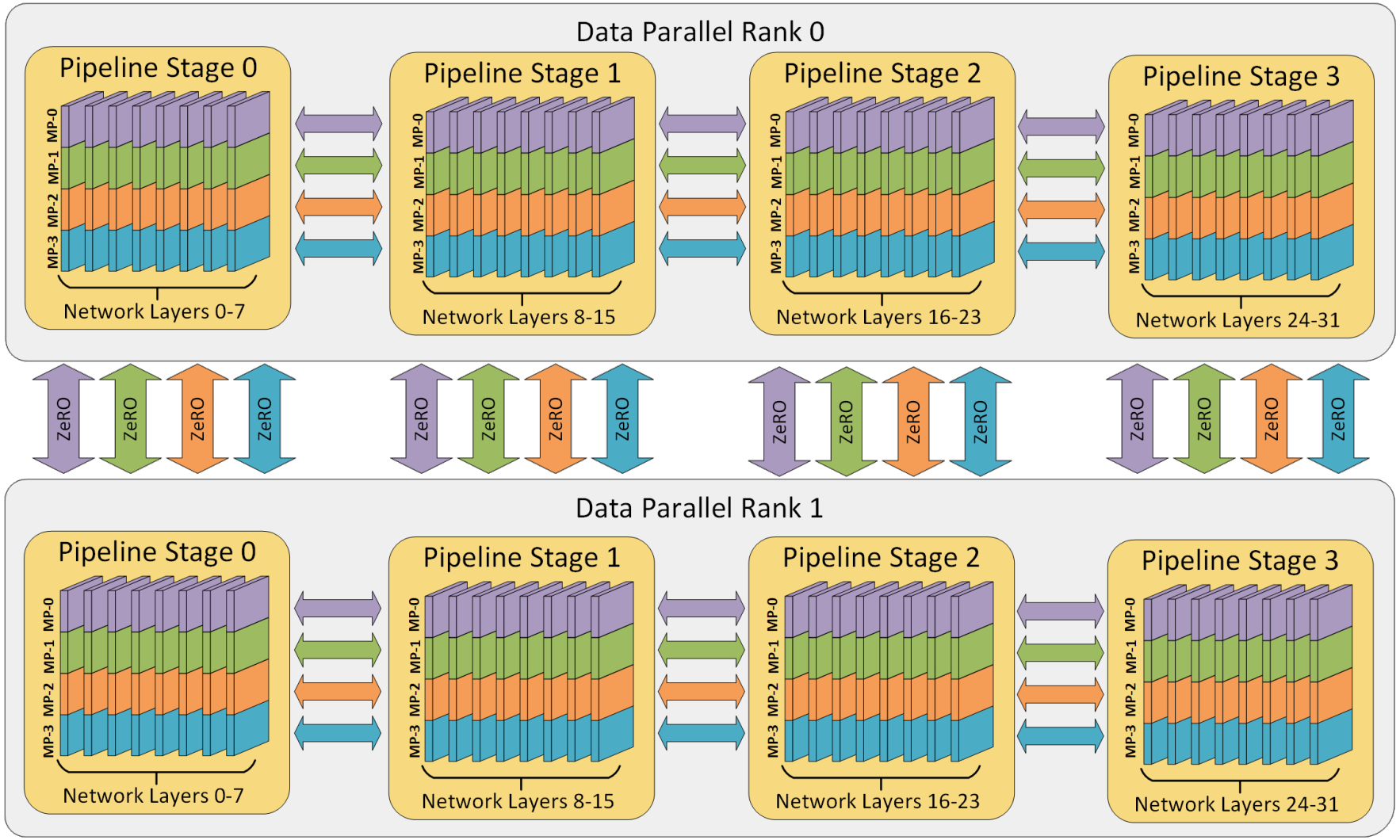


- MoE(Mixture of Experts)
  - Send/Recv(all2all)
- FSDP(Full Sharded DP)
  - AllGather
- Long Sequence
  - AllGather/AllReduce

多卡训练：多维并行 Multi Parallelism



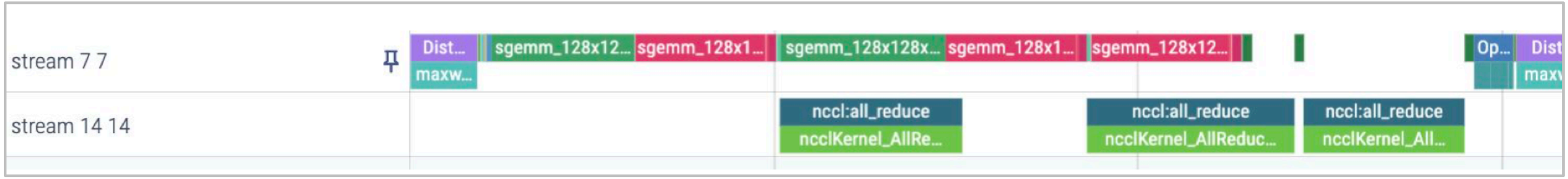
多卡训练：多维并行 Multi Parallelism



02 XCCL 基本架构

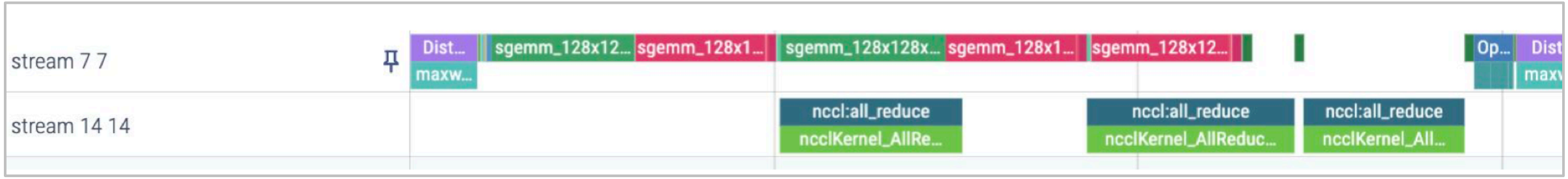
计算与通信解耦

- 神经网络训练过程中，每一层神经网络都会计算出一个梯度 Grad，如果反向传播得到一个梯度马上调用集合通信 AllReduce 进行梯度规约，在集群中将计算与通信同步串行，那么集群利用率（MFU）性能就很差。
  - e.g.，如 GPT3 176B 有 96 层 Transformers 对应 Grad 个数 96×12，设计算梯度 1ms，通信 500ms，每次时间 501ms，总体需要 $501\times 96 \times 12 = 577152ms$，近 577s 完成一次梯度迭代。



计算与通信解耦

- 将计算与通信解耦，计算的归计算，通信的归通信，通过性能优化策略减少通信的次数：
  - 提升集群训练性能（模型利用率 MFU/算力利用率 MFU）；
  - 避免通信与计算假死锁（计算耗时长，通信长期等待）；



计算与通信解耦：分布式加速库

- **分布式加速库**：解耦计算和通信，分别提供计算、通信、内存、并行策略的优化方案。
  - DeepSpeed、Megatron-LM、MindSpeed、ColossalAI

大模型是怎么训起来的？分布式并行框架介绍 #大模型 #分布式并行 #

1.1万 4–3

分布式并行框架DeepSpeed介绍 #大模型 #分布式并行 #训练

8643 4–6

DeepSpeed优化器并行ZeRO1/2/3原理 #大模型 #分布式并行 #训练

3180 5–10

分布式训练框架Megatron-LM代码概览 #大模型 #分布式并行 #训练

2861 5–11

分布式PTD多维并行与GPU集群关系 #大模型 #分布式并行 #分布式训

1960 5–11

Megatron-LM张量并行的行切分和列切分核心原理 #大模型 #分布式

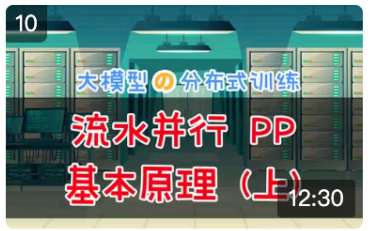
2425 5–14

Megatron-LM 张量并行 TP 代码剖析 #大模型 #分布式并行 #分布式训

1495 5–28

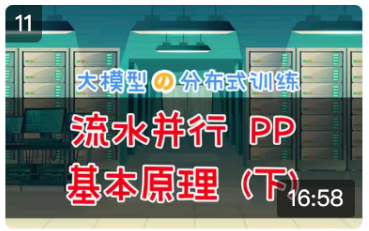
Megatron-LM 序列并行 SP 代码剖析 #大模型 #分布式并行 #分布式训

1474 5–29



流水并行 PP 基本原理(Gpipe原理) #大模型 #分布式并行 #分布式训练

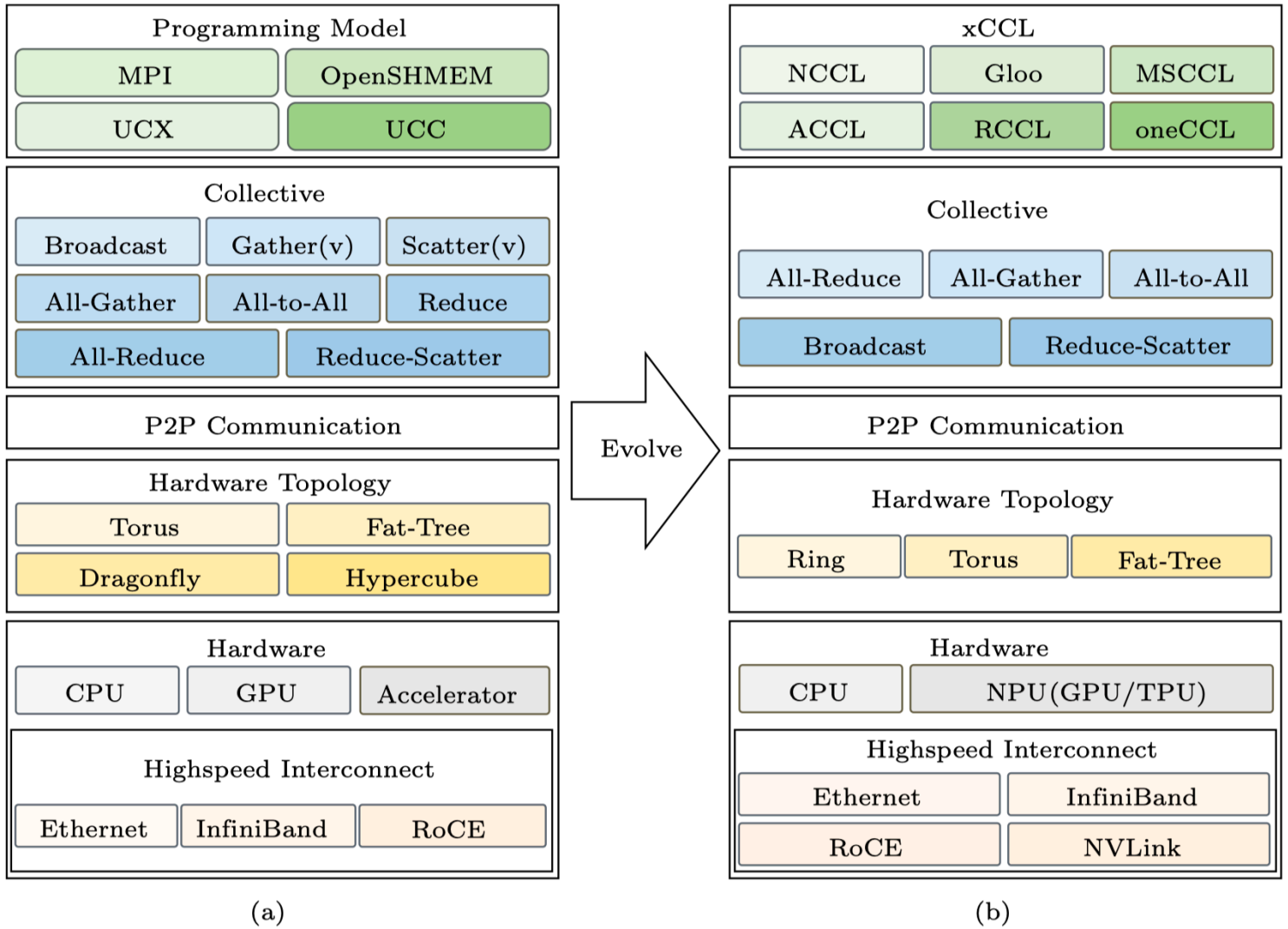
1481 5–30



流水并行 PP 基本原理(1F1B、1F1B Interleaved原理) #大模型 #分布式

1308 5–31

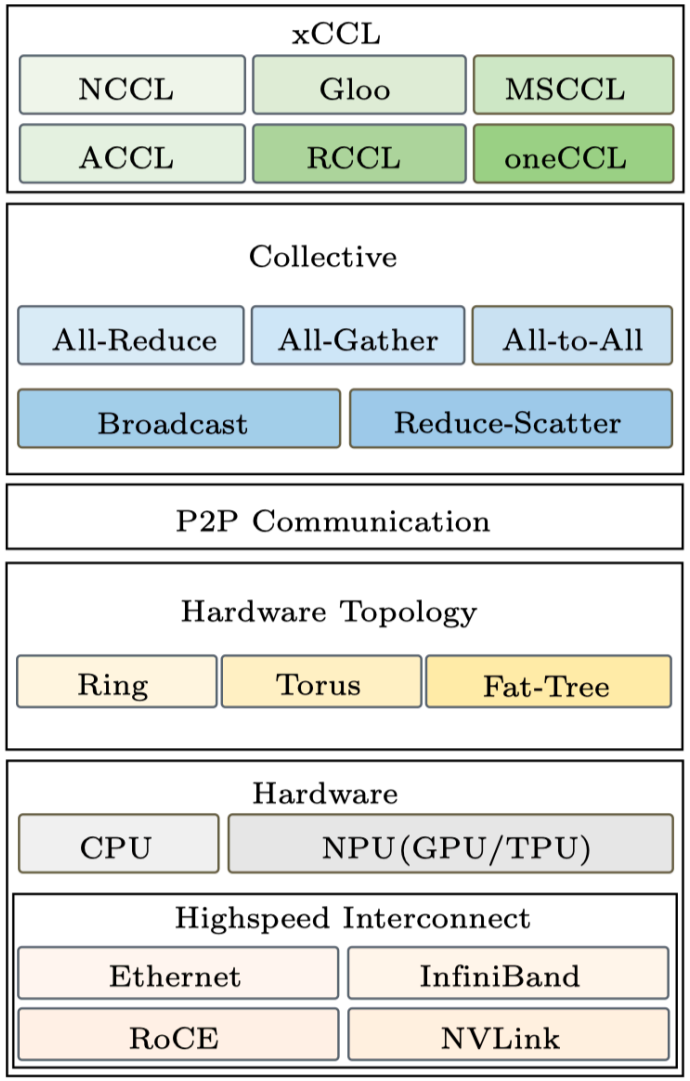
HPC 到 AI 通信栈基本架构

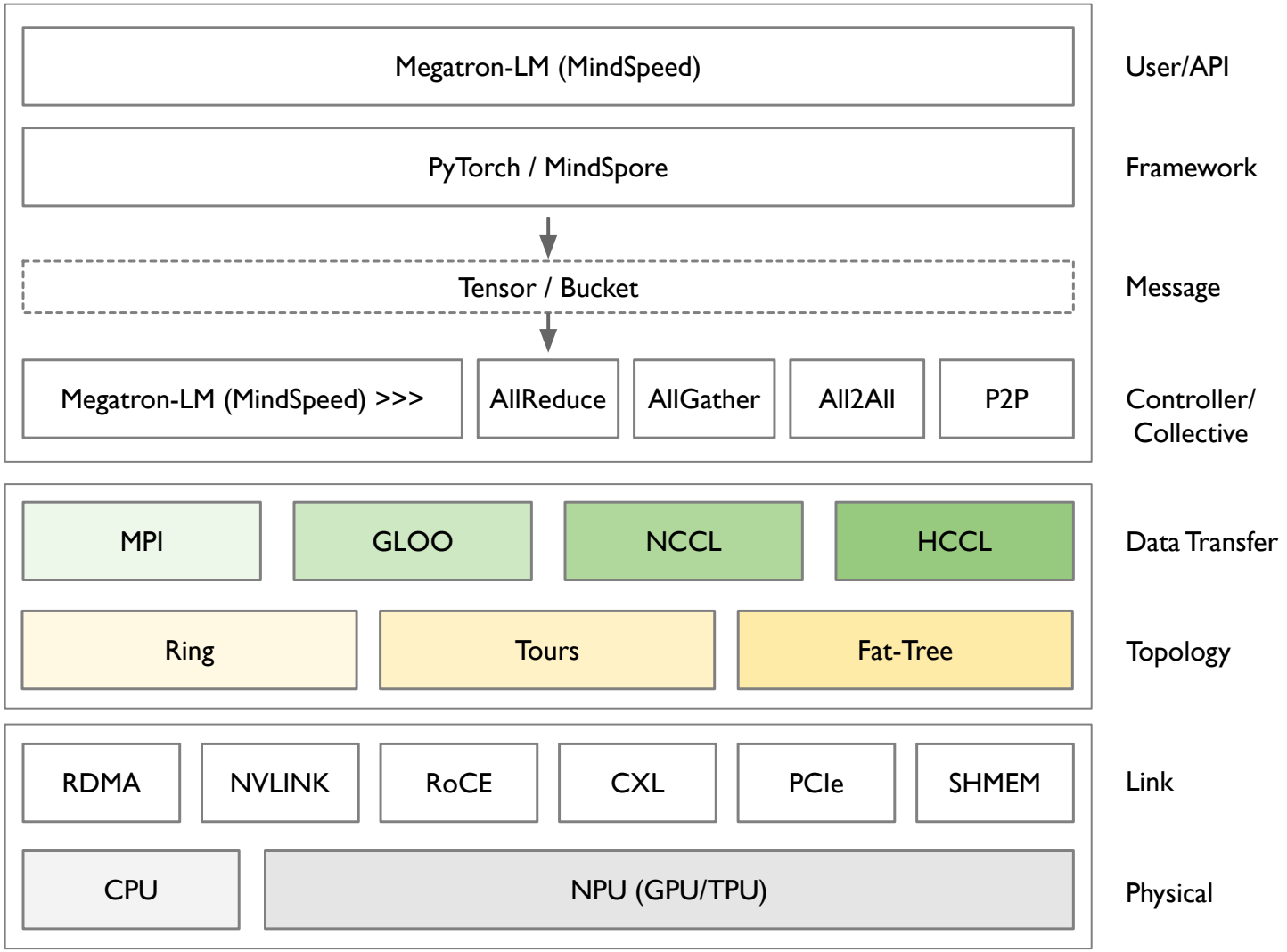


- Overview of collective com munication evolution.
  a) Classic HPC scenarios.
  b) Emerging AI scenarios.

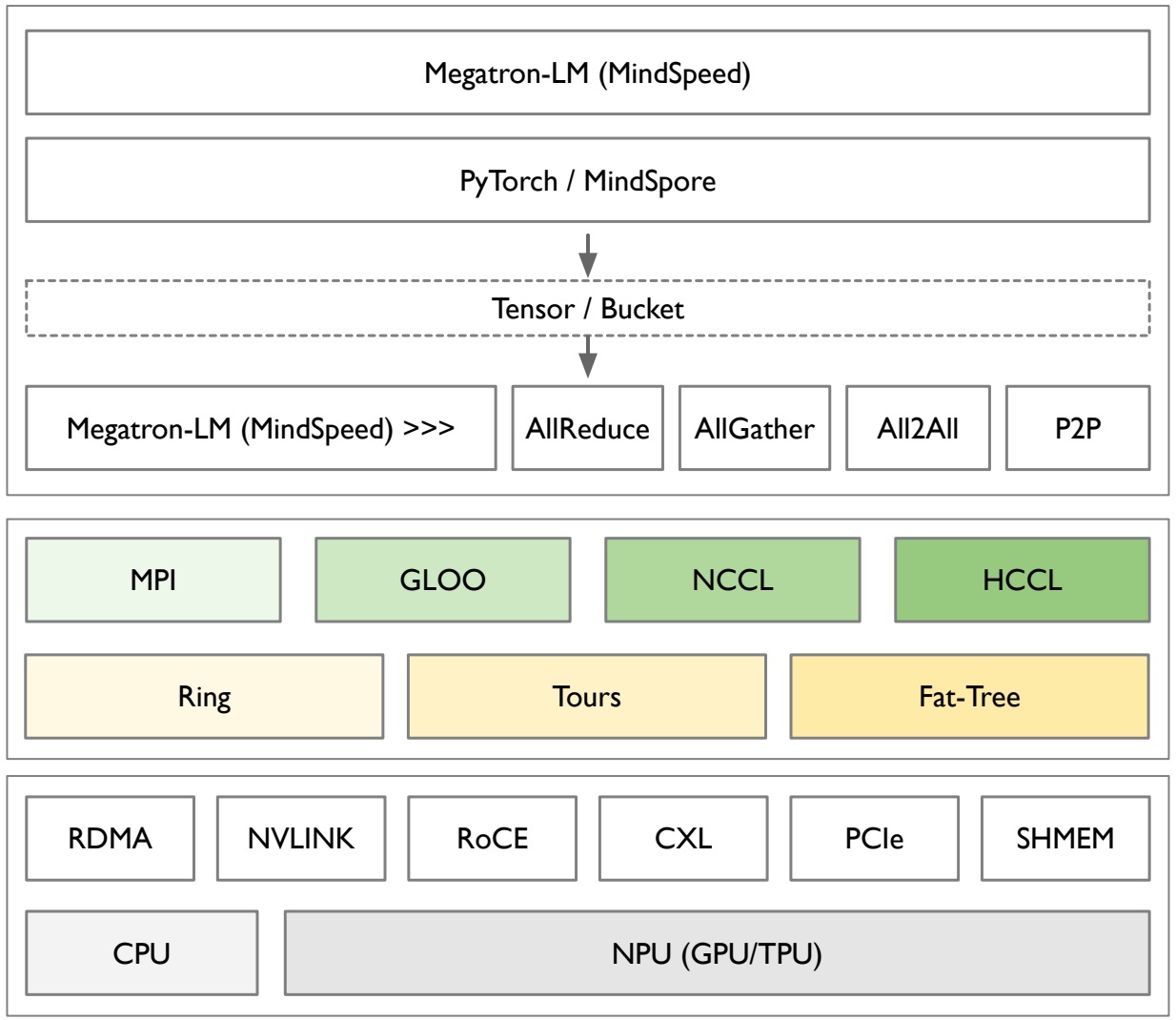
xCCL: A Survey of Industry-Led Collective Communic ation Libraries for Deep Learning

XCCL 在 AI 系统中的位置





XCCL 在 AI 系统中的位置



- Megatron-LM/MindSpeed 分布式加速框架解耦了计算与通信：
  1. 计算主要通过 PyTorch 等AI框架执行；
  2. 通信通过 XCCL 通信库来执行；

- 将框架计算出来 Tensor 记录到 Bucket 中
- 通过控制层在后台启动 loop 线程
- 周期性的从 Bucket 中读取 Tensor
- 控制层在节点之间协商一致后，进行消息分发到具体 NPU 上执行通信

03 小结与思考

小结与思考

了解完本内容后：

1. AI 神经网络模型学习/训练阶段为什么要通信（AI 基础知识、训练推理、分布式并行）
2. XCCL 在 AI 系统中的位置（HPC 通信架构 to XCCL 通信架构）
3. 集合通信原语（All Reduce, etc.）
4. 集合网络拓扑（Hypercube 、 Ring 、 Torus 、 Fat-Tree 、 Dragonfly & Dragonfly+ ）
5. PyTorch 集合通信与计算并行

Thank you

把AI系统带入每个开发者、每个家庭、每个组织，构建万物互联的智能世界

Bring AI System to every person, home and organization for a fully connected, intelligent world.

ZOMI

Course chenzomi12.github.io

GitHub github.com/chenzomi12/AISystem