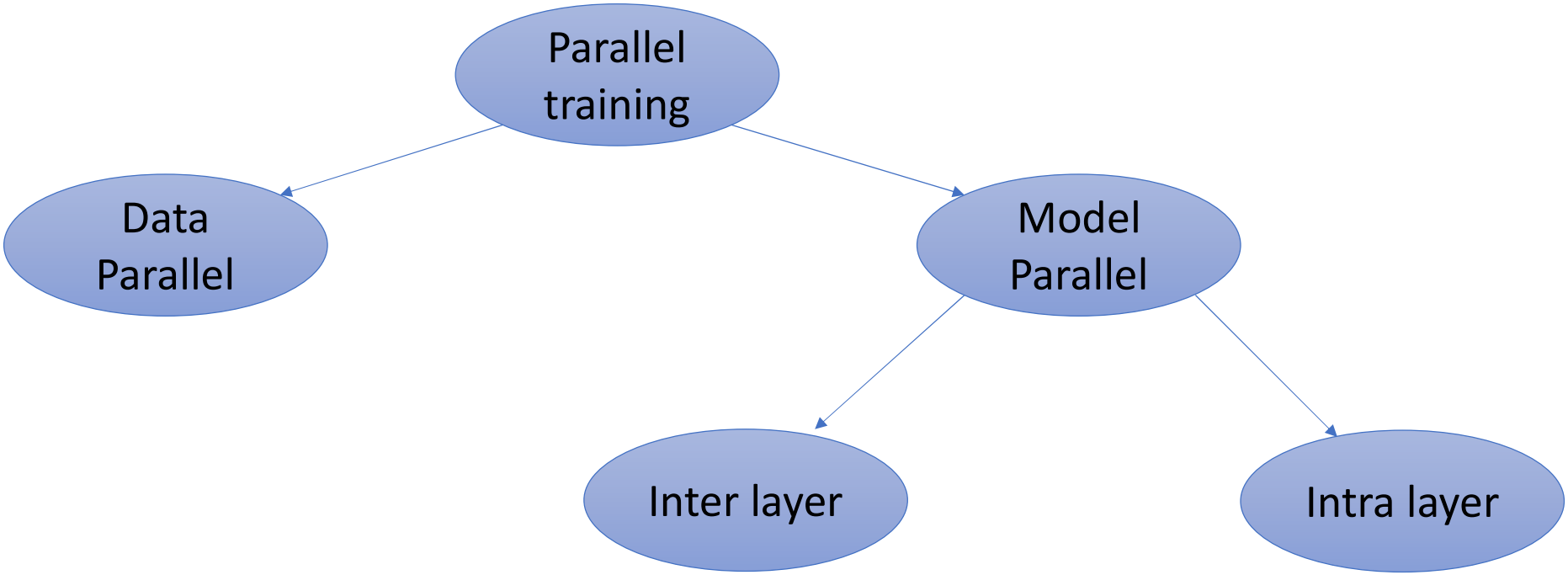
Распределенное обучение нейросетей

Михненко Наталья БПМИ182

Мотивация

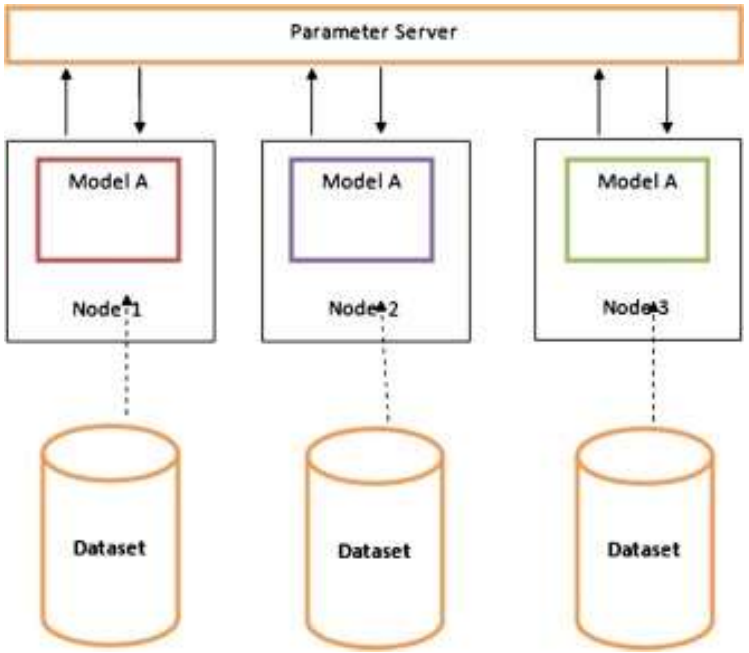
- Вычислительная задача обучения нейросети сложная
- Слишком много параметров
- Обучение может занимать месяцы

Стандартные подходы

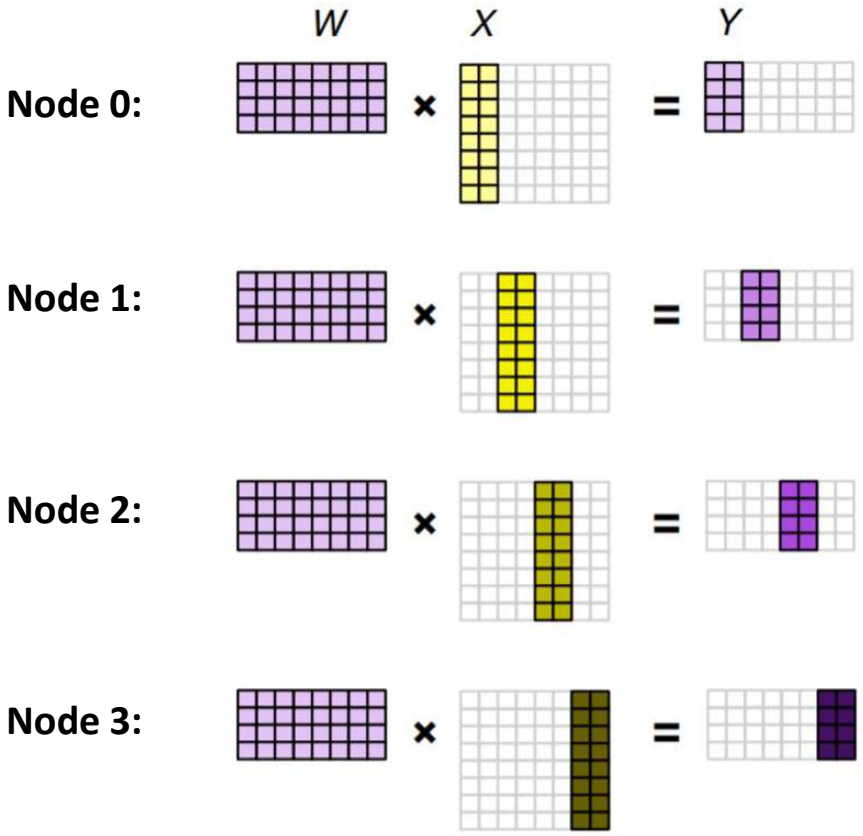


Data Parallel

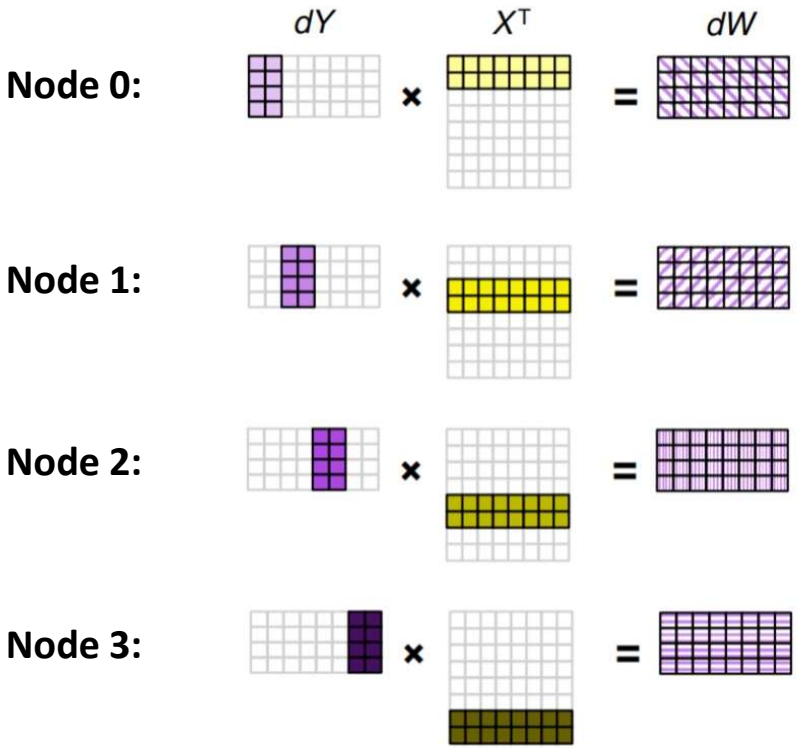
- Каждый узел получает полную копию всей нейросети и отвечает за вычисления только на части данных
- На прямом проходе каждый считает входные активации для своей части минибатчей
- На обратном проходе нужны коммуникации между узлами



Data Parallel: Forward Pass



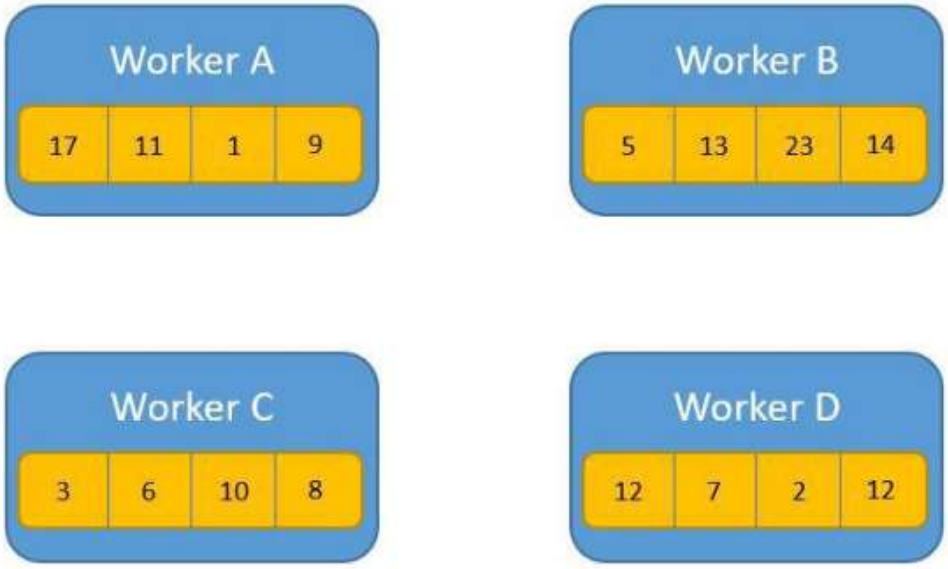
Data Parallel: Backward Pass



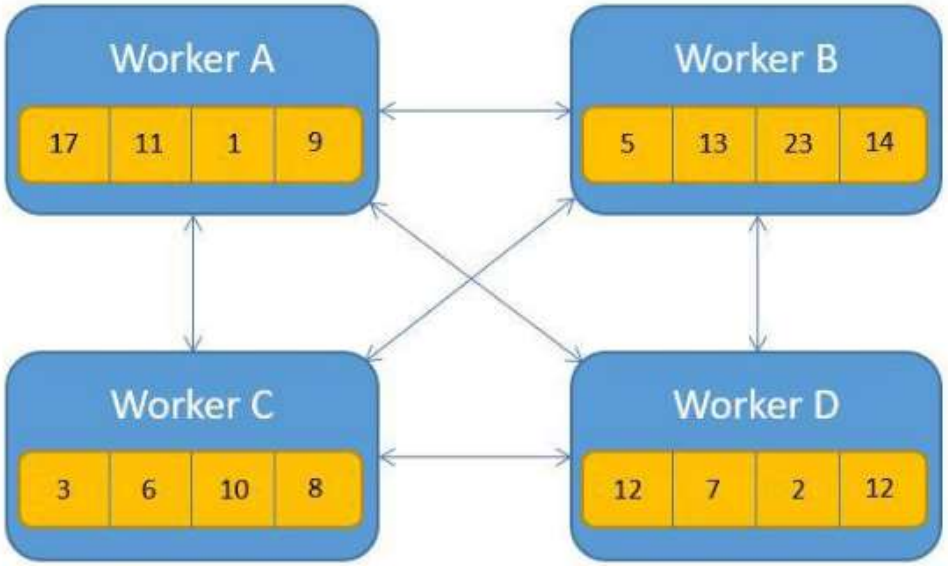
Data Parallel: Communications

- Allreduce – один из самых популярных видов коммуникации
- После распространения на всех узлах одинаковые данные
- Коммуникация = дополнительные расходы времени

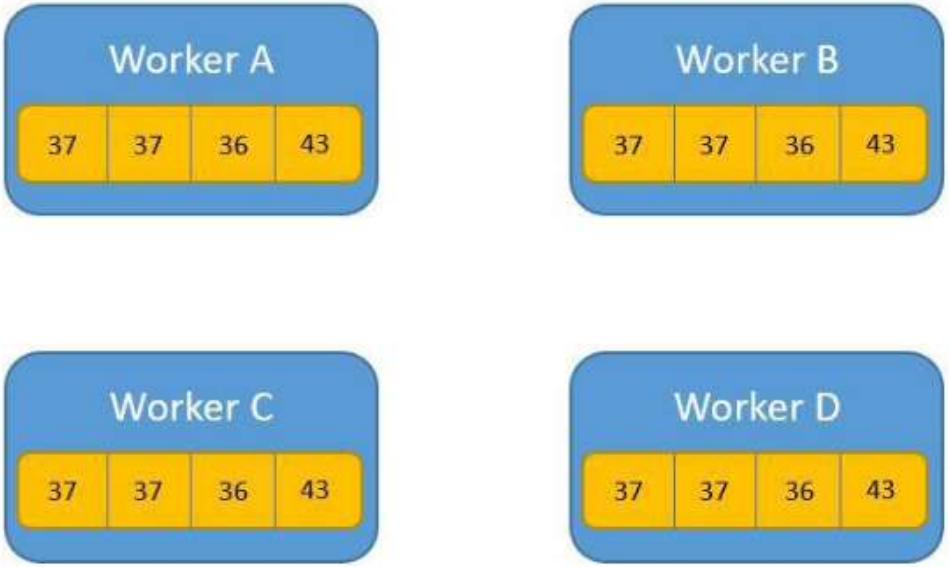
Communication



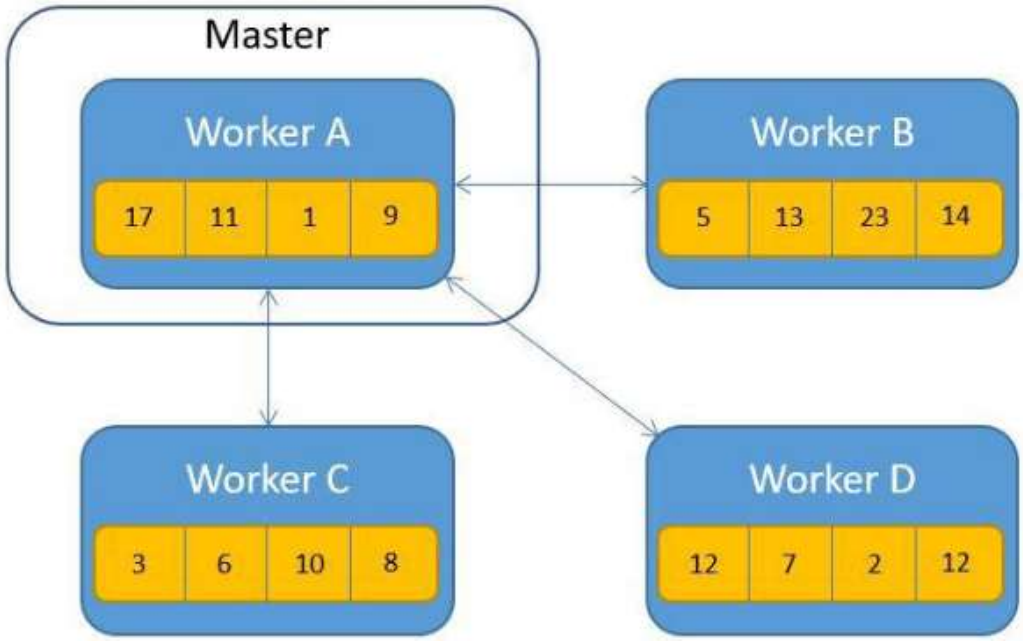
Communication



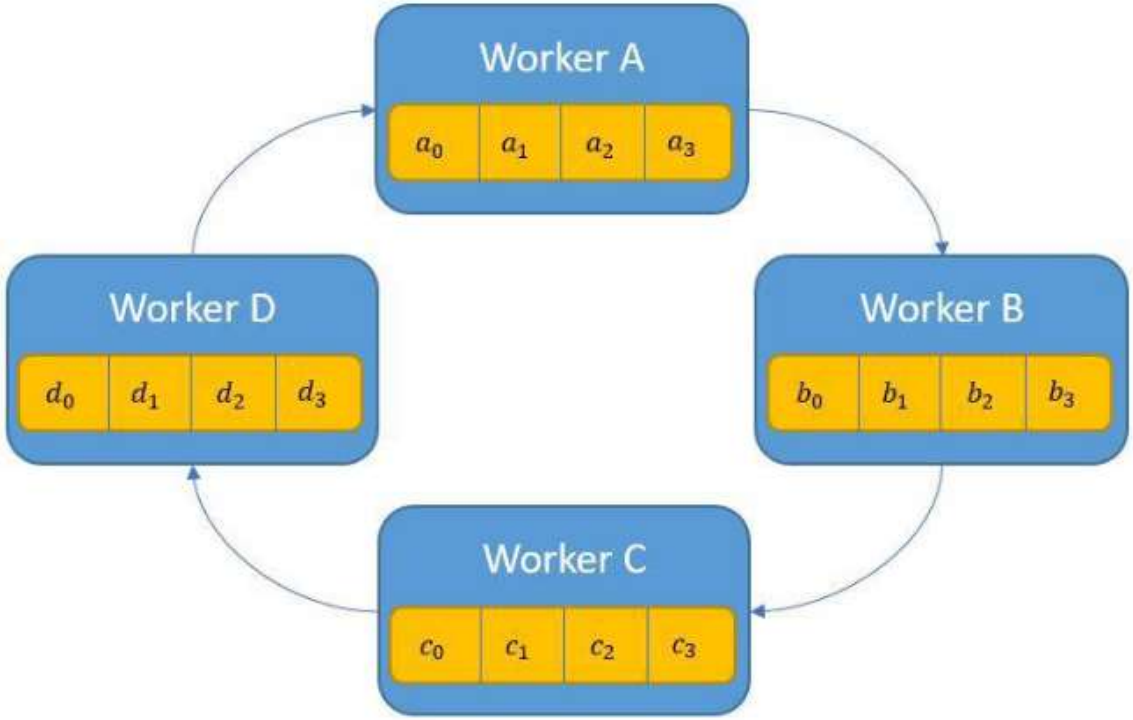
Communication



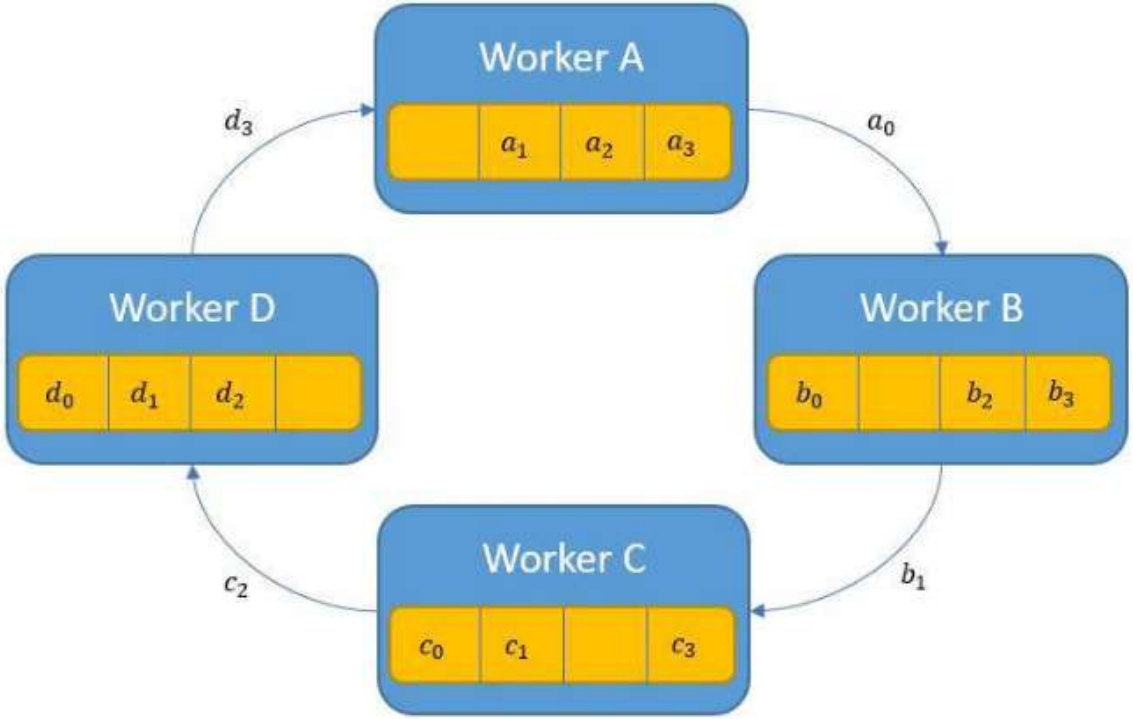
Communication



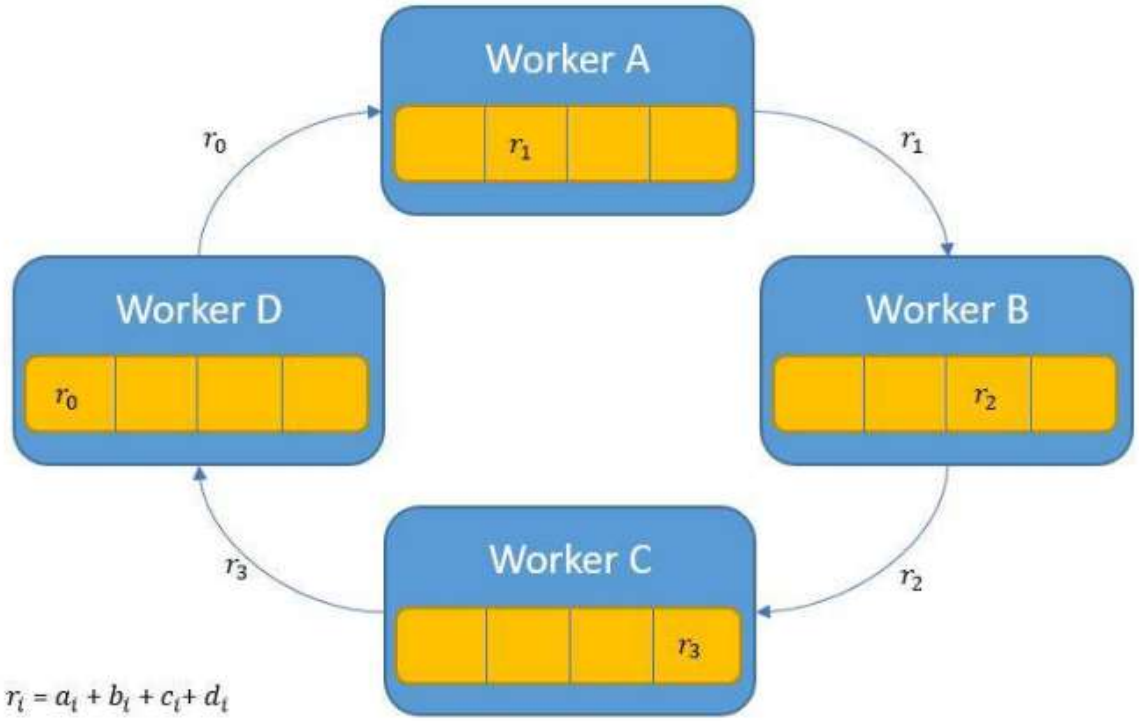
Communication



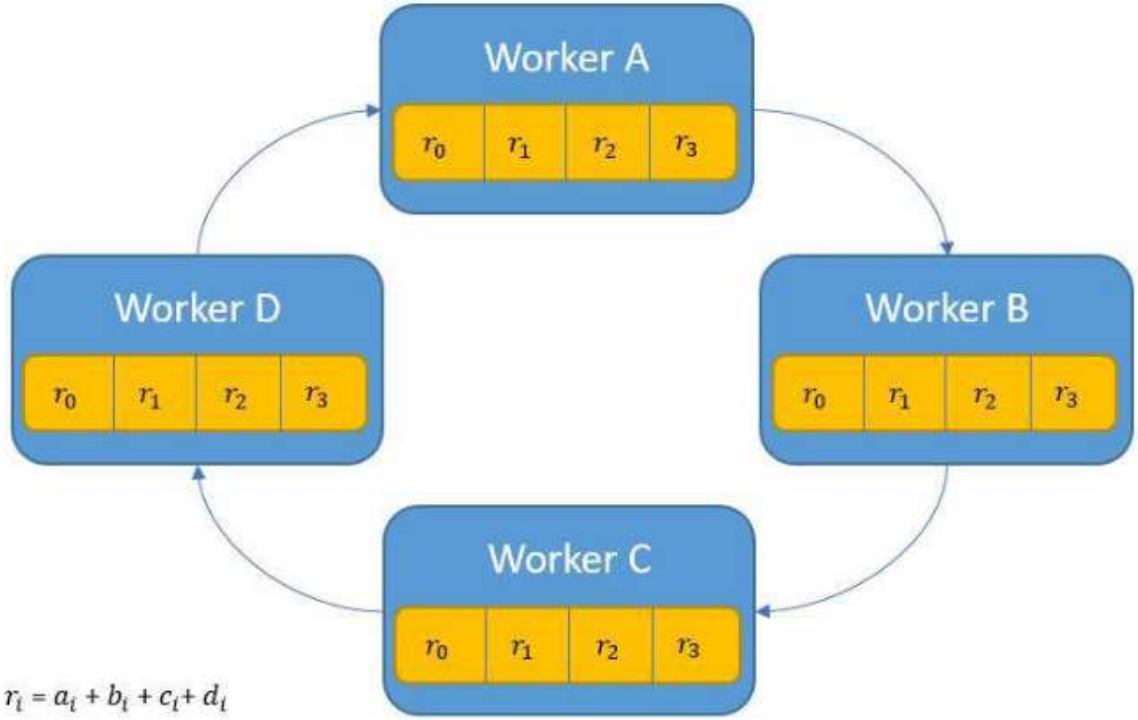
Communication



Communication



Communication



Model Parallel

Существует 2 вида параллелизма:

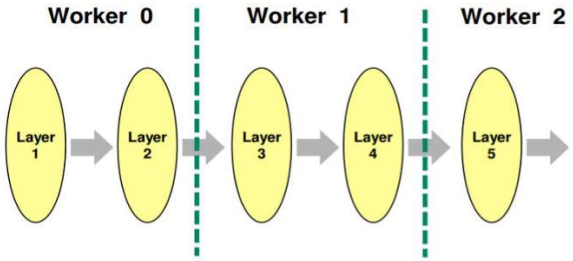
1. Каждый слой делится между несколькими узлами
2. Слои делятся между узлами

Редко применяется в обучении нейронных сетей, так как

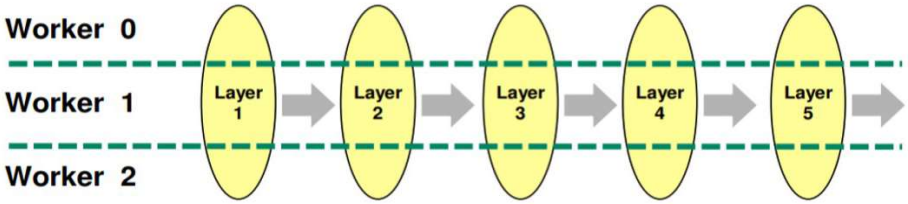
1. Как разделить модель между узлами – сложная задача
2. Происходит недозагрузка узлов

Model Parallel

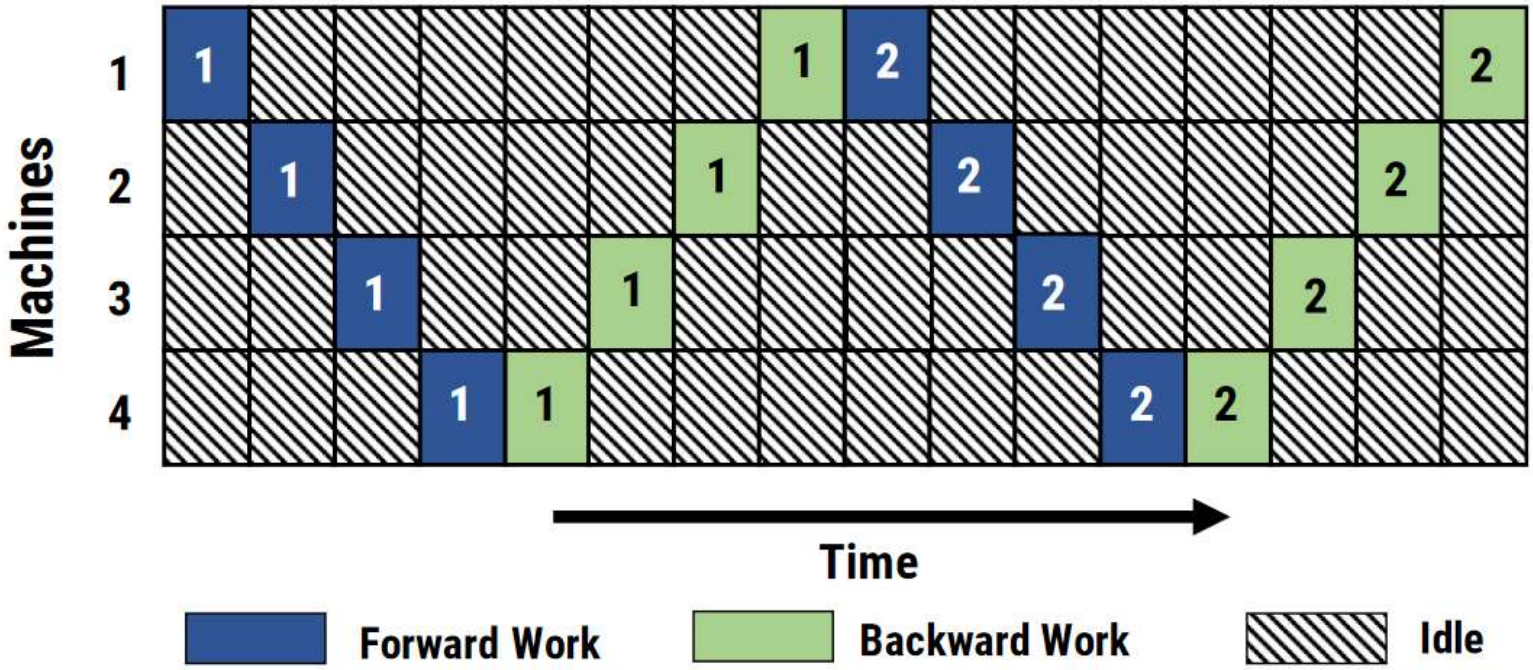
Inter Layer



Intra Layer

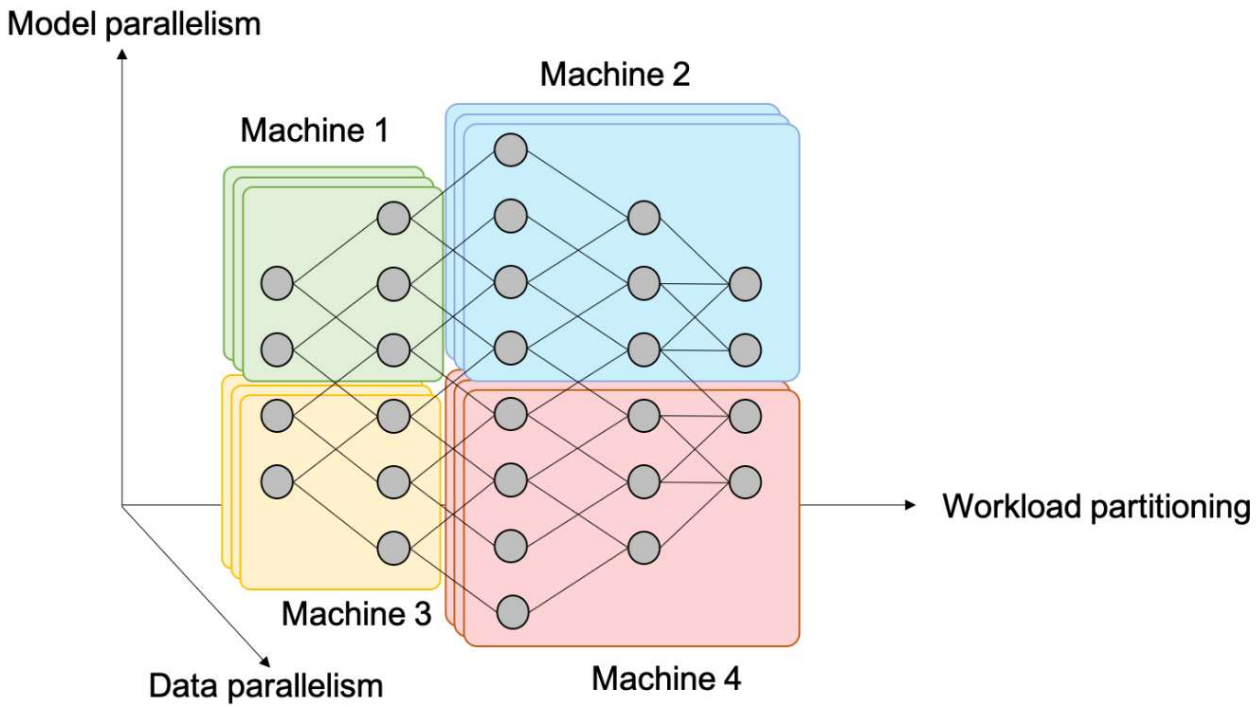


Model Parallel



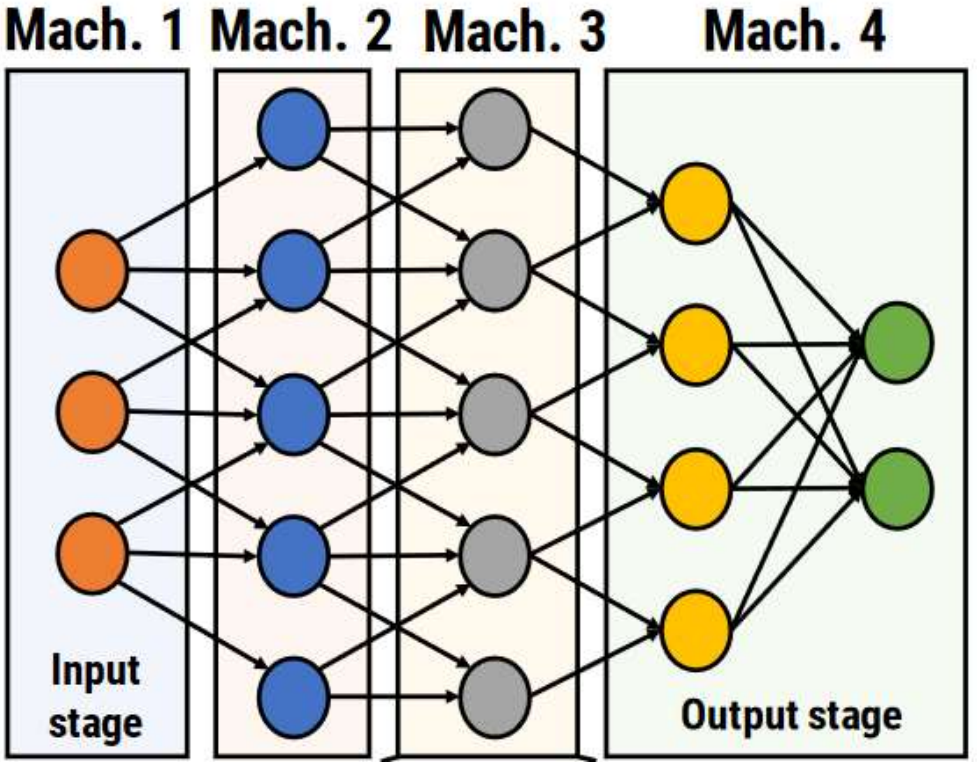
PipeDream

- PipeDream – разработка Microsoft
- Совмещает параллелизм данных и модели



Pipeline Parallel

- Разбиваем сеть на фазы
- Каждая фаза выполняет как прямой проход, так и обратный
- Чтобы ни один узел не простаивал, минибатчи вводим один за другим, не дожидаясь конца прямого прохода



Pipeline Parallel

Преимущества перед параллелизмом данных:

- Требуется меньше коммуникаций
- Позволяет совмещать коммуникацию и вычисления

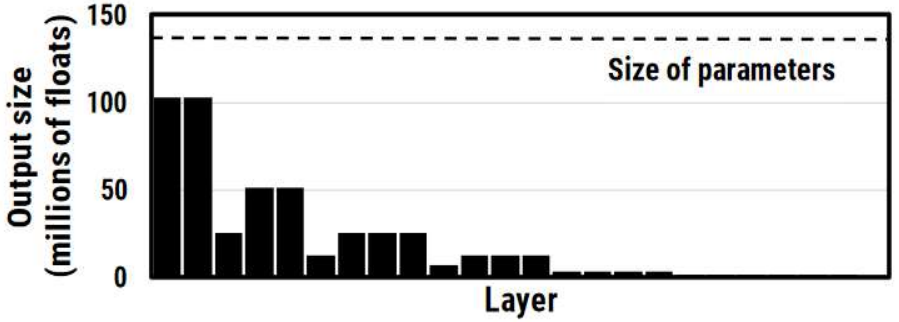
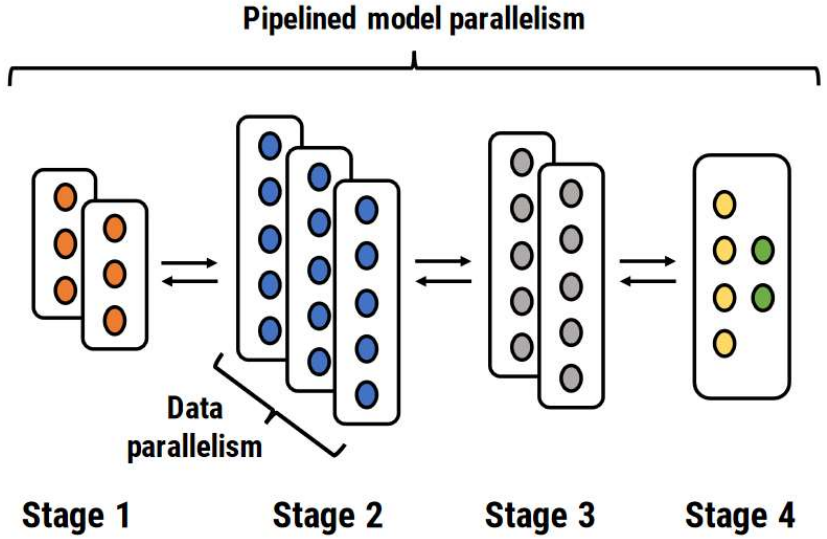


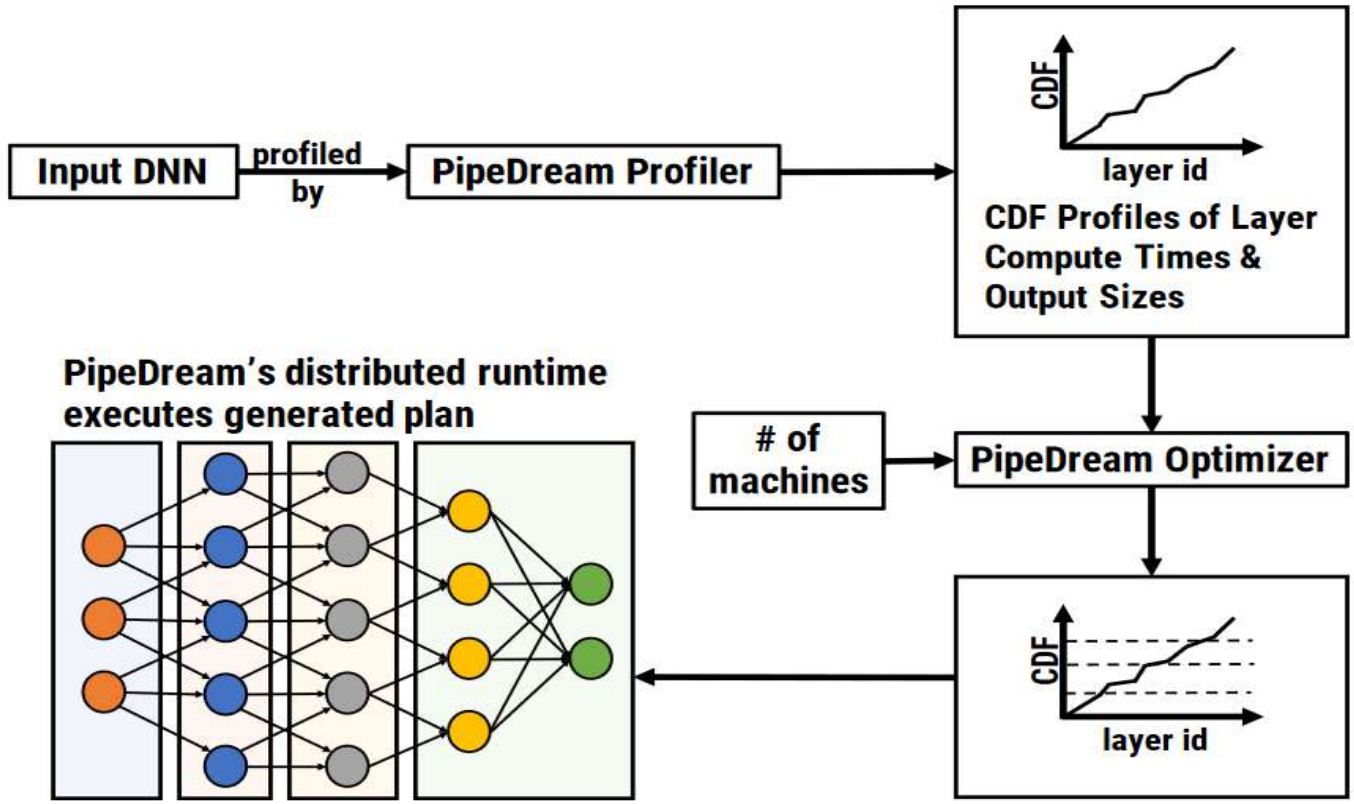
Figure 5: Sizes of layer output data for VGG16 with a minibatch size of 32 on ImageNet1K data. The black dotted line indicates the size of the model parameters.

Pipeline Parallel: Problems

- Автоматическое разделение работы по доступным вычислительным ресурсам
- Планирование вычислений для максимизации производительности при одновременном обеспечении дальнейшего прогресса в выполнении учебной задачи
- Обеспечение эффективности обучения в условиях неопределенности, возникающей в результате конвейеризации



Pipeline Parallel: Problems



Pipeline Parallel: Profiling

Для каждого слоя l на одной машине считают:
T_l - общее время, затраченное на прямой и обратный проход
a_l - количество выходных параметров
w_l - количество параметров в l слое

C_l - время, необходимое для передачи активаций от l слоя до l+1 в конвейере, оценивается a_l
W_l^m- время синхронизации весов (коммуникации) l слоя, оценивается 4 × (m−1) ×$|w_l|$/ m, где m количество узлов

Pipeline Parallel: Partitioning Algorithm

Алгоритм разбиения на фазы принимает выходные данные этапа профилирования и вычисляет:
1) разбиение слоев на фазы
2) коэффициент репликации для каждого этапа
3) оптимальное количество минибатчей для обеспечения занятости обучающего конвейера

Задача: минимизировать общее время обучения модели

Pipeline Parallel: Partitioning Algorithm

A(j,m) - время, затраченное на самую медленную фазу в оптимальном пайплайне между слоями 1 и j с использованием m машин

$T(i \to j, m)$ - время, затраченное на одну фазу, охватывающую слои i-j, с использованием m машин

Цель алгоритма - найти A(N,M) и соответствующее разбиение.

$$T(i \to j, m) = \frac{1}{m} \max\left(\sum_{l=i}^{j} T_l, \sum_{l=i}^{j} W_l^m\right)$$

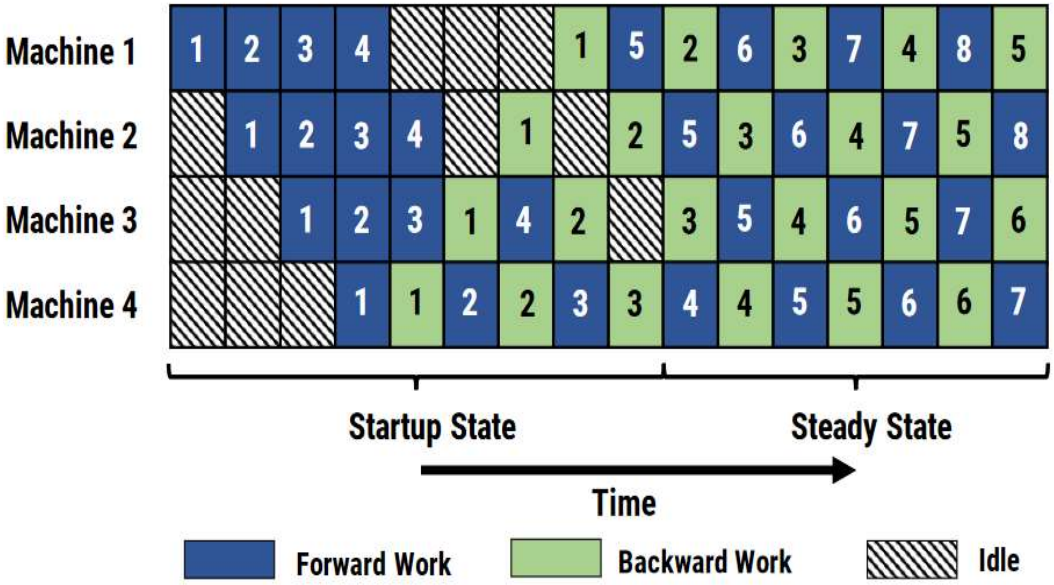
$$A(j, m) = T(1 \to j, m) \qquad \text{или} \qquad A(j, m) = \min_{1 \le i < j} \min_{1 \le m' < m} \max \begin{cases} A(i, m - m') \\ 2 \cdot C_i \\ T(i + 1 \to j, m') \end{cases}$$

Инициализация: A(1,m):=T(1→1,m), A(i,1):=T(1→i,1)

Pipeline Parallel: Scheduling

Каждая машина в системе должна сделать выбор между двумя вариантами:

1. выполнить прямой проход для минибатча, тем самым передавая минибатч нижестоящим машинам
2. выполнить обратный проход для другого минибатча, тем самым обеспечивая прогресс в обучении.



Pipeline Parallel: Weights

Обычно считаем:

$$w^{(t+1)} = w^{(t)} - \nu \cdot \nabla f(w_1^{(t)}, w_2^{(t)}, \ldots, w_n^{(t)})$$

Хранение веса:

$$w^{(t+1)} = w^{(t)} - \nu \cdot \nabla f(w_1^{(t-n+1)}, w_2^{(t-n+2)}, \ldots, w_n^{(t)})$$

Вертикальная синхронизация:

$$w^{(t+1)} = w^{(t)} - \nu \cdot \nabla f(w_1^{(t-n+1)}, w_2^{(t-n+1)}, \ldots, w_n^{(t-n+1)})$$

n – количество машин

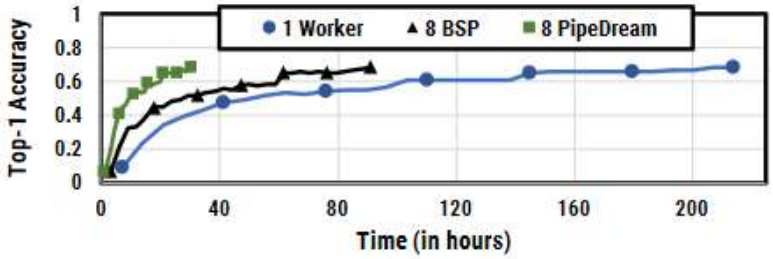
PipeDream: Results

DNN Model	# Machines (Cluster)	BSP speedup over 1 machine	PipeDream Config	PipeDream speedup over 1 machine	PipeDream speedup over BSP	PipeDream communication reduction over BSP
VGG16	4 (A)	1.47×	2-1-1	3.14×	2.13×	90%
	8 (A)	2.35×	7-1	7.04×	2.99×	95%
	16 (A)	3.28×	9-5-1-1	9.86×	3.00×	91%
	8 (B)	1.36×	7-1	6.98×	5.12×	95%
Inception-v3	8 (A)	7.66×	8	7.66×	1.00×	0%
	8 (B)	4.74×	7-1	6.88×	1.45×	47%
S2VT	4 (A)	1.10×	2-1-1	3.34×	3.01×	95%

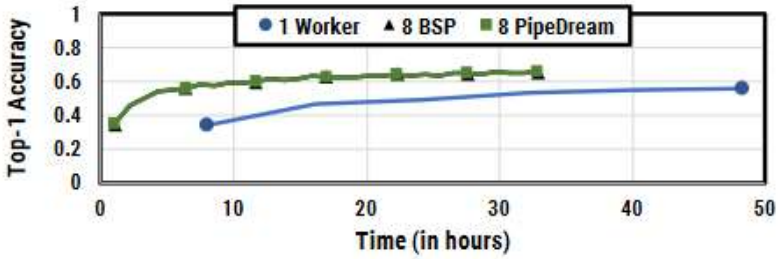
Table 1: Summary of results comparing PipeDream with data-parallel configurations (BSP) when training models to their advertised final accuracy. "PipeDream config" represents the configuration generated by our partinioning algorithm—e.g., "2-1-1" is a configuration in which the model is split into three stages with the first stage replicated across 2 machines.

Cluster-A is a private cluster of NVIDIA Titan X GPUs with 12 GB of GPU device memory.
Cluster-B is public cloud cluster of NVIDIA V100 GPUs, with 16 GB of GPU device memory.

PipeDream: Results



(a) VGG16



(b) Inception-v3

Итоги

- Распределенное обучение может значительно ускорить процесс обучения
- PipeDream работает до 5 раз быстрее, чем современные подходы

Список источников

1. https://arxiv.org/pdf/1806.03377.pdf
2. https://www.sciencedirect.com/science/article/pii/B9780128167182000087
3. https://m.habr.com/ru/company/yandex/blog/525020/