ZeRO: Memory Optimizations Toward Training Trillion Parameter Models

By: Samyam Rajbhandari*, Jeff Rasley*, Olatunji Ruwase, Yuxiong He

Background and Motivation

- **Exponential Model Growth:**
  - GPT2 1.5B, T5 11B, Megatron-LM 8.3B

- **Challenges in Training Large Models**:
  - **Memory Bottlenecks**: Hardware memory limitations
  - **Parallelism Limitations**: Model Parallelism, Data Parallelism, Pipeline Parallelism

Minimize memory usage on GPU while maintain low communication volume and high computational granularity

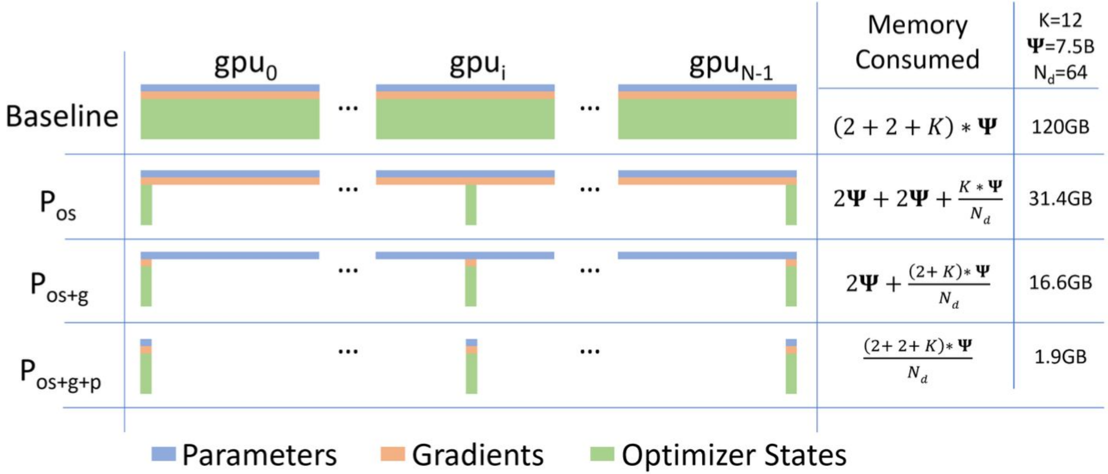
Related Work

Trade offs between functionality usability memory, compute/communication efficiency

- **Parallelism**
  - **Data Parallelism (DP)**: Replicates the full model across devices, causing redundant memory consumption and limiting scalability.
  - **Model Parallelism (MP)**: Splits models across devices (vertically) but incurs significant communication overhead, especially across nodes.
  - **Pipeline Parallelism (PP)**: Splits the model across layers (horizontally) and devices but is complex and has limitations with tied weights and batch normalization
- Non- Parallelism:
  - **Reducing Activation Memory**
  - **CPU Offload**: Storing model state to CPU memory

Zero Redundancy Optimizer (ZeRO)

- Model State:
  - Optimizer states, gradients and parameters
  - **ZeRO-DP**
- Residual State:
  - Activations, temporary buffers, unusable fragmented memory
  - **ZeRO-R**



Parameter: 2Ψ
Gradient: 2Ψ
Optimizer: 12Ψ

- Parameters: 4Ψ
- Momentum: 4Ψ
- Variance: 4Ψ

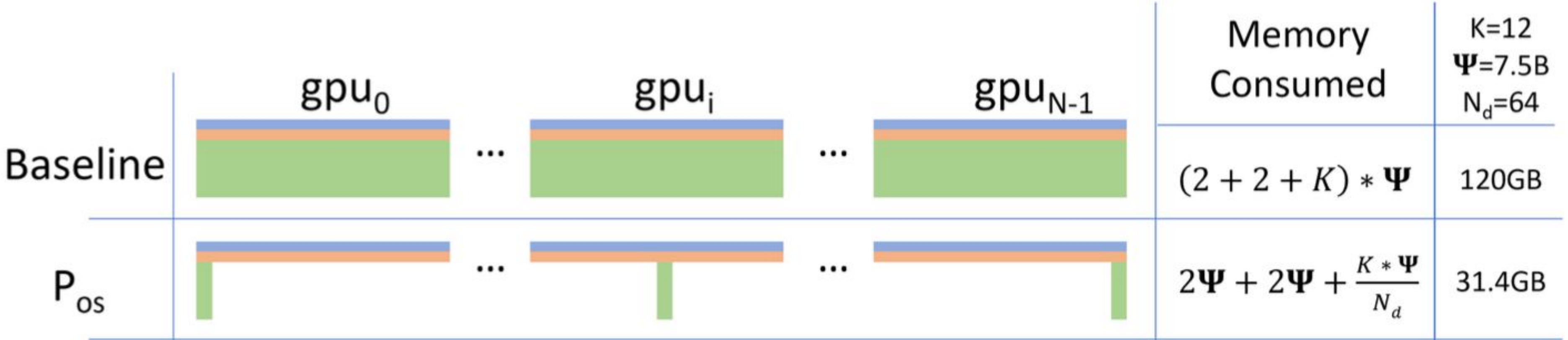
ZeRO-DP

- DP vs MP
    - DP has more inefficiency memory-wise
    - DP has better scaling efficiency
    - Both keep all model states

Eliminates the memory redundancy by **partitioning** the optimizer states, gradients and parameters across data parallel process.

ZeRO-DP

- **Optimizer State Partitioning (Pos)**:
  - group the optimizer states into Nd equal partitions, each data parallel process only needs to store and update 1/Nd of the total optimizer states and then only update 1/Nd of the parameters.
  - **Memory**: Reduces memory by 4x
  - **Communication**: After each training step, an all-gather operation (Ψ) is performed across all devices to ensure all optimizer states are synchronized.



ZeRO-DP

- **Add Gradient Partitioning (Pos + g)**:
  - Only the gradients for the corresponding partitioned parameters are needed and was bucked to be update at once
  - **Memory**: Reduces memory by 8x
  - **Communication**: requires a scatter-reduce operation (Ψ)

ZeRO-DP

- **Add Parameter Partitioning (Pos + g+p)**:
  - Similarly each process only stores the parameters corresponding to its partition.
  - **Memory**: Reduces memory by 16x
  - **Communication**: An addition (Ψ) needed as parameters are all-gathered on-demand during forward propagation, but communication is kept efficient through pipelining and only fetching the parameters needed for the current operation.
  - Communication in total 3Ψ = 1.5 * 2 Ψ

ZeRO-R

- **Tackle remaining memory issue:**
  - **Activations:**
    - eg. GPT2 with 1.5B parameter sequence length of 1k and batch size of 32 needs 60GB
    - Activation checkpoint 8Gb
  - **Temporary buffer**
    - eg. with 1.5B parameter model, fp32 buffer needs 6GB of memory
    - Buffer size would change with the model size (non-trivial)
  - **Unusable memory fragments**
    - Not enough contiguous memory blocks even though there are more free memory than needed
    - OOM when 30% of memory still available

ZeRO-R

- **Partitioned Activation Checkpointing (Pa/+cpu)**:
  - After the forward pass, the activations are split (partitioned) across GPUs, and they are only gathered when needed during the backward pass.
  - **On-demand Reconstruction**: When a GPU needs an activation that it doesn't have locally for the backward pass, it gathers the required data from other GPUs through an **all-gather** operation.
  - **Memory**: Reduces memory by the layer by Mp degrees

ZeRO-R

- **Constant Size Buffer (Cb)**:
  - Previously buffer size is proportional to model size
  - ZeRO-R caps the buffer size at a constant value. This prevents buffer memory from becoming unmanageable as models grow.
- **Memory Defragmentation (Md):**
  - performs **on-the-fly memory defragmentation** by pre-allocating contiguous memory chunks for activations and gradients.

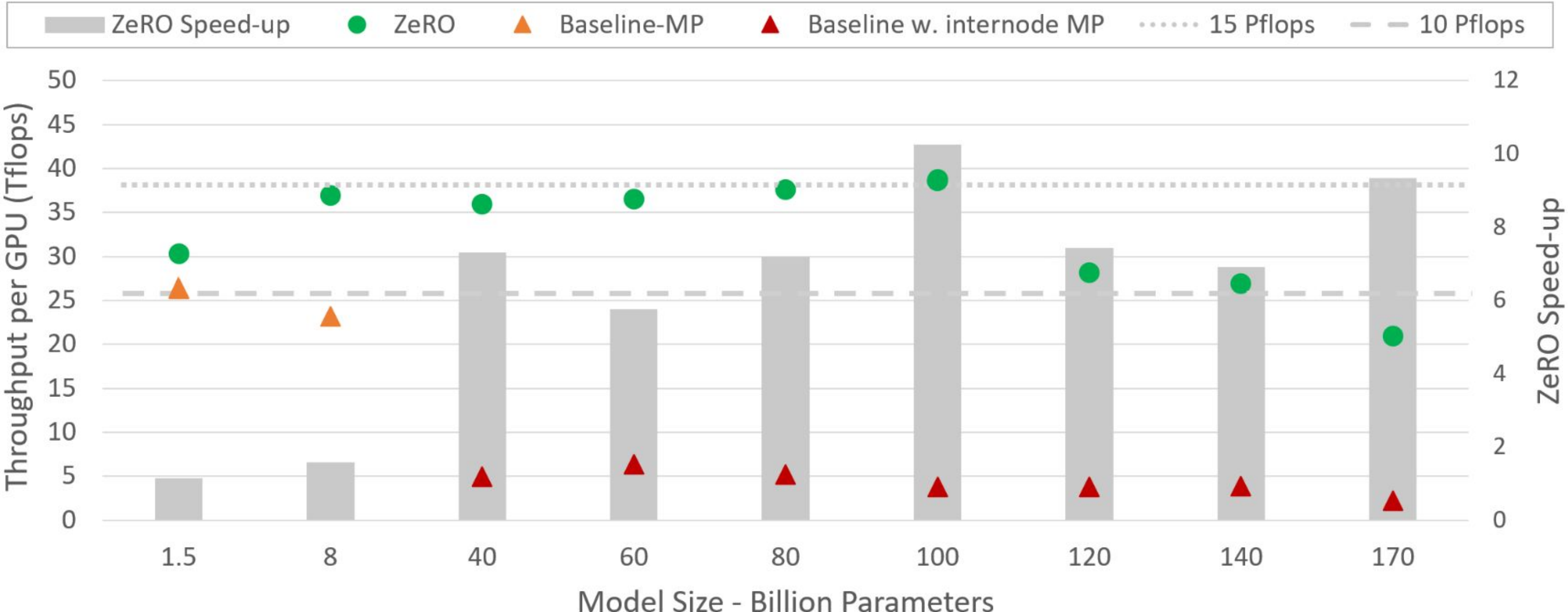
Evaluation Setup and Result

- **400 NVIDIA V100 GPUs** (distributed across 25 DGX-2 nodes) with **800 Gbps inter-node communication bandwidth**.
- **ZeRO-100B:** (Pos+g and ZeRO-R)
  - efficiently run models with up to 170B parameters on 400 GPUs, more than 8x bigger than Megatron-LM.

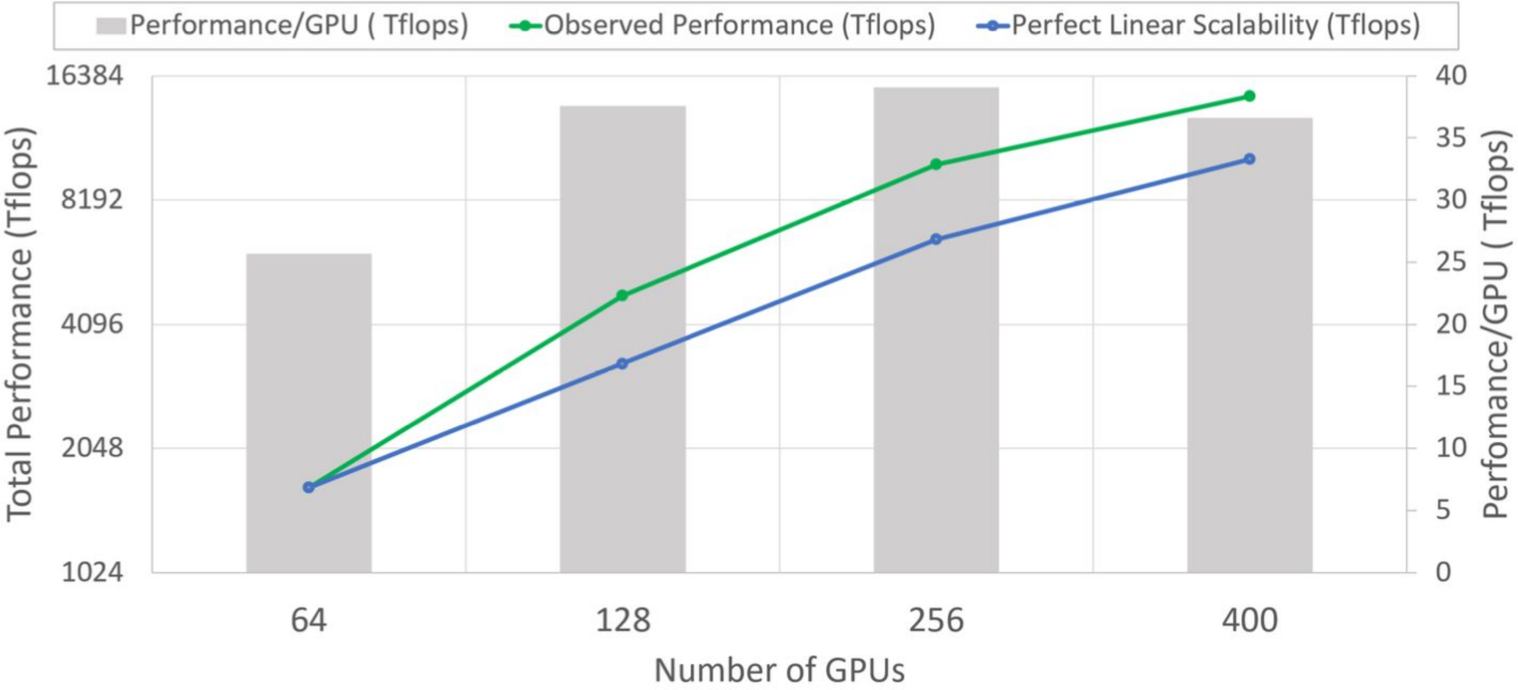
MP	GPUs	Max Theoretical Model Size				Measured Model Size	
		Baseline	P_{os}	P_{os+g}	P_{os+g+p}	Baseline	*ZeRO*-DP (P_{os})
1	64	2B	**7.6B**	14.4B	128B	1.3B	**6.2B**
2	128	4B	**15.2B**	28.8B	256B	2.5B	**12.5B**
4	256	8B	**30.4B**	57.6B	0.5T	5B	**25B**
8	512	16B	**60.8B**	115.2B	1T	*10B*	**50B**
16	1024	32B	**121.6B**	230.4B	*2T*	20B	**100B**

Speed and Model Size

Successfully run models with up to 170B parameters on 400 GPUs



Superlinear Scalability



Optimizations Analysis

Used 5 different combinations of ZeRO-DP + ZeRO-R

	ZeRO-DP	*ZeRO*-R
1	P_{os}	C_B+M_D
2	P_{os}	C_B+M_D+P_a
3	P_{os+g}	C_B+M_D
4	P_{os+g}	C_B+M_D+P_a
5	P_{os+g}	C_B+M_D+P_{a+cpu}

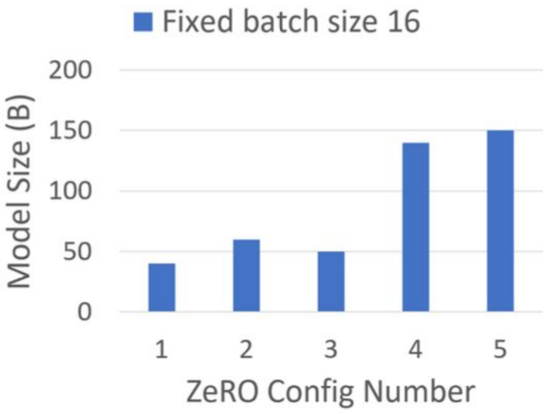


Figure 6: Max model size

Figure 7: Max cache allocated.

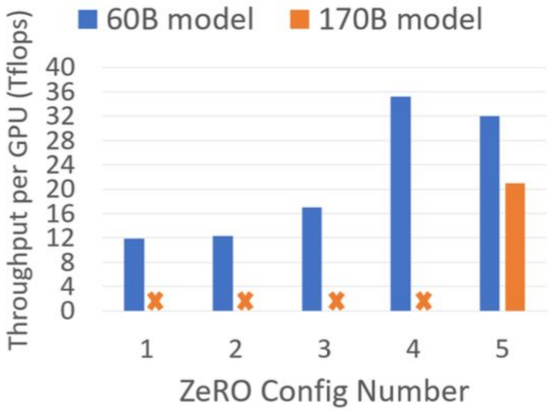


Figure 8: Throughput per GPU.

Thoughts

Strength:

- reduces memory bottlenecks
- Allow model size to increase dramatically
- Easy interface

Weakness:

- Would the fix buffer size still be applicable with trillion and trillions parameter model?
- It would be nice if in the experiment they also add the parameter partition to show the communication overhead vs memory