FlashAttention-3: Optimizing FlashAttention for H100 GPUs

Jay Shah*, Ganesh Bikshandi*, Ying Zhang, Vijay Thakkar, Pradeep Ramani, Tri Dao

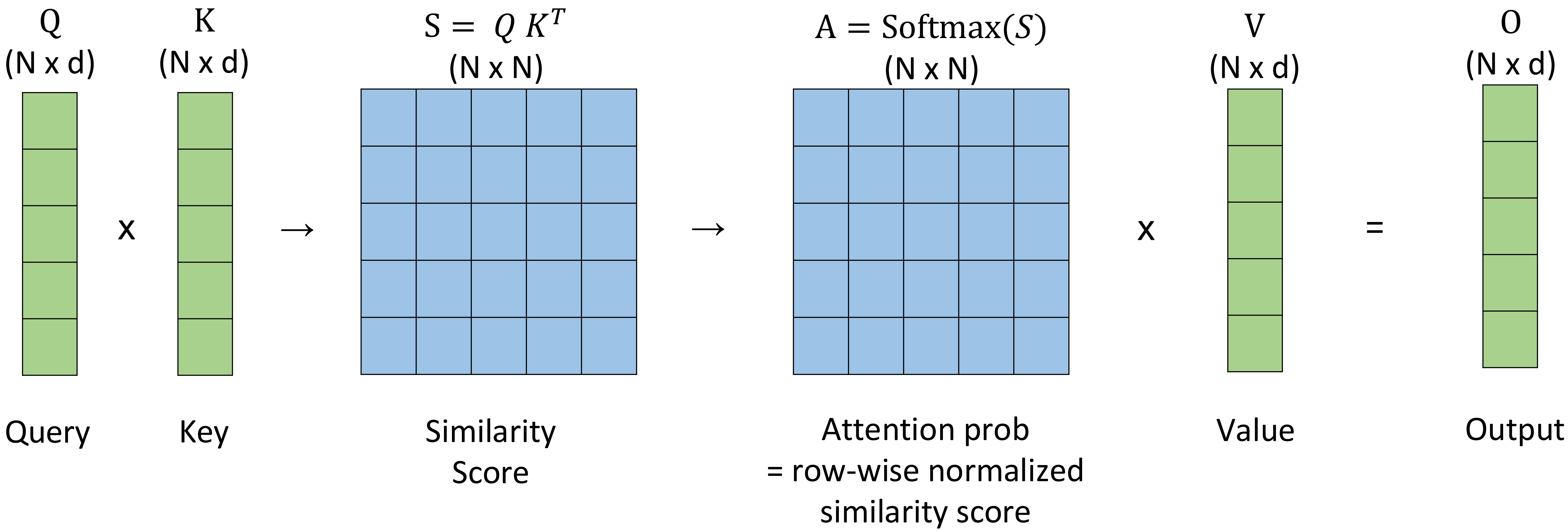
1. **New Hopper Instructions**
   - **WGMMA**: higher throughput
   - **TMA**: faster loading from gmem <-> smem, saves registers

2. **Asynchrony**
   - Builds on asynchronous wgmma and TMA
   - Inter-warpgroup overlapping: warp-specialization, pingpong
   - Intra-warpgroup overlapping: softmax and async matmul

3. **Low-precision** – FP8: layout conformance, incoherent processing

Upshot: **1.6-3x** speedup, up to 85% utilization with BF16, 1.3 PFLOPS with FP8

Background: Attention Mechanism



Typical sequence length N: 1K – 8K
Head dimension d: 64 – 128

$$\text{Softmax}([s_1, \cdots, s_N]) = \left[\frac{e^{s_1}}{\sum_i e^{s_i}}, \cdots, \frac{e^{s_N}}{\sum_i e^{s_i}}\right]$$

$$O = \text{Softmax}(QK^T)V$$

Attention scales quadratically in sequence length N

How FlashAttention Reduced HBM Reads/Writes: Compute by Blocks

Challenges:

(1) Compute softmax normalization without access to full input.

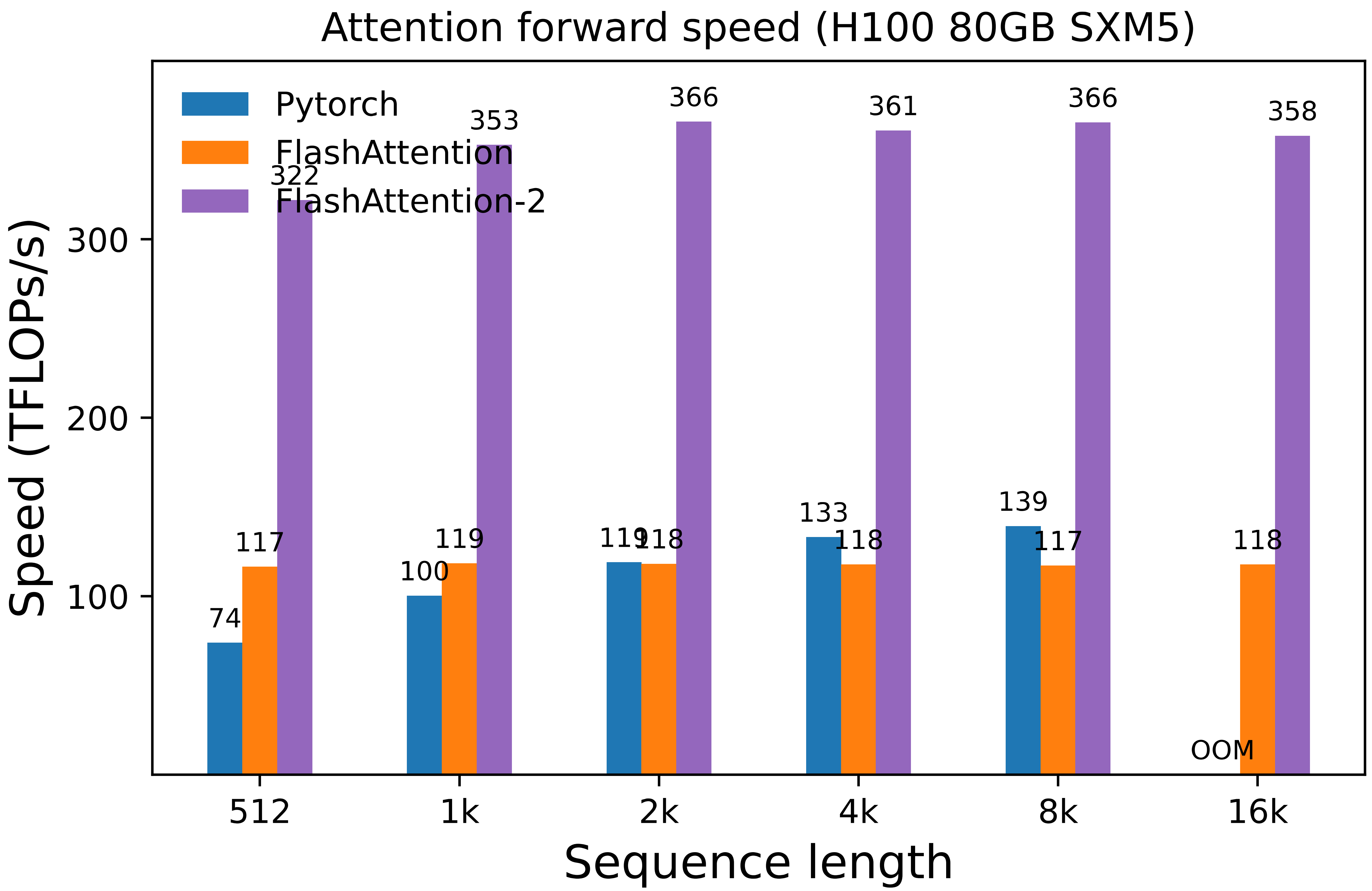
(2) Backward without the large attention matrix from forward.

Approaches:

(1) Tiling and online softmax: Restructure algorithm to load block by block from HBM to SRAM to compute attention.

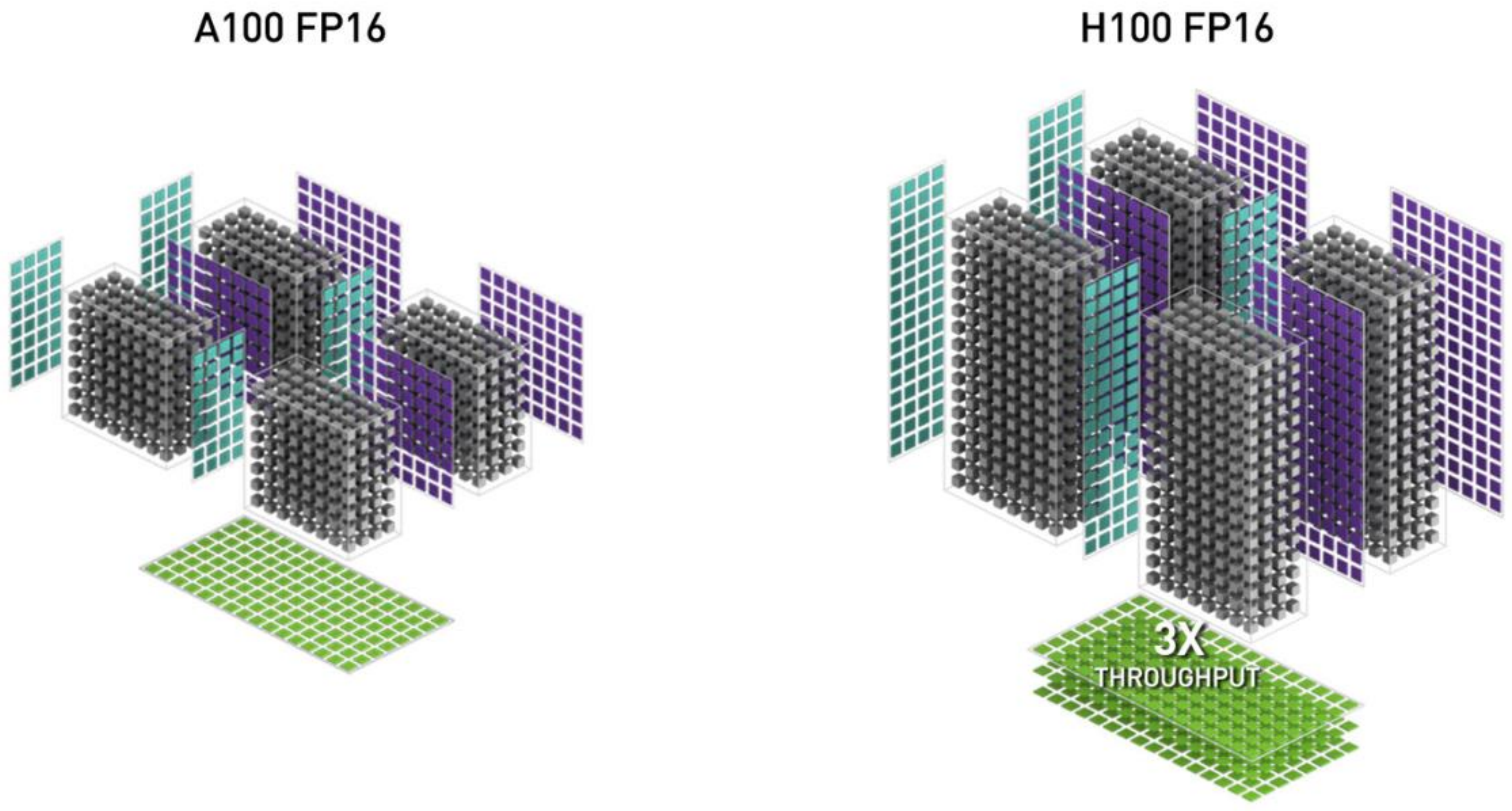
(2) Recomputation: Don't store attn. matrix from forward, recompute it in the backward.

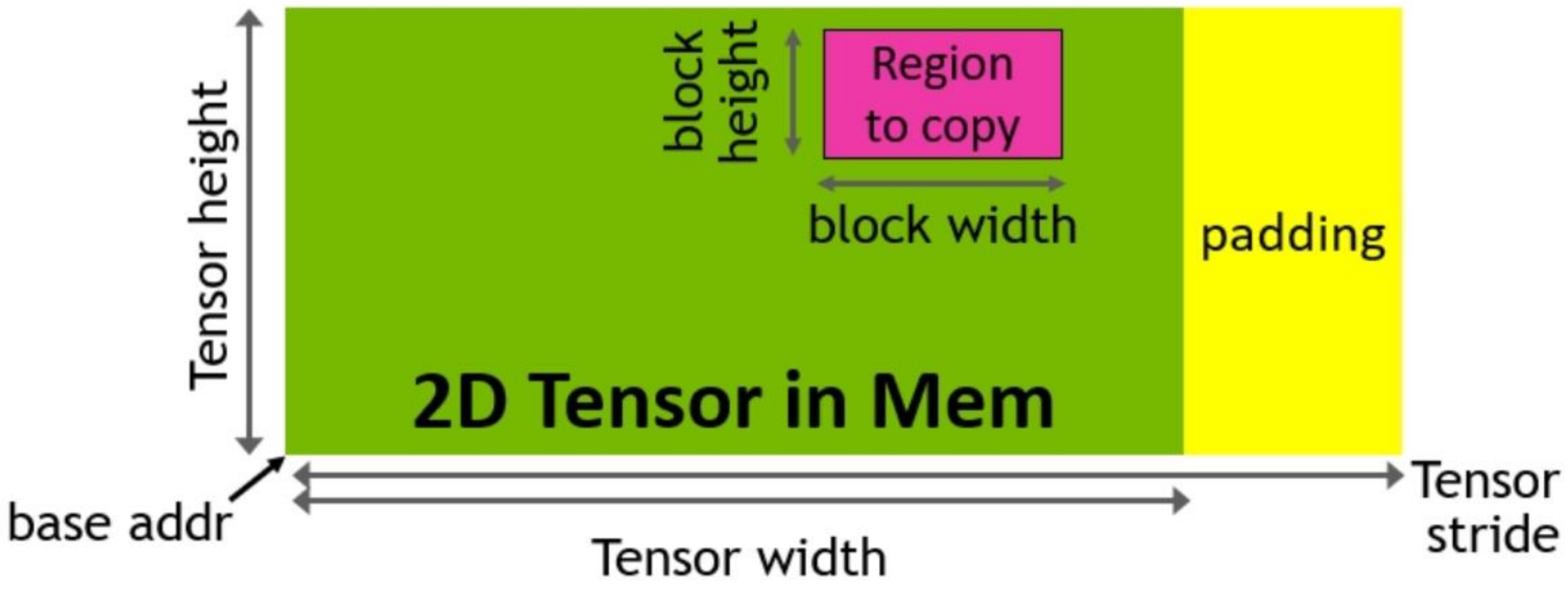
Challenge: Optimizing FlashAttention for Modern Hardware - H100



FA2 only gets to 35-40% utilization (no WGMMA, no TMA)

New Instructions: WGMMA (Warpgroup MMA) & TMA





wgmma uses 4 warps (= 1 warpgroup) and is necessary to reach peak throughput on H100.

TMA: accelerate gmem -> smem, saves registers as TMA is issued by a single thread

WGMMA and TMA integrate into a warp-specialized pipelined design for both GEMM and attention.

Asynchrony: Overlapping GEMM and Softmax

Why overlapping?

Example: headdim 128, block size 128 x 192

FP16 WGMMA: 2 x 2 x 128 x 192 x 128 = 12.6 MFLOPS, 4096 FLOPS/cycle -> **3072 cycles**

MUFU.EX2: 128 x 192 = 24.6k OPS, 16 OPS/cycle -> **1536 cycles**

MUFU.EX2 takes 50% the cycles of WGMMA.

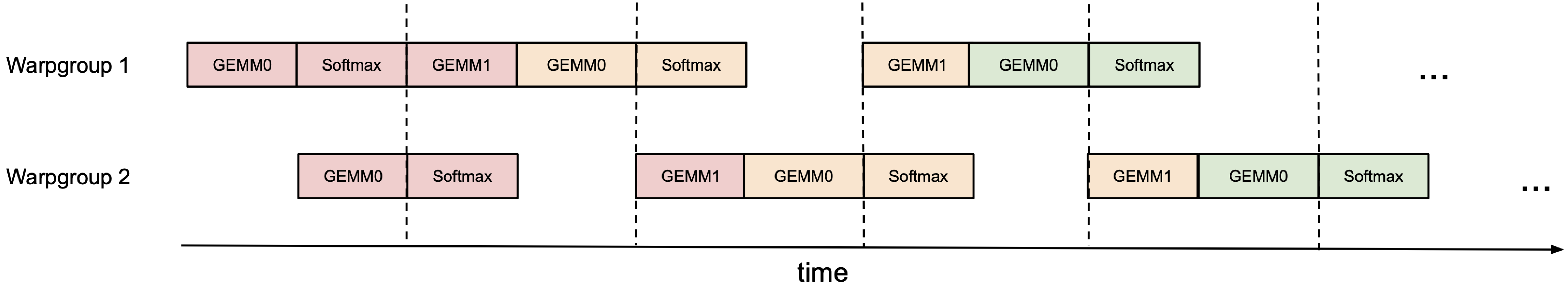
FP8 is even worse: WGMMA and MUFU.EX2 both take 1536 cycles!

We want to be doing EX2 while tensor cores are busy with WGMMA.

Inter-warpgroup Overlapping of GEMM and Softmax

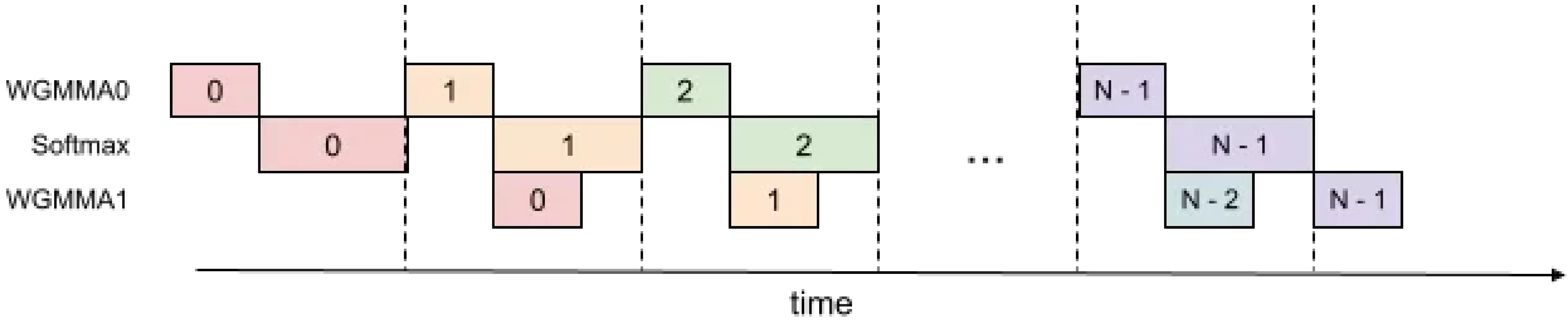
Easy solution: leave it to the scheduler!

This works reasonably well, but we can do better



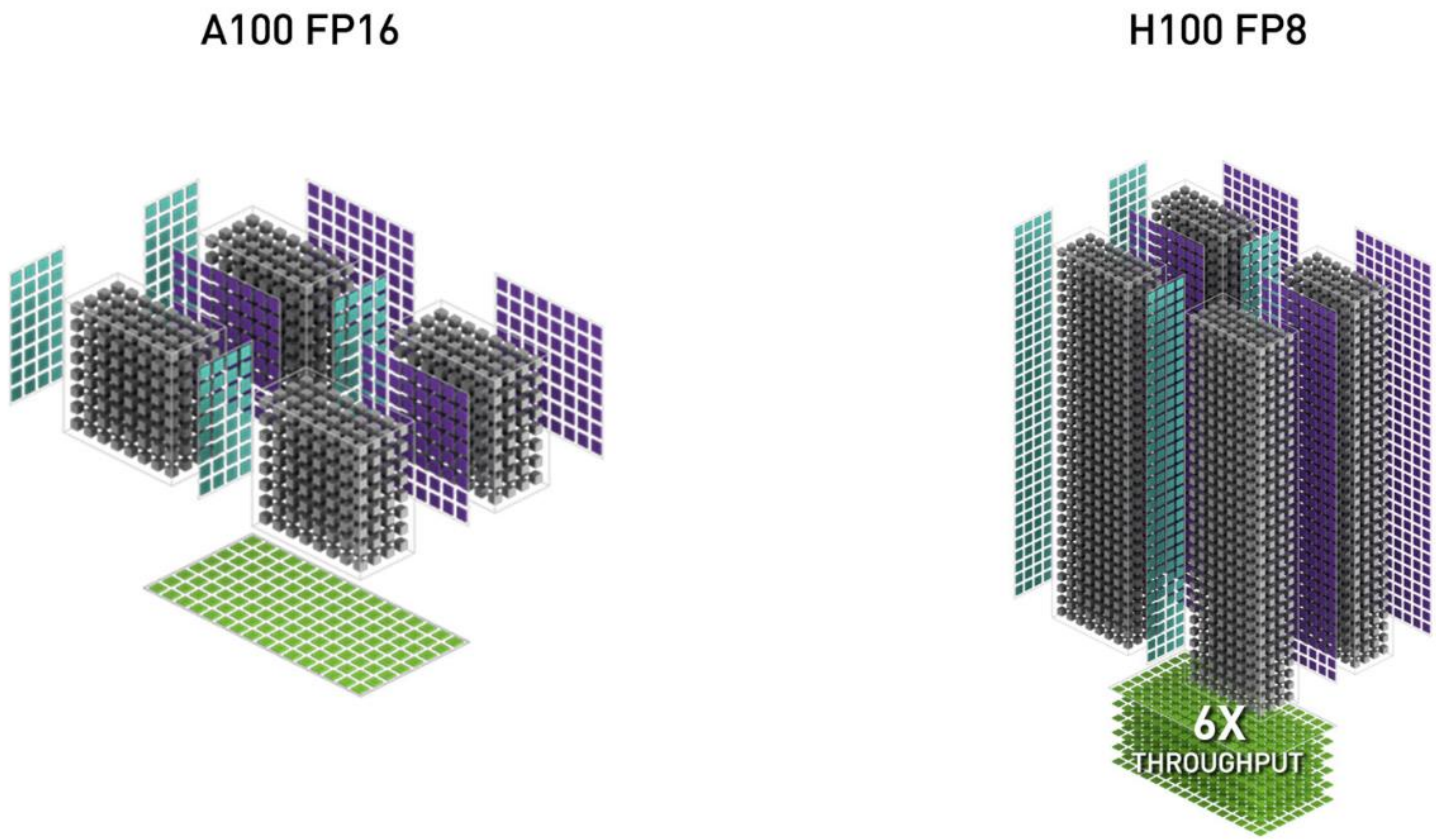
Pingpong scheduling with synchronization barriers (bar.sync):
580 TFLOPS -> 640 TFLOPS

Intra-warpgroup Overlapping of GEMM and Softmax



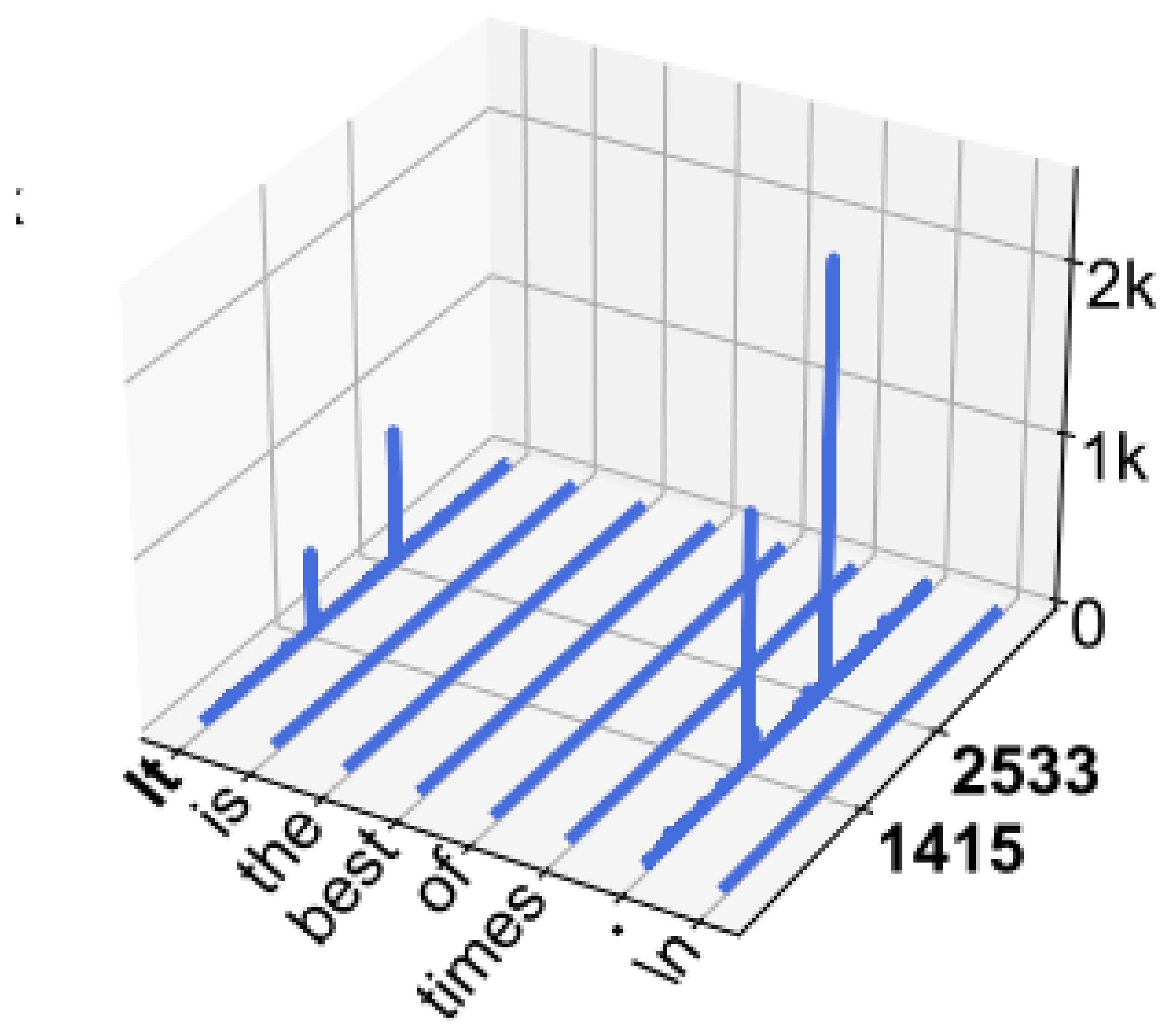
2-stage pipelining: 640 TFLOPS -> 670 TFLOPS (but higher register pressure)

Low-precision: FP8



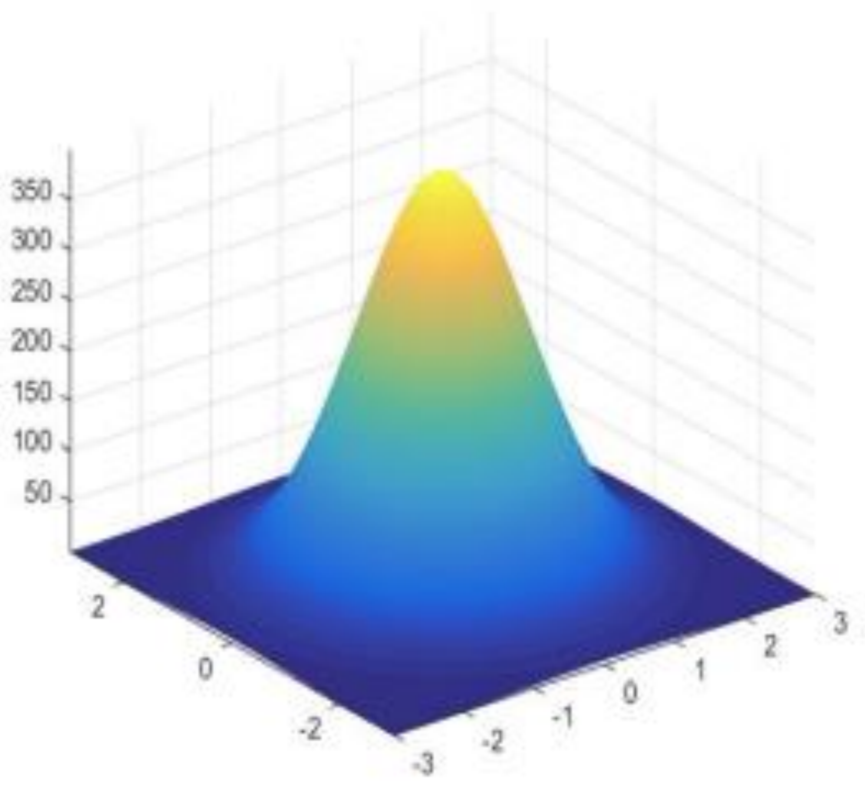
FP8 doubles WGMMA throughput, but trades off accuracy

Incoherent Processing to Smooth out Outlier Features



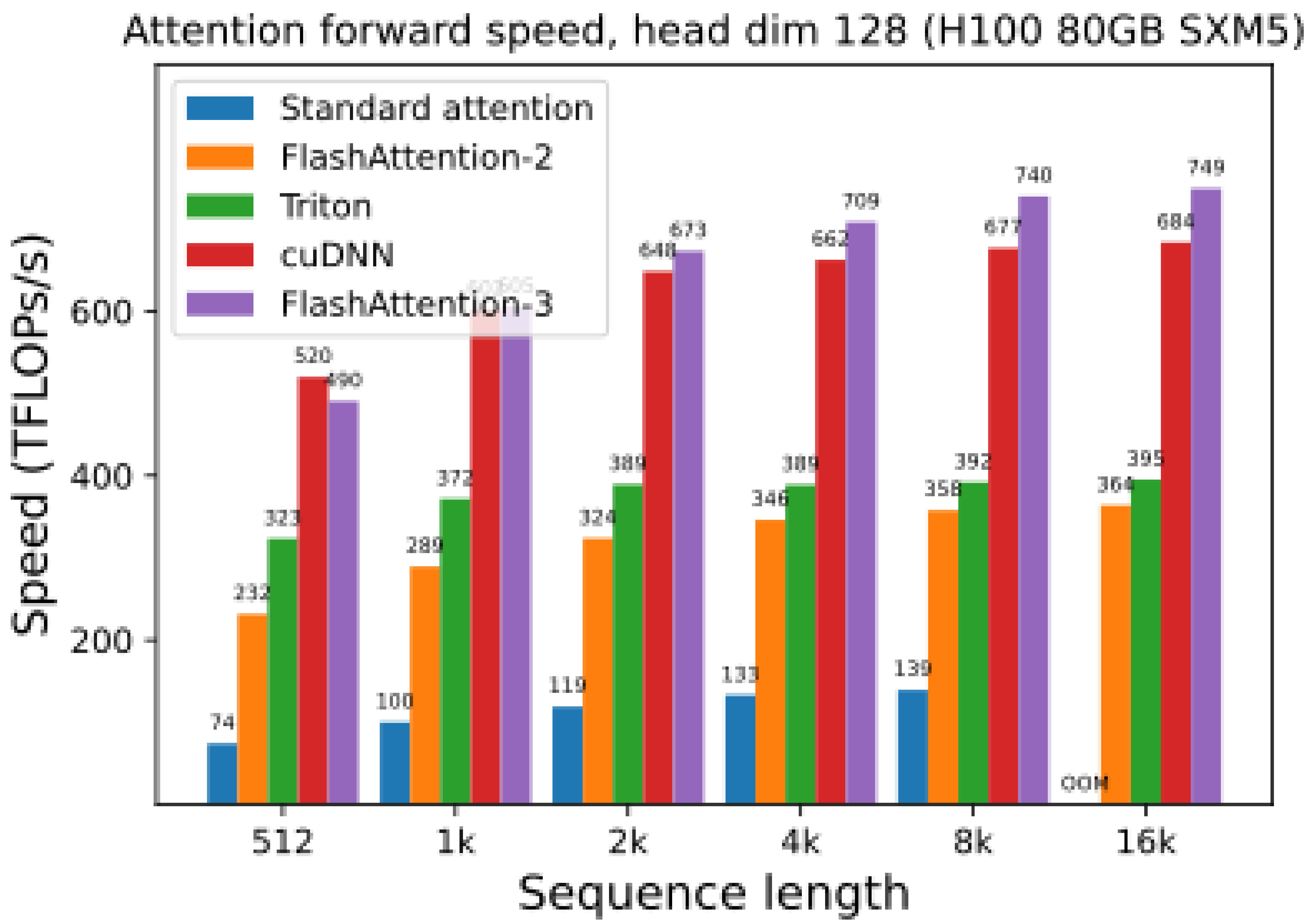
For random orthogonal matrix M (where M M^T = I):
Q -> QM -> quantize(QM)
K -> KM -> quantize(KM)
Dot product QK^T is preserved, but outliers are ”spread out”

Hadamard Transform

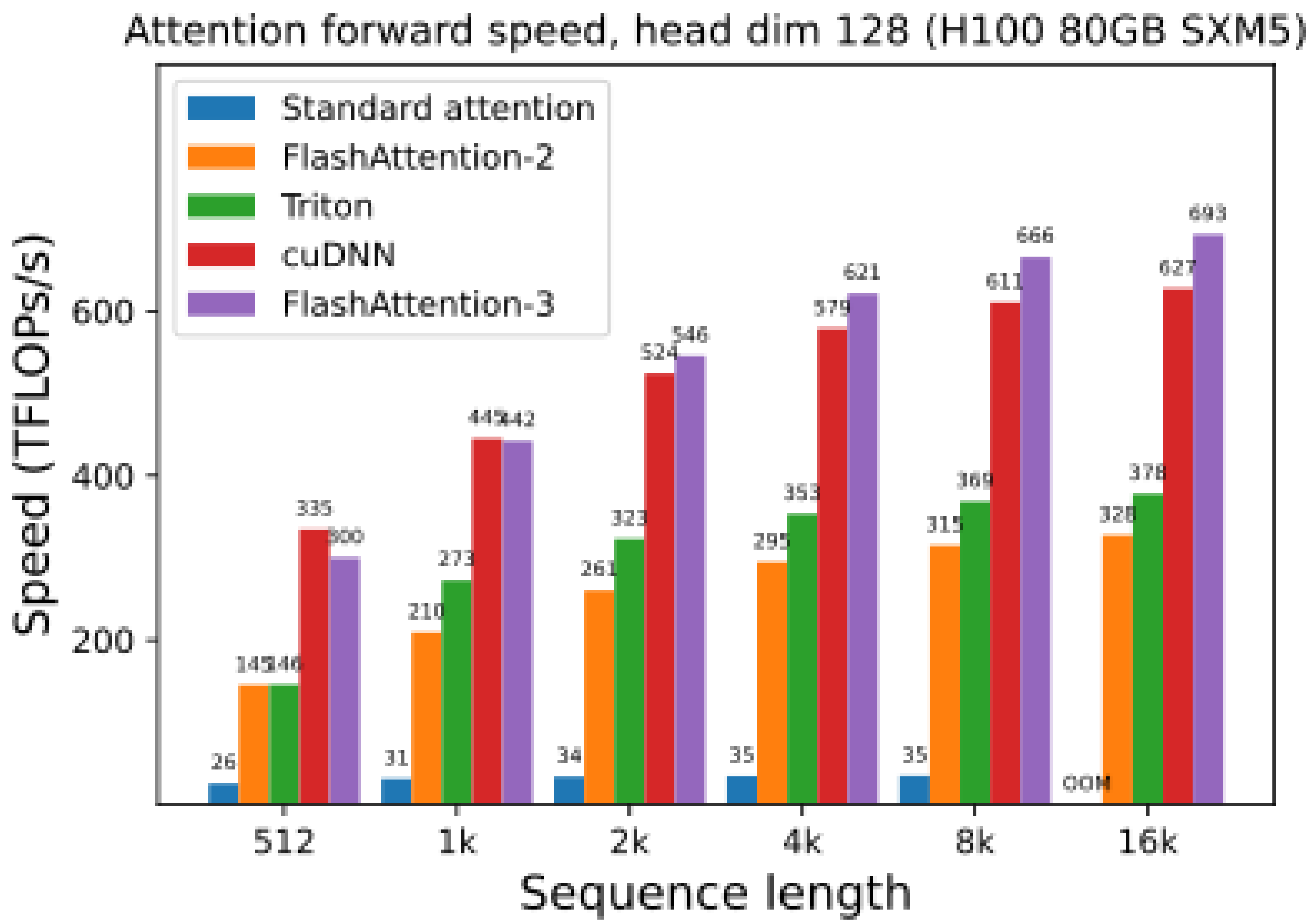


Fast transform (O(d log d), not O(d^2), can be fused with rotary embedding “for free”

BF16 Benchmark: 1.6-2.0x speedup

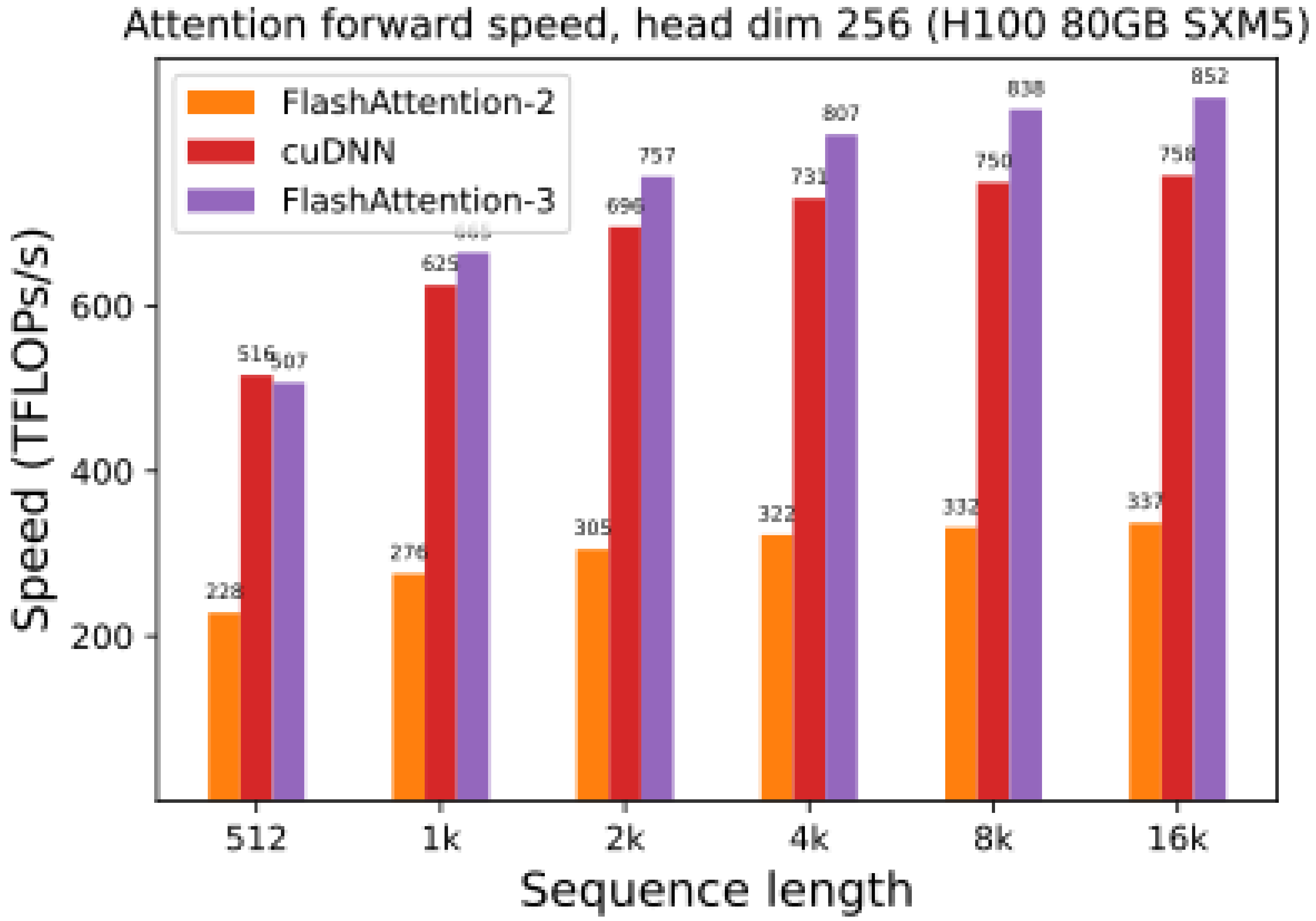


Without causal mask

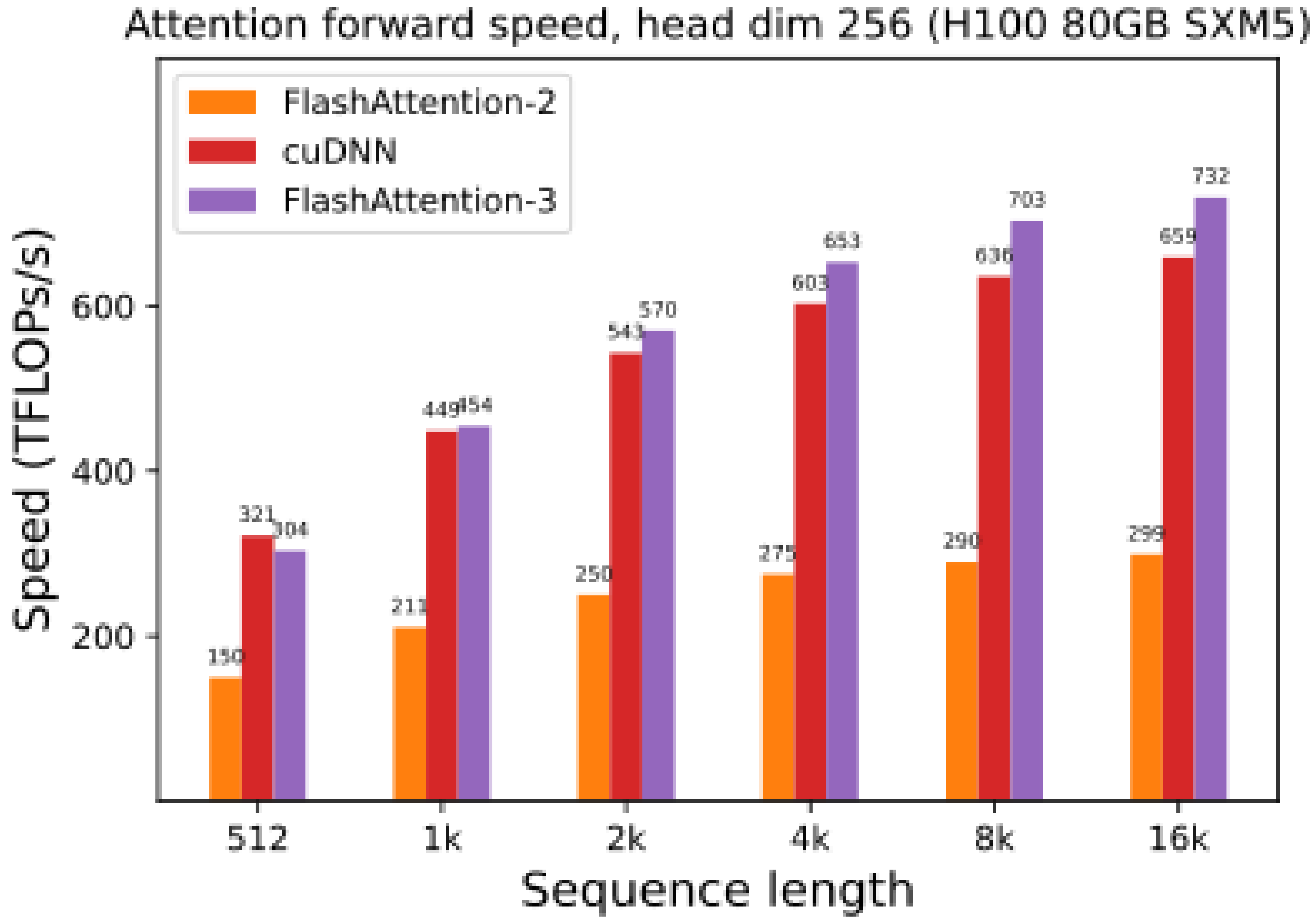


With causal mask

BF16 Benchmark: reach up to 850 TFLOPS

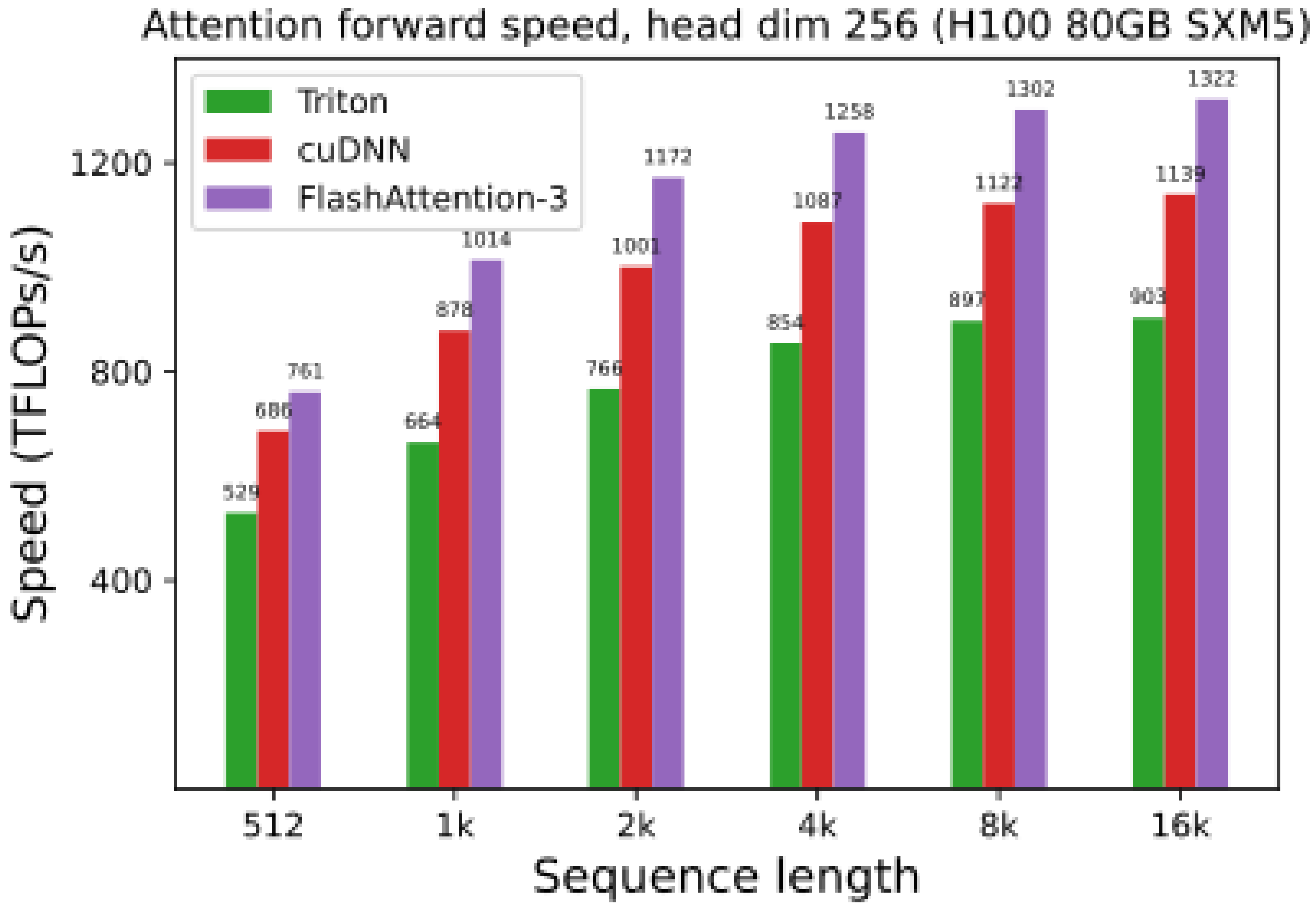


Without causal mask

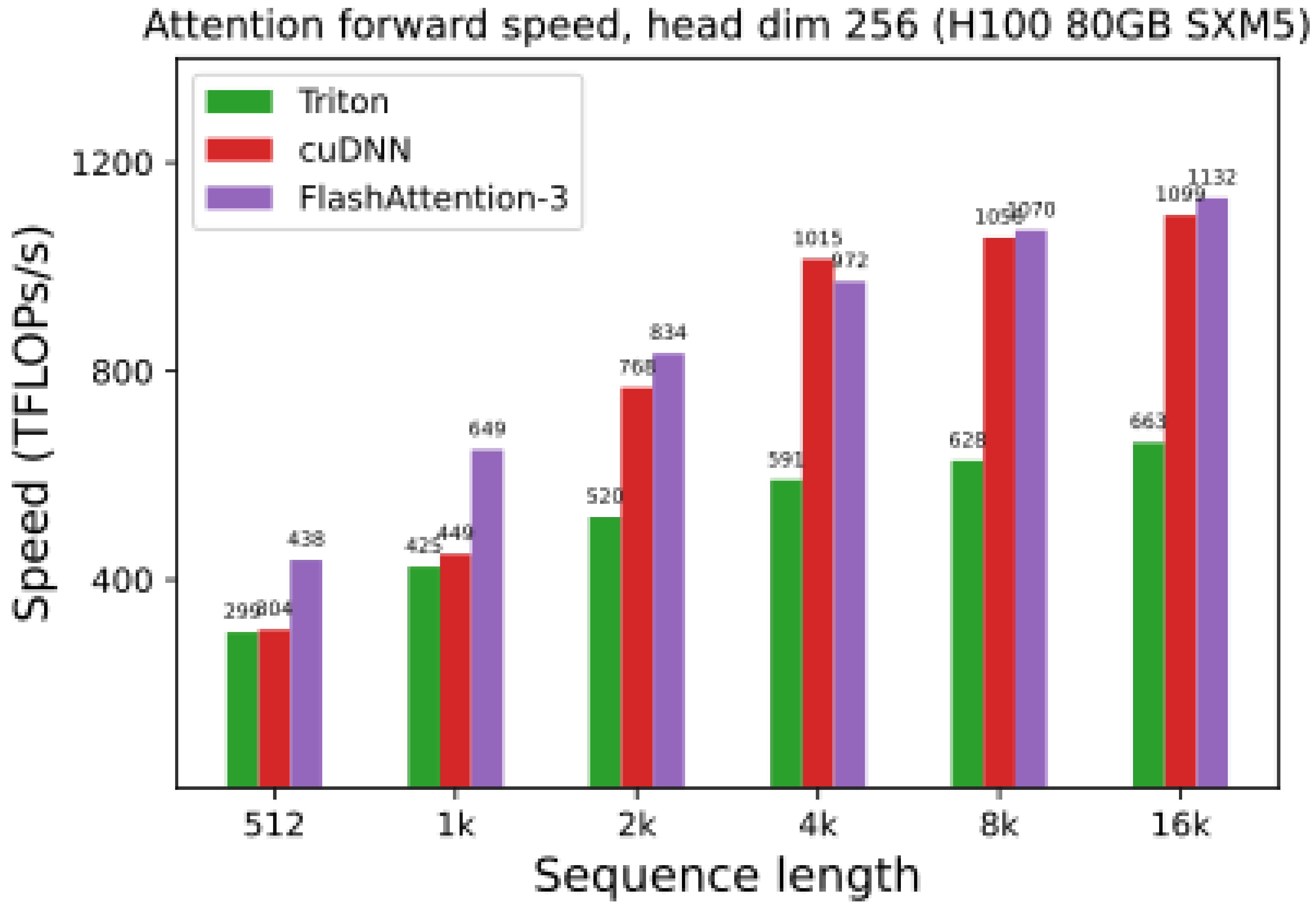


With causal mask

FP8 Benchmark: up to 1.3 PFLOPS



Without causal mask



With causal mask

Summary – FlashAttention-3

Fast and **accurate** attention optimized for modern hardware

Key algorithmic ideas: **asynchrony**, **low-precision**

Upshot: **faster** training, **better** models with **longer** sequences

Code: https://github.com/Dao-AILab/flash-attention