# BLINK: FAST AND GENERIC COLLECTIVES FOR DISTRIBUTED ML(MLSys 2020)

Wang Jian

2020 年 11 月 12 日

## 目录

**摘要**

跨 GPU 的模型参数同步为大规模数据并行训练引入了高开销。面对不断增加的硬件异构性，现有的参数同步协议无法有效地利用可用的网络资源。我们提出 Blink(a collective communication library that dynamically generates optimal communication primitives by packing spanning trees)，与 NCCL(modern communication libraries, NVIDIA’s Collective Communications Library) 进行比较 (soa)。

# 1 INTRODUCTION

In data-parallel training, each GPU has a full copy of the model parameters and GPUs frequently exchange parameters with other GPUs involved in training.

通信开销占 50%-90%，the fact that GPU computation is getting faster and model sizes are growing larger, thus making communication overheads stand out

现有的通信库：NVIDIA’s Collective Communications Library (NCCL), Baidu’s Ring AllReduce

实现 gpu 间集群峰值性能的核心障碍是由于拓扑异构导致的链路利用率不足。我们发现这种情况的发生主要有两个原因：

1. 服务器配置不同，协议必须具有拓扑意识，才能有效地使用硬件

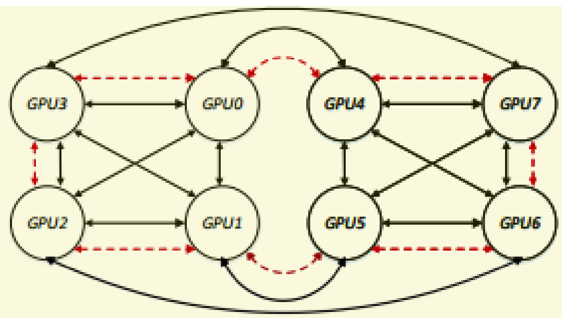



*Figure 1.* Hybrid mesh-cube topology of NVLink in the DGX-1 8-GPU server. Solid lines here indicate the bi-directional NVLinks on the DGX-1-P100, red dashed-lines are the additional NVLinks in DGX-1-V100 servers. NVLink Gen1 has bi-directional pairwise throughput of 18-20GB/s (DGX-1-P100); Gen2 goes up to 22-25GB/s (DGX-1-V100).

2. 在多租户集群中，对 gpu 之间的互连拓扑是不敏感的。许多 job 被分配到同一个机器上。必须能够利用碎片化时间来避免排队延迟。

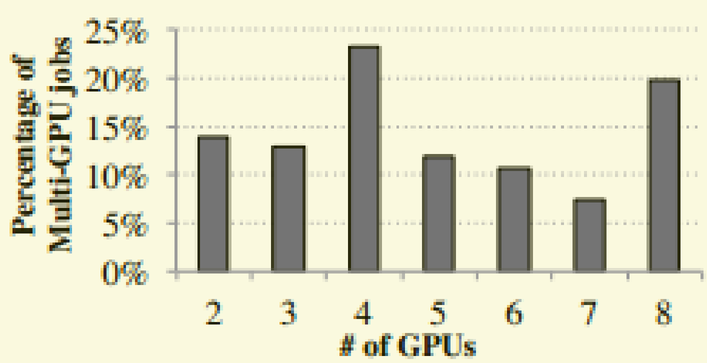



*Figure 2.* Number of GPUs within each 8-GPU server allocated to 40,000 multi-GPU jobs

尽管 jobs 大多以 2 的幂次请求 gpu ，大多数机器还是分配了 3,5,6,7 个 gpu 。

**Contributions**:

- 提出 Blink，以达到接近最佳的链路利用率。
- 动态生成给定拓扑的最佳通信原语
- probes the set of links available for a given job at runtime and builds a topology with appropriate link capacities.
- ...
- 提供 NCCL-compatible API, 无缝衔接深度学习框架。

# 2 MOTIVATION

## 2.1 The case for packing trees

我们的动力来自通信的开销

## 2.2 Micro Benchmarks

- Depth Test
  - A simple chain topology

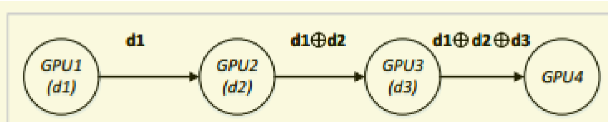
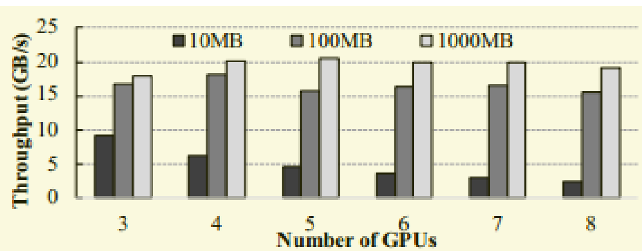


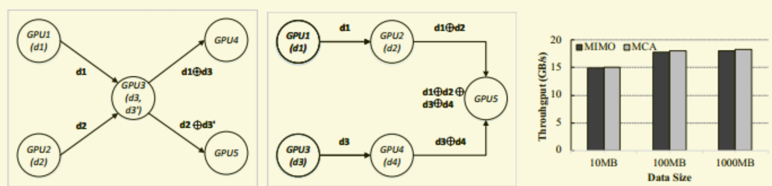

*Figure 6.* Depth test of Reduce+forward, over a chain of GPUs.




7. Throughput for Reduce+forward over a chain of GPUs.

数据集越小，吞吐量下降；chain length 越长（GPU 越多），吞吐量从 21 降到 19

- Multi-transfer Test Two topologies: a multi-input, multi-output (MIMO) as Fig. 8(a) and a multi-chain aggregation (MCA) as Fig. 8(b).



(a) MIMO (b) MCA (c) MIMO, MCA Throughput

*Figure 8.* MIMO, MCA topology and test throughput.

## 2.3 Blink Approach

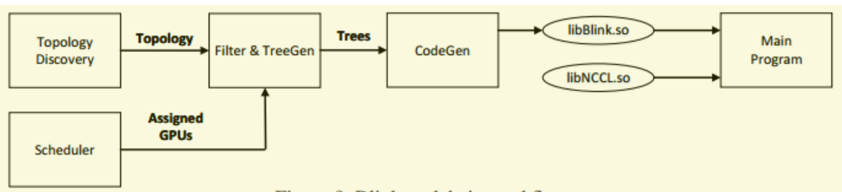


*Figure 9.* Blink toolchain workflow

工作流程：

- 在运行时，一旦深度学习任务被调度并分配了一组 gpu, Blink 就能够探测机器的拓扑，并通过所分配的 gpu 推断互连拓扑。
- TreeGen，这一步输出一组生成树和对应于应该通过它们发送多少数据的权重
- CodeGen parses the spanning trees and generates CUDA code.
- 设置 $LD_{PRELOAD}$ flag，动态加载 Blink 的实现。

# 3 DESIGN

## 3.1 Packing Spanning Trees

GPU, links –> 图论 –> 寻找最大权重的 spanning tree

$$\max \sum_i w_i \tag{1}$$

$$\text{such that } \forall e \in E, \sum_i \kappa_i * w_i < c_e \tag{2}$$

$$\text{where } \kappa_i = \begin{cases} 1, & \text{if } e \in T_i \\ 0, & \text{otherwise} \end{cases} \tag{3}$$

直接寻找事件复杂度较大，下面介绍优化方法。

## 3.2 Approximate Packing

multiplicative weight update(MWU):

- 初始化权重和 capacity
- 循环：
  - 寻找最小权重的 spanning tree
  - 降低该 tree 的权重，更新图的权重

### 3.2.1 Minimizing Number of Trees

将权重限制为 0 or 1 –> 整数线性规划 integer linear program(ILP)

## 3.3 Handling many-to-many operations

前面的讨论关注于 one-to-many, 为了解决 many-to-many, 我们发现所有机器是双向的，所以图变成了无向图。

### 3.4 DGX-2 and Multi-server settings

# 4 IMPLEMENTATION

## 4.1 CodeGen Implementation

简化问题, 只讨论两种 collective communication: Broadcast and AllReduce

## 4.2 CodeGen Optimizations

两个问题

### 4.2.1 Automatic chunk size selection

块的大小影响整体的效率。

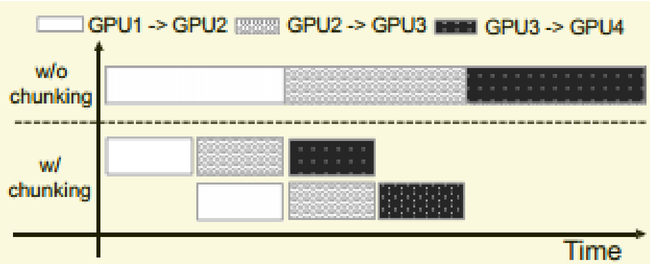



*Figure 10.* Data chunking to reduce multi-hop latency.

使块小点，能够提高系统效率。因此我们需要设计如何自动选择块的大小。

multiplicative increase, additive decrease (MIAD): 初始化块大小很小，当吞吐量提高，以乘法增加块大小，直到达到一个平稳状态。

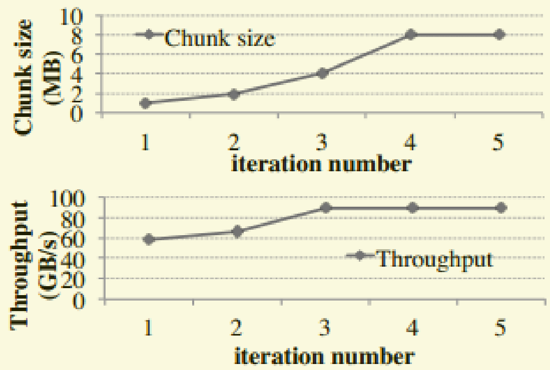



*Figure 11.* Automatic chunk size selection with MIAD (multiple-increase, additive-decrease.)

### 4.2.2 Link Sharing

问题：CUDA functions do not provide any direct control on how links are shared

解决：reusing CUDA streams when the same link is used in multiple trees at roughly the same position

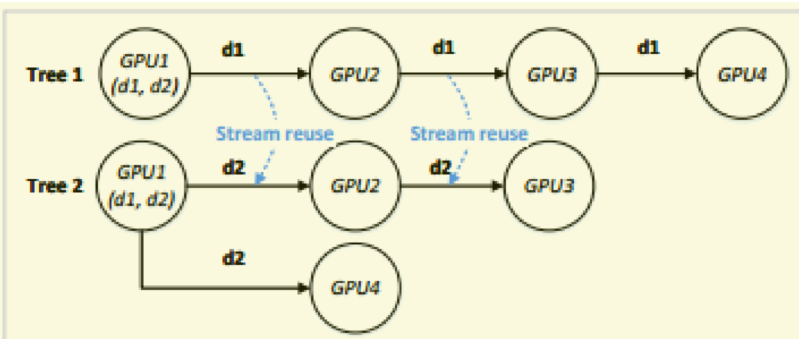



*Figure 12.* Stream reuse for fair sharing of links.

# 5 EVALUATION

## 5.1 Tree Packing Benefits

## 5.2 Broadcast, AllReduce Micro-benchmarks

## 5.3 End-to-end Training

# 6 RELATED WORK

- Topology-fixed Schemes: MPI, Horovod, All-Reduce, Gloo
- Topology-aware Protocols

# 7 CONCLUSION

Blink 是一个用于加速分布式 ML 的快速通用的集体通信库。为了处理现代 GPU 硬件中普遍存在的拓扑异构，Blink 动态地包生成树以最大化链路利用率。与最先进的基于环的协议 (如 NCCL2) 相比，Blink 实现了高达 8 倍的模型同步，并将端到端训练时间减少了 40%。