PR #26643 完整报告

sgl-project/sglang

[DP] Fix FlashInfer dispatcher workspace sizing and set_dp_buffer_len

合并时间：2026-06-03 04:25

原文链接：http://prhub.com.cn/sgl-project/sglang/pull/26643

执行摘要

- 一句话：修复 DP 下 FlashInfer workspace 及 set_dp_buffer_len
- 推荐动作：值得精读，特别是 DP buffer 管理模式的统一方式。建议关注 set_dp_buffer_len 的签名扩展和 global_num_tokens_cpu 的传递机制，可能为后续 DP buffer 重构提供基础。

功能与动机

根据 PR 描述，FlashInfer dispatcher workspace sizing 需要反映 max(chunked_prefill_size, max_running_req) 以确保预填充批次正确分配。Review 中 ch-wan 指出应使用 max_prefill_tokens，但作者 hanming-lu 认为 chunked_prefill_size 足够，且当前测试验证通过。另外 set_dp_buffer_len 需要基于 global_dp_buffer_len 设置 global_num_tokens_cpu，原实现缺乏此传递导致潜在不一致。

实现拆解

1. 重构 global_num_tokens_gpu 填充：在 cuda_graph_runner.py 和 model_runner.py 中，将原先的分支直接 copy tensor 改为先构建 global_num_tokens_cpu 列表，再统一构造 tensor 进行 copy，减少重复代码。
2. 传递 global_num_tokens_cpu 给 set_dp_buffer_len：在 run_once 函数中，将 global_num_tokens_cpu 作为额外参数传入 set_dp_buffer_len（原函数签名扩展），使其能基于 DP 粒度正确设置 buffer。
3. 修复 FlashInfer dispatcher workspace sizing：在 flashinfer.py 的 __init__中，将 workspace 分配大小从 max_running_req 改为 max(chunked_prefill_size, max_running_req) * ep_size，并调整相关断言。
4. 同步所有 CUDA graph runner：包括 speculative decode 相关的 4 个 runner（eagle_draft、frozen_kv_mtp、multi_layer_eagle_draft_extend、eagle_draft_extend）以及 piecewise_cuda_graph_runner，均采用相同的 global_num_tokens_cpu 模式并传递新参数。

关键文件：

- python/sglang/srt/model_executor/cuda_graph_runner.py（模块 CUDA 图；类别 source；类型 data-contract；符号 capture_one_batch_size, run_once）：核心文件，重构 global_num_tokens_gpu 填充方式并引入 global_num_tokens_cpu 参数传递，影响所

有 DP gather 路径。

- python/sglang/srt/model_executor/model_runner.py（模块 模型执行；类别 source；类型 data-contract；符号 _dummy_run, run_once）：模型主执行器，同样应用 global_num_tokens_cpu 模式，与 cuda_graph_runner 对应，影响所有 CUDA graph 捕获调用。
- python/sglang/srt/speculative/eagle_draft_cuda_graph_runner.py（模块 推测草稿；类别 source；类型 core-logic；符号 capture_one_batch_size, run_once）：推测解码草稿器的 CUDA graph 捕获器，需要与主路径保持相同的 global_num_tokens_cpu 逻辑，属于 core-logic 变更。
- python/sglang/srt/layers/moe/token_dispatcher/flashinfer.py（模块 MoE 调度；类别 source；类型 core-logic；符号 init）：修复 workspace sizing 计算，将 max_running_req 改为 max(chunked_prefill_size, max_running_req)，确保预填充 batch 有足够空间。
- python/sglang/srt/model_executor/piecewise_cuda_graph_runner.py（模块 分段图；类别 source；类型 data-contract；符号 run_once）：piecewise CUDA graph 捕获器，需要同步 global_num_tokens_cpu 参数传递，改动较小但体现完整性。

关键符号：capture_one_batch_size, run_once, set_dp_buffer_len, get_forward_batch, _dummy_run, init

关键源码片段

python/sglang/srt/model_executor/cuda_graph_runner.py

核心文件，重构 global_num_tokens_gpu 填充方式并引入 global_num_tokens_cpu 参数传递，影响所有 DP gather 路径。

```
# capture_one_batch_size 中统一计算 global_num_tokens_cpu
if self.require_mlp_tp_gather:
    global_num_tokens_cpu = [num_tokens] * self.dp_size
elif self.require_attn_tp_gather:
    global_num_tokens_cpu = [num_tokens]
else:
    global_num_tokens_cpu = None

# 基于 global_num_tokens_cpu 构造 tensor 并填充 buffer
if global_num_tokens_cpu is not None:
    global_dp_buffer_len = sum(global_num_tokens_cpu)
    num_tokens_tensor = torch.tensor(
        global_num_tokens_cpu, dtype=torch.int32, device=input_ids.device
    )
    buffers.global_num_tokens_gpu.copy_(num_tokens_tensor)
    buffers.global_num_tokens_for_logprob_gpu.copy_(num_tokens_tensor)
else:
    global_dp_buffer_len = None

# 在 run_once 中传递 global_num_tokens_cpu 给 set_dp_buffer_len
```

```
    set_dp_buffer_len(
        global_dp_buffer_len,
        num_tokens,
        forward_batch.dp_padding_mode.is_max_len(),
        global_num_tokens_cpu, # 新增参数
    )
```

python/sglang/srt/layers/moe/token_dispatcher/flashinfer.py

修复 workspace sizing 计算，将 max_running_req 改为 max(chunked_prefill_size, max_running_req)，确保预填充 batch 有足够空间。

```
    # flashinfer.py 构造函数片段
    if self.payload_in_workspace:
        # workspace 大小取 prefill 与 decode 最大值
        self.workspace_allocated_size = (
            max(self.chunked_prefill_size, self.max_running_req) * self.ep_size
        )
        # 原实现仅用 max_running_req，现在兼容预填充大 batch
        self.workspace_buffer = torch.empty(
            self.workspace_allocated_size,
            dtype=torch.int32,
            device="cuda",
        )
        assert self.workspace_buffer.numel() > 0
```

评论区精华

- ch-wan 评论建议复用 cuda_graph_runner 中已有代码逻辑（L986-L1019），hanming-lu 回复 "good point. Done"，随后采纳。
- ch-wan 对 flashinfer.py 评论 Let's use max_prefill_tokens. chunked_prefill_size only restricts per-request input size.，hanming-lu 回应 "IIUC the buffer here is used per DP rank, so chunked_prefill_size is the max that can get prefilled per DP rank, so should be sufficient." 并保留当前方案。
- FlashInfer workspace 大小依据 (correctness): hanming-lu 认为 per DP rank 下 chunked_prefill_size 已足够，保留当前逻辑。
- cuda_graph_runner 代码复用 (design): hanming-lu 同意并实施统一模式。

风险与影响

- 风险：
  1. CUDA graph 稳定性：capture_one_batch_size 路径改动可能引入 GPU 内存访问错误，尤其是在 spec decode 联合使用时。
  2. Workspace 显存增加：max(chunked_prefill_size, max_running_req) 可能显著增大 workspace，尤其是 chunked_prefill_size 较大时，需确认单次推理峰值。
  3. 多路径一致性：5 个 CUDA graph runner 有类似的代码复制，未来修改需同步所有文件，维护成本高。

4. 缺少测试覆盖：PR 未增加对应测试文件，DP 路径的回归风险由 CI 中已有测试承担。

- 影响：
  - 用户：影响使用 Data Parallel（DP）与 FlashInfer MoE 推理的场景，预填充时 workspace 不足错误得到修复，提升可靠性。
  - 系统：set_dp_buffer_len 现在能正确接收 DP 维度的 buffer 长度，减少 *decode* 时 buffer 大小不匹配的概率。
  - 团队：代码重复性降低（但仍存在），后续类似修改应建立中心化工具函数。
  - 风险标记：核心路径变更，缺少测试覆盖，多路径一致性问题，显存占用增加

关联脉络

- 暂无明显关联 PR