编译论坛

论文分享：

Breaking the Computation and Communication Abstraction Barrier in Distributed Machine Learning

Abhinav Jangda
University of Massachusetts Amherst
United States

Jun Huang
Ohio State University
United States

Guodong Liu
Chinese Academy of Sciences
China

Amir Hossein Nodehi Sabet
University of California, Riverside
United States

Saeed Maleki
Microsoft Research
United States

Youshan Miao
Microsoft Research
China

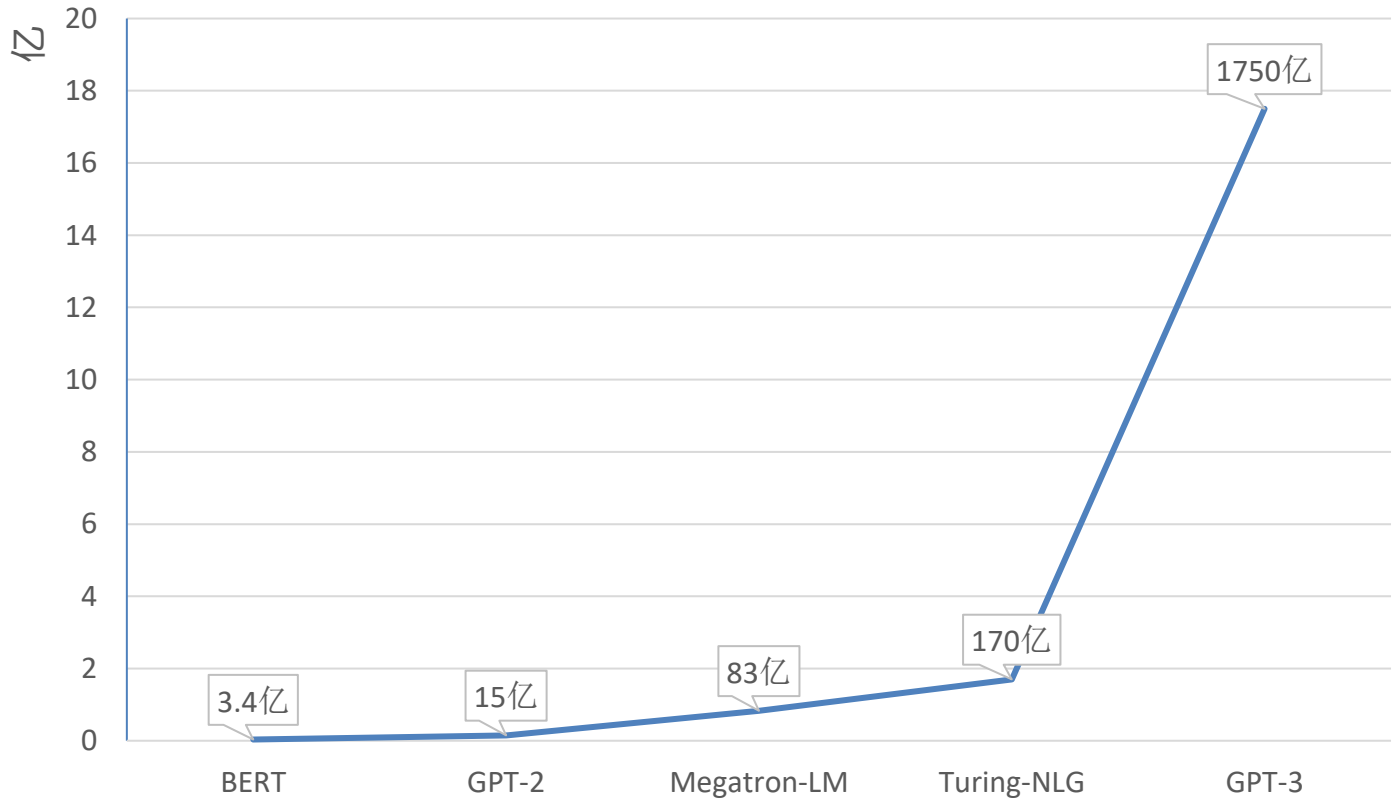
Madanlal Musuvathi
Microsoft Research
United States

Todd Mytkowicz
Microsoft Research
United States

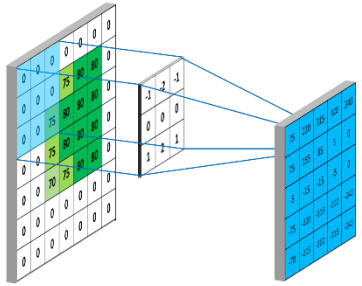
Olli Saarikivi
Microsoft Research
United States

嘉宾：李鹏飞

目录



随着模型参数的不断增加，模型的训练和推理必须要使用分布式的方式。此外，随着计算需求变得越来越高，即使对最后的百分比进行优化也能在时间、能源和金钱节省方面带来巨大好处。

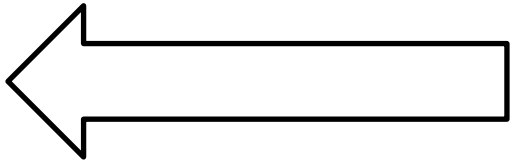


卷积运算

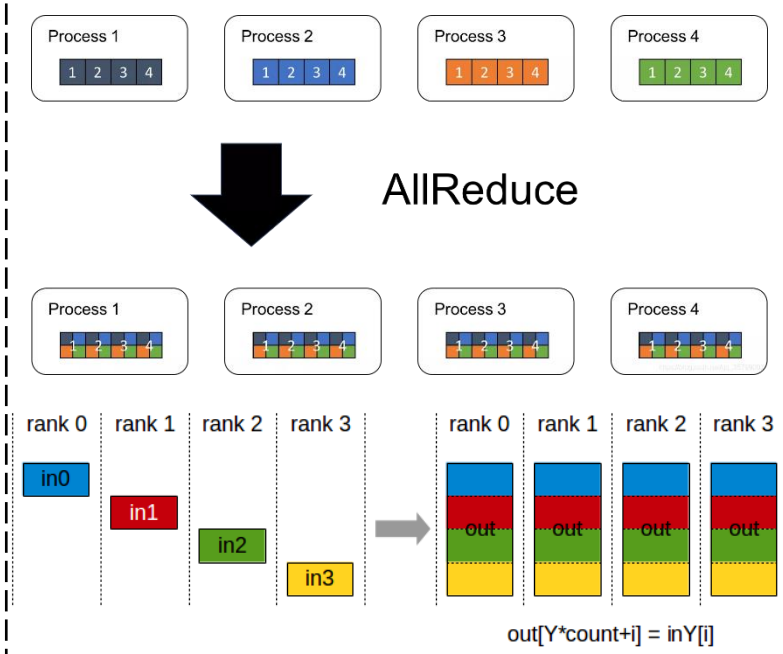
$$\begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix} \begin{bmatrix} B_{11} & B_{12} \\ B_{21} & B_{22} \end{bmatrix}$$

矩阵乘

计算



无法联合优化



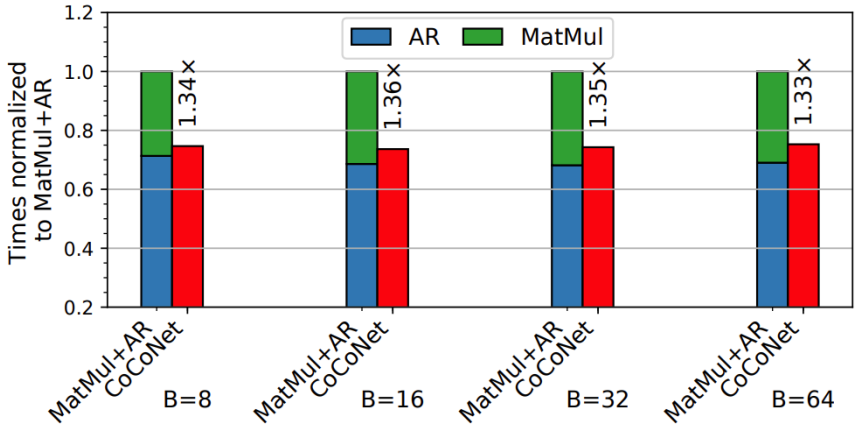
AllGather

通信

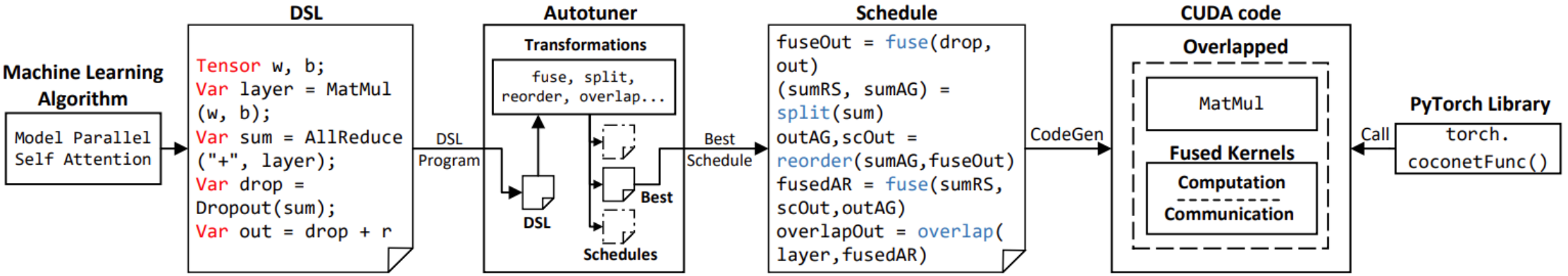
具体的优化内容：

- 接口优化（Interface optimization）消除了抽象调用者和被调用者之间的不匹配。
- 融合优化（Fusion optimization）通过生成单个内核来执行多个通信和计算操作，从而减少内存带宽使用。
- 重新排序优化（Reorder optimization）可以在通信操作之前或之后移动计算，从而分散计算或启用新的融合可能性。
- 重叠优化（Overlapping optimization）以细粒度的方式协调多个计算和通信操作，以充分利用网络和计算资源。

作为分布式机器学习方法之一的模型并行性，将每一层分布在多个GPU中，每一层的计算包括每个节点上的矩阵乘法（MatMul），然后是AllReduce。现有的模型并行化实现会调用单独优化的库函数来实现 MatMul 和 AllReduce。但是，由于网络在 MatMul 期间（也就是计算过程中）处于空闲状态，因此无法同时利用网络和计算资源。通过以细粒度的方式将 MatMul 的计算与 AllReduce 的通信 overlapping，就可以同时充分利用网络和计算资源。

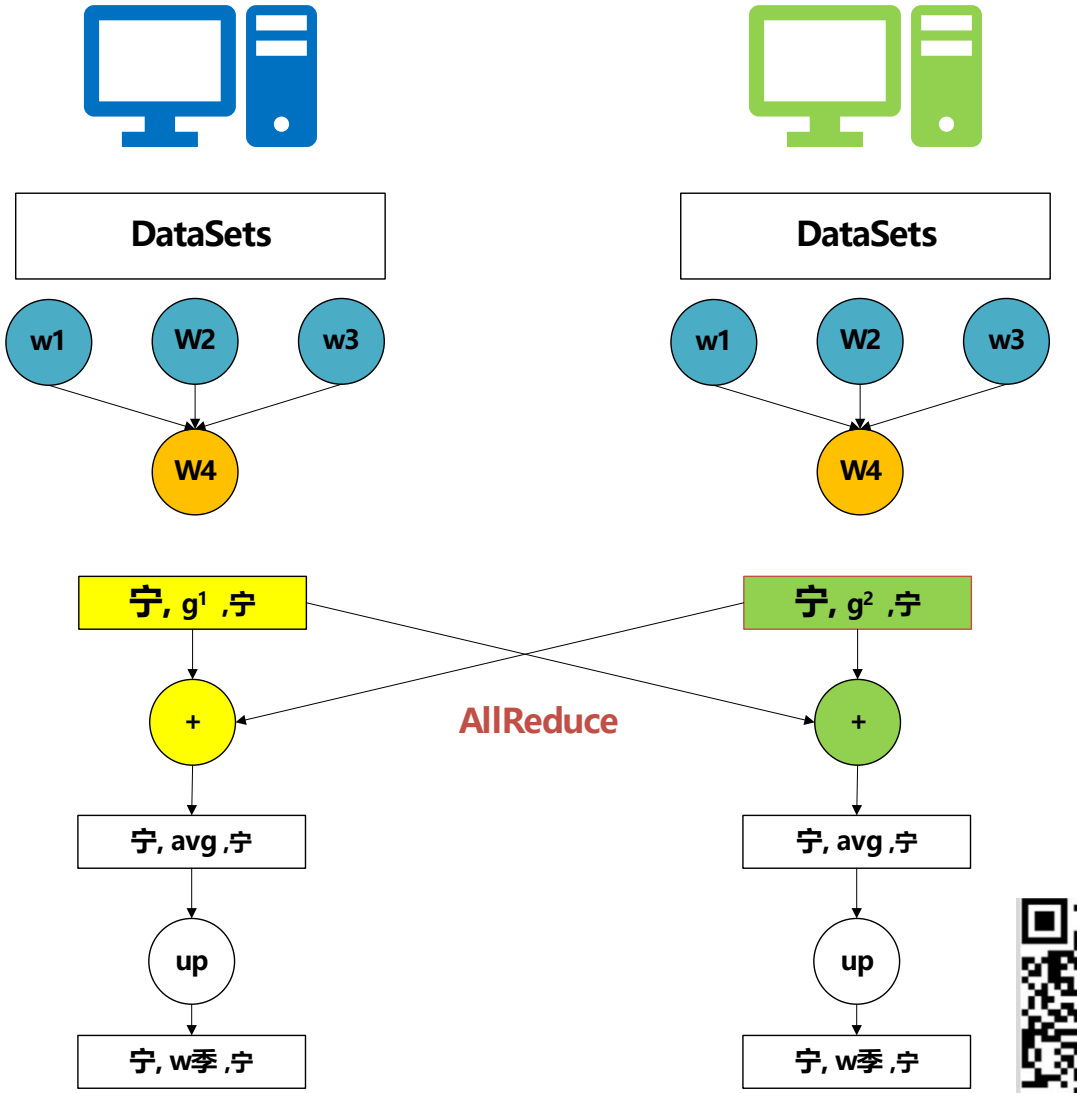


目录

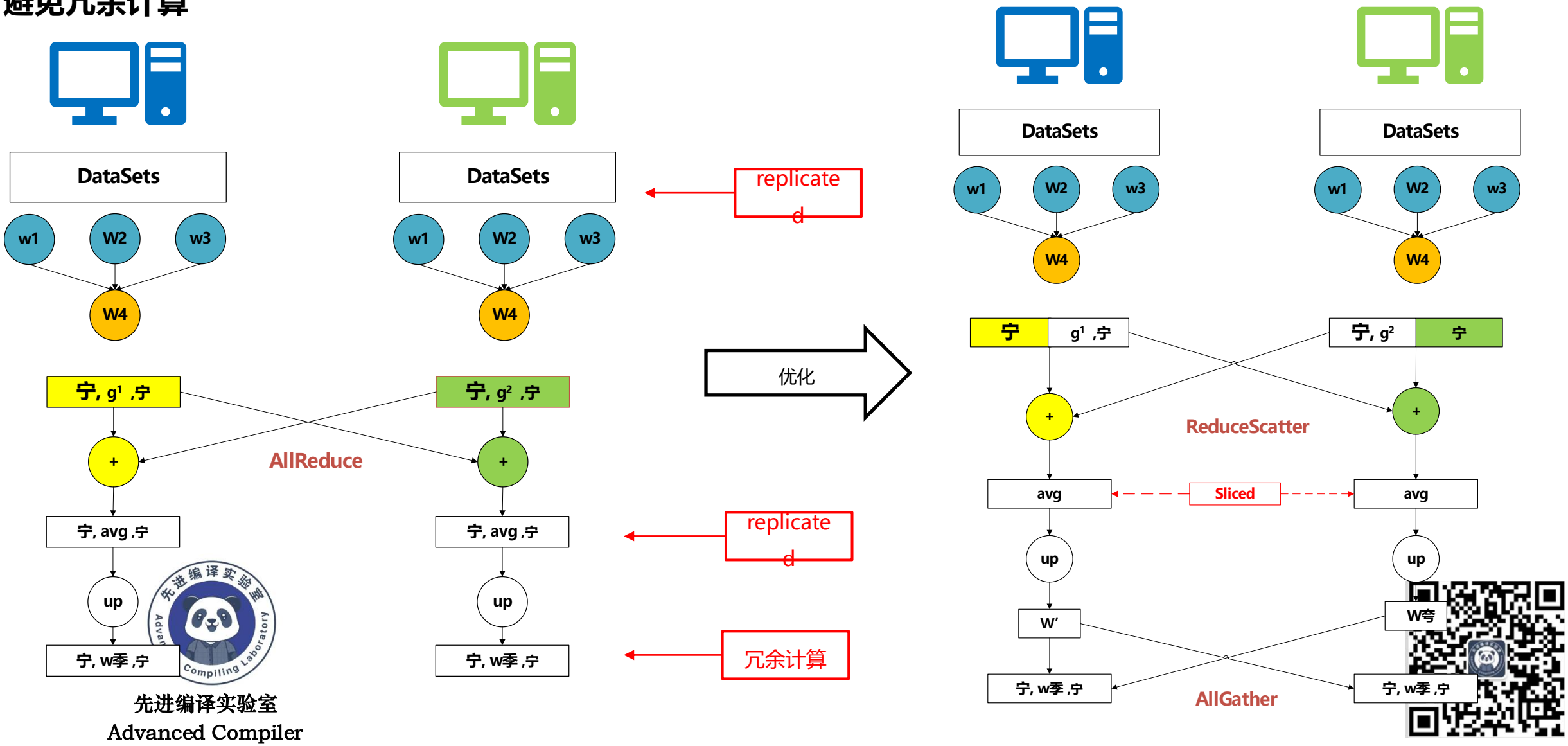

Machine Learning Algorithm
Model Parallel Self Attention
DSL
Tensor w, b;
Var layer = MatMul (w, b);
Var sum = AllReduce ("+", layer);
Var drop = Dropout(sum);
Var out = drop + r
DSL Program
Autotuner
Transformations
fuse, split, reorder, overlap...
DSL
Best
Schedules
Best Schedule
Schedule
fuseOut = fuse(drop, out)
(sumRS, sumAG) = split(sum)
outAG,scOut = reorder(sumAG,fuseOut)
fusedAR = fuse(sumRS, scOut,outAG)
overlapOut = overlap(layer,fusedAR)
CodeGen
CUDA code
Overlapped
MatMul
Fused Kernels
Computation
Communication
Call
PyTorch Library
torch.
coconetFunc()

Tensor layout

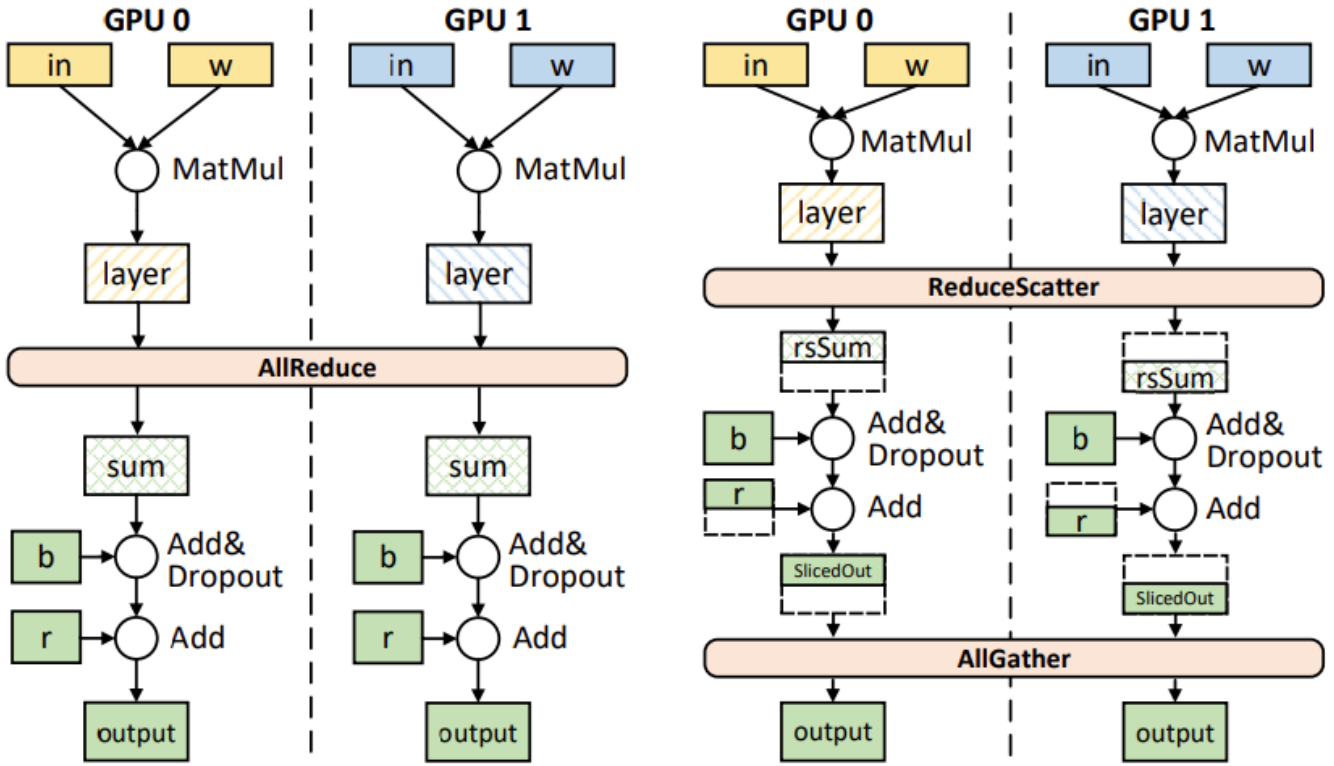
- sliced切分：在所有机器上对数据集进行切分
- replicated复制：在所有机器上都是相同的数据
- local：所有机器上都是不同的。



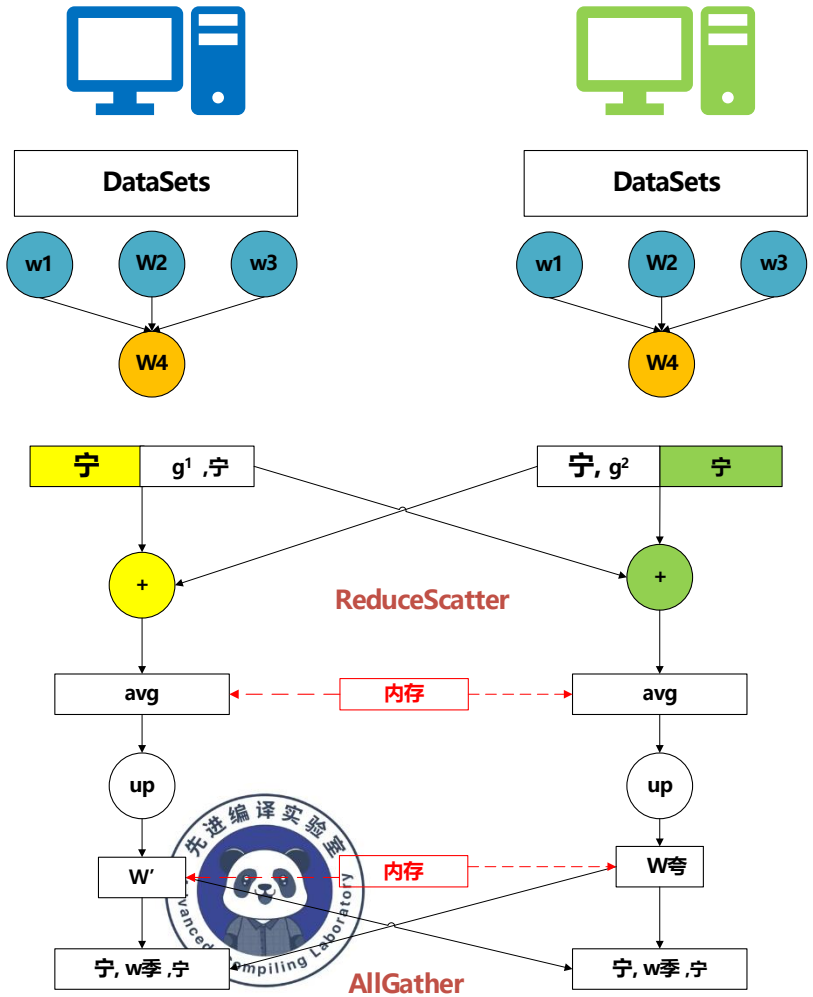
避免冗余计算


DataSets
w1
W2
w3
W4
DataSets
w1
W2
w3
W4
replicated
宁, g1 ,宁
宁, g2 ,宁
+
+
AllReduce
宁, avg ,宁
宁, avg ,宁
up
up
宁, w季 ,宁
宁, w季 ,宁
replicated
冗余计算
优化
DataSets
w1
W2
w3
W4
DataSets
w1
W2
w3
W4
宁
g1 ,宁
宁, g2
宁
+
+
ReduceScatter
avg
Sliced
avg
up
up
W'
W夸
宁, w季 ,宁
宁, w季 ,宁
AllGather

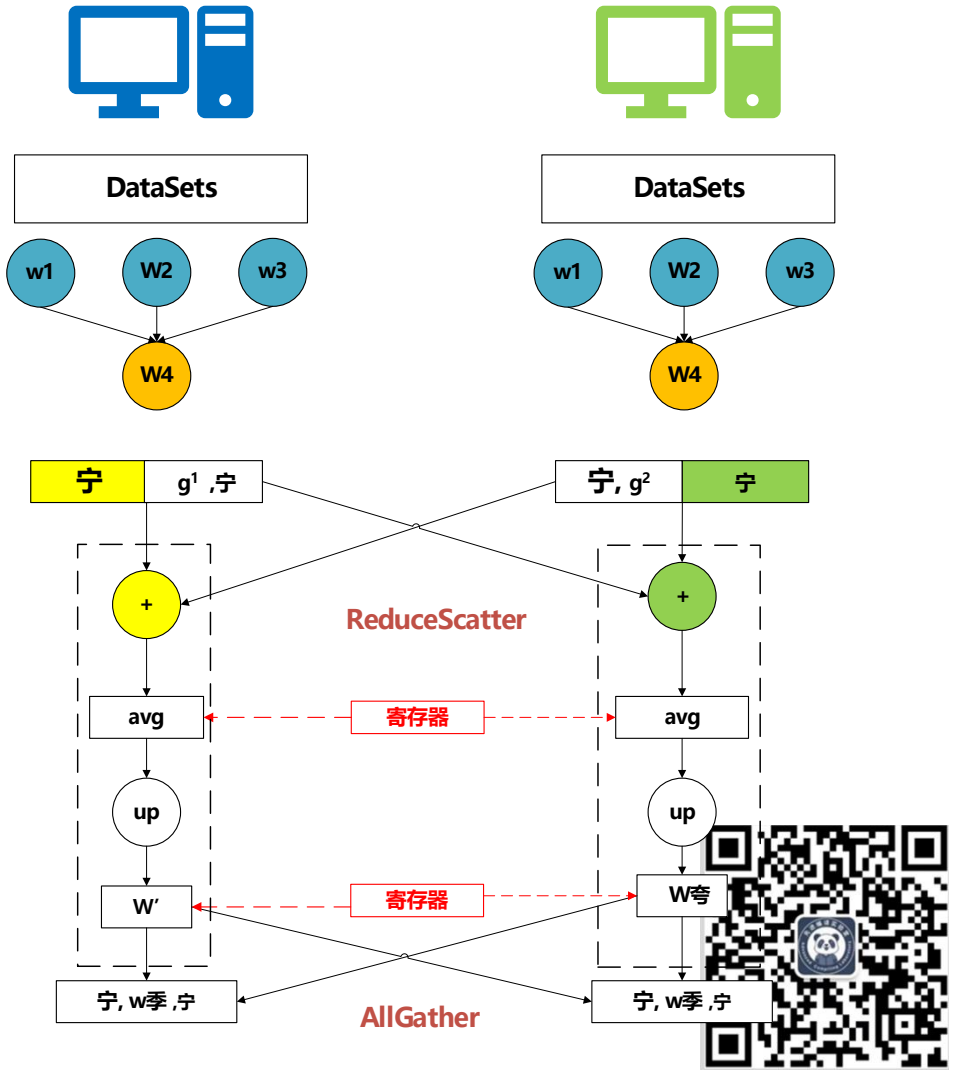
避免冗余计算


GPU 0
GPU 1
in
w
MatMul
layer
AllReduce
sum
b
Add&
Dropout
r
Add
output
GPU 0
GPU 1
in
w
MatMul
layer
ReduceScatter
rsSum
b
Add&
Dropout
r
Add
SlicedOut
AllGather
output

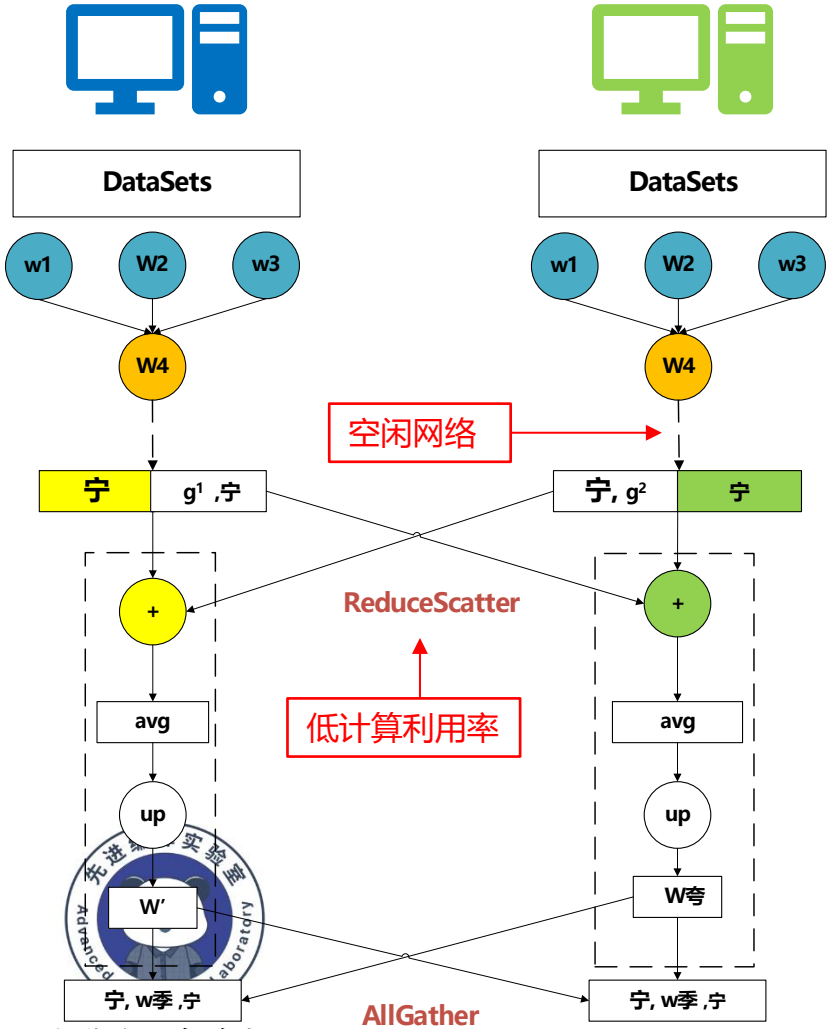
无用的内存访问


DataSets
DataSets
w1
W2
w3
W4
宁
g¹ ,宁
宁, g²
宁
+
+
ReduceScatter
avg
内存
avg
up
up
W'
内存
W夸
宁, w季 ,宁
宁, w季 ,宁
AllGather

融合

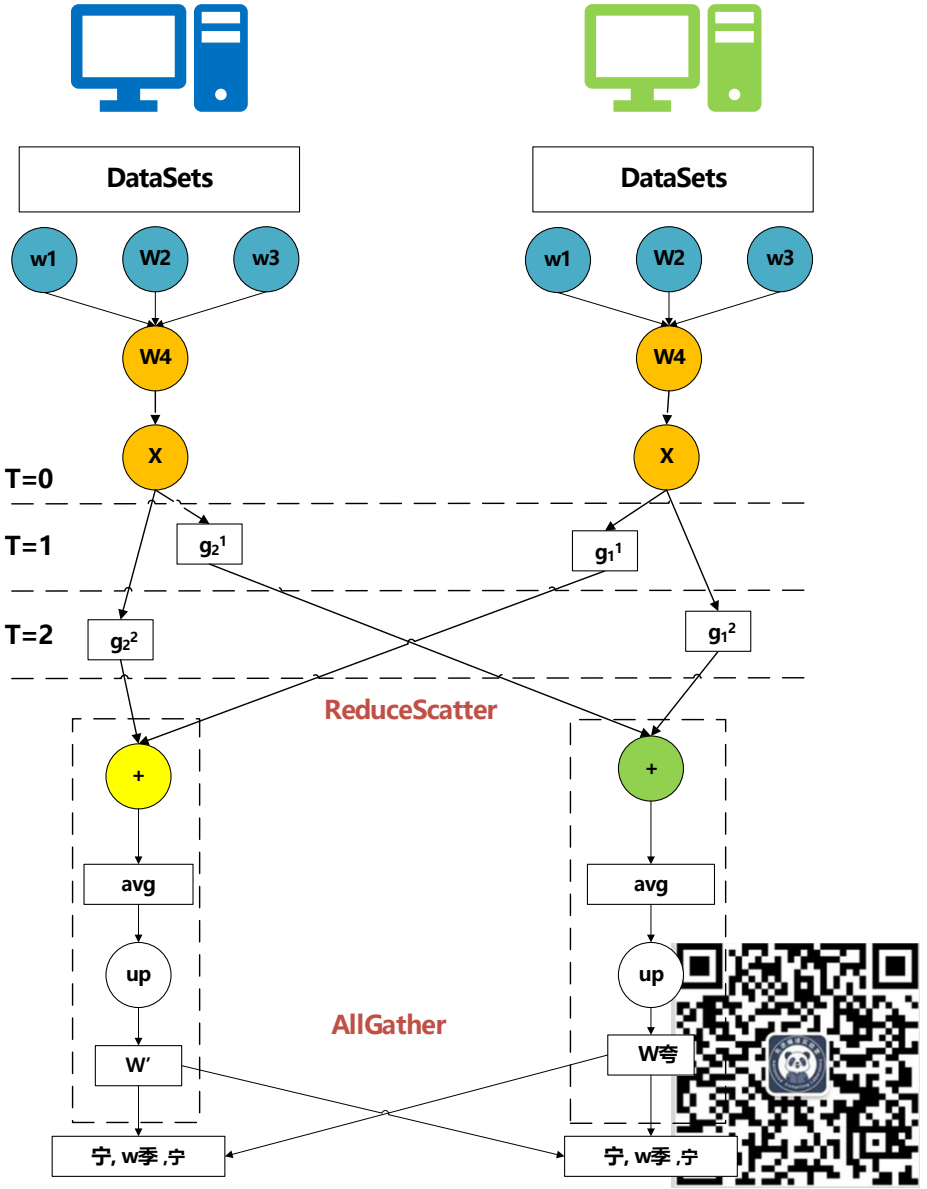

DataSets
DataSets
w1
W2
w3
W4
宁
g¹ ,宁
宁, g²
宁
+
+
ReduceScatter
avg
寄存器
avg
up
up
W'
寄存器
W夸
宁, w季 ,宁
宁, w季 ,宁
AllGather

计算和网络资源的不充分利用

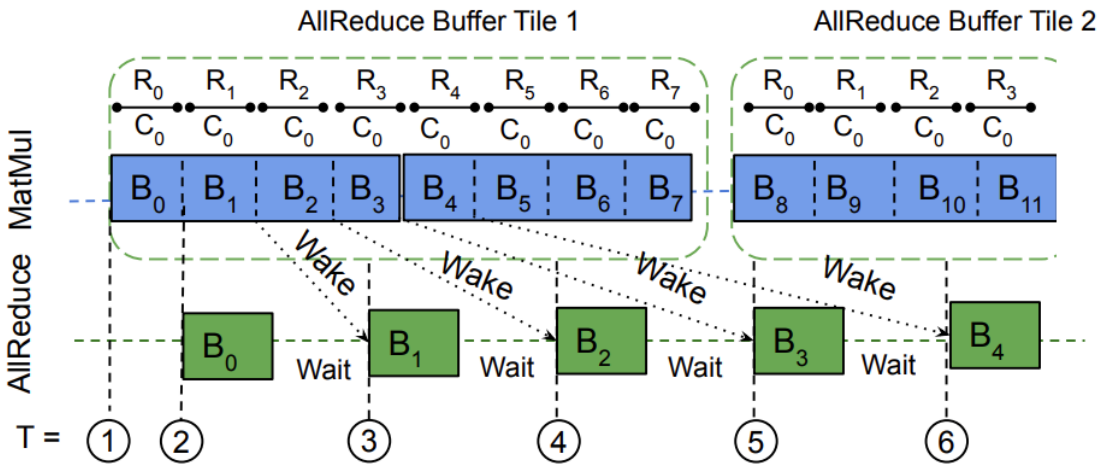


先进编译实验室
Advanced Compiler

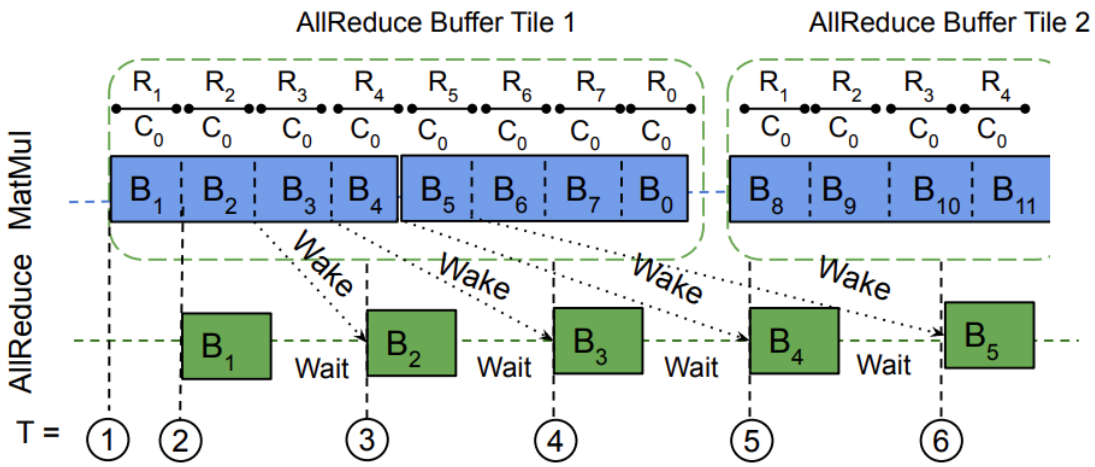
overlapping



计算和网络资源的不充分利用



(a) Workflow of overlap on rank 0. Rank 0 starts with chunk 0.



(b) Workflow of overlap on rank 1. Rank 1 starts with chunk 1.

目录

作者使用的评估环境：

一个由16个NVIDIA DGX-2节点组成的集群，并且每个节点包含了两个 24 核 Intel Xeon （至强）CPU 和 16 个 NVIDIA Tesla V100 (32GB) GPU。

评估测试项：

- 对三个不同的参数配置的BERT模型进行数据并行训练
- BERT和GPT-2的模型并行推理
- GPT-2和GPT-3的管道并向推理

数据并行训练

右图展示了CoCoNet在训练三个BERT模型中相对于基准测试提供的加速比

模型	Speedup of CoCoNet over		
	NV BERT	PyTorch DDP	ZeRo
BERT 336M	1.18X	1.22X	1.10X
BERT 1.2B	1.53X	1.52X	1.10X
BERT 3.9B	—	—	1.22X

模型并行训练

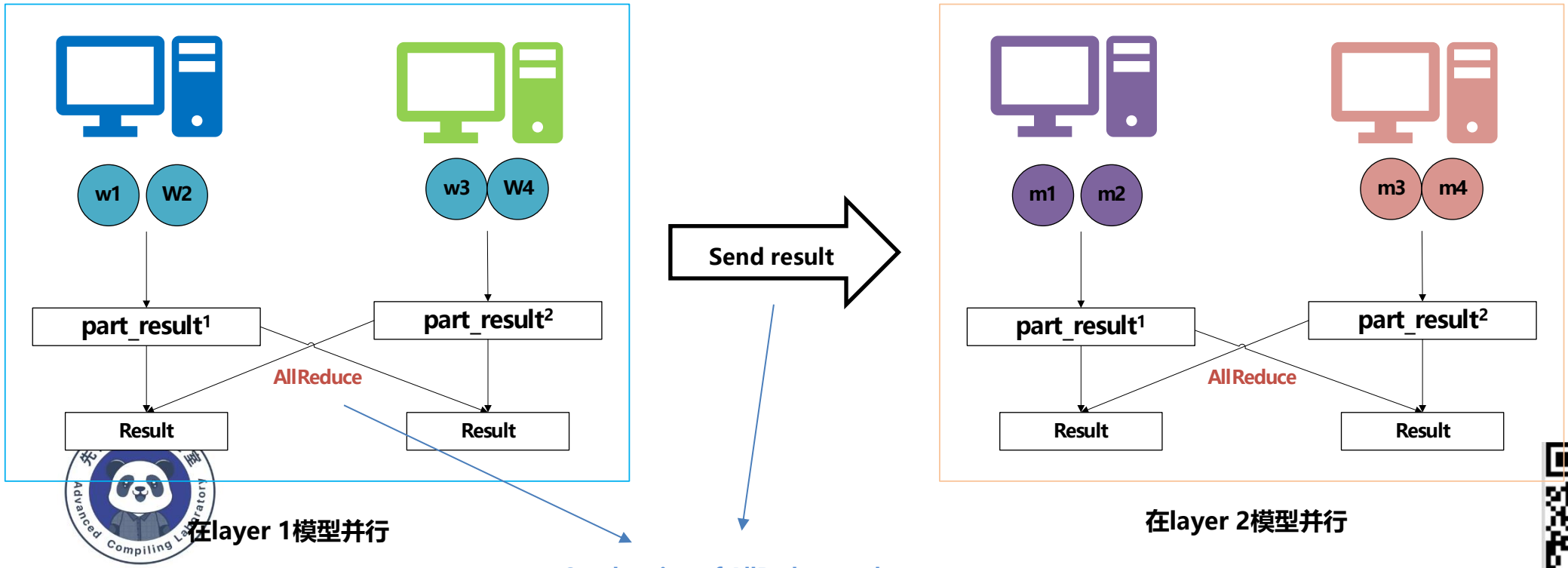
借助于计算和AllReduce通信源语之间的overlapping，CoCoNet在模型并行上也可以实现一定的加速效果。

模型	Speedup of CoCoNet over
	PyTorch DDP
BERT 3.9B	1.51X
GPT-2 8.3B	1.48X

pipeline并行训练

首先解释一下什么是管道并行推理：管道并行推理可以被视为一种模型并行方法，其中模型的不同层被分配到不同的节点上。

模型	Speedup of CoCoNet over
	PyTorch DDP
GPT-2 8.3B	1.77X
GPT-3 175B	1.33X



目录

1. Abhinav Jangda, Jun Huang, Guodong Liu, Amir Hossein Nodehi Sabet, Saeed Maleki, Youshan Miao, Madanlal Musuvathi, Todd Mytkowicz, and Olli Saarikivi. 2022. Breaking the computation and communication abstraction barrier in distributed machine learning workloads. In Proceedings of the 27th ACM International Conference on Architectural Support for Programming Languages and Operating Systems (ASPLOS '22). Association for Computing Machinery, New York, NY, USA, 402–416. https://doi.org/10.1145/3503222.3507778
2. https://www.youtube.com/watch?v=qQiZYJDiJ30&ab_channel=ACMSIGARCH

谢谢