ID2223
Scalable Machine Learning and Deep Learning

Review Questions 5

FALL 2021

Yuchen Gao
Weikai Zhou

December 11, 2021

KTH EECS Embedded Systems

1 Explain how does the data-parallelized learning work?

The model is small that can be used on each machine. But the data is large that we should divide it into different parts. Therefore, we should replicate a whole model on every device and train all replicas simultaneously, using a different mini-batch for each. Assume we have k devices. For each device, the loss function is

$$J_j(w) = \sum_{i=1}^{b_j} l(y_i,\ \hat{y}_i),\ \forall j = 1,\ 2,\ ...,\ k$$

Then we need to compute $\widetilde{g}_B J_j(w)$, which is the gradient of $J_j(w)$ with respect to a set of B randomly chosen points at device j. After that, we compute the mean of the gradients

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

Finally, we can update the model with

$$w = w - \eta \widetilde{g}_B J(w)$$

The figure below (Figure 1) illustrates the process of the data-parallelized learning with $k = 4$.

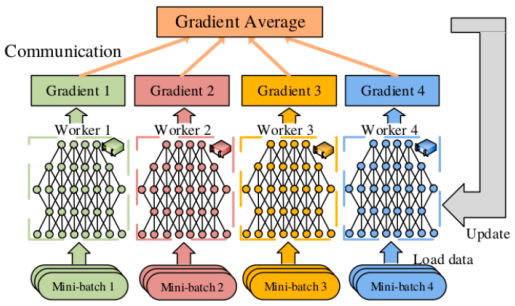


Figure 1: Illustration of data-parallelized learning

2 Explain how does the model-parallelized learning work?

The model is large that we can not load it on single device. Then we split the model across multiple devices. The figure below (Figure 2) gives us two examples of splitting the model across 4 devices.

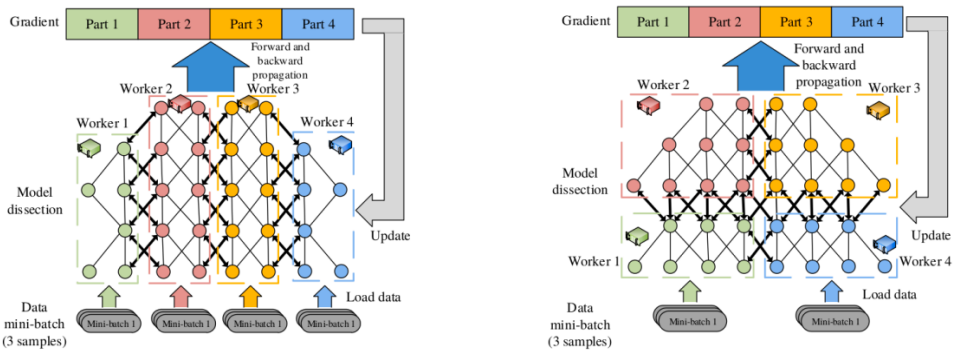


Figure 2: Example of model-parallelized learning

The partition of the model depends on the architecture of the NN. Here we list three partition strategies.

- Hash partitioning: Randomly assign vertices to devices proportionally to the capacity of the devices by using a hash function.
- Critical path: Assigning the complete critical path to the fastest device. A critical path is the path with the longest computation time from source to sink vertex.
- Multi-objective heuristics: Assign vertices to devices based on the objective, e.g., memory, importance, traffic, and execution time. But it is a complex optimization problem to solve.

3 Explain different synchronization approaches in data-parallelized learning?

There are four models introduced in the class.

- Synchronous

  After each iteration, the workers synchronize their parameter updates. Every worker must wait for all workers to finish the transmission of all parameters in the current iteration, before the next training. Stragglers can influence the overall system throughput. High communication cost that limits the system scalability. As shown below (Figure 3), the Gradient/Model Aggregation waits until all the workers finish Backward Propagation.

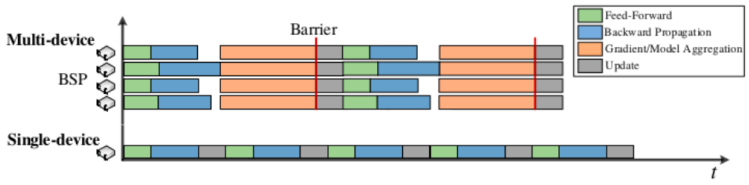


Figure 3: Synchronous model

- Stale-synchronous

  It alleviates the straggler problem without losing synchronization. The faster workers to do more updates than the slower workers to reduce the waiting time of the faster workers. Staleness bounded barrier (Figure 4) is used to limit the iteration gap between the fastest worker and the slowest worker. For a maximum staleness bound s, the update formula of worker i at iteration $t+1$:

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

  The update has three parts:

  1. Guaranteed pre-window updates from clock 1 to t over all workers.
  2. Guaranteed read-my-writes in-window updates made by the querying worker i.

3. Best-effort in-window updates. $S_{i,t+1}$ is some subset of the updates from other workers during period $[t-s]$.

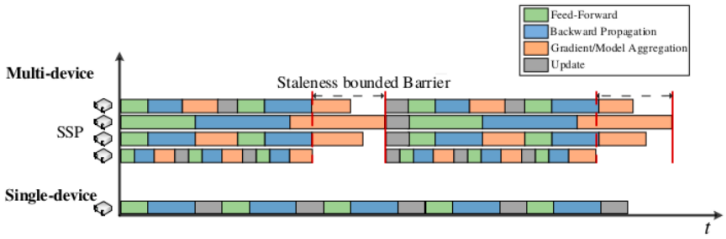


Figure 4: Stale-synchrounous model

- Asynchronous

  It completely eliminates the synchronization (Figure 5). Each work transmits its gradients to the PS after it calculates the gradients. The PS updates the global model without waiting for the other workers.

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

  $\tau_{k,i}$ is the time delay between the moment when worker i calculates the gradient at the current iteration.

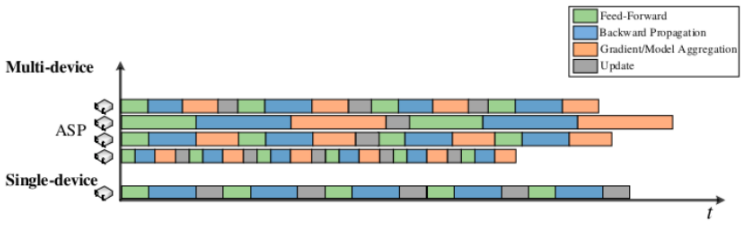


Figure 5: Asynchronous model

- Local SGD

  All workers run several iterations, and then averages all local models into the newest global model (Figure 6). If $\mathcal{I}_T$ represents the synchronization timestamps, then:

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

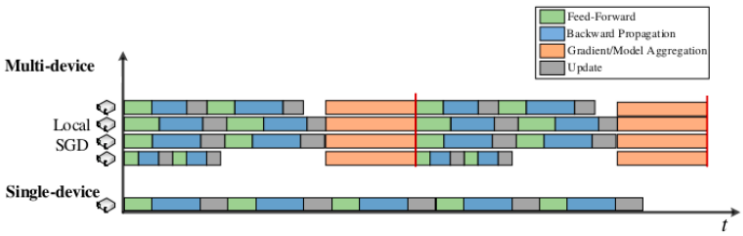


Figure 6: Local SGD model

4 Briefly explain gradient quantization and gradient sparsification.

Gradient Quantization: It is a technique used for useing lower bits to represent the data. For example (Figure 7), if the original gradient for one element has 32 bits, now we will only use 4 bits to represent.

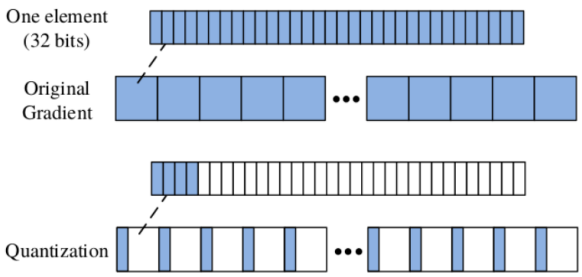


Figure 7: Illustration of gradient quantization

Gradient Sparsification: It reduces the number of elements that are transmitted at each iteration. Only significant gradients are required to update the model parameter to guarantee the convergence of the training, e.g., the zero-valued elements are no need to transmit. For example (Figure 8), we do not transmit the whole gradient but choose some of them to transmit.

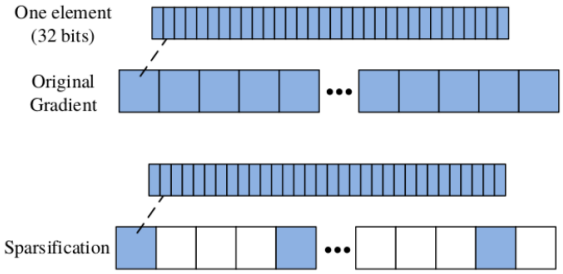


Figure 8: Illustration of gradient sparsification

5 Use the following picture (Figure 9) and show step-by-step how the ring-allreduce works to compute the maximum of all elements?

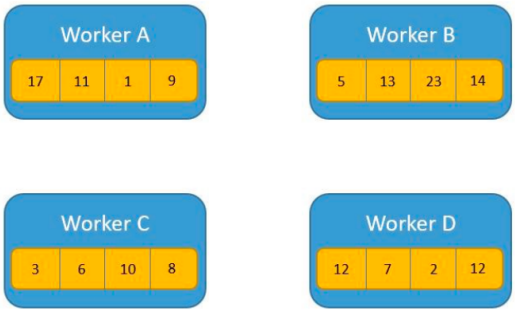


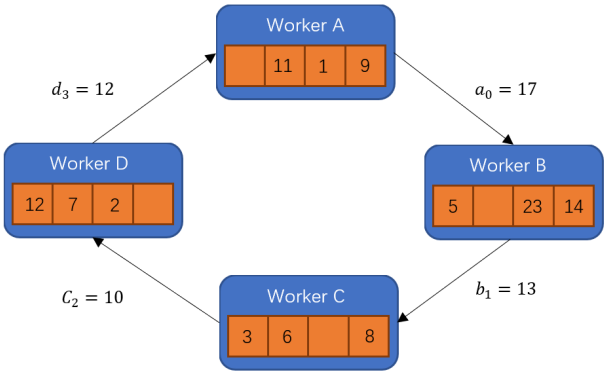
Figure 9: Figure for question 5

The Ring-Allreduce has two phases:

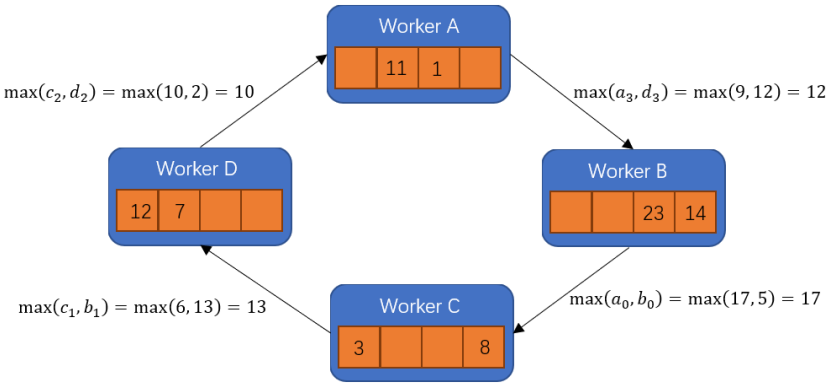
1. First, the share-reduce phase
2. Then, the share-only phase

In the share-reduce phase, each process p sends data to the process $(p+1)\%m$.

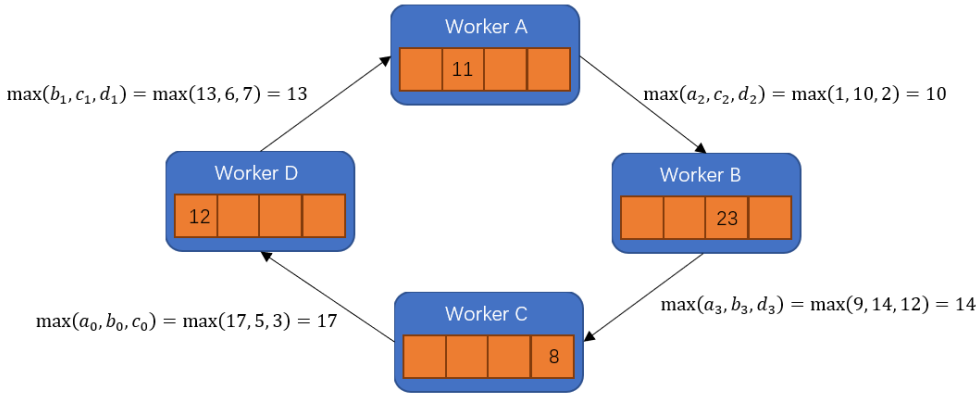
In the first share-reduce step, A sends 17 to B, B sends 13 to C, C sends 10 to D and D sends 12 to A.



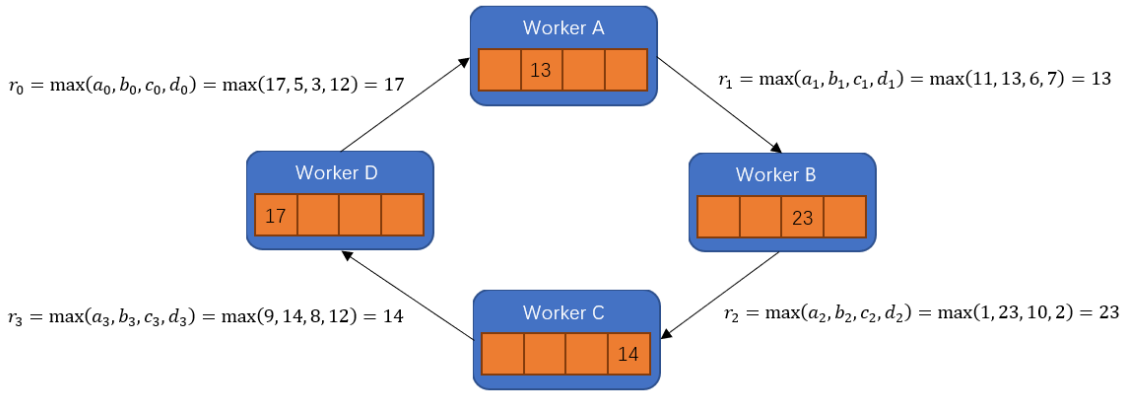
In the second share-reduce step, they will apply the reduce operator (e.g., max). Then, they will send it to the next one. B sends 17 to C, C sends 13 to D, D sends 10 to A and A sends 12 to B.



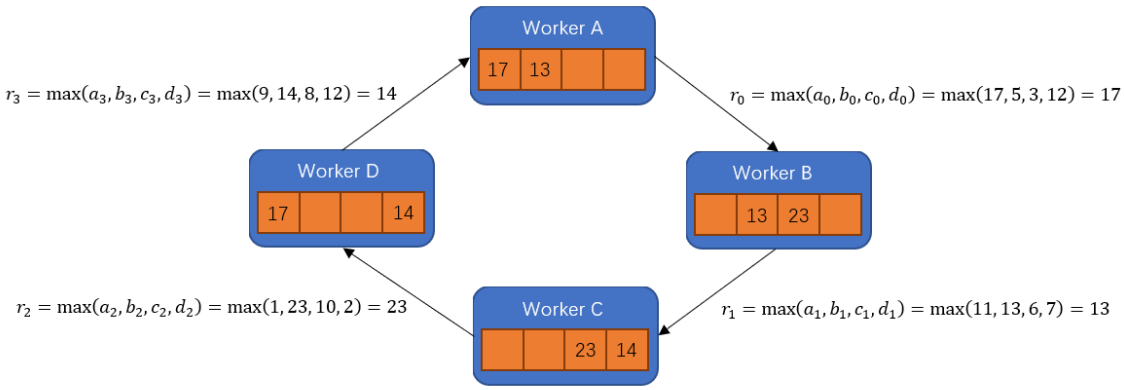
In the third share-reduce step, they will apply the reduce operator (e.g., max). Then, they will send it to the next one. C sends 17 to D, D sends 13 to A, A sends 10 to B and B sends 14 to C.

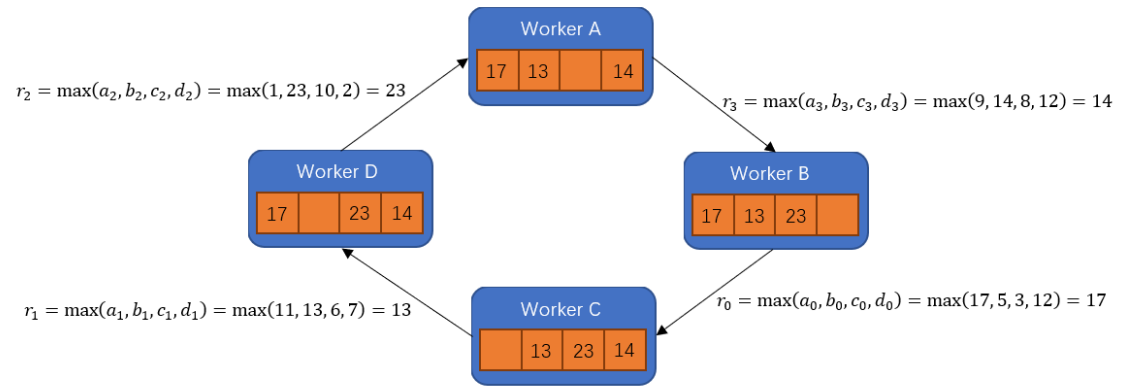


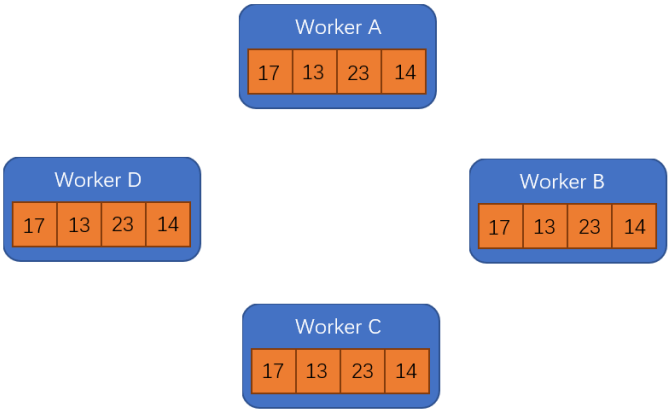
In the fourth (final) share-reduce step, they will apply the reduce operator (e.g., max). Then, they will send it to the next one. D sends 17 to A, A sends 13 to B, B sends 23 to C and C sends 14 to D. And at this point, each worker holds a part of the end result.



The share-only steps are similar process of sharing the data in a ring-like fashion without applying the reduce operation.






Worker A
17
13
23
14
Worker D
17
13
23
14
Worker B
17
13
23
14
Worker C
17
13
23
14