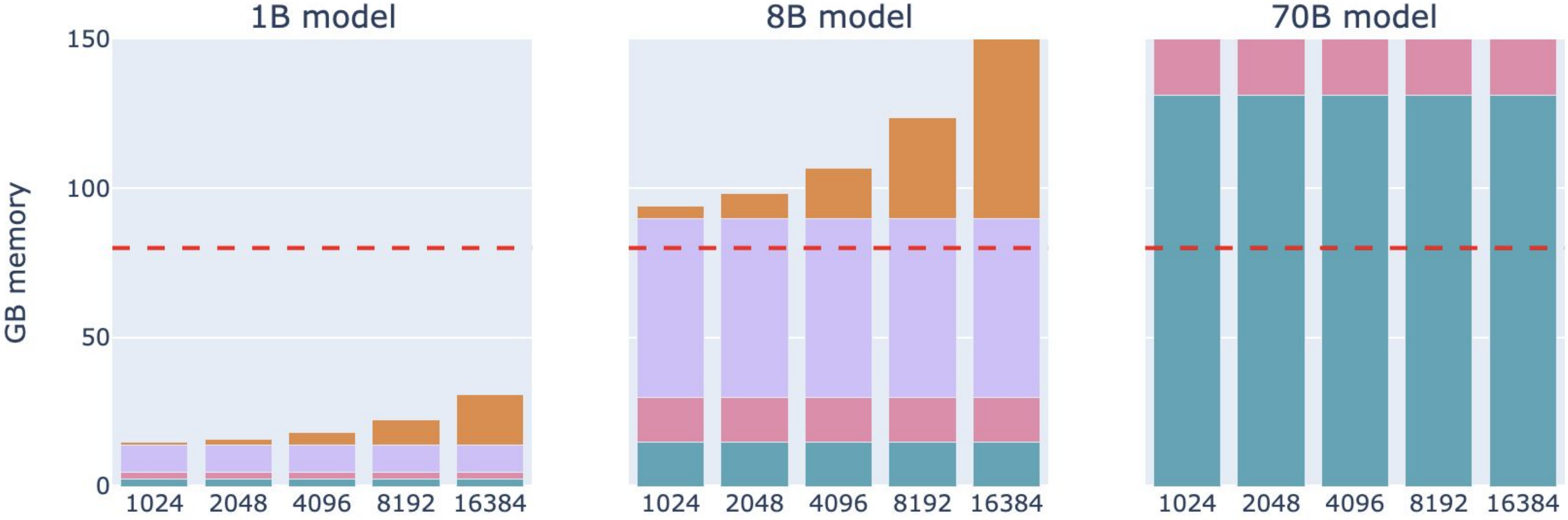
Data Parallelism [ZERO:]

made with ❤ for “Little ML book club”

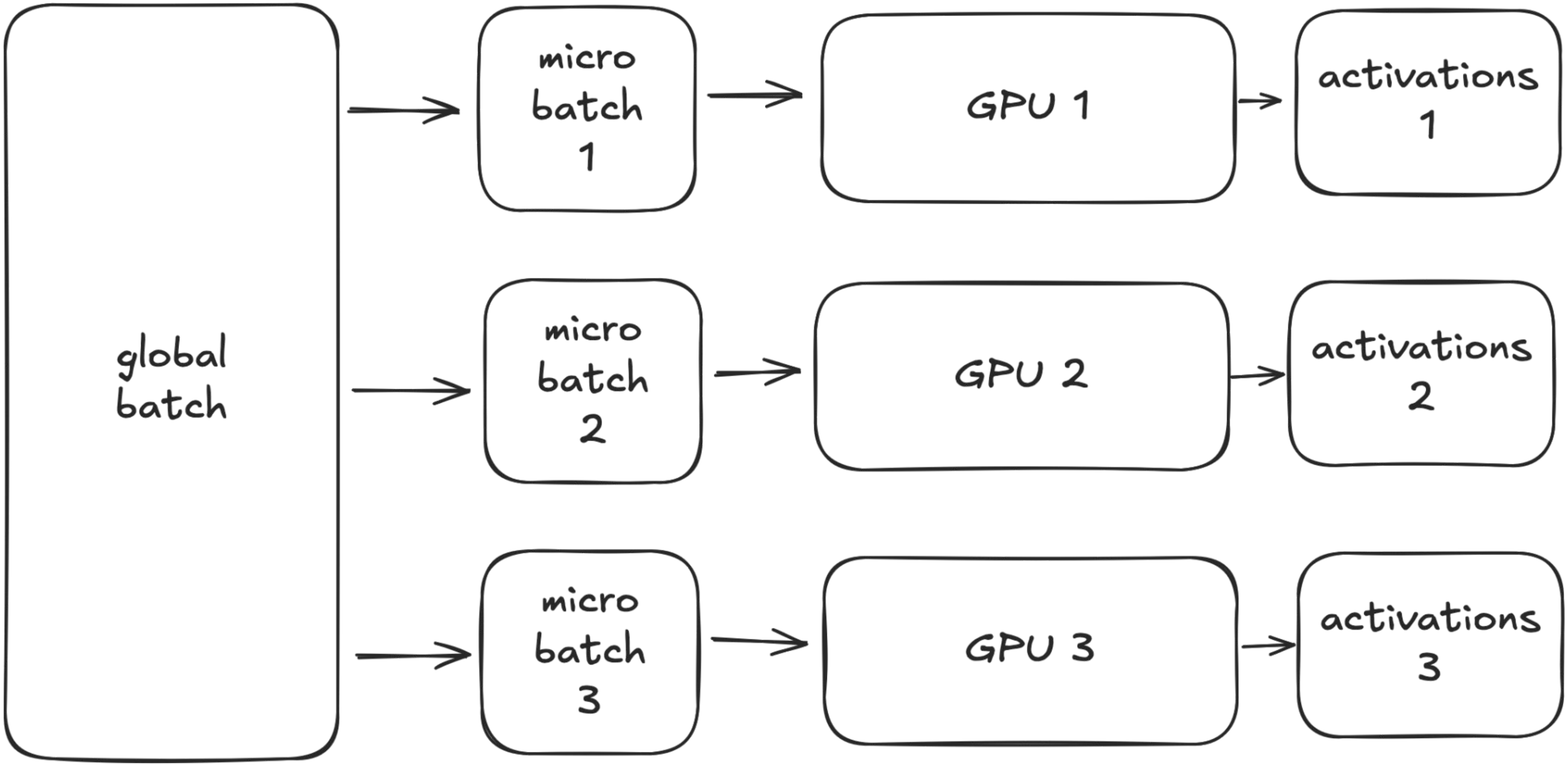

1B model
8B model
70B model
GB memory
150
100
50
0
1024
2048
4096
8192
16384
1024
2048
4096
8192
16384
1024
2048
4096
8192
16384

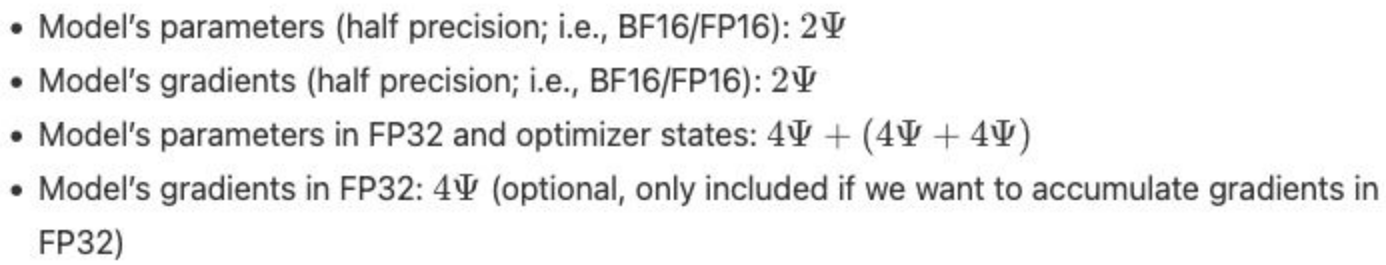
Types of ZeRO

1. Optimizer
2. Optimizer + Gradients
3. Optimizer + Gradients + Weights

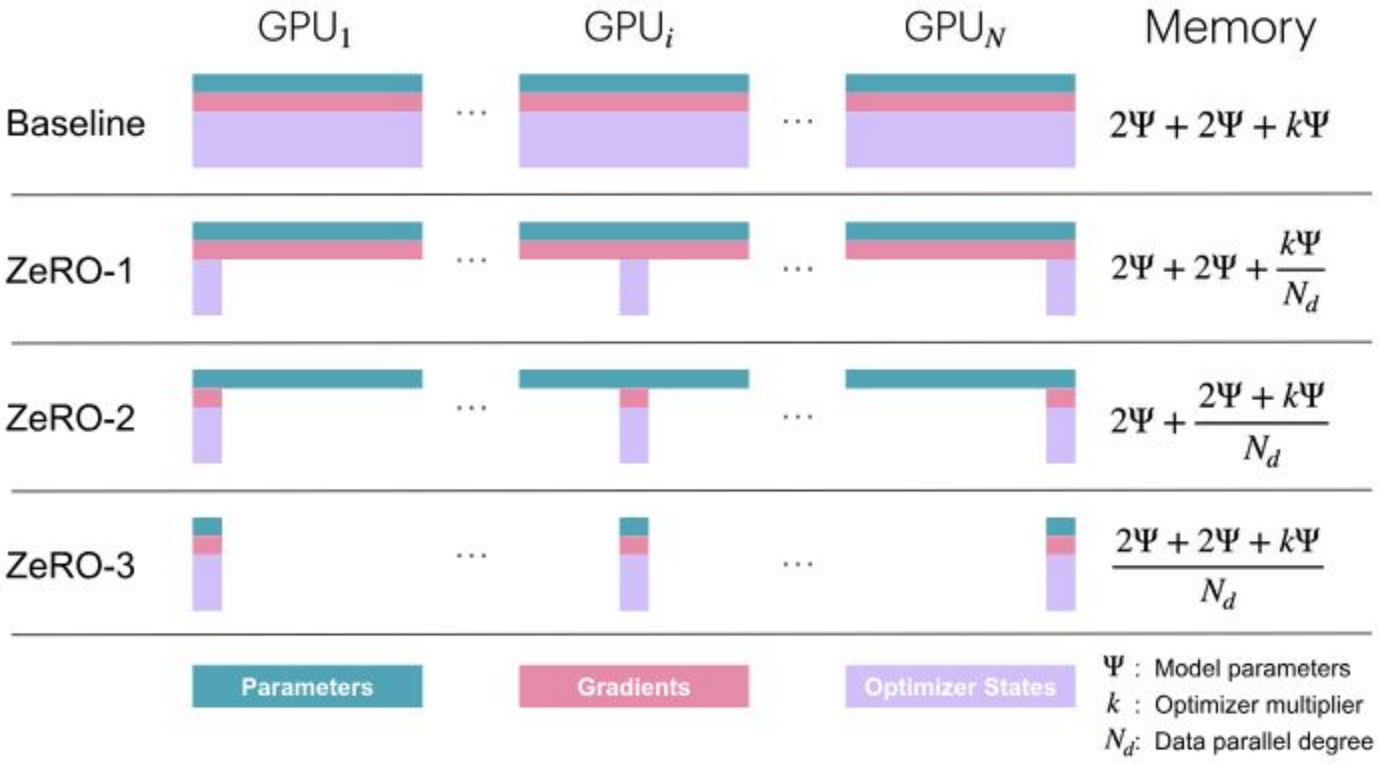
Types of ZeRO

1. Optimizer
2. Optimizer + Gradients
3. Optimizer + Gradients + Weights


global batch
micro batch 1
micro batch 2
micro batch 3
GPU 1
GPU 2
GPU 3
activations 1
activations 2
activations 3



- Model’s parameters (half precision; i.e., BF16/FP16): 2Ψ
- Model’s gradients (half precision; i.e., BF16/FP16): 2Ψ
- Model’s parameters in FP32 and optimizer states: $4\Psi + (4\Psi + 4\Psi)$
- Model’s gradients in FP32: 4Ψ (optional, only included if we want to accumulate gradients in FP32)



GPU$_1$ GPU$_i$ GPU$_N$ Memory

Baseline $2\Psi + 2\Psi + k\Psi$

ZeRO-1 $2\Psi + 2\Psi + \frac{k\Psi}{N_d}$

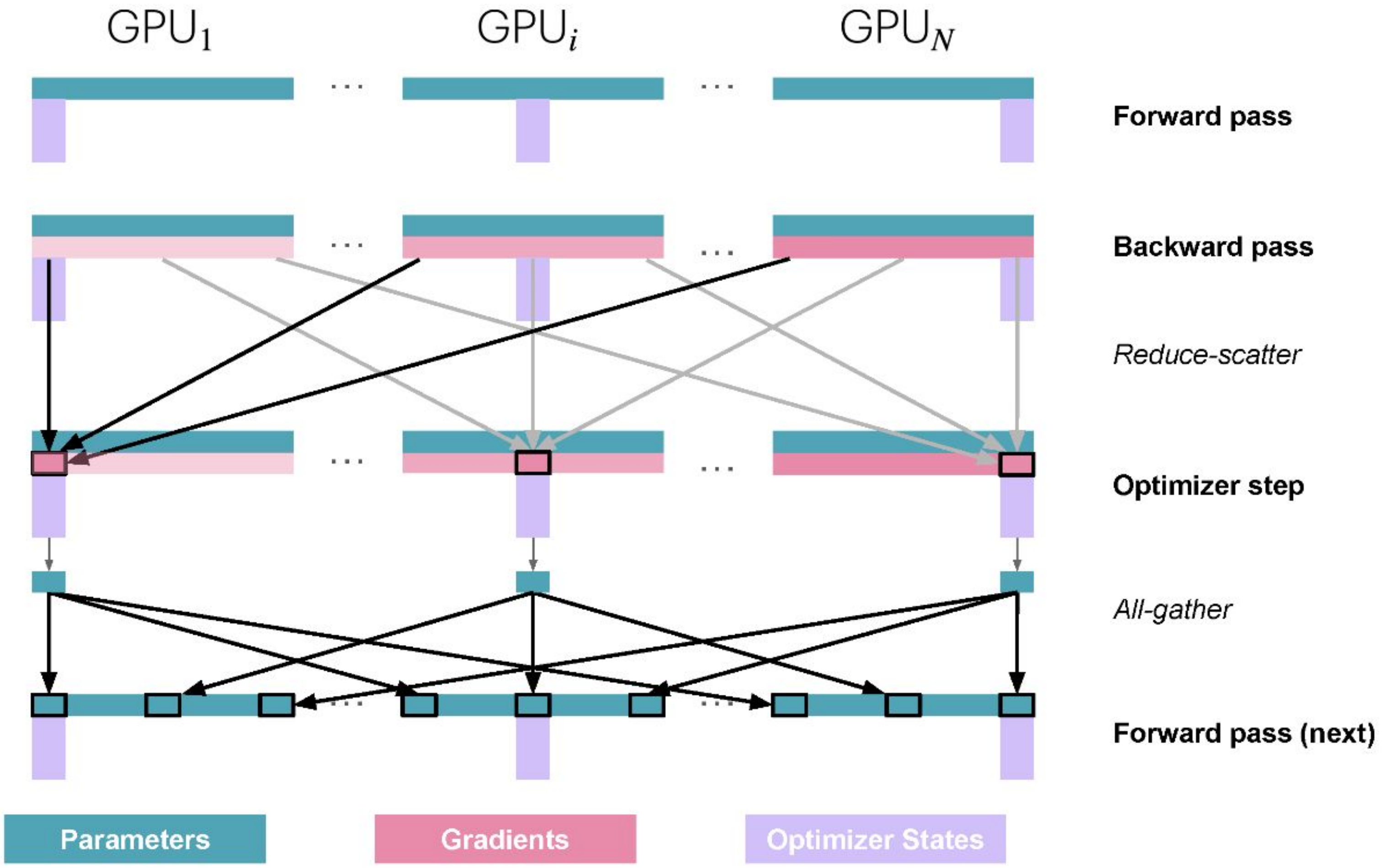
ZeRO-2 $2\Psi + \frac{2\Psi + k\Psi}{N_d}$

ZeRO-3 $\frac{2\Psi + 2\Psi + k\Psi}{N_d}$

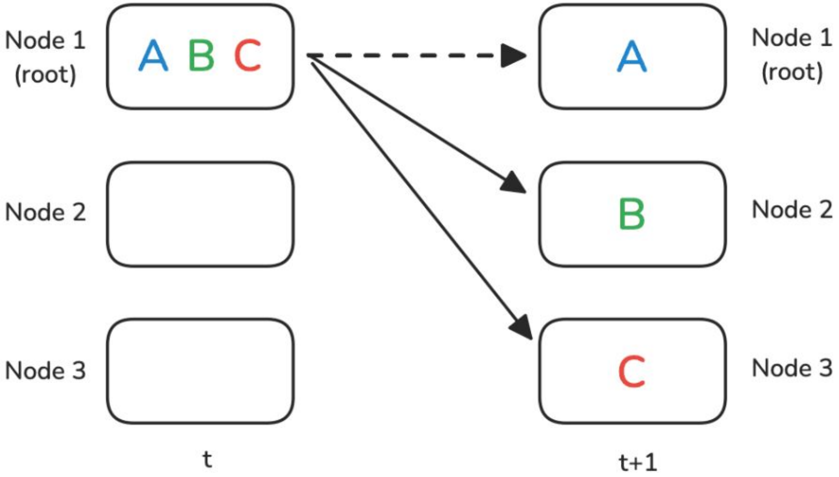
Parameters Gradients Optimizer States

Ψ : Model parameters
k : Optimizer multiplier
N_d: Data parallel degree

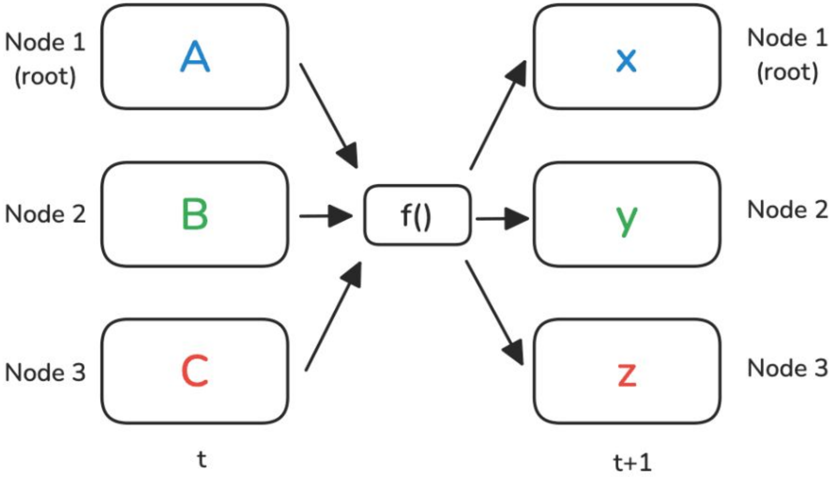
ZeRO 1

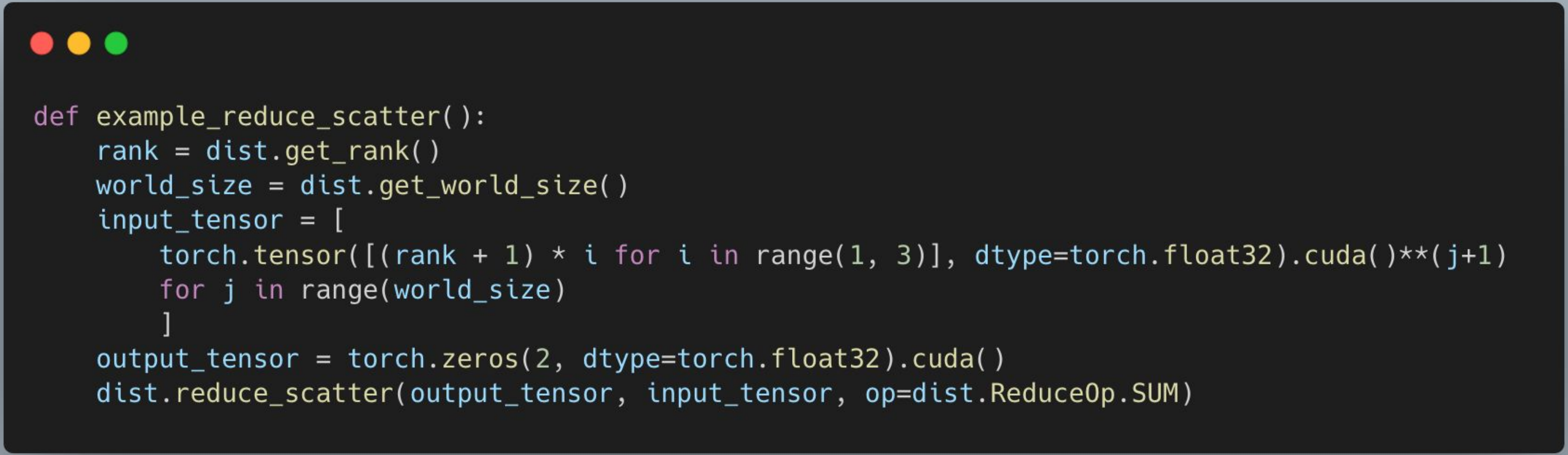

GPU_1
GPU_i
GPU_N
Forward pass
Backward pass
Reduce-scatter
Optimizer step
All-gather
Forward pass (next)
Parameters
Gradients
Optimizer States

Scatter

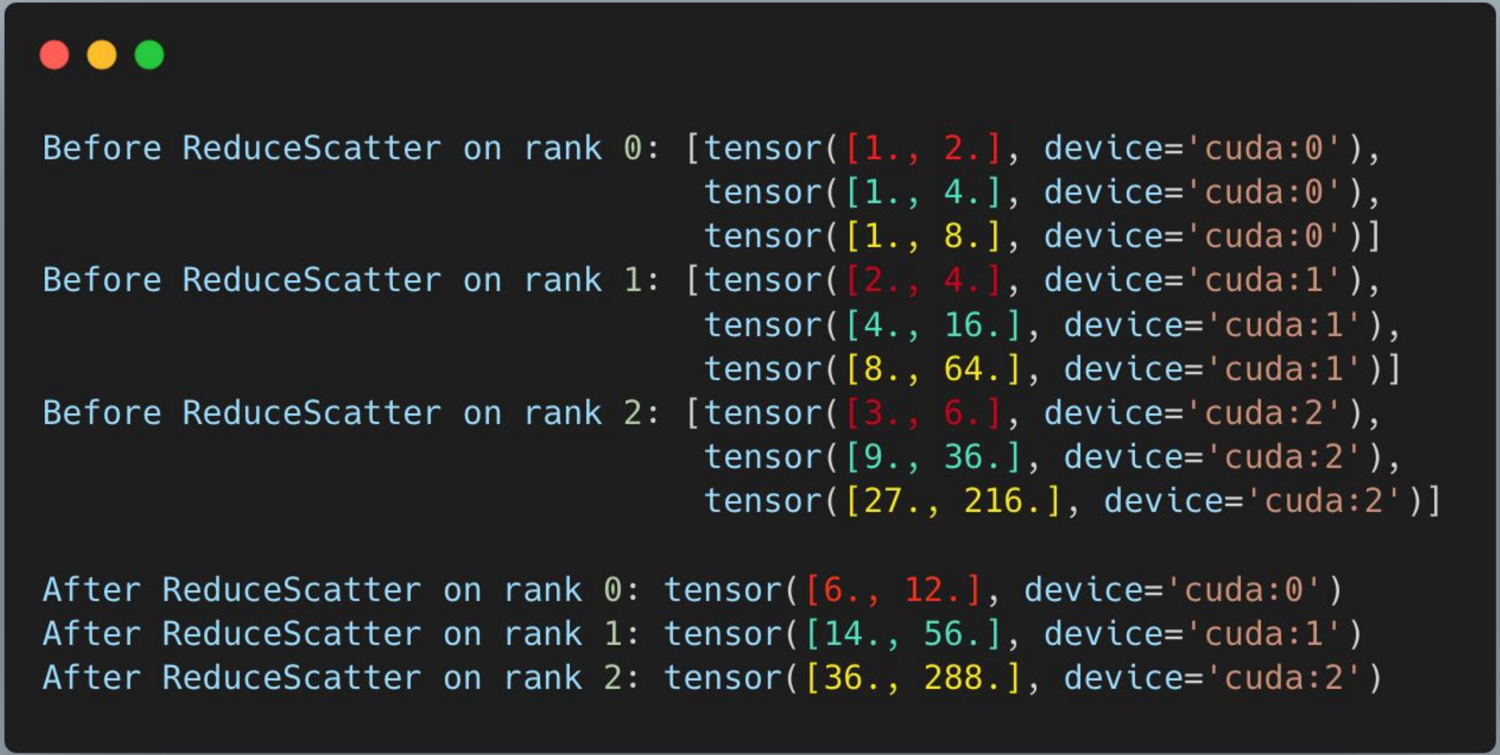


ReduceScatter

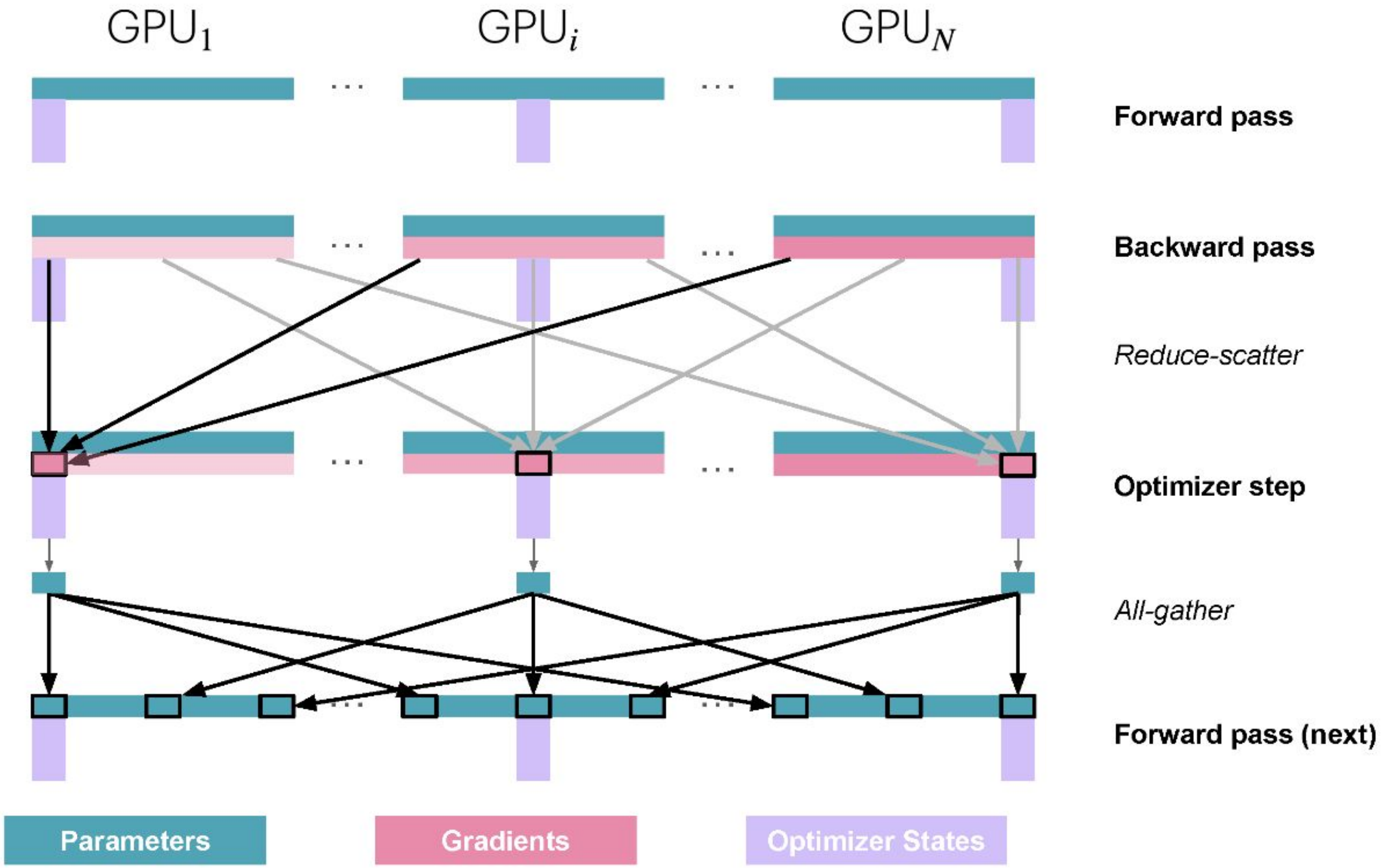


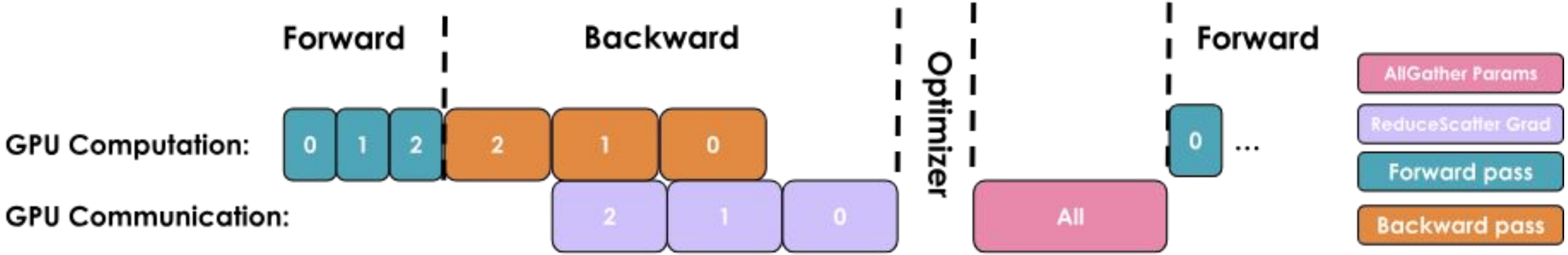


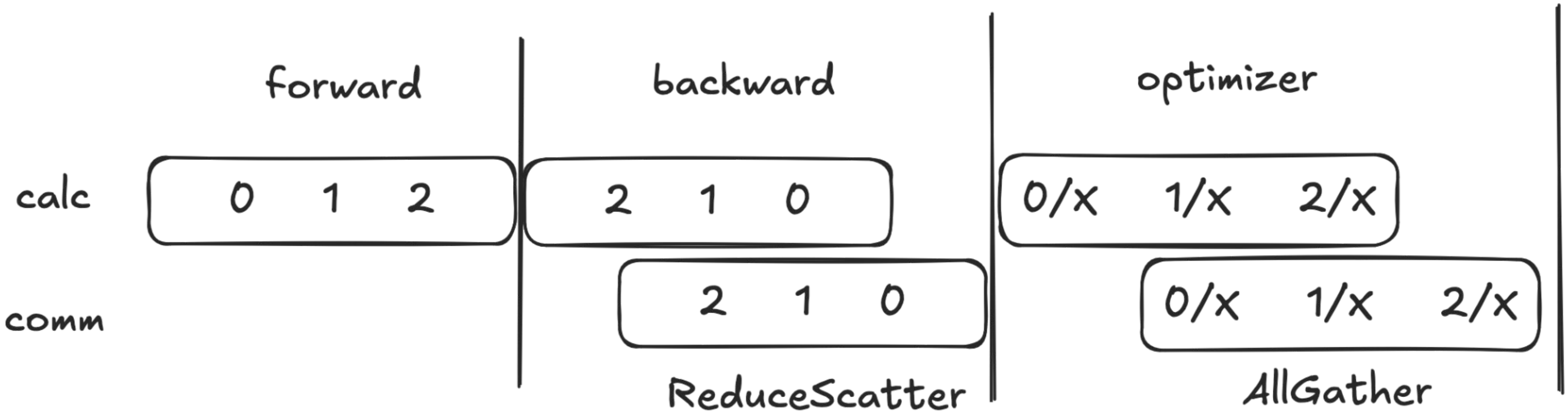
```
def example_reduce_scatter():
    rank = dist.get_rank()
    world_size = dist.get_world_size()
    input_tensor = [
        torch.tensor([(rank + 1) * i for i in range(1, 3)], dtype=torch.float32).cuda()**(j+1)
        for j in range(world_size)
        ]
    output_tensor = torch.zeros(2, dtype=torch.float32).cuda()
    dist.reduce_scatter(output_tensor, input_tensor, op=dist.ReduceOp.SUM)
```



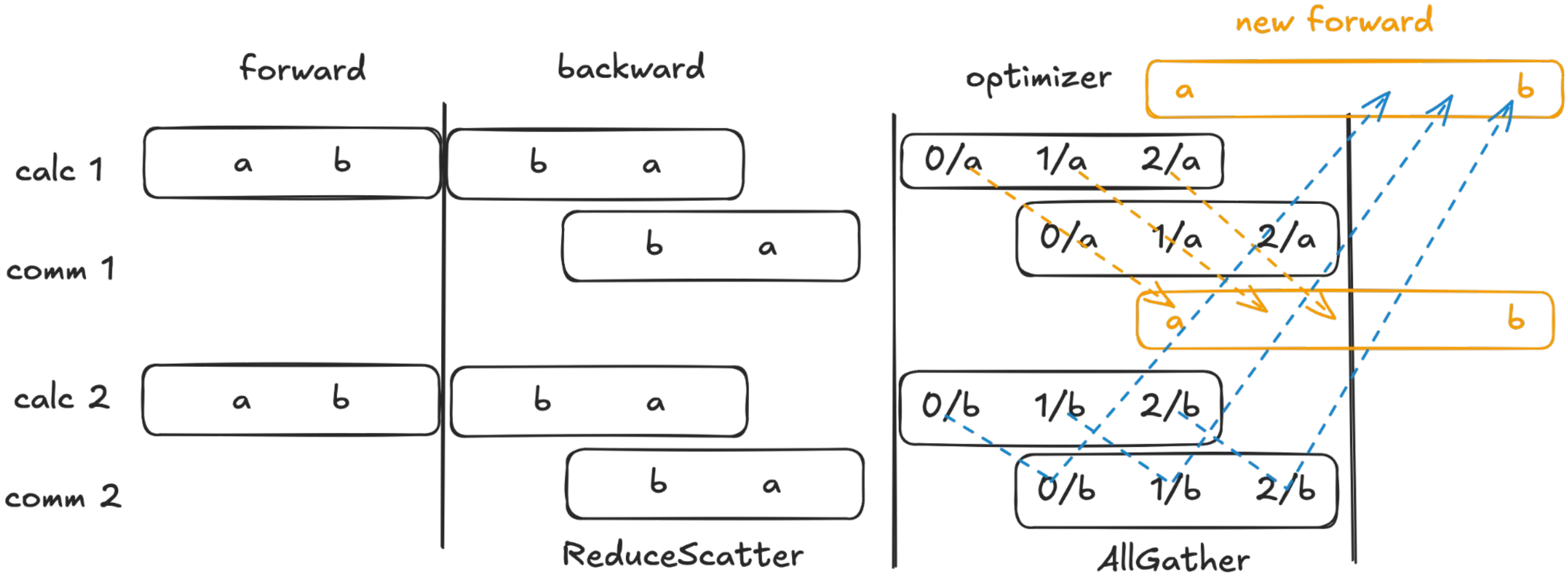
```
Before ReduceScatter on rank 0: [tensor([1., 2.], device='cuda:0'),
                                 tensor([1., 4.], device='cuda:0'),
                                 tensor([1., 8.], device='cuda:0')]
Before ReduceScatter on rank 1: [tensor([2., 4.], device='cuda:1'),
                                 tensor([4., 16.], device='cuda:1'),
                                 tensor([8., 64.], device='cuda:1')]
Before ReduceScatter on rank 2: [tensor([3., 6.], device='cuda:2'),
                                 tensor([9., 36.], device='cuda:2'),
                                 tensor([27., 216.], device='cuda:2')]

After ReduceScatter on rank 0: tensor([6., 12.], device='cuda:0')
After ReduceScatter on rank 1: tensor([14., 56.], device='cuda:1')
After ReduceScatter on rank 2: tensor([36., 288.], device='cuda:2')
```


GPU_1
GPU_i
GPU_N
...
Forward pass
Backward pass
Reduce-scatter
Optimizer step
All-gather
Forward pass (next)
Parameters
Gradients
Optimizer States

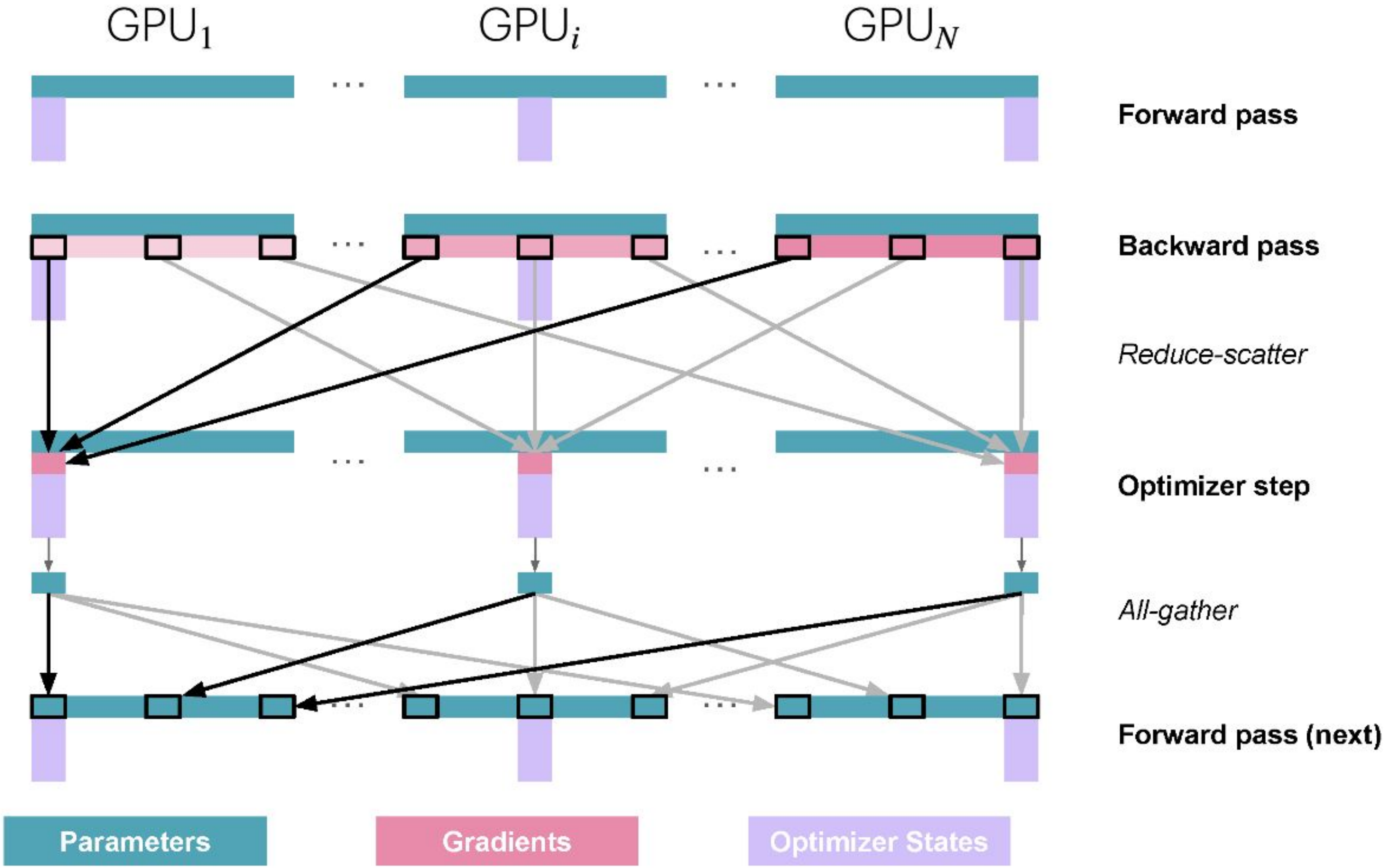

Forward
Backward
Optimizer
Forward
GPU Computation:
0
1
2
2
1
0
0
...
GPU Communication:
2
1
0
All
AllGather Params
ReduceScatter Grad
Forward pass
Backward pass

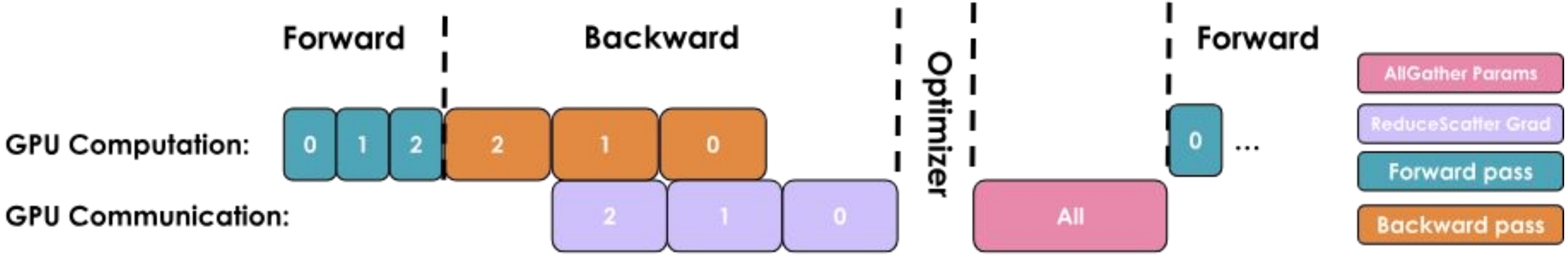


forward
backward
optimizer
calc
0 1 2
2 1 0
0/x 1/x 2/x
comm
2 1 0
0/x 1/x 2/x
ReduceScatter
AllGather

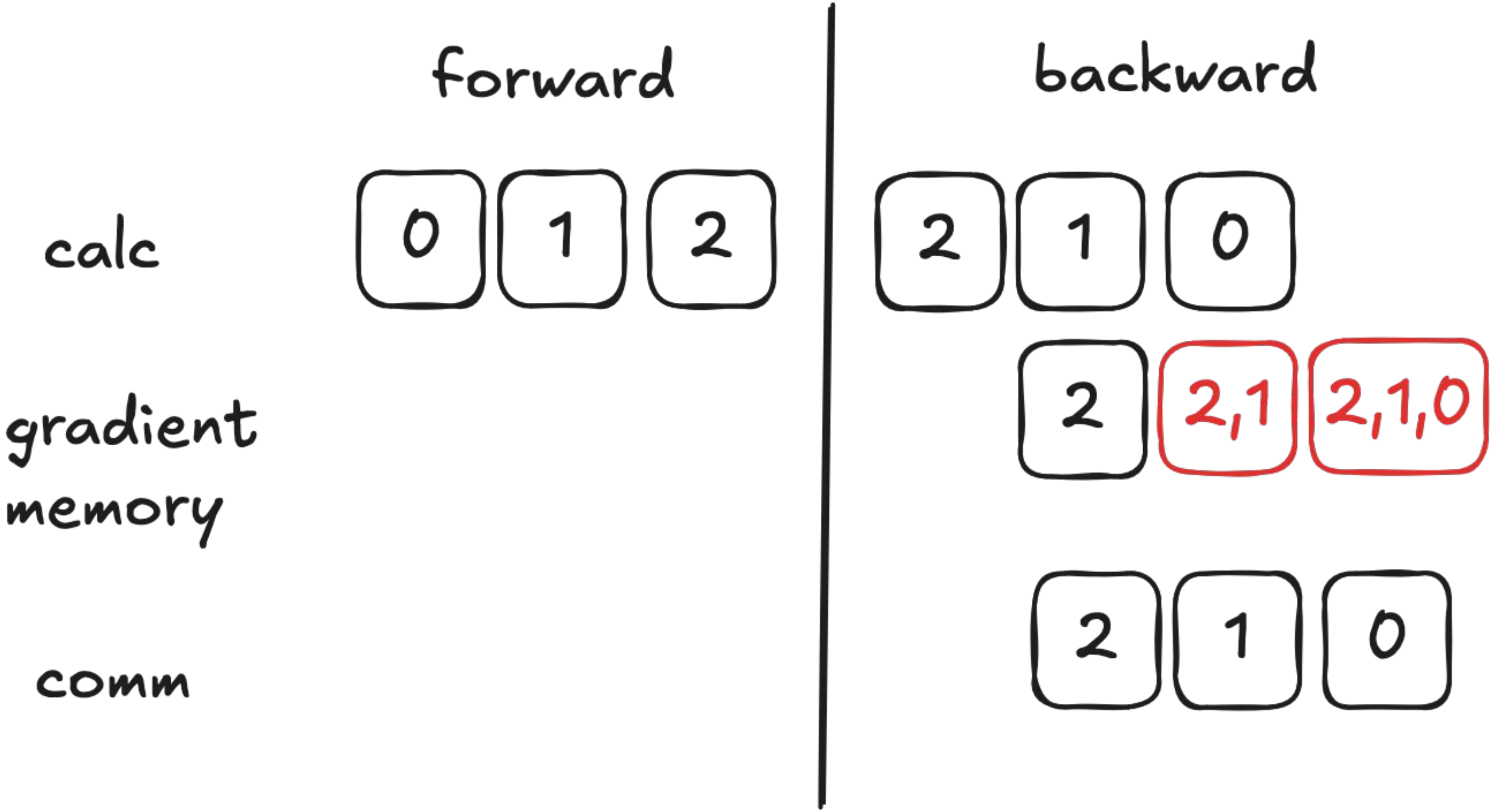

new forward
forward
backward
optimizer
calc 1
comm 1
calc 2
comm 2
a
b
0/a
1/a
2/a
0/b
1/b
2/b
ReduceScatter
AllGather

ZeRO 2

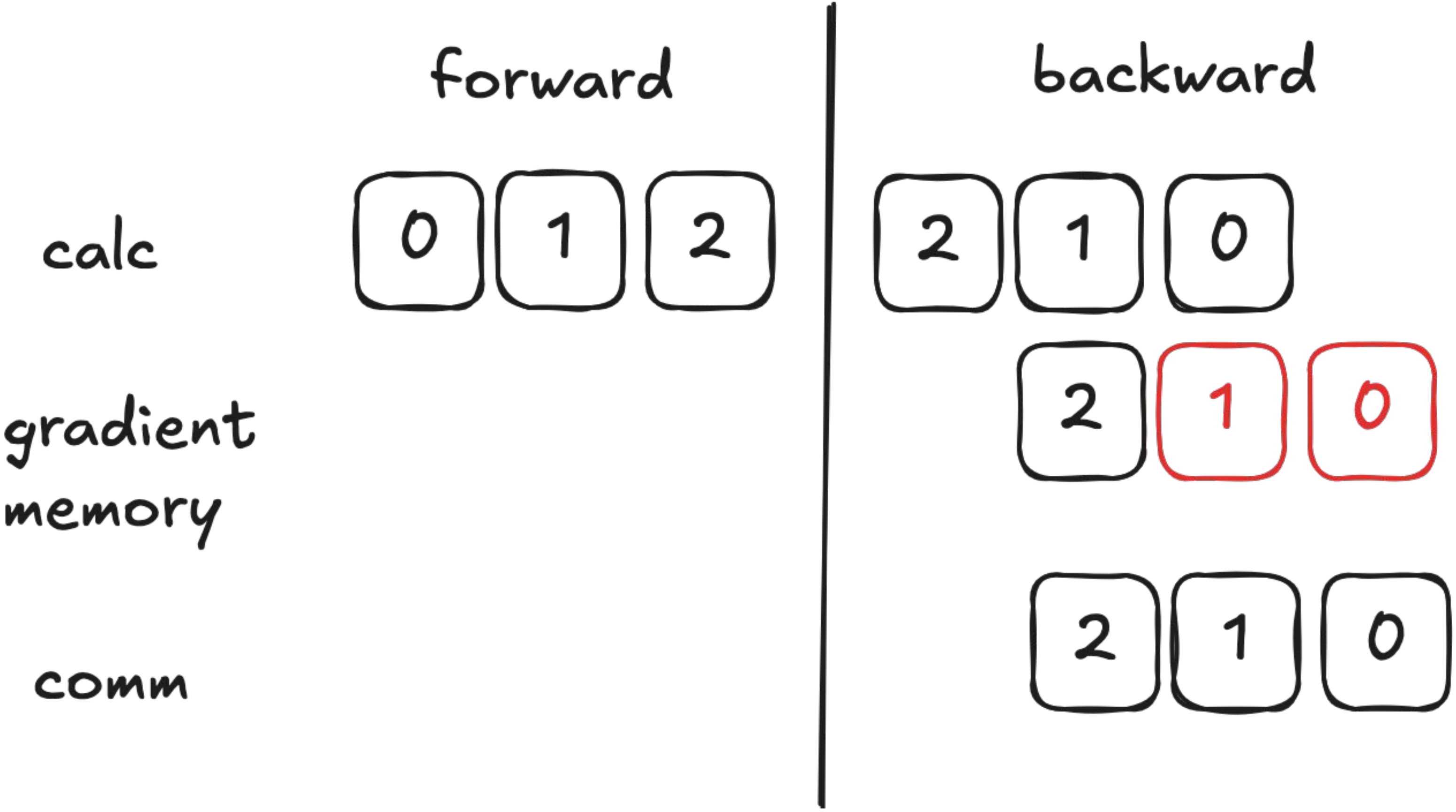

GPU$_1$
GPU$_i$
GPU$_N$
Forward pass
Backward pass
Reduce-scatter
Optimizer step
All-gather
Forward pass (next)
Parameters
Gradients
Optimizer States


Forward
Backward
Optimizer
Forward
GPU Computation:
0
1
2
2
1
0
0
...
GPU Communication:
2
1
0
All
AllGather Params
ReduceScatter Grad
Forward pass
Backward pass

ZeRO-1

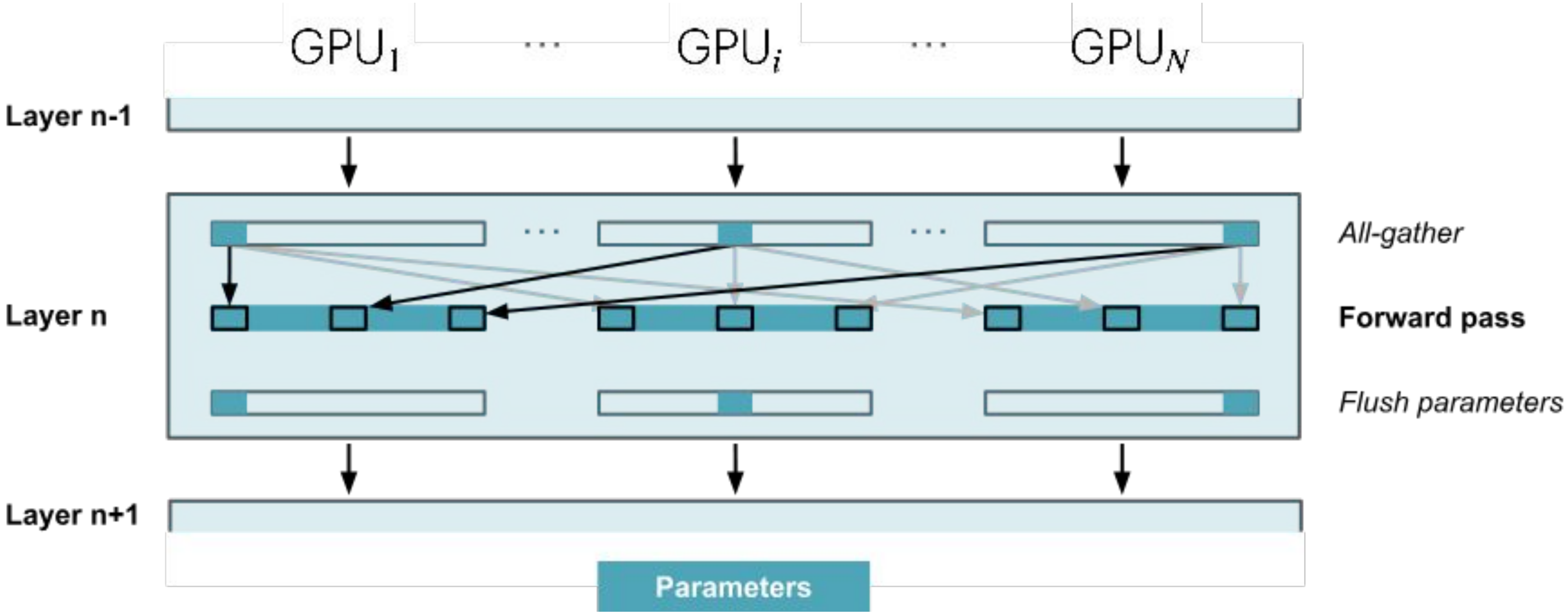

forward
backward
calc
0
1
2
2
1
0
gradient memory
2
2,1
2,1,0
comm
2
1
0

ZeRO-2



ZeRO 3

FSDP (Fully Sharded Data Parallelism) in pytroch



GPU_1 ... GPU_i ... GPU_N

Layer n-1

Layer n

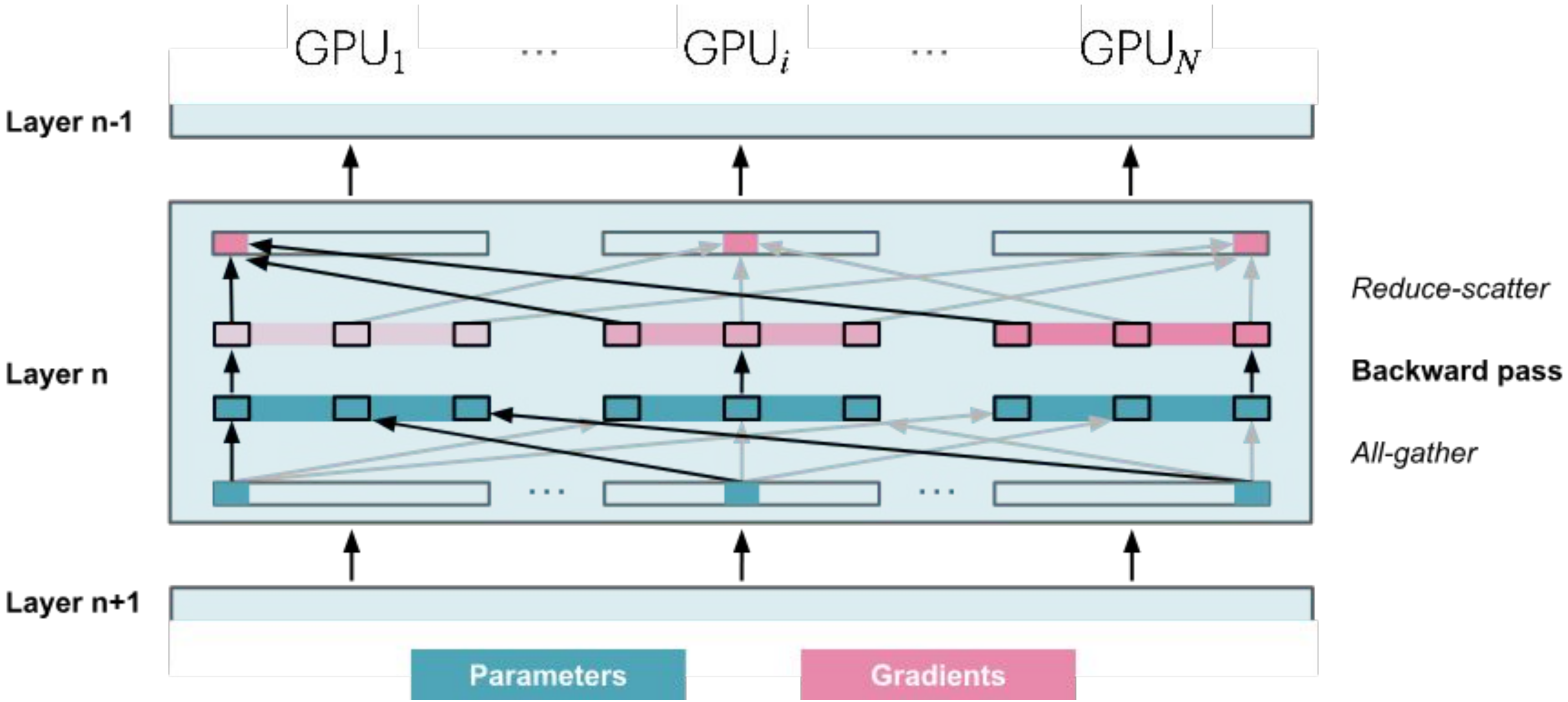
Layer n+1

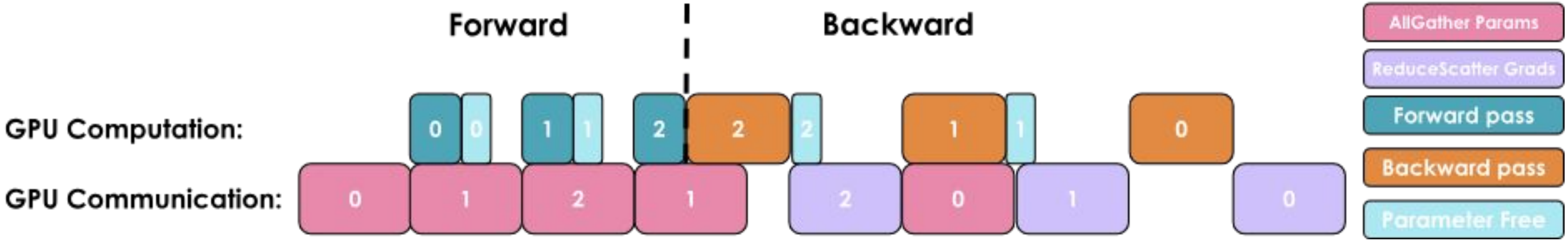
All-gather

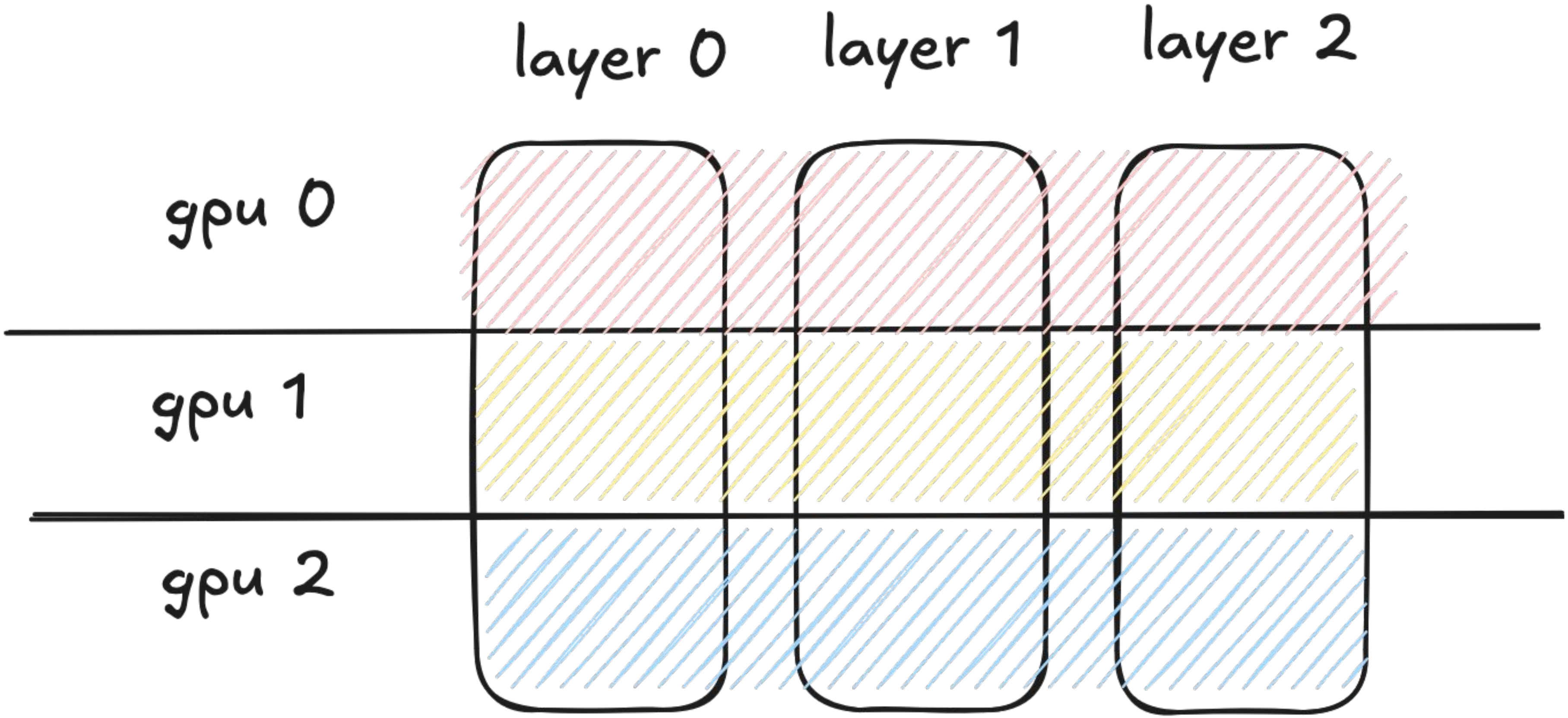
Forward pass

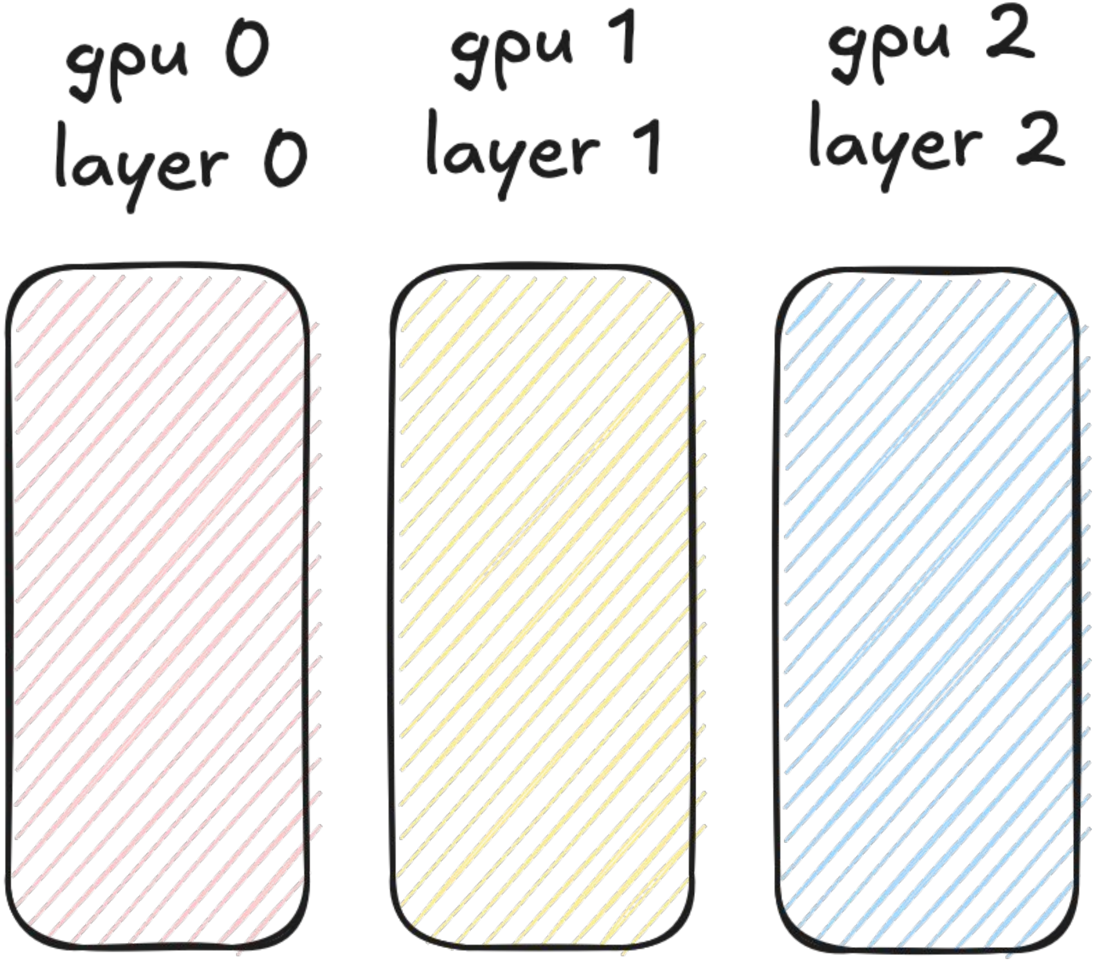
Flush parameters

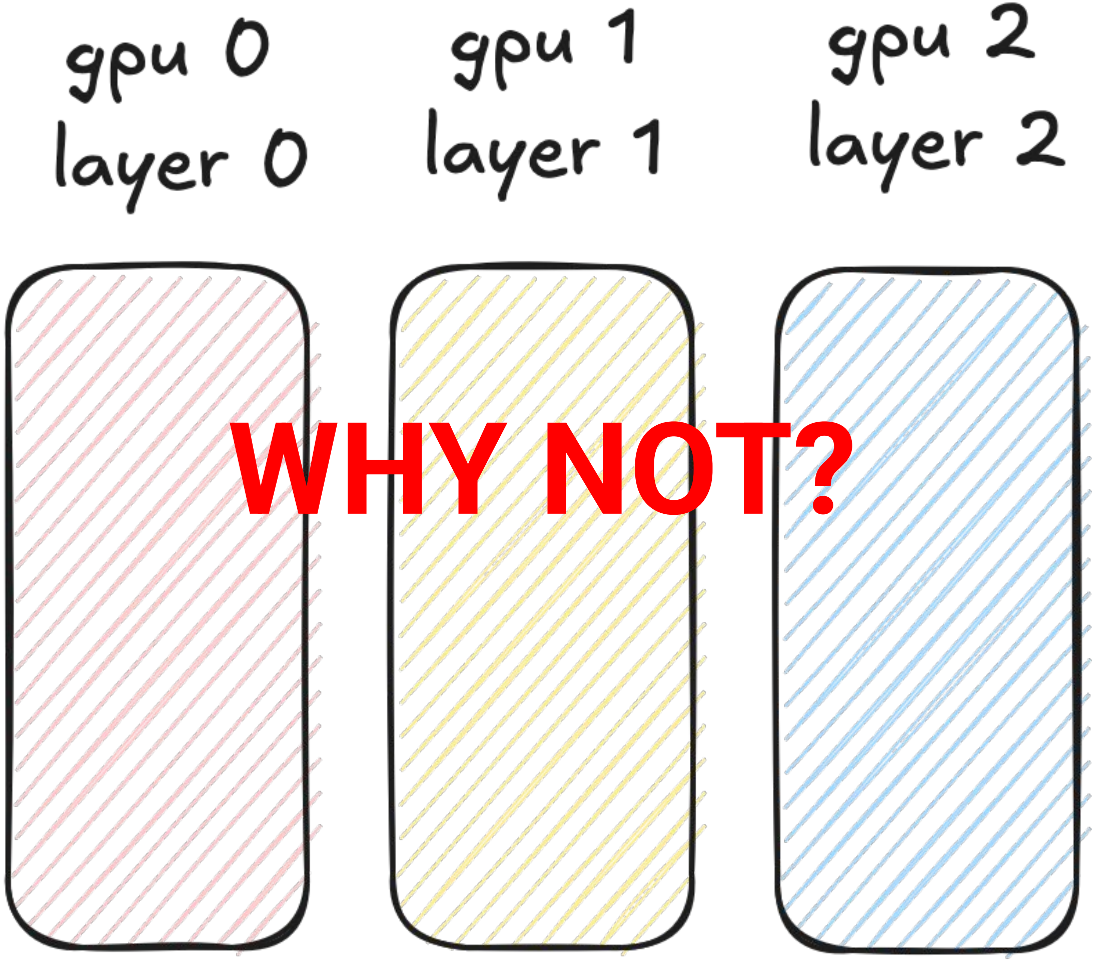
Parameters


GPU$_1$
…
GPU$_i$
…
GPU$_N$
Layer n-1
Layer n
Layer n+1
Reduce-scatter
Backward pass
All-gather
…
…
Parameters
Gradients


Forward
Backward
GPU Computation:
0
0
1
1
2
2
2
1
1
0
GPU Communication:
0
1
2
1
2
0
1
0
AllGather Params
ReduceScatter Grads
Forward pass
Backward pass
Parameter Free


layer 0
layer 1
layer 2
gpu 0
gpu 1
gpu 2


gpu 0
layer 0
gpu 1
layer 1
gpu 2
layer 2

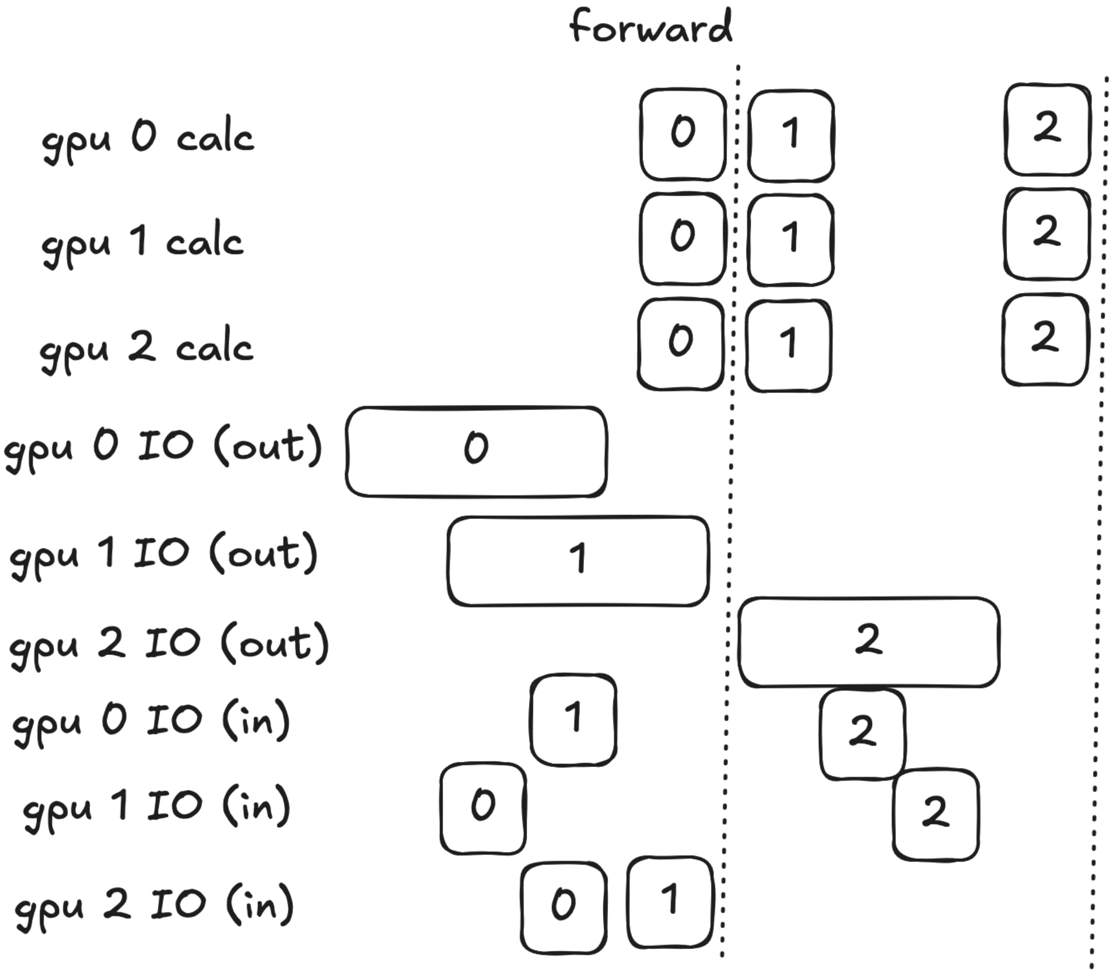


gpu 0
layer 0

gpu 1
layer 1

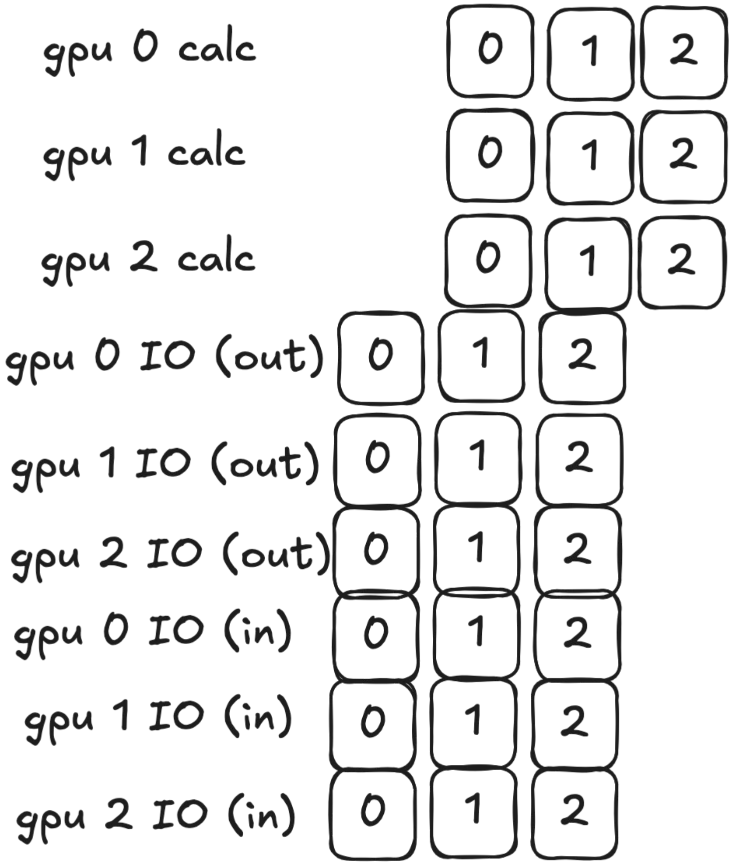
gpu 2
layer 2

WHY NOT?

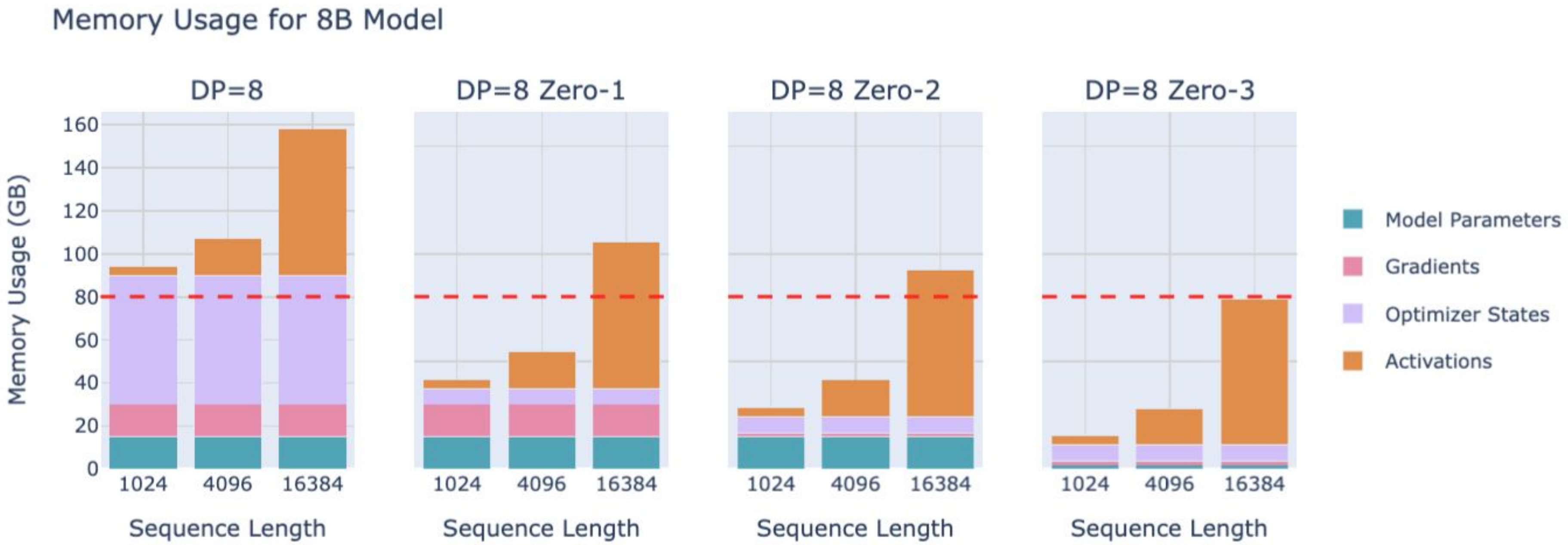


forward
gpu 0 calc
gpu 1 calc
gpu 2 calc
gpu 0 IO (out)
gpu 1 IO (out)
gpu 2 IO (out)
gpu 0 IO (in)
gpu 1 IO (in)
gpu 2 IO (in)
0
1
2
0
1
2
0
1
2
0
1
2
1
2
0
2
0
1

forward



gpu 0 calc		0	1	2
gpu 1 calc		0	1	2
gpu 2 calc		0	1	2
gpu 0 IO (out)	0	1	2	
gpu 1 IO (out)	0	1	2	
gpu 2 IO (out)	0	1	2	
gpu 0 IO (in)	0	1	2	
gpu 1 IO (in)	0	1	2	
gpu 2 IO (in)	0	1	2	



Memory Usage for 8B Model

DP=8

DP=8 Zero-1

DP=8 Zero-2

DP=8 Zero-3

mom,
but I want to play
with veeeeery long
sequences

see you next week for
tensor parallelism