Multi-GPU training of deep learning models on Piz Daint

Advanced Data Parallelism with DeepSpeed

Rafael Sarmiento
ETHZürich / CSCS
CSCS/USI Summer University 2022

ETH zürich
CSCS
Centro Svizzero di Calcolo Scientifico
Swiss National Supercomputing Centre

Limitations of (vanilla) data parallelism

- Data parallelism does not reduce memory per device
- Training of models with more than 1.4 billion parameters runs out of memory with current generation of GPUs
- Models with billions of parameters which offer significant accuracy gains are no longer uncommon
- Alternatives to data parallelism can be model parallelism, pipeline parallelism and CPU offloading however they might not give the best performance

Outline

- Introducing DeepSpeed
- DeepSpeed's Zero Redundancy Optimizer (ZeRO)
- [lab] Understanding the effect of the ZeRO-{1, 2, 3} on the memory.

DeepSpeed

- DeepSpeed is an open source deep learning optimization library for PyTorch developed my Microsoft
- Designed to reduce computing power and memory use
- Enables the training of large models with better parallelism on existing computer hardware
- Mixed precision training, single-GPU, multi-GPU and multi-node
- Zero Redundancy Optimizer (ZeRO) for training models with 100 billion or more parameters

https://github.com/microsoft/DeepSpeed
https://www.deepspeed.ai

DeepSpeed's Zero Redundancy Optimizer (ZeRO)

- ZeRO partitions the various model training states (weights, gradients, and optimizer states) across devices
- It's implemented in incremental stages of optimizations, each including the previous one:
    - Stage 1: **Partitioning of the optimizer states** (e.g., for Adam optimizer, 32-bit weights, and the first, and second moment estimates) across the processes
    - Stage 2: **Partitioning of the gradients** for updating the model weights. Processes retain only the gradients corresponding to their portion of the optimizer states
    - Stage 3: **Partitioning of the model parameters** partitioned across the processes

S. Rajbhandari et al. ZeRO: Memory Optimizations Toward Training Trillion Parameter Models

Adam optimizer

- The model parameters are updated by an expression that contains the **first momentum** and **second momentum**:

$$m_t = \beta_1 m_{t-1} + (1-\beta_1)\partial w_t \ , \qquad s_t = \beta_2 s_{t-1} + (1-\beta_2)\partial w_t^2$$

- The two following quantities are computed:

$$\tilde{m}_t = \frac{m_t}{\sqrt{1-\beta_1^{t+1}}} \ , \qquad \tilde{s}_t = \frac{s_t}{\sqrt{1-\beta_2^{t+1}}}$$

- and the parameters are updated with

$$w_t = w_{t-1} - \alpha \frac{\tilde{m}_t}{\sqrt{\tilde{s}_t} + \epsilon}$$

ZeRO [Adam optimizer mixed precision (MP)]

Element	dtype	size (P bytes)
∂w	`fp16`	2
w	`fp16`	2
w_{fp32}	`fp32`	4
m	`fp32`	4
s	`fp32`	4

- ∂w are the gradients in `fp16` • w_{fp32} are the master copy of the weights in `fp32` • m and s are the first and second momentum respectively
- The remaining memory is consumed by activations, temporary buffers and unusable fragmented memory

- `ZeRo-0`: $[w] + [\partial w] + [w_{\text{fp32}}] + [m] + [s]$
- `ZeRo-1`: $[w] + [\partial w] + \frac{[w_{\text{fp32}}]+[m]+[s]}{N}$
- `ZeRo-2`: $[w] + \frac{[\partial w]+[w_{\text{fp32}}]+[m]+[s]}{N}$
- `ZeRo-3`: $\frac{[w]+[\partial w]+[w_{\text{fp32}}]+[m]+[s]}{N}$

- $[x]$ stands for 'size of x'

ZeRO [Adam optimizer MP] Ex. 7.5 billion weights in 64 GPUs

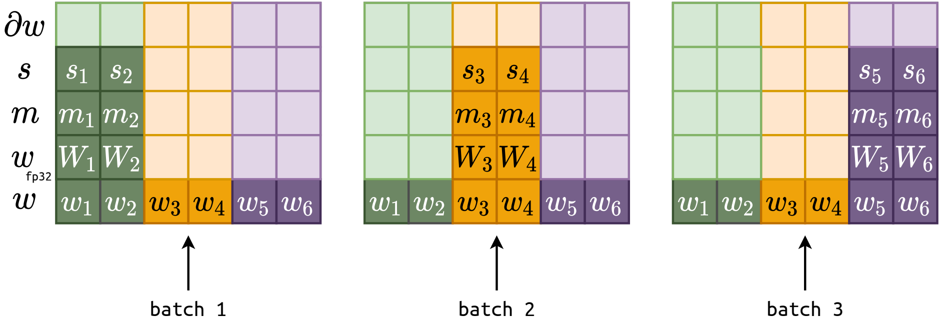
Element	dtype	size (P bytes)
∂w	`fp16`	2
w	`fp16`	2
w_{fp32}	`fp32`	4
m	`fp32`	4
s	`fp32`	4

- ∂w are the gradients in `fp16` • w_{fp32} are the master copy of the weights in `fp32` • m and s are the first and second momentum respectively
- The remaining memory is consumed by activations, temporary buffers and unusable fragmented memory

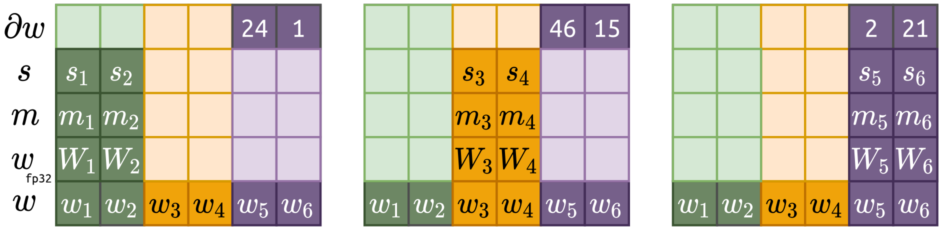
- `ZeRo-0`: 16 * 7.5 billion = 120 GB/GPU
- `ZeRo-1`: 4.2 * 7.5 billion = 31.5 GB/GPU
- `ZeRo-2`: 2.2 * 7.5 billion = 16.6 GB/GPU
- `ZeRo-3`: 0.3 * 7.5 billion = 1.9 GB/GPU

- $[x]$ stands for 'size of x'

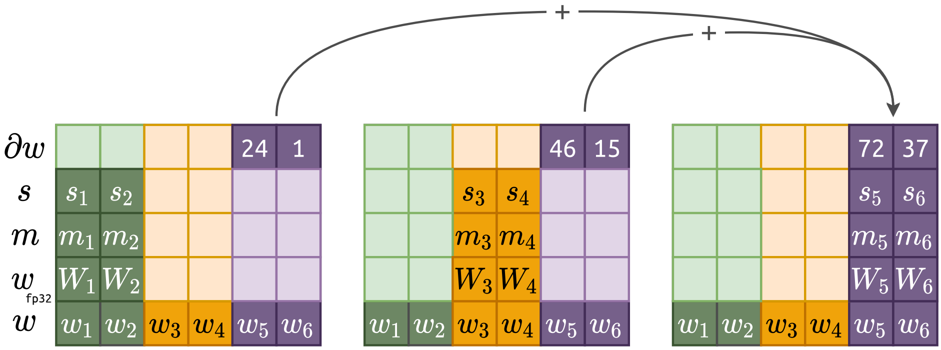
ZeRO {1, 2} [Adam optimizer] Communication



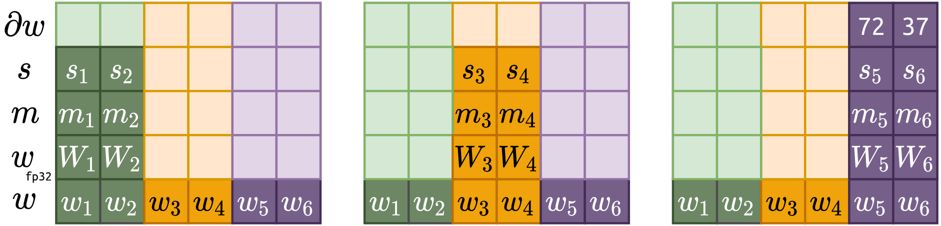
ZeRO {1, 2} [Adam optimizer] Communication



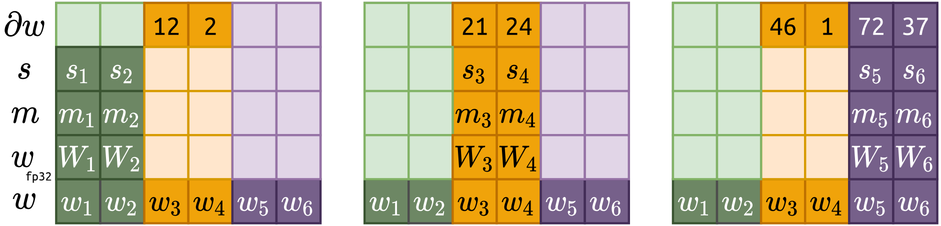
ZeRO {1, 2} [Adam optimizer] Communication



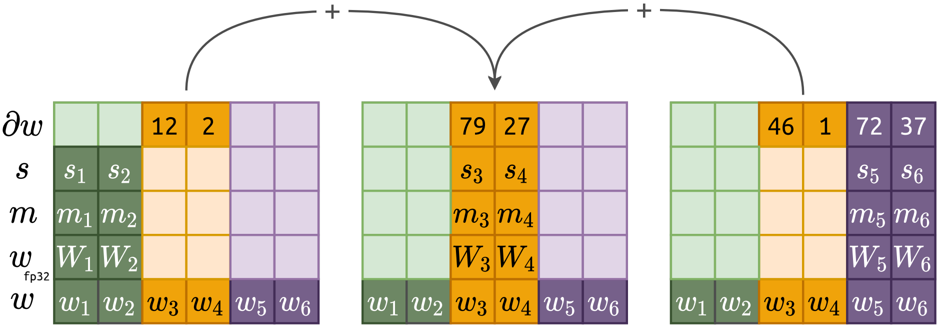
ZeRO {1, 2} [Adam optimizer] Communication



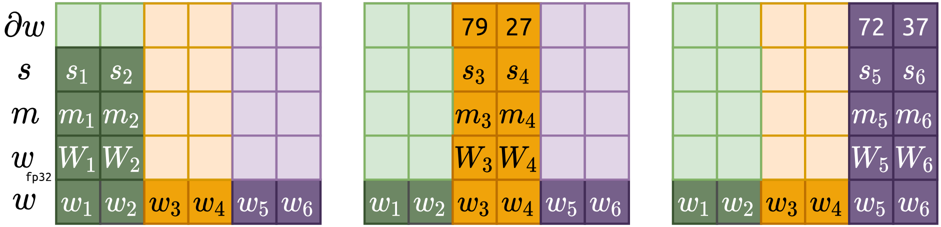
ZeRO {1, 2} [Adam optimizer] Communication



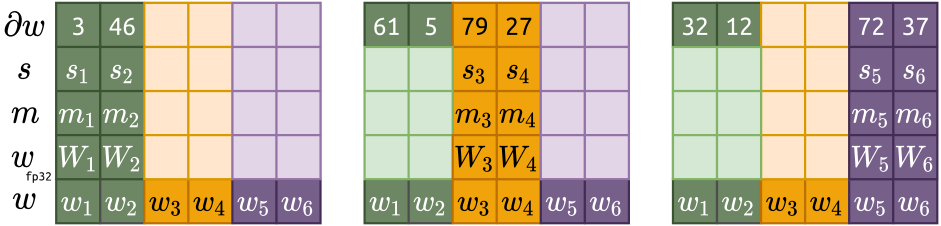
ZeRO {1, 2} [Adam optimizer] Communication



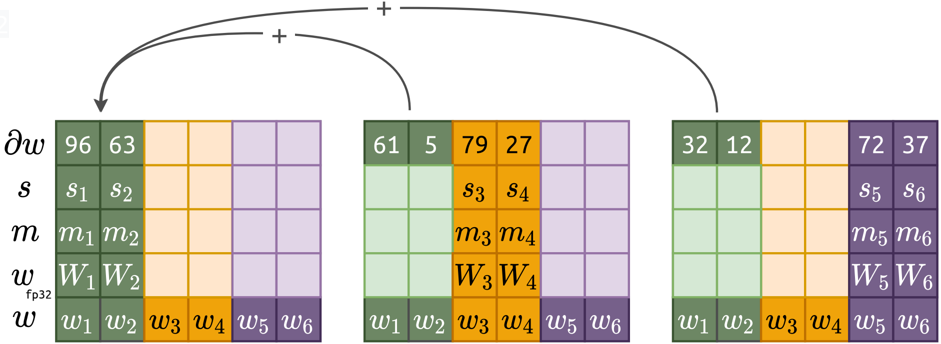
ZeRO {1, 2} [Adam optimizer] Communication



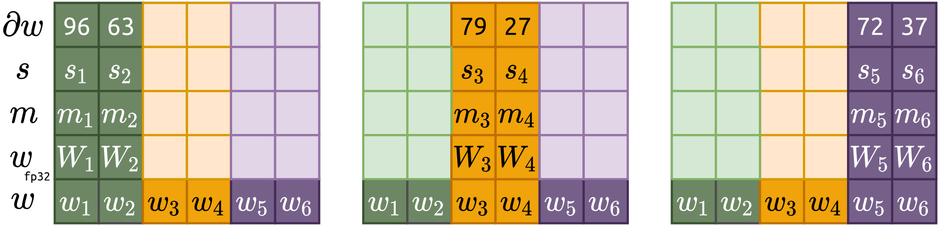
ZeRO {1, 2} [Adam optimizer] Communication



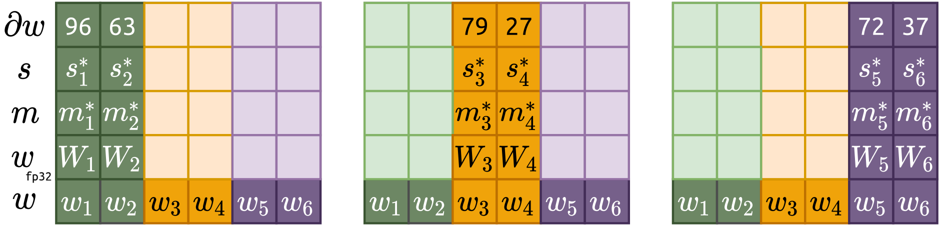
ZeRO {1, 2} [Adam optimizer] Communication



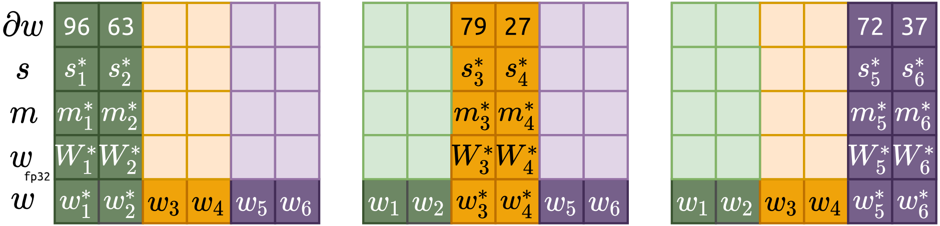
ZeRO {1, 2} [Adam optimizer] Communication



ZeRO {1, 2} [Adam optimizer] Communication

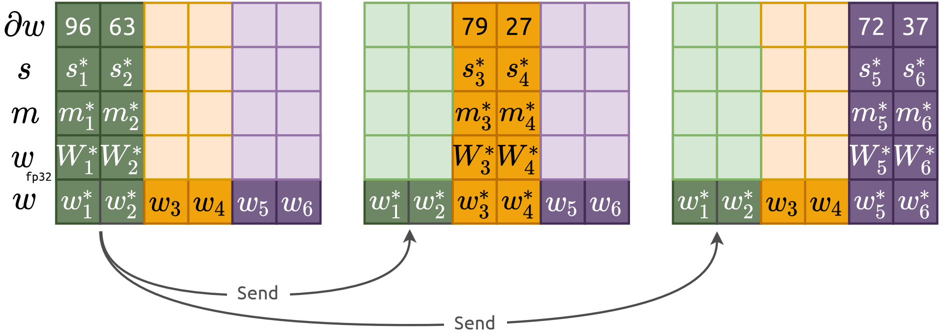


ZeRO {1, 2} [Adam optimizer] Communication

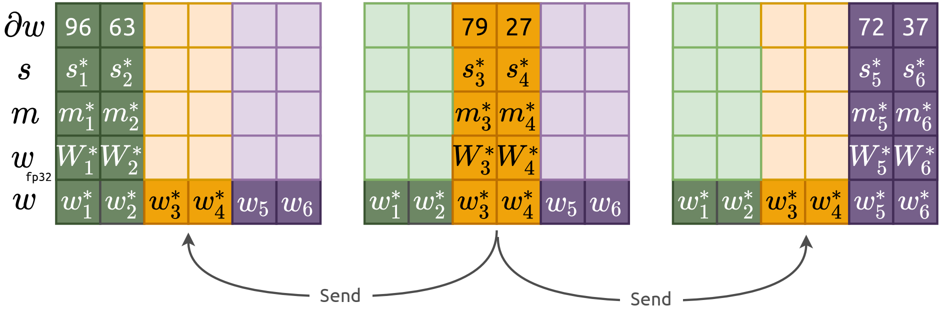


ZeRO {1, 2} [Adam optimizer] Communication

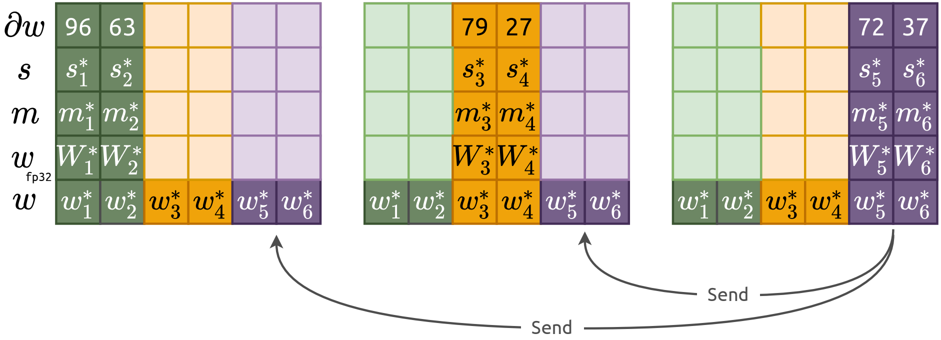
12



ZeRO {1, 2} [Adam optimizer] Communication



ZeRO {1, 2} [Adam optimizer] Communication



ZeRO {1, 2} [Adam optimizer] Communication

- Each of the N workers communicates with the rest of the workers $2(N-1)$ times per iteration
- The values of the reductions of the gradients are obtained with the first $N-1$ communications
- The second $N-1$ communications are performed to update the weights on all workers
- The total amount of data sent by each worker $2(N-1)\frac{\Psi}{N} \approx 2\Psi$ is the same that in regular data parallelism

Ψ is the array size

S. Rajbhandari et al. ZeRO: Memory Optimizations Toward Training Trillion Parameter Models

ZeRO 3 [Adam optimizer]

Let's take 10 minutes and watch the video ZeRo 4-way data parallel training in the post **ZeRO & DeepSpeed: New system optimizations enable training models with over 100 billion parameters** from the DeepSpeed Team.

The video shows how ZeRO-3 performs a training step.

ZeRO 3 [Adam optimizer] Communication

- Each of the N workers communicates with the rest of the workers $N-1$ times for the reduction of the gradients
- The update of the weights is not communicated after each optimizer step. Instead, each weight is sent $N-1$ times during the forward pass and $N-1$ times during the backpropagation
- The total amount of data sent by each worker $3(N-1)\frac{\Psi}{N} \approx 3\Psi$ is 1.5 times that of the regular data parallelism (2Ψ)

Ψ is the array size

S. Rajbhandari et al. ZeRO: Memory Optimizations Toward Training Trillion Parameter Models

More on Deepspeed and ZeRO

- Easy turning on/off mixed precision training
- 1-bit Adam
- ZeRO-Offload: a ZeRO optimization that offloads the optimizer memory and computation from the GPU to the host CPU enabling the training of models up to 13 billion parameters on a single GPU
- ZeRO performs on-the-fly memory defragmentation by moving activation checkpoints and gradients to pre-allocated contiguous memory buffers
- ZeRO is being implemented on natively on PyTorch (see `ZeroRedundancyOptimizer`

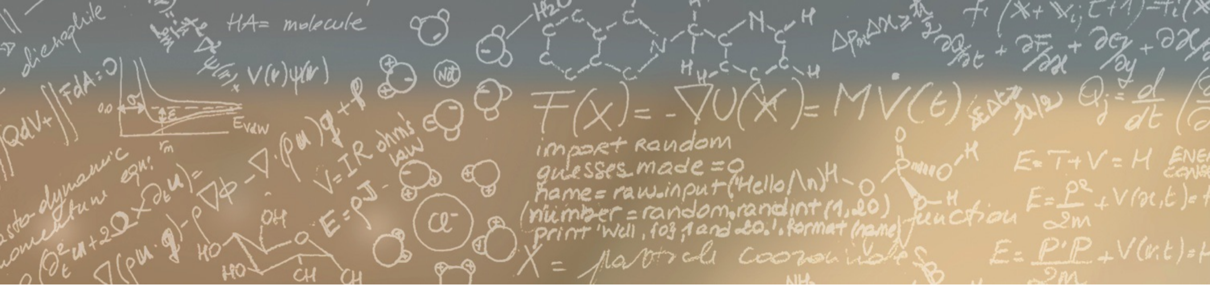
[lab] Understanding the effect of ZeRo-{1, 2, 3} on memory

Let's go to the terminal and run the script `zero/pt_deepspeed_check_mem.py`:

```
srun python --pty pt_deepspeed_check_mem.py \
                  --deepspeed_config ds_config.json \
                  --data-dim 10000
```

and fill in the following table

N. Nodes	Batch size	N. params	Mem init	Mem train	ZeRo stage
-	-	-	-	-	-



Thank you for your attention!

ETH zürich

CSCS
Centro Svizzero di Calcolo Scientifico
Swiss National Supercomputing Centre