Microsoft

人工智能系统 System for AI

深度学习中的分布式训练——算法

Distributed training algorithms

课程概要

分布式计算简介

深度学习并行训练简介

分布式训练算法分类

深度学习并行训练实例

分布式计算

从串行计算到并行计算

单处理器串行执行

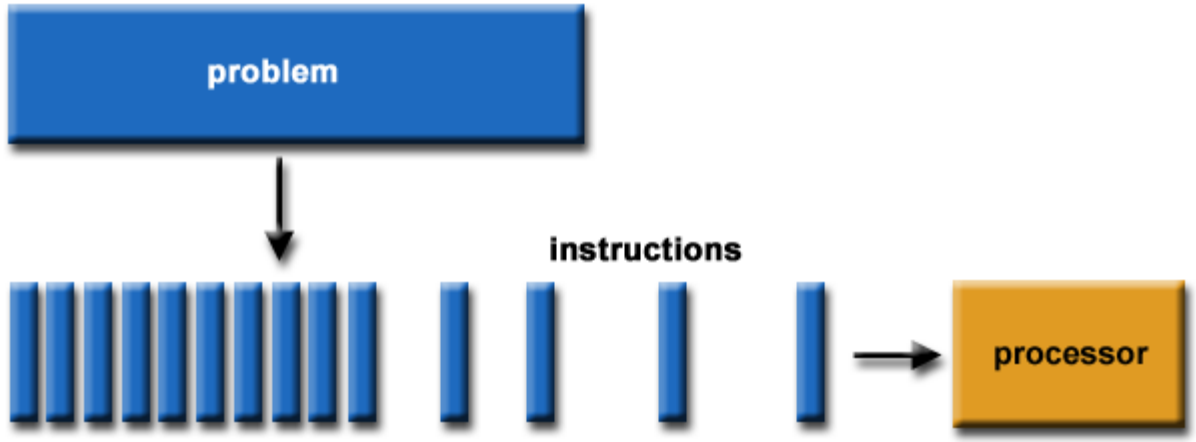
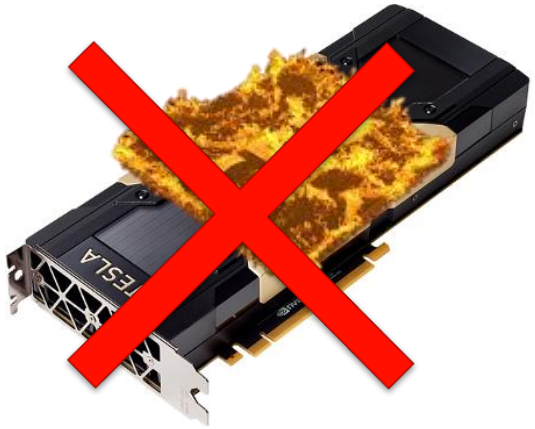


Figure: https://computing.llnl.gov/tutorials/parallel_comp/

从串行计算到并行计算

问题规模 (数据量+计算量)逐渐增大，单一设备处理速度无法满足



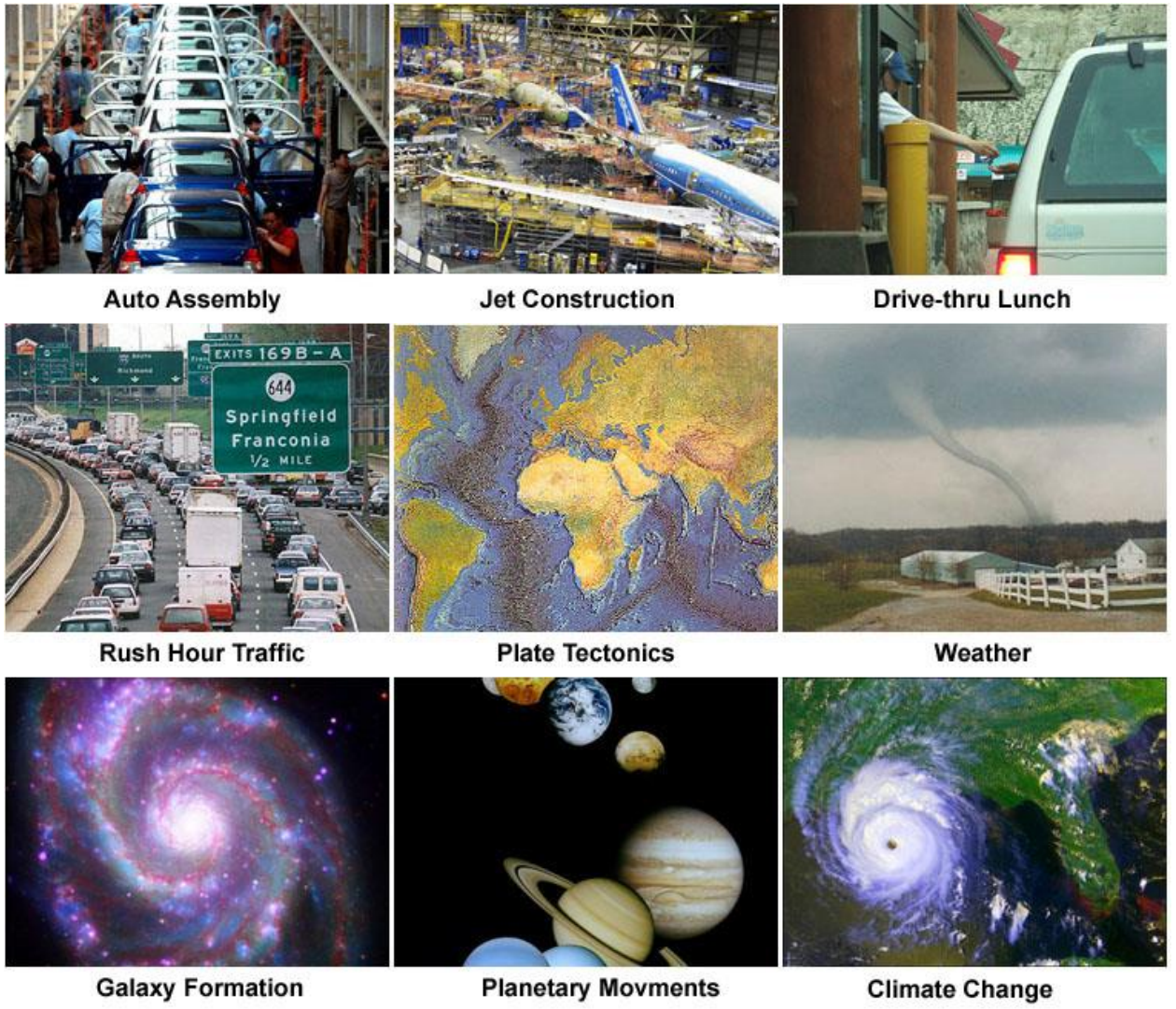


Figure: https://computing.llnl.gov/tutorials/parallel_comp/

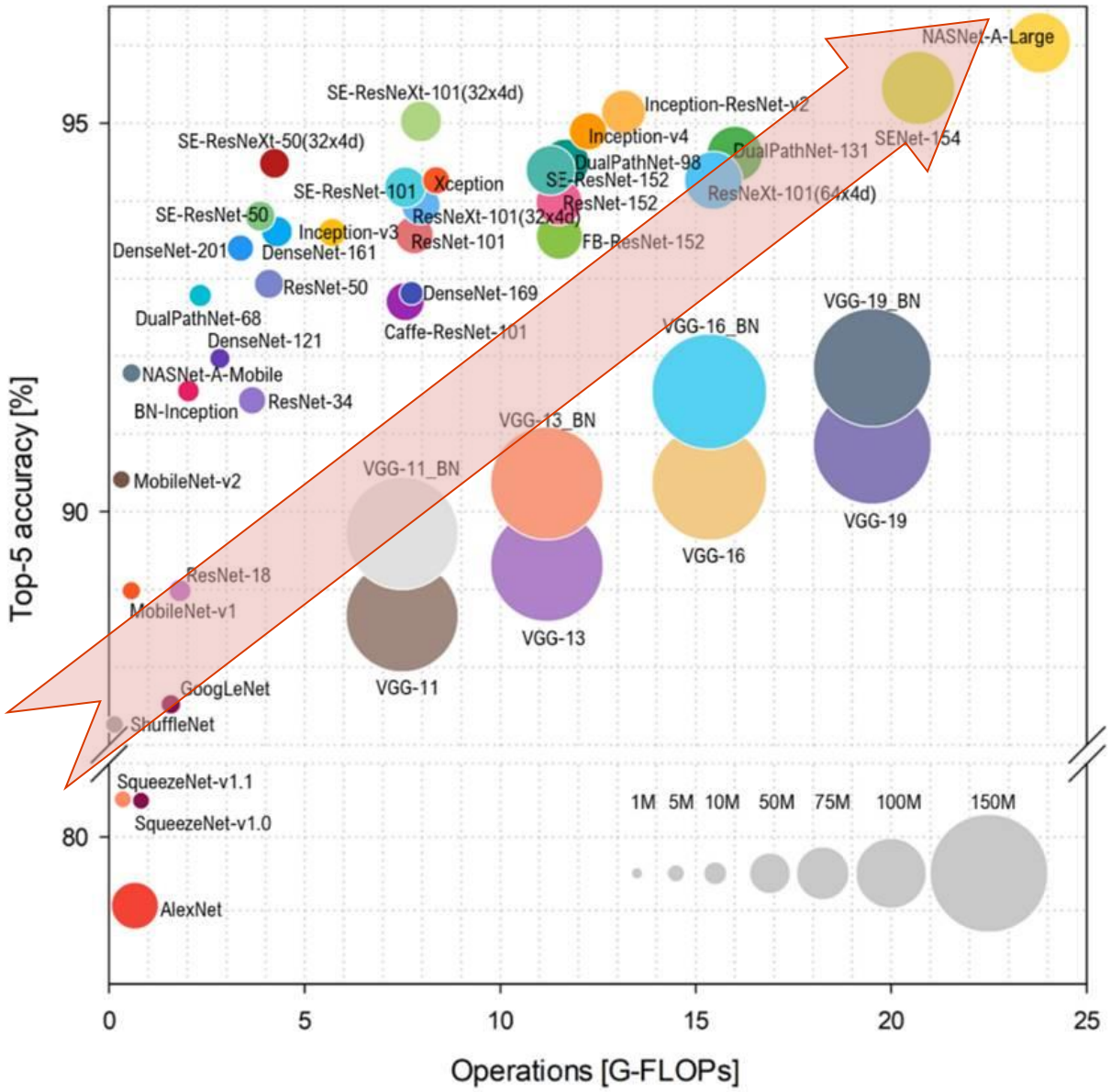
深度学习的计算复杂度

深度学习训练的趋势

更优的模型 -> 更大的计算复杂度

深度学习训练巨大的训练耗时

e.g. 语言模型 BERT(Large) 用 V100 GPU训练需时 >1个月



Top-5 accuracy vs. computational complexity (single forward pass)

Benchmark Analysis of Representative Deep Neural Network Architectures
https://arxiv.org/pdf/1810.00736.pdf

从串行计算到并行计算

多处理器并行执行

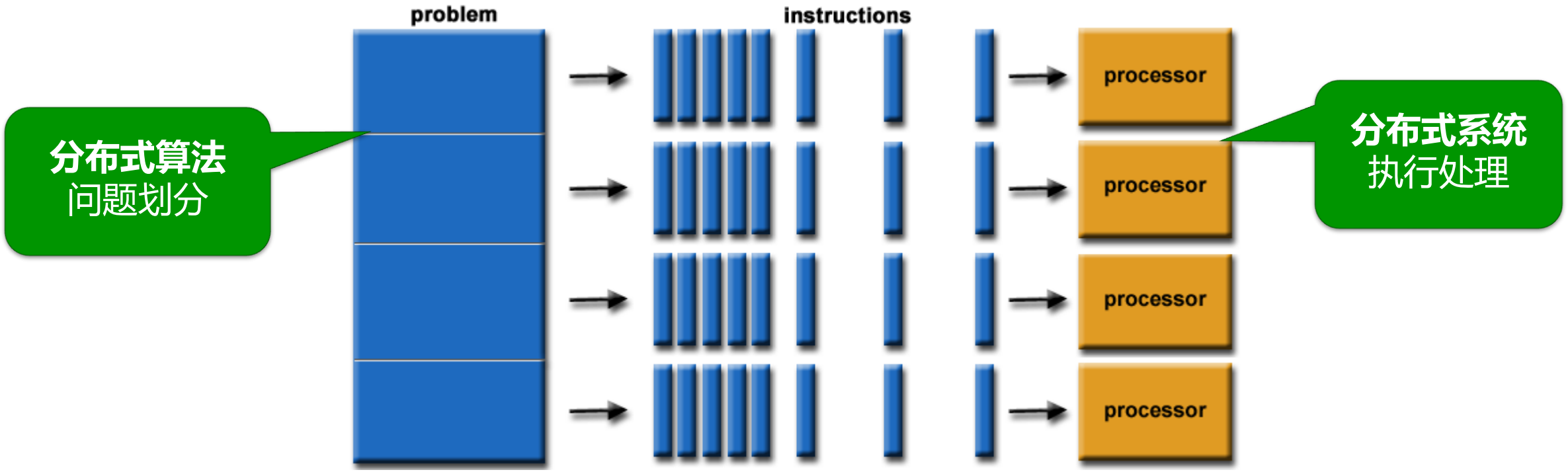


Figure: https://computing.llnl.gov/tutorials/parallel_comp/

分布式深度学习的意义

深度学习训练耗时：

训练数据规模 x 单步计算量 / 计算速率

模型相关，相对固定　　可变因素

计算速率：

单设备计算速率 x 设备数 x 并行效率

Moore定律，
相对有限

可变因素

工作重点

分布式算法与系统

分布式算法

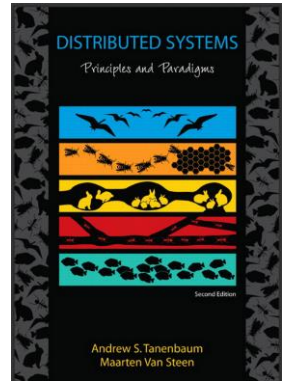
本章内容

"Distributed algorithms are algorithms designed to run on multiple processors, without tight centralized control." -- 《MIT 6.852J/18.437J》

分布式系统

下章内容

"A distributed system is a collection of independent computers that appears to its users as a single coherent system." -- 《Distributed Systems: Principles and Paradigms》



从串行训练
到并行训练

回顾：深度学习串行训练

TensorFlow

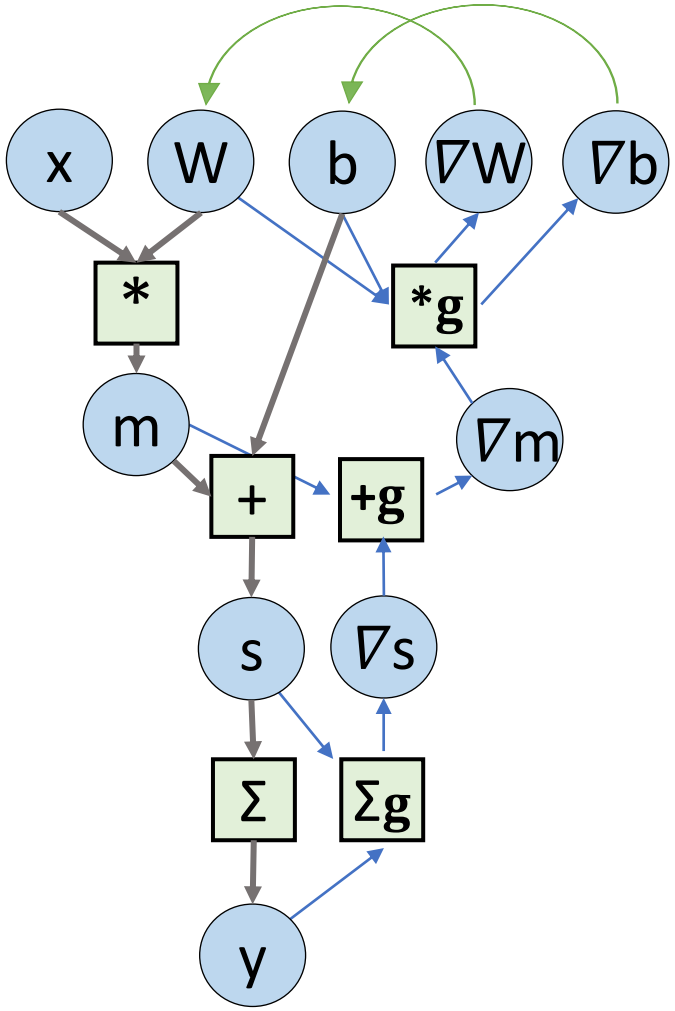
```
x = tf.placeholder(tf.float32)
W = tf.Variable(tf.float32)
b = tf.Variable(tf.float32)

m = W * x
s = m + b
y = tf.reduce_sum(s)


grad_W, grad_b = tf.gradients(y, [W, b])


update = optimizer.apply_gradients({[W, grad_W],
                                    [b, grad_b]})
```

Data-Flow Graph (DFG)



回顾：深度学习训练可并行潜力

TensorFlow

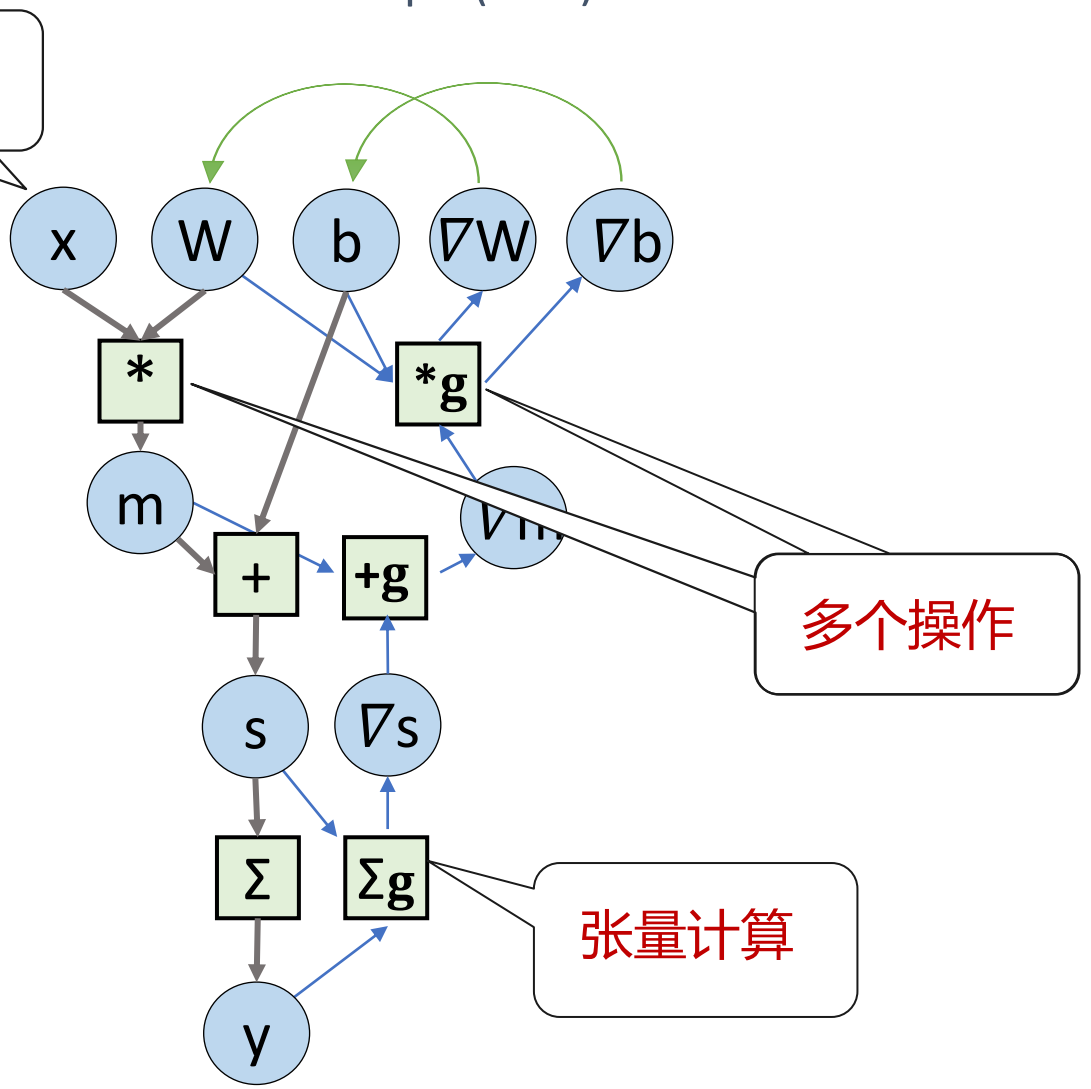
```
x = tf.placeholder(tf.float32)
W = tf.Variable(tf.float32)
b = tf.Variable(tf.float32)

m = W * x
s = m + b
y = tf.reduce_sum(s)


grad_W, grad_b = tf.gradients(y, [W, b])


update = optimizer.apply_gradients({[W, grad_W],
                                    [b, grad_b]})
```

Data-Flow Graph (DFG)



并行化的基本方案

算子内并行

算子并行：并行单个张量计算子内的计算（GPU多处理单元并行）

算子间并行

数据并行：多个样本并行执行

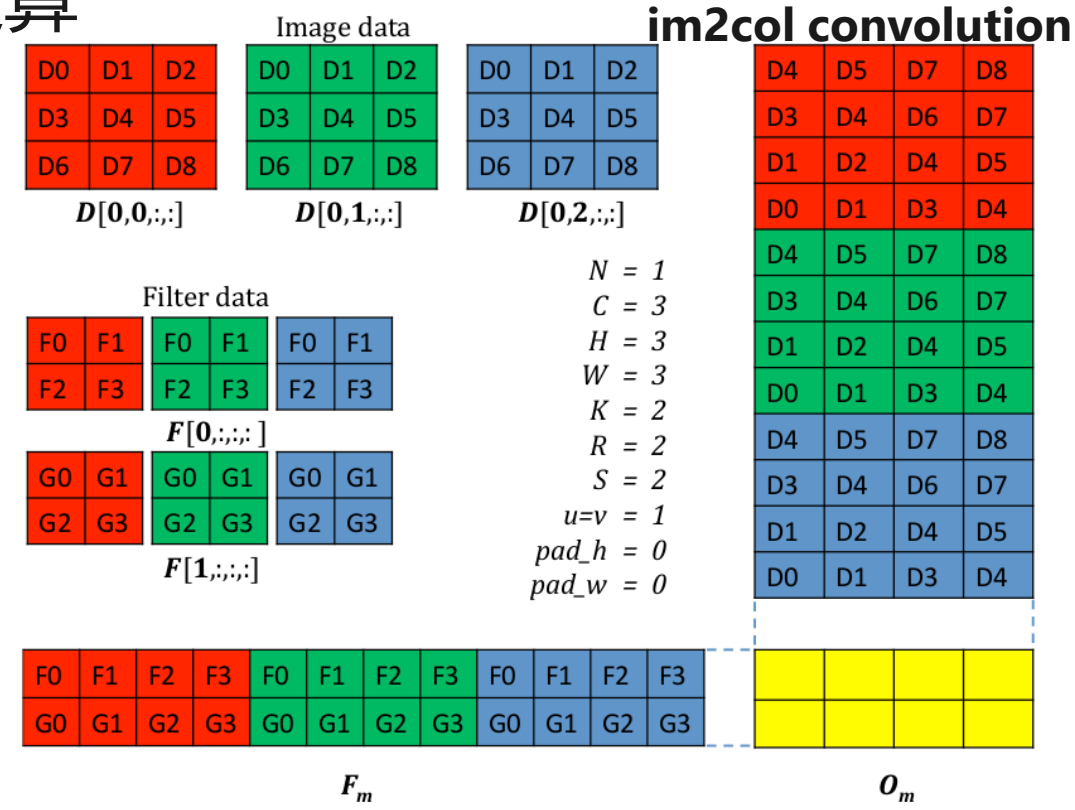
模型并行：多个算子并行执行

组合并行：多种并行方案组合叠加

https://blog.skymind.ai/distributed-deep-learning-part-1-an-introduction-to-distributed-training-of-neural-networks/

运算子内并行

- 利用线性计算和卷积等操作内部的并行性
- 多个并行维度
  - Batch, 空间维度, 时间维度, …
- 利用SIMD架构等多执行单元，同时运算



并行化的基本方案

算子内并行

算子并行：并行单个张量计算子内的计算（GPU多处理单元并行）

算子间并行

数据并行：多个样本并行执行

模型并行：多个算子并行执行

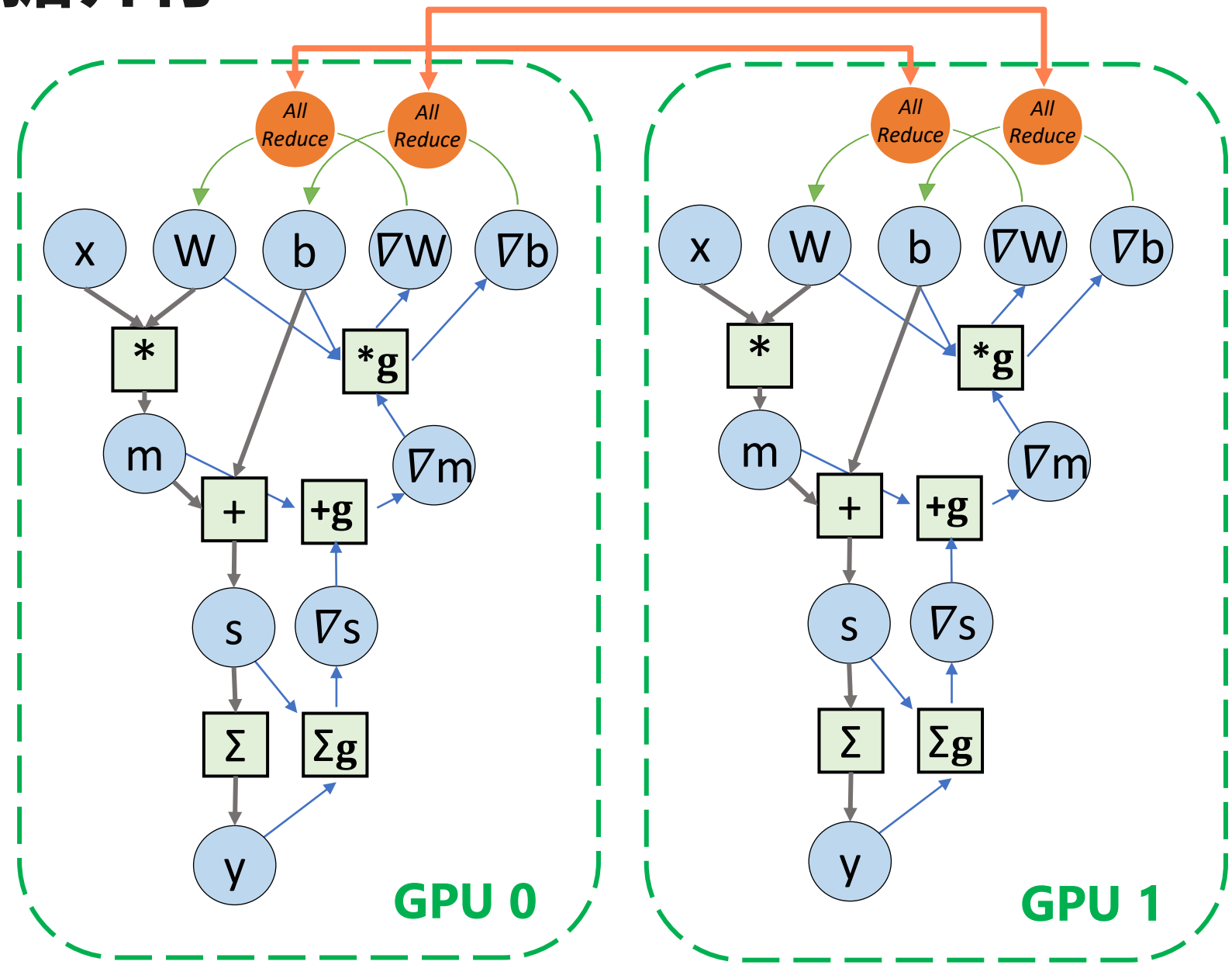
组合并行：多种并行方案组合叠加

https://blog.skymind.ai/distributed-deep-learning-part-1-an-introduction-to-distributed-training-of-neural-networks/

并行化方案——数据并行

步骤

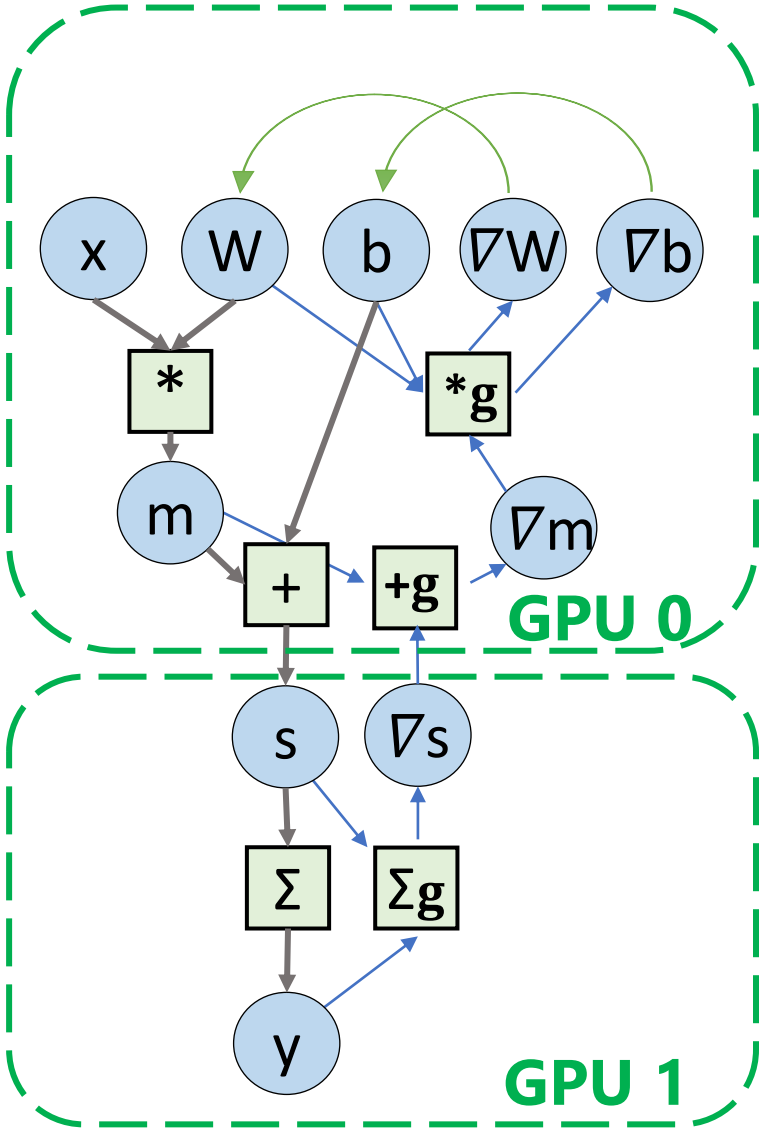
- 不同设备上执行相同计算图
- 跨设备聚合梯度
- 利用聚合后梯度更新模型



并行化方案——模型并行

步骤

- 计算图划分至不同设备上执行
- 跨设备传递激活
- 设备分别利用梯度更新模型的本地部分



数据/模型并行对比

	非并行	数据并行	模型并行
样本数据量	1	1/N	1
传输数据量	0	模型大小	激活大小
存储占用	1	N	1
负载平衡度	-	强	弱
并行限制	-	单步样本量	算子数量

设备数量：N

深度学习的分布式训练算法分类

并行化算法通信类别

同步并行

采用具有同步障的通信协调并行

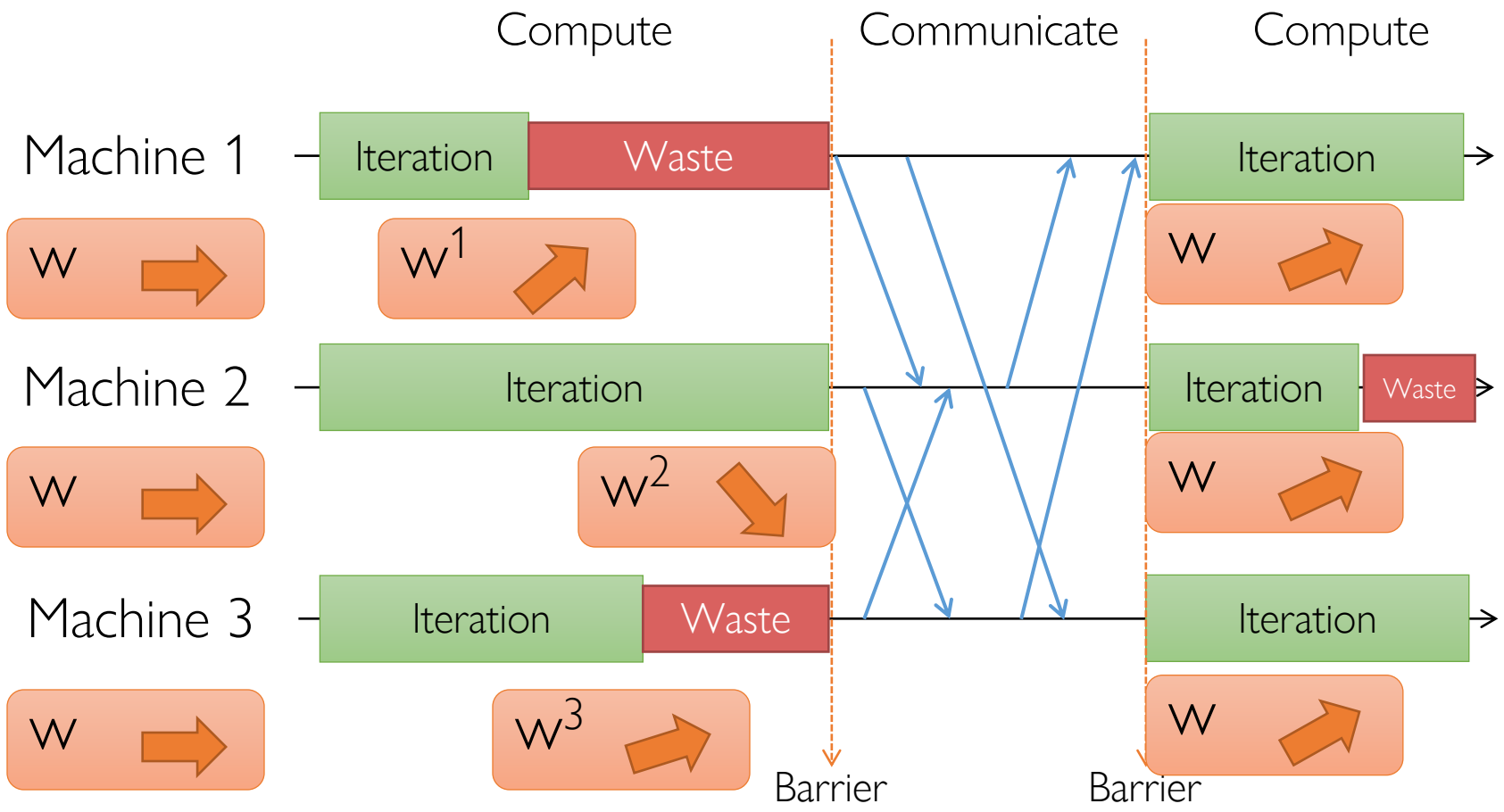
异步并行

采用不含同步障的通信协调并行

半同步并行

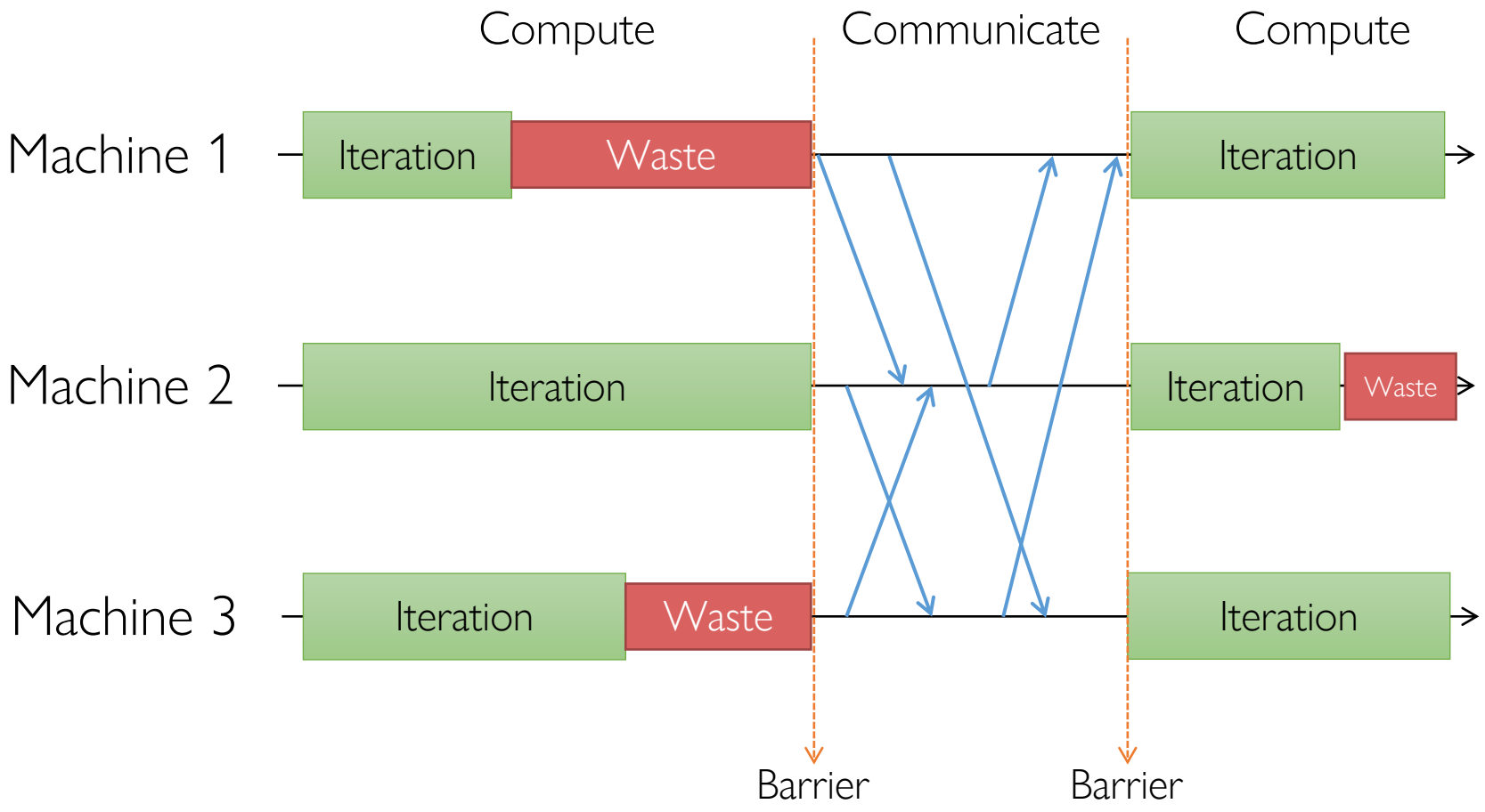
采用具有限定的宽松同步障的通信协调并行

同步算法



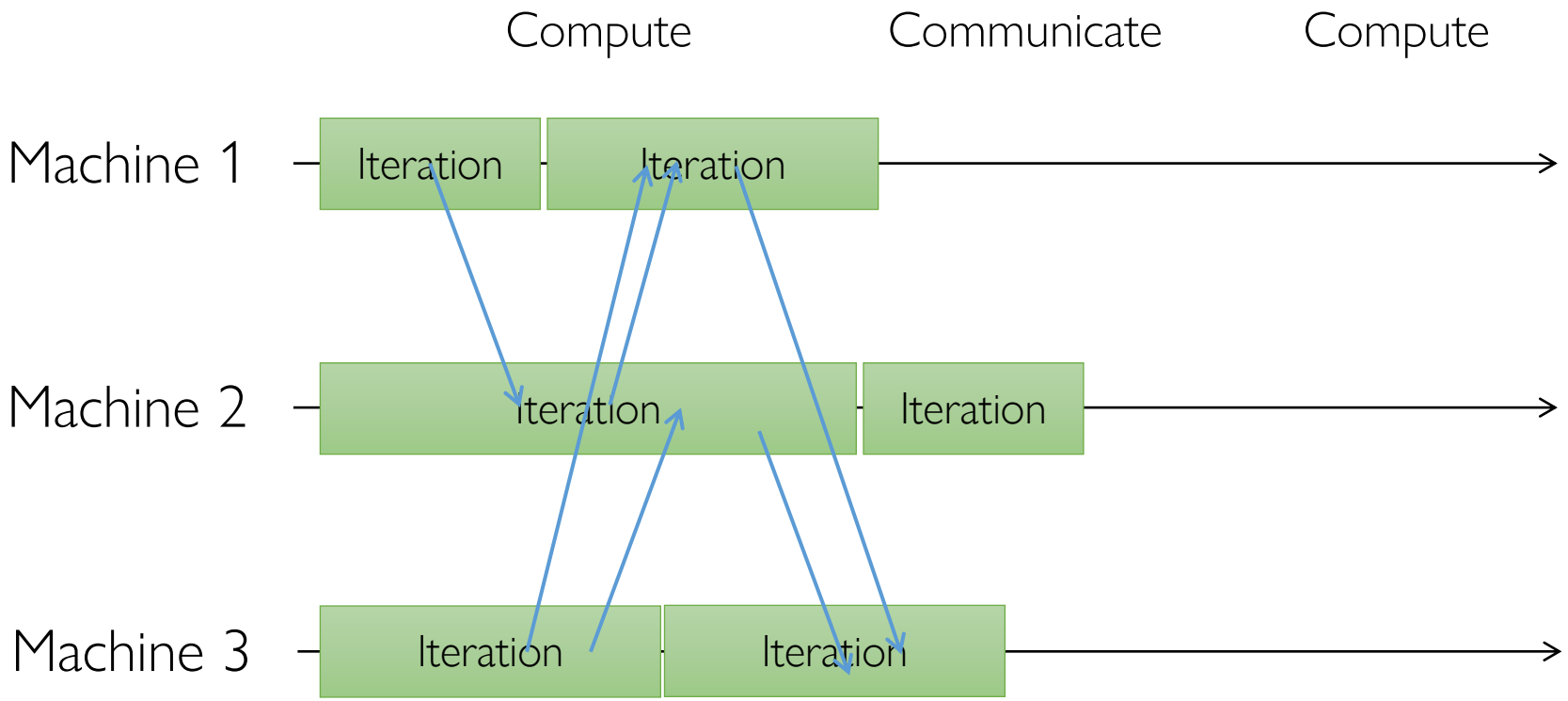
Joseph E. Gonzalez *AI-Systems Distributed Training*

异步算法



Joseph E. Gonzalez *AI-Systems Distributed Training*

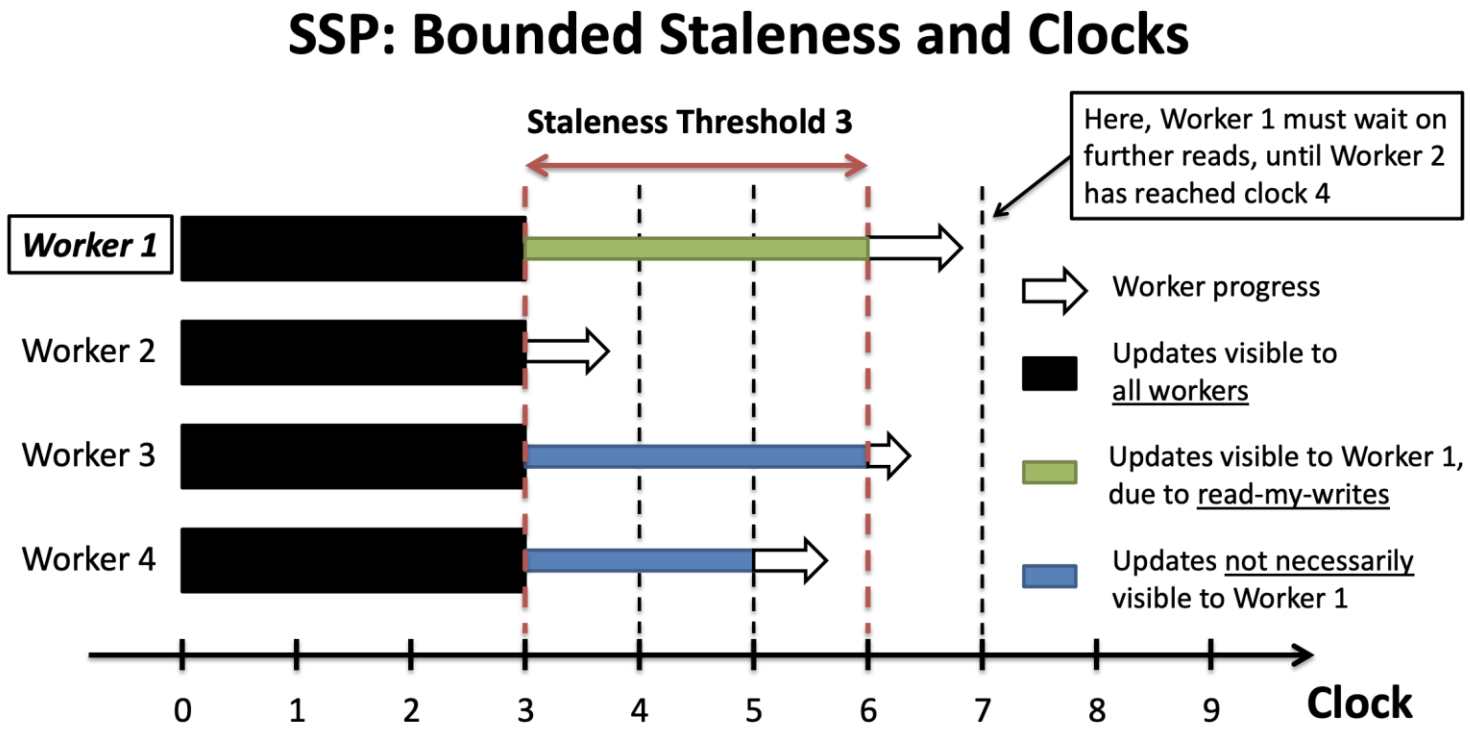
异步算法



Joseph E. Gonzalez *AI-Systems Distributed Training*

半同步算法

e.g., Stale Synchronous Parallel (SSP)



并行化算法通信类别

同步并行

采用具有同步障的通信协调并行　　收敛性AAA

异步并行

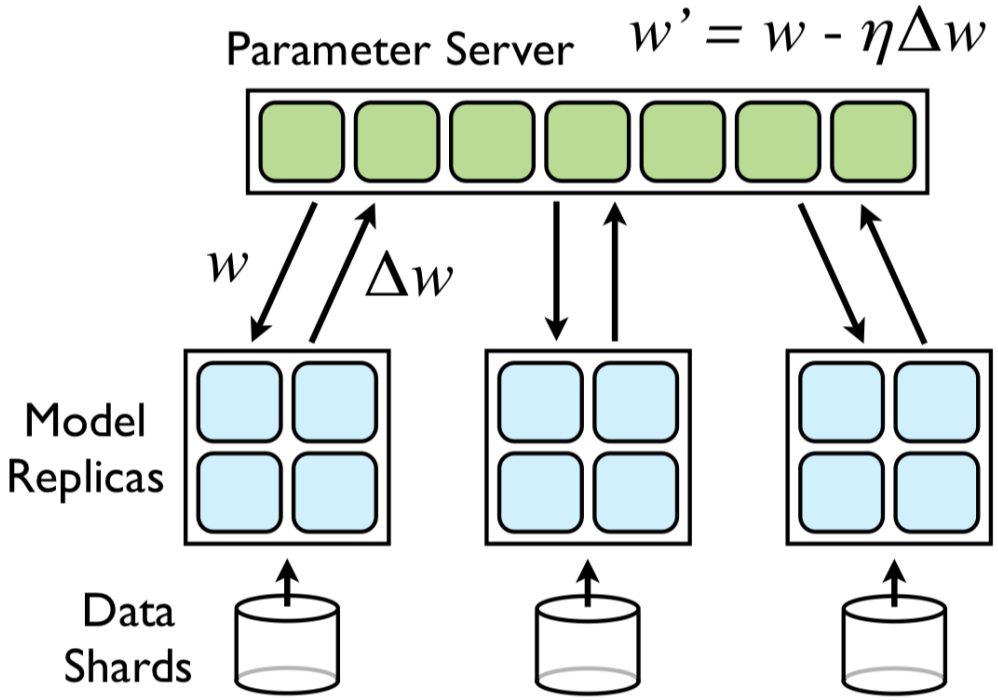
采用不含同步障的通信协调并行　　收敛性A

半同步并行

采用具有限定的宽松同步障的通信协调并行　　收敛性AA

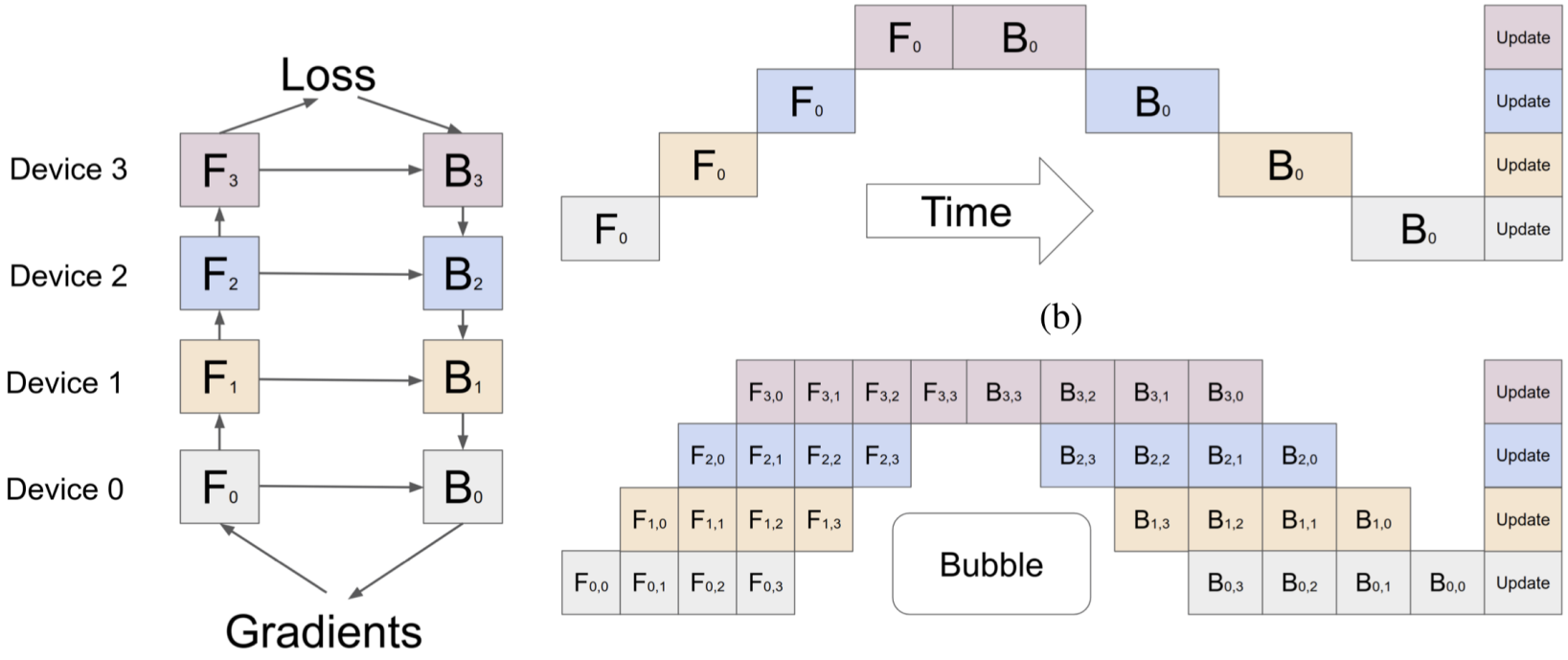
深度学习并行训练实例

Downpour SGD



GPipe

通过对minibatch进行拆分，减少设备空闲(Bubble)，从而更好地利用多设备进行流水化并行

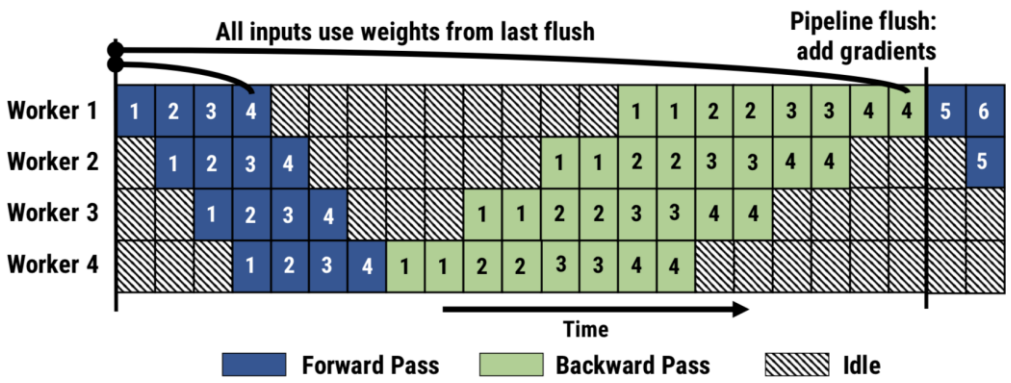


[GPipe: Easy Scaling with Micro-Batch Pipeline Parallelism]

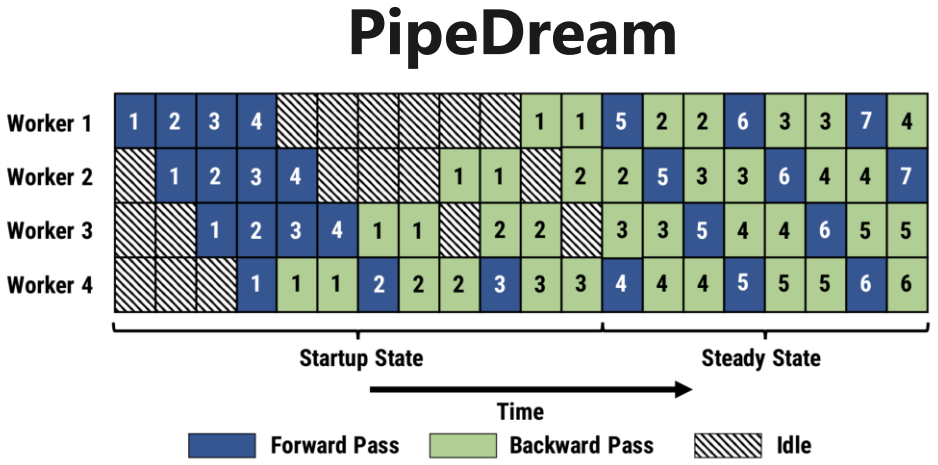
PipeDream

采用非同步机制，在Gpipe基础上进一步减少设备空闲

GPipe



PipeDream



本章小节

分布式是解决大计算量深度学习训练的有效方法

分布式深度学习算法按照并行维度可分为：

运算子/数据/模型/混合并行

分布式深度学习算法按照通讯协调方式包括：

同步并行/异步并行/半同步并行

阅读列表

- Scaling Distributed Machine Learning with the Parameter Server (OSDI'14)
  - Paper describing the parameter server system
- PipeDream: Generalized Pipeline Parallelism for DNN Training (SOSP'19)
  - Latest paper exploring pipeline parallel training
- Adaptive Communication Strategies to Achieve the Best Error-Runtime Trade-off in Local-Update SGD (SysML'19)
  - Dynamic averaging approach to distributed training

参考文献