PR #36624 完整报告

sgl-project/sglang

[Cohere Command-A-Plus] Optimize decode and BCG capture on SM10X

合并时间：2026-09-01 10:37

原文链接：http://prhub.com.cn/sgl-project/sglang/pull/36624

执行摘要

- 一句话：SM10X Command-A-Plus 解码提速 2.3 倍，BCG 捕获内存降 78%
- 推荐动作：值得精读。三项改动都落在核心路径且跨模块（模型实现 + 配置门控 + 白名单 + 测试），评分 7。重点学习：流的按角色租用如何规避缓存分配器分池导致的捕获 OOM；多流并行时 in-place 别名引发的跨流数据竞争及其消除方法；decode-only 重叠的收益边界（prefill 已饱和时事件开销大于收益）；fail-closed 门控如何在不确定配置下避免把用户带进崩溃路径，并用 " 不可读配置 " 测试用例固化契约。

功能与动机

PR body 自述三个集中在 SM10X 上的问题：其一，Cohere2MoeSparseMoeBlock.__init__ 每层调用 torch.cuda.Stream()，模型构建了 32 条流，缓存分配器按流分池后 prefill 捕获膨胀到 19.02 GiB graph pool + 23.83 GiB 滞留内存，超过约 45.7 GiB 余量直接 OOM，导致 BCG prefill 后端在此架构不可用；其二，y = x + attn(norm(x)) + mlp(norm(x)) 两条分支相互独立却同流串行，decode 时两者都不能填满 GPU；其三，auto 的 MoE runner 会把 routing、expandInputRows、activation、finalize 拆成独立 kernel 围绕 expert GEMM，而 trtllm-gen 可全部融合进一个。

实现拆解

1. 共享流租用，修复 prefill 捕获 OOM：python/sglang/srt/models/cohere2_moe.py 中 Cohere2MoeSparseMoeBlock.__init__ 把 torch.cuda.Stream() 换成从 sglang.srt.runtime_context 导入的 get_stream("alt")，Cohere2MoeDecoderLayer.__init__ 新增 self.mlp_stream = get_stream("alt_mlp")。改因：缓存分配器按流分池，32 条独立流让捕获内存膨胀到约 42 GiB 超过约 45.7 GiB 余量。同时 python/sglang/srt/configs/model_config.py 将 Cohere2VisionForConditionalGeneration 从 multimodal_piecewise_cuda_graph_supported_model_archs 移到 multimodal_breakable_cuda_graph_supported_model_archs，使该架构走 breakable prefill（捕获后降到 5.34 GB）。
2. 解码阶段 attention 与 MLP 跨流重叠：Cohere2MoeDecoderLayer.forward 在 get_is_capture_mode() 且 forward_batch.forward_mode.is_decode() 时把 mlp(hidden_states) 放到 mlp_stream，主流同时执行 self_attn，末尾 wait_stream 汇合；仅 decode 生效是因为 prefill chunk 已占满 GPU，拆分只增加事件开销。配套把 Cohere2MoeSparseMoeBlock 里的 FusedMoE 构造改为 inplace=False：默认 in-place

别名会把 routed-expert 输出写回输入 buffer，而另一条流上的 qkv_proj 仍要读该 buffer，形成跨流数据竞争。该改动对非捕获路径同样生效，作者实测保留 shared_input.clone() 整体更快。

3. MoE runner 自动门控（fail-closed）：新增 python/sglang/srt/arg_groups/model_overrides/cohere2_moe.py，通过 @_register_for 声明 Cohere2MoeForCausalLM 与 Cohere2VisionForConditionalGeneration 两个架构：当 moe_runner_backend == "auto"、平台为 SM10X、且（未量化或 _is_nvfp4_pack_quantized 判定为 NVFP4）时返回 {"moe_runner_backend": "flashinfer_trtllm"}，否则返回空 dict。FP8 保持 auto，因为 CompressedTensorsW8A8Fp8MoE 交给 trtllm-gen 的 TritonMoeQuantInfo 在首次 forward 就会被拒绝。在 model_overrides/__init__.py 注册导入。commit 2db4ddc 记录：main 分支出时已把 per-family 声明拆到 arg_groups/model_overrides/ 包，本分支原加在 overrides.py 的门控被迁移并适配 get_platform().is_sm100、model_config_of() 等新惯用法。
4. 测试配套：test/registered/unit/test_model_overrides.py 新增 test_cohere2_moe_runner_gate 与 test_cohere2_moe_runner_gate_fails_closed。前者覆盖 NVFP4 → flashinfer_trtllm、bf16 → flashinfer_trtllm、FP8 → auto、显式 triton 不被覆盖、非 SM10X 保持 auto、_publish 后执行端生效、vision wrapper 从顶层读 quantization_config；后者验证无法读取的量化配置（如独立 hf_quant_config.json）保持 auto，未量化才钉 runner。

关键文件：

- python/sglang/srt/models/cohere2_moe.py（模块 模型实现；类别 source；类型 core-logic；符号 Cohere2MoeSparseMoeBlock, Cohere2MoeDecoderLayer）：模型核心路径：共享流租用替代每层 Stream，新增 mlp_stream 实现 decode 阶段 attention/MLP 跨流重叠，FusedMoE 改 inplace=False 消除跨流写回竞争
- python/sglang/srt/arg_groups/model_overrides/cohere2_moe.py（模块 配置覆盖；类别 source；类型 configuration；符号 _is_nvfp4_pack_quantized, _cohere2_moe_runner_overrides）：新增 fail-closed 的 MoE runner 门控：SM10X 上未量化或 NVFP4 的 Command-A-Plus 自动钉到 flashinfer_trtllm，不可读量化配置与 FP8 保持 auto，是后端正确性的关键决策点
- test/registered/unit/test_model_overrides.py（模块 单元测试；类别 test；类型 test-coverage；符号 test_cohere2_moe_runner_gate, test_cohere2_moe_runner_gate_fails_closed, _NVFP4_QUANT, _FP8_QUANT）：新增 2 个门控用例，覆盖 NVFP4/bf16/FP8/ 显式后端 / 非 SM10X 及不可读量化配置的 fail-closed 行为，是后端选择正确性的回归保障
- python/sglang/srt/configs/model_config.py（模块 模型配置；类别 source；类型 configuration；符号 multimodal_breakable_cuda_graph_supported_model_archs, multimodal_piecewise_cuda_graph_supported_model_archs）：把 Cohere2VisionForConditionalGeneration 从 piecewise 白名单迁到 breakable 白名单，是该架构可用 BCG prefill 的开关
- python/sglang/srt/arg_groups/model_overrides/__init__.py（模块 注册入口；类别 source；类型 configuration）：注册 cohere2_moe 覆盖模块，使其声明在导入包时自动生

效

关键符号：Cohere2MoeSparseMoeBlock.forward, Cohere2MoeSparseMoeBlock.init, Cohere2MoeDecoderLayer.forward, _cohere2_moe_runner_overrides, _is_nvfp4_pack_quantized

关键源码片段

python/sglang/srt/models/cohere2_moe.py

模型核心路径：共享流租用替代每层 Stream，新增 mlp_stream 实现 decode 阶段 attention/MLP 跨流重叠，FusedMoE 改 inplace=False 消除跨流写回竞争

```
# ============ Command-A-Plus（Cohere2Moe）SM10X 多流优化 ============

class Cohere2MoeSparseMoeBlock(nn.Module):
    def __init__(self, config, layer_id, quant_config, prefix):
        # ... 省略 gate、topk、shared_experts 构造 ...
        # 流租用：按角色共享一条 'alt' 流，而不是每层新建 Stream。
        # 每层 torch.cuda.Stream() 会让缓存分配器为每条流建独立内存池，
        # 32 层累计在 prefill 图捕获时膨胀到 42 GiB+，直接 OOM。
        self.alt_stream = (
            get_stream('alt') if is_cuda() and self.shared_experts is not None else None
        )

    def forward(self, hidden_states: torch.Tensor) -> torch.Tensor:
        orig_shape = hidden_states.shape
        hidden_states = hidden_states.view(-1, self.hidden_size)
        # FusedMoE 已改为 inplace=False，但 shared_experts 与 routed experts
        # 并发运行仍需各自持有 post-norm 输入，实测保留 clone 更快。
        shared_input = hidden_states.clone()

        if self.alt_stream is not None and get_is_capture_mode():
            # 捕获期：shared_experts 在 alt 流，gate + topk + routed experts
            # 留在主流，两条链并行；非捕获期同步开销大于收益，退化为串行。
            current_stream = torch.cuda.current_stream()
            shared_input.record_stream(self.alt_stream)
            self.alt_stream.wait_stream(current_stream)
            with torch.cuda.stream(self.alt_stream):
                shared_out = self.shared_experts(shared_input)
            router_logits, _ = self.gate(hidden_states)
            topk_output = self.topk(hidden_states, router_logits)
            routed_out = self.experts(hidden_states, topk_output)
            current_stream.wait_stream(self.alt_stream)
        else:
            # 非捕获模式（eager）下单流串行，避免事件同步开销
            router_logits, _ = self.gate(hidden_states)
            topk_output = self.topk(hidden_states, router_logits)
            routed_out = self.experts(hidden_states, topk_output)
            shared_out = self.shared_experts(shared_input)
```

```
        final_hidden_states = routed_out + shared_out
        if self.shared_expert_combination_strategy == 'average':
            final_hidden_states = final_hidden_states / 2
        return final_hidden_states.view(orig_shape)


class Cohere2MoeDecoderLayer(nn.Module):
    def forward(self, positions, hidden_states, forward_batch):
        # 并行块 y = x + attn(norm(x)) + mlp(norm(x))：两条链路只读同一
        # post-norm 输入，天然独立，可安全跨流并发。
        residual = hidden_states
        hidden_states = self.input_layernorm(hidden_states)
        if (
            self.mlp_stream is not None
            and get_is_capture_mode()
            and forward_batch.forward_mode.is_decode()
        ):
            # 仅 decode 启用重叠：prefill chunk 已占满 GPU，拆分只会引入
            # 事件开销。注意 get_is_capture_mode() 对 prefill 捕获同样为真，
            # 所以必须叠加 forward_mode.is_decode() 判断。
            current_stream = torch.cuda.current_stream()
            self.mlp_stream.wait_stream(current_stream)
            with torch.cuda.stream(self.mlp_stream):
                mlp_out = self.mlp(hidden_states)
            attn_out = self.self_attn(
                positions=positions,
                hidden_states=hidden_states,
                forward_batch=forward_batch,
            )
            # 两个张量都被引用到汇合点之前，无需 record_stream
            current_stream.wait_stream(self.mlp_stream)
        else:
            attn_out = self.self_attn(
                positions=positions,
                hidden_states=hidden_states,
                forward_batch=forward_batch,
            )
            mlp_out = self.mlp(hidden_states)
        combined = attn_out + mlp_out
        if self.tp_size > 1:
            combined = tensor_model_parallel_all_reduce(combined)
        return residual + combined
```

python/sglang/srt/arg_groups/model_overrides/cohere2_moe.py

新增 fail-closed 的 MoE runner 门控：SM10X 上未量化或 NVFP4 的 Command-A-Plus 自动钉到 flashinfer_trtllm，不可读量化配置与 FP8 保持 auto，是后端正确性的关键决策点

```
# ============ Command-A-Plus MoE runner 门控（fail-closed） ============
```

```
def _is_nvfp4_pack_quantized(hf_config: Any) -> bool:
    # nvfp4-pack-quantized 是 llm-compressor 的输出格式，三个官方
    # Command-A-Plus checkpoint 都只量化 experts，因此顶层 format 与
    # config_groups 中任意一个含 nvfp4 即视为 NVFP4 量化。
    qc = getattr(hf_config, 'quantization_config', None)
    if not isinstance(qc, dict):
        return False
    groups = qc.get('config_groups') or {}
    formats = [qc.get('format', '')] + [
        g.get('format', '') for g in groups.values() if isinstance(g, dict)
    ]
    return any('nvfp4' in str(fmt) for fmt in formats)


@_register_for('Cohere2VisionForConditionalGeneration', 'Cohere2MoeForCausalLM')
def _cohere2_moe_runner_overrides(server_args: Any, hf_config: Any) -> dict:
    cfg = resolving_view(server_args)
    if cfg.moe_runner_backend != 'auto':
        return {} # 用户显式指定后端时不覆盖
    if not get_platform().is_sm100:
        return {} # 门控只在 SM10X（Blackwell）生效
    if model_config_of(server_args).quantization is not None:
        # 量化且读不出 NVFP4 格式时保持 auto：flashinfer_trtllm 会在第一次
        # forward 时拒绝 CompressedTensorsW8A8Fp8MoE 的 TritonMoeQuantInfo，
        # 错误的钉死会让服务中途崩溃，所以宁可保守。
        if not _is_nvfp4_pack_quantized(hf_config):
            return {}
    logger.info(
        'Command-A-Plus on SM10X: moe_runner_backend=flashinfer_trtllm '
        '(trtllm-gen fused MoE).'
    )
    return {'moe_runner_backend': 'flashinfer_trtllm'}
```

评论区精华

评审中 kpham-sgl（含 codex 代笔意见）与作者 mmangkad 有三个回合的交锋：

1. 意图确认：kpham-sgl 问 _is_nvfp4_pack_quantized 的意图，mmangkad 说明它是为官方 w4a4 checkpoint（CohereLabs/command-a-plus-05-2026-w4a4）服务。
2. FlashInfer TRTLLM 与 LoRA/A2A 兼容性：codex 指出 flashinfer_trtllm 与普通 MoE LoRA wrapper 不兼容，且只为 moe_a2a_backend=none 注册 fused op，建议 LoRA 或非 none A2A 时跳过自动覆盖。mmangkad 反驳：w4a4 路径在 main 上已因 compressed_tensors_w4a4_nvfp4_moe.py 只选 fused runner 而 pre-existing 失败，MoE LoRA 的 assert 与后端选择无关；真正 gap 是 runner.py:164 的 guard 位于 if not lora_enabled: 内，kimi_k3、minimax_m2 同样会 mid-forward assert，应在 MoeRunner 统一修复而非每架构 skip。
3. NVFP4 group 判定：codex 提出任何 NVFP4 group 都被当作 experts 是 NVFP4 的证据，混合配置（NVFP4 attention + FP8 experts）会被误钉并首次 forward 失败，应按目标

匹配专家 group。mmangkad 回应：三个官方 Command-A-Plus checkpoint 都只量化 experts（attention、gate、vision tower 保持 BF16），该架构上 NVFP4 group 就是 expert group，反向布局不存在。

- flashinfer_trtllm 与 MoE LoRA / A2A 后端兼容性 (design): 作者论证被接受，LoRA/A2A 路径未在本 PR 修改，问题责任归属 MoeRunner 层统一修复。
- NVFP4 判定是否应按专家分组匹配 (correctness): 作者以官方 checkpoint 布局为依据说明当前检测足够，未加 group 目标匹配。
- _is_nvfp4_pack_quantized 函数意图 (question): 确认检测目标是 llm-compressor 输出的 nvfp4-pack-quantized 格式。

风险与影响

- 风险：
  1. 跨流数据竞争：多流重叠依赖 wait_stream 事件同步与 FusedMoE(inplace=False) 配合。inplace=False 是 FusedMoE 构造参数的全局变更，即使未启用多流（非 CUDA 平台、eager 非捕获路径）也生效，可能小幅影响内存峰值与吞吐，作者基准仅覆盖 SM103。
1. decode-only 判定的脆弱性：Cohere2MoeDecoderLayer.forward 依赖 forward_mode.is_decode() 区分 decode 与 prefill chunk，get_is_capture_mode() 对两者都为真；若未来 forward mode 语义变化，可能漏判或误判重叠分支。
2. 量化布局假设：门控隐含 "NVFP4 group 即 expert group" 的现网 checkpoint 布局假设。评审中 codex 已指出混合布局会被误判，作者以 " 当前三个官方 checkpoint 均无此布局 " 回应；若未来 Cohere 发布混合量化 checkpoint，flashinfer_trtllm 会在首次 forward 崩溃，且现有测试无法覆盖该场景。
3. LoRA/A2A 组合未处理：门控未显式排除 LoRA 或非 none A2A 后端。作者论证为预存在问题并建议在 MoeRunner（runner.py:164）统一修复，但本 PR 未落地，LoRA 用户仍可能踩到 assert。
4. 白名单迁移影响多模态路径：Cohere2VisionForConditionalGeneration 从 piecewise 移到 breakable，带 embed 的 batch 会在 replay 时被拒绝并退回 eager，视觉编码器仍在图外执行，需关注多模态场景回归。
5. 测试覆盖缺口：自动化测试只覆盖配置门控，多流路径的数值一致性仅有手工 gsm8k 基准，无 CI 回归测试守护。 - 影响：对用户：Command-A-Plus 在 SM10X（Blackwell）上 BCG prefill 从默认配置 OOM 变为可用，decode TPOT w4a4 降低 1.43-2.28 倍、bf16 降低 1.28-1.58 倍，TTFT 改善 15-37%（c1 时 w4a4 129→109 ms，bf16 209→140 ms），吞吐最高 9,152 tok/s（w4a4, c256）。对系统：prefill 图捕获内存从约 23 GB 降至 5.3 GB，KV cache 可在默认 --mem-fraction-static 下保留完整容量（对比场景中 0.75 让渡约 10%）。对团队：提供了 " 按角色租用流避免缓存分配器分池 " 与 " fail-closed runner 门控 + 黄金测试 " 两个可复制的工程模式，后续 MoE 模型可直接引用。其它架构 / 平台行为不变：门控 fail-closed、流改动限定 CUDA + 图捕获，唯一全局生效的是 FusedMoE(inplace=False)。 - 风险标记：核心路径多流并发，跨流数据竞争处理，量化布局依赖现网 checkpoint，LoRA/A2A 组合未覆盖，多流路径缺少自动化回归测试

关联脉络

- PR #35120 [FlashInfer v0.6.18] add FlashInfer CuTe DSL NVFP4 W4A16 mode: 同属 MoE runner 后端（flashinfer_trtllm / flashinfer_cutedsl）与 NVFP4 量化路径的演进，本 PR 的 runner 门控选择延续了这条技术路线。
- PR #34967 [MoE] Add FlashInfer SM90 MXFP4 W4A8 CUTLASS MoE: MoE runner 与量化矩阵路径的关联实现，本 PR 的 inplace=False 与门控逻辑与 MoE 后端选择机制直接相关。
- PR #37195 fix(config): retain pre-engine resolution declarations: model overrides 声明机制链路的配套修复；本 PR commit 2db4ddc 显示 main 已将 per-family 声明拆入 arg_groups/model_overrides/ 包，二者共同维护该机制的正确性。