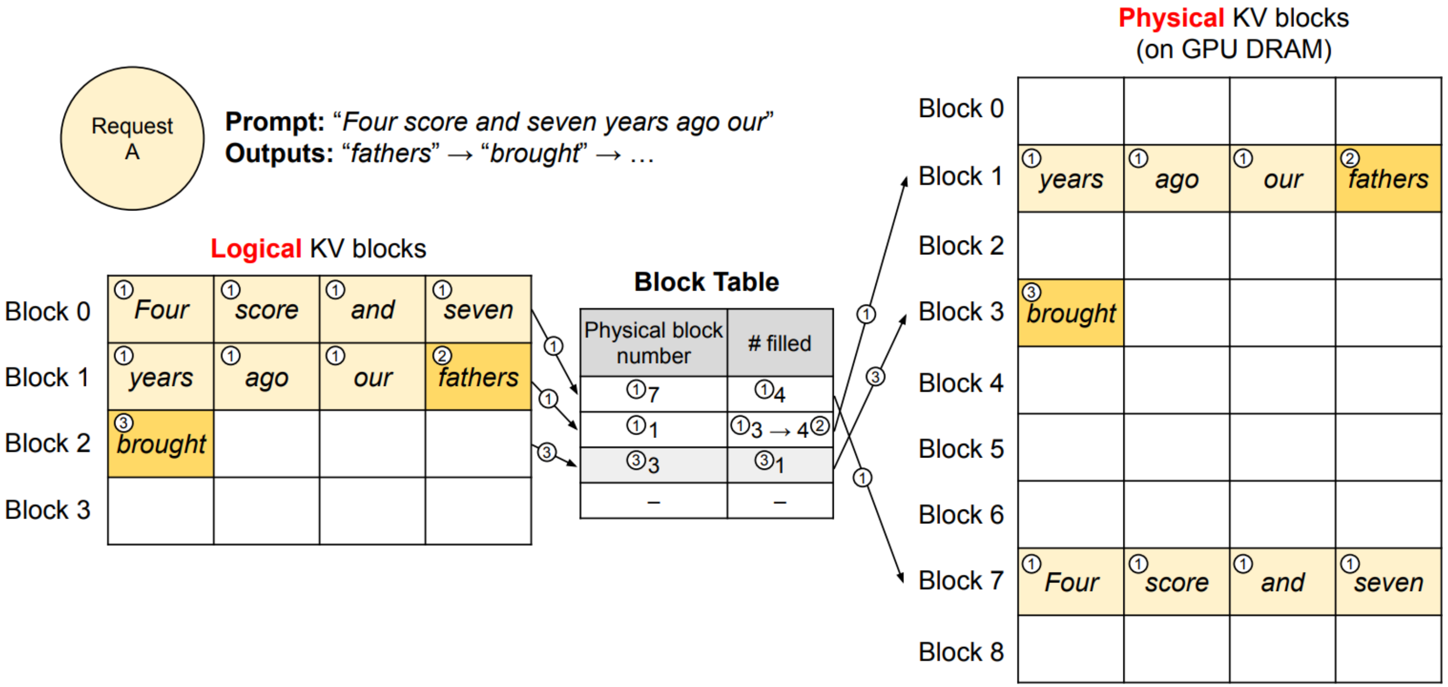
FlashInfer

INTRODUCTION TO LLM INFERENCE SERVING SYSTEMS

CHUHONG YUAN

Review of PageAttention



Prompt: “*Four score and seven years ago our*”
Outputs: “*fathers*” → “*brought*” → …

Logical KV blocks

Block 0	① *Four*	① *score*	① *and*	① *seven*
Block 1	① *years*	① *ago*	① *our*	② *fathers*
Block 2	③ *brought*			
Block 3				

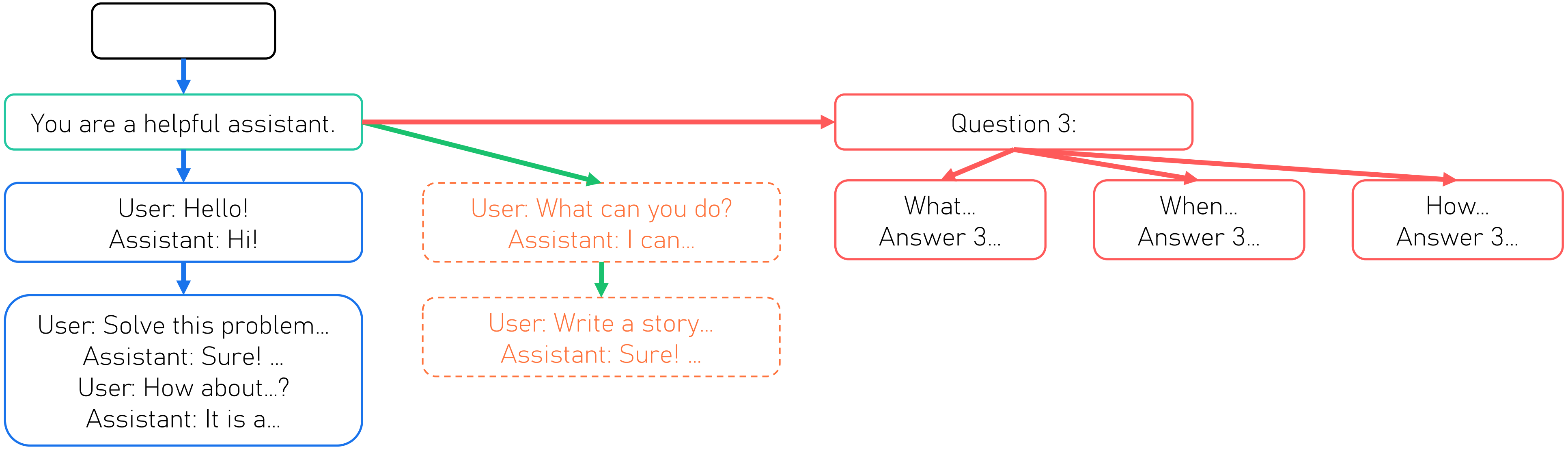
Block Table

Physical block number	# filled
①7	①4
①1	①3 → 4②
③3	③1
–	–

Physical KV blocks
(on GPU DRAM)

Block 0				
Block 1	① *years*	① *ago*	① *our*	② *fathers*
Block 2				
Block 3	③ *brought*			
Block 4				
Block 5				
Block 6				
Block 7	① *Four*	① *score*	① *and*	① *seven*
Block 8				

Review of RadixAttention



You are a helpful assistant.

User: Hello!
Assistant: Hi!

User: Solve this problem...
Assistant: Sure! ...
User: How about...?
Assistant: It is a...

User: What can you do?
Assistant: I can...

User: Write a story...
Assistant: Sure! ...

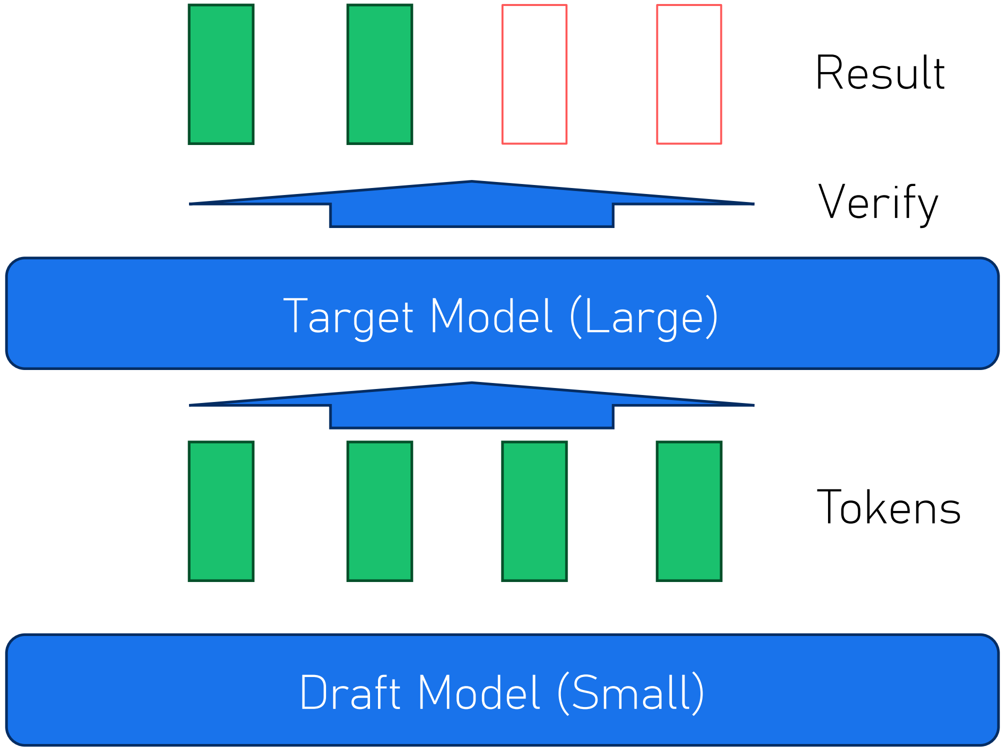
Question 3:

What...
Answer 3...

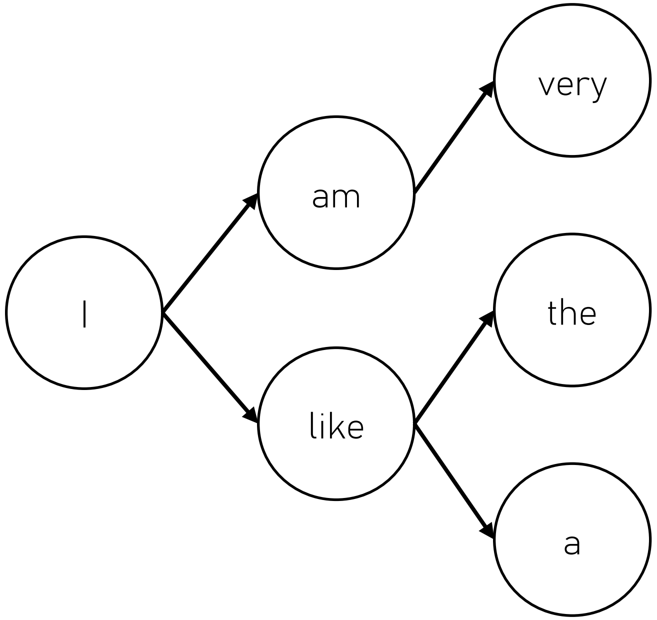
When...
Answer 3...

How...
Answer 3...

Background – Speculative Decoding



Background – Speculative Decoding



Reference: https://www.youtube.com/watch?v=iOLBJwENuvA

Background – Sparse Attention

	K1	K2	K3
Q4	√	√	√
Q5	√	√	√
Q6	√	√	√

Background – Sparse Attention

	K1	K2	K3
Q4	√	√	√
Q5	√	√	√
Q6	√	√	√

	K1	K2	K3
Q4	√	√	
Q5		√	√
Q6	√		

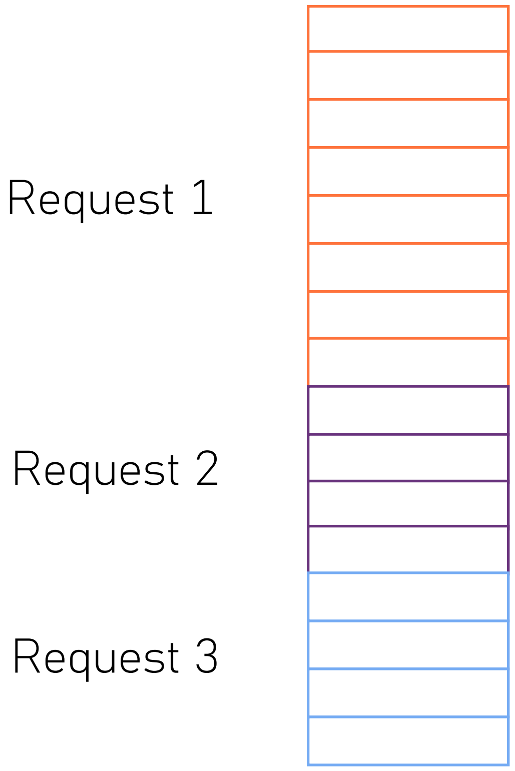
Background – KV Cache Sparsity

	I	am	like	the
I	√			
am	√	√		
like	√		√	
the	√	√		√

Block-Sparse Matrix

- Still use a paged KV Cache management
- Allow sparsely selecting the K/V values to compute
- Organize the data into blocks to fit the size of tensor cores
- The block size can be customized

Block-Sparse Matrix - Usage

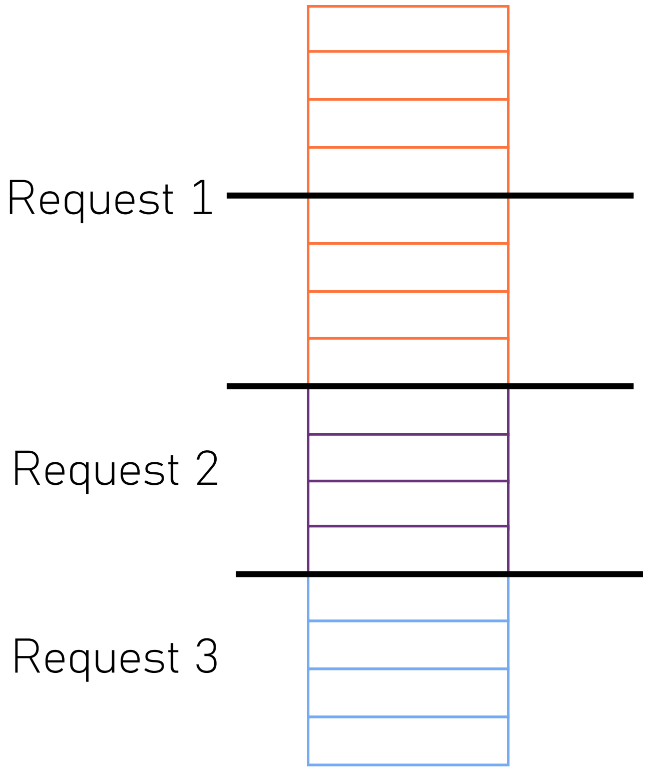


Request 1

Request 2

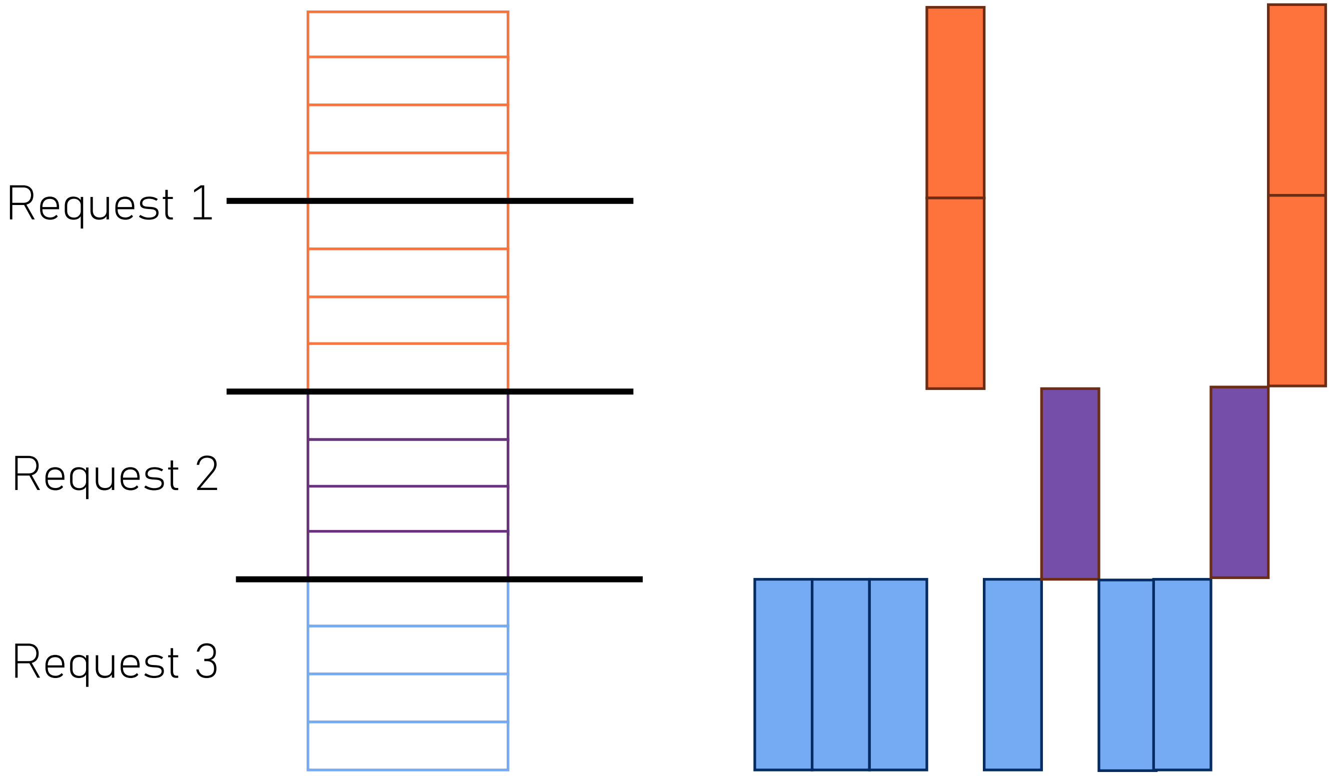
Request 3

Block-Sparse Matrix - Usage



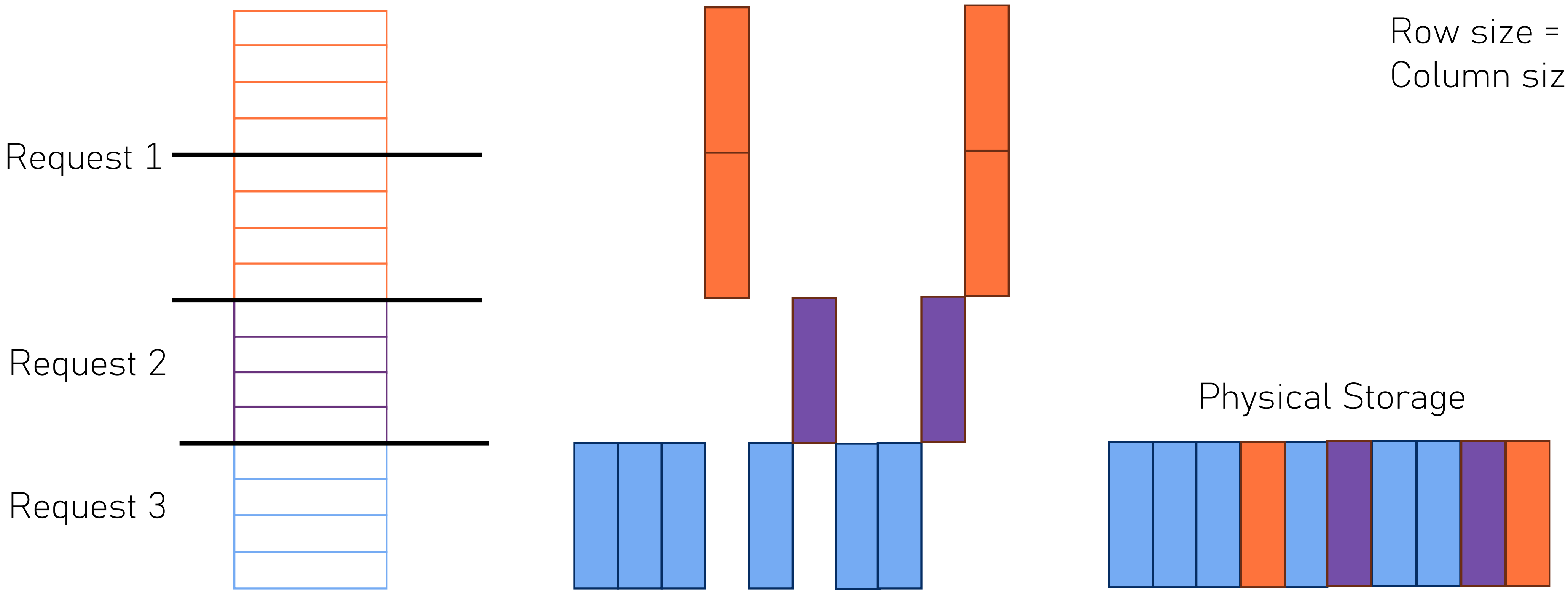
Row size = 4

Block-Sparse Matrix - Usage



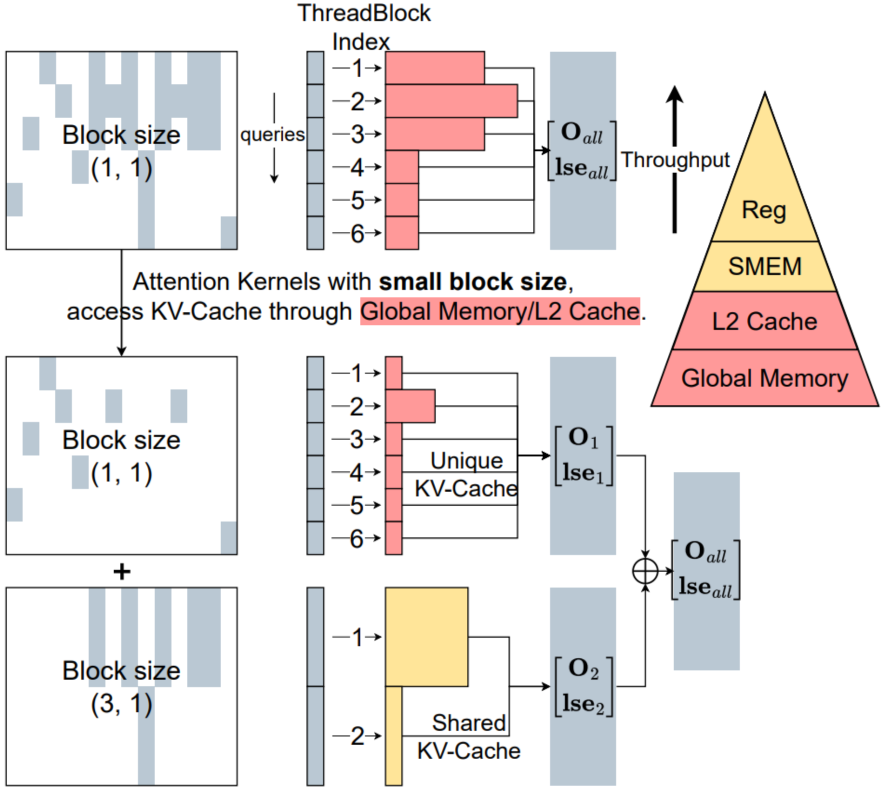
Row size = 4
Column size = 1

Block-Sparse Matrix - Usage



Row size = 4
Column size = 1

Composable Formats


ThreadBlock Index
Block size (1, 1)
queries
1
2
3
4
5
6
$\begin{bmatrix}\mathbf{O}_{all}\\ \mathbf{lse}_{all}\end{bmatrix}$
Throughput
Reg
SMEM
L2 Cache
Global Memory
Block size (1, 1)
Unique KV-Cache
$\begin{bmatrix}\mathbf{O}_{1}\\ \mathbf{lse}_{1}\end{bmatrix}$
+
Block size (3, 1)
Shared KV-Cache
$\begin{bmatrix}\mathbf{O}_{2}\\ \mathbf{lse}_{2}\end{bmatrix}$
$\oplus$
$\begin{bmatrix}\mathbf{O}_{all}\\ \mathbf{lse}_{all}\end{bmatrix}$

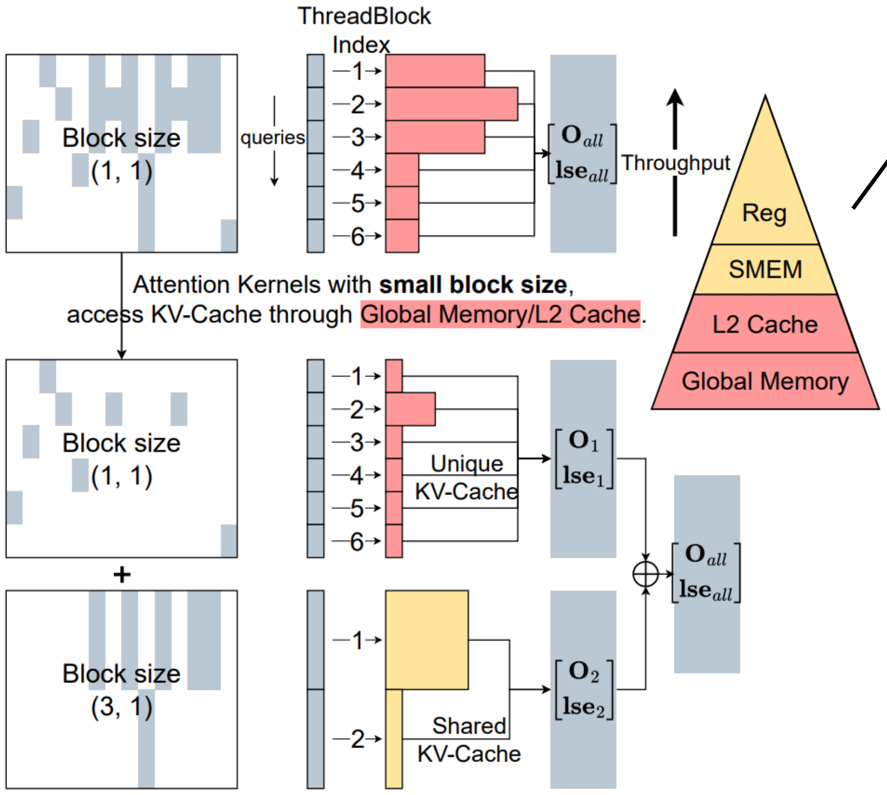
Attention Kernels with **small block size**, access KV-Cache through Global Memory/L2 Cache.

Composable Formats for Memory-Efficient Computation

Attention Kernels with **large block size**, requests inside the same block access common KV-Cache (e.g. shared-prefix) via Shared Memory/Registers

Composable Formats

Larger Bandwith

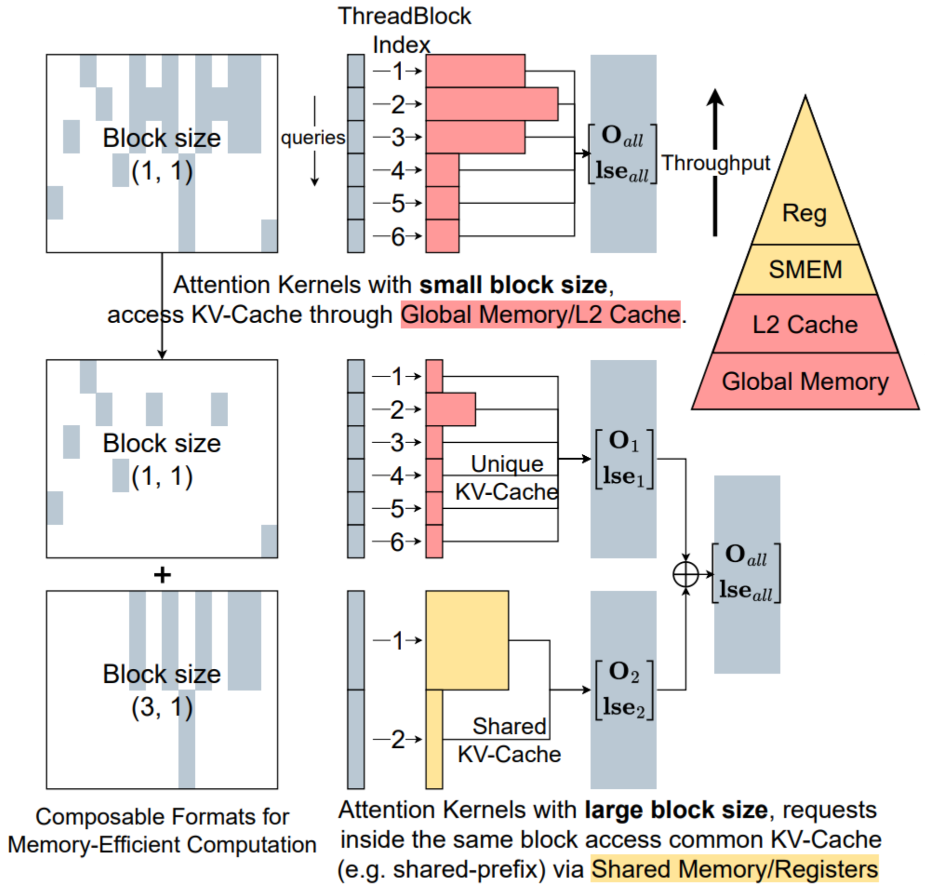

ThreadBlock Index
Block size (1, 1)
queries
1
2
3
4
5
6
$\begin{bmatrix} \mathbf{O}_{all} \\ \mathbf{lse}_{all} \end{bmatrix}$
Throughput
Reg
SMEM
L2 Cache
Global Memory
Block size (1, 1)
Unique KV-Cache
$\begin{bmatrix} \mathbf{O}_1 \\ \mathbf{lse}_1 \end{bmatrix}$
+
Block size (3, 1)
Shared KV-Cache
$\begin{bmatrix} \mathbf{O}_2 \\ \mathbf{lse}_2 \end{bmatrix}$
$\oplus$
$\begin{bmatrix} \mathbf{O}_{all} \\ \mathbf{lse}_{all} \end{bmatrix}$

Attention Kernels with **small block size**, access KV-Cache through Global Memory/L2 Cache.

Composable Formats for Memory-Efficient Computation

Attention Kernels with **large block size**, requests inside the same block access common KV-Cache (e.g. shared-prefix) via Shared Memory/Registers

Composable Formats


ThreadBlock Index
Block size (1, 1)
queries
1
2
3
4
5
6
$\begin{bmatrix} \mathbf{O}_{all} \\ \mathbf{lse}_{all} \end{bmatrix}$
Throughput
Reg
SMEM
L2 Cache
Global Memory
Block size (1, 1)
Unique KV-Cache
$\begin{bmatrix} \mathbf{O}_{1} \\ \mathbf{lse}_{1} \end{bmatrix}$
+
Block size (3, 1)
Shared KV-Cache
$\begin{bmatrix} \mathbf{O}_{2} \\ \mathbf{lse}_{2} \end{bmatrix}$
$\oplus$
$\begin{bmatrix} \mathbf{O}_{all} \\ \mathbf{lse}_{all} \end{bmatrix}$

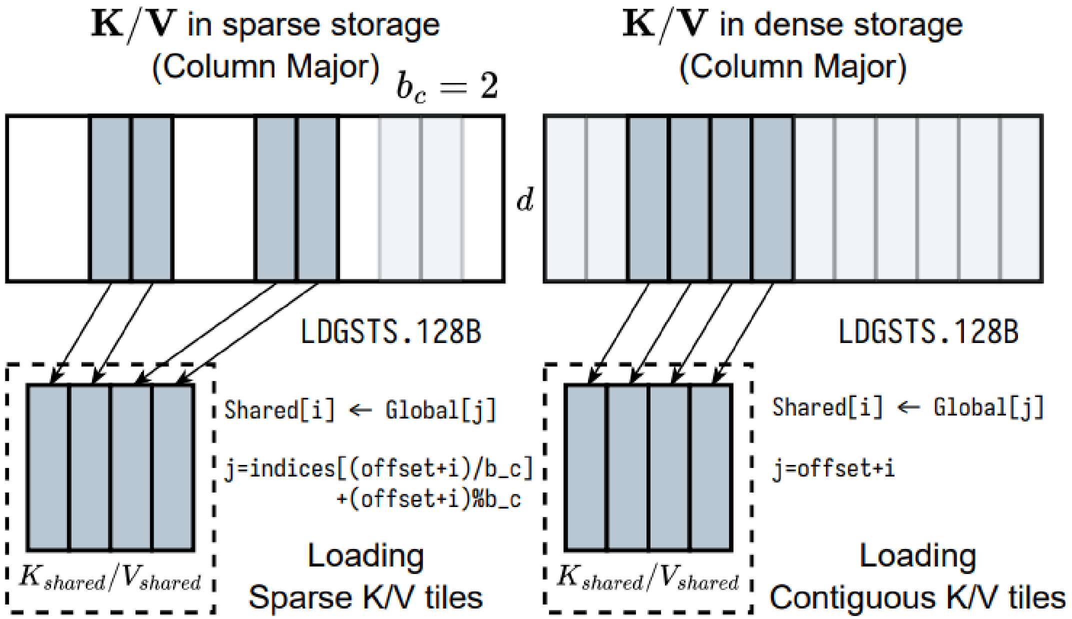
Attention Kernels with **small block size**, access KV-Cache through Global Memory/L2 Cache.

Attention Kernels with **large block size**, requests inside the same block access common KV-Cache (e.g. shared-prefix) via Shared Memory/Registers

Composable Formats for Memory-Efficient Computation

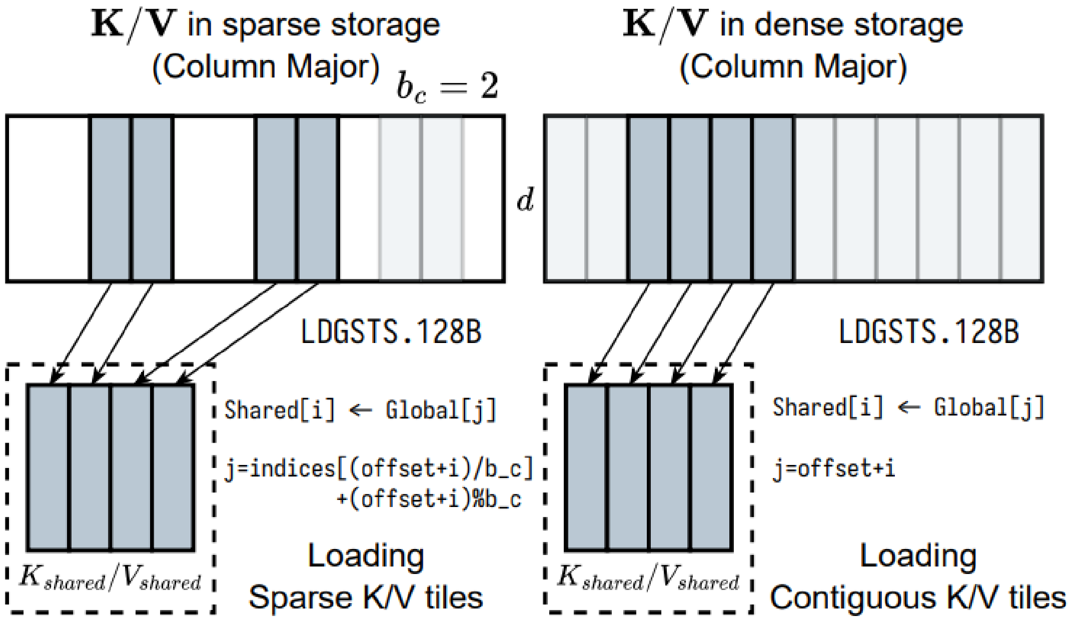
Smaller Bandwith

Global To Shared Memory Data Movement



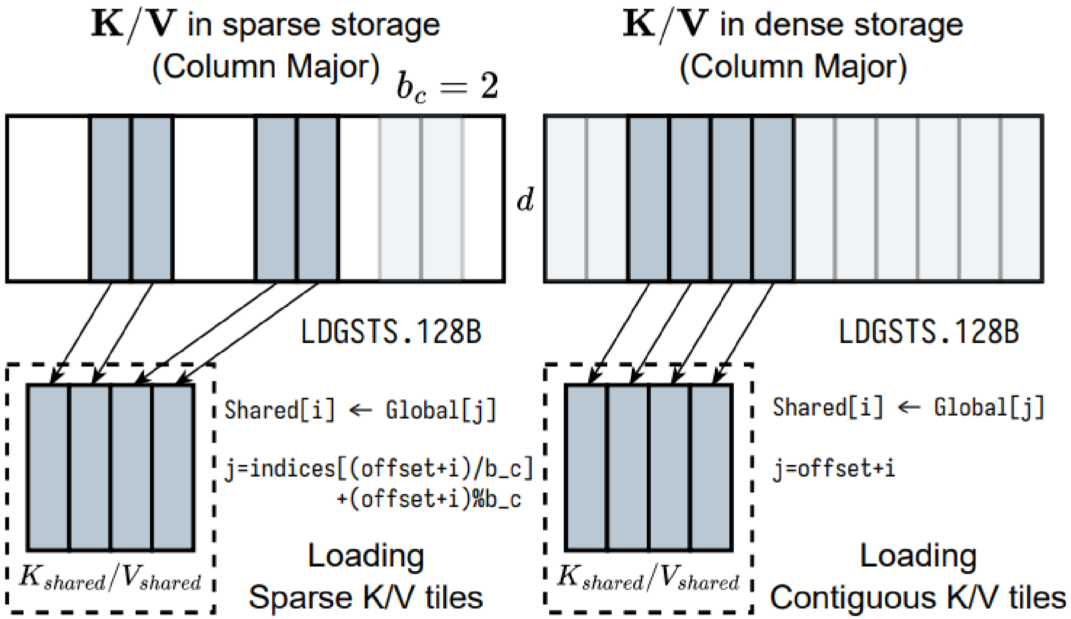
Global To Shared Memory Data Movement

Async copy



Global To Shared Memory Data Movement

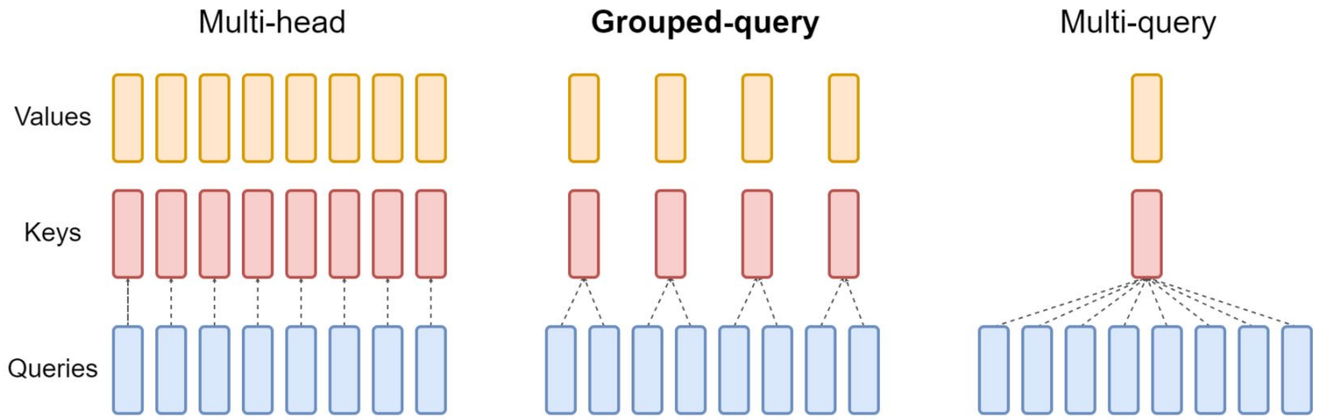
Async copy



Tensor Memory Accelerator for Hopper

JIT Compiler – Background

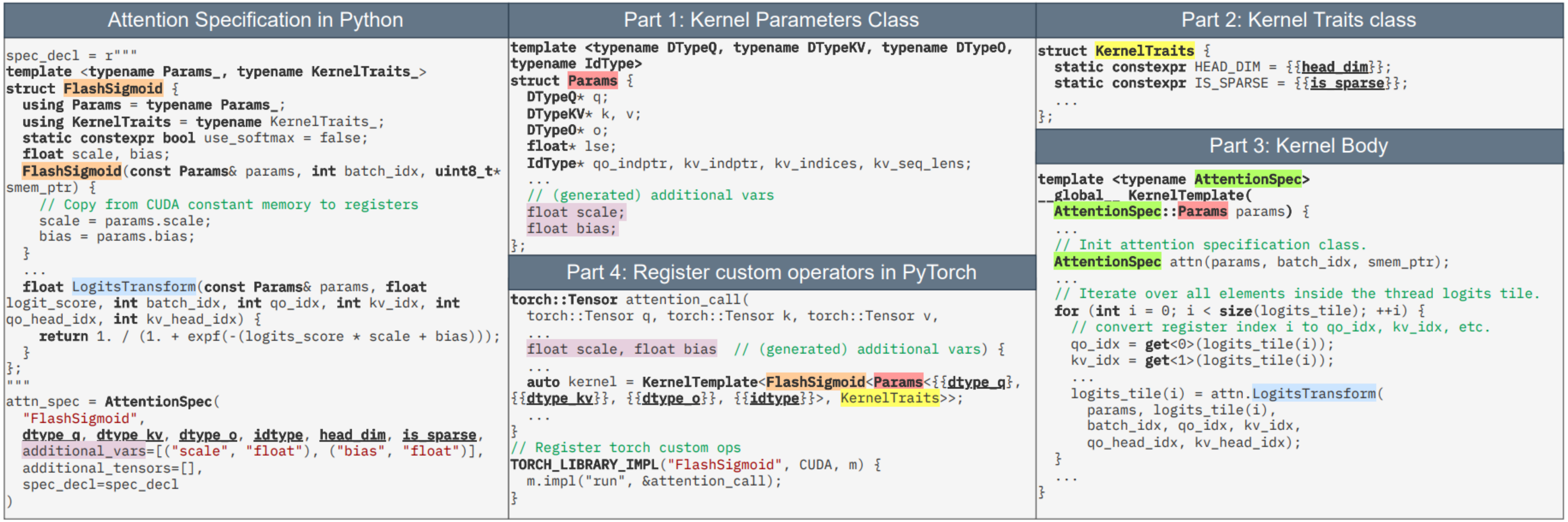
- Different attention algorithms
- Grouped-query attention
  - A group of query heads share the same K and V



JIT Compiler

- Enable defining the specifications of attention variants
- Provide interfaces that can be customized
  - QueryTransform, KeyTransform, ValueTransform
  - OutputTransform
  - LogitsTransform, LogitsMask

JIT Compiler - Example



Attention Specification in Python

```
spec_decl = r"""
template <typename Params_, typename KernelTraits_>
struct FlashSigmoid {
  using Params = typename Params_;
  using KernelTraits = typename KernelTraits_;
  static constexpr bool use_softmax = false;
  float scale, bias;
  FlashSigmoid(const Params& params, int batch_idx, uint8_t*
smem_ptr) {
    // Copy from CUDA constant memory to registers
    scale = params.scale;
    bias = params.bias;
  }
  ...
  float LogitsTransform(const Params& params, float
logit_score, int batch_idx, int qo_idx, int kv_idx, int
qo_head_idx, int kv_head_idx) {
    return 1. / (1. + expf(-(logits_score * scale + bias)));
  }
};
"""
attn_spec = AttentionSpec(
  "FlashSigmoid",
  dtype_q, dtype_kv, dtype_o, idtype, head_dim, is_sparse,
  additional_vars=[("scale", "float"), ("bias", "float")],
  additional_tensors=[],
  spec_decl=spec_decl
)
```

Part 1: Kernel Parameters Class

```
template <typename DTypeQ, typename DTypeKV, typename DTypeO,
typename IdType>
struct Params {
  DTypeQ* q;
  DTypeKV* k, v;
  DTypeO* o;
  float* lse;
  IdType* qo_indptr, kv_indptr, kv_indices, kv_seq_lens;
  ...
  // (generated) additional vars
  float scale;
  float bias;
};
```

Part 4: Register custom operators in PyTorch

```
torch::Tensor attention_call(
  torch::Tensor q, torch::Tensor k, torch::Tensor v,
  ...
  float scale, float bias  // (generated) additional vars) {
  ...
  auto kernel = KernelTemplate<FlashSigmoid<Params<{{dtype_q}},
{{dtype_kv}}, {{dtype_o}}, {{idtype}}>, KernelTraits>>;
  ...
}
// Register torch custom ops
TORCH_LIBRARY_IMPL("FlashSigmoid", CUDA, m) {
  m.impl("run", &attention_call);
}
```

Part 2: Kernel Traits class

```
struct KernelTraits {
  static constexpr HEAD_DIM = {{head_dim}};
  static constexpr IS_SPARSE = {{is_sparse}};
  ...
};
```

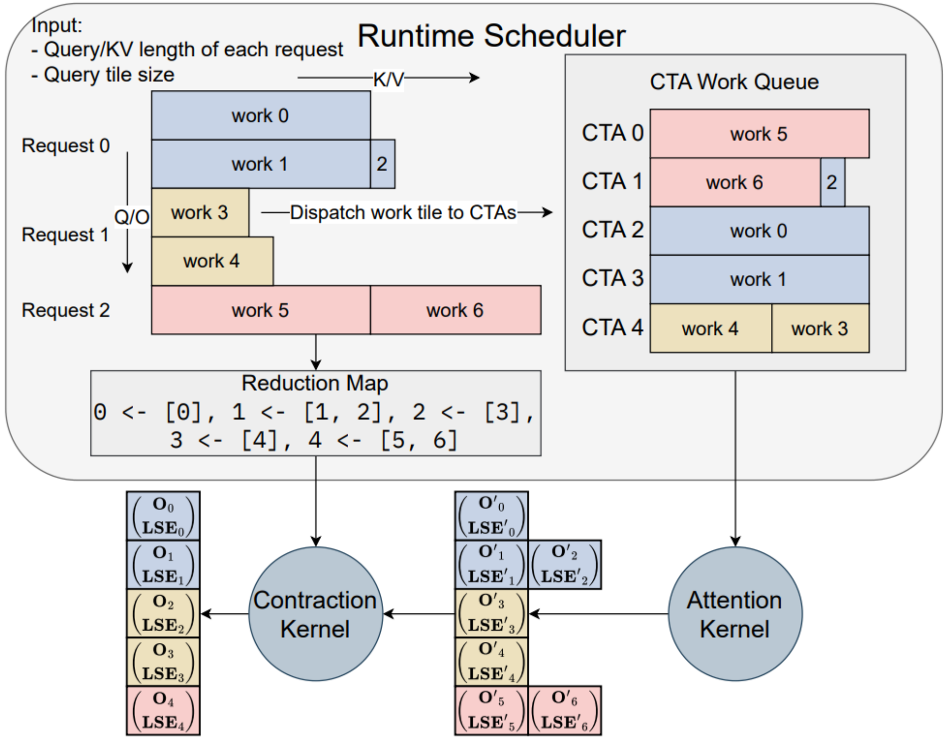
Part 3: Kernel Body

```
template <typename AttentionSpec>
__global__ KernelTemplate(
  AttentionSpec::Params params) {
  ...
  // Init attention specification class.
  AttentionSpec attn(params, batch_idx, smem_ptr);
  ...
  // Iterate over all elements inside the thread logits tile.
  for (int i = 0; i < size(logits_tile); ++i) {
    // convert register index i to qo_idx, kv_idx, etc.
    qo_idx = get<0>(logits_tile(i));
    kv_idx = get<1>(logits_tile(i));
    ...
    logits_tile(i) = attn.LogitsTransform(
      params, logits_tile(i),
      batch_idx, qo_idx, kv_idx,
      qo_head_idx, kv_head_idx);
  }
  ...
}
```

Load-Balanced Scheduling

- Motivation: different categories of workloads
  - Prefill – initial input without the need to do attention with previous tokens
  - Decoding – only a query vector each time
  - Append – additional inputs that need to do attention with previous tokens
- Schedule based on the needed KV Cache size for one chunk

Load-Balanced Scheduling



Input:
- Query/KV length of each request
- Query tile size

Runtime Scheduler

K/V

Q/O

Request 0

Request 1

Request 2

work 0

work 1

2

work 3

work 4

work 5

work 6

Dispatch work tile to CTAs

CTA Work Queue

CTA 0 work 5

CTA 1 work 6 2

CTA 2 work 0

CTA 3 work 1

CTA 4 work 4 work 3

Reduction Map

```
0 <- [0], 1 <- [1, 2], 2 <- [3],
   3 <- [4], 4 <- [5, 6]
```

Contraction Kernel

Attention Kernel

$\begin{pmatrix} O_0 \\ LSE_0 \end{pmatrix}$ $\begin{pmatrix} O_1 \\ LSE_1 \end{pmatrix}$ $\begin{pmatrix} O_2 \\ LSE_2 \end{pmatrix}$ $\begin{pmatrix} O_3 \\ LSE_3 \end{pmatrix}$ $\begin{pmatrix} O_4 \\ LSE_4 \end{pmatrix}$

$\begin{pmatrix} O'_0 \\ LSE'_0 \end{pmatrix}$ $\begin{pmatrix} O'_1 \\ LSE'_1 \end{pmatrix}$ $\begin{pmatrix} O'_2 \\ LSE'_2 \end{pmatrix}$ $\begin{pmatrix} O'_3 \\ LSE'_3 \end{pmatrix}$ $\begin{pmatrix} O'_4 \\ LSE'_4 \end{pmatrix}$ $\begin{pmatrix} O'_5 \\ LSE'_5 \end{pmatrix}$ $\begin{pmatrix} O'_6 \\ LSE'_6 \end{pmatrix}$

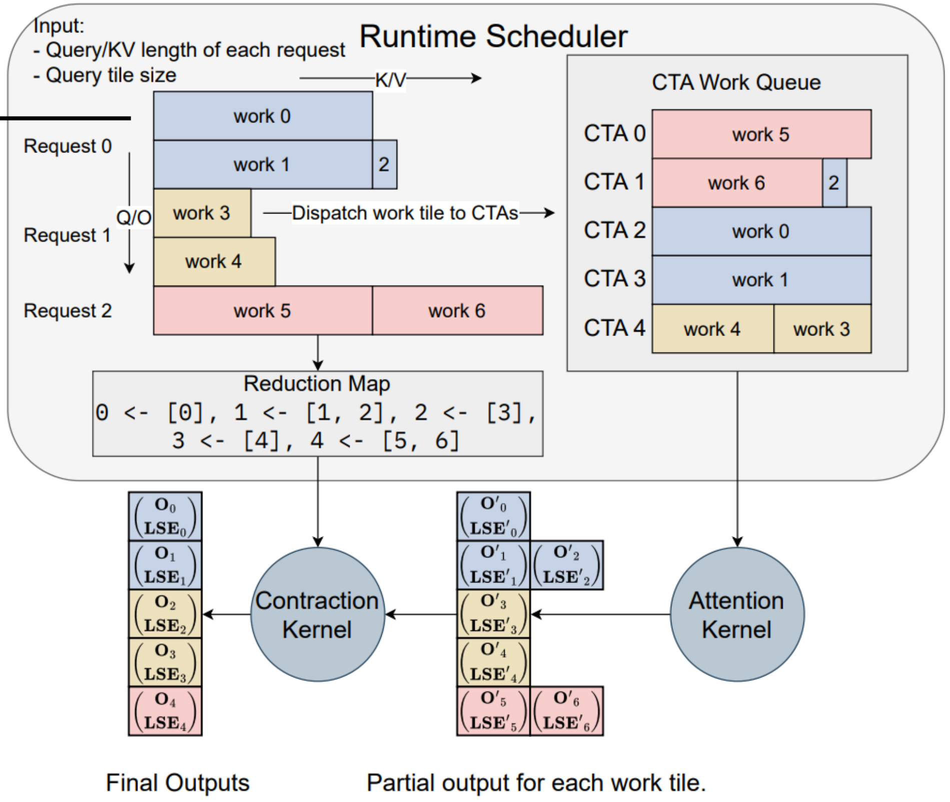
Final Outputs

Partial output for each work tile.

Load-Balanced Scheduling

Length means cost

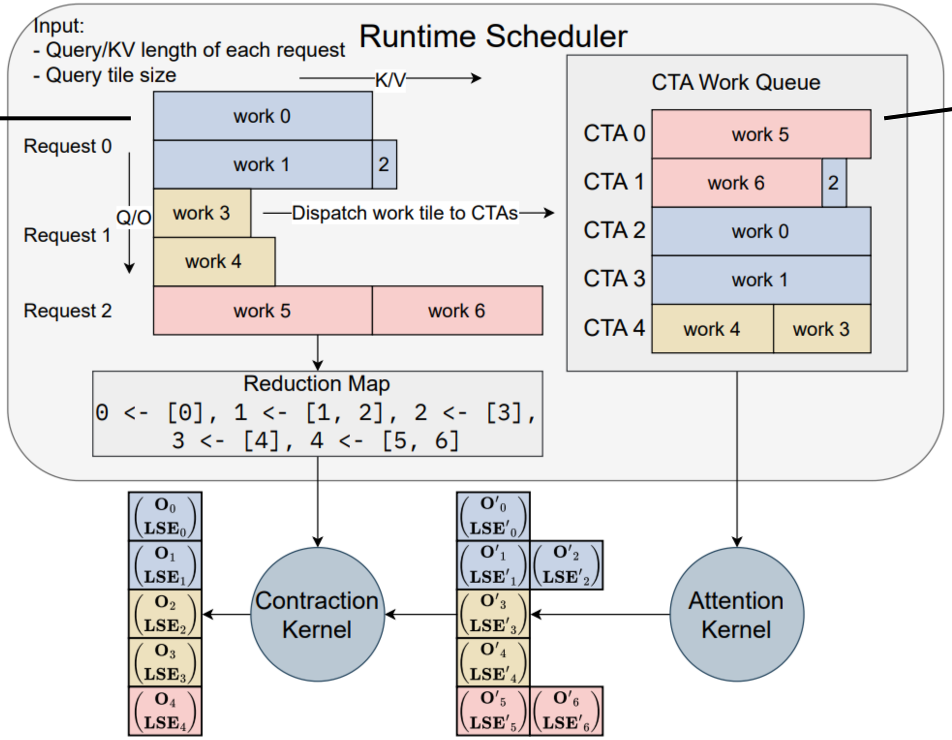
Cost is calculated by the length of Q, K, V



Load-Balanced Scheduling

Length means cost

Cost is calculated by the length of Q, K, V



Final Outputs

Partial output for each work tile.

Assign works with a priority queue

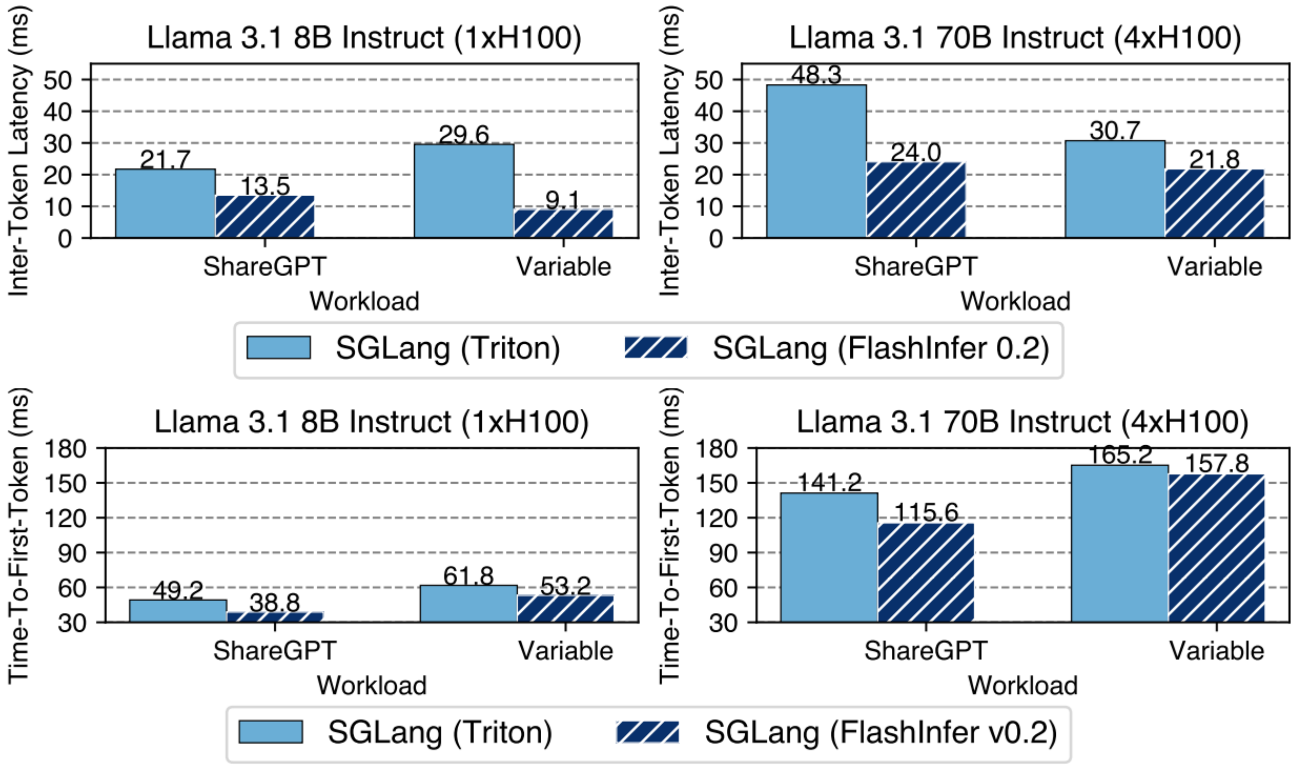
Evaluation – Experiment Settings

- NVIDIA A100 40GB, H100 80GB
- CUDA 12.4
- Pytorch 2.4.0
- Precision: f16
- Serving engine: SGLang 0.3.4
- Models: Llama3.1 8B, 70B, Vicuna-13B (fine-tuned from Llama)

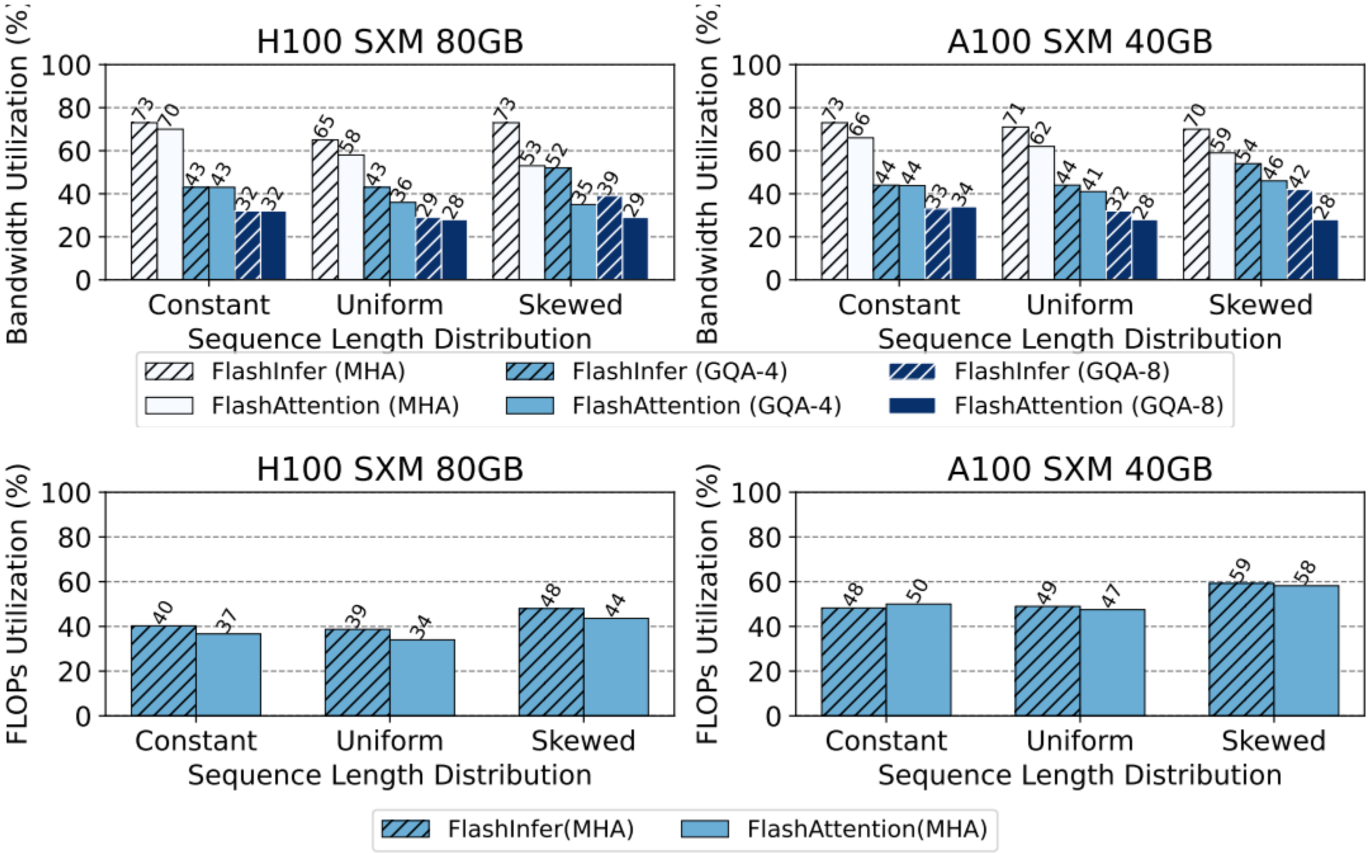
Evaluation – Workloads & Metrics

- Datasets
  - ShareGPT
  - Variable: different sequence lengths in different distribution (512-2048)
- Metrics
  - TTFT and TPOT (median)
  - Bandwith and FLOPs utilization

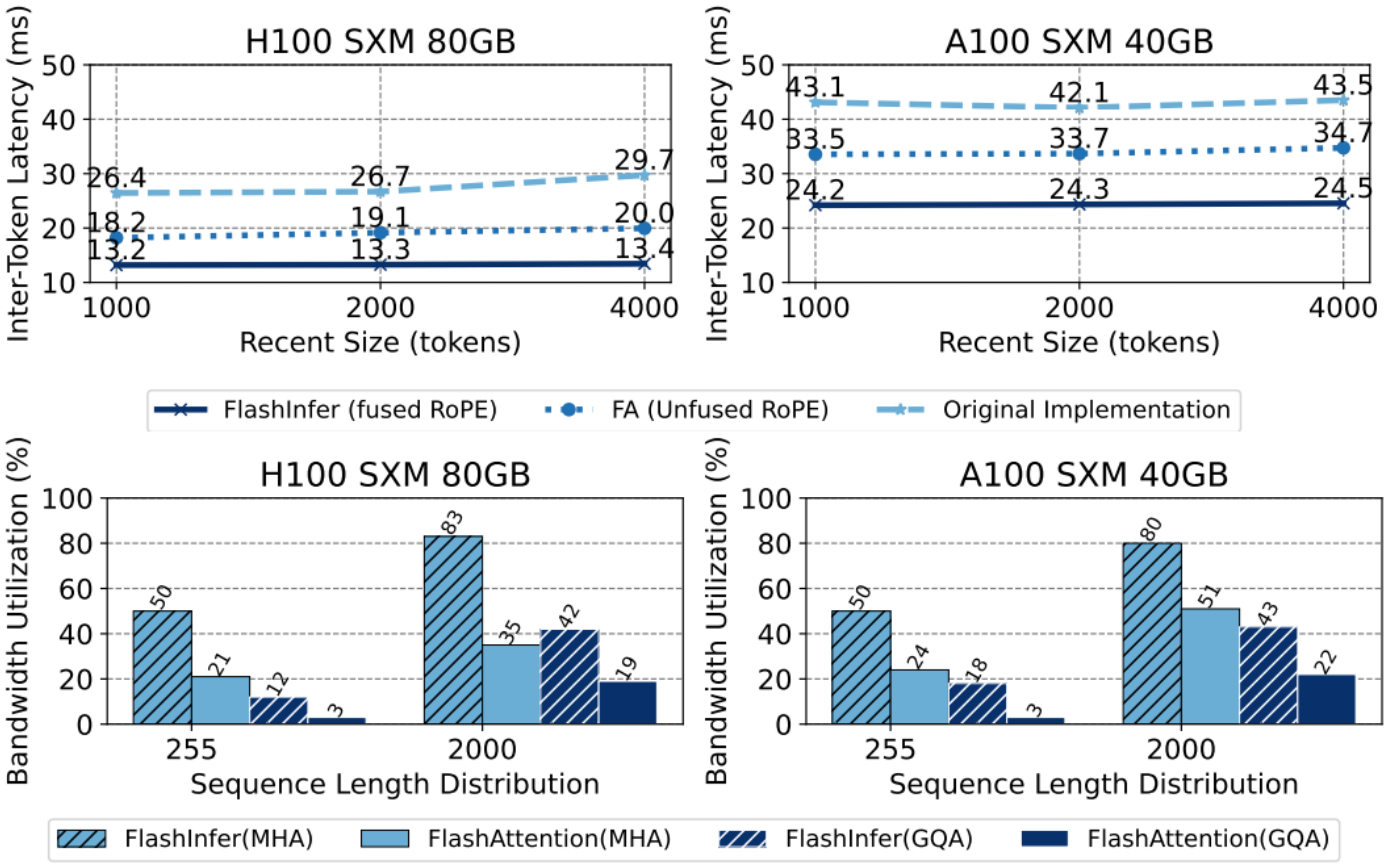
Evaluation – Latency



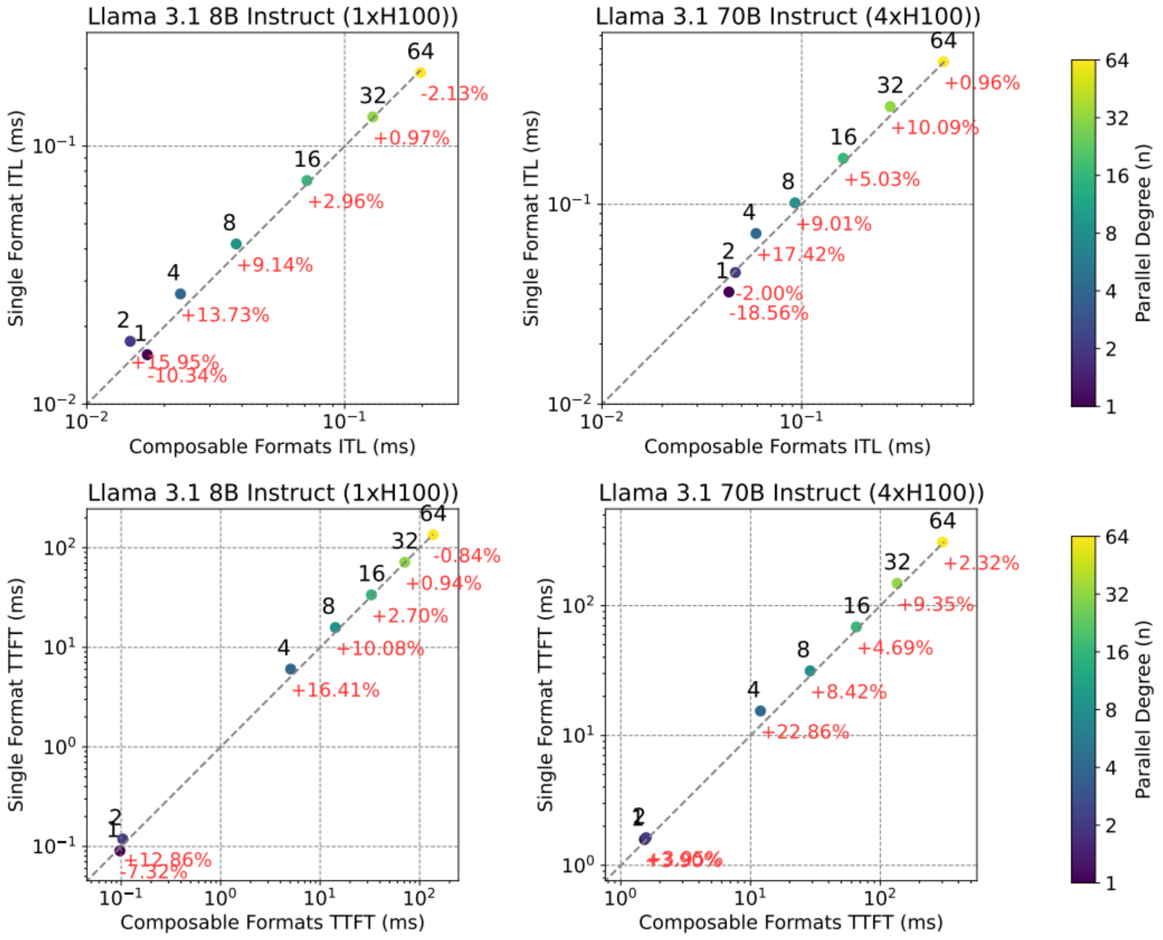
Evaluation – Kernel Performance



Evaluation – Long-Context Inference



Evaluation – Parallel Generation



Homework

- Read the paper **Taming Throughput-Latency Tradeoff in LLM Inference with Sarathi-Serve**, summarize the paper, specifically, including the points below:
  - What are the motivations/challenges of this work?
  - How does the design of this paper address the challenges?
  - How does the paper evaluate its design (experiment settings, workloads, metrics)?
  - How does the evaluation prove its claims?
- Related link:
  - Paper of Sarathi-Serve: https://www.usenix.org/system/files/osdi24-agrawal.pdfLinks to an external site.

Q & A