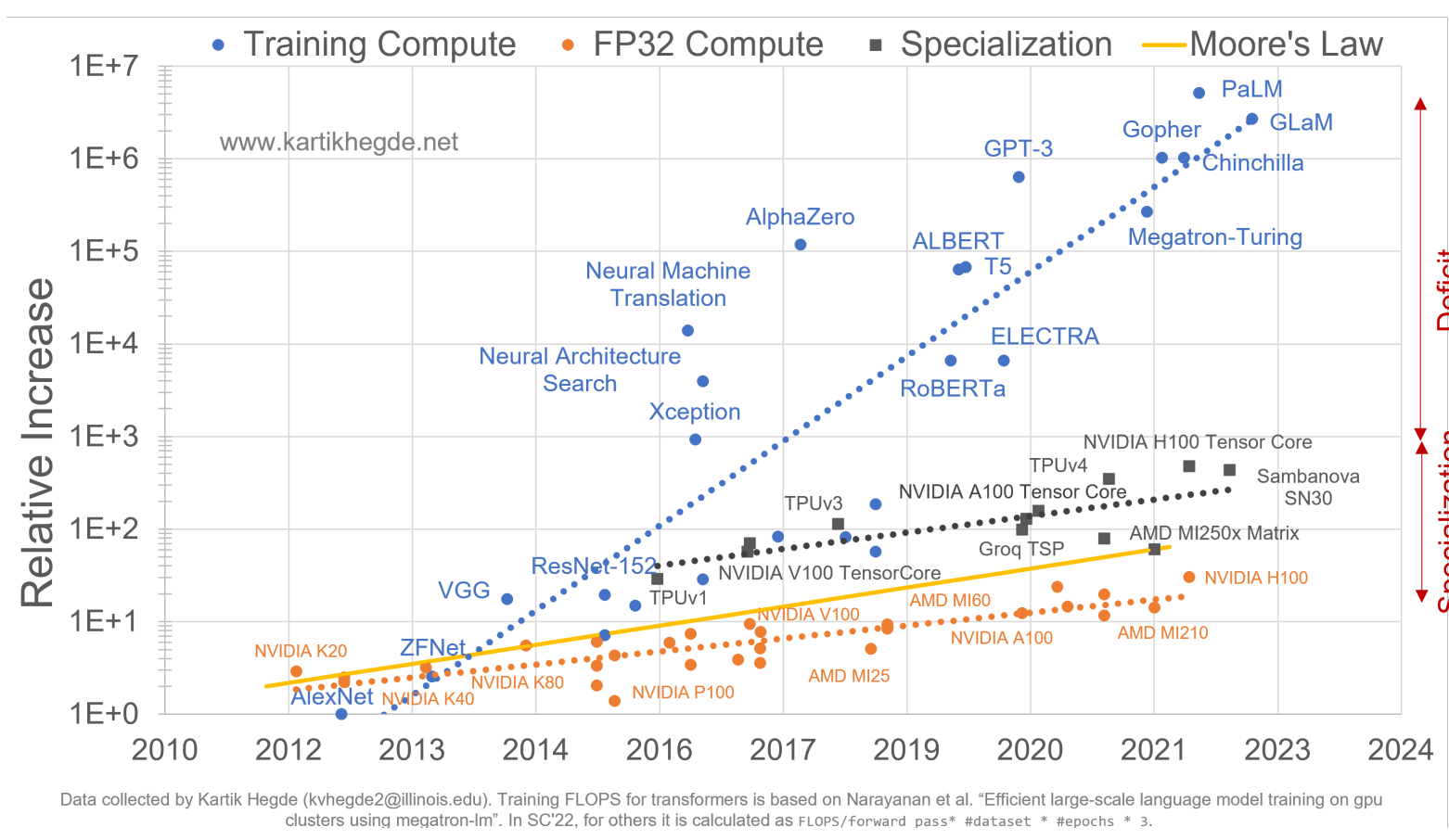
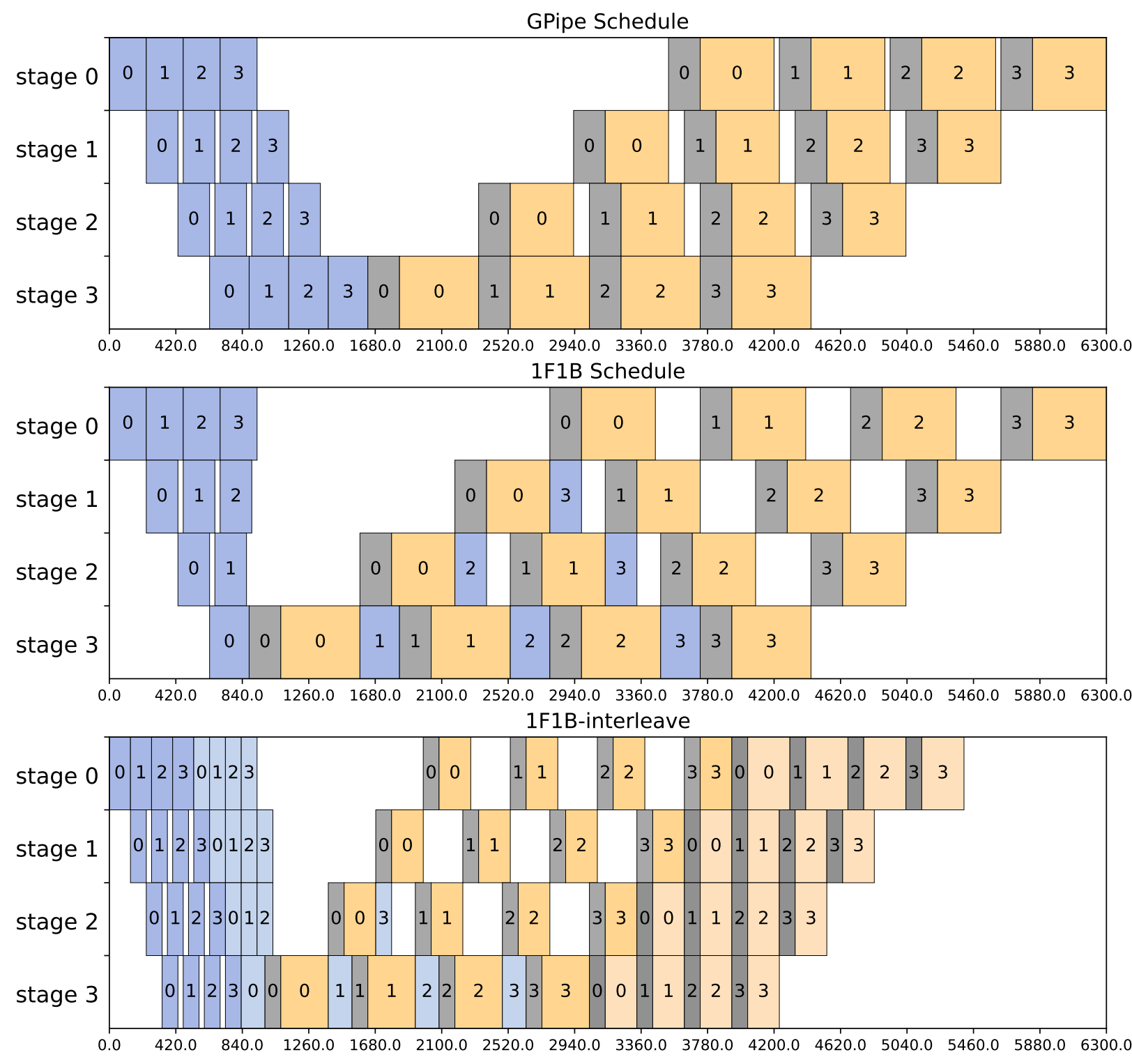


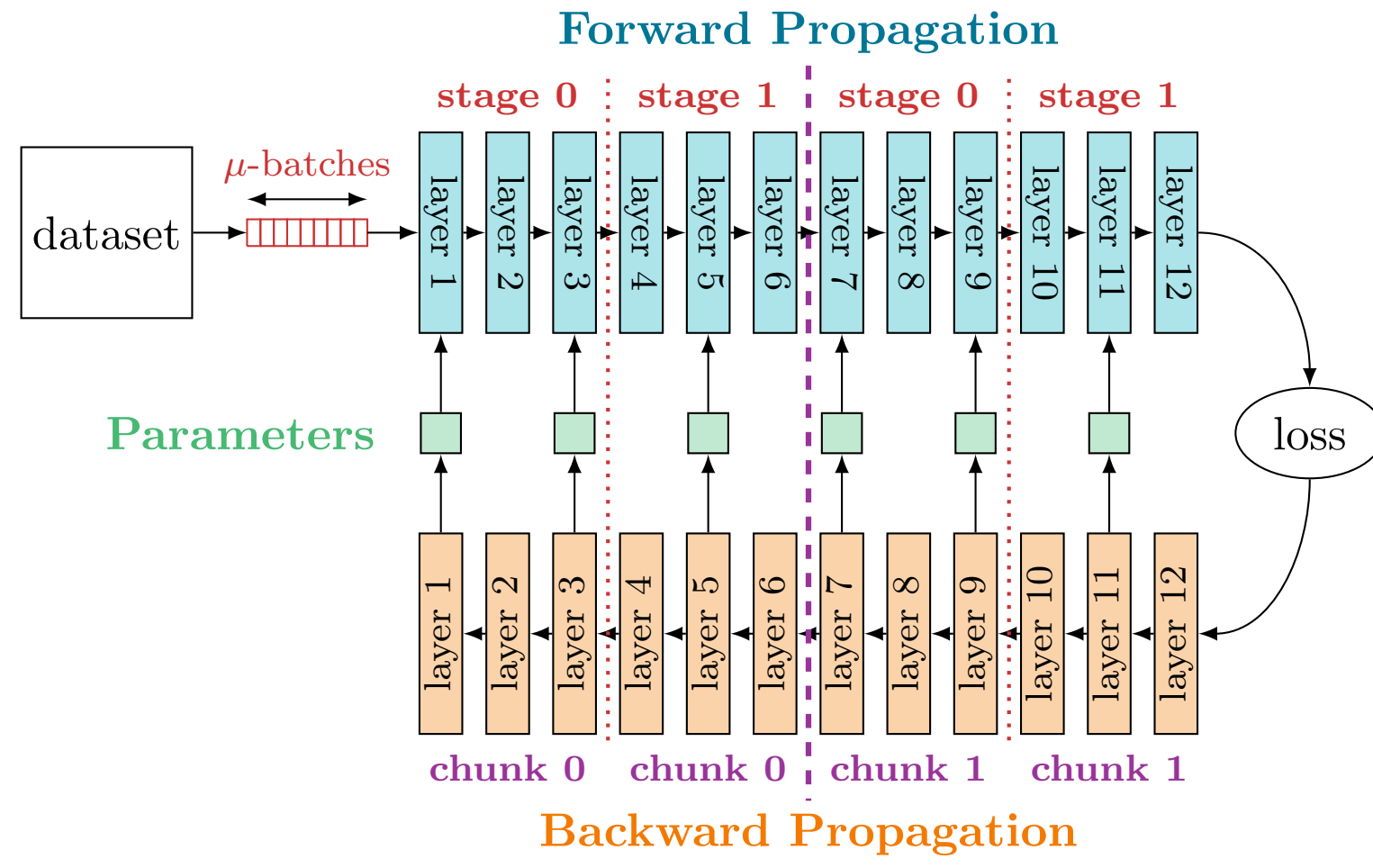
Background



- DNN models increase exponentially, making single-device training impossible.
- Pipeline parallelism** and **activation recomputation** are widely adopted optimization techniques to scale DNN training on large accelerator clusters.

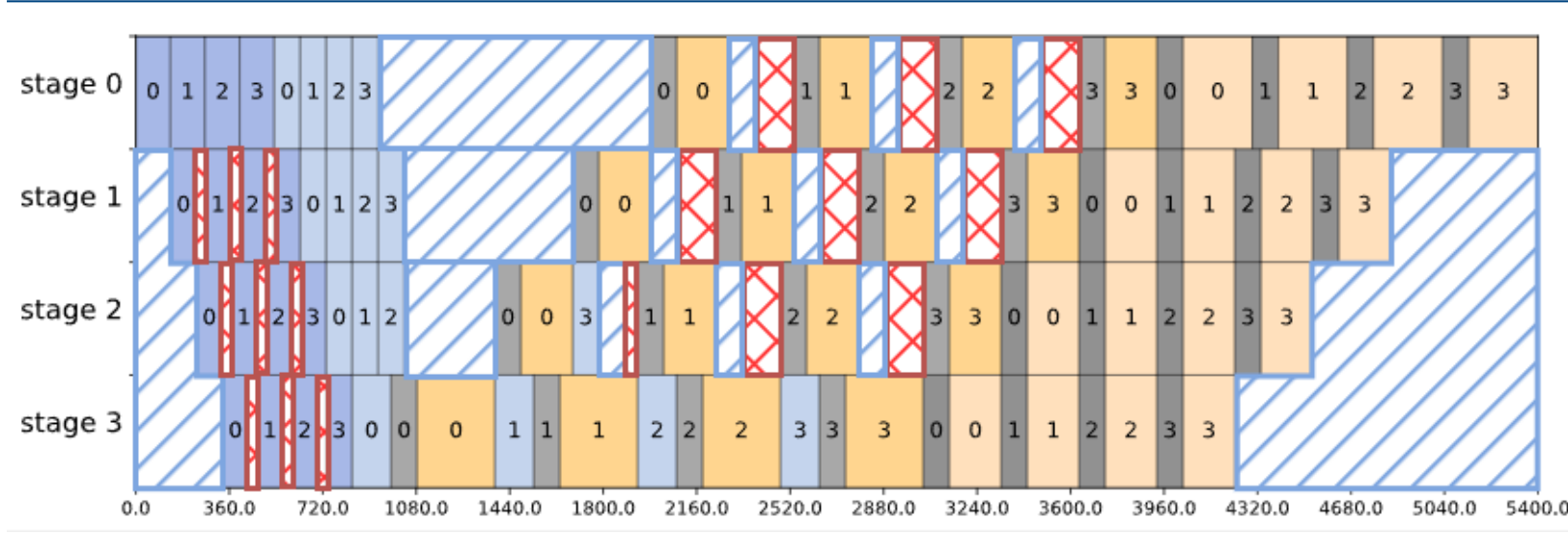


Execution scheduling has progressed from *GPipe* [1]'s fill drain to *PipeDream* [2]'s memory saving 1F1B and *Megatron* [3]'s finer-grained **1F1B interleaved schedule**, squeezing bubbles further with more complexity in the computation and memory behavior of each stage.



Pipeline schedules power large DNN training: pipeline interleave training today's multi-billion-parameter models hinges on pipeline parallelism, which slices the network into stages/chunks and streams μ -sized micro-batches, thus forward and backward passes overlap, trimming per-device memory and latency.

Motivation



The pipeline leaks three kinds of idle “bubbles”:

- recomputation** (gray blocks)
- computation imbalance** (red blocks)
- warm-up/cool-down preparation** (blue blocks)

State-of-the-art planners tackle bubbles piecemeal:

- PipeDream* for imbalance computation
- BPipe* [4] for imbalance activation memory
- Merak* [5] for recompute
- AdaPipe* [6] for recompute and imbalance computation
- Zero-Bubble* [7] for preparation

SOTAs modeling only a subset of the intertwined axes of computation, memory, stages, and chunks. The prior techniques are **orthogonal**, and naively stacking them often leads to negative interference.

A holistic planner must reason about all three bubble categories under a **unify cost model**

System Design

Abstraction:

$$S^* = \begin{bmatrix} S^0 \\ S^1 \\ \vdots \\ S^{P-1} \end{bmatrix} = \begin{bmatrix} [H, l_0^r, l_1^r, \dots] \dots [l_{j_0}^r, l_{j_0+1}^r, \dots] \\ [l_{i_1}^r, l_{i_1+1}^r, \dots] \dots [l_{j_1}^r, l_{j_1+1}^r, \dots] \\ \vdots \\ [l_{i_{P-1}}^r, \dots] \dots [\dots, l_{L-1}^r, T] \end{bmatrix}$$

Cost Model:

$$T(S^x) = FP(S^x) + BP(S^x)$$

$$M(S^x) = M_S(S^x) + M_D(S^x)$$

Bubble Overhead:

$$(B_R, B_I, B_P) = \mathcal{F}(FP(S^x), BP(S^x))$$

Constraints:

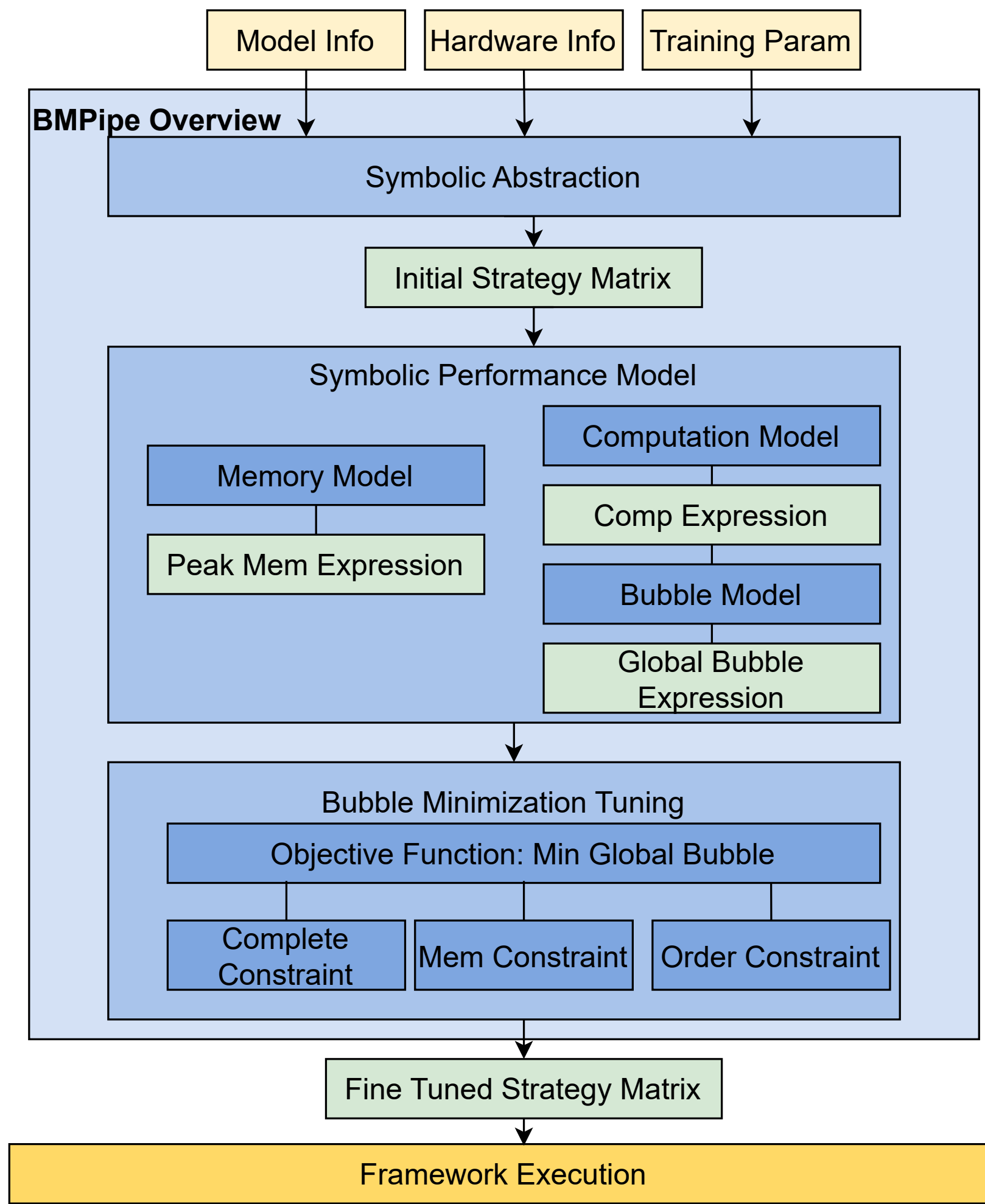
$$(C_i) = \mathcal{G}(FP(S^x), BP(S^x), M_S(S^x), M_D(S^x))$$

Algorithm 1: *Our planner*

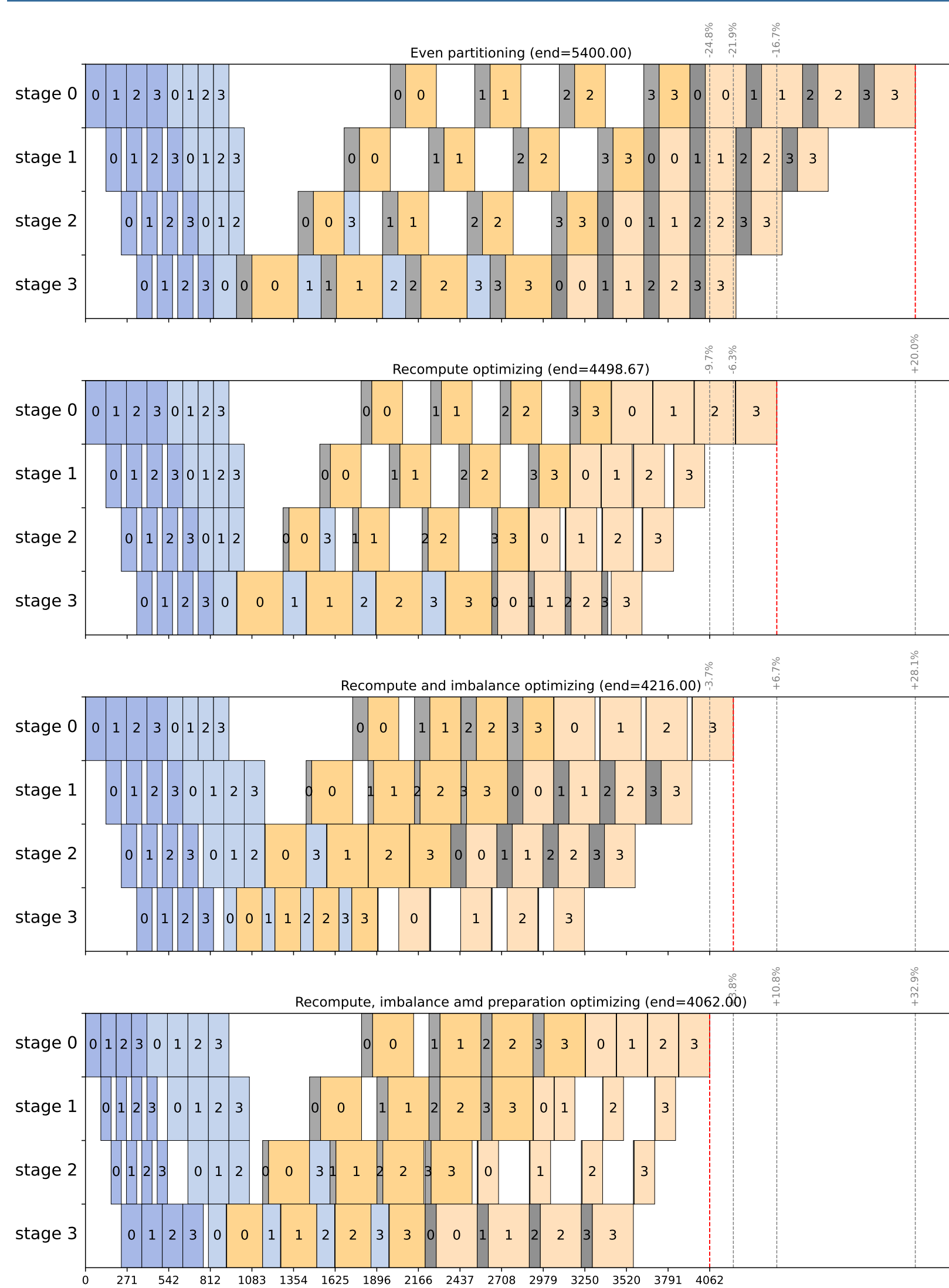
Input: Layer set $\mathcal{L}$; recompute set $\mathcal{R}$; stages P ; chunks C ; device memory D_M ; micro-batches μ

Output: Optimal assignment matrix S^*

```
// Phase 1: cost pre-computation
1 foreach distinct  $t \in \mathcal{T}$  do
2    $(FP, BP) \leftarrow \text{GetCompCost}(t)$ ;  $M_S \leftarrow \text{GetMemCost}(t)$ ;
3 foreach distinct  $(t, r) \in \mathcal{T} \times \mathcal{R}$  do
4    $Recom \leftarrow \text{GetCompCost}(t, r)$ ;  $M_A \leftarrow \text{GetMemCost}(t, r)$ ;
// Phase 2: ILP build
5 define  $l_{t,r,c,x} \in \mathbb{Z}_{\geq 0}$ ,  $b_{t-1,c,x} \in \{0, 1\}$ ;
6 minimize  $B_G = B_R + B_I + B_P$ ;
7 add constraints C1, C2, C3;
// Phase 3: solve
8 solve ILP with Gurobi; obtain  $l_{t,r,c,x}^*$ ,  $b_{t-1,c,x}^*$ ;
// Phase 4: build  $S^*$ 
9 foreach  $(t, r, c, x)$  do
10   assign  $l_{t,r,c,x}^*$  layers of type  $t$  to chunk  $c$  of stage  $x$  with strategy  $r$ ;
11 return  $S^*$ 
```



Running Example

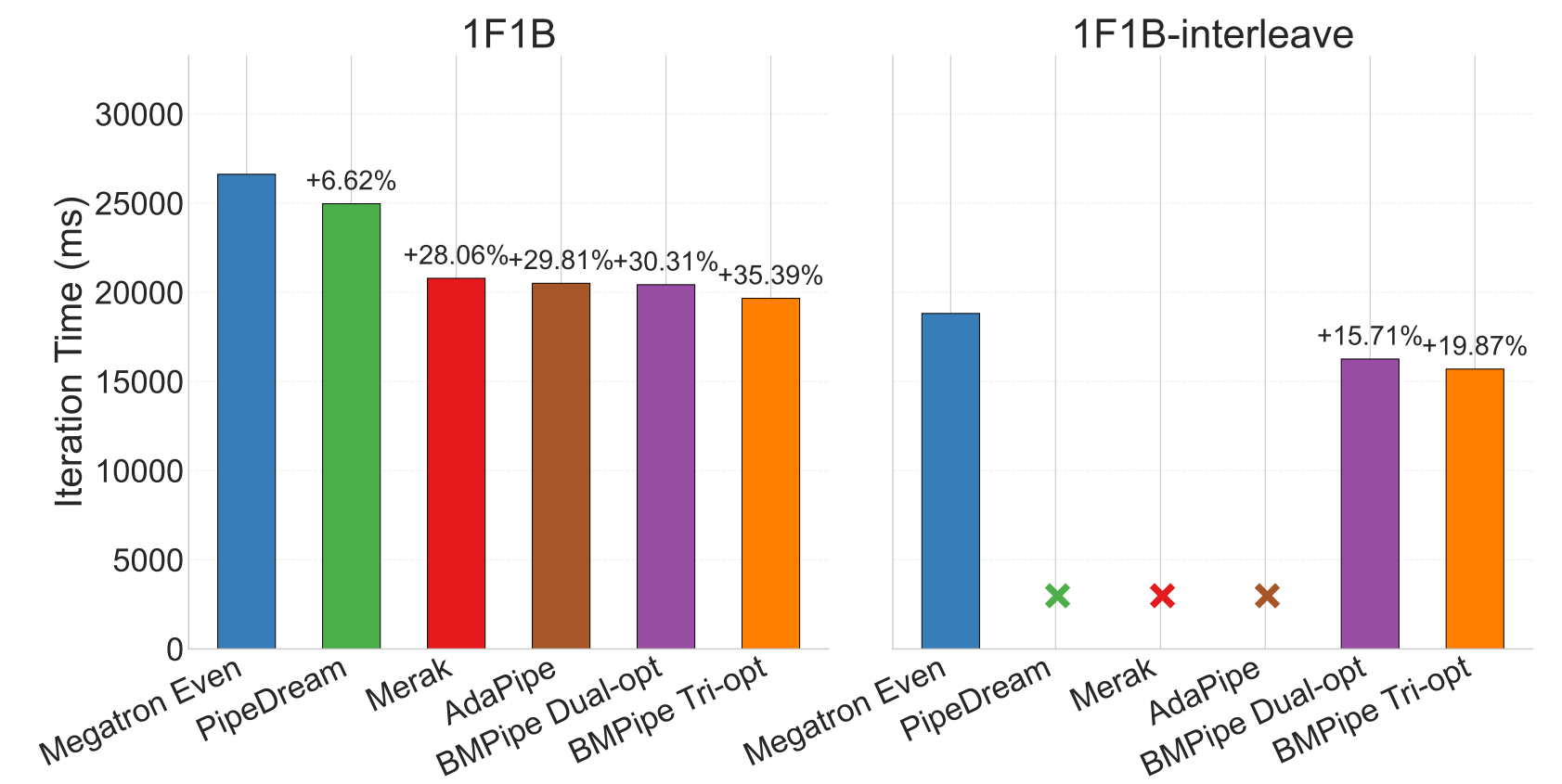


Evaluation

End-to-End performance

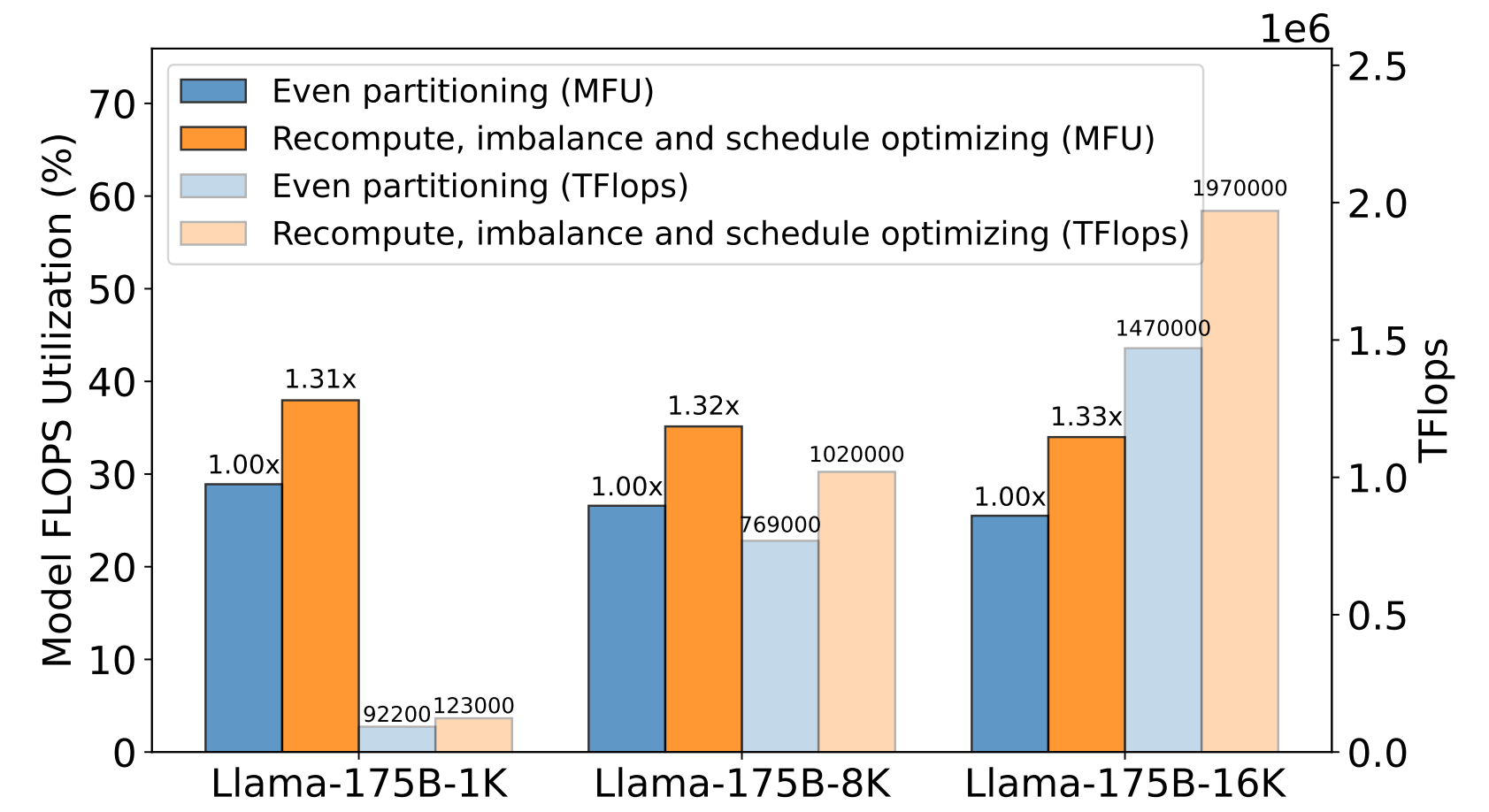
Experimental setup:

Bench-marked on a 16384× Ascend-910 NPU cluster (CANN 7.1, MindSpore 2.3) across 6 SOTA models: LLaMA-70/175B, BLOOM-176B, Qwen2-72B, Pangu- α -230B, LLaMA-MoE-1.04T, Multi-modal model-70B, against *Megatron-Even*, *PipeDream*, *Merak*, *AdaPipe*.



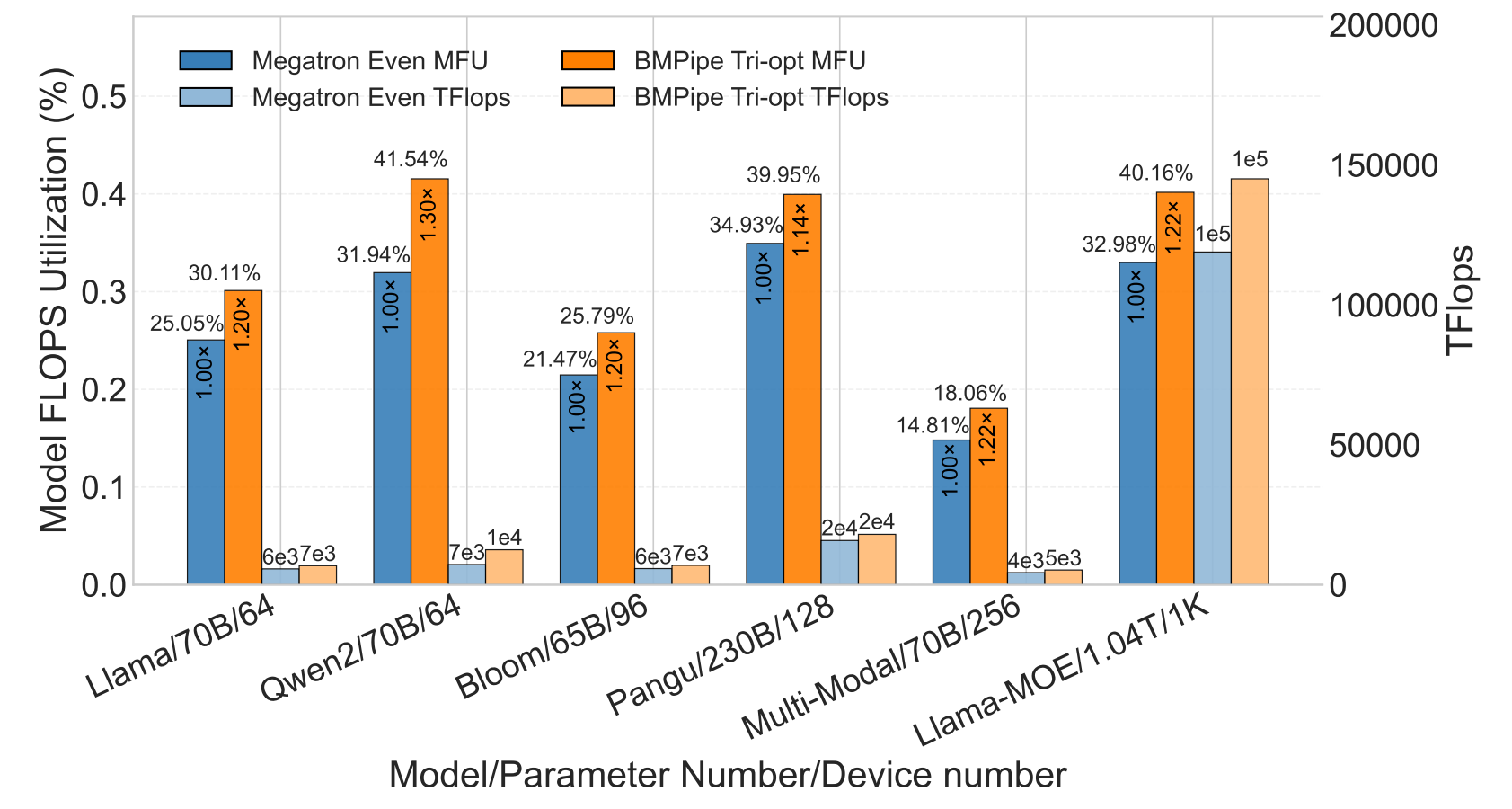
Very-large DNN model and cluster evaluation (LLaMA-175B @ 16k devices):

- Vanilla 1F1B*: *PipeDream* 1.07 ×, *Merak* 1.29 ×, *AdaPipe* 1.30 ×, *Our Solution* dual/tri 1.31 × / 1.36 × versus *Megatron-Even* (up to 36 % faster).
- 1F1B-Interleaved*: *Megatron-Even* is 1.41, × over vanilla; *Our Solution-Dual* adds 1.16 ×, tri 1.20 ×;
- 1.70 × *Our Solution* on 1F1B-Interleaved vs. *Megatron-Even* on vanilla.



Scalability:

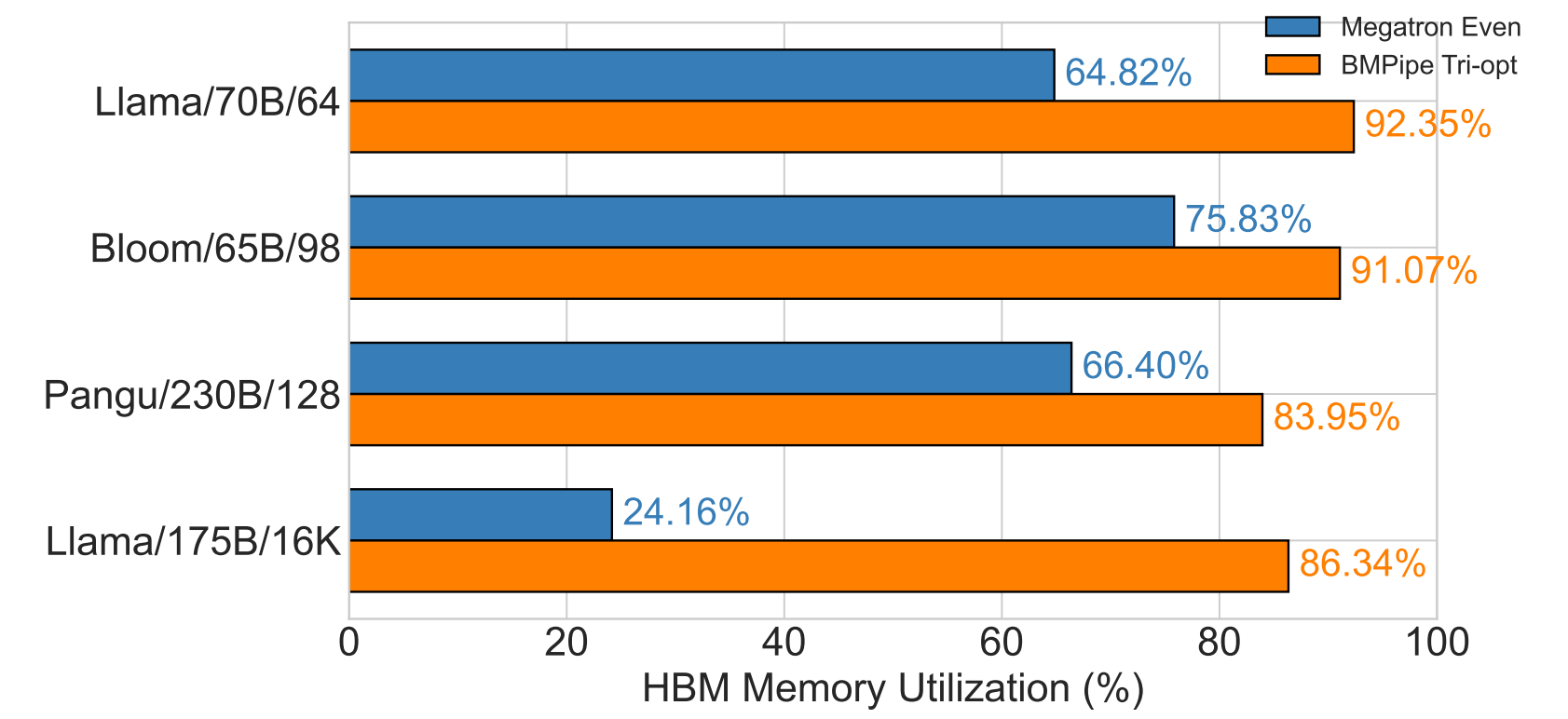
As device count scales $1,024 \rightarrow 8,192 \rightarrow 16,384$, *Our Solution-Tri* sustains $1.18 - 1.20 \times$ speed-up; MFU climbs $28.4\% \rightarrow 37.9\%$.



Generality:

The MFU of six SOTA models increases by $1.20 - 1.31 \times$.

Memory utilization



Memory Efficiency:

On 64–128-device runs, *Megatron-Even* taps only 64 – 75 % HBM; *Our Solution-Tri* raises this to 83 – 92 %, which equates to an effective capacity gain of $1.20 - 1.42 \times$.

References

- Y. Huang et al., "Gpipe: Efficient training of giant neural networks using pipeline parallelism," in *Proceedings of the 33rd International Conference on Neural Information Processing Systems*, Red Hook, NY, USA: Curran Associates Inc., 2019.
- D. Narayanan et al., "Pipedream: Generalized pipeline parallelism for dnn training," in *Proceedings of the 27th ACM Symposium on Operating Systems Principles*, ser. SOSP '19, Huntsville, Ontario, Canada: Association for Computing Machinery, 2019, pp. 1–15, isbn: 9781450368735. doi: 10.1145/3341301.3359646.
- V. A. Korthikanti et al., "Reducing activation recomputation in large transformer models," in *Proceedings of Machine Learning and Systems*, D. Song et al., Eds., vol. 5, Curan, 2023, pp. 341–353.
- T. Kim et al., "BPipe: Memory-balanced pipeline parallelism for training large language models," in *Proceedings of the 40th International Conference on Machine Learning*, A. Krause et al., Eds., ser. Proceedings of Machine Learning Research, vol. 202, PMLR, Jul. 2023, pp. 16 639–16 653.
- Z. Lai et al., "Merak: An efficient distributed dnn training framework with automated 3d parallelism for giant foundation models," *IEEE Trans. Parallel Distrib. Syst.*, vol. 34, no. 5, pp. 1466–1478, May 2023, issn: 1045-9219. doi: 10.1109/TPDS.2023.3247001.
- Z. Sun et al., "Adapipe: Optimizing pipeline parallelism with adaptive recomputation and partitioning," in *Proceedings of the 29th ACM International Conference on Architectural Support for Programming Languages and Operating Systems*, Volume 3, ser. ASPLOS '24, La Jolla, CA, USA: Association for Computing Machinery, 2024, pp. 86–100, isbn: 9798400703867. doi: 10.1145/3620666.3651359.
- P. Qi et al., "Zero bubble (almost) pipeline parallelism," in *The Twelfth International Conference on Learning Representations*, 2024.