ID2223 - Scalable Machine Learning and Deep Learning

- Review Questions 5 -

PAOLO TETA RALFS ZANGIS

teta | zangis @kth.se

December 11, 2021

1 Question 1

DATA-PARALLELIZED LEARNING $\Longrightarrow$ In order to cope with the large size of the dataset and to increase the performance of the algorithm, one idea is to use more devices and share the workload. By using data-parallelized learning, the idea is to replicate the entire model on different k devices and train them simultaneously selecting a different mini-batch from the training dataset for each sub-model. Thus, during the training process, each device j computes the gradient $J_j(w)$ with respect to a set of B randomly chosen points and then we compute the mean of all the gradients as shown below (synchronization). In neural networks, this means that data parallelism uses the same weights and different mini-batches in each thread.

$$\tilde{g}_B J(w) = \frac{1}{k} \sum_{j=1}^{k} \tilde{g}_B J_j(w)$$

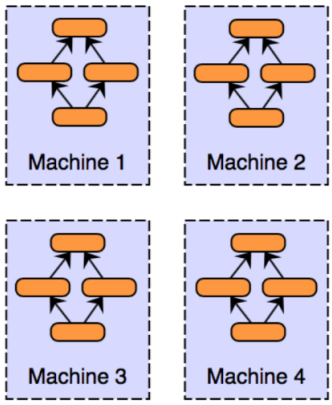

Figure 1: Data-parallelized learning scheme

[source: https://xiandong79.github.io/Intro-Distributed-Deep-Learning]

2 Question 2

MODEL-PARALLELIZED LEARNING $\Longrightarrow$ Again, here we want to address the problem of a huge dataset and long training time. By using model-parallelized learning, the idea is to split the model across multiple devices. For example, each layer in the neural network may be assigned to a different machine. Thus, the weights of the network are split equally among the threads that work on a single mini-batch. The intermediate output needs to be synchronized and stacked providing the input to the following layer.

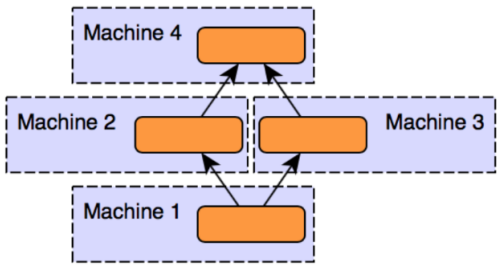

Figure 2: Model-parallelized learning scheme

[source: https://xiandong79.github.io/Intro-Distributed-Deep-Learning]

3 Question 3

Here, we discuss the synchronization approaches in data-parallelized learning.
SYNCHRONOUS $\Longrightarrow$ Each iteration all the workers synchronize with each other (fully-synchronization) and update their parameters. This implies that every worker must wait for all the other workers to finish the transmission of their parameters in the current iteration before the next training iteration. This communication mechanism has limits with respect to the system scalability.
STALE-SYNCHRONOUS $\Longrightarrow$ In this case, the faster workers are allowed to perform more updates than the slower workers without a complete fully-synchronization. This allows to reduce the waiting time of the faster workers during the aggregation step in the staleness bounded barrier. The weight update formula is the following:

$$w_{i,t+1} = w_0 - \eta \cdot \left(\sum_{k=1}^{t} \sum_{j=1}^{n} G_{j,k} + \sum_{k=t-s}^{t} G_{i,k} + \sum_{(j,k) \in S_{i,t+1}} G_{j,k} \right)$$

ASYNCHRONOUS $\Longrightarrow$ No synchronization, which means that each worker transmits its gradients to the parameter server just after the computation. Then, the PS updates the global model without waiting for the other workers. The weight update formula is the following:

$$w_{t+1} = w_t - \eta \cdot \sum_{i=1}^{n} G_{i,t-\tau_{k,i}}$$

where $\tau_{k,i}$ is the time delay between the calculation of the gradient from worker i and the current iteration.
LOCAL SGD $\Longrightarrow$ In this case, all the workers run for multiple iterations and update locally their parameters, then it computes the average of all the local models into the new global model before starting the next iteration. The weight update formula is the following:

$$w_{i,t+1} = \begin{cases} w_{i,t} - \eta \cdot G_{i,t} & \text{if } t+1 \notin I_T \\ w_{i,t} - \eta \cdot \frac{1}{n} \cdot \sum_{i=1}^{n} G_{i,t} & \text{if } t+1 \in I_T \end{cases}$$

where I_T stands for the synchronization timestamps.

4 Question 4

GRADIENT QUANTIZATION $\Longrightarrow$ It means that we reduce the number of bits to represent the data. In deep learning, quantization is the process of approximating a neural network which is using floating-point numbers to a network of low bit width numbers. This implies a reduction in the memory usage and the computation cost, but it affects the accuracy.
GRADIENT SPARSIFICATION $\Longrightarrow$ It means that we reduce the number of elements that are transmitted at each iteration when executing an algorithm. When training a neural network, it's important to update weights, thus with gradient sparsification only significant gradients are needed to update the model parameters ensuring the convergence of the training task.

5 Question 5

SHARE-REDUCE PHASE $\Longrightarrow$ The basic idea is that the process sends data to the following process making a circle. This can be described using the following formula:

$$(p+1)\%n$$

where p is the current process and n is total number of participants. Following, this data is separated into number of chunks equal to the process count. The exchange starts by each process sending the chunk that corresponds to the process index. Subsequently, reduce operation (such as sum, max, etc.) is performed on

the incoming data with respect to the processes own data at the corresponding chunk, this information is then forwarded further and the interaction is repeated until a process holds complete reduction of the chunk. This is described in Figures 3-7.

SHARE-ONLY PHASE $\Longrightarrow$ The completed reductions are exchanged in the same ring structure, with the end goal being all processes have all chunks completed. This is described in Figures 8-11.

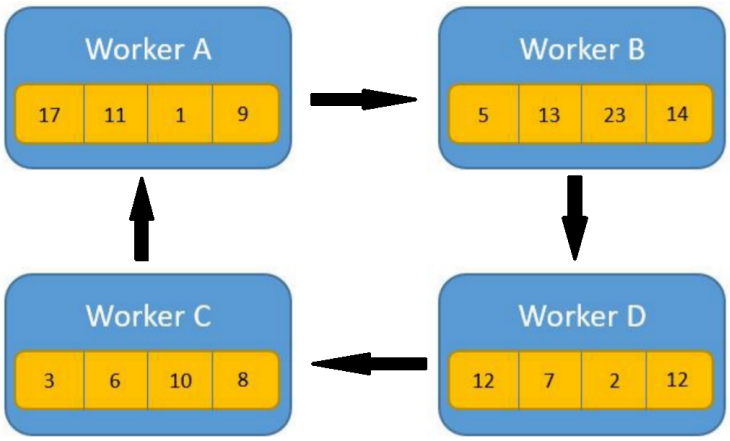

Figure 3: Step 0

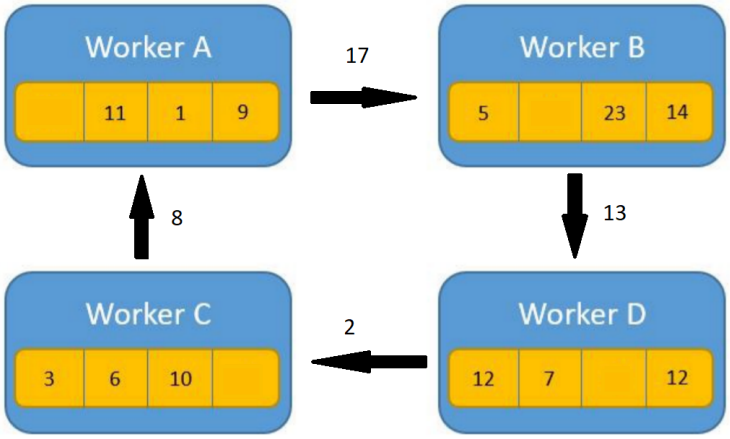

Figure 4: Step 1

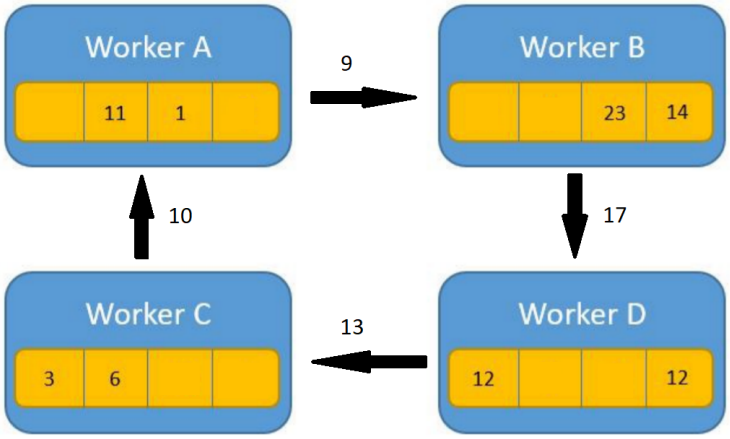

Figure 5: Step 2

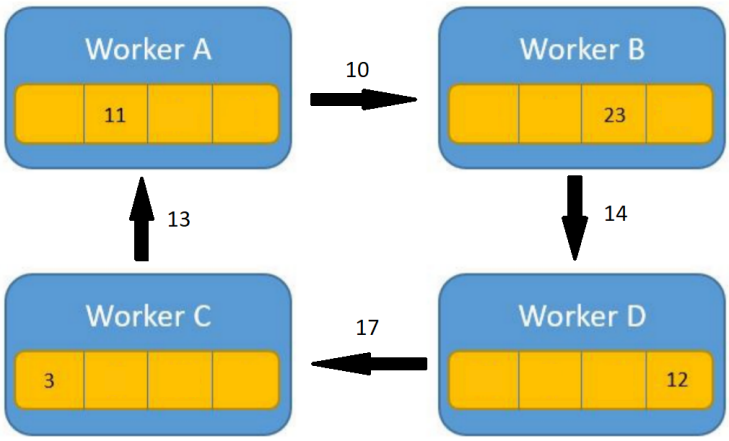

Figure 6: Step 3

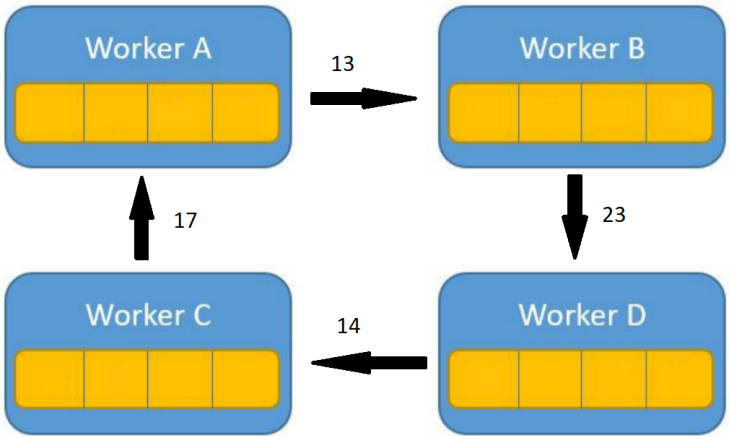

Figure 7: Step 4

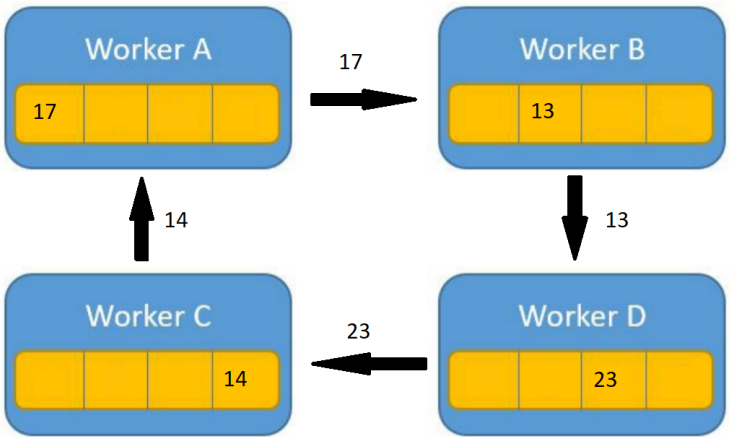

Figure 8: Step 5

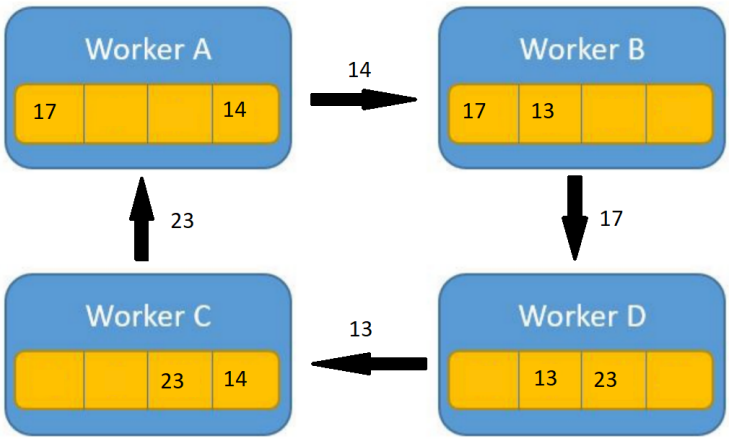

Figure 9: Step 6

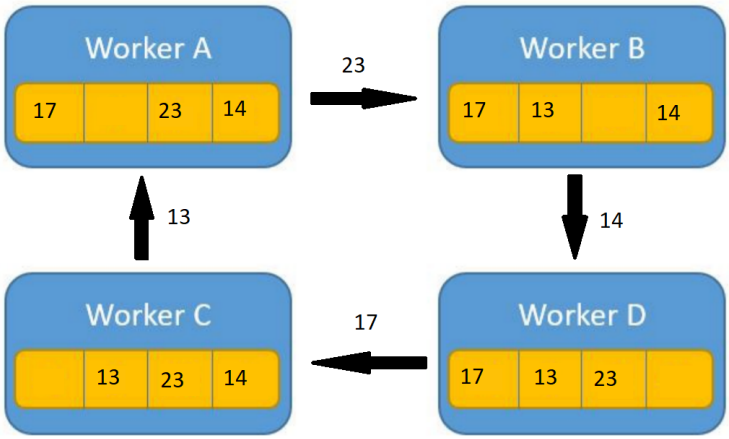

Figure 10: Step 7

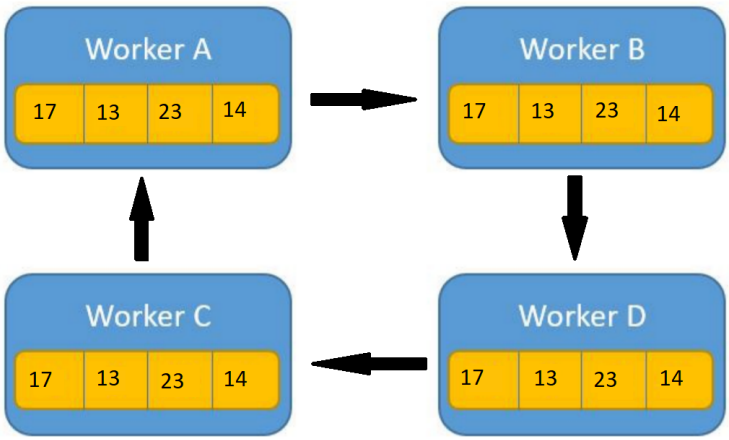

Figure 11: Step 8