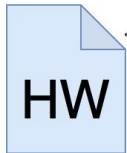


Calculon: a Methodology and Tool for High-Level Codesign of Systems and Large Language Models

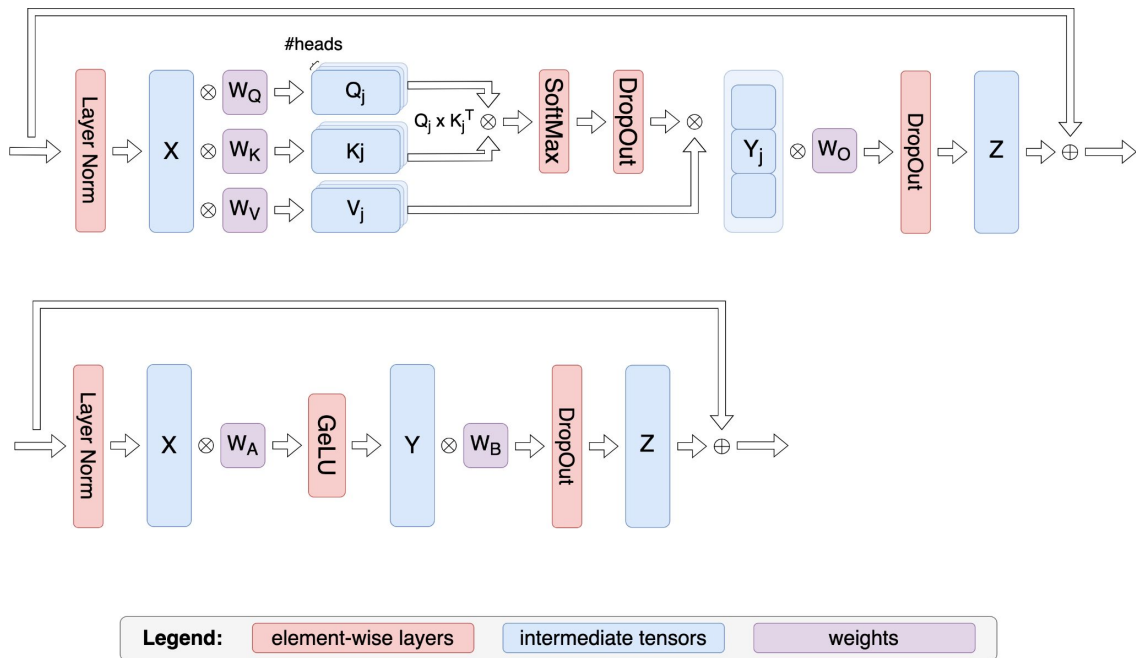
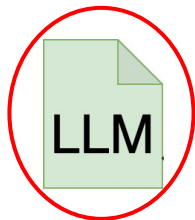
Mikhail Isaev¹, Nic McDonald²,
Larry Dennison², Rich Vuduc¹

¹Georgia Institute of Technology, ²Nvidia

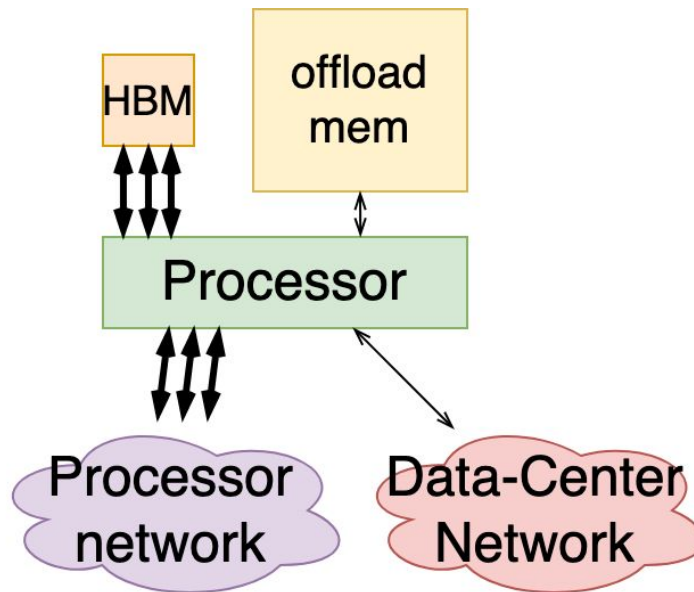
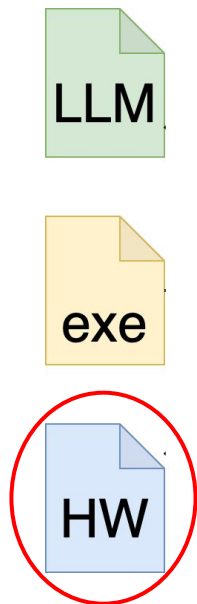
Motivation: future LLM modeling for future HW



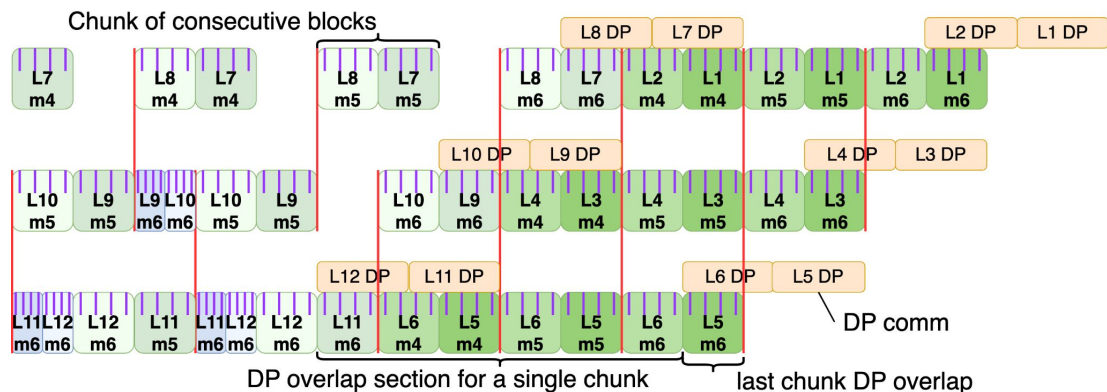
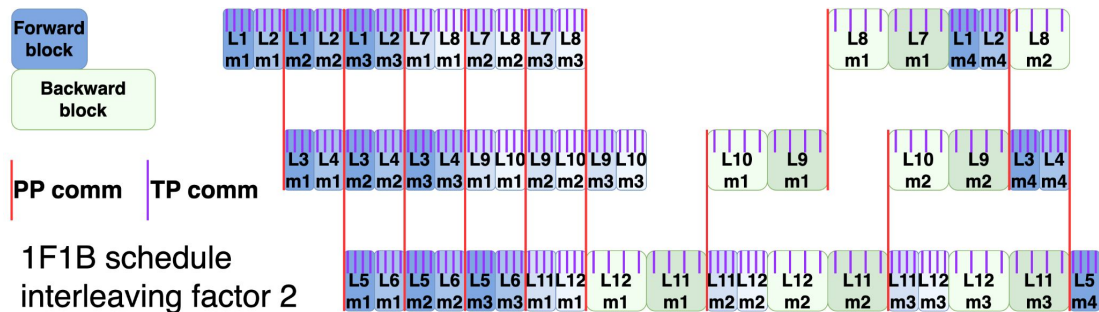
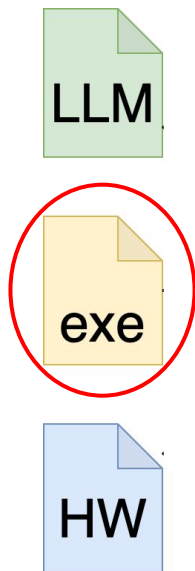
Motivation: future LLM modeling for future HW



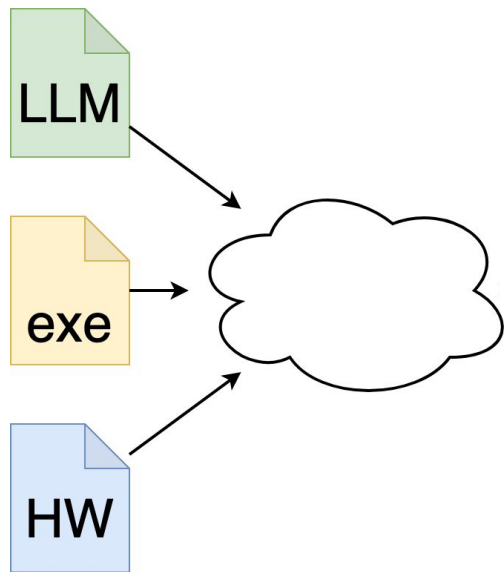
Motivation: future LLM modeling for future HW



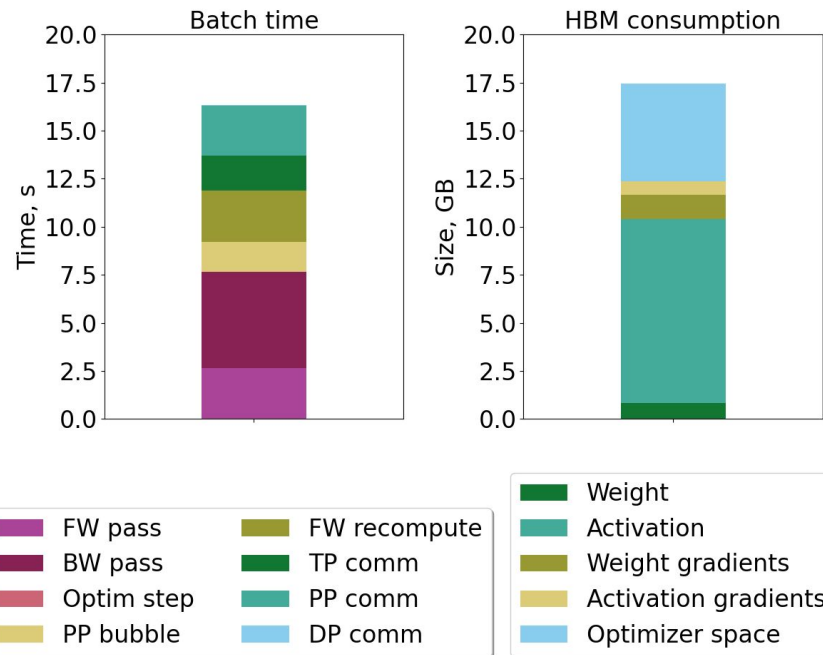
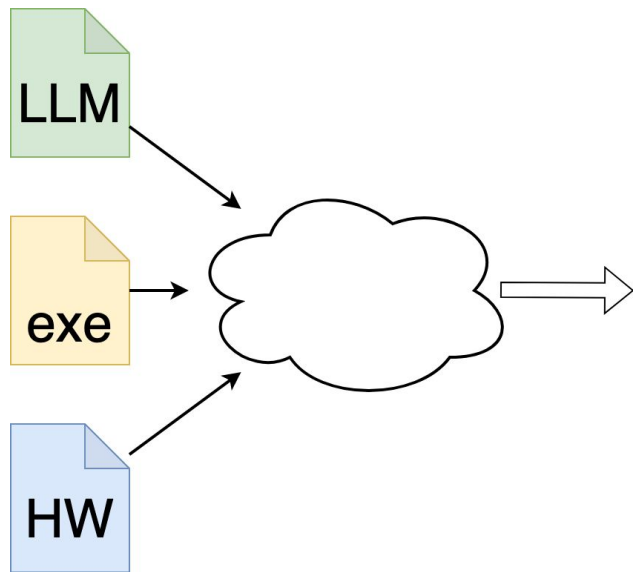
Motivation: future LLM modeling for future HW



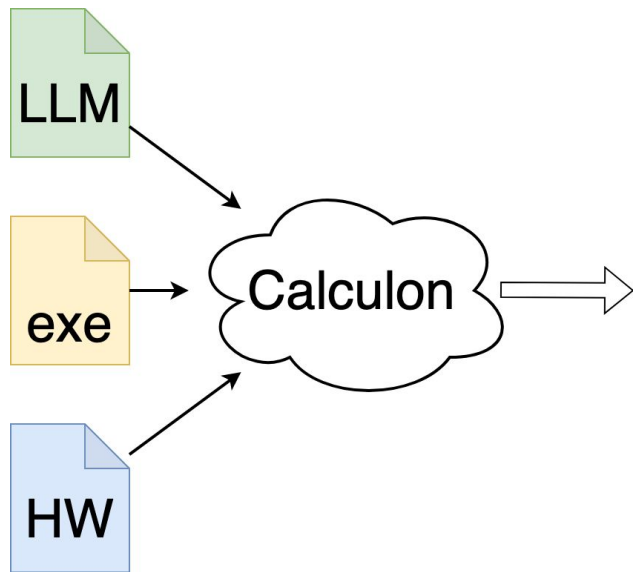
Motivation: future LLM modeling for future HW



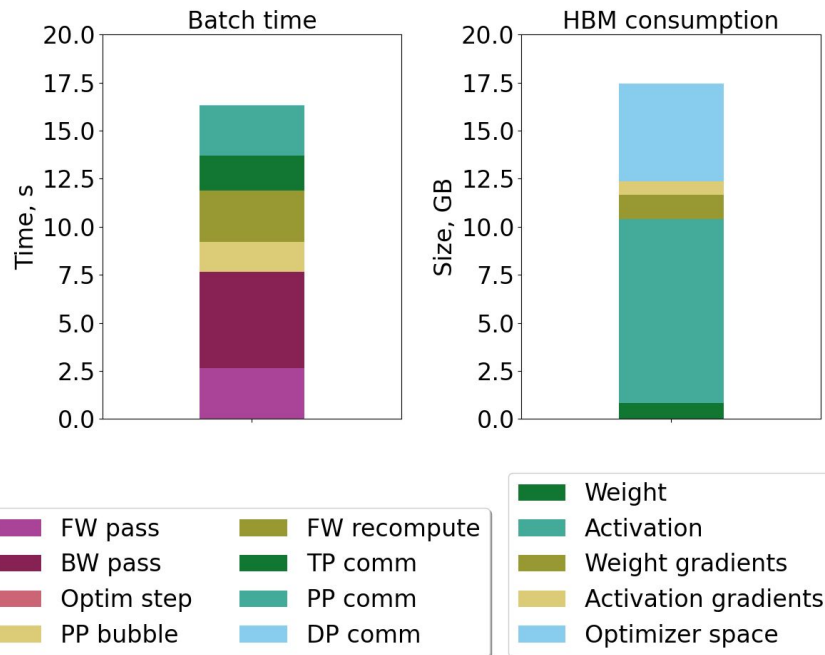
Motivation: future LLM modeling for future HW



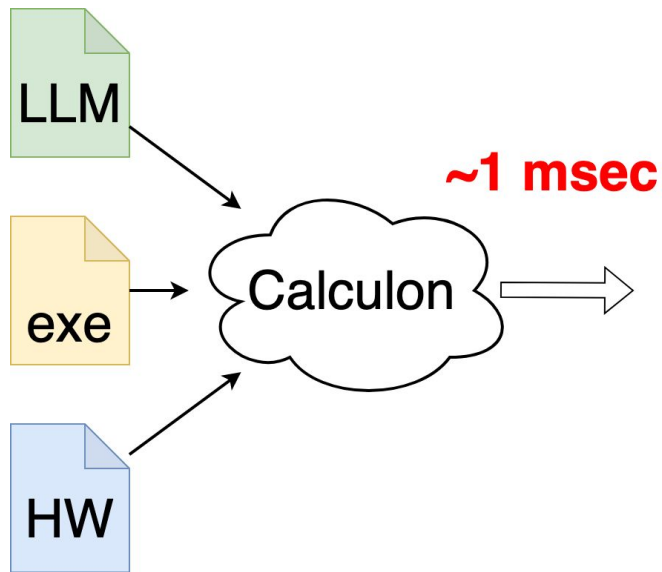
Modeling with Calculon



Calculon – analytical model of LLM.

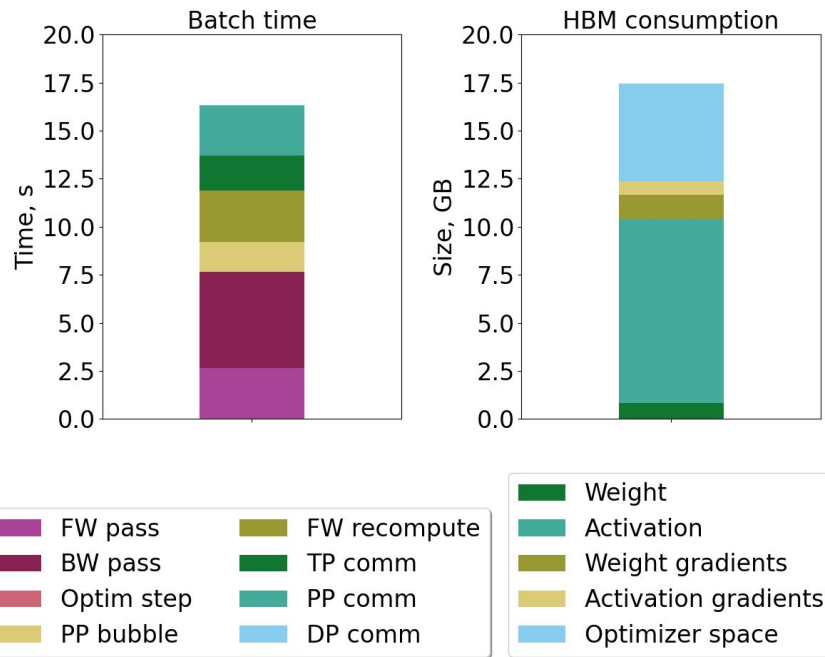


Modeling with Calculon

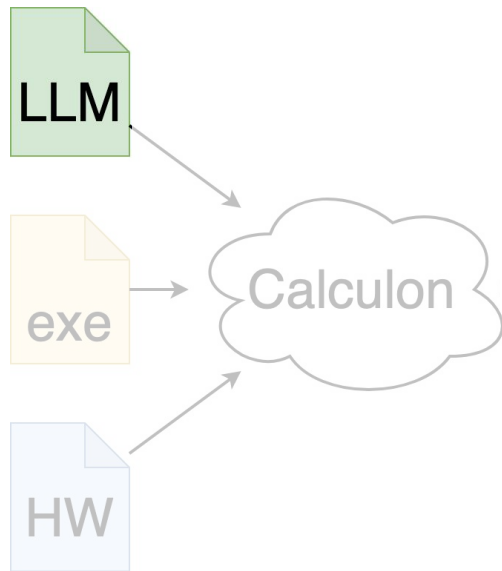


Calculon – analytical model of LLM.

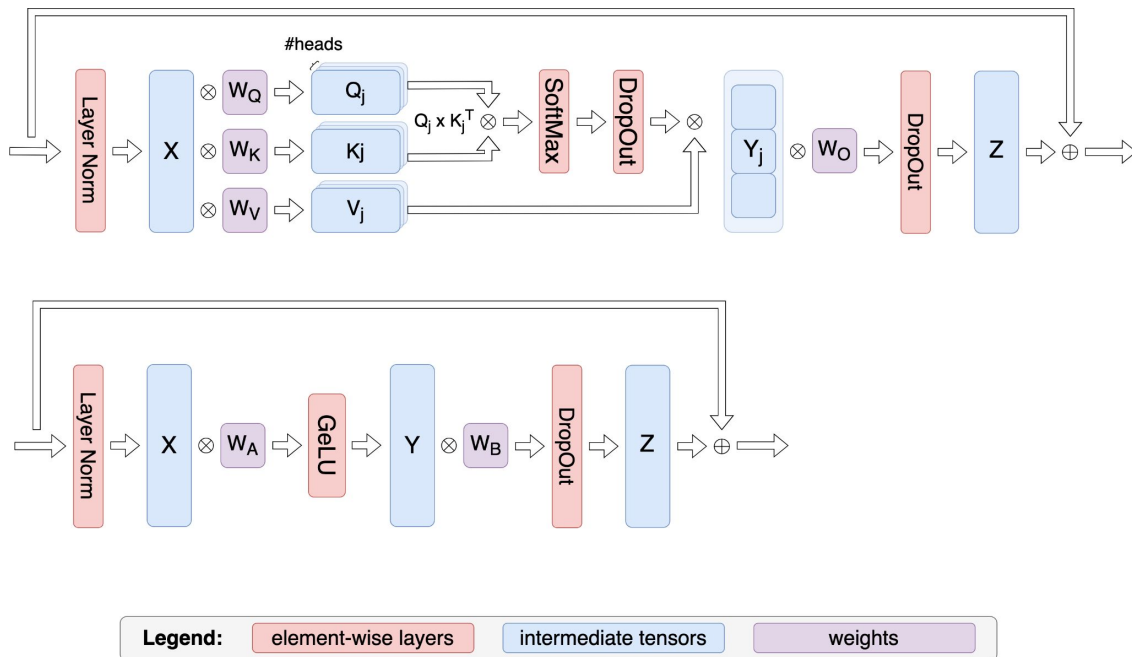
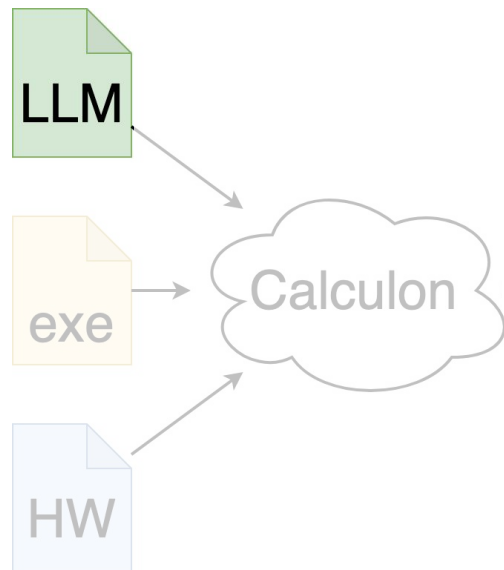
Result – profile-like detailed timing and utilization, *ultra-fast* space *exploration*



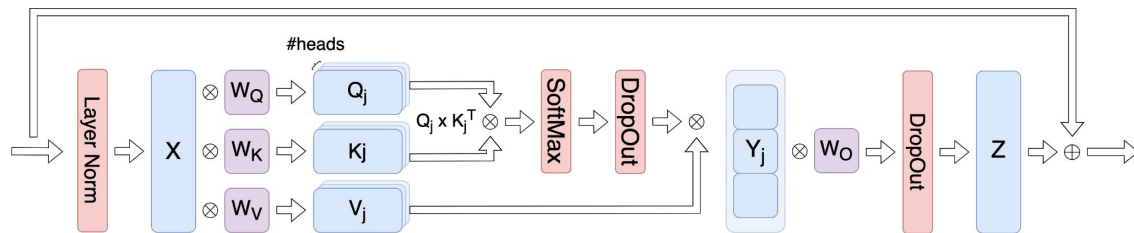
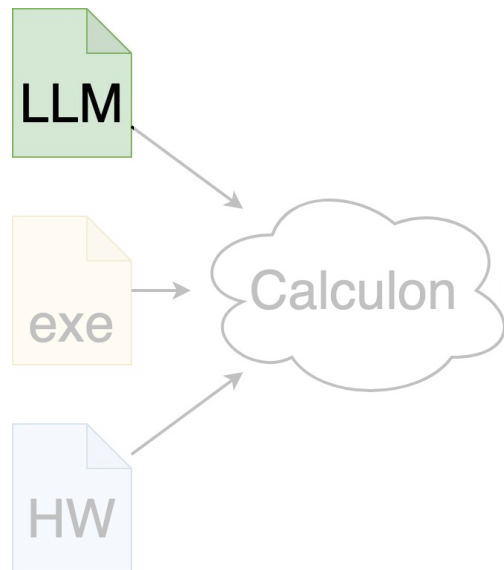
Inputs: LLM description



Inputs: LLM description



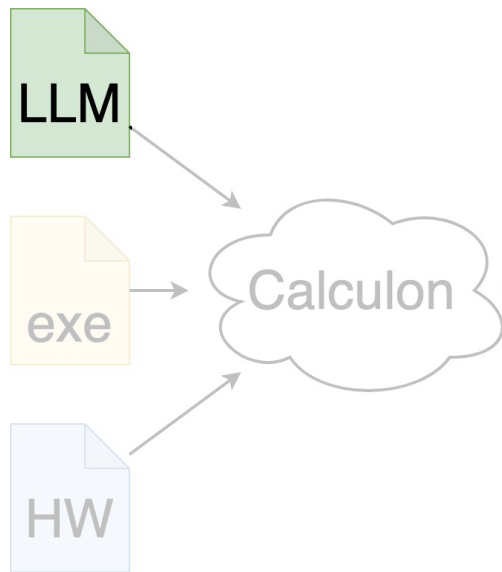
Inputs: LLM description



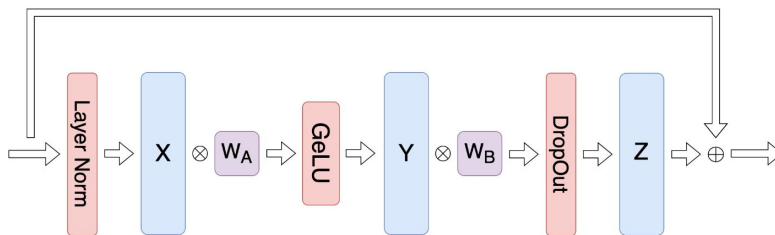
Multi-Head Attention (MHA) block

Legend: element-wise layers intermediate tensors weights

Inputs: LLM description



Multi-Layer Perceptron (MLP) block



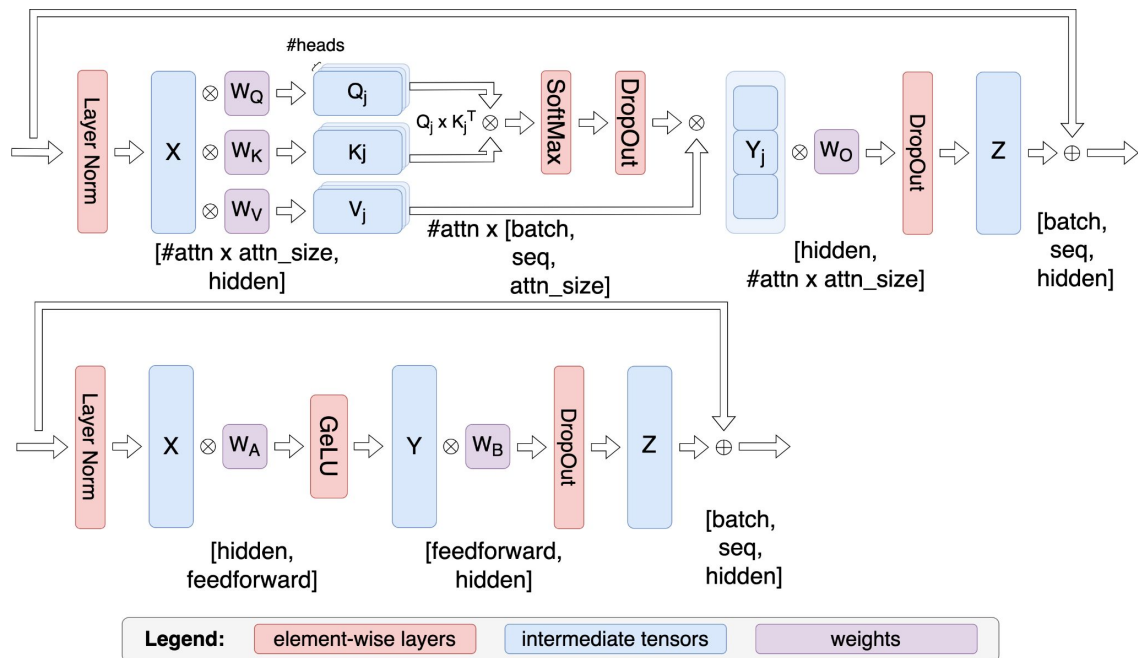
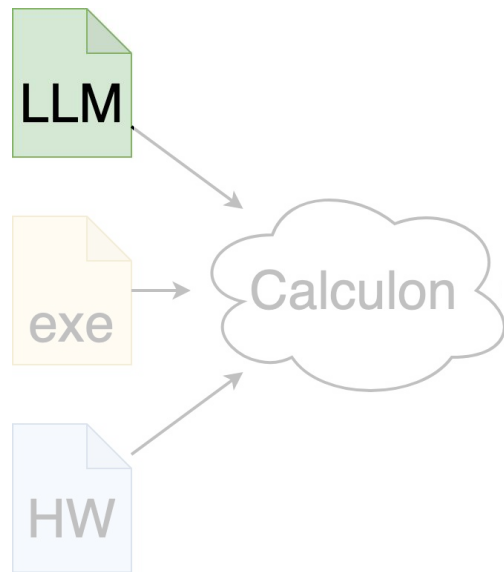
Legend:

element-wise layers

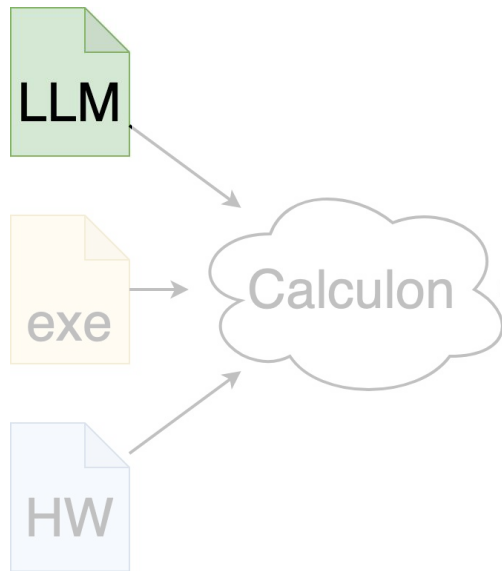
intermediate tensors

weights

Inputs: LLM description



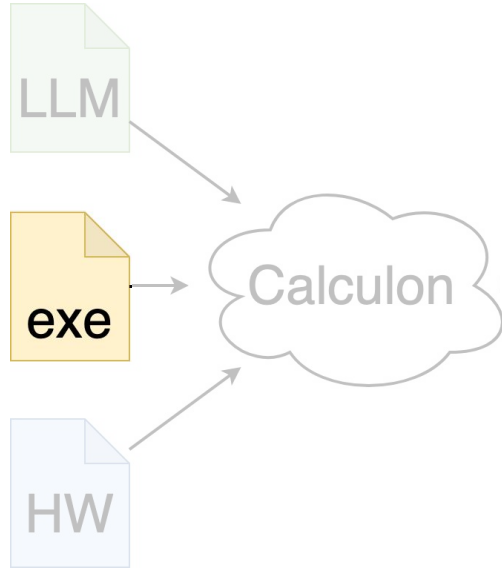
Inputs: LLM description



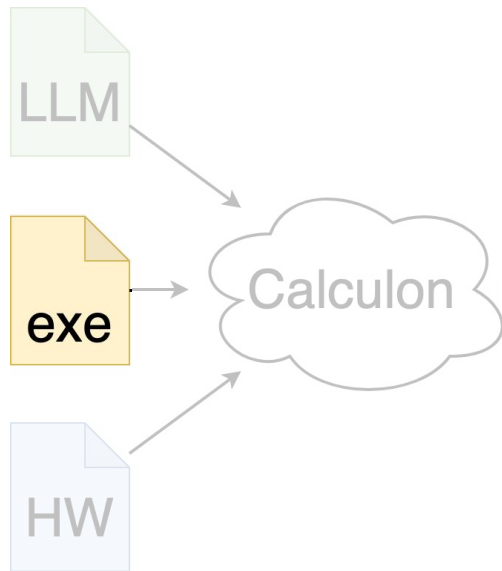
GPT3-175B

```
{  
  "hidden": 12288,  
  "feedforward": 49152,  
  "seq_size": 2048,  
  "attn_heads": 96,  
  "attn_size": 128,  
  "num_blocks": 96  
}
```

Inputs: execution strategy

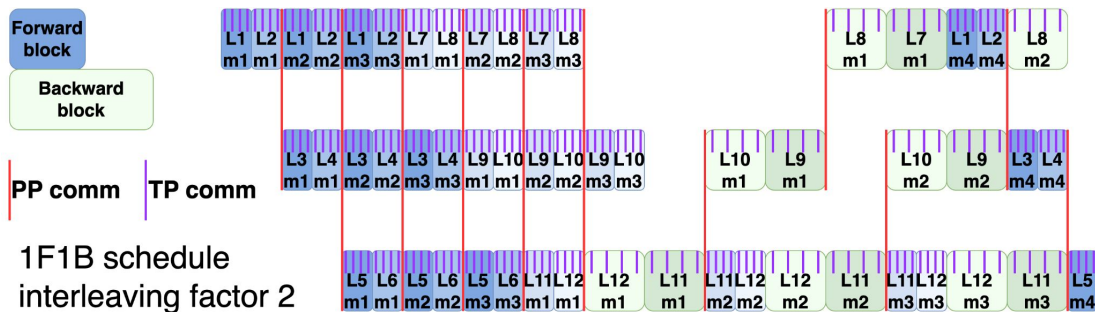
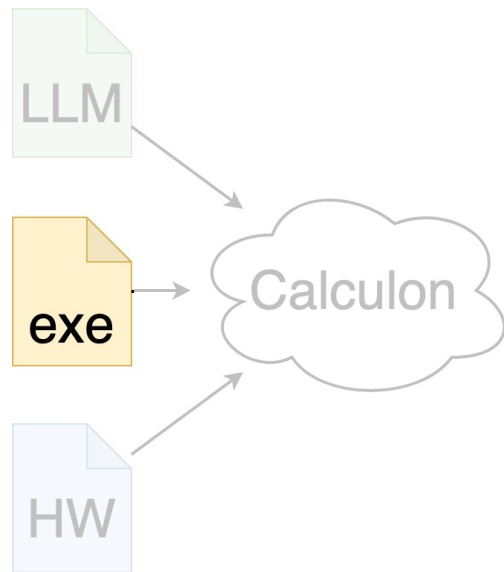


Inputs: execution strategy



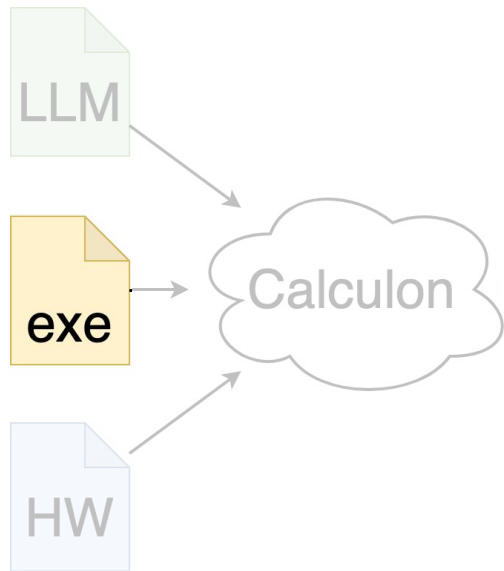
Optimization	Year	Related system	Comp time	Comp util	Mem time	Mem cap	Mem BW	Net time	Net BW	Range
Data parallelism (DP) [55]	1989	network	–	↑	–	↑↑↑	–	↑	↑	1 .. batch
DP overlap [23]	2017	network	↑	↓	–	–	–	↓↓↓	–	true/false
Optimizer sharding [22]	2019	network	↓	–	–	↓↓	–	–	–	true/false
Recompute [5, 10]	2000	compute	↑↑	–	–	↓↓↓	–	–	–	full/attn/none
Fused layers [26]	2018	compute	–	↑↑	↓↓	↓↓	↓	–	–	true/false
Microbatch training [13]	2019	compute	–	↑↑	–	↑↑↑	–	–	–	1 .. batch/DP
Pipeline parallelism (PP) [7, 13]	2012	network	↑	↓↓	–	↓↓	–	↑	↑	1 .. blocks
PP 1F1B schedule [7, 28]	2012	network	–	–	–	↓↓	–	–	–	true/false
PP interleaving [29]	2021	network	↓	↑↑	–	↑	–	↑	↑↑	1 .. blocks/PP
PP RS + AG [20]	2022	network	–	–	–	–	–	↓	↓↓	true/false
Tensor parallelism (TP) [7, 21, 43]	2012	network	↓↓	↓	–	↓↓	↓↓	↑↑↑	↑↑↑	1 .. attn
TP RS + AG instead AR [29]	2021	network	–	–	↑	↑	–	↓	↓	true/false
Sequence parallelism (SP) [20]	2022	network	↓	–	↓	↓↓	↓	↑	↑	true/false
TP redo for SP [20]	2022	network	–	–	–	↓	–	↑	↑	true/false
TP overlap [52]	2022	network	↑	↓	–	–	–	↓↓	–	none/pipe/ring
Weight offload [42]	2021	memory	–	–	↑	↓↓↓	↑	–	–	true/false
Activation offload [42]	2021	memory	–	–	↑	↓↓↓	↑	–	–	true/false
Optimizer offload [42]	2021	memory	–	–	↑	↓	↑	–	–	true/false

Inputs: execution strategy

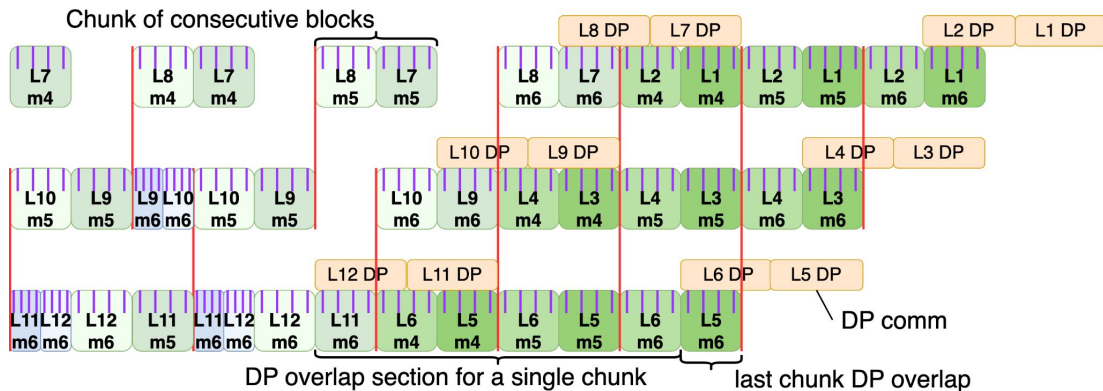


Forward pass

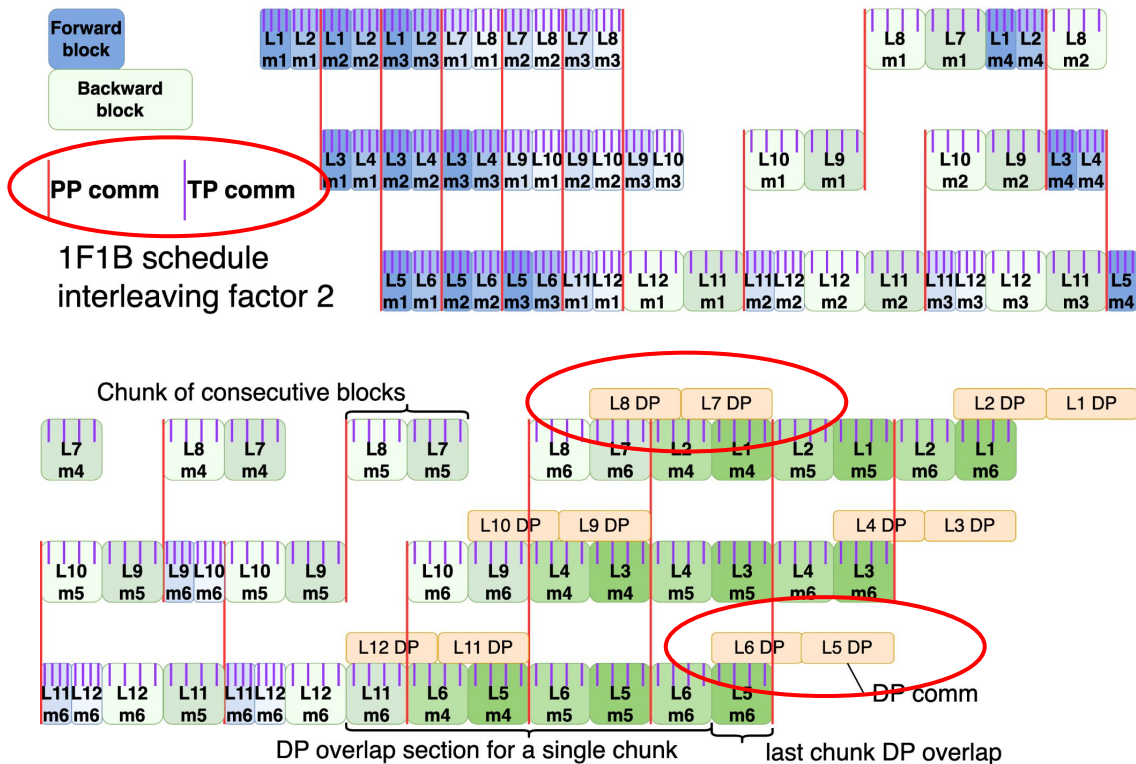
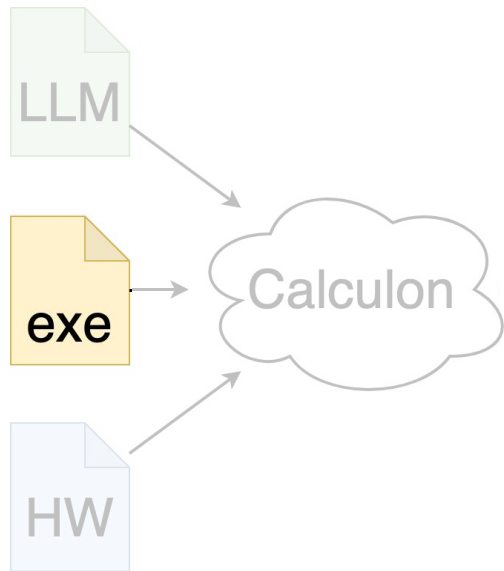
Inputs: execution strategy



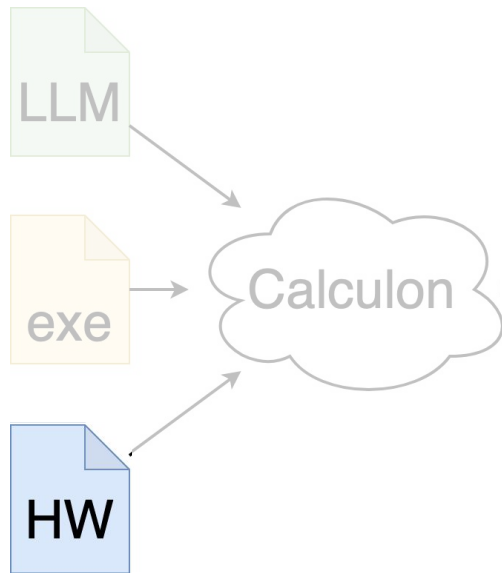
Backward pass



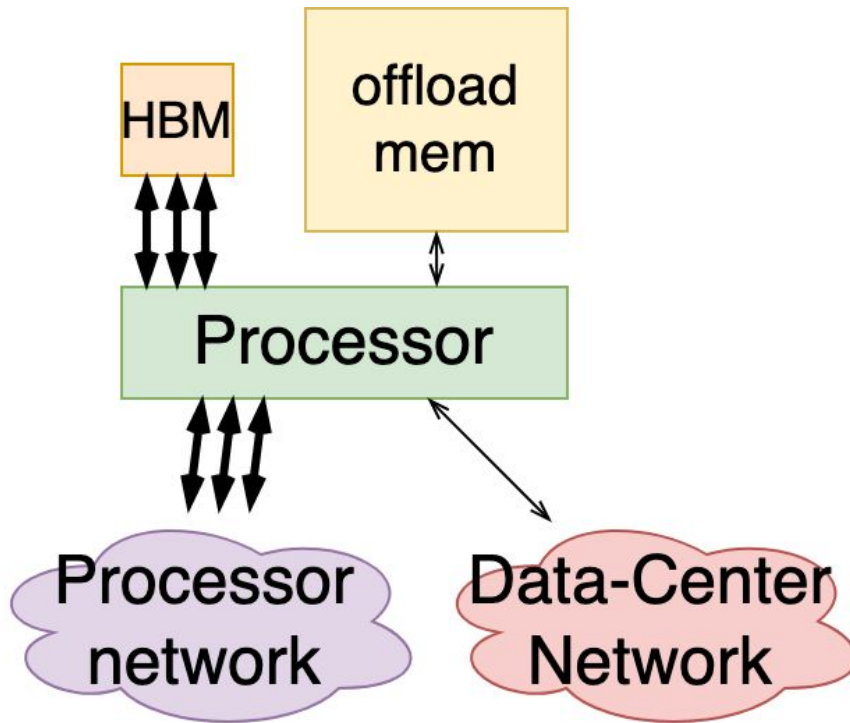
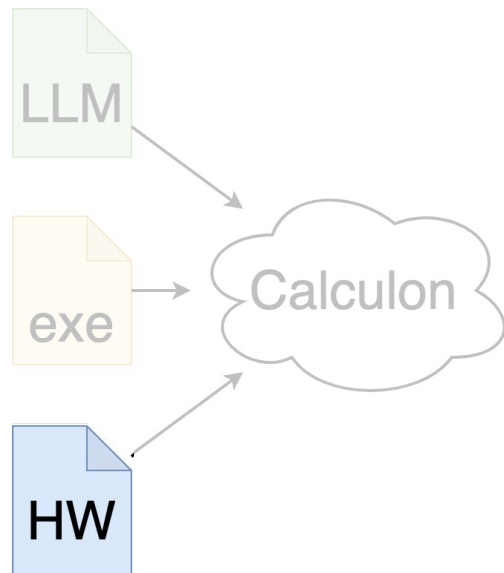
Inputs: execution strategy



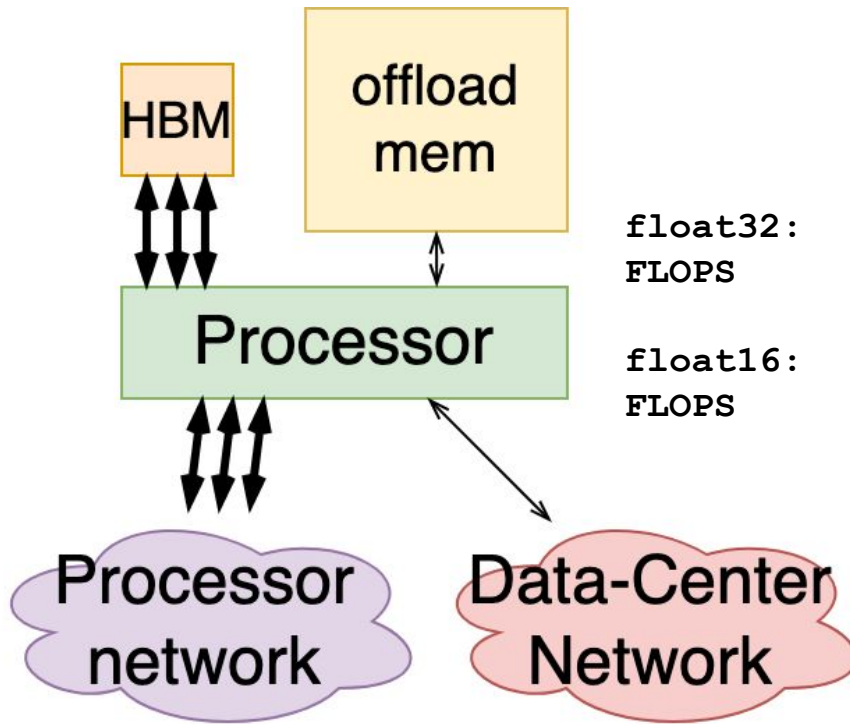
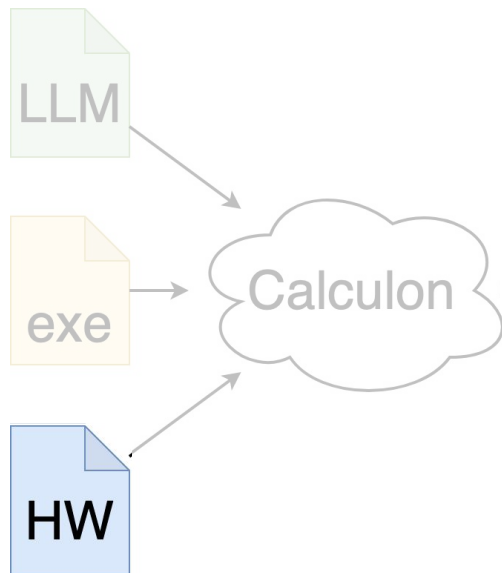
Inputs: hardware



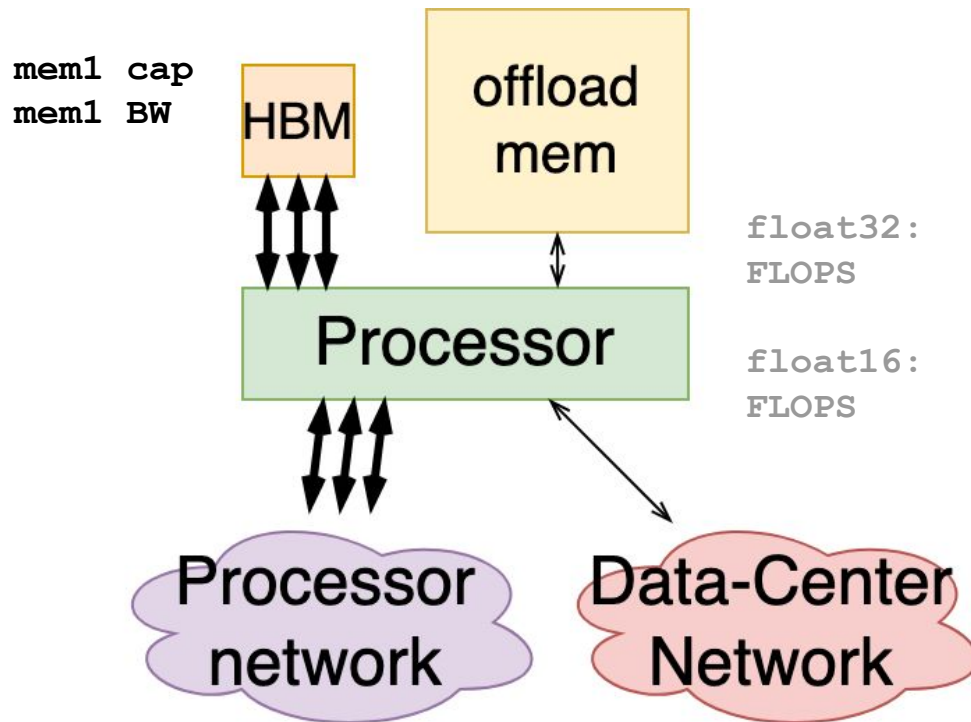
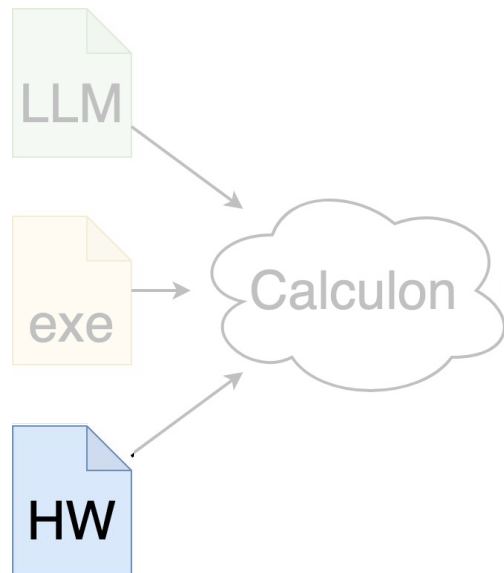
Inputs: hardware



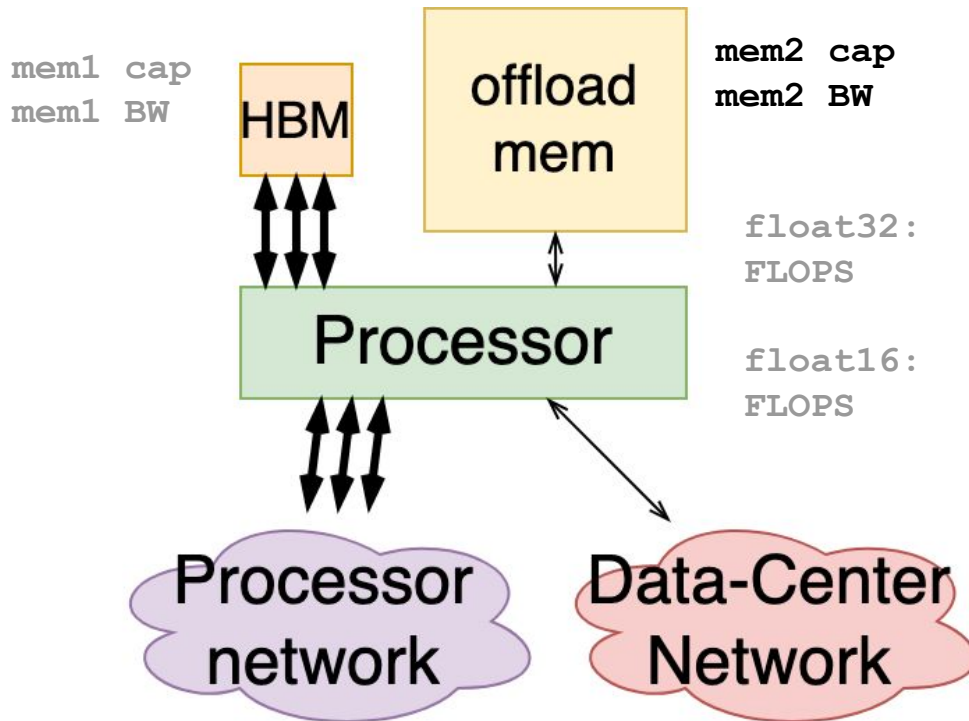
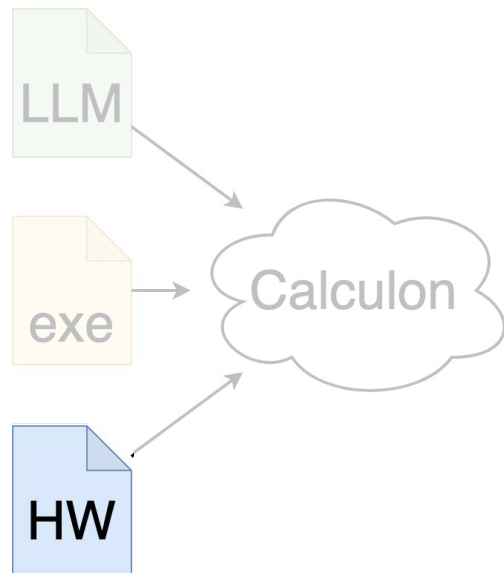
Inputs: hardware



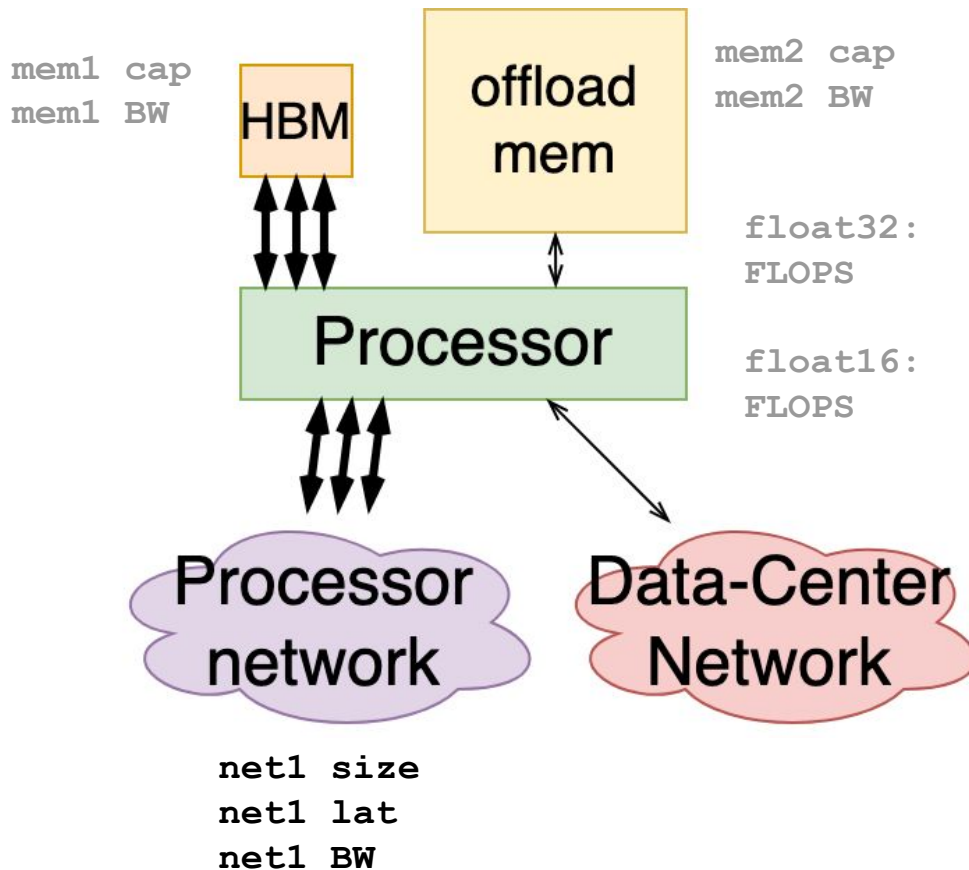
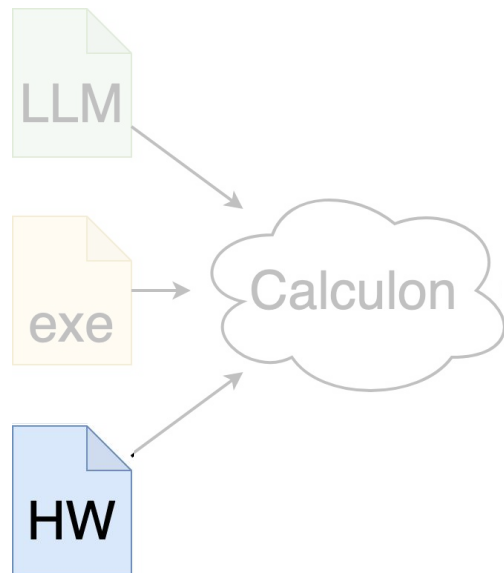
Inputs: hardware



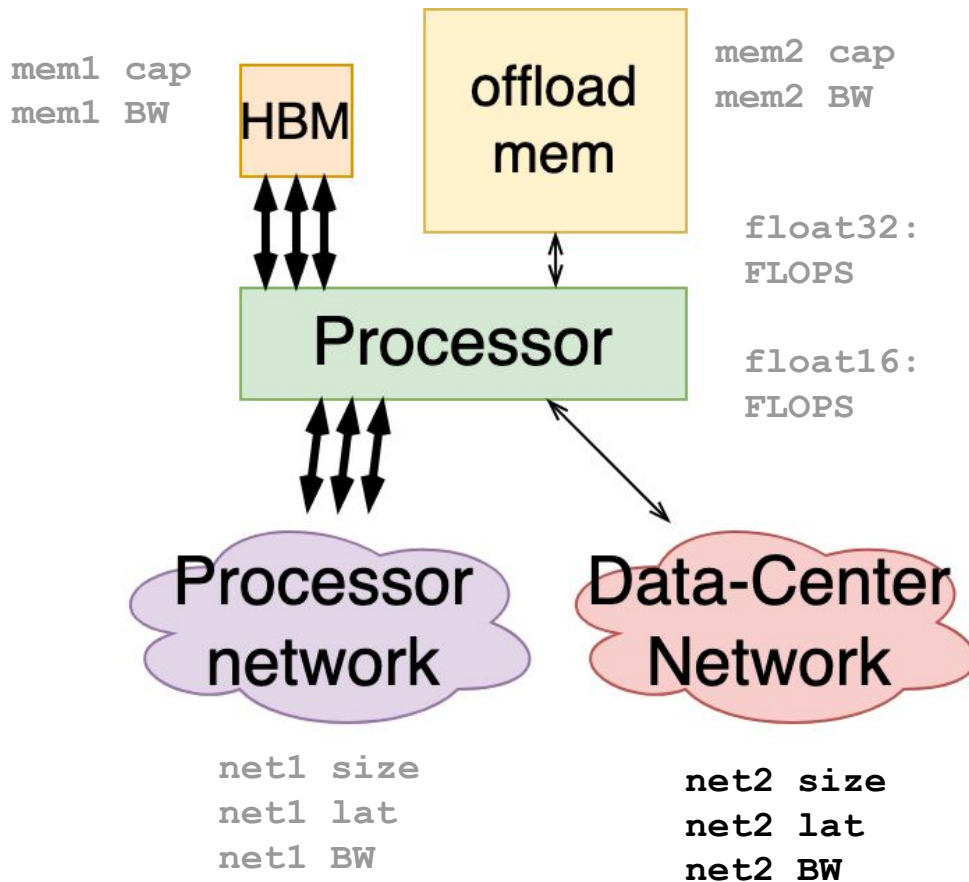
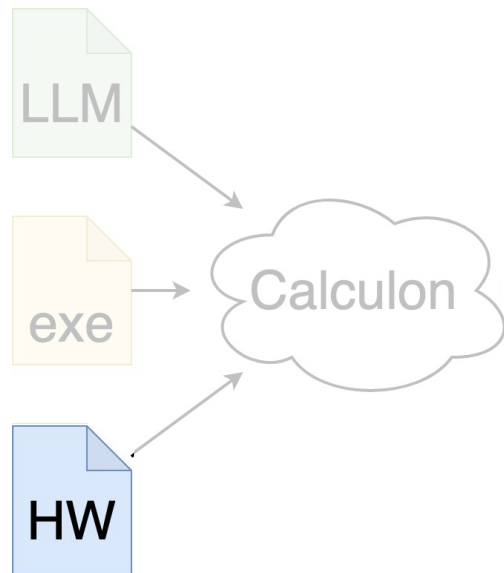
Inputs: hardware



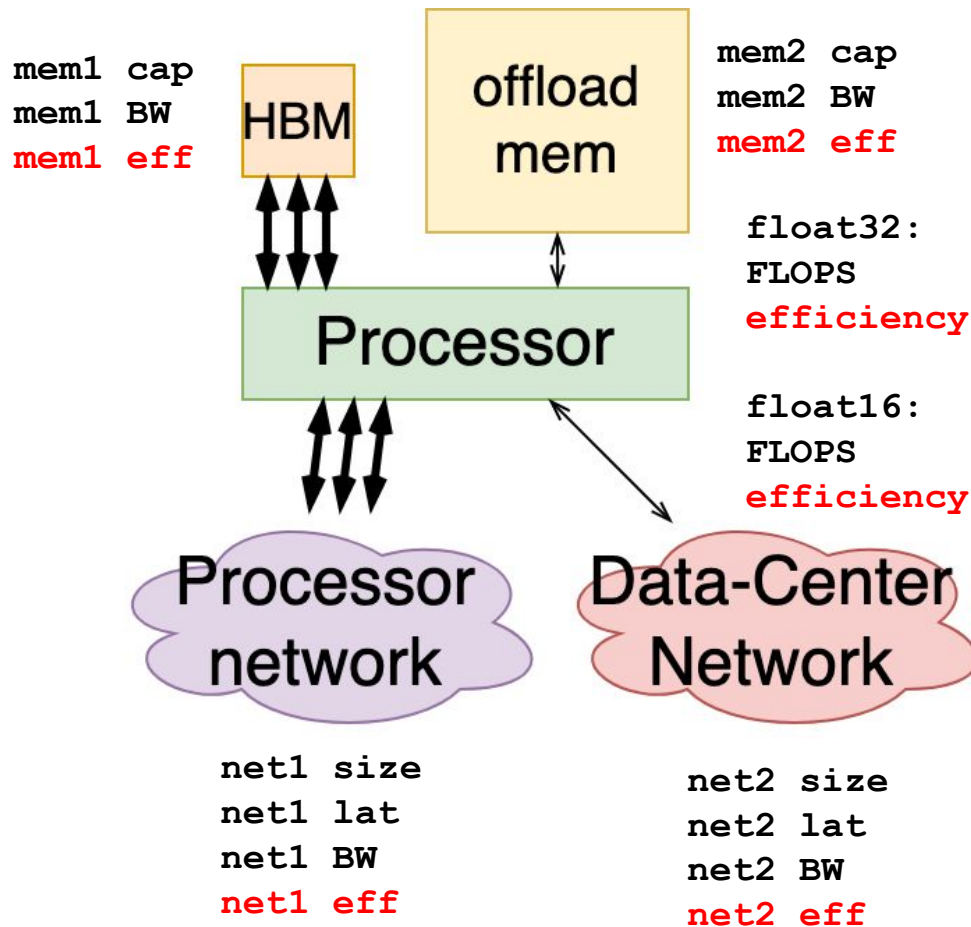
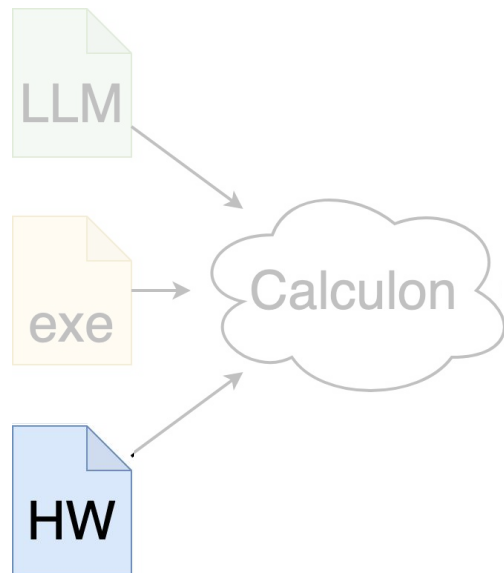
Inputs: hardware



Inputs: hardware



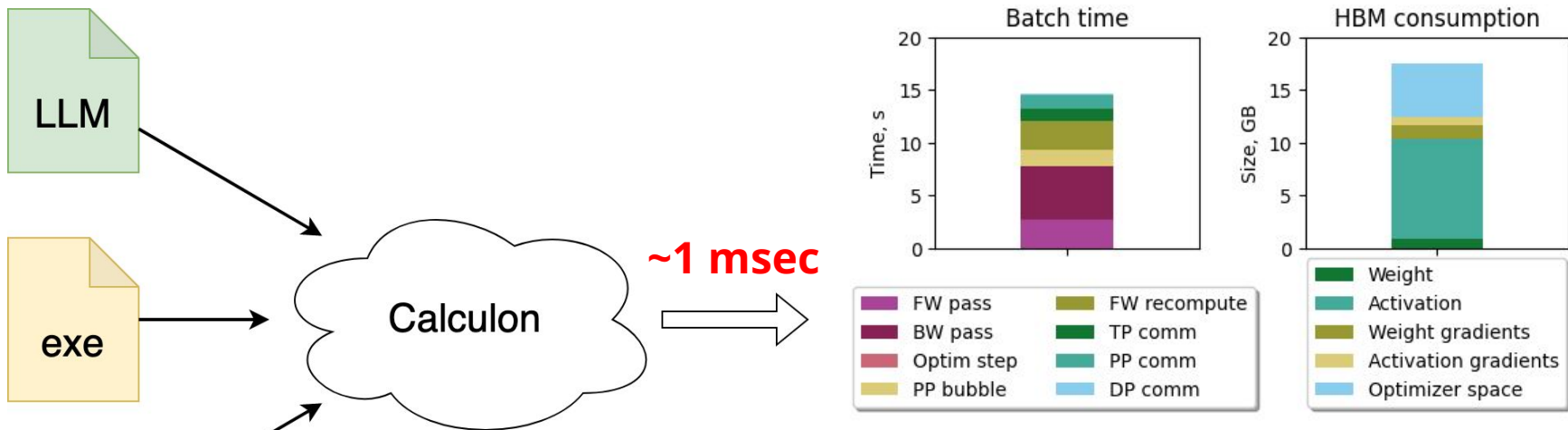
Inputs: hardware



		22B	175B	530B	1T
<u>Full</u>	Selene	1.42	18.13	49.05	94.42
	Calculon	1.43	18.30	50.46	91.70
	Delta	-0.40%	-0.94%	-2.88%	-2.88%
<u>Seq+Sel</u>	Selene	1.10	13.75	37.83	71.49
	Calculon	1.17	13.92	35.09	67.74
	Delta	-6.36%	-1.24%	7.25%	5.24%

Validation against run from *Vijay Korthikanti et al. "Reducing activation recomputation in large transformer models." Proceedings of Machine Learning and Systems 5 (2023).*

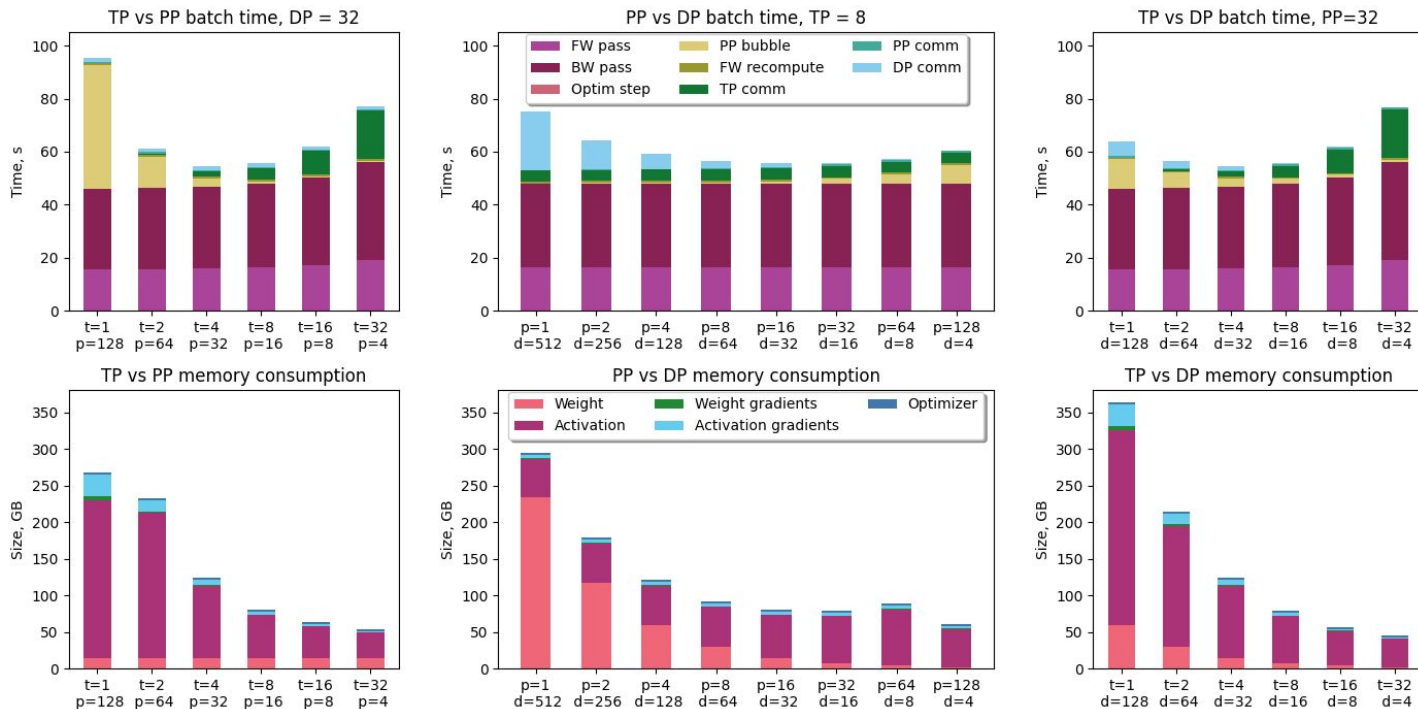
Modeling with Calculon



- Detailed time and memory allocation report
- Balancing various optimizations and parallelism modes
- Ultra-fast exploitation for accurate SW/HW co-design

Study 1: parallelism trade-offs

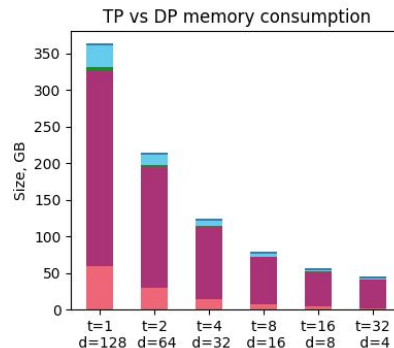
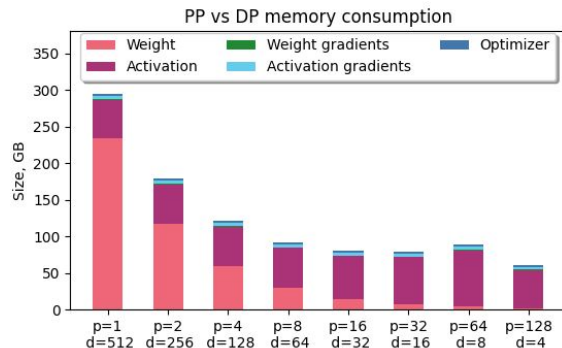
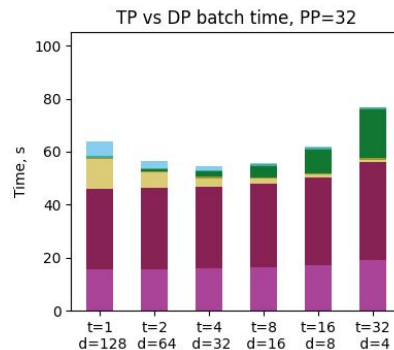
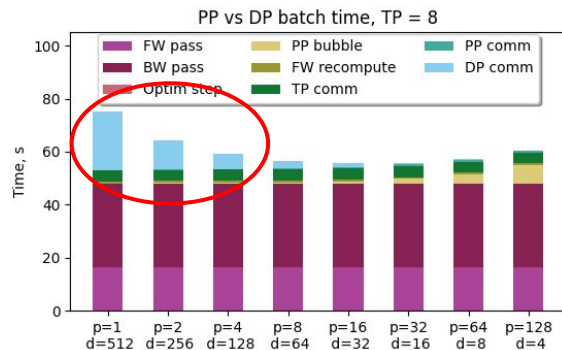
Megatron-1T single batch training on 4096 GPUs with various parallelism strategies



Study 1: data parallelism (DP) trade-offs

Megatron-1T single batch training on 4096 GPUs with various parallelism strategies

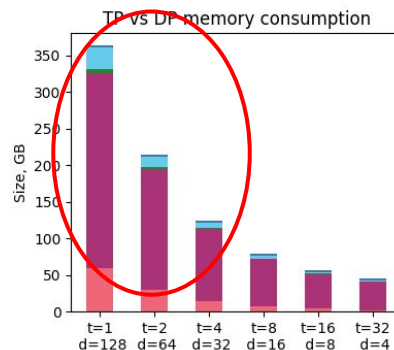
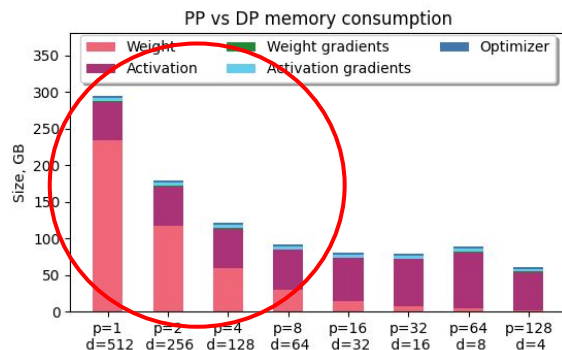
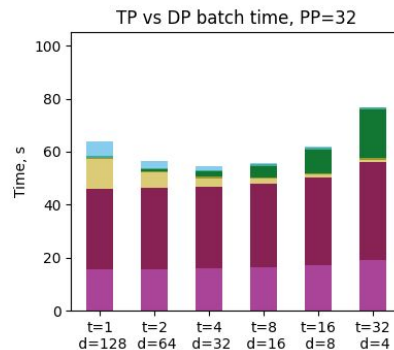
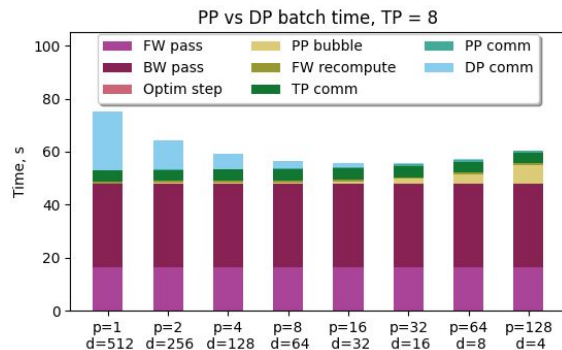
1. Significant comm. time



Study 1: data parallelism (DP) trade-offs

Megatron-1T single batch training on 4096 GPUs with various parallelism strategies

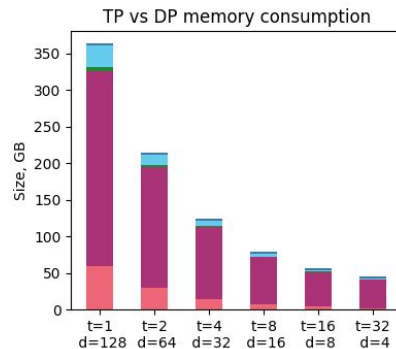
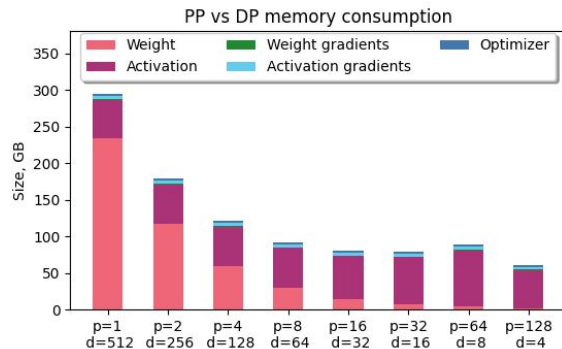
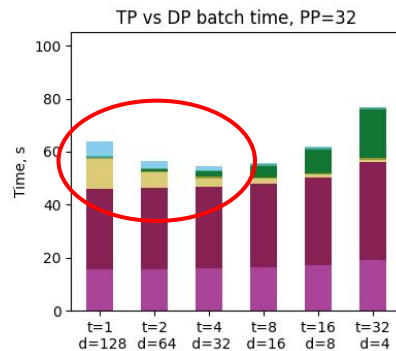
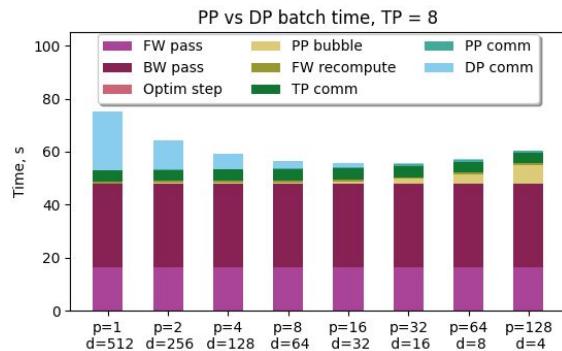
1. Significant comm. time
2. Enormous memory consumption



Study 1: data parallelism (DP) trade-offs

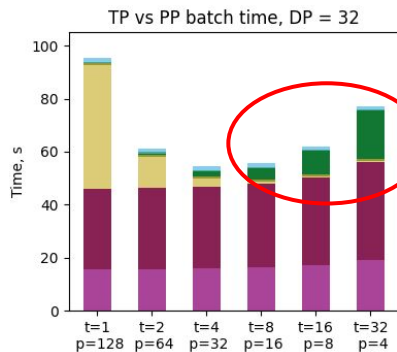
Megatron-1T single batch training on 4096 GPUs with various parallelism strategies

1. Significant comm. time
2. Enormous memory consumption
3. Inflating pipeline parallelism bubble

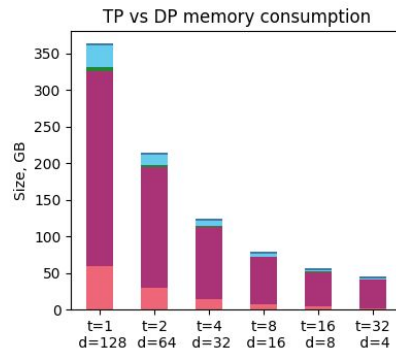
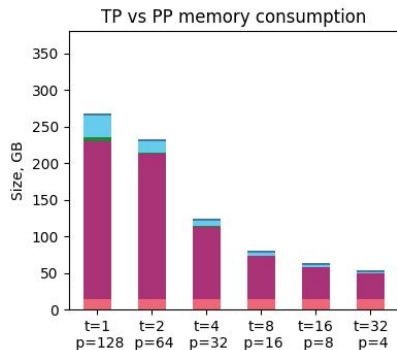
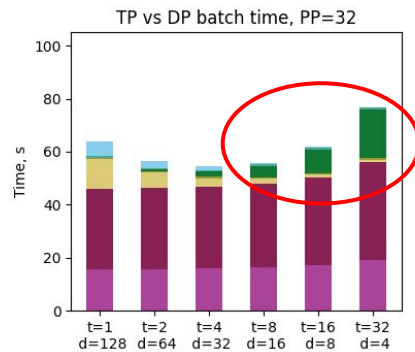


Study 1: tensor parallelism (TP) trade-offs

Megatron-1T single batch training on 4096 GPUs with various parallelism strategies

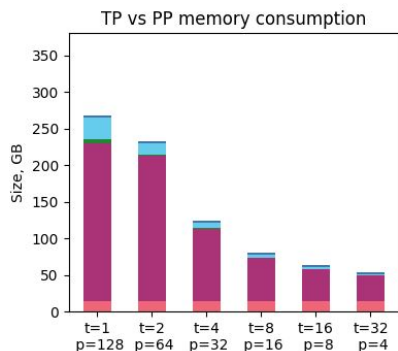
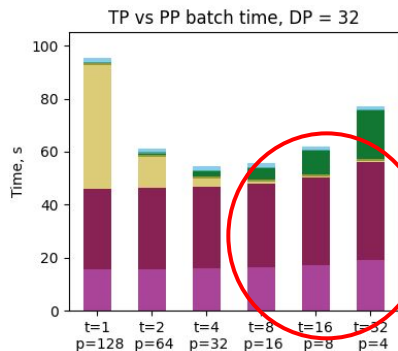


1. Communication on a critical path

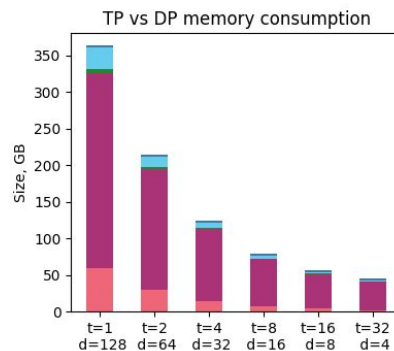
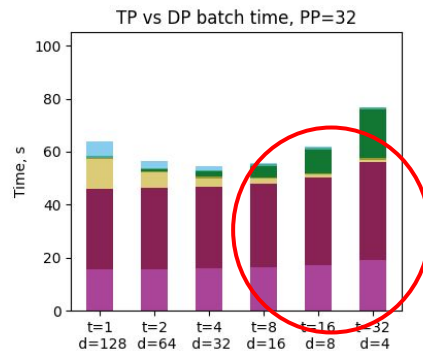


Study 1: tensor parallelism (TP) trade-offs

Megatron-1T single batch training on 4096 GPUs with various parallelism strategies

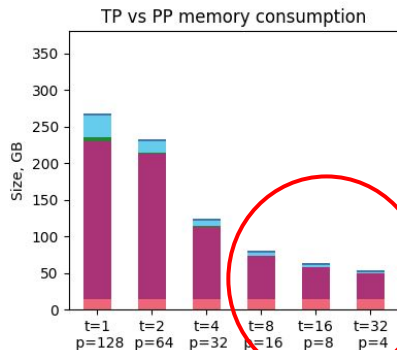
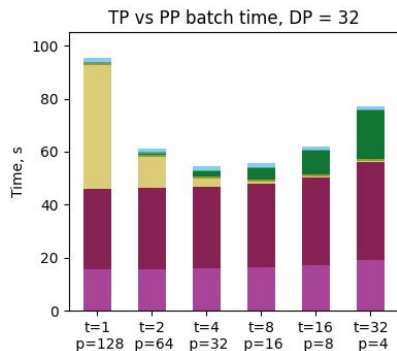


1. Communication on a critical path
2. Lower matrix multiplication efficiency

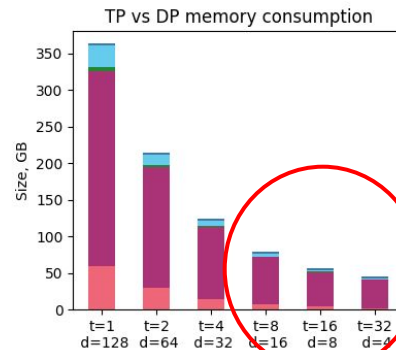
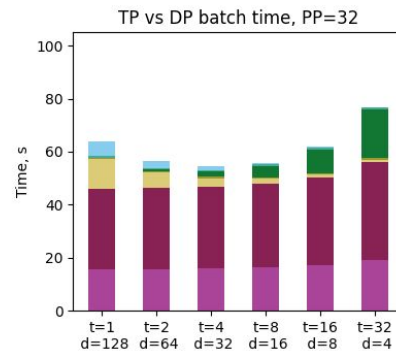


Study 1: tensor parallelism (TP) trade-offs

Megatron-1T single batch training on 4096 GPUs with various parallelism strategies

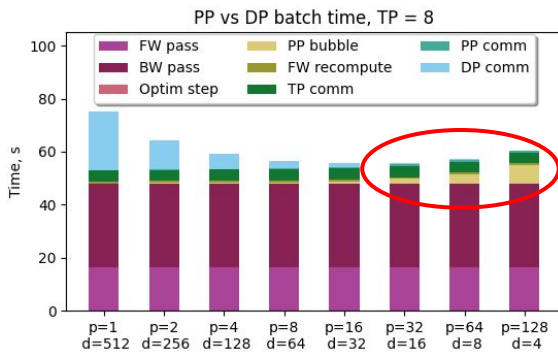
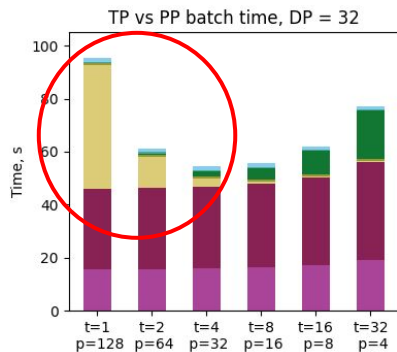


1. Communication on a critical path
2. Lower matrix multiplication efficiency
3. Best for memory consumption reduction

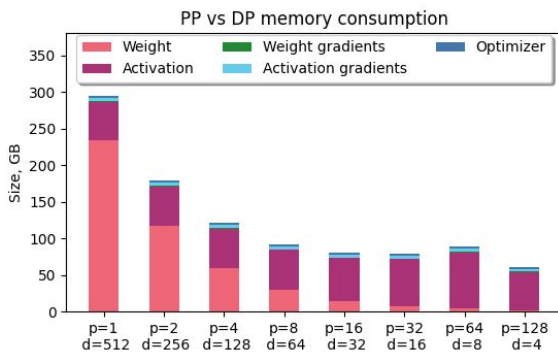
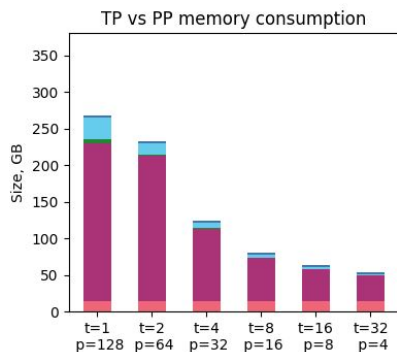


Study 1: pipeline parallelism (PP) trade-offs

Megatron-1T single batch training on 4096 GPUs with various parallelism strategies

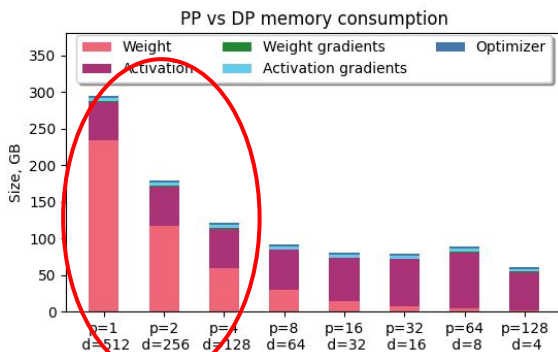
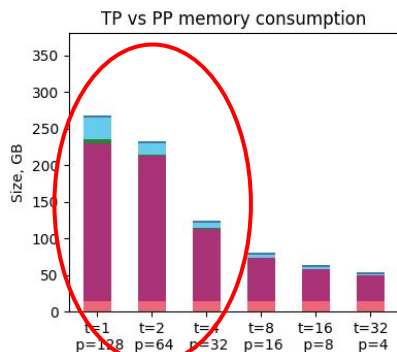
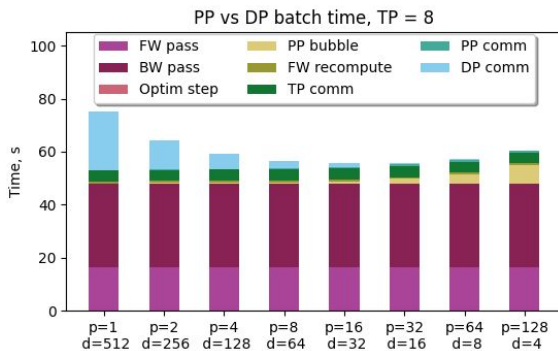
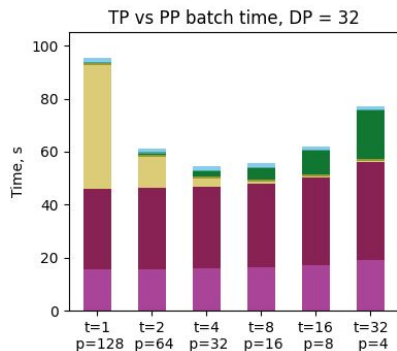


1. Pipeline bubble inefficiency



Study 1: pipeline parallelism (PP) trade-offs

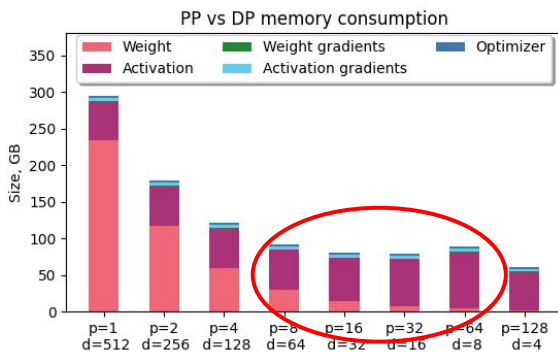
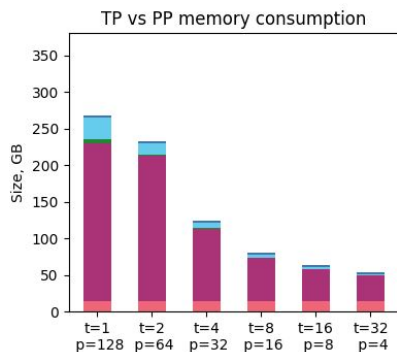
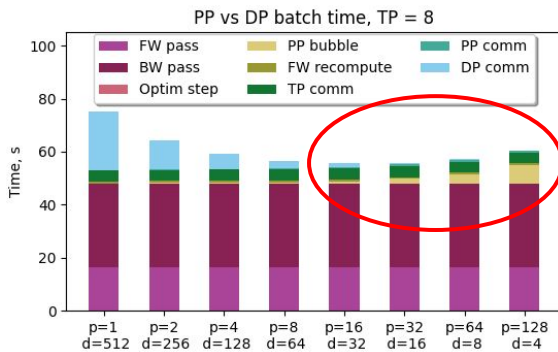
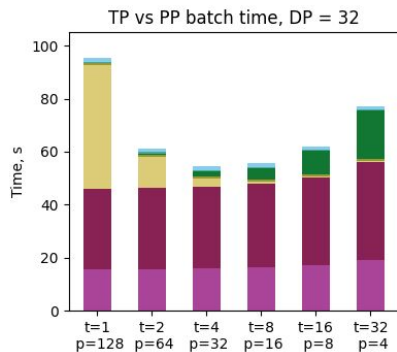
Megatron-1T single batch training on 4096 GPUs with various parallelism strategies



1. Pipeline bubble inefficiency
2. Much less efficient than TP in memory savings

Study 1: pipeline parallelism (PP) trade-offs

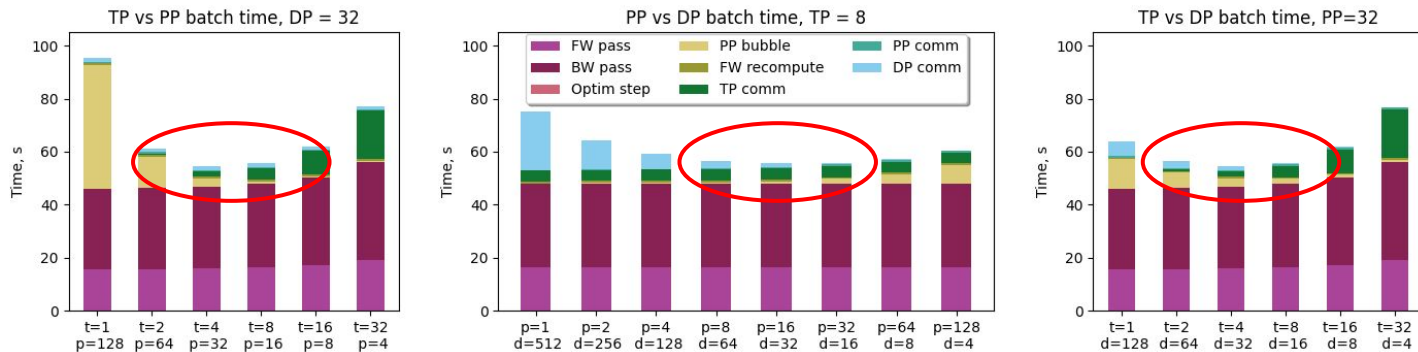
Megatron-1T single batch training on 4096 GPUs with various parallelism strategies



1. Pipeline bubble inefficiency
2. Much less efficient than TP in memory savings
3. Scales even worse than DP

Study 1: parallelism trade-offs

Megatron-1T single batch training on 4096 GPUs with various parallelism strategies



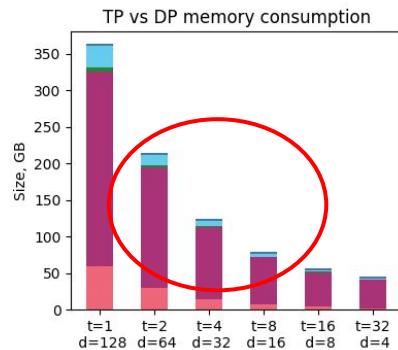
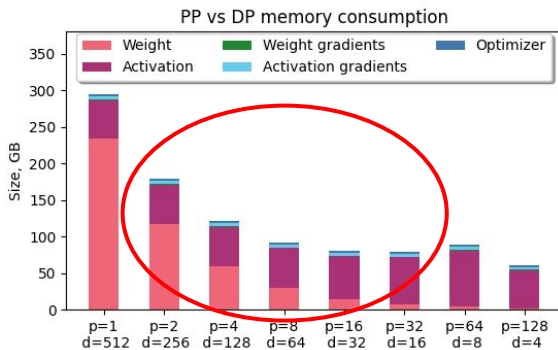
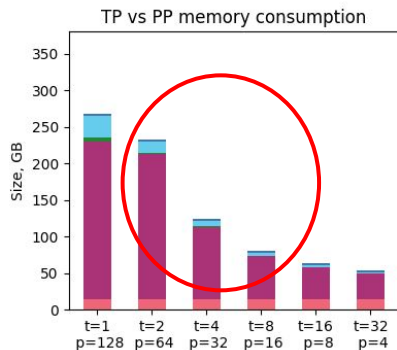
Each parallelism comes with **trade offs** (not only network), but there exist an **optimal parallelization split**.

We need **O(100GB/s)** for TP, **O(10GB/s)** for PP and DP, all communication **independent**, mapped to **3D-Torus** or **tapered FT**

Study 1: parallelism trade-offs

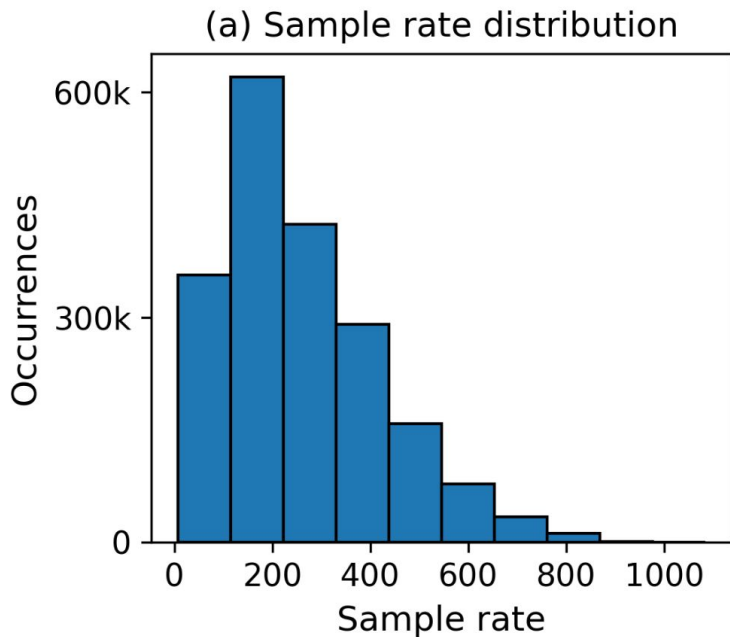
Most configurations are **limited by available memory**.

More memory can balance parallelization strategies with **better efficiency**.



Study 2: optimal execution strategy

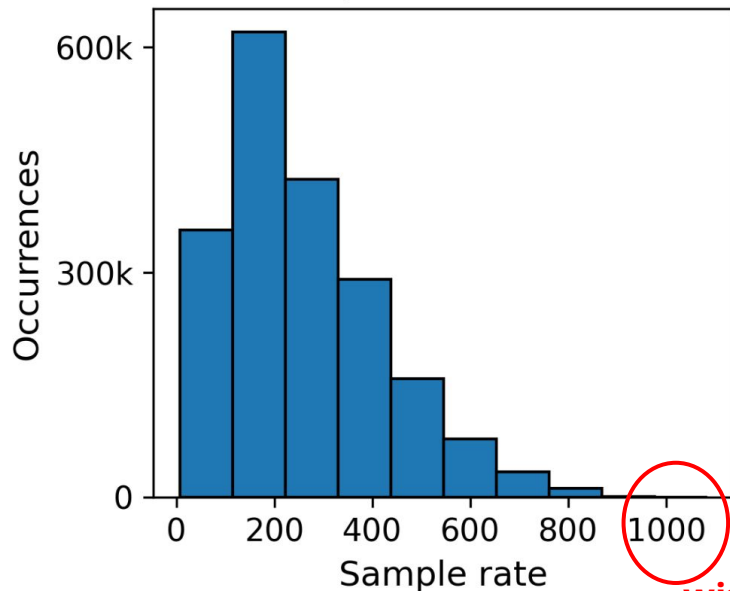
1,974,902 execution strategies
for GPT3 175B on 4096 GPUs



Study 2: optimal execution strategy

1,974,902 execution strategies
for GPT3 175B on 4096 GPUs

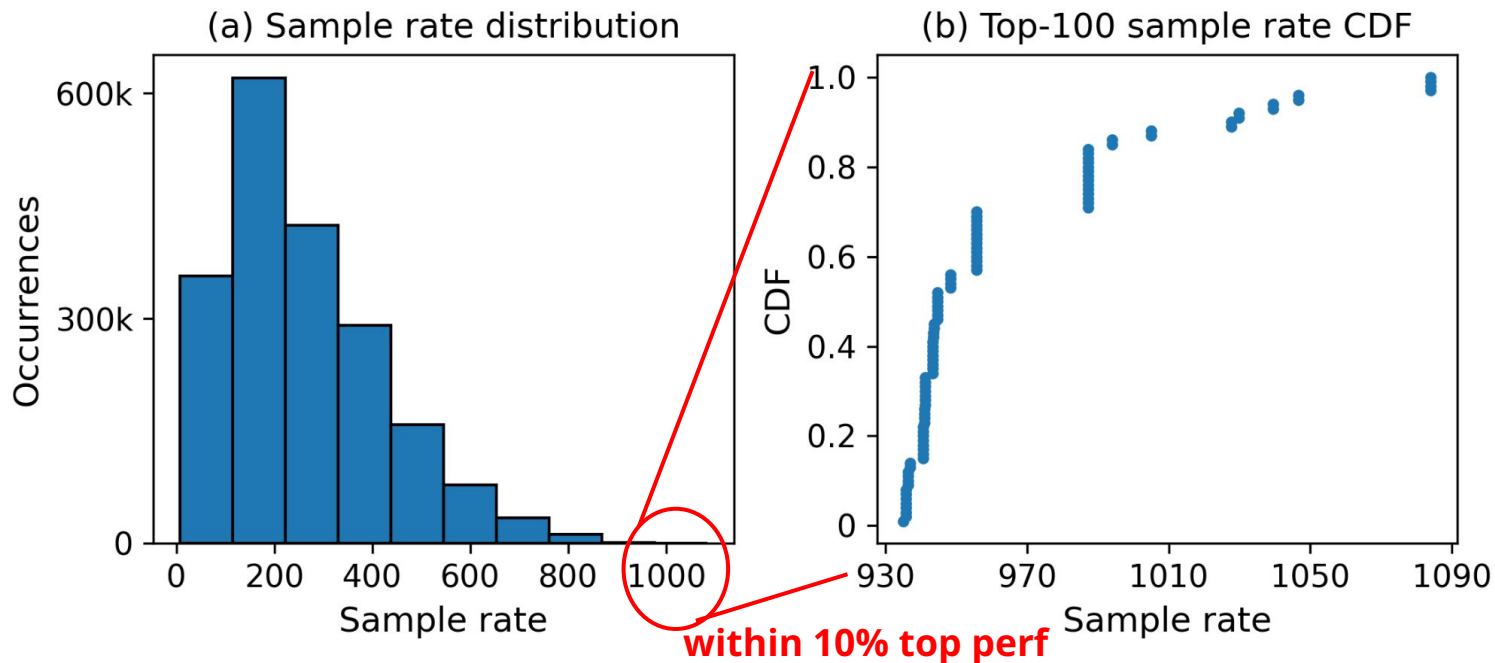
(a) Sample rate distribution



within 10% top perf

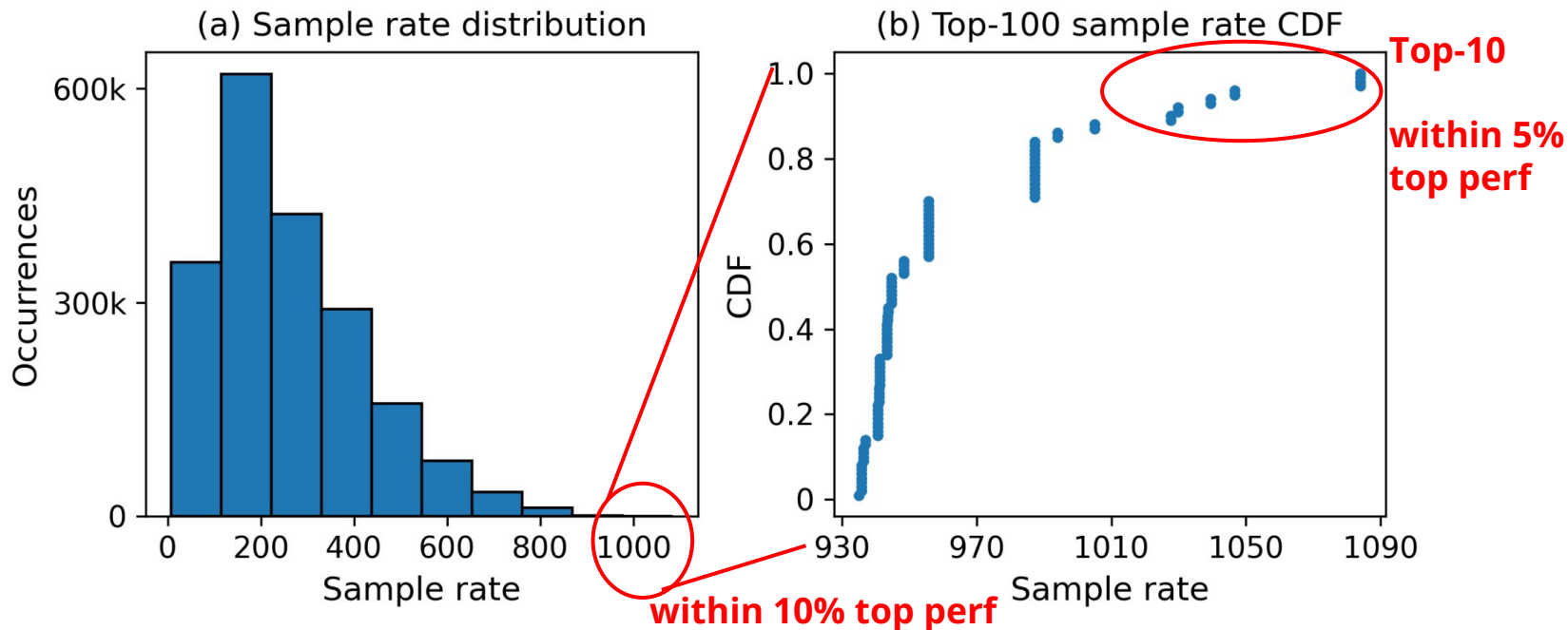
Study 2: optimal execution strategy

1,974,902 execution strategies
for GPT3 175B on 4096 GPUs



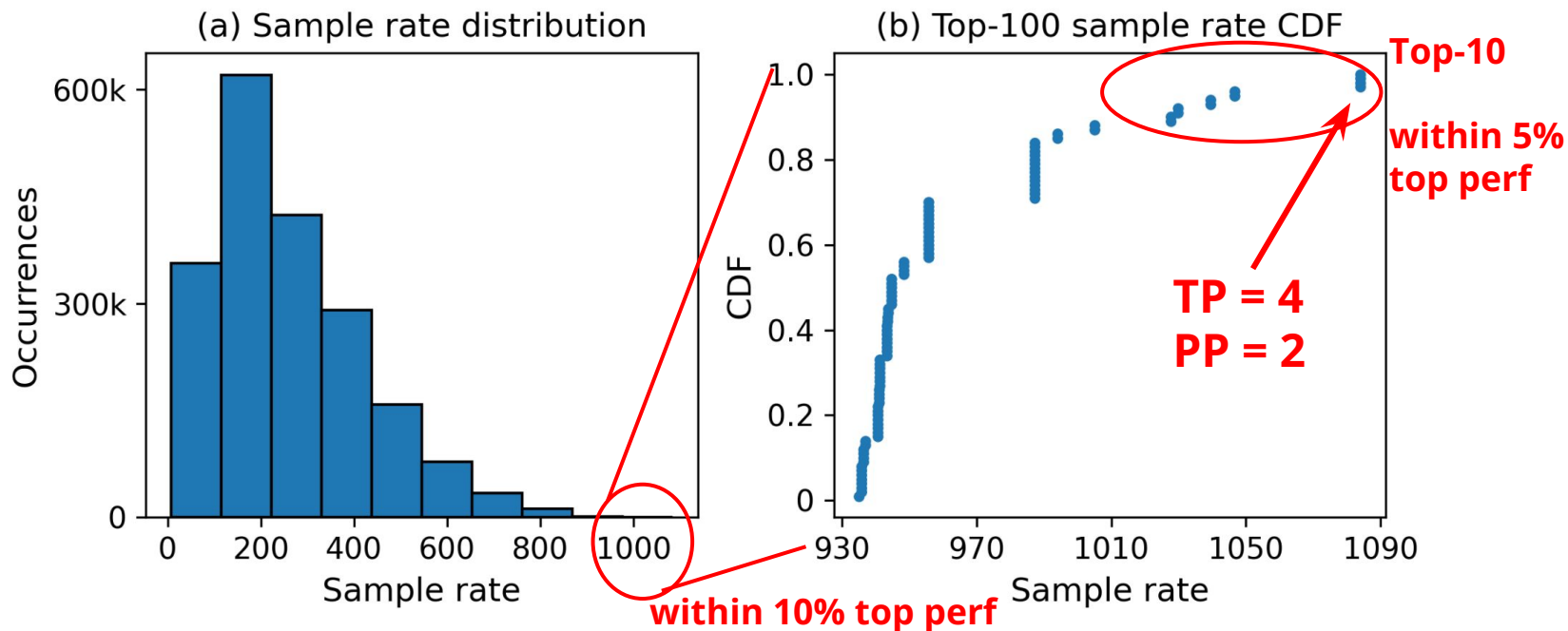
Study 2: optimal execution strategy

1,974,902 execution strategies
for GPT3 175B on 4096 GPUs



Study 2: optimal execution strategy

1,974,902 execution strategies
for GPT3 175B on 4096 GPUs

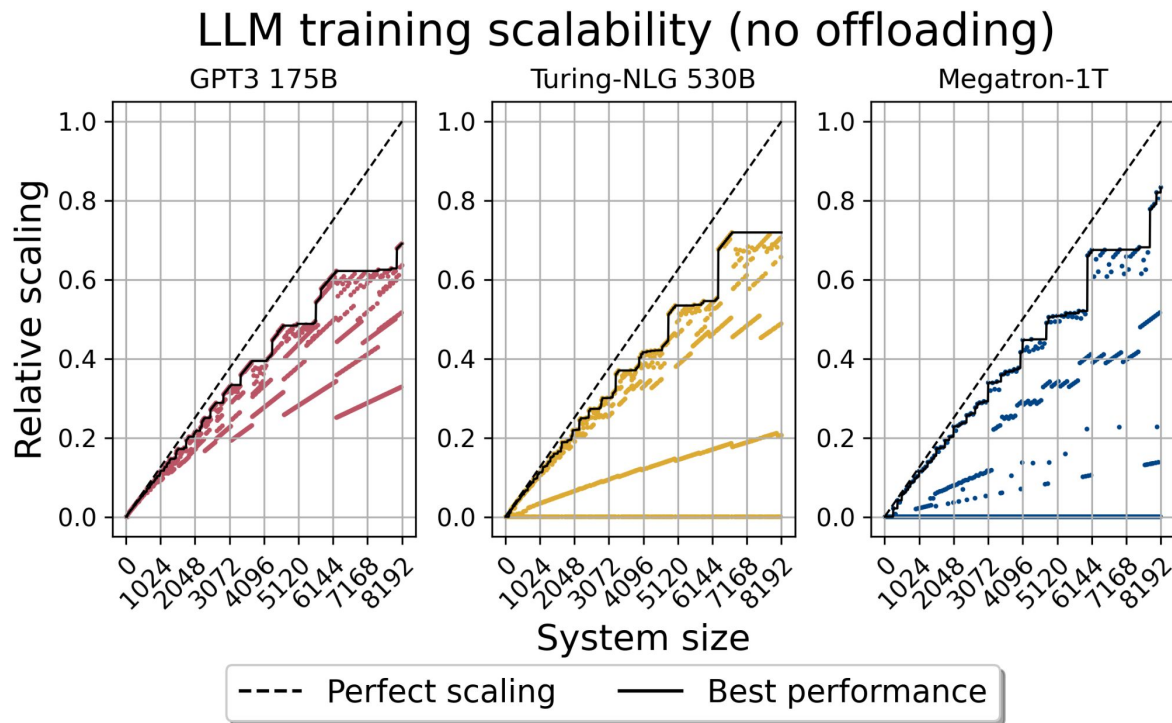


Study 3: optimal system size

System sizes with step 8

All processors are used

Each point is 10^5 - 10^7 experiments



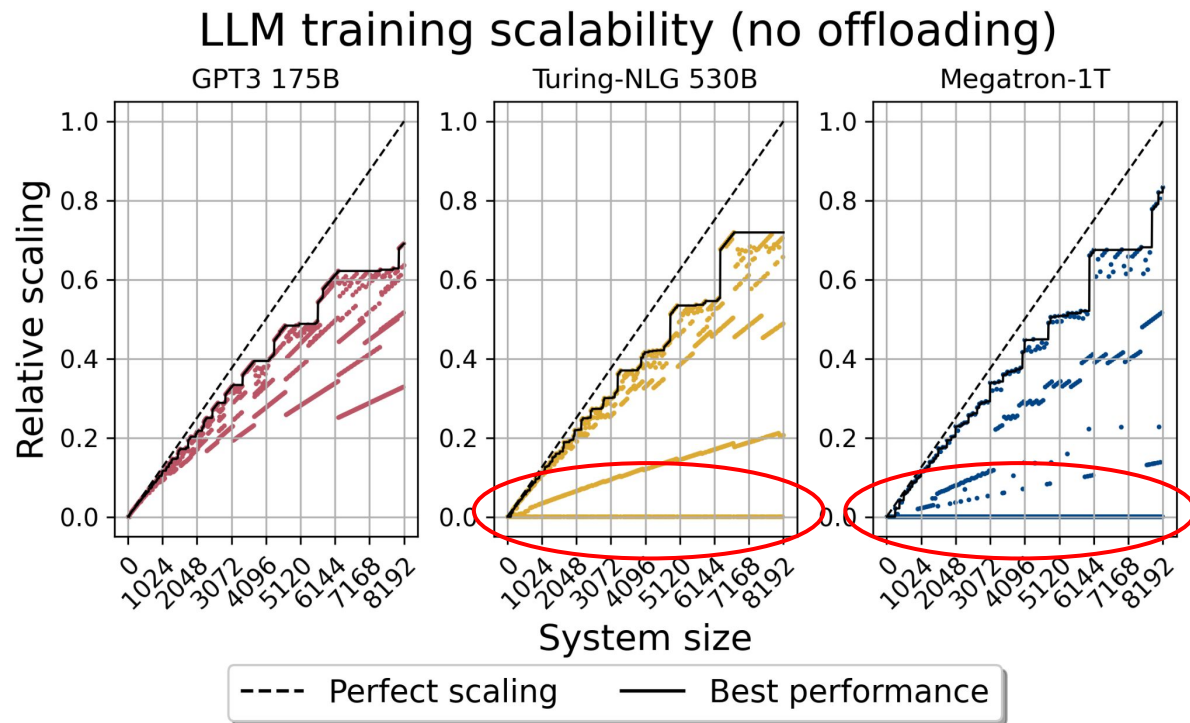
Study 3: optimal system size

System sizes with step 8

All processors are used

Each point is 10^5 - 10^7 experiments

Some configurations don't run due to mapping issues



Study 3: optimal system size

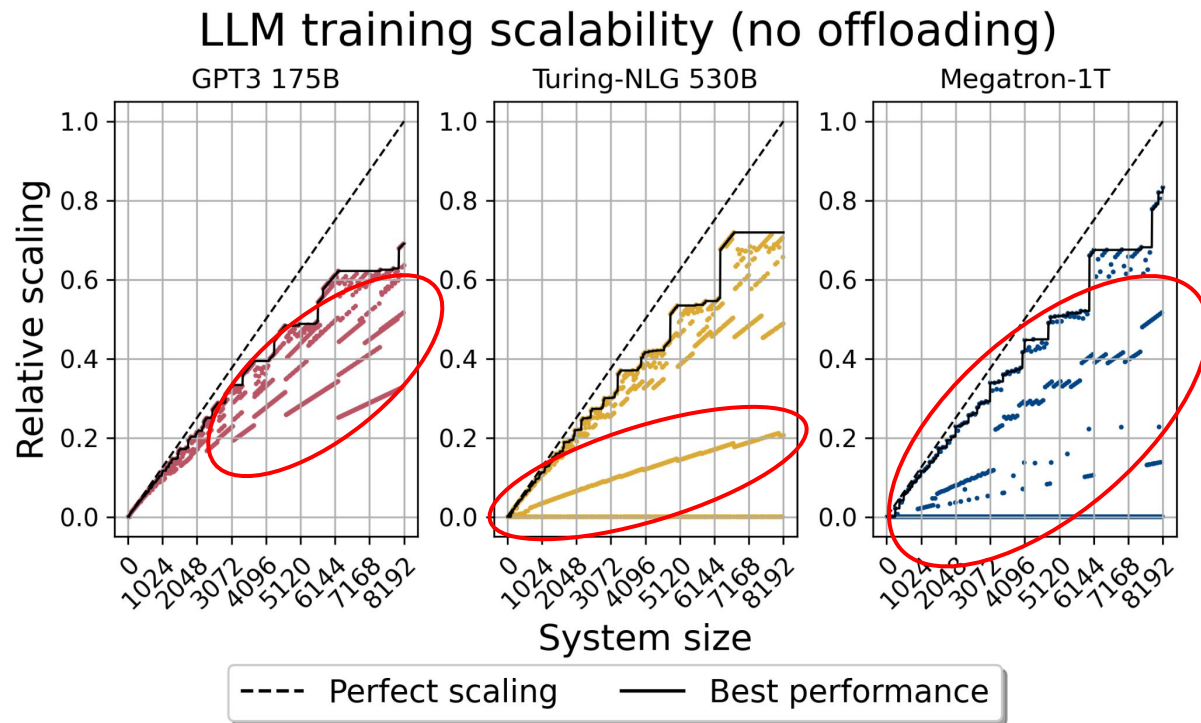
System sizes with step 8

All processors are used

Each point is 10^5 - 10^7 experiments

Some configurations don't run due to mapping issues

Other configurations have lower performance, unable to find good parallelization strategy



Study 3: optimal system size

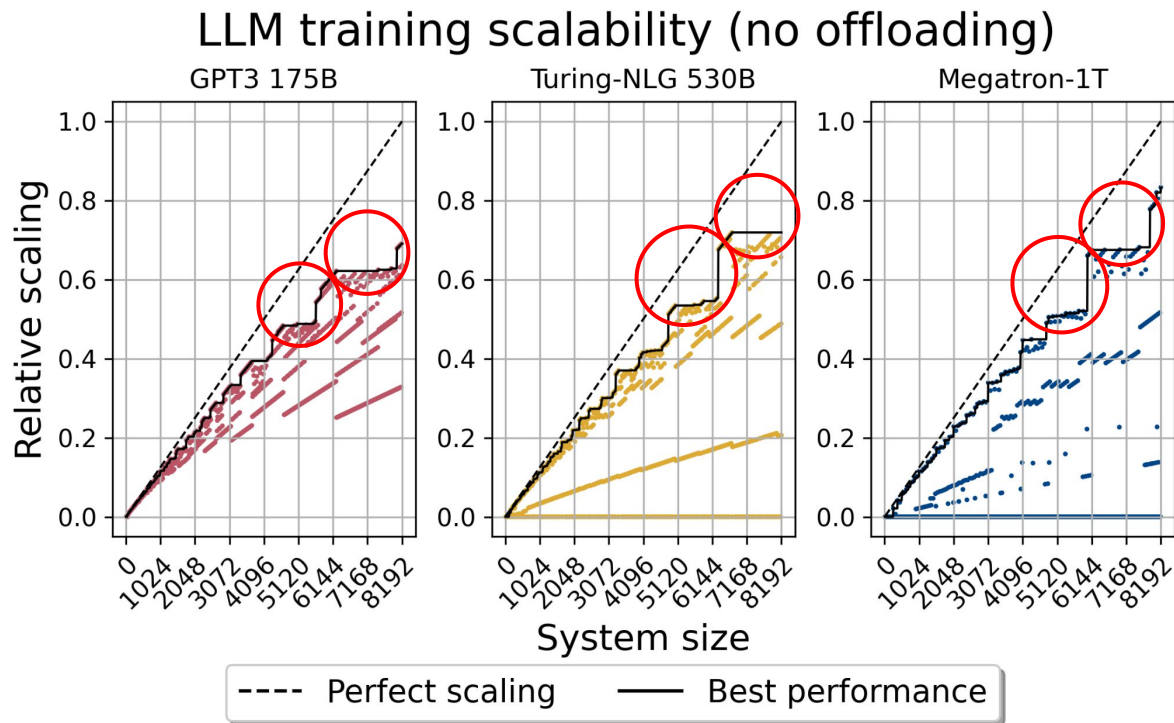
System sizes with step 8

All processors are used

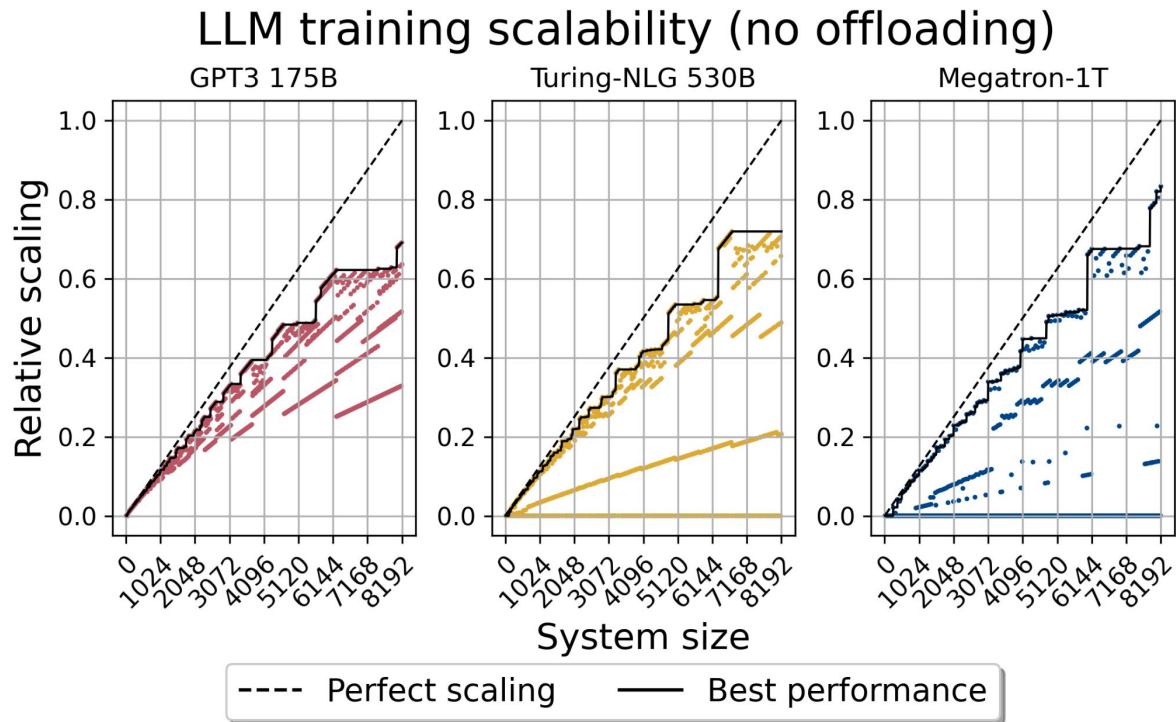
Each point is 10^5 - 10^7 experiments

Some configurations don't run due to mapping issues

Other configurations have lower performance, unable to find good parallelization strategy

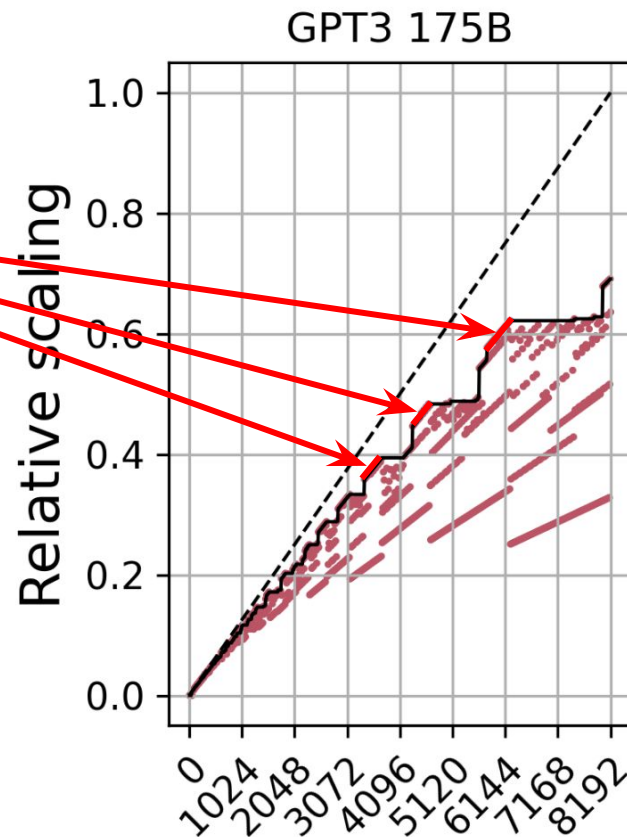


But what if we lost a node?



But what if we lost a node?

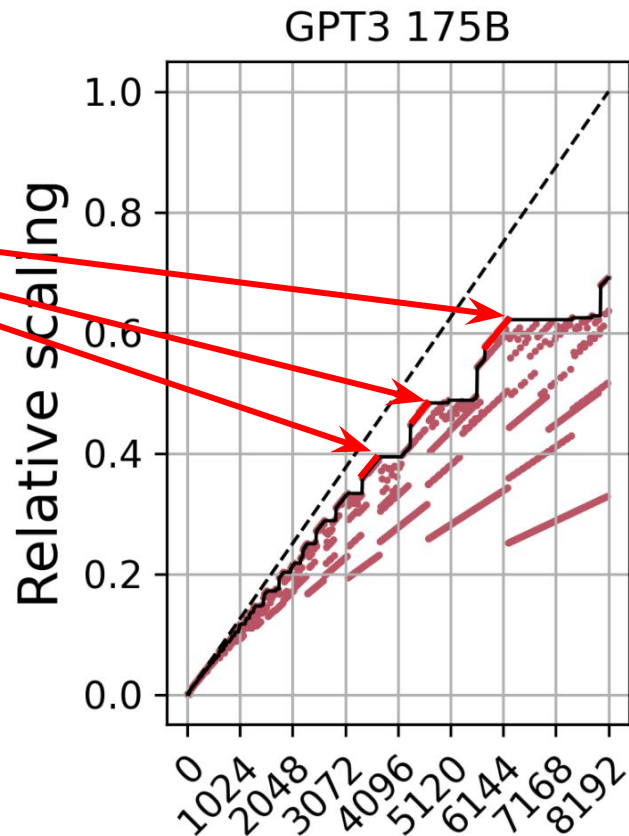
We want to stay on the
linear scaling sections



But what if we lost a node?

We want to stay on the
linear scaling sections

If we are on the “right side”,
Calculon can find a new optimal
config, **we recompile** and
degrade performance gracefully

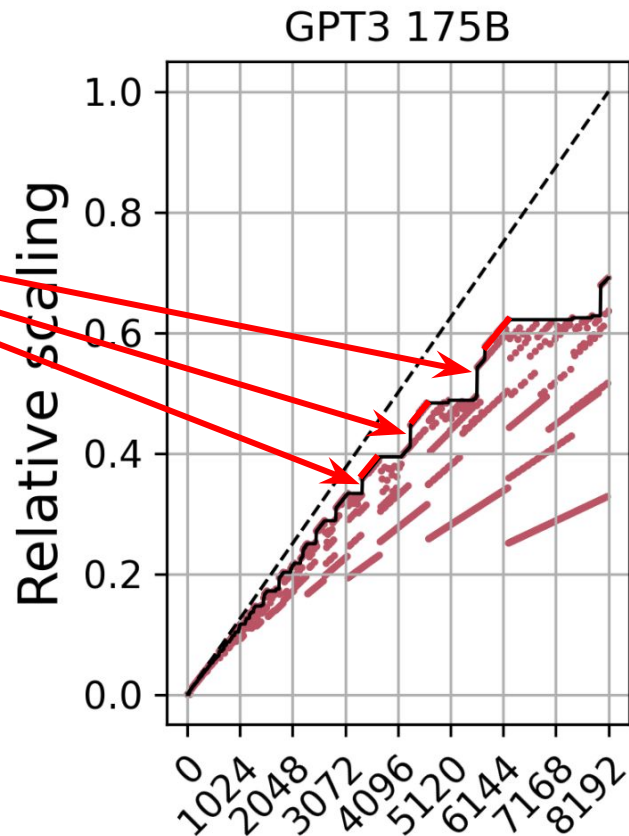


But what if we lost a node?

We want to stay on the
linear scaling sections

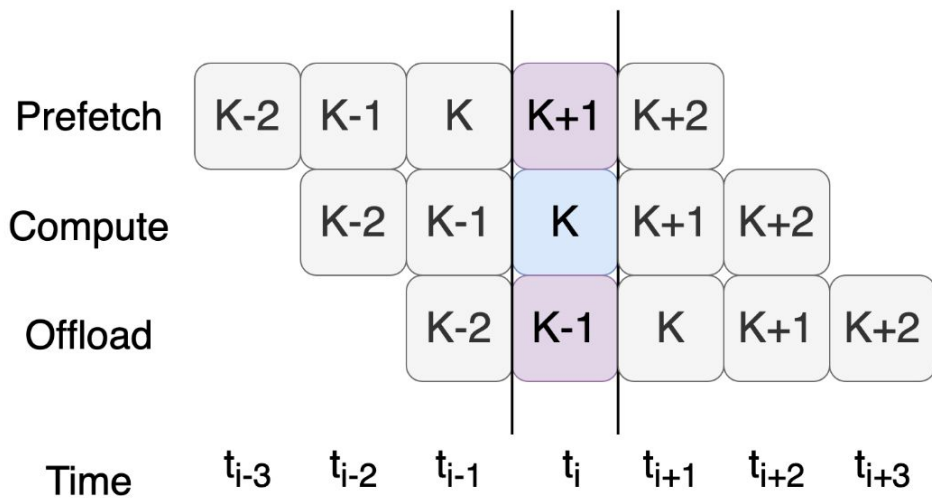
If we are on the “right side”,
Calculon can find a new optimal
config, **we recompile** and
degrade performance gracefully

Otherwise, **we fall off the cliff**

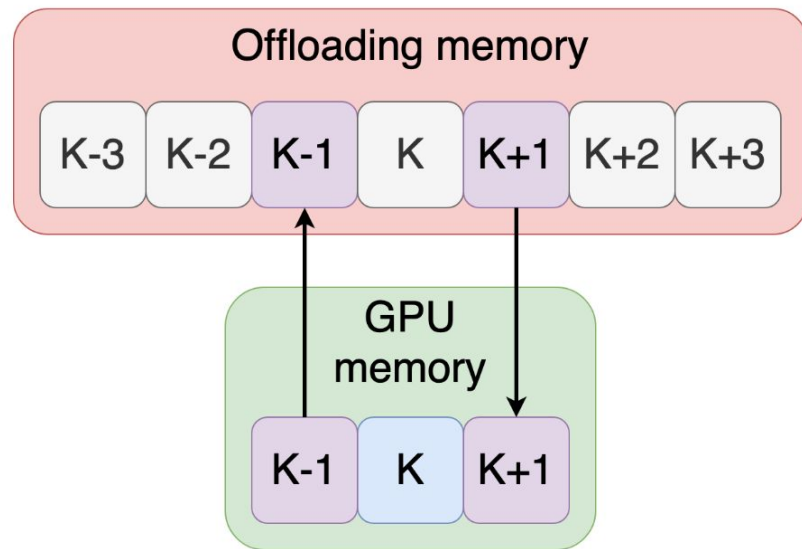


Study 4: tensor offloading

Compute pipeline



Memory exchange at time t



Study 4: tensor offloading analysis

With infinite offloading memory, we consume > 512 GB @ up to 150 GB/s

Megatron-1T training on 4096 H100 80 GiB GPUs with a secondary memory available for tensor offloading

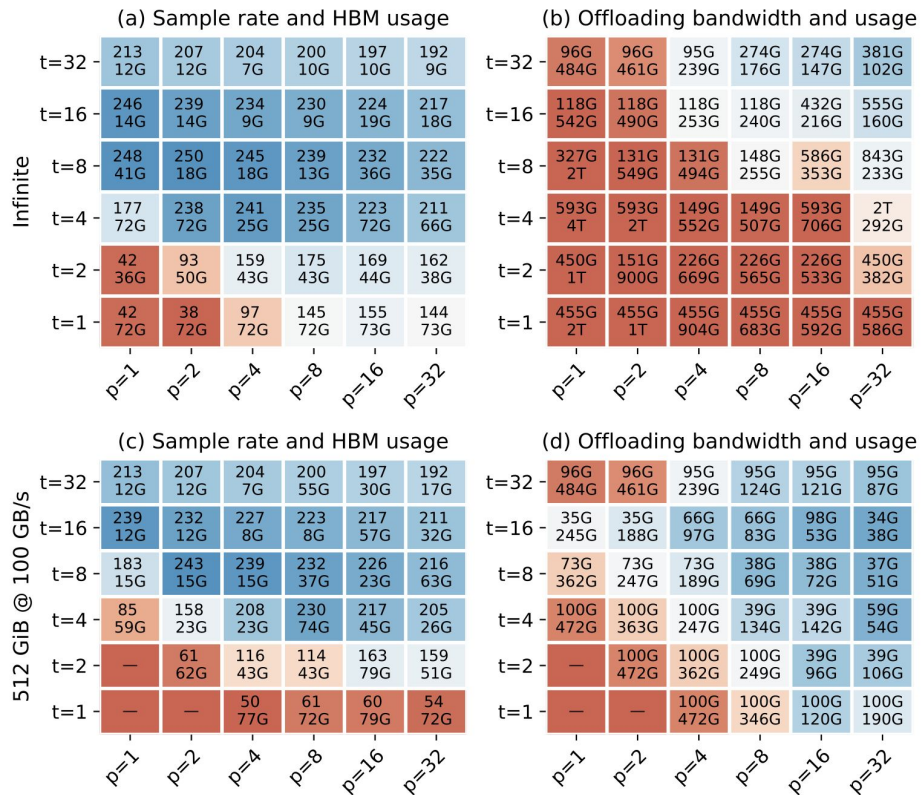
(a) Sample rate and HBM usage							(b) Offloading bandwidth and usage							
Infinite	t=32	213 12G	207 12G	204 7G	200 10G	197 10G	192 9G	t=32	96G 484G	96G 461G	95G 239G	274G 176G	274G 147G	381G 102G
	t=16	246 14G	239 14G	234 9G	230 9G	224 19G	217 18G	t=16	118G 542G	118G 490G	118G 253G	118G 240G	432G 216G	555G 160G
	t=8	248 41G	250 18G	245 18G	239 13G	232 36G	222 35G	t=8	327G 2T	131G 549G	131G 494G	148G 255G	586G 353G	843G 233G
	t=4	177 72G	238 72G	241 25G	235 25G	223 72G	211 66G	t=4	593G 4T	593G 2T	149G 552G	149G 507G	593G 706G	2T 292G
	t=2	42 36G	93 50G	159 43G	175 43G	169 44G	162 38G	t=2	450G 1T	151G 900G	226G 669G	226G 565G	226G 533G	450G 382G
	t=1	42 72G	38 72G	97 72G	145 72G	155 73G	144 73G	t=1	455G 2T	455G 1T	455G 904G	455G 683G	455G 592G	455G 586G
		p=1	p=2	p=4	p=8	p=16	p=32		p=1	p=2	p=4	p=8	p=16	p=32

Study 4: tensor offloading analysis

With infinite offloading memory, we consume > 512 GB @ up to 150 GB/s

Performance changes insignificantly when 512 GB @ 100 GB/s used

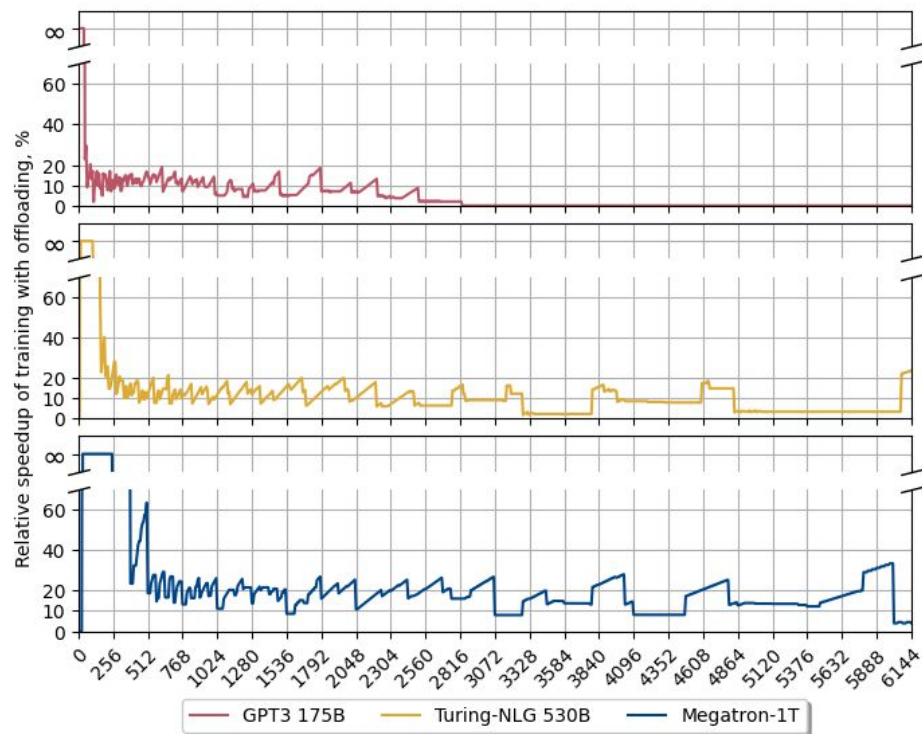
Megatron-1T training on 4096 H100 80 GiB GPUs with a secondary memory available for tensor offloading



Study 4: tensor offloading analysis

Performance speedup compared to
“no offloading” case (study 3)

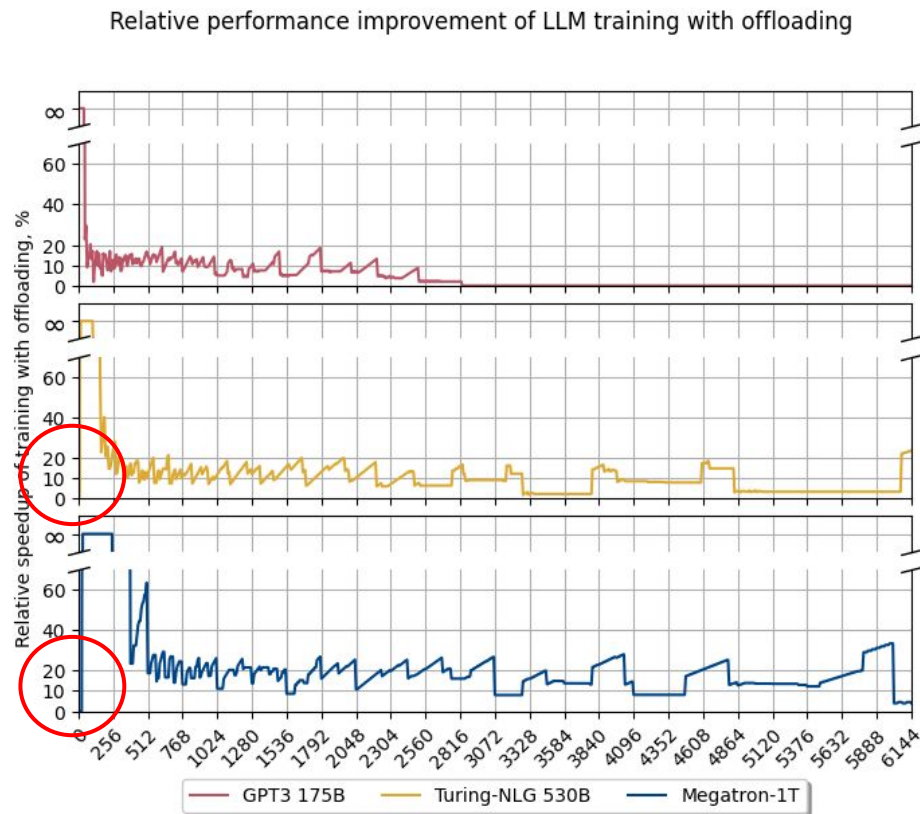
Relative performance improvement of LLM training with offloading



Study 4: tensor offloading analysis

Performance speedup compared to “no offloading” case (study 3)

Extra memory brings up to 20% performance for large LLM training

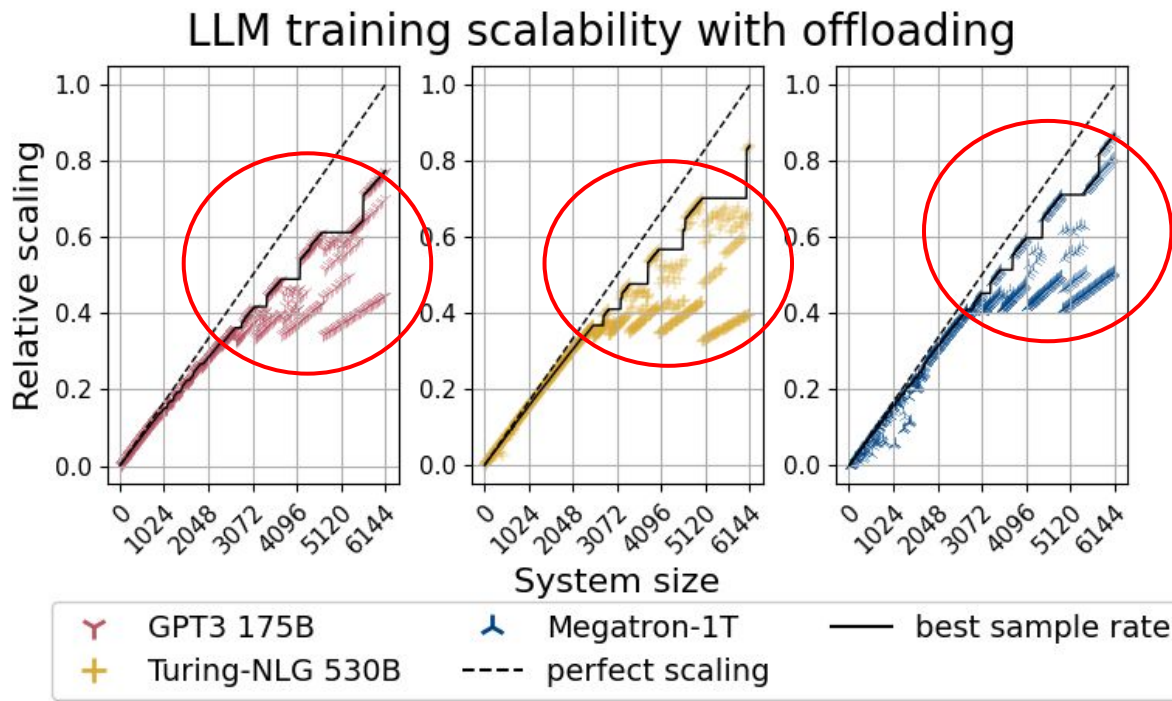


Study 4: tensor offloading analysis

More memory means more parallelization options

Better stability for larger LLMs, as

- Utilization grows
- Flat perf regions shrink



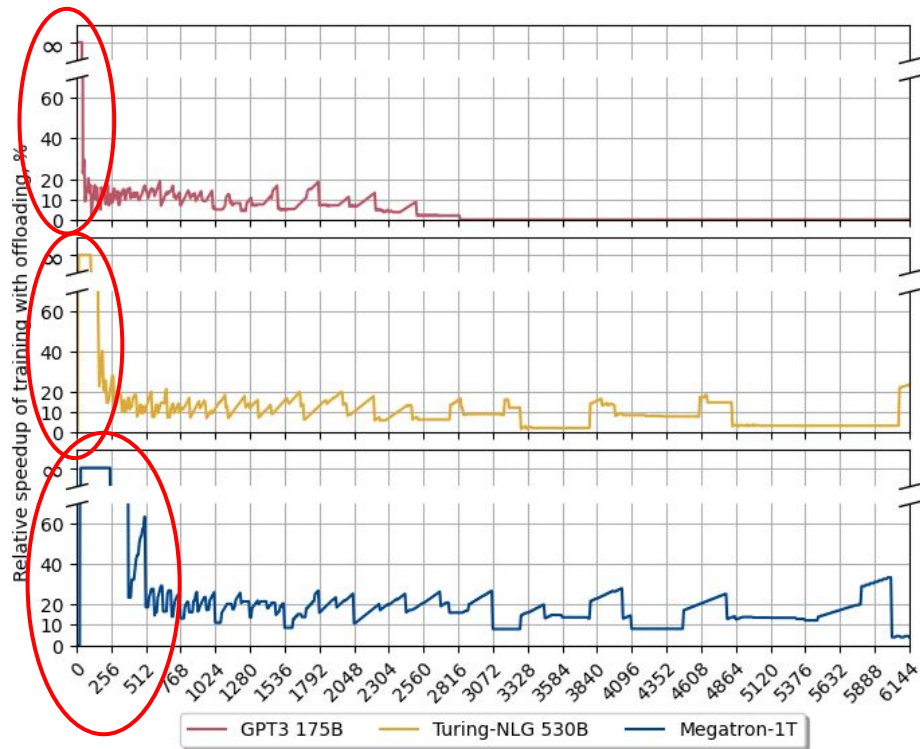
Study 4: tensor offloading analysis

Performance speedup compared to “no offloading” case (study 3)

Extra memory brings up to 20% performance for large LLM training

It enables large LLM training on small systems up to 256 nodes (fine-tuning)

Relative performance improvement of LLM training with offloading



Study 5: system co-design for current LLMs

- H100-based system with 8 GPUs per node (DGX box)
- Variable fast (HBM3) and slow (DDR5) memory for offloading
- Fixed budget of \$125M per system
- Best performing configuration for 175B, 530B, and 1T model training
- Non-linear cost model

Study 5: system co-design for current LLMs

Table 3: Price and performance results for various systems and LLMs under a fixed budget of \$125M.

HBM3	DDR5	Price	Max GPUs	1T			530B			175B		
				GPUs	Perf	Perf/\$M	GPUs	Perf	Perf/\$M	GPUs	Perf	Perf/\$M
20G	0	\$22.5k	5616	5120	105	93	5600	365	293	5472	1143	939
40G	0	\$25k	5000	4864	223	184	4984	475	381	4672	1211	1037
80G	0	\$30k	4160	4096	232	189	4080	405	331	3744	1014	903
120G	0	\$40k	3120	3072	179	145	2880	293	254	3120	875	701
20G	256G	\$25k	5048	4912	297	244	5040	531	426	4672	1253	1083
40G	256G	\$27.5k	4544	4544	275	220	4440	396	325	4544	1219	975
80G	256G	\$32.5k	3840	3840	234	187	3840	415	333	3744	1014	834
120G	256G	\$42.5k	2936	2928	167	135	2880	316	258	2928	812	652
20G	512G	\$32.5k	3872	3872	236	189	3864	418	335	3744	1014	840
40G	512G	\$35k	3568	3504	215	175	3360	367	312	3568	967	774
80G	512G	\$40k	3120	3072	189	154	2880	316	272	3120	865	693
120G	512G	\$50k	2496	2496	156	125	2496	280	224	2496	700	561
40G	1T	\$42.5k	2952	2944	181	146	2880	316	259	2944	803	645
40G	1T	\$45k	2776	2728	171	139	2760	302	244	2672	749	623
80G	1T	\$50k	2496	2496	156	125	2496	280	224	2496	700	561
120G	1T	\$60k	2080	2080	132	106	2080	238	190	2080	596	478

Study 5: system co-design for current LLMs

Table 3: Price and performance results for various systems and LLMs under a fixed budget of \$125M.

HBM3	DDR5	Price	Max GPUs	1T			530B			175B		
				GPUs	Perf	Perf/\$M	GPUs	Perf	Perf/\$M	GPUs	Perf	Perf/\$M
20G	0	\$22.5k	5616	5120	105	93	5600	365	293	5472	1143	939
40G	0	\$25k	5000	4864	223	184	4984	475	381	4672	1211	1037
80G	0	\$30k	4160	4096	232	189	4080	405	331	3744	1014	903
120G	0	\$40k	3120	3072	179	145	2880	293	254	3120	875	701
20G	256G	\$25k	5048	4912	297	244	5040	531	426	4672	1253	1083
40G	256G	\$27.5k	4544	4544	275	220	4440	396	325	4544	1219	975
80G	256G	\$32.5k	3840	3840	234	187	3840	415	333	3744	1014	834
120G	256G	\$40.5k	3024	3024	167	135	2880	316	258	3024	812	652

More HBM – more expensive systems, **less** processors and **perf/\$\$\$**

Offloading memory has **no extra cost** with saving \$\$\$ by buying less HBM

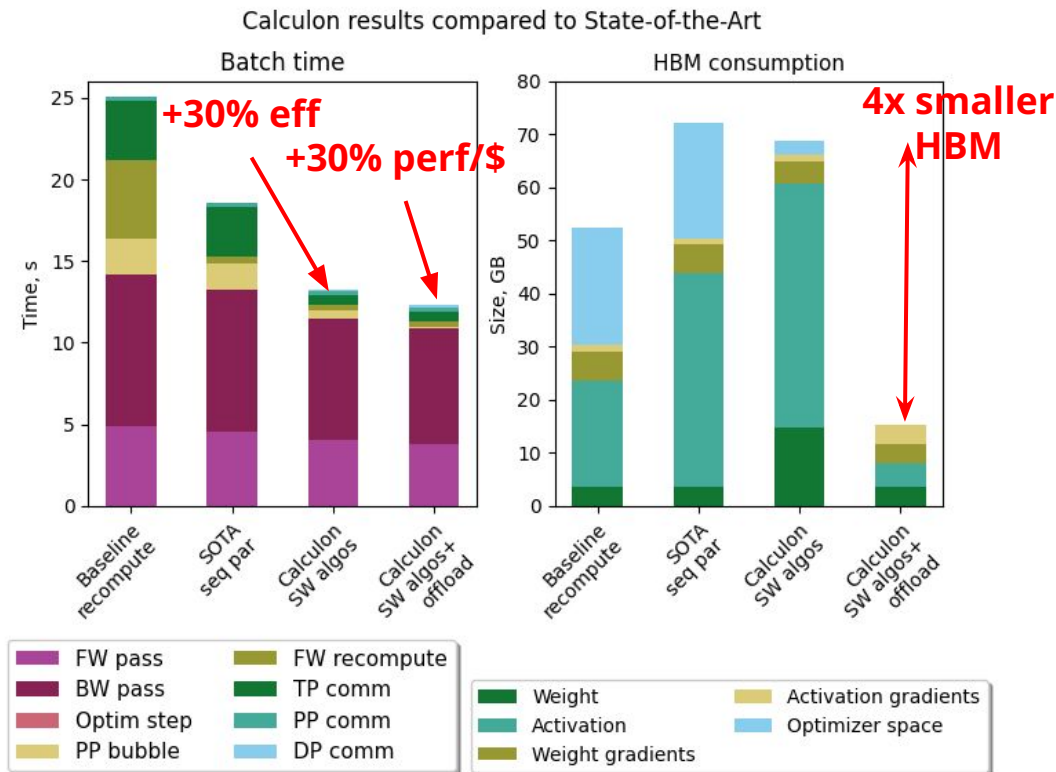
Co-design for current LLMs

New non-trivial system design discovered

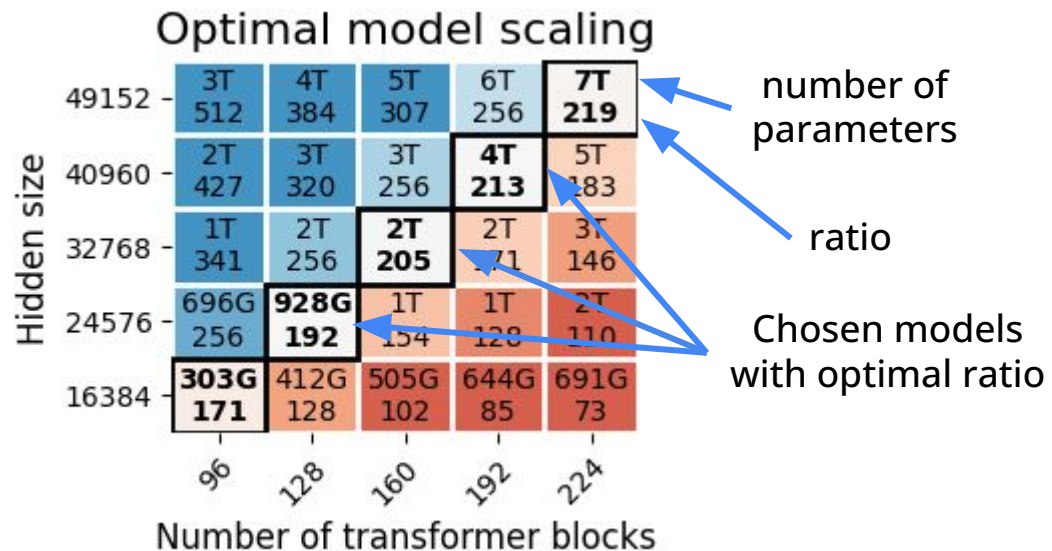
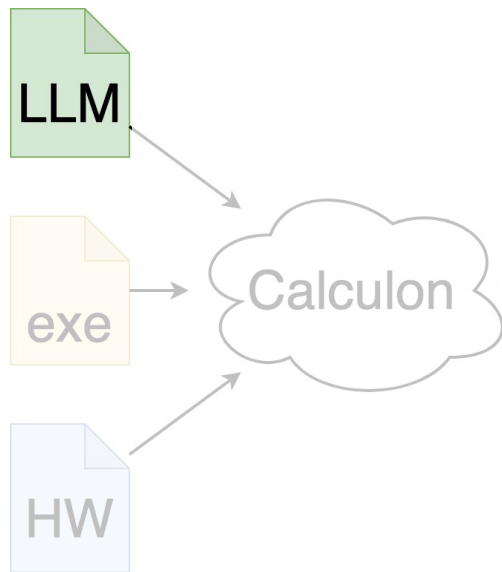
30% performance boost from execution strategy Calculon found

30% performance per \$\$\$ improvement with extra memory for offloading

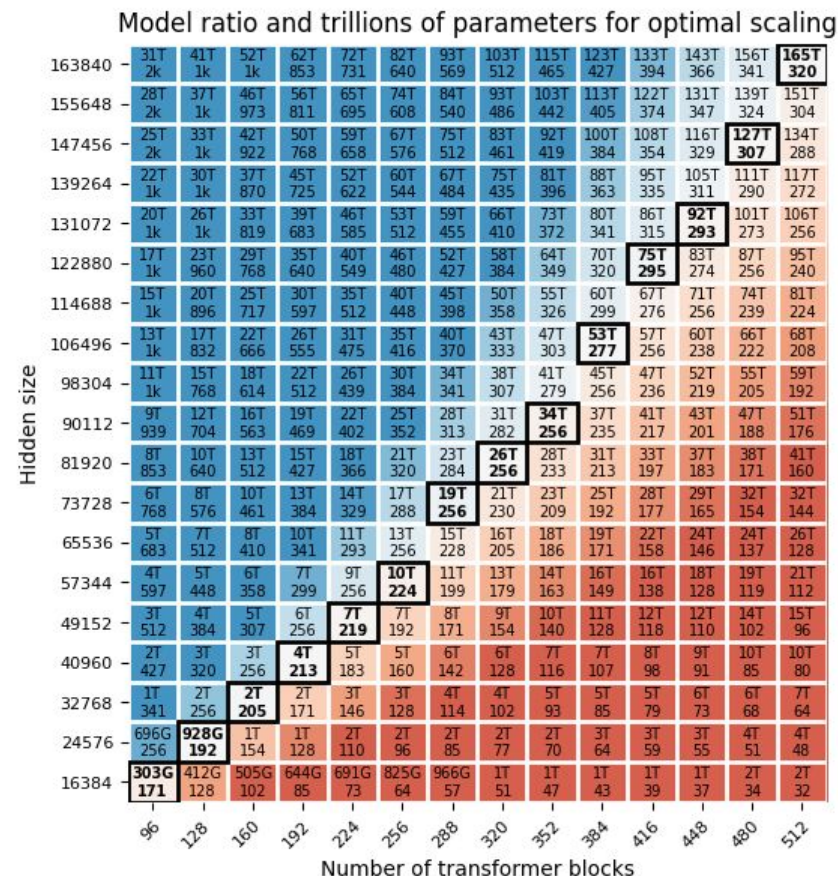
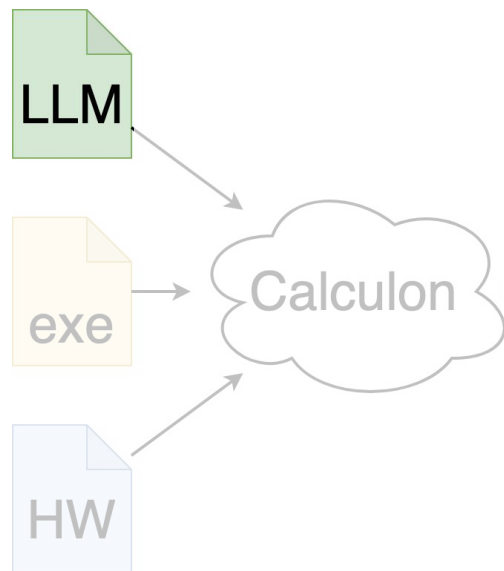
Best optimizations set is not available in current software



Inputs: LLM description

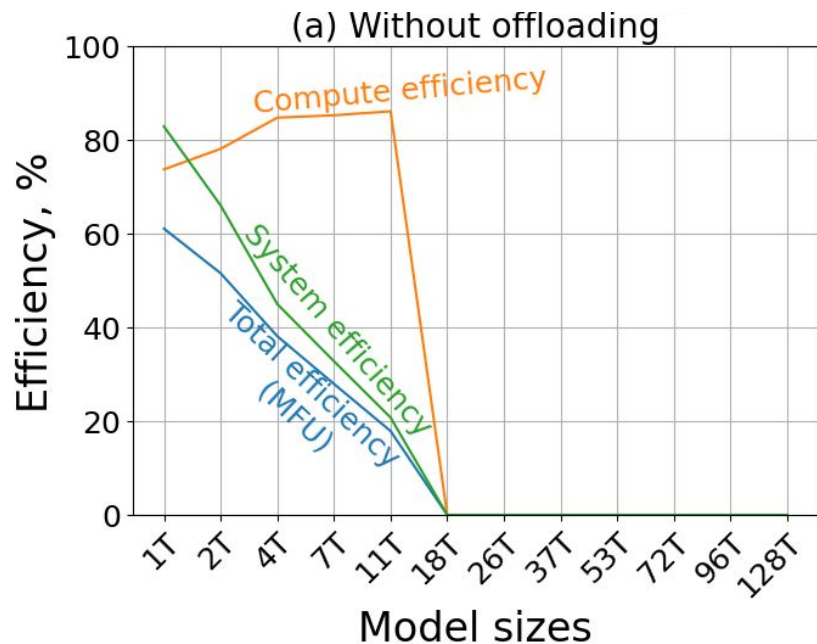


Inputs: LLM description

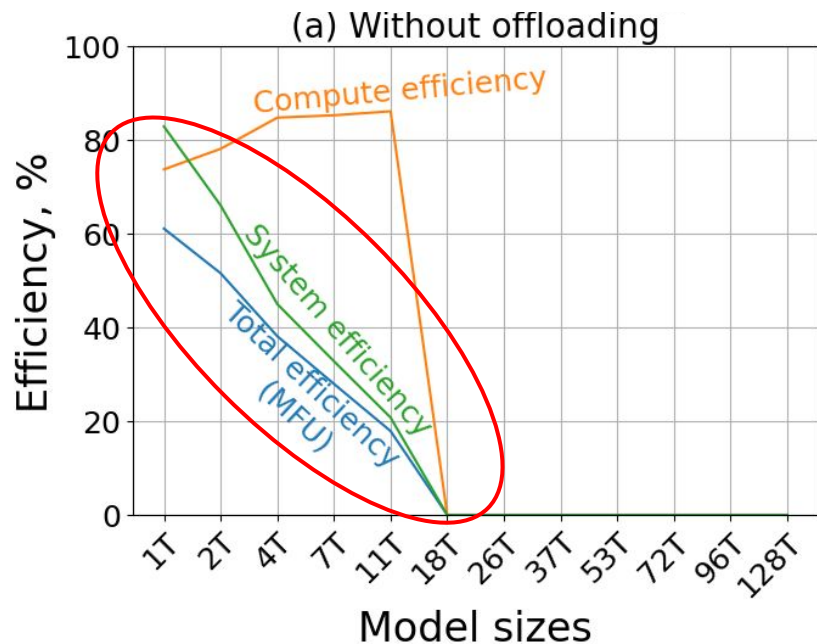


Modeling *future* LLMs, including 100T+ parameters

Tensor offloading performance

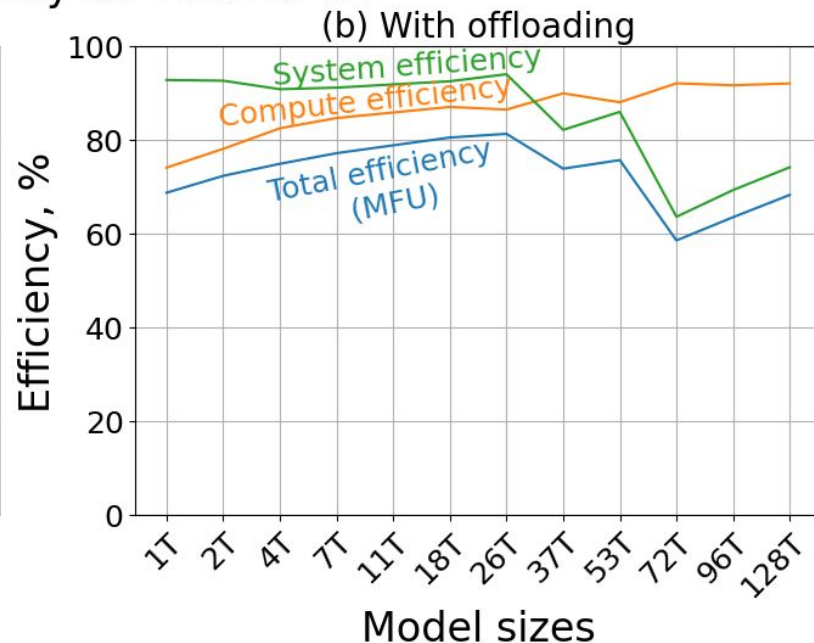
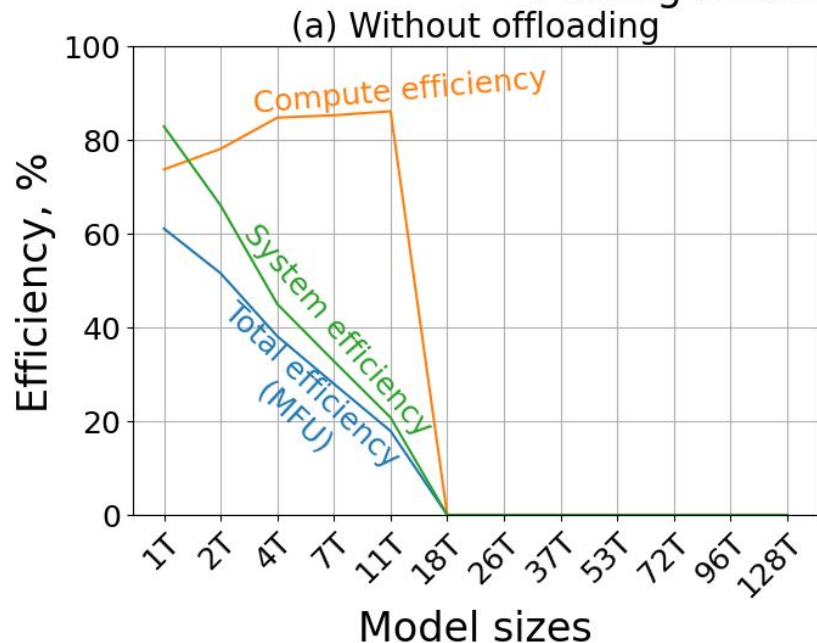


Tensor offloading performance



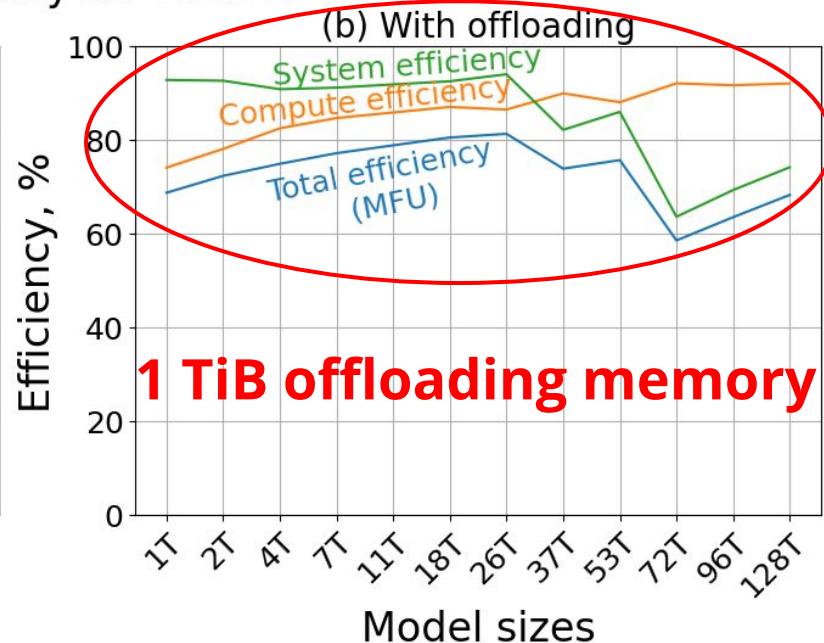
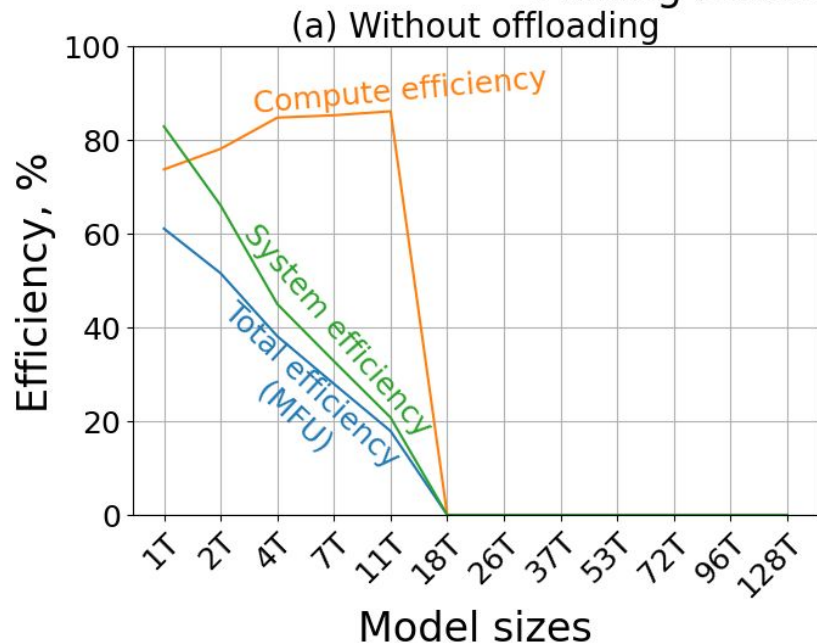
Tensor offloading performance

Training efficiency on 4096 GPUs



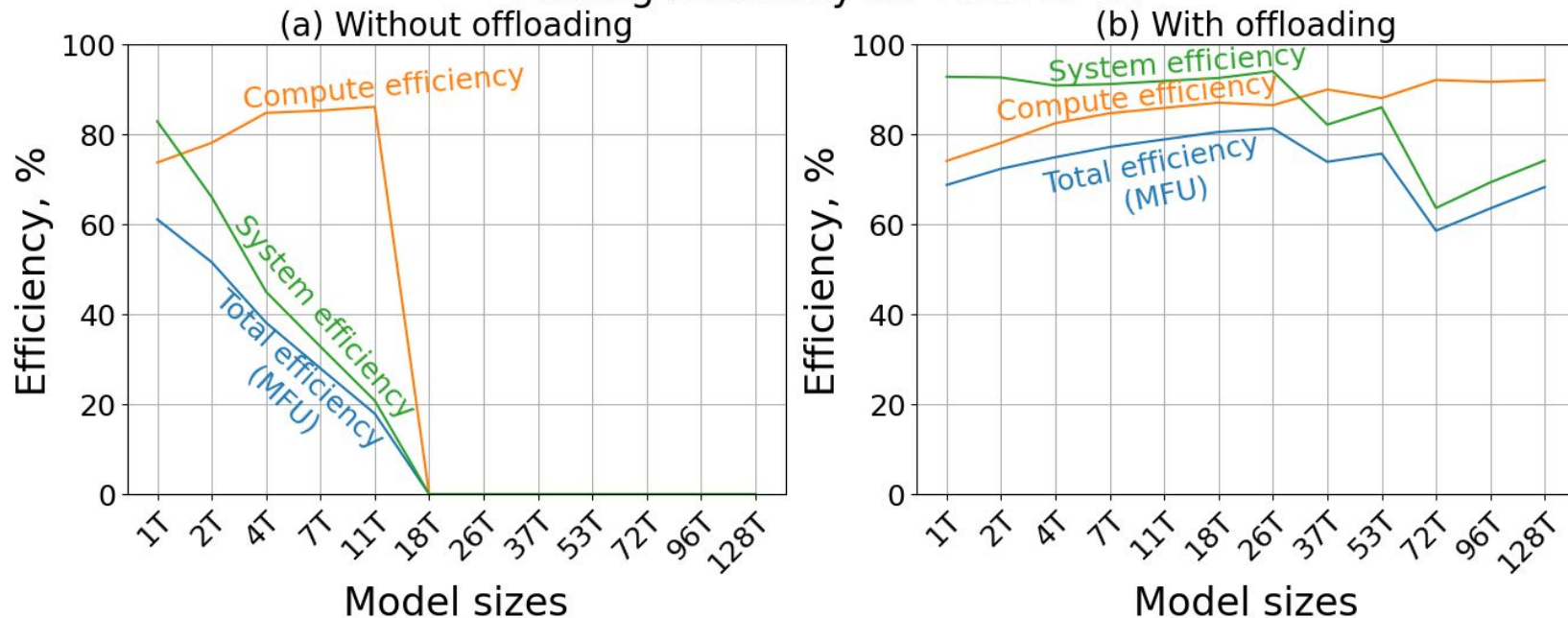
Tensor offloading performance

Training efficiency on 4096 GPUs



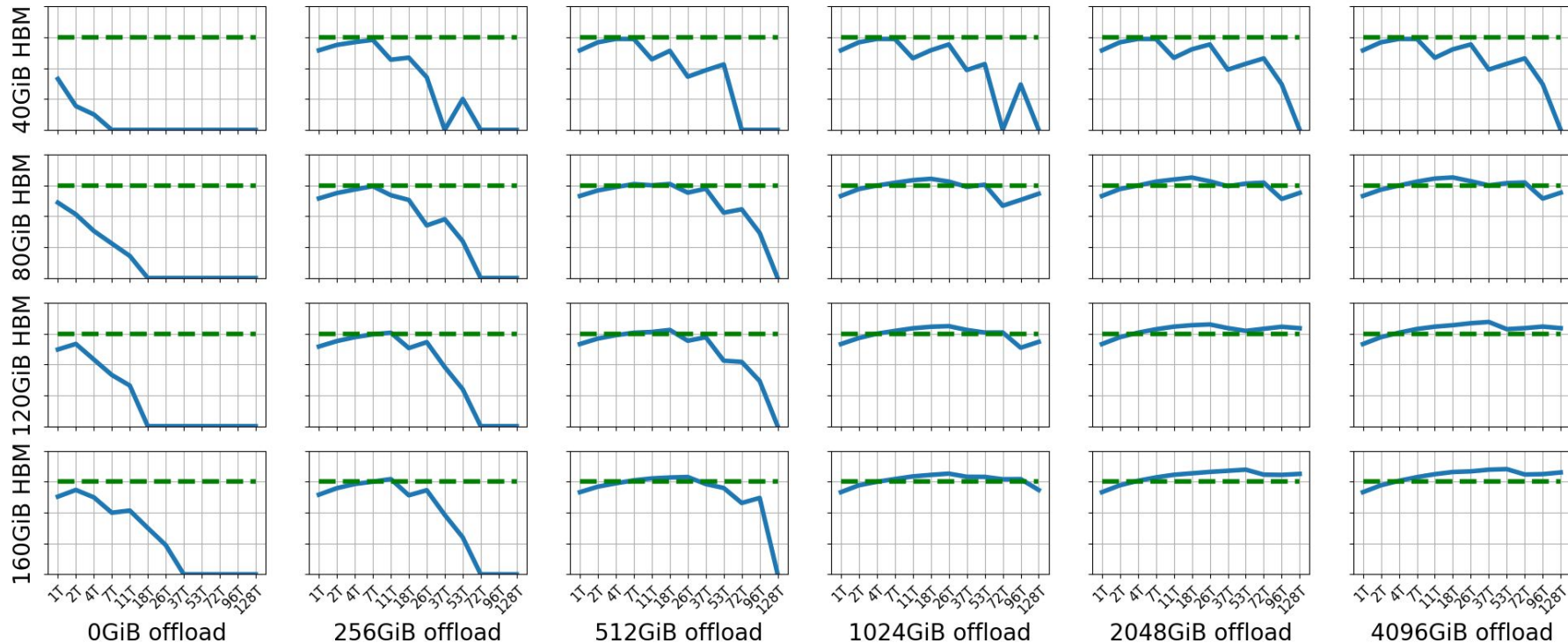
Tensor offloading performance

Training efficiency on 4096 GPUs

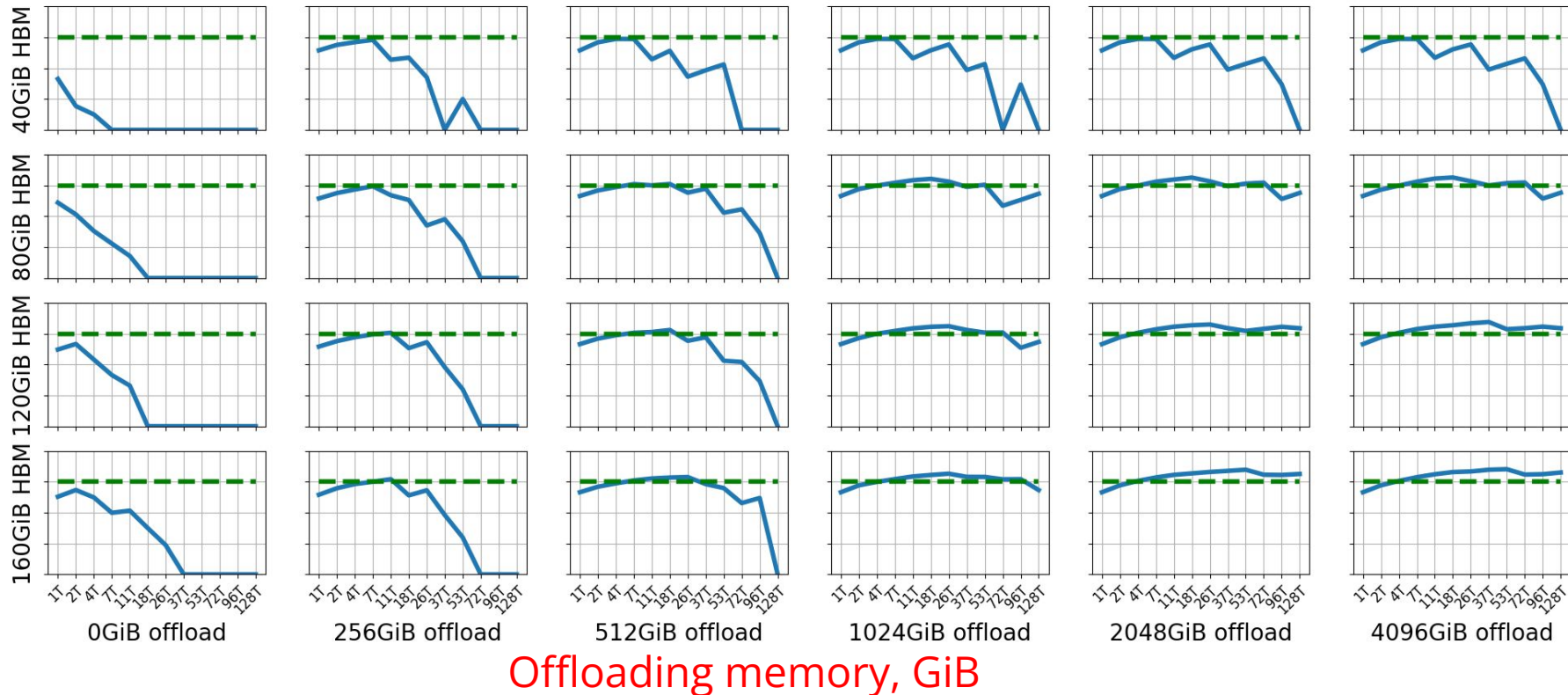


Instrumental to go beyond 10T parameters, with just **100GB/s** bandwidth

Achieved: train 100+T LLMs with 75% MFU (green line)

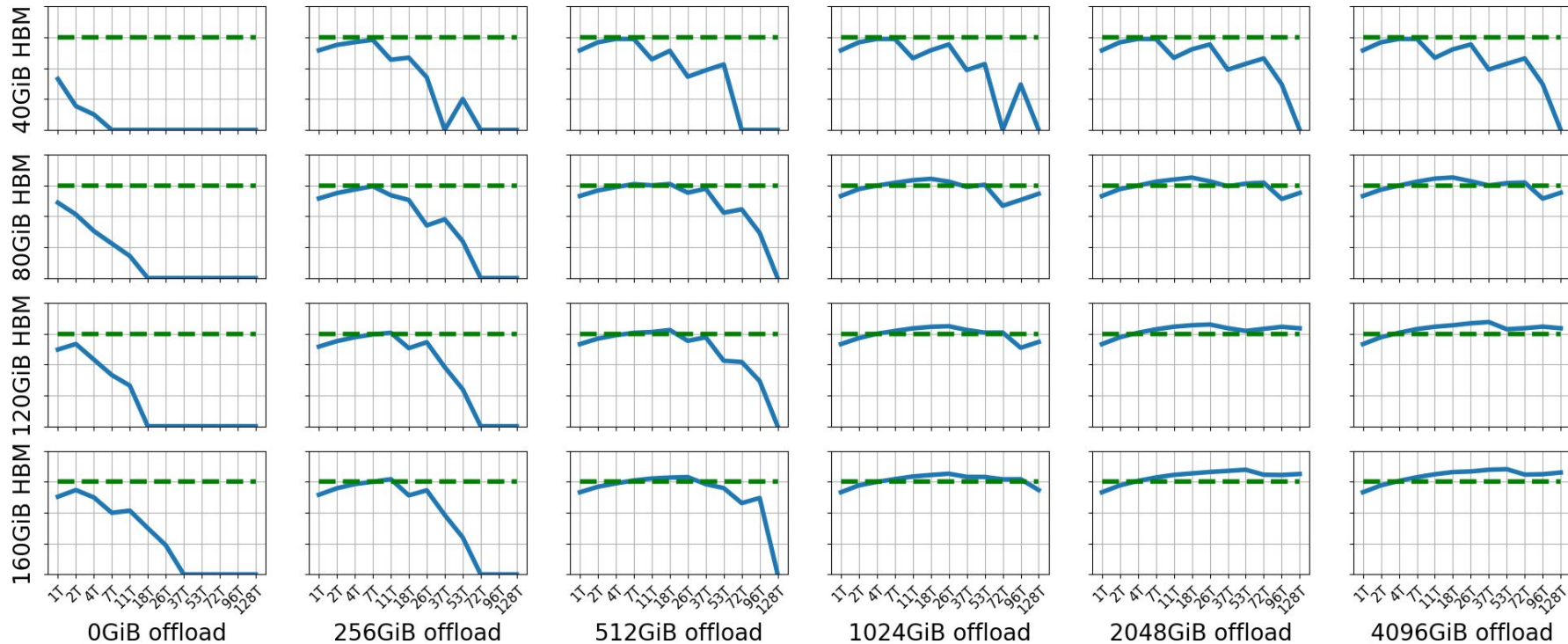


Achieved: train 100+T LLMs with 75% MFU (green line)

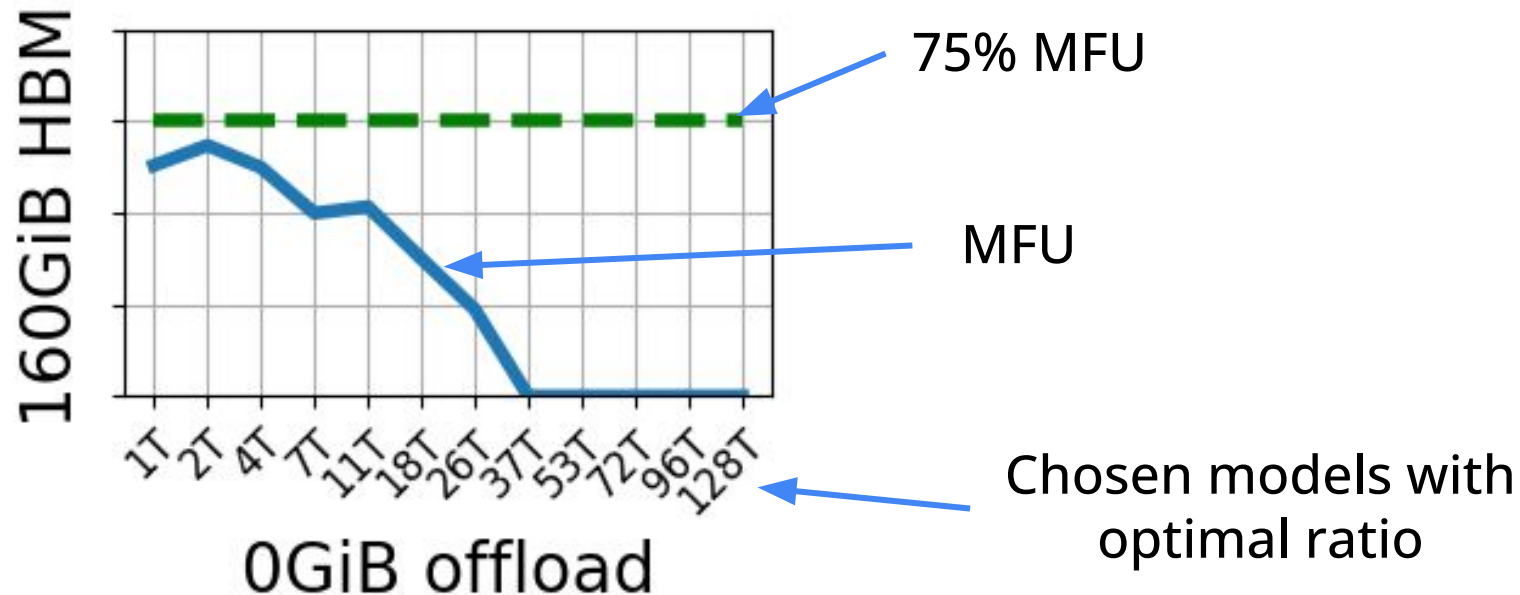


Achieved: train 100+T LLMs with 75% MFU (green line)

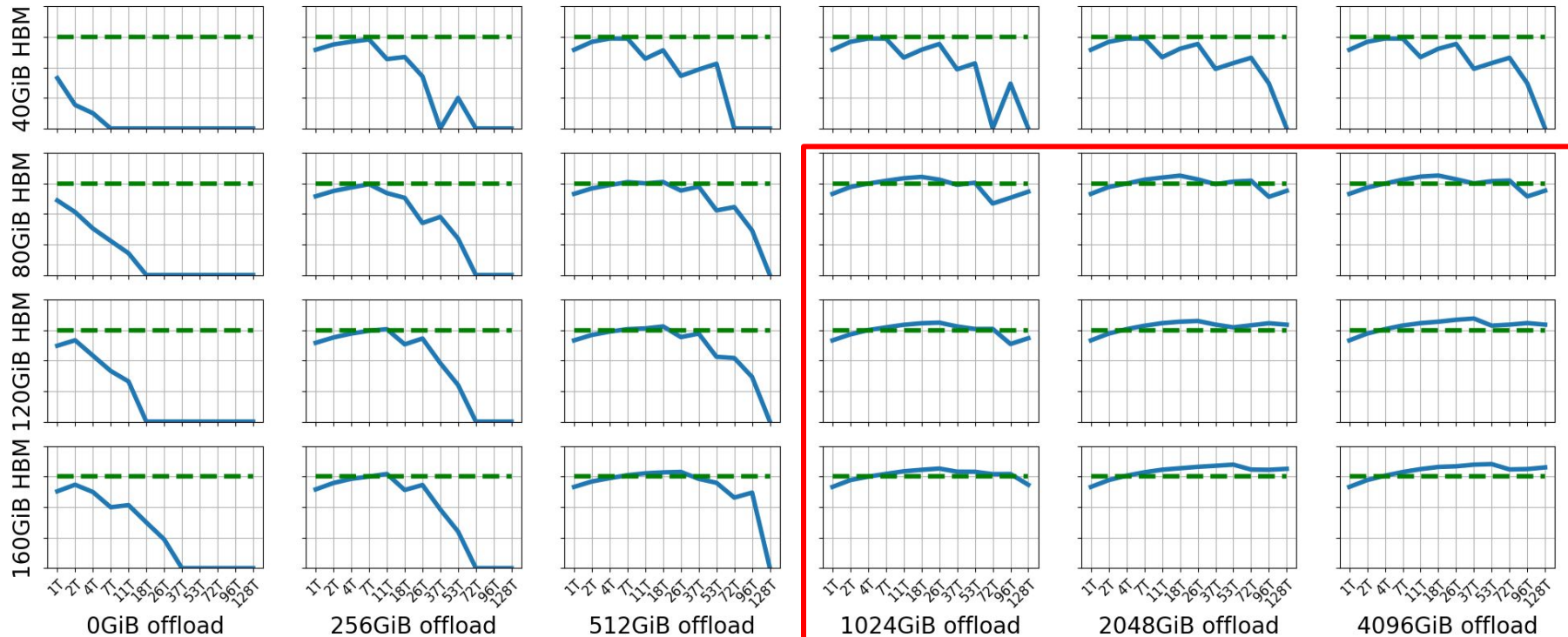
HBM memory, GiB



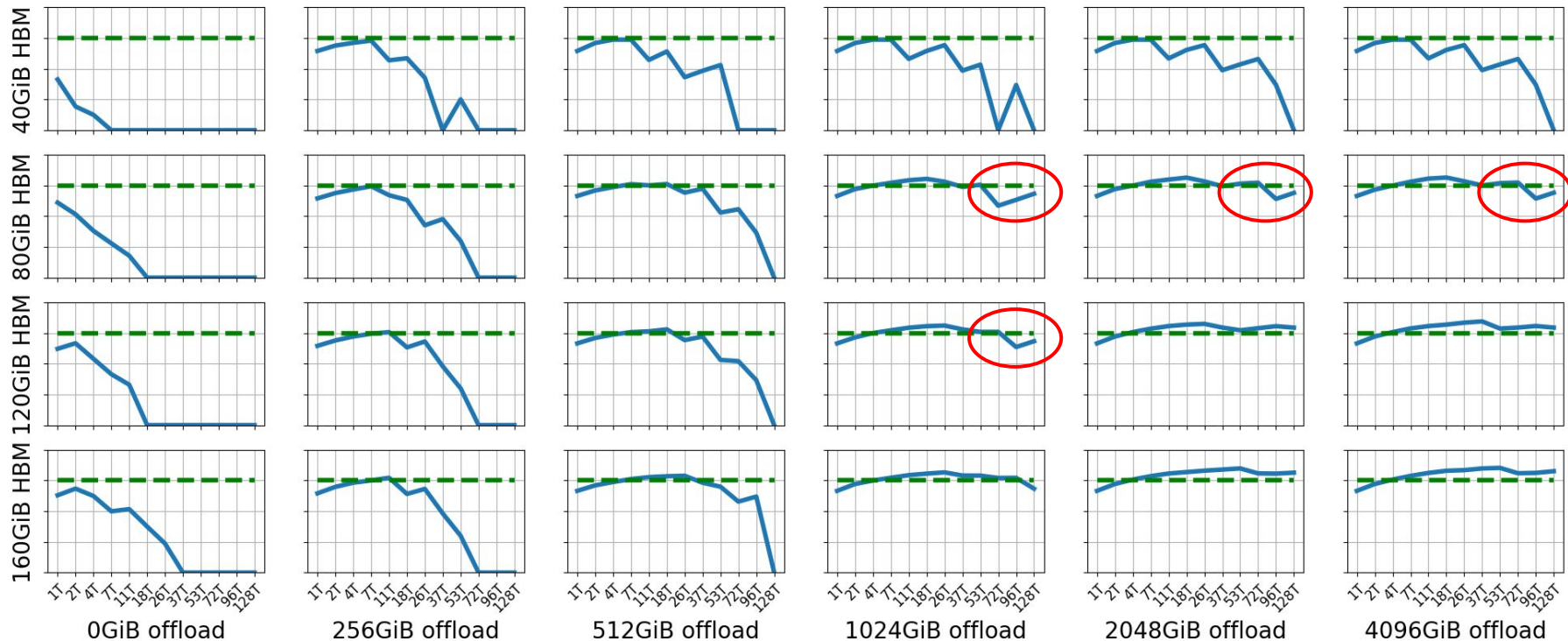
Achieved: train 100+T LLMs with 75% MFU (green line)



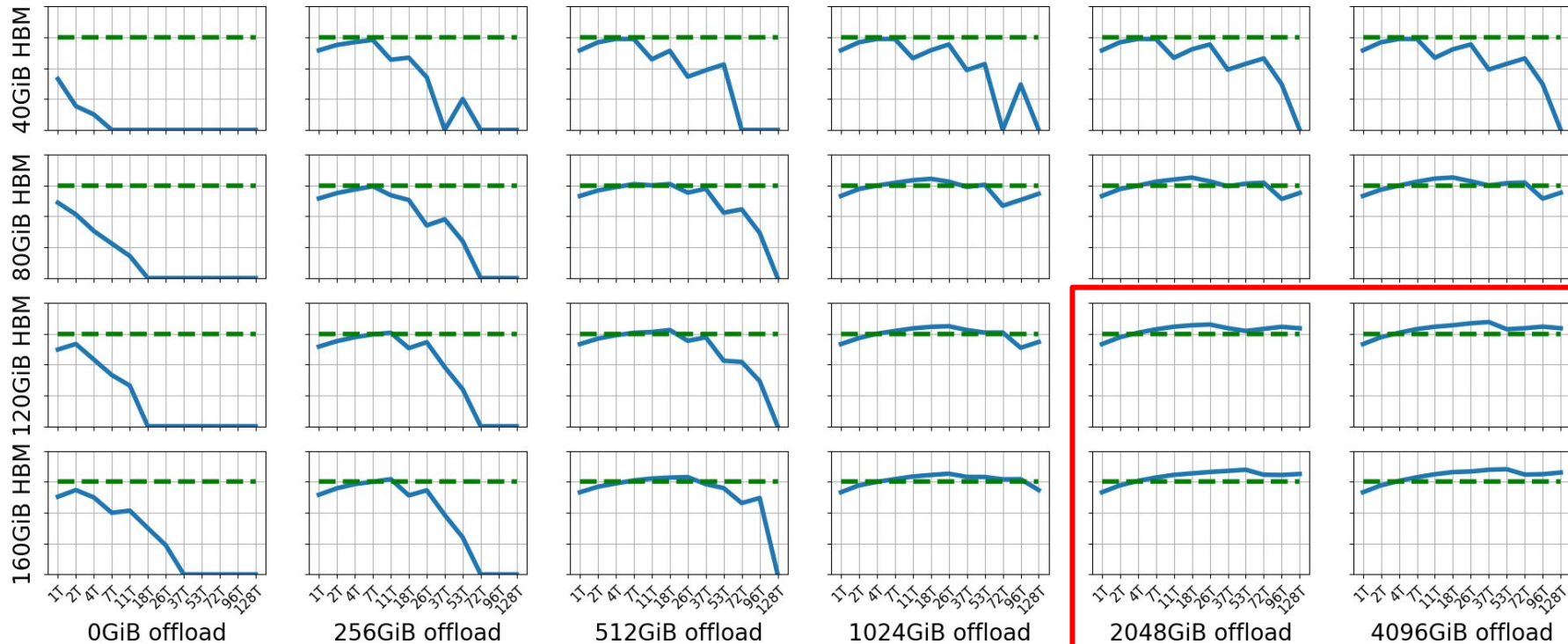
Achieved: train 100+T LLMs with 75% MFU (green line)



Achieved: train 100+T LLMs with 75% MFU (green line)



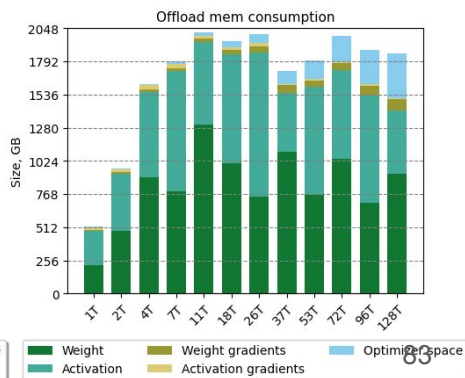
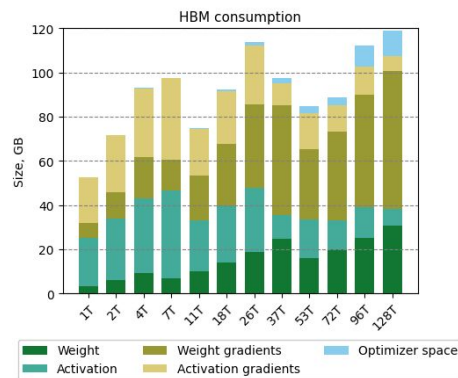
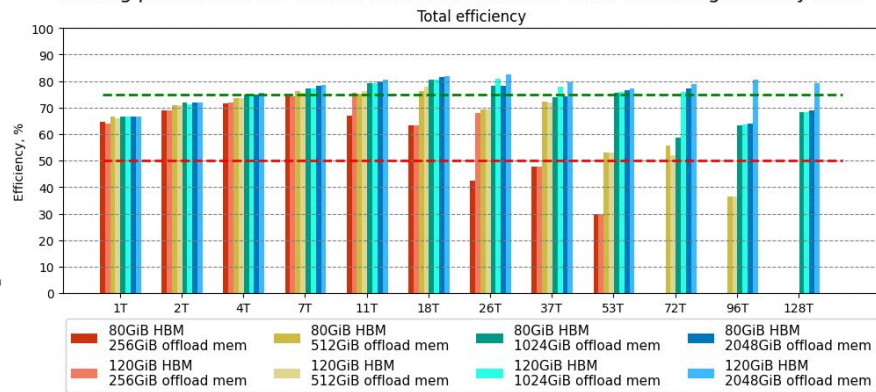
Achieved: train 100+T LLMs with 75% MFU (green line)



System considerations for 100+T parameter LLMs

1. **80 GB HBM** fine for up to **70T**, but we need **120 GB HBM** for **100+T**
2. **512 GB DDR** scale well up to **~10T**
3. **1 TB DDR** for **30-50T**, **2 TB** for **100+T**
4. **Larger memory** allows to **scale DP**
5. 1T model is limited by global batch size as **DP won't scale further**

Training parameters on 4096 GPUs with various HBM and offloading memory sizes

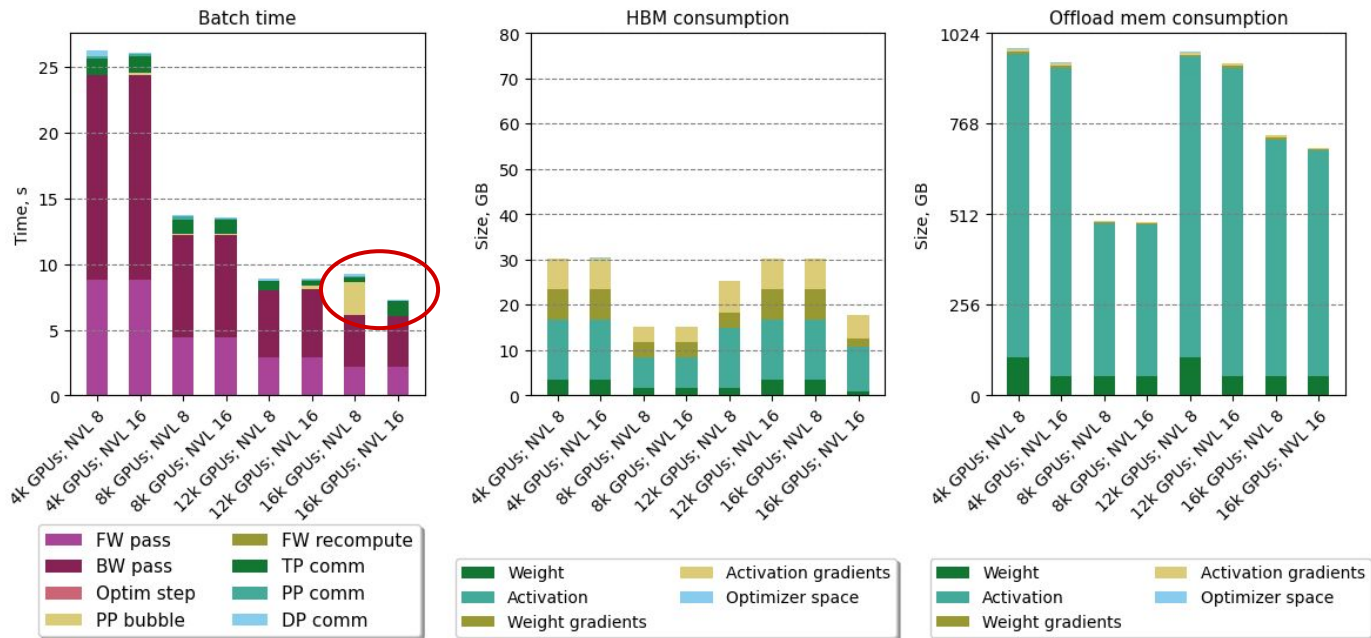


Strong scale bottlenecks

At **16k** nodes we
run out of DP

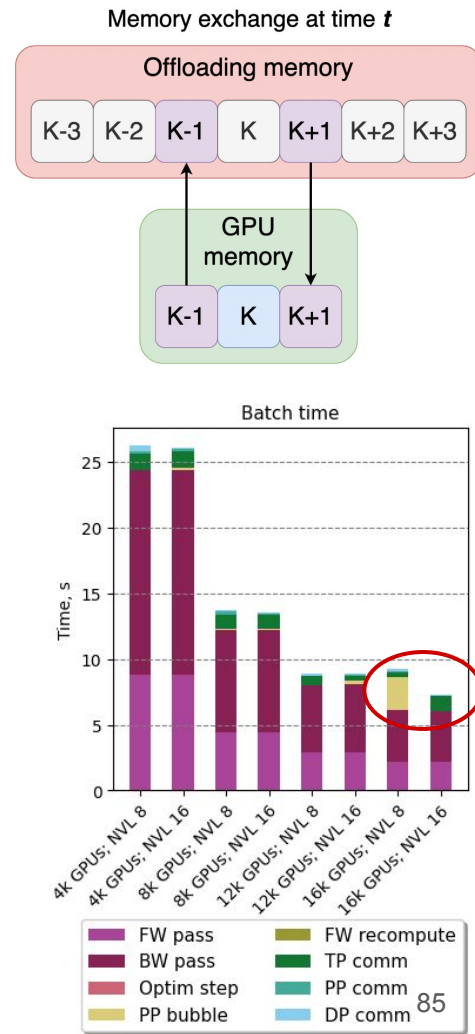
We need **larger TP**,
and **larger NVlink**
to deliver scalability

1T model time and memory consumption breakdown for different system configurations



Calculon summary

1. Calculon can find **optimal LLM split** and optimal **optimization set**, faster than profiling
2. We need **$O(100\text{GB/s})$ for TP**, **$O(10\text{GB/s})$ for PP and DP**, all communication independent, mapped to Torus
3. We can **recompile/redistribute LLM** if node is lost
4. LLMs are **capacity limited**, need memory at 100GB/s, **512GB for < 10T**; **1TB for < 30-50T**; **2TB for ~100T**
5. For scale out **beyond 10k** we need **larger TP, NVLink**



Conclusion

Calculon – open-sourced analytical model of LLMs

Find us at <https://github.com/calculon-ai/>

It facilitates agile co-design considering future hardware, software and optimizations, future LLM models

Calculon helped to find and motivate novel HW systems with large-capacity memory for tensor offloading

Novel systems showed 75+% MFU and great scalability for future LLMs

Future work: support MoE, inference, 3D distribution for matmul

