PR #46727 完整报告

vllm-project/vllm

[Feat] Support thinking_token_budget in Model Runner V2

合并时间：2026-08-07 11:36

原文链接：http://prhub.com.cn/vllm-project/vllm/pull/46727

执行摘要

- 一句话：V2 采样链路支持 thinking_token_budget 强制收尾
- 推荐动作：值得精读。核心看 vllm/v1/worker/gpu/sample/thinking_budget.py 的两个 Triton kernel：增量扫描缓存 + 冷扫描块向量化的设计、_load_effective_token 对 committed 与 spec-draft token 的统一读取，以及 forced-end 置 1e9 而非全 vocab 覆写的取舍。njhill 最终提交（clamp、UVA 脏标记、prompt_len 去除）也是很好的 MRV2 状态类优化范本。同时应关注讨论中未解决的 parser 隐式结束 gap，它会在 ParserEngine 迁移时再次触碰同一逻辑。

功能与动机

PR body 明确动机："In some quantized models, such as GLM-5.2 or Qwen quantized models, the model may generate long reasoning traces." 量化模型的长思考链会显著浪费算力与响应延迟，thinking_token_budget 允许用户在推理 token 达到预算后被强制收尾。此前该能力只在 V1 Model Runner 中可用，V2 入口在 input_processor._validate_params 直接抛 VLLMValidationError、config/vllm.py 打印 warning，因此本 PR 的目标就是 "Support thinking_token_budget in Model Runner V2"，并补齐 GPU 侧扫描与强制结束的实现。

实现拆解

1. 配置契约扩展（vllm/config/reasoning.py）：ReasoningConfig 新增 _natural_reasoning_end_token_ids 私有字段与 natural_reasoning_end_token_ids 属性；initialize_token_ids 优先从 reasoning_parser 取 end_token 作为自然结束标记，未配置 parser 时回退到 reasoning_end_str。由此 reasoning_end_token_ids 的语义变为"预算耗尽时强制的结束短语"，可与模型自然结束标记不同（例如包含过渡句子）。
2. 核心状态模块（新增 vllm/v1/worker/gpu/sample/thinking_budget.py 的 ThinkingBudgetState）：__init__先将 start / end / natural-end 三类标记 token ids 落成 GPU tensor，enabled 为三者同时存在，未启用直接 return 不分配任何 tensor；per-request budget 存于 UvaBackedTensor，add_request 中 clamp 到 int32 并记录 _reset_reqs，apply_staged_writes 在 reset 时清空增量扫描缓存、仅在 budget 实际变化时 copy_to_uva。
3. GPU 扫描与强制收尾：_update_committed_marker_cache_kernel 维护 cached_last_start / cached_last_end / cached_scan_pos 缓存，增量 decode 只扫描新增 token，冷启动或冷恢复时以 1024 块从历史末尾反向回扫；_thinking_budget_kernel

处理 expanded 视角（含 spec-decode draft token），借助 _load_effective_token 混合读取 all_token_ids 与 input_ids，预算耗尽时仅将 reasoning_end_token_ids 对应 logit置 1e9，避免全 vocab 覆写。

4. Sampler 接线与校验放开（sampler.py、input_processor.py、config/vllm.py、model_runner.py）：Sampler 接收 reasoning_config 并创建 thinking_budget_state，add_request / apply_staged_writes 同步透传，apply_sampling_params 在 apply_temperature 之前调用 thinking_budget_state.apply 保证强制 logit 不受 temperature 缩放影响；__call__与 rejection_sampler.py 的 _verify 同步补传 idx_mapping；input_processor 删除"V2 不支持"分支，config/vllm.py 删除 warning，model_runner.py 构造 Sampler 时透传 vllm_config.reasoning_config。
5. 测试与基准配套：新增 tests/v1/worker/test_gpu_thinking_budget.py（CUDA + triton 条件跳过）覆盖预算耗尽强制 end、多 token end 标记续写、独立 natural end 停止强制、prefill 已结束不强制、增量扫描不重扫全史、超大 budget clamp 不崩溃等场景；新增 benchmarks/kernels/benchmark_thinking_budget.py 提供 cached / incremental-decode / cold-prefill / cold-resume-worst-case / forced-end / batched-budgeted / batched-mixed 七种模式并支持 --max-slowdown 断言；tests/entrypoints/openai/chat_completion/test_thinking_token_budget.py 移除 VLLM_USE_V2_MODEL_RUNNER=0，三个 server fixture 默认使用 V2。

关键文件：

- vllm/v1/worker/gpu/sample/thinking_budget.py（模块 采样器；类别 source；类型 core-logic；符号 ThinkingBudgetState, init, add_request, apply_staged_writes）：PR 的核心新增模块：ThinkingBudgetState 管理 per-request thinking 预算，两个 Triton kernel 分别负责已提交 token 的增量 / 冷扫描标记缓存与逐 token 预算判定、强制结束，所有关键设计都在此文件。
- vllm/config/reasoning.py（模块 配置；类别 source；类型 core-logic；符号 ReasoningConfig, initialize_token_ids, natural_reasoning_end_token_ids）：扩展 ReasoningConfig 配置契约，新增 natural_reasoning_end_token_ids，区分模型自然结束标记与预算耗尽时的强制结束短语，是 thinking budget 语义正确性的基础。
- vllm/v1/worker/gpu/sample/sampler.py（模块 采样器；类别 source；类型 dependency-wiring；符号 Sampler, init, add_request, apply_staged_writes）：Sampler 是 ThinkingBudgetState 的接入点：构造、add_request、apply_staged_writes 生命周期透传，并在 apply_sampling_params 中 temperature 之前调用 budget 强制逻辑，决定强制 logit 的生效顺序。
- benchmarks/kernels/benchmark_thinking_budget.py（模块 基准脚本；类别 source；类型 benchmark；符号 ReasoningConfig, BenchmarkCase, create_case, benchmark_cached）：新增 kernel 级性能回归基准，响应 rishitdholakia13 对长生成性能退化的担忧；覆盖 cached / incremental-decode / cold / forced-end / batched 等模式，并支持 --max-slowdown 作为 CI 断言。
- tests/v1/worker/test_gpu_thinking_budget.py（模块 单元测试；类别 test；类型 test-coverage；符号 MockReasoningConfig, MockMultiTokenEndReasoningConfig, MockDistinctEndReasoningConfig, _make_req_states）：GPU 单元测试覆盖预算耗尽强

制、多 token 结束标记、独立 natural end、prefill 已结束不强制、增量扫描不重扫全史、超大 budget clamp 等关键行为，是正确定性的主要保障。

- vllm/v1/engine/input_processor.py（模块 输入校验；类别 source；类型 core-logic；符号 InputProcessor._validate_params）：移除 V2 model runner 不支持 thinking_token_budget 的 VLLMValidationError，是功能放开的入口校验变化，仅保留未配置 reasoning_config 的报错。
- vllm/v1/worker/gpu/model_runner.py（模块 模型执行器；类别 source；类型 data-contract；符号 ModelRunner.load_model）：数据契约接入点：构造 Sampler 时透传 vllm_config.reasoning_config，使 ThinkingBudgetState 能拿到标记 token ids。
- vllm/v1/worker/gpu/spec_decode/rejection_sampler.py（模块 投机解码；类别 source；类型 data-contract；符号 RejectionSampler._verify）：spec-decode 拒采路径同步补传 idx_mapping，保证 thinking budget 的 expanded 索引在 reject/accept 后仍正确，是采样链一致性的关键配套。

关键符号：ThinkingBudgetState.init, ThinkingBudgetState.add_request, ThinkingBudgetState.apply_staged_writes, ThinkingBudgetState.apply, _update_committed_marker_cache_kernel, _thinking_budget_kernel, _load_effective_token, ReasoningConfig.initialize_token_ids, Sampler.apply_sampling_params, Sampler.call

关键源码片段

vllm/config/reasoning.py

扩展 ReasoningConfig 配置契约，新增 natural_reasoning_end_token_ids，区分模型自然结束标记与预算耗尽时的强制结束短语，是 thinking budget 语义正确性的基础。

```
# vllm/config/reasoning.py
# ReasoningConfig 原先只有一个 ending 概念；本 PR 拆成两套：
# - natural_reasoning_end_token_ids：parser 定义的自然结束标记（模型自己会输出）
# - reasoning_end_token_ids：预算耗尽时强制插入的结束短语（可含过渡句子）
# 这样强制短语不会在模型自然结束时被重复插入，也不会因预算截断而缺失。

def initialize_token_ids(self, model_config):
    if (self._reasoning_start_token_ids is not None
            and self._reasoning_end_token_ids is not None
            and self._natural_reasoning_end_token_ids is not None):
        self._enabled = True
        return # 已经初始化过则跳过

    tokenizer = cached_tokenizer_from_config(model_config=model_config)
    reasoning_start_str = self.reasoning_start_str
    reasoning_end_str = self.reasoning_end_str
    natural_reasoning_end_str = ""
    if self.reasoning_parser:
        parser_cls = ReasoningParserManager.get_reasoning_parser(self.reasoning_parser)
        reasoning_parser = parser_cls(tokenizer)
```

```
            start_token = reasoning_parser.reasoning_start_str
            if start_token and not reasoning_start_str:
                reasoning_start_str = start_token
            end_token = reasoning_parser.reasoning_end_str
            if end_token and not reasoning_end_str:
                reasoning_end_str = end_token
            # parser 的 end 字符串即"自然结束"标记，与强制结束短语区分开
            natural_reasoning_end_str = end_token or ""

        if not natural_reasoning_end_str:
            # 未配置 parser 时退化为 equal reasoning_end_str，保持旧行为
            natural_reasoning_end_str = reasoning_end_str

        if not reasoning_start_str or not reasoning_end_str:
            return
        self._reasoning_start_token_ids = tokenizer.encode(
            reasoning_start_str, add_special_tokens=False)
        self._reasoning_end_token_ids = tokenizer.encode(
            reasoning_end_str, add_special_tokens=False)
        self._natural_reasoning_end_token_ids = tokenizer.encode(
            natural_reasoning_end_str, add_special_tokens=False)
        # 三个字段任一为空都视为配置失败，抛出 ValueError
```

vllm/v1/worker/gpu/sample/sampler.py

Sampler 是 ThinkingBudgetState 的接入点：构造、add_request、apply_staged_writes 生命周期透传，并在 apply_sampling_params 中 temperature 之前调用 budget 强制逻辑，决定强制 logit 的生效顺序。

```
# vllm/v1/worker/gpu/sample/sampler.py
# 接线点：thinking budget 处理必须放在 temperature 之前，
# 这样 1e9 的强制 logit 不会被 temperature 缩放抵消，
# 保证预算耗尽后必然采样到 reasoning 结束标记。

def apply_sampling_params(
    self, logits, expanded_idx_mapping, idx_mapping,
    idx_mapping_np, pos, input_ids, expanded_local_pos, return_logprobs,
):
    # ... 先应用 penalties / logit_bias / bad_words 等 state ...

    # Force the reasoning end marker once a request's thinking budget is
    # reached; applied before temperature so the forced token is always kept.
    self.thinking_budget_state.apply(
        logits,
        expanded_idx_mapping,
        idx_mapping,
        idx_mapping_np,
        input_ids,
        expanded_local_pos,
    )
```

```
# Apply temperature in place.
self.sampling_states.apply_temperature(
    logits, expanded_idx_mapping, idx_mapping_np)
```

评论区精华

njhill 是主要评审者，认可方向后重点推动扫描优化："我们每次都需要扫描整个 context，其实只需要考虑新增 token"，并最终亲自提交了增量扫描、冷扫描 1024 块向量化（64k 历史 worst-case 约 34x 提速）、budget int32 clamp 等修复。rishitdholakia13 提出两个性能建议：128k prefill 正序扫描是 O(128k)，应反向扫描以降低 TTFT；forced-end 不必整块 logits 写 -inf，只需把结束 token 提升到 1e9，chaunceyjiang 表示 make sense 并落地。dafeliton 报告了 Gemma 场景强制结束短语泄漏进主响应的 bug，修复后确认 works much better，但又指出 parser 隐式结束（reasoning 直接切到 <|tool_call> 不发结束符）仍未覆盖，chaunceyjiang 承认该 gap 留待 tool parser 迁移到 ParserEngine 后解决。njhill 还建议删除 remove_request 以保持 state 类一致性，以及未启用时不分配 tensor。

- decode 步每次全量重扫 context 的优化 (performance): 演进为 cached_scan_pos 增量扫描 + 冷扫描 1024 块向量化；njhill 最终提交记录显示 64k 历史下 cold resume worst-case 约 34x 提速。
- prefill 长上下文应反向扫描标记 (performance): 冷扫描改为从 history 末尾按 1024 块回退扫描；增量 decode 只扫新 token，prefill 结束后 cached_scan_pos 更新到当前长度。
- 避免全 vocab logits 覆写 (performance): chaunceyjiang 采纳（make sense），提交 Optimize thinking budget forcing by avoiding full-vocab logits overwrite。
- remove_request 与 state 类一致性 (design): 按建议移除，ThinkingBudgetState 不单独管理移除逻辑。
- 未启用时不应分配 tensor (design): 最终 __init__先计算 enabled 再分配，未启用直接 return。
- 强制结束短语泄漏进主响应 (correctness): 作者修复后 dafeliton 确认 works much better；问题源于强制短语与自然结束标记的判断边界，修复后仍遗留隐式结束 gap。
- 隐式 parser 结束（tool_call 直接截断推理）未覆盖 (correctness): chaunceyjiang 承认问题，计划等多数 tool parser 迁移到 ParserEngine 后再统一处理；本 PR 未修复。
- thinking budget 性能回归测试 (testing): 新增 benchmarks/kernels/benchmark_thinking_budget.py，支持 --max-slowdown 让 CI 可断言缓存与增量 decode 路径的延迟不随 history 长度退化。

风险与影响

- 风险：
  1. 隐式结束 gap（未解决）：V2 budget 状态只扫描 natural_reasoning_end_token_ids，Gemma 等模型从 reasoning 直接切到 <|tool_call> 时不输出结束符，长 tool call 仍可能被当作推理阶段，强制短语会被插入 tool call 或其参数；dafeliton 与 chaunceyjiang 在讨论中确认，待 tool parser 统一迁移到 ParserEngine 后处理。

2. spec-decode 交互复杂：_thinking_budget_kernel 依赖 expanded_idx_mapping / expanded_local_pos 定位 draft token，_load_effective_token 中 cur_req_first_pos + pos - total_len + 1 的偏移计算较为微妙；rejection_sampler.py 仅一行改动，结合 PR#50939 的 -1 占位 token 场景，任何 expanded 索引语义变化都可能回归。
3. 冷扫描仍随历史长度增长：prefill 冷启动与 resume 需扫描 committed 历史，虽然 1024 块向量化带来约 34x 提速，但 benchmark 特意将 cold-resume-worst-case 排除在 --max-slowdown 判定之外，说明 worst-case 依然可能随 history 增长而退化。
4. 测试平台覆盖有限：test_gpu_thinking_budget.py 在非 CUDA 平台整体 skip，ROCm / CPU 下的 Triton kernel 行为缺少回归保障。
5. 配置契约变更：ReasoningConfig 新增字段并改变 reasoning_end_token_ids 语义（由自然结束改为强制短语），依赖旧语义的外部配置或序列化结果需要重新走 initialize_token_ids。 - 影响：用户侧：V2 用户可直接使用 thinking_token_budget 控制推理长度，无需退回 V1 或设置 VLLM_USE_V2_MODEL_RUNNER=0，对 Qwen3.5-27B-FP8、GLM-5.2 等长思考链量化模型收益明显；e2e 测试已默认覆盖 V2 路径。系统侧：采样阶段新增 GPU kernel，仅当 batch 中存在带 budget 请求时才执行，普通请求路径通过 use_thinking_budget 掩码快速返回；强制 logit 置 1e9 发生在 temperature 之前，语义稳定。团队侧：MRV2 sampler 的 state 类接线模式继续扩展（与 penalties / logit_bias / bad_words 对齐），后续 ParserEngine 迁移时需回访隐式结束问题。 - 风险标记：核心采样路径新增 GPU Kernel, spec-decode 交互耦合，隐式结束标记未覆盖，冷扫描仍需全量扫历史，新增配置契约字段

关联脉络

- PR #50939 [Model Runner V2] Fix -1 placeholder draft token ids in rejection sam…: 同属 MRV2 采样链正确性敏感区，本 PR 也修改 rejection_sampler.py 的 idx_mapping 传递，两处改动在 spec-decode 路径上相互依赖。
- PR #51210 [ModelRunner V2] Minor indexing optimizations: 同为 model_runner.py / input_batch.py 的索引映射改动，thinking budget kernel 深度依赖 expanded_idx_mapping 与 expanded_local_pos，后续索引语义变化需同步回归。
- PR #50931 [ModelRunner v2] Enable decoder token-wise pooling: 同属把 V1 能力补进 V2 采样器的功能线，接线模式一致（Sampler state 类生命周期 + model_runner 参数透传）。
- PR #51304 [V1] Copy NaN-in-logits counts to host asynchronously: 同为 V1 worker 采样路径的性能优化方向，反映 worker 采样链路正在持续追求减少同步与拷贝。