The Ultra-Playbook Cheatsheet

Step 1: Fit model into memory

GPU rich case: 🤑

- **Small models (<10B):** use a single parallelism technique, e.g. TP or ZeRO-3/DP with Full Recompute across 8 GPUs.
- **Large models (10B+):** requires more than 8 GPUs, you have several options:
  - Combining Tensor Parallelism (TP=8) with Pipeline Parallelism
  - Combining Tensor Parallelism (TP=8) with Data Parallelism (ZeRO-3)
  - Using only ZeRO-3 (i.e. only pure Data Parallelism)
- **512+ GPU scale**: pure DP/ZeRO-3 becomes inefficient due to communication cost - better to then combine DP with either TP or PP
- **1024+ GPU scale**, a recommended setup can be TP=8 with DP (ZeRO-2) and PP
- Special cases: for **long context** consider CP and for **MoE** arch use EP

GPU poor case 😭:

- **Reduce memory:** use full activation checkpointing and/or gradient accumulation

Step 2: Satisfy target global batch size

Experiments tell us which batch size is ideal for training (4-40M tokens). So we either have to increase or decrease the batch size based on step 1 to meet it.

Increase Global Batch Size:

- Scale up DP or CP or gradient accumulation steps

Decrease Global Batch Size:

- Reduce DP or CP in favor of other parallelization strategies

Step 3: Optimizing Training Throughput

There is no general recipe for the best configuration so at this point we should experiment:

- **Scale up TP** up to the node size to reduce other parallel strategies
- **Increase DP** with ZeRO-3 while keeping target GBS
- **Use PP** if communication becomes a bottleneck for DP
- Play with **micro batch size** to balance max GBS, model size, compute/comms

Parallelization Strategies

Strategy	Batch Size	Memory Reduction	Compute Reduction	Communication	Compute/Communication Overlap
Data Parallelism	gbs scales with DP	can reduce mbs by increasing dp → reduce activations	can reduce mbs by increasing dp	bwd: allreduce grads_bf16	overlapped with microbatch's backward: `(DP-1) * num_params * peak_flops / (2 * peak_bw * num_tokens * DP)`
DP+ZeRO-1	gbs scales with DP	model_fp32/dp optimstates/dp	can reduce mbs by increasing dp	bwd: allreduce grads_bf16 step_end: allgather model_fp32	Same as above
DP+ZeRO-2	gbs scales with DP	model_fp32/dp grads_fp32/dp optimstates/dp	can reduce mbs by increasing dp	bwd: reduce-scatter grads_bf16 step_end: allgather model_fp32	overlapped with microbatch's backward: `(DP-1) * num_params * peak_flops / (4 * peak_bw * num_tokens * DP)`
DP+ZeRO-3 (FSDP)	gbs scales with DP	model_bf16/dp model_fp32/dp grads_fp32/dp optimstates/dp	can reduce mbs by increasing dp	(x num_layers) fwd: allgather model_fp32 bwd: allgather model_fp32 bwd: reduce-scatter grads_fp32	overlapped with next layer's fwd/bwd: `(DP-1) * peak_flops / (2 * seq * mbs * peak_bw)`
Tensor Parallelism	No effect	model_bf16/tp model_fp32/tp grads_fp32/tp optimstates/tp activs/tp	model_bf16/tp	(x 4 x num_layers) fwd: allgather activs_bf16 bwd: reduce-scatter grads_bf16	overlapped with next TP region (attn/MLP/layernorm): `(TP-1) * peak_flops / (24 * hidden_size * peak_bw)`
Pipeline Parallelism (1f1b)	prefers large gas to reduce bubble	model_bf16/pp model_fp32/pp grads_fp32/pp optimstates/pp	model_bf16/pp	(x gas) fwd: recv activs_bf16 fwd: send activs_bf16 bwd: recv grads_bf16 bwd: send grads_bf16	overlapped with next microbatch's fwd/bwd: `PP * peak_flops / (32 * hidden_size * num_layers * peak_bw)`
Context Parallelism	prefers large seq for better overlap	activations/cp	seq/cp	(x cp-1 x num_layers) fwd: send/recv activs_bf16 bwd: send/recv grads_bf16	Overlap with attention computation(ring attention): `(CP-1) * B * L/CP * H_kv × (num_k + num_v)`
Expert Parallelism	Batch size scales with EP	experts/ep	experts/ep	(x num_layers) fwd: all2all activs_bf16 bwd: all2all grads_bf16	overlapped with MoE block `(EP-1) * peak_flops / (12 * num_experts * hidden_size * peak_bw)`

Glossary

Parallelization Terms:

- tp: Tensor parallelism degree
- pp: Pipeline parallelism degree
- dp: Data parallelism degree
- cp: Context parallelism degree
- ep: Context parallelism degree
- dp_if_zero1/2/3: Effective data parallelism when using ZeRO stages (if ZeRO2 is used means both dp_if_zero1 and dp_if_zero2 are used)

Batch Size Terms:

- mbs: Micro batch size per GPU
- gas: Gradient accumulation steps
- mseqlen: Sequence length per GPU (after CP)
- GBS: Global batch size = mbs * dp * gas * mseqlen

Memory Terms:

- model_bf16: Model parameters in bfloat16 format = model_bf16(model_config,tp,pp,dp_if_zero3)
- model_fp32: Model parameters in float32 format (for optimizer) = 2 * model_bf16 / dp_if_zero1
- grads_fp32: Gradients in float32 format = 2 * model_bf16 / dp_if_zero2
- optimstates: Optimizer states (e.g. Adam momentum/variance) in float32 = 4 * model_bf16 / dp_if_zero1
- activs: Activation tensors from forward pass = activs(model_config,mseqlen,mbs,tp,cp,pp,dp_if_zero3)

Useful formulas:

Total peak memory usage per GPU for a training step can be approximated as:
`peak_memory = model_bf16 + model_fp32 + grads_fp32 + optimstates + activs`
where `model_bf16 = bf16_bytes * num_params = 2 * num_layers * 16 * hidden_size^2`

Compute per GPU for a training step can be approximated as:
`compute = 6* model_bf16 * mbs * seq * gas`