PR #28677 完整报告

sgl-project/sglang

fix(runner): size eager static buffers for prefill budget and MLP-sync autotune

合并时间：2026-06-20 04:23

原文链接：http://prhub.com.cn/sgl-project/sglang/pull/28677

执行摘要

- 一句话：修复 eager runner 静态缓冲区尺寸不足问题
- 推荐动作：建议阅读。虽然改动较小，但修复了两个不易察觉的边界条件 bug，体现了作者对调度器逻辑的深入理解。max_prefill_buffer_tokens() 的设计可作为后续类似缓冲区尺寸计算的模式参考，值得精读以理解 PP dynamic chunking 和 MLP sync 的交互细节。

功能与动机

PR body 指出 eager runner 的静态缓冲区在 token 维度和 batch 维度上尺寸不足。Token 维度：prefill 缓冲区按 chunked_prefill_size 分配，但在 PP dynamic chunking 下运行时 predictor 可增长 chunk 到 max_prefill_tokens，且 latency profiler 会以 1.25 倍探测，导致溢出并引发“Profiling prefill latency for dynamic chunking”挂起。Batch 维度：FlashInfer autotune dummy forward 使用 eager bs 上限 max_running_requests，但 MLP sync 下调度器在 prepare_mlp_sync_batch 中对实际 batch 进行 ceil 对齐，使得注册表尺寸小于 load_batch 实际拷贝的数据，导致 autotune 以错误形状调优。

实现拆解

1. 新增 max_prefill_buffer_tokens() 方法：在 server_args.py 的 ServerArgs 类中添加该方法，作为 prefill 缓冲区 token 上限的唯一来源。非动态分块时返回 chunked_prefill_size；仅在 PP deep mind 开启动态分块且 chunked_prefill_size > 0 时，取 chunked_prefill_size、max_prefill_tokens、ceil(chunked_prefill_size * 1.25) 中的最大值，避免了非动态配置下的过度分配。
2. 修改 eager_runner.py 的缓冲区初始化逻辑：在 EagerRunner.__init__ 中将 prefill 上限由 sa.chunked_prefill_size 替换为 sa.max_prefill_buffer_tokens()；在 MLP sync 开启时，对 max_bs 和 max_num_token 分别应用与 prepare_mlp_sync_batch 相同的对齐逻辑：ceil_align 到 attn_tp_size，再 ceil_align 到 get_cp_padding_align_size()。这样保证了 autotune dummy forward 使用的形状与实际调度器产生的 batch 一致。
3. 新增导入依赖：在 eager_runner.py 中新增 from sglang.srt.utils.common import ceil_align, require_mlp_sync，并惰性导入 get_cp_padding_align_size 以避免循环依赖。

关键文件：

- python/sglang/srt/server_args.py（模块 参数配置；类别 source；类型 core-logic；符号 max_prefill_buffer_tokens）：新增 max_prefill_buffer_tokens() 方法作为 prefill 缓冲区

token 上限的唯一事实来源，是本次修复的核心。

- python/sglang/srt/model_executor/runner/eager_runner.py（模块 调度执行器；类别 source；类型 core-logic；符号 EagerRunner.init）：修改 __init__ 中缓冲区尺寸计算，使用新方法并添加 MLP sync 对齐逻辑，是修复生效的关键执行者。

关键符号：max_prefill_buffer_tokens, EagerRunner.init

关键源码片段

python/sglang/srt/server_args.py

新增 max_prefill_buffer_tokens() 方法作为 prefill 缓冲区 token 上限的唯一事实来源，是本次修复的核心。

```
def max_prefill_buffer_tokens(self) -> int:
    """Prefill-buffer ceiling: chunked_prefill_size, except PP dynamic
    chunking can grow chunks toward max_prefill_tokens and probe at 1.25x.
    """
    # 获取 chunked_prefill_size 的值，若为 None 或 <=0 则视为 0
    chunked = (
        self.chunked_prefill_size
        if self.chunked_prefill_size and self.chunked_prefill_size > 0
        else 0
    )
    tokens = chunked
    # 仅当启用了动态分块、PP 规模 >1 且 chunked_prefill_size 有效时，
    # 才放宽上限到 max_prefill_tokens 或 chunked_prefill_size 的 1.25 倍
    if self.enable_dynamic_chunking and self.pp_size > 1 and chunked:
        tokens = max(
            tokens, self.max_prefill_tokens or 0, math.ceil(chunked * 1.25)
        )
    return tokens
```

python/sglang/srt/model_executor/runner/eager_runner.py

修改 __init__ 中缓冲区尺寸计算，使用新方法并添加 MLP sync 对齐逻辑，是修复生效的关键执行者。

```
# 以下为 EagerRunner.__init__ 中缓冲区尺寸计算的核心变更片段
# (... 省略上下文 ...)
# 镜像 prepare_mlp_sync_batch 的对齐逻辑，使注册表能够容纳 load_batch 实际拷贝的数据
if require_mlp_sync(sa):
    from sglang.srt.layers.utils.cp_utils import get_cp_padding_align_size
    # 先对齐 attn_tp_size，再对齐 CP 对齐尺寸
    max_bs = ceil_align(max_bs, self.attn_tp_size)
    max_bs = ceil_align(max_bs, get_cp_padding_align_size())
# 使用新增的 max_prefill_buffer_tokens() 作为 prefill 上限
prefill_ceiling = (
    sa.max_prefill_buffer_tokens()
    if sa.chunked_prefill_size and sa.chunked_prefill_size > 0
    else mr.max_total_num_tokens
```

```
)
max_num_token = max(prefill_ceiling, max_bs * num_tokens_per_bs)
if require_mlp_sync(sa):
    # 对 total token 数也应用相同的对齐，避免 prefill chunk 超限
    max_num_token = ceil_align(max_num_token, self.attn_tp_size)
    max_num_token = ceil_align(max_num_token, get_cp_padding_align_size())
# (... 剩余代码 ...)
```

评论区精华

Review 中 bot 提出了三点建议：

- 在 server_args.py 中，建议避免在非动态分块时将 prefill 缓冲区尺寸设为 max_prefill_tokens，因为 PrefillAdder 仍然受 chunked_prefill_size 约束。作者已采纳，最终实现中仅当 enable_dynamic_chunking and pp_size > 1 时才放宽上限。
- 在 eager_runner.py 中，建议镜像完整的 MLP sync padding 序列（包括 CP 对齐）。作者已采纳，在 max_bs 和 max_num_token 上均增加了 CP 对齐步骤。
- 在同一文件中，建议对 prefill token 上限也应用对齐。作者已采纳，对 max_num_token 添加了与 batch 相同的对齐逻辑。
- 非动态分块时应避免使用 max_prefill_tokens (correctness): 作者采纳，最终实现中仅在 enable_dynamic_chunking 且 pp_size > 1 时才放宽到 max_prefill_tokens 和 1.25 倍。
- 镜像完整的 MLP sync padding 序列 (correctness): 作者采纳，在 max_bs 对齐中加入了 CP 对齐步骤。
- prefill token 上限也应对齐 (correctness): 作者采纳，对 max_num_token 添加了与 max_bs 相同的对齐逻辑。

风险与影响

- 风险：风险较低。主要变更集中在非主流的 PP dynamic chunking 和 MLP sync 配置路径上，默认无动态分块时逻辑不变。由于未添加测试覆盖，回归需依赖现有 CI 测试验证。此 PR 修改了 eager runner 的缓冲区尺寸，若计算错误可能导致显存浪费或仍存在溢出，但由于从同一逻辑（prepare_mlp_sync_batch）衍生，一致性有保障。
- 影响：影响范围有限，仅影响启用 PP dynamic chunking 或 MLP sync 的部署配置。修复了两个在特定配置下可能引起运行时挂起或 autotune 错误的 bug，提升了系统稳定性。用户无需更改配置，升级后自动受益。团队后续维护时应注意 max_prefill_buffer_tokens() 随未来其他特性（如更激进的 chunk 增长）同步更新。
- 风险标记：缺少测试覆盖

关联脉络

- PR #28388 依赖关联（PR body 提及 stacked on）：PR 在 body 中注明 Stacked on #28388，表明此 PR 依赖 #28388 的实现。