

Geryon: Accelerating Distributed CNN Training by Network-Level Flow Scheduling

INFOCOM 20

**DML GROUP MEETING
2.18**

OUTLINE

- Introduction
- Background and motivation
- Design
- Evaluation

Introduction

- 分布式机器学习通信与计算之间存在严重的不匹配，导致通信逐渐成为dml的瓶颈（这种趋势随着机器学习模型规模增大越发明显）
- 一种有效的方法：改善计算和通信的重叠，将通信开销隐藏在计算之后
- 一些工作试图通过在终端主机上显式地安排参数传输的顺序来实现计算和通信的重叠
 - 1) TICTAC通过分析模型计算图的关键路径获得接近最优的调度，然后将参数按顺序传递给通信模块
 - 2) P3对参数进行切片并根据消耗顺序为它们分配优先级，最终主机队列中按优先级发送
- 基于终端主机的解决方案存在的问题：**只控制每个终端主机的发送顺序，而忽略了网络的整体情况**
 - 1) 只能对同一节点的数据包进行优先级排序，不能很好地协调来自不同节点的流量
 - 2) 当有多个节点向网络发送流量时，来自一个发送方的低优先级数据包将与来自另一个发送方的高优先级数据包竞争带宽资源，导致优先级调度失去作用
- 改进方案：采用网络级流量调度方案，基于此提出Geryon
- Geryon关键思想：**利用具有不同优先级的多个流来传递不同紧急级别的参数。**对于紧急的参数，以更高的优先级传输到最终主机。高优先级标签可以在整个网络中识别，因此紧急参数的传输可以得到更好的保证

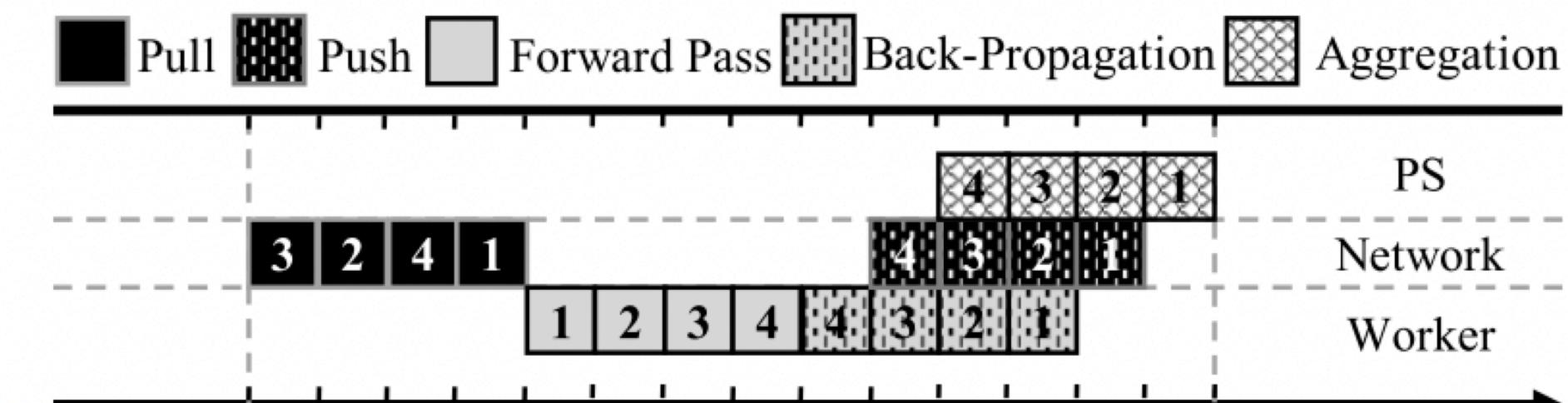
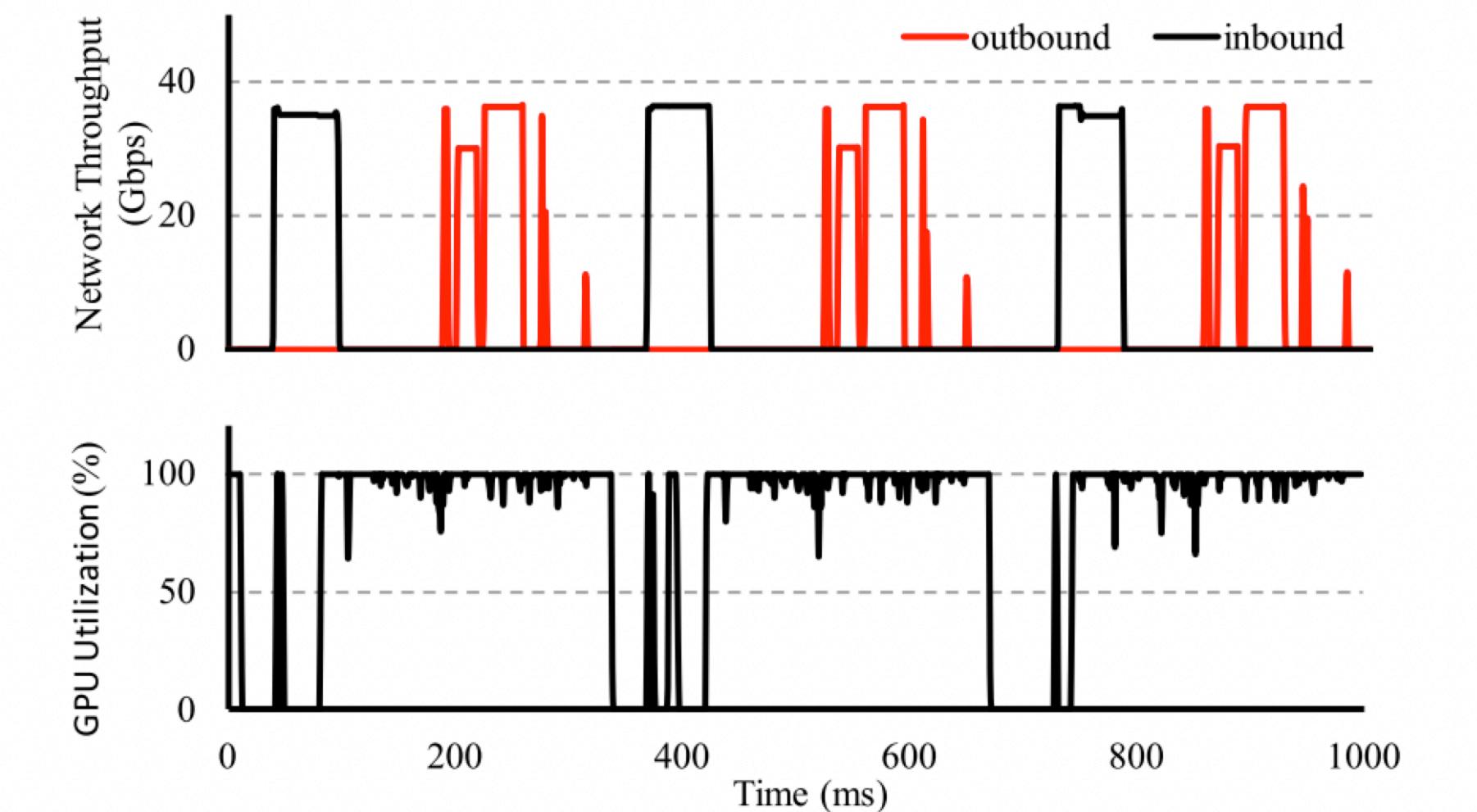
Background and motivation

Distributed CNN

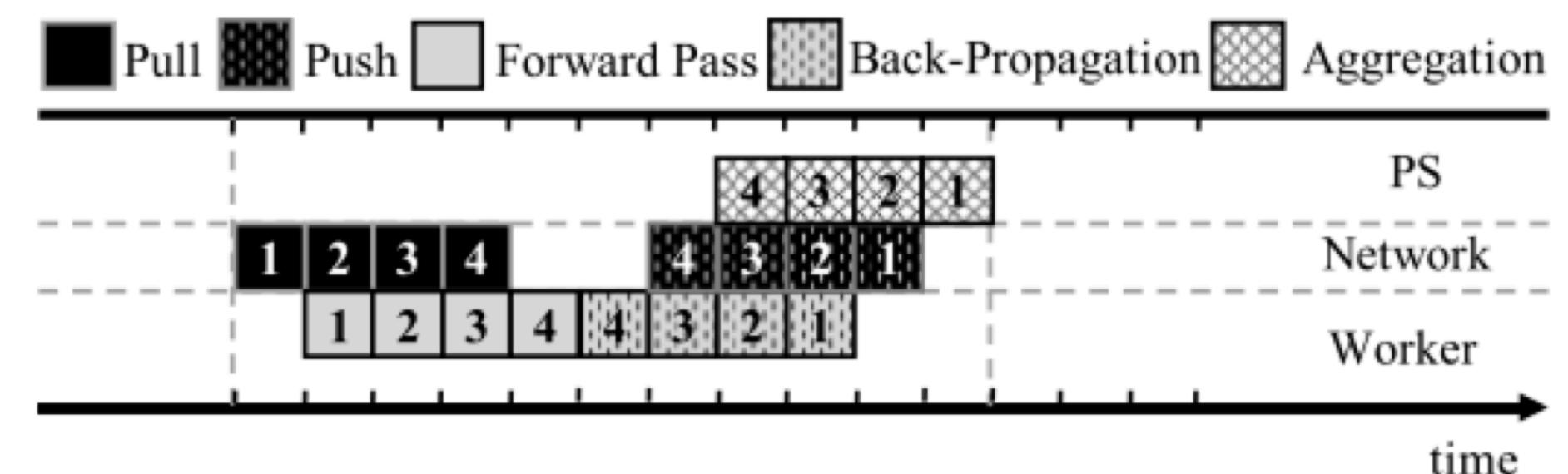
- CNN根据其功能可分为CONV、POOL、ReLU和FC层等。
- 特点：CONV需要占用大量计算资源，但实际参数数量不多。除了输入层，ReLU和POOL都不包含参数。
- 迭代步骤：pull参数、正向计算、反向传播计算、push和聚合梯度
- 主要想法：重叠部分迭代步骤以实现更快的训练速度

Inefficient overlap in current practice

- 计算和通信的重叠度很小，拉取参数时大部分时间无法执行计算。为了充分利用 GPU 的计算能力，计算和通信的重叠度需要提高。
- 如果参数以随机顺序拉取，那么在计算前向传播过程中可能会出现阻塞的情况。



(a) Training using random parameter transfer order



(b) Training using specified parameter transfer order

Background and motivation

Potential performance gains

- 第l层参数获取结束时间: $t_p^l = \sum_{i=1}^j \tau_p^{k_i}$
- 第l层参数前向传播计算结束时间:

两个先决条件: 1) 层l的参数已从PS中提取 2) 层l-1计算完毕

$$t_{ff}^l = \begin{cases} t_p^l + \tau_f^l, & l = 1 \\ \max\{t_{ff}^{l-1}, t_p^l\} + \tau_f^l, & 2 \leq l \leq L \end{cases}$$

- 第l层参数后向传播计算结束时间:

$$t_b^l = t_{ff}^L + \sum_{i=l}^L \tau_b^i$$

- 第l层参数向server push结束的时间:

两个先决条件: 1) 计算了层l的梯度 2) 层l+1的梯度被推到PS

$$t_{bf}^l = \begin{cases} t_{ff}^L + \tau_g^l, & l = L \\ \max\{t_{bf}^{l+1}, t_b^l\} + \tau_g^l, & 1 \leq l \leq L - 1 \end{cases}$$

TABLE I
NOTATION USED IN THIS PAPER

Name	Description
L	The number of layers of the ML model
P^l	The number of parameters of layer l
B	Link bandwidth
τ_{iter}	Time to finish one iteration
τ_f^l	Time to execute forward pass computation of layer l
τ_b^l	Time to execute backward propagation computation of layer l
τ_g^l	Time to push gradient of layer l from the worker to the PS
τ_p^l	Time to pull parameter of layer l from the PS to the worker
t_{ff}^l	Timestamp to finish executing forward pass computation of layer l
t_p^l	Timestamp to finish pulling parameter of layer l
t_{bf}^l	Timestamp to finish pushing gradient of layer l
t_b^l	Timestamp to finish executing backward propagation computation of layer l

Background and motivation

Potential performance gains

- 归纳：

$$\begin{aligned}\tau_{iter} &= t_{bf}^1 = \max\{t_{bf}^2, t_b^1\} + \tau_g^1 \\ &= \max\{\max\{t_{bf}^3, t_b^2\} + \tau_g^2, t_{ff}^L + \sum_{i=1}^L \tau_b^i\} + \tau_g^1 \quad (5) \\ &= \dots\end{aligned}$$

- 由式5可得：迭代时间由 $\tau_f^l, \tau_b^l, \tau_g^l, \tau_p^l$ 和参数传递顺序决定

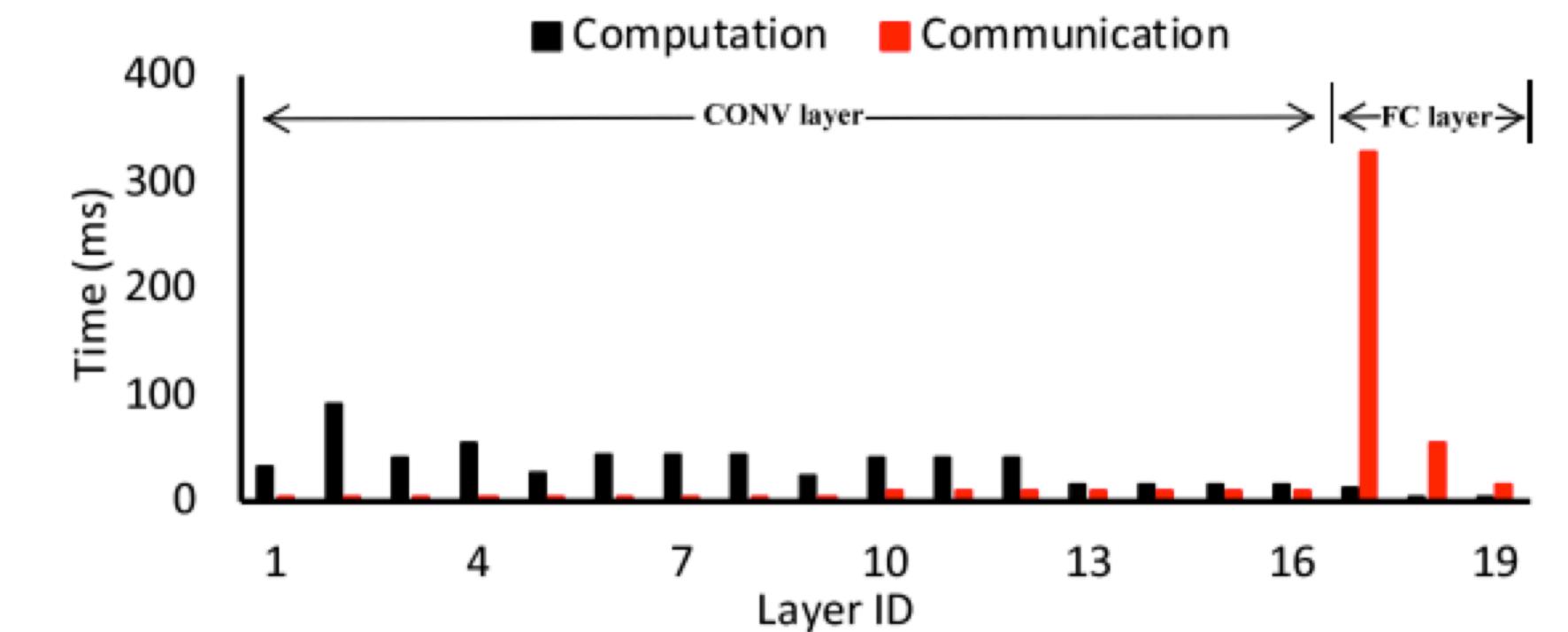
TABLE I
NOTATION USED IN THIS PAPER

Name	Description
L	The number of layers of the ML model
P^l	The number of parameters of layer l
B	Link bandwidth
τ_{iter}	Time to finish one iteration
τ_f^l	Time to execute forward pass computation of layer l
τ_b^l	Time to execute backward propagation computation of layer l
τ_g^l	Time to push gradient of layer l from the worker to the PS
τ_p^l	Time to pull parameter of layer l from the PS to the worker
t_{ff}^l	Timestamp to finish executing forward pass computation of layer l
t_p^l	Timestamp to finish pulling parameter of layer l
t_{bf}^l	Timestamp to finish pushing gradient of layer l
t_b^l	Timestamp to finish executing backward propagation computation of layer l

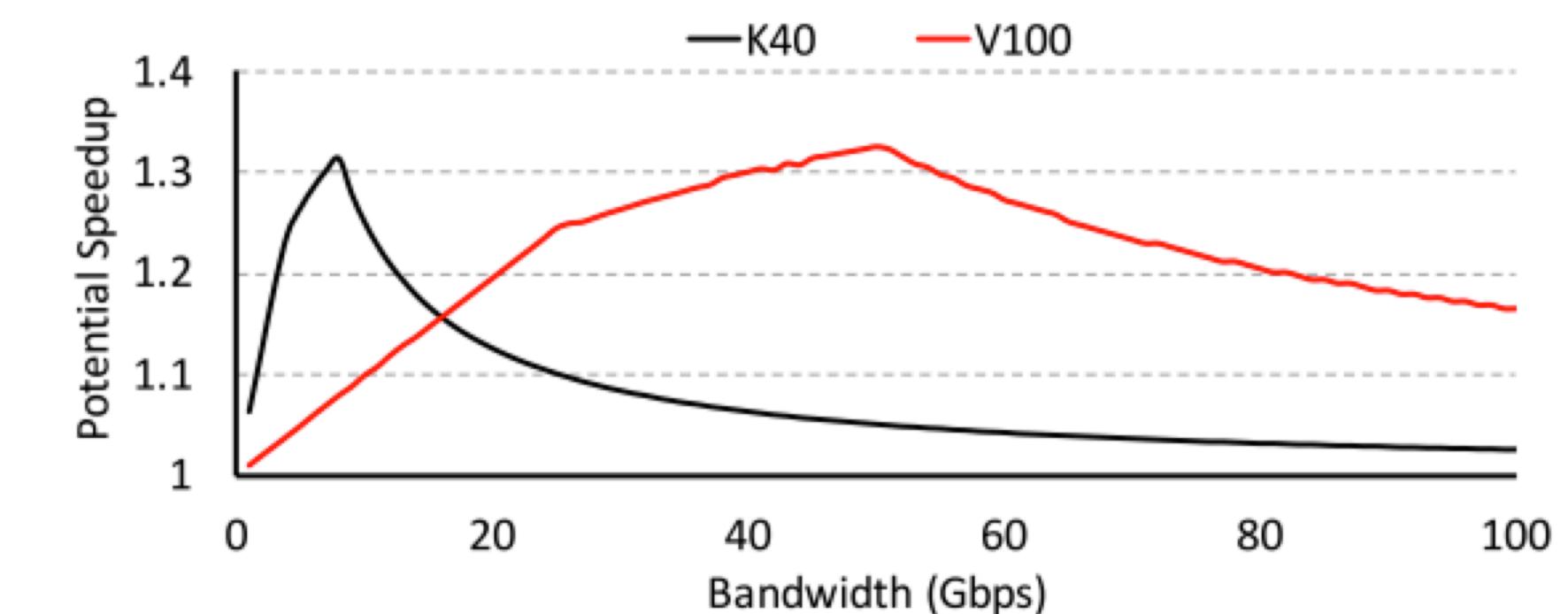
Background and motivation

Potential performance gains

- push和pull的时间计算： $\tau_p^l = \tau_g^l = P^l / B$
- 潜在加速比定义为当参数按顺序从层1传输到层L时的训练吞吐量与参数按相反顺序传输时的训练吞吐量之比
- 在一定的带宽范围内，通信时间与计算时间相当，潜在的改进空间是巨大的。也就是如果参数传输得到优化，可以显著节省硬件计算资源开销



(a) Computation time and communication time of different layers with K40 GPU under 10Gbps bandwidth.



(b) Potential speedup with K40/V100 GPU under different bandwidths.

Fig. 4. VGG-19 model training.

Background and motivation

Necessity of Network-level Parameter Scheduling

- 现有方案：TICTAC和P3
 - 1) TICTAC通过分析 DAG 依赖关系来确定发送方的传输的优先级，从而逼近最佳调度
 - 2) P3对参数进行切片并为其分配优先级，不同优先级的分片排队，优先级最高的分片首先发送到fabric
- 缺点：
 - 1) 现有方案仅在终端主机控制参数传输的顺序
 - 2) 基于终端主机级的解决方案无法保证整个传输过程的优先级。当参数进入网络结构时，终端主机级调度无法帮助协调来自不同 PS 的参数
- 改进：参数调度应该考虑到整个 DML 系统，并且调度需要在终端主机和网络中同时工作
 - 1) 在端主机，不同优先级（紧急）的参数被调度到不同的发送队列，保证了高优先级参数在端主机共存时，优先于低优先级参数发送
 - 2) 当高优先级参数和低优先级参数都进入网络结构时，通过交换机转发的优先级调度仍然可以生效，保证了高优先级参数的提前传递

Design

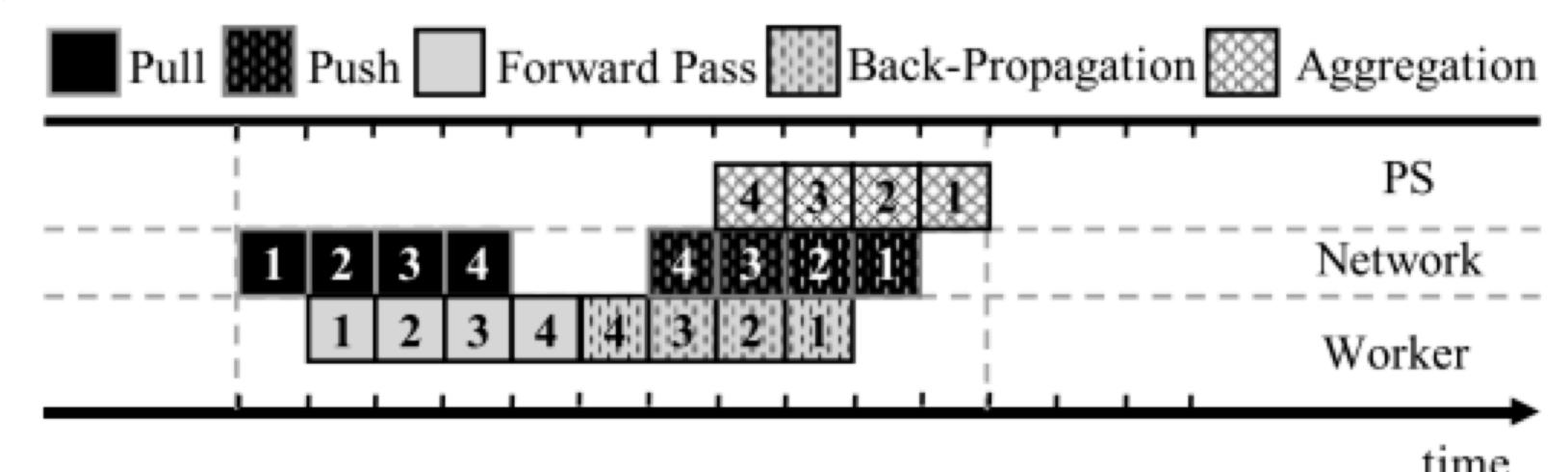
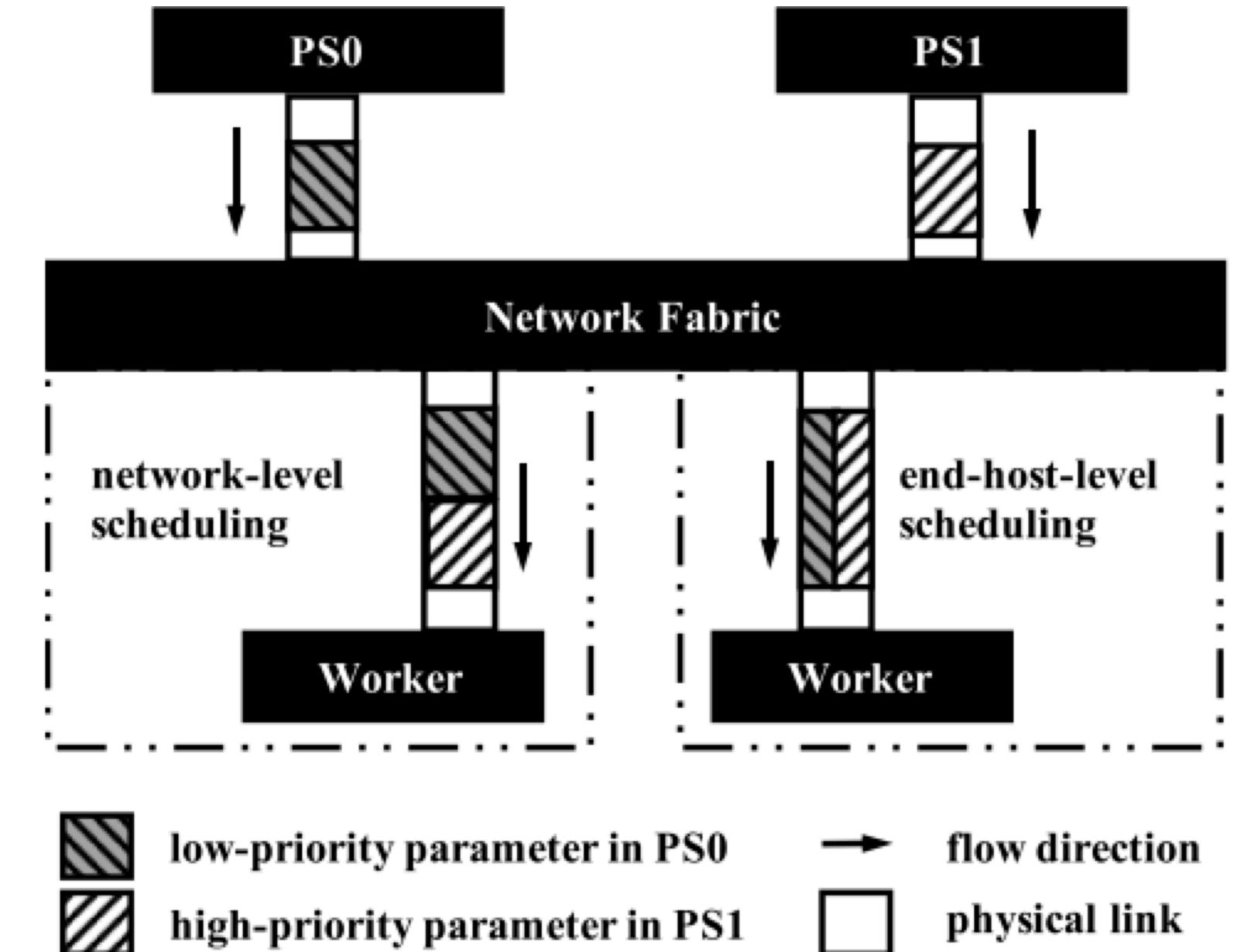
Network-level scheduling

- Geryon设计：

- 1) 根据参数的紧急程度将参数划分为不同的组，并使用相同数量具有不同优先级的流来传输不同的组
- 2) 携带最紧急参数的流被分配到最高优先级，随着紧急程度的降低，携带参数的流的优先级逐渐降低
- 3) 每个包头中的优先级标签可以被交换机识别。当不同优先级的报文共存时，交换机会优先转发高优先级的报文，帮助它们更早到达工作节点

- Geryon的优势：

- 1) Geryon带来了更好的计算与通信的重叠
- 2) 在Geryon中，将参数分配给不同流的过程由 DML 框架执行，其他调度逻辑由底层网络结构控制。而终端主机级解决方案则需要在DML框架中明确控制参数传输顺序



(b) Training using specified parameter transfer order

Design

Coarse categorization of parameters

- 基于优先级的流量调度存在的问题：
 - 1) 交换机缓冲区有限，仅能使用有限数量的优先级队列，通常为2个优先级，因此不能为每层参数分配一个唯一的优先级
 - 2) 更多的优先级会增加终端主机的调度复杂性以及调度开销
- 解决方案：
 - 1) 观察得到：FC层的参数占据了大部分，而CONV层则只占据了一小部分，但是CONV层的计算时间占总计算时间的绝大部分
 - 2) 只使用两个优先级，CONV层优先级高，FC层优先级低

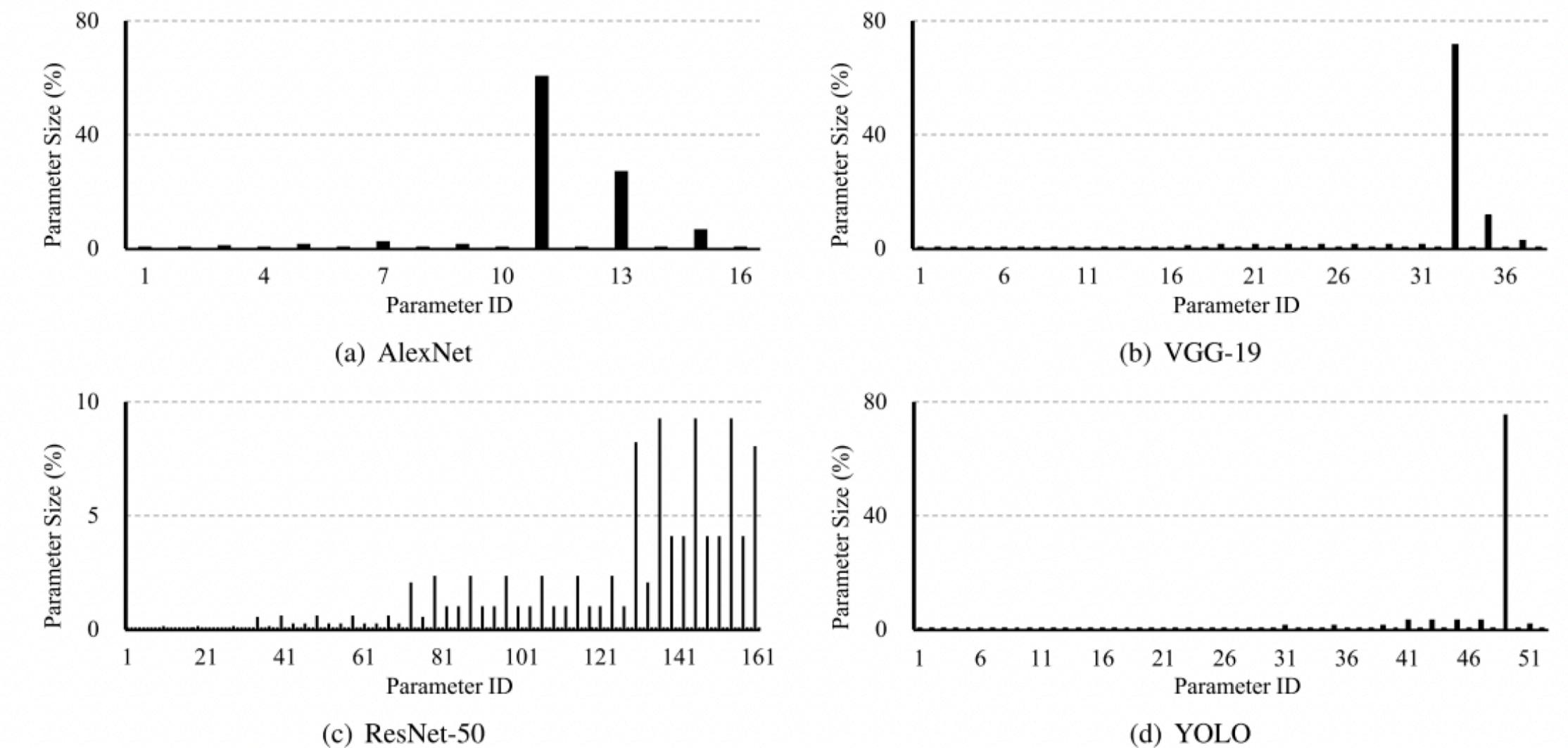


Fig. 6. Parameter size distribution.

Design

Flexible categorization of parameters

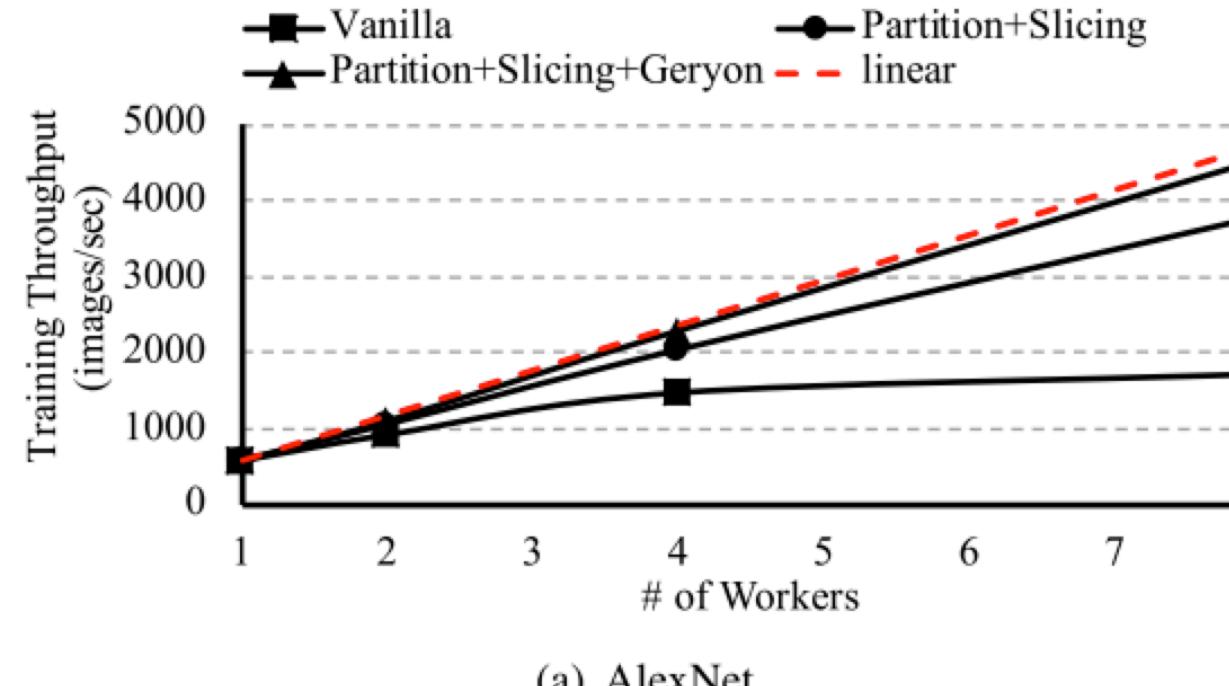
- 粗分类存在的问题：在一些模型中，FC层的参数只占总参数的一小部分。例如，ResNet-50中FC层的参数仅为总参数的8.02%
- 解决方案：引入优先级阈值来决定发送优先级
 - 如果一个参数的紧急级别大于优先级阈值，则该参数将高优先级发送，反之亦然
 - 在前几次迭代中，依次使用一些不同的阈值，并收集相应的迭代次数。然后拟合迭代时间与优先级阈值的关系，从而为后续训练选择与迭代时间最小值对应的阈值
- 值得注意的是，优先级阈值搜索过程仅在最初的几次迭代中执行，并且不需要在确定优先级阈值之后从头开始再次开始训练
- 未来趋势：随着RDMA技术的快速发展和硬件内存的增加，未来可以支持更多的优先级，以进行更精细的分类并获得更好的性能

Evaluation

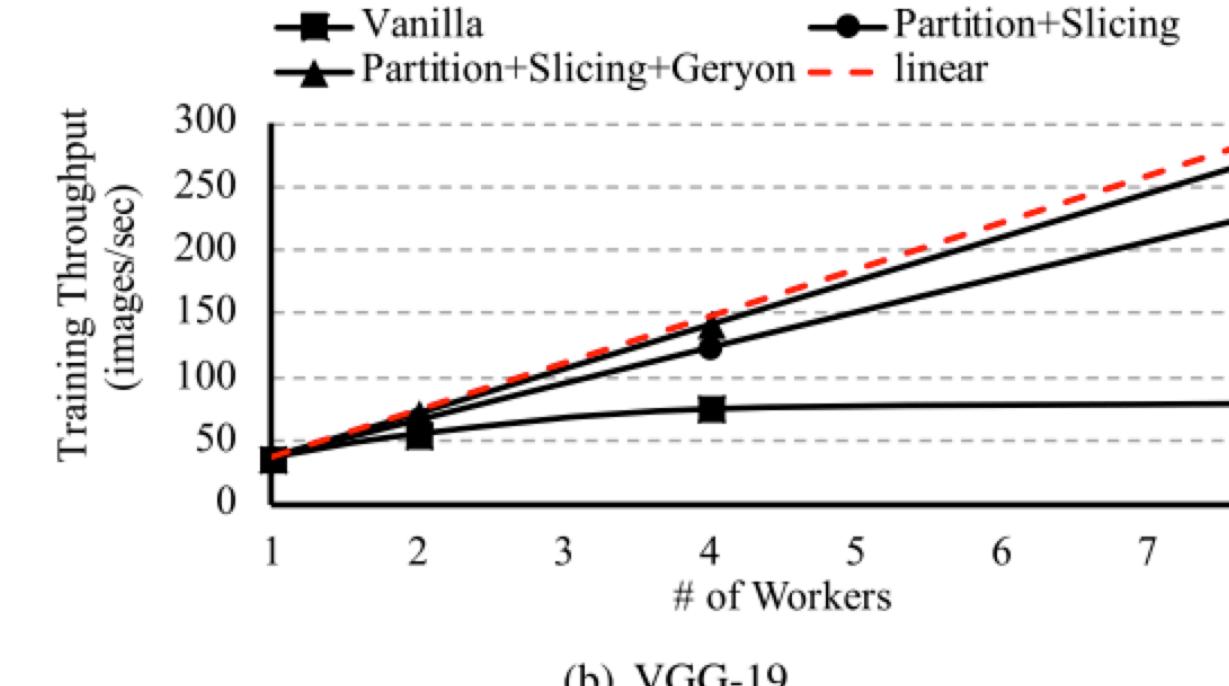
训练吞吐量：每秒所有worker处理的图像总数

1) 加速比，定义为Geryon训练吞吐量与vanilla TensorFlow训练吞吐量之比

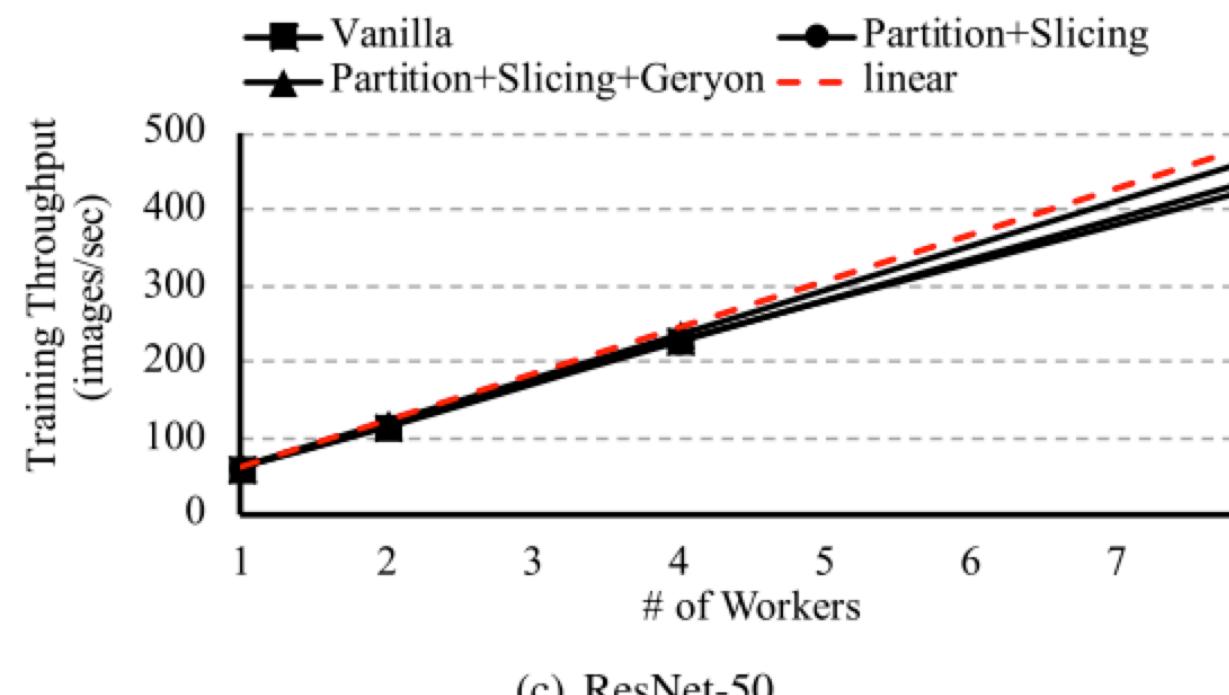
2) 缩放效率，定义为在DML训练中每秒由一个worker处理的图像数量与在单GPU训练中每秒处理的图像数量的比率



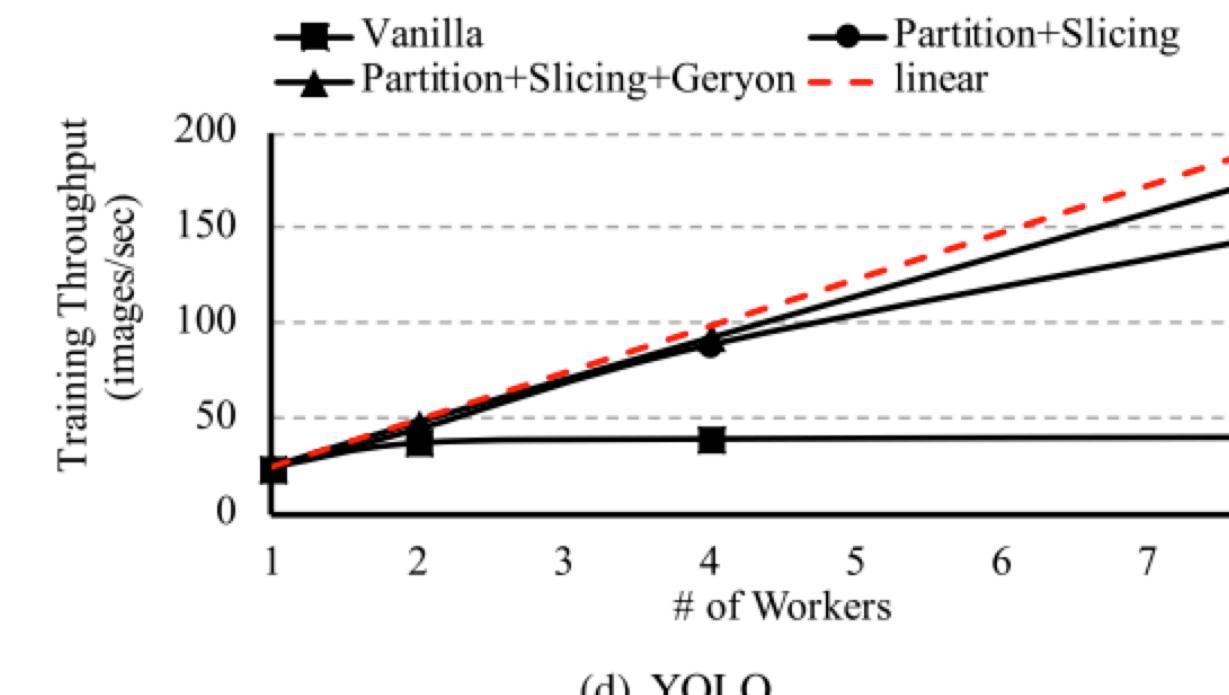
(a) AlexNet



(b) VGG-19



(c) ResNet-50



(d) YOLO

TABLE II
SCALING EFFICIENCY WITH 8 WORKERS USING 10GBE BANDWIDTH

Model	Vanilla	Partition + Slicing	Partition + Slicing + Geryon
AlexNet	36.4%	80.2%	95.4%
VGG-19	26.9%	79.3%	94.6%
ResNet-50	87.6%	90.0%	95.7%
YOLO	20.9%	75.8%	91.5%

TABLE III
SCALING EFFICIENCY WITH 8 WORKERS USING 40GBE BANDWIDTH

Model	Vanilla	Partition + Slicing	Partition + Slicing + Geryon
AlexNet	79.4%	90.0%	95.1%
VGG-19	61.7%	85.3%	94.8%
ResNet-50	94.1%	94.1%	96.1%
YOLO	53.2%	84.2%	93.6%

Evaluation

TABLE IV
TRAINING THROUGHPUT WITH DIFFERENT SLICING THRESHOLD

Slicing threshold	Training throughput (images/sec)	
	Geryon	TICTAC
1024K	279.2	272.0
512K	276.0	268.0
256K	272.8	216.8

Conclusion

- 缺点：
 - 1) 目前只针对CNN可以进行优化，并且从实验结果可以看到针对resnet-50这种网络的优化并不是特别明显
 - 2) 在evaluation部分，比较的维度仅为训练的吞吐量，而没有训练精度的对比
 - 3) 数据包有了优先级之后就会存在公平性问题

THANKS!