PR #7358 完整报告

verl-project/verl

[megatron] feat: bucket packed sequence lengths

合并时间：2026-08-11 23:10

原文链接：http://prhub.com.cn/verl-project/verl/pull/7358

执行摘要

- 一句话：Megatron 新增打包序列长度分桶配置
- 推荐动作：建议精读：该 PR 展示了一个典型的“配置驱动 + 全链路透传 + 显式拒绝不兼容组合”的工程模式，尤其是 math.lcm 统一处理 bucket、CP 与 FP8 对齐的做法值得借鉴。若你在使用 Megatron 引擎且受 THD 形状抖动造成的编译开销困扰，可以开启 pad_to_length 验证收益。

功能与动机

PR body 明确说明：Adds FSDP-compatible pad_to_length controls to the Megatron engine and optionally rounds each packed THD micro-batch to a 512-token bucket. This reduces packed-shape variability and associated JIT/compilation overhead while leaving existing behavior unchanged by default (pad_to_length: false). Megatron 引擎此前缺少与 FSDP 侧一致的序列长度对齐控件，打包序列的全局形状频繁变化会导致 JIT 编译反复触发，影响训练吞吐；同时作者确认与现有 PR 无重复。

实现拆解

1. 新增引擎配置：在 verl/workers/config/engine.py 的 McoreEngineConfig 中新增 pad_to_length（默认 False）与 pad_to_length_bucket（默认 512），并在 verl/trainer/config/engine/megatron.yaml 及生成的 _generated_ppo_megatron_trainer.yaml 中同步声明，保证配置解析与文档一致。
2. 核心分桶逻辑：在 verl/models/mcore/util.py 的 preprocess_thd_engine 中新增 pad_to_length_bucket 可选参数；在完成常规序列对齐与 FP8 总长对齐后，以 math.lcm(pad_to_length_bucket, cp_size) 计算全局对齐单位，并进一步与 FP8 的 total_align 取 LCM，将 cu_seqlens_padded 与 seqlens_in_batch_padded 的最后一项补足，使全局打包长度落在固定格点上。
3. 全链路参数透传：在 verl/models/mcore/model_forward.py 的 gptmodel_forward_model_engine、verl/models/mcore/model_forward_fused.py 的 fused_forward_model_engine_inner，以及 verl/workers/engine/megatron/transformer_impl.py 的 MegatronEngineWithLMHead.forward_step / MegatronEngineWithValueHead.forward_step 中，将 pad_to_length_bucket 传递给所有 preprocess_thd_engine 调用点，覆盖标准、fused、参考模型与值模型路径。

4. 组合约束校验：在 _check_dcp_unsupported_features 中拒绝 dynamic_context_parallel 与 pad_to_length 同时开启；在 forward_step 中拒绝与 top-K 蒸馏、router replay 的组合，报错信息明确说明原因，避免静默产生错误结果。
5. 测试与配置配套：在 tests/utils/test_megatron_bshd_preprocess.py 新增 513 token 分桶至 1024 的单元测试（CP=1 场景）；同步更新 test_model_forward_fused.py、test_dynamic_cp_scheduler.py、test_megatron_value_head_cp_layout_on_cpu.py 中的测试替身参数，并重新生成 trainer 配置。

关键文件：

- verl/models/mcore/util.py（模块 序列预处理；类别 source；类型 data-contract；符号 preprocess_thd_engine）：核心实现：preprocess_thd_engine 新增 pad_to_length_bucket 分桶对齐逻辑，是全局打包长度落在固定格点的关键。
- verl/workers/engine/megatron/transformer_impl.py（模块 引擎执行；类别 source；类型 core-logic；符号 _check_dcp_unsupported_features, MegatronEngineWithLMHead.forward_step, MegatronEngineWithValueHead.forward_step）：引擎主控：forward_step 解析 pad_to_length 配置并校验与蒸馏、router replay 的组合；同时 DCP 校验拒绝该特性。
- verl/workers/config/engine.py（模块 引擎配置；类别 source；类型 configuration；符号 McoreEngineConfig）：配置契约：McoreEngineConfig 新增 pad_to_length 与 pad_to_length_bucket 字段，默认值保证向后兼容。
- verl/models/mcore/model_forward.py（模块 模型前向；类别 source；类型 data-contract；符号 gptmodel_forward_model_engine）：标准 forward 路径：gptmodel_forward_model_engine 将参数透传给 preprocess_thd_engine 的 input_ids、label、loss_mask 等所有调用点。
- verl/models/mcore/model_forward_fused.py（模块 融合前向；类别 source；类型 data-contract；符号 fused_forward_model_engine_inner）：Fused forward 路径：fused_forward_model_engine_inner 同步透传参数，保证 use_fused_kernels=True 时分桶行为一致。
- tests/utils/test_megatron_bshd_preprocess.py（模块 单元测试；类别 test；类型 test-coverage；符号 test_preprocess_thd_engine_rounds_packed_length_to_bucket_without_cp）：核心单元测试：验证 513 token 在 bucket=512 且 CP=1 时被取整到 1024。
- verl/trainer/config/engine/megatron.yaml（模块 训练配置；类别 config；类型 configuration）：默认配置：声明 pad_to_length 与 pad_to_length_bucket，并注明与 FSDP 控件对齐的动机。

关键符号：preprocess_thd_engine, gptmodel_forward_model_engine, fused_forward_model_engine_inner, MegatronEngineWithLMHead.forward_step, MegatronEngineWithValueHead.forward_step, _check_dcp_unsupported_features

关键源码片段

verl/models/mcore/util.py

核心实现：preprocess_thd_engine 新增 pad_to_length_bucket 分桶对齐逻辑，是全局打包长度落在固定格点的关键。

```
# 核心函数：将嵌套 THD 序列打包并按需分桶对齐
def preprocess_thd_engine(
    input_ids: torch.Tensor,
    pre_process: bool = True,
    need_roll: bool = False,
    use_fp8_padding: bool = False,
    local_cp_size: int | None = None,
    min_local_rows: int | None = None,
    pad_to_length_bucket: int | None = None, # 新增：全局打包长度的存储桶大小
    cp_layout: ContextParallelLayout = "zigzag",
):
    # ... 前面解析 TP/CP 拓扑、计算 align_size 的代码省略 ...

    # 逐条序列补齐到 align_size，保证每条序列可被 CP 整除
    pad_size = (align_size - seqlens_in_batch % align_size) % align_size
    seqlens_in_batch_padded = seqlens_in_batch + pad_size
    cu_seqlens_padded[1:] = torch.cumsum(seqlens_in_batch_padded, dim=0)

    # FP8 额外要求：总长度可被 total_align 整除
    if use_fp8_padding:
        pad_size_last = (total_align - cu_seqlens_padded[-1] % total_align) % total_align
        cu_seqlens_padded[-1] += pad_size_last
        seqlens_in_batch_padded[-1] += pad_size_last

    # 存储桶分桶：把全局打包长度向上取整到 bucket 与 cp_size 的 LCM
    # 这样无论是否开启 CP，都能保证形状落在固定格点上，从而减少 JIT 重编译
    if pad_to_length_bucket is not None:
        if pad_to_length_bucket <= 0:
            raise ValueError("pad_to_length_bucket must be a positive integer")
        total_alignment = math.lcm(pad_to_length_bucket, cp_size)
        if use_fp8_padding:
            # 与 FP8 的对齐要求再取一次 LCM，避免两种 padding 冲突
            total_alignment = math.lcm(total_alignment, total_align)
        pad_size_last = (-cu_seqlens_padded[-1]) % total_alignment
        cu_seqlens_padded[-1] += pad_size_last
        seqlens_in_batch_padded[-1] += pad_size_last
```

verl/workers/engine/megatron/transformer_impl.py

引擎主控：forward_step 解析 pad_to_length 配置并校验与蒸馏、router replay 的组合；同时 DCP 校验拒绝该特性。

```
# forward_step 中根据配置解析有效的 bucket，并拒绝不支持的组合
pad_to_length_bucket = (
    self.engine_config.pad_to_length_bucket
    if self.engine_config.pad_to_length and self.engine_config.use_remove_padding
    else None
```

```
)

# 分桶与依赖动态路由的机制不兼容：padding 会改变路由掩码与蒸馏标签的对齐语义
if pad_to_length_bucket is not None and distillation_use_topk:
    raise RuntimeError("pad_to_length is not supported with top-K distillation")
if pad_to_length_bucket is not None and self.enable_routing_replay:
    raise RuntimeError("pad_to_length is not supported with router replay")

# 动态 CP 依赖每微批次的真实长度做调度，固定 bucket 会破坏其假设
# 对应校验位于模块级函数 _check_dcp_unsupported_features 中：
# if engine_config.pad_to_length:
# raise NotImplementedError("dynamic_context_parallel does not support pad_to_length")
```

评论区精华

在 verl/models/mcore/util.py 第 406 行，审核者 wuxibin89 提问："Does pad_to_length_bucket need cp_size>1?"（分桶逻辑是否需要要求 CP 大于 1）。作者 ISEEKYAN 回复："No. Bucket padding is useful independently of context parallelism. With CP=1, lcm(512, 1) = 512, so a packed length of 513 is padded to 1024. With CP>1, the LCM keeps the global length divisible by both the configured bucket and CP size. Commit 01734cc4 makes the CP=1 regression case explicit." 结论：分桶对 CP=1 同样有意义，LCM 设计自然覆盖两种情形；提交 01734cc4 将 CP=1 的回归场景固化进测试。

- pad_to_length_bucket 是否需要 cp_size>1 限制 (design): 不需要 cp_size>1 限制；LCM 设计同时覆盖 CP=1 与 CP>1，已补 CP=1 测试。

风险与影响

- 风险：
  - 回归风险：pad_to_length 默认 False，现有行为不变；但新增参数在所有 forward 路径中逐层透传，若未来新增调用点漏传 pad_to_length_bucket，会导致打包长度不一致或功能静默失效，需要依赖测试保障。
  - 性能与显存：开启后每个微批次的全局打包长度最多会多补近一个 bucket（默认 512 token），带来无效计算与显存开销；在长序列大 batch 场景下收益与开销需要用户根据 shape 分布权衡。
  - 兼容性：与动态上下文并行（dynamic_context_parallel）、top-K 蒸馏、router replay 的组合会显式抛错拒绝，避免错误语义；FP8 padding 路径通过 LCM 融合了 total_align，但该组合缺少自动化测试。
  - 测试覆盖：新增单元测试仅覆盖 CP=1 的 513→1024 场景，CP>1、FP8、DCP 拒绝路径均无测试覆盖。
  - 影响：影响范围限于 Megatron 引擎的 THD 打包前向路径，涉及 preprocess_thd_engine 函数签名（新增可选参数，默认 None 保持兼容）、McoreEngineConfig 配置类、标准 /fused/ 参考 / 值模型四条 forward 链路，以及生成的 trainer YAML。默认不开启，对现有用户无影响；开启后受益于减少 packed shape 变异导致的 JIT 重编译，适合序列长度分布宽、训练步数多的场景。对团队的后续影响：

任何新增的 Megatron forward 入口都必须感知并传递该参数。

- 风险标记：默认关闭不改变现有行为，开启后增加 padding 计算与显存开销，与 DCP/蒸馏 /router replay 不兼容，新增参数需全链路透传，测试仅覆盖 CP=1 场景

关联脉络

- PR #7272 [fsdp,veomni] feat: support pad_to_length to reduce jit compile time: FSDP/VeOmni 侧同功能控件，本 PR 为 Megatron 补齐相同配置。
- PR #6555 [megatron] feat: add dynamic context parallel scheduling: 动态 CP 调度与本 PR 的分桶策略冲突，PR 中显式拒绝两者组合。
- PR #7261 [megatron] fix: pad multidimensional THD tensors along the sequence dimension: 同一核心工具函数 util.py 的 THD padding 修复，本 PR 在其基础上扩展全局分桶对齐。