Addressing Network Bottlenecks with Divide-and-Shuffle Synchronization for Distributed DNN Training

关于优化分布式DNN的参数同步瓶颈的方法

Introduction

在当前的集群的训练中，使用BSP在不同的机器上训练DNN模型，每次迭代中都会强制执行全局同步，聚合所有worker的梯度。但是物理带宽的限制，同步的过程会引起网络传输的瓶颈，导致较大的传输的空闲等待。

因此，以BSP为基准，分布式DNN训练的理想同步方案应达到以下目标：

- 通信效率：具有拓扑感知，充分利用网络带宽避免产生瓶颈，减少BSP在每次迭代中的空闲等待
- 收敛精度：在相似的迭代中，它应该保持与BSP相同的收敛精度

现有的解决方案有：

- 系统级优化：替换同步策略，替换逻辑拓扑。造成较长的依赖链，如果产生通信延迟，会导致下游链路的拥塞。难以避免空闲等待
- 算法优化：使用异步并行ASP。收敛速度慢，需要更多轮的迭代，还可能导致收敛不准确

因此作者提出了DS-Sync (divide-and-shuffle synchronization)

将worker节点划分成不同大小的不相交的子集，独立地进行同步，并定期在不同子集之间进行参数交换，达到全局同步。

每次迭代中，worker节点使用局部梯度优化模型，将参数进行局部同步，在组内选择

每次迭代时选择一个节点与其他组进行全局同步，组内进行局部同步

通过多次迭代后也能够收敛

Background

- 集群的网路瓶颈
  - 静态拓扑的异构性
  - 多个任务竞争带宽：降低spine的通信速度
  - 多任务竞争GPU带宽：降低worker的通信速度
- 解决方案
  - 系统级优化
  - 算法优化

Design

- 拓扑检测

  使用DPDK或iPerf进行带宽测量，可以得到拓扑信息

- DS-Sync Group 初始化

  根据静态拓扑异构，DS-Sync初始化周期性划分洗牌模式，将worker节点划分为不同的组（inter-rack和intra-rack）。以此避免过载

- DS-Sync Group 调整

  检测到一段时间内发生带宽争用，DS-Sync会进行调整，将Group划分为更小的组，

- 参数同步

  DS-Sync并行同步组内的参数，组内同步完毕后会使用AllReduce对组间参数进行平均。

根据不同的网络瓶颈对worker进行分组和shuffle

- 将网络瓶颈和其他瓶颈分离，将worker保持在一个较小的组中缓解瓶颈
- 保证所有worker通过shuffle机制能够直接或间接交换信息

三种瓶颈：

- 静态拓扑异构性

  worker分布在不同的物理网络中，分为机架内组和机架间组。每次从每个机架中挑选一个worker作为机架内组的代表，担任机架间组的参数同步。

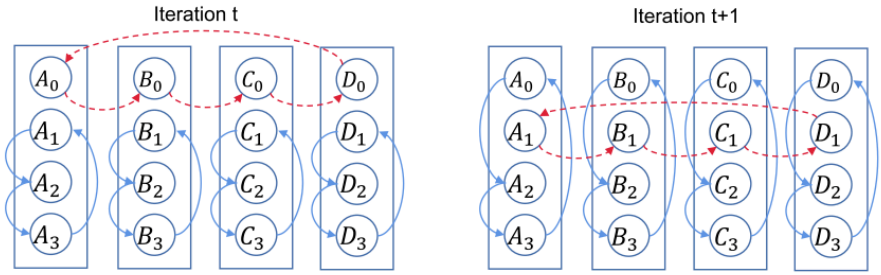

Fig. 4: Group initialization divides all workers into inter-rack and intra-rack groups and shuffles workers to be the representative in the inter-rack group by turns.

- 机架间链路带宽争用

  当某个机架的代表因为其他任务导致通信缓慢，DS-Sync会将该机架进行单独划分

  红框内的机组同时在执行其他任务，所以组间的通信就变成了2组

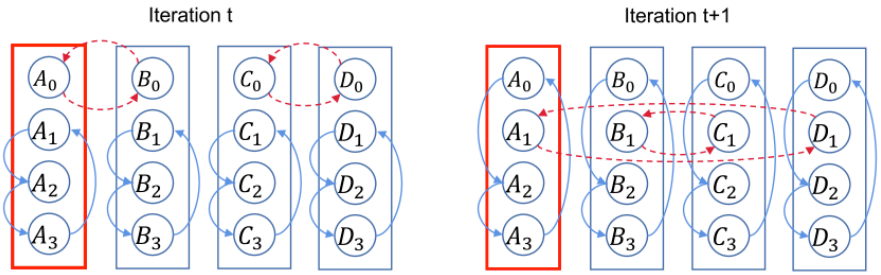

Fig. 5: Rack A (in the bold and red rectangles) has some background flows from other tasks to share inter-rack bandwidth. DS-Sync further divides inter-rack groups to reduce the group size in group adjustment during training.

- 终端主机网卡带宽争用

  同一节点的另一个任务共享在共享终端主机的网卡带宽，DS-Sync会调整组模式。也就是会将带宽受限的主机单独划分到一个组中，尽可能减小带宽受限给其他主机带来的影响

  A5同时在执行其他任务，带宽受限了，那么就将A5单独与组内的一台机器单独汇聚

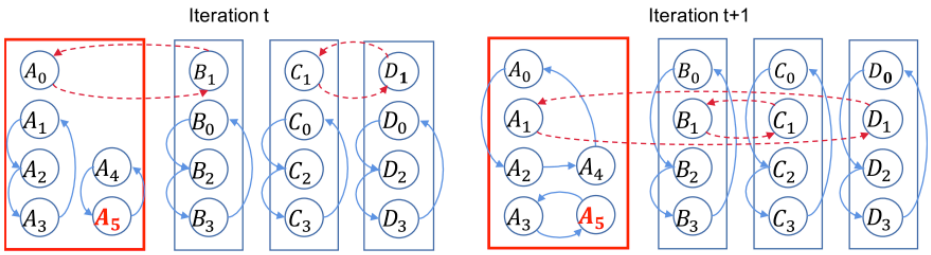

Fig. 6: In rack A, worker A_5 (in the bold and red characters) has another task sharing the end-host NIC. The upper-level link of rack A also has smaller bandwidth due to background flows. Besides the inter-rack group, DS-Sync further divides the intra-rack group to alleviate the bottleneck on worker A_5.

Analysis

通信时间分析：

- N个worker，C个机架组，每个组有G个节点。α是传播延迟，β_1是组内1字节的传输延迟，β_2是组间1字节的传输延迟，S是参数的大小

TABLE II: Communication Cost Summary

Methods	Latency	Transfer Delay
PS	$2(N-1)L\alpha$	$\frac{2[(C-1)GS\beta_2+(G-1)S\beta_1]}{P}$
Ring	$2(N-1)L\alpha$	$2(N-1)S\beta_2/N$
Double Tree	$2(logN+k)L\alpha$	$2(LogN+k)S\beta_2/k$
Hierarchical	$2(G+C-2)L\alpha$	$\frac{2(G-1)S\beta_1}{G}+\frac{2(C-1)S\beta_2}{C}$
Gossip	$2L\alpha$	$\geq 4S\beta_2$
DS-Sync	$2L\alpha$ or $2L(G-1)\alpha$	$2S\beta_2$ or $\frac{2(G-2)S\beta_1}{G-1}$

DS-Sync keeps the bottlenecked links and related workers in the smallest group of two workers, and $G-1$ is the size of the intra-rack group.

- PS容易受到组间传播延迟的影响
- Ring和Tree的同步过程有拓扑的长依赖，因此容易受到其他任务的干扰
- 分层算法有多个串行通信步骤，如果一个worker同时在组内和组间负责传输，这个worker的传输效率会降低
- Gossip算法
- DS-Sync算法会将worker进行非重叠并行同步，将lagger划分到小的组中，降低影响。

收敛性分析

Assumption：

- 利普希茨光滑：

> **Assumption IV.1. Lipschitzian smooth:** Any local function of worker i $F_i(\cdot)$ is with L-Lipschitzian gradients.
>
> $$\|\bigtriangledown F_i(x;\xi) - \bigtriangledown F_i(y;\xi)\| \leq L\|x-y\|$$

- 有界误差:

> **Assumption IV.2. Bounded variance:** Assume the variance of stochastic gradient $\mathbb{E}_{i\sim\mathcal{U}([n])}E_{\xi\sim\mathcal{D}_i}\|\bigtriangledown F_i(x;\xi) - \bigtriangledown f(x)\|^2$ is bounded for any parameters x with worker i uniformly sampled from $\{1,\cdots,n\}$ and data batch ξ from the distribution $\mathcal{D}_i$. This implies there exist constants σ such that
>
> $$\mathbb{E}_{i\sim\mathcal{U}([n])}E_{\xi\sim\mathcal{D}_i}\|\bigtriangledown F_i(x;\xi) - \bigtriangledown f(x)\|^2 \leq \sigma^2$$

达到共识的证明：

分布式平均问题中每个节点都会和邻近的节点交换参数，最终要达到的目的是在K次迭代后，全局的参数要逼近worker参数的平均

在DS-Sync中，

Conclusion

不足：没有考虑到容错机制，