CompactAttention: Accelerating Chunked Prefill with Block-Union KV Selection

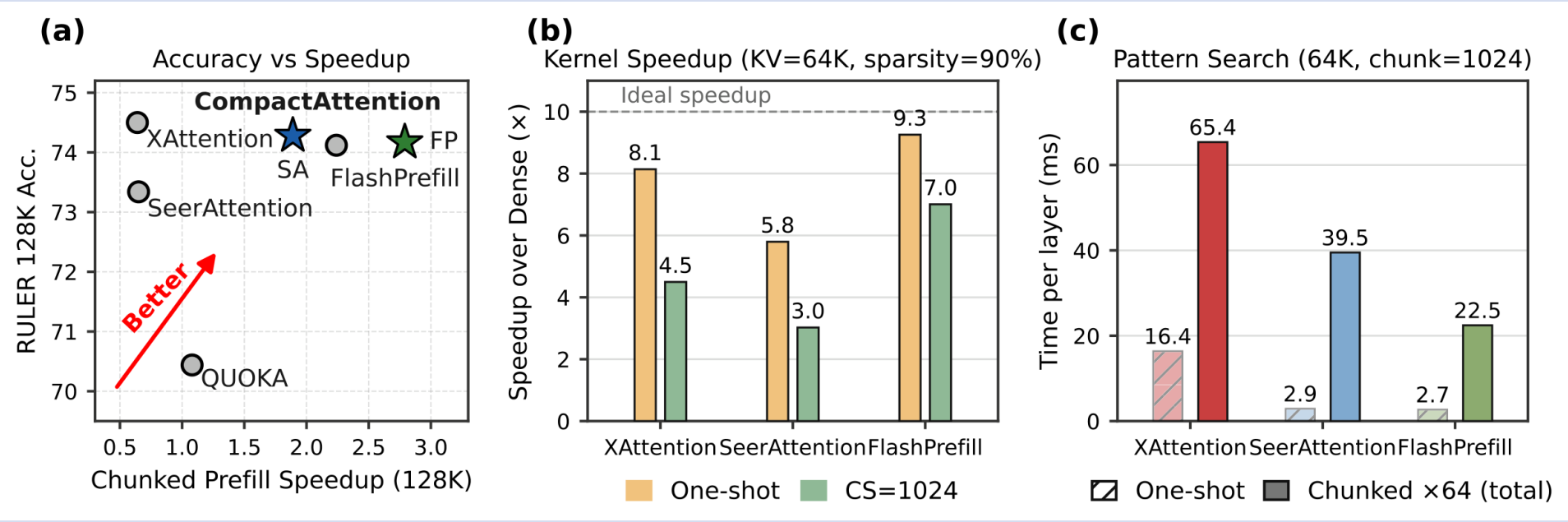
NeurIPS 2026

Spotlight

Jiwon Song, Dongwon Jo, Beomseok Kang, Jae-Joon Kim*

Seoul National University

Chunked prefill breaks sparse attention



Block-sparse attention

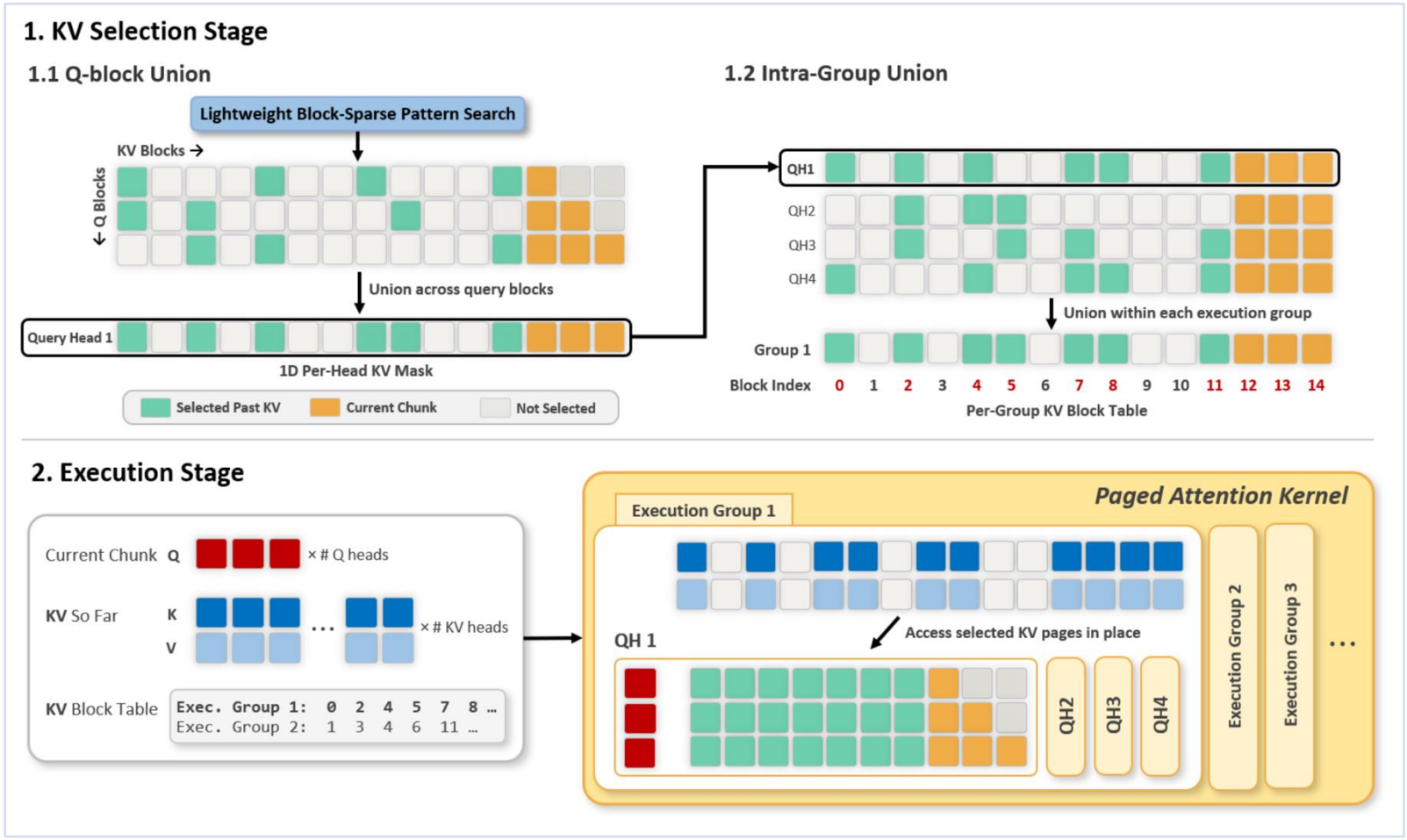
- **Kernel:** Q ≪ KV leaves too few query blocks, so 90% sparsity gives only 3.0–7.0× vs the ideal 10×
- **Search:** scoring reruns every chunk, e.g. XAttention 16.4 → 65.4 ms per layer at 64K

QUOKA (query-subsampled, token-level)

- **Coverage:** KV needed only by unsampled queries is missed
- **Copy:** token-level selection needs an explicit gather before attention

Need: cover all query blocks, avoid sparse kernels, and access KV blocks without copying

Sparse selection, dense paged execution



① Block-Union KV Selection

- **Q-block union** over query blocks
- **Intra-group union** over heads in a group
- One table per group: **T[b,g] = {j | G[b,g,j] = 1}**
- Coverage-preserving and minimal under a shared table

② Zero-copy Paged Execution

- KV-head-major layout: each (batch, KV head, block) is a page
- Only metadata (kv_indptr, kv_indices) goes to a dense paged kernel

Trade-off: union lowers sparsity, so we start from a more aggressive mask: 90.2% sparsity before union, 69.7% after, comparable to Flash Prefill (71.8%).

Accuracy stays close to dense attention

Method	LLaMA-3.1-8B-Instruct 32K	64K	128K	Avg	Qwen3-30B-A3B-Instruct-2507 32K	64K	128K	Avg
Dense	88.23	83.52	76.59	82.78	93.01	90.56	87.81	90.46
QUOKA	83.52	79.99	70.44	77.98	81.96	79.57	79.03	80.19
XAttention	89.16	83.27	74.50	82.31	92.29	88.92	88.13	89.78
SeerAttention	90.23	83.82	73.34	82.46			–	
CompactAttention-SA	88.92	84.12	74.28	82.44			–	
FlashPrefill	88.99	83.94	74.12	82.35	91.55	87.73	86.77	88.68
CompactAttention-FP	88.77	83.96	74.17	82.30	92.41	88.43	87.26	89.37

RULER accuracy across context lengths (chunk size 1024); CA = CompactAttention

Near-dense accuracy

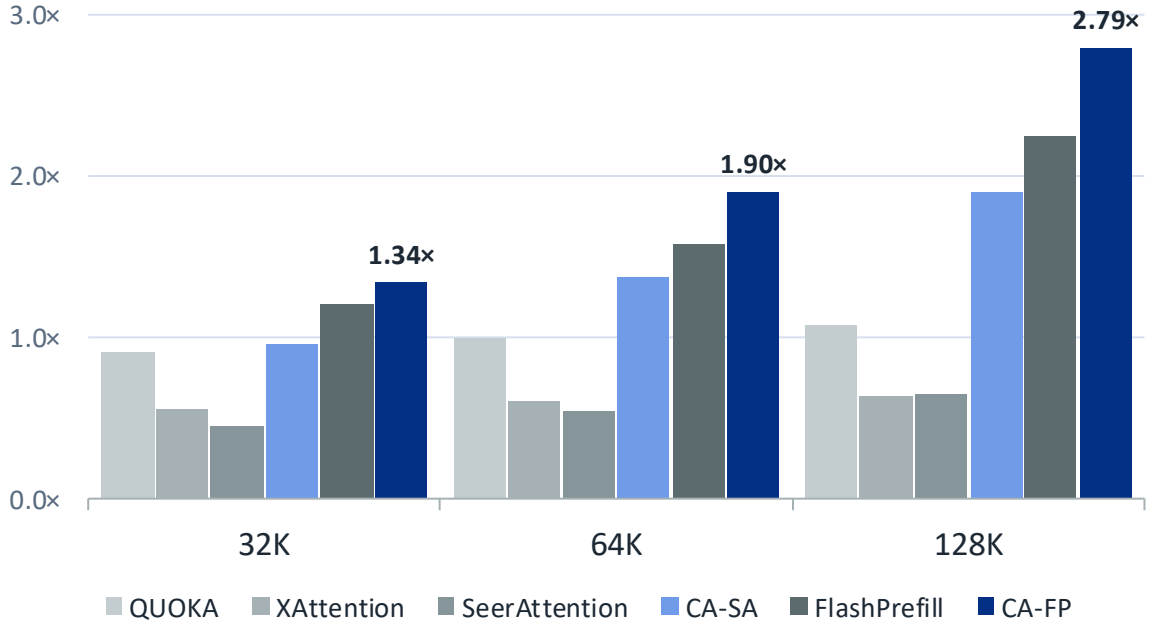
- LLaMA avg: CA-FP 82.30, CA-SA 82.44 vs dense 82.78
- Qwen3-30B-A3B avg: CA-FP 89.37 vs dense 90.46
- Same as the selector it reuses: all selected blocks are kept

QUOKA drops

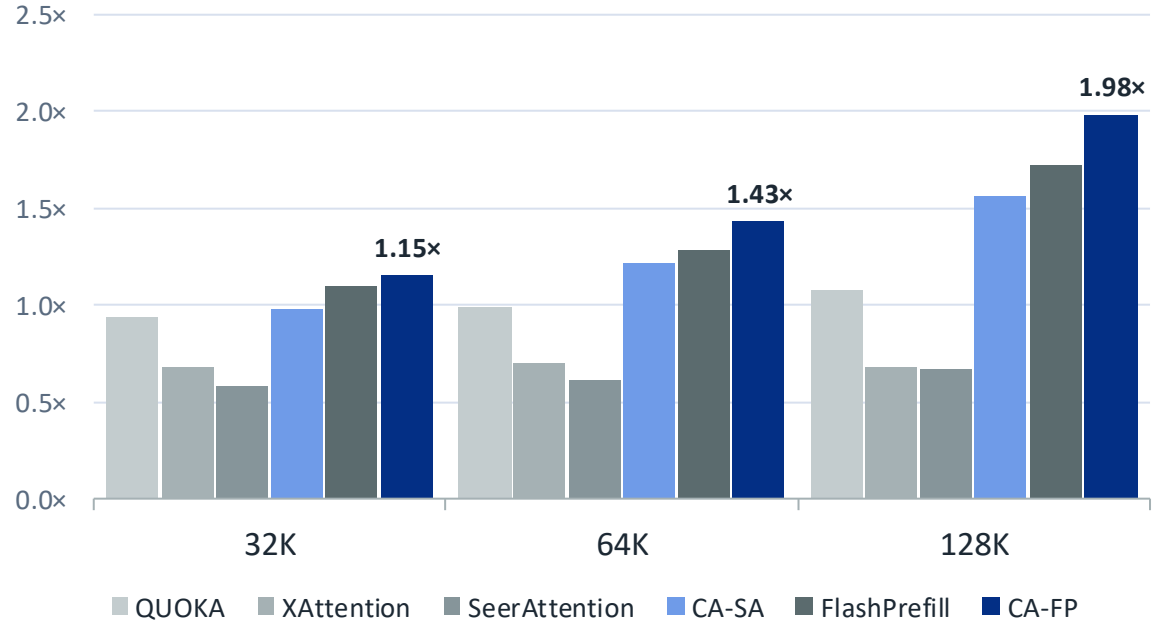
- Avg 77.98 (LLaMA) and 80.19 (Qwen3)
- Multi-key NIAH-3 92.0 → 76.0, CWE 11.2 → 0.0 at 64K
- LongBench V2: CA stays near dense; QUOKA drops on Hard

Up to 2.79× faster attention at 128K

Attention speedup over dense



End-to-end speedup over dense



LLaMA-3.1-8B-Instruct, H200 SXM (TP2, batch 8, chunk 1024)

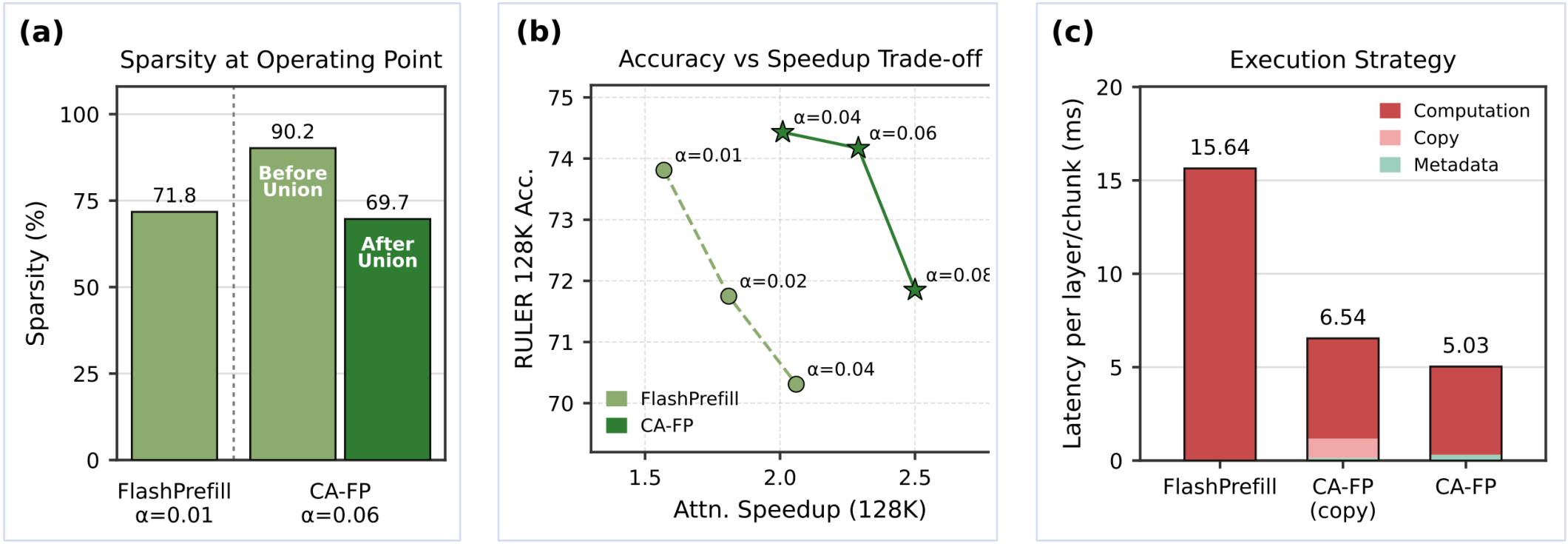
H200, 128K context

- **2.79×** attention vs dense, **1.25×** vs FlashPrefill
- **1.98×** end-to-end vs dense, **1.15×** vs FlashPrefill
- RTX PRO 6000 shows the same trend: 2.74× attention, 1.75× E2E

Why the baselines lag

- XAttention and SeerAttention are often slower than dense at Q ≪ KV
- QUOKA gains little: gather cost offsets the smaller KV

Execution matters as much as selection



(a) sparsity at the operating point · (b) accuracy–speedup trade-off under α sweep · (c) execution-only ablation at matched sparsity

Execution path (same unioned mask)

- Block-sparse kernel 15.64 ms → gather + dense 6.54 ms → **paged in place 5.03 ms**
- Metadata cost is small; in-place access removes the copy

Sparsity after union

- α = 0.06 mask: 90.2% before union → 69.7% after
- Comparable to FlashPrefill at α = 0.01 (71.8%)

Conclusion

CompactAttention accelerates chunked prefill via:

✓ **Decoupling** block-level KV selection from kernel execution

✓ **Block-union tables** that keep every block chosen by the selector

✓ **Zero-copy paged attention:** 2.79× attention, 1.98× end-to-end at 128K on H200

✓ Near-dense accuracy on RULER and LongBench V2

Limitations

- Quality is bounded by the selector; missed blocks cannot be recovered
- Sparsity is set indirectly; some groups become nearly dense
- Gains shrink at short context or small batch
- vLLM / SGLang integration needs per-group block tables and a KV-head-major cache