

PR #55353 完整报告

vllm-project/vllm

[Deprecation] Deprecate items scheduled for 0.29

合并时间: 2026-09-11 23:17

原文链接: <http://prhub.com.cn/vllm-project/vllm/pull/55353>

执行摘要

- 一句话: 清理 0.29 到期弃用项: 移除隐式同步属性与旧配置项
- 推荐动作: 建议精读, 但只精读三处, 其余可略过: ① vllm/v1/attention/backend.py 的契约收缩与 `unpadded()` 的替代传递路径, 理解“为什么 CPU 侧长度必须显式获取”; ② vllm/models/deepseek_v4/compressor.py 与 vllm/v1/attention/backends/mamba_attn.py 的 `async_spec_decode` 守卫写法——这是“乐观上界不可用于精确判断”的范式, 任何新增读取 `seq_lens_cpu_upper_bound` 的内核都应照此判断; ③ `tests/v1/e2e/spec_decode/draft_model/test_async.py` 用 `VLLM_GPU_SYNC_CHECK=error` 取代自建 fixture 的取舍。对使用方, 需要在发布说明中明确列出被删除的环境变量与配置别名, 并检查自有启动脚本中对 `envs.VLLM_MM_HASHER_ALGORITHM` 之类的直接引用。设计上值得讨论的遗留问题: `ReplaySSM` 的校验是否应前移到启动期参数校验, 以及是否该给所有 `seq_lens_cpu_upper_bound` 消费点加统一断言。

功能与动机

PR body 只有一句“Deprecate items scheduled for 0.29”, 动机来自代码中已经写死的到期标记: vllm/config/multimodal.py 的 `_get_mm_hasher_algorithm()` 通过 `get_from_deprecated_env_if_set("VLLM_MM_HASHER_ALGORITHM", "v0.27", "mm_hasher_algorithm")` 做环境变量回退; vllm/config/cache.py 用 "v0.29" 标记 `VLLM_PREFIX_CACHE_RETENTION_INTERVAL` 回退; vllm/config/attention.py 的 `use_fp4_indexer_cache` 告警明确写着“deprecated and will be removed in v0.29”; vllm/v1/attention/backend.py 的 `seq_lens_cpu / num_computed_tokens_cpu` 属性被 `@deprecated` 标注为“Prefer using device seq_lens directly to avoid implicit H<>D sync which breaks full async scheduling”。也就是说, 这些 API 一旦被误用就会引入隐式 GPU-CPU 同步, 直接破坏 full async scheduling 的收益, 因此到达承诺的删除窗口后需要彻底切断, 而不是继续保留会静默降级的兼容层。

实现拆解

1. 收缩 `CommonAttentionMetadata` 的 CPU 侧契约 (改动核心): 在 `vllm/v1/attention/backend.py` 删除 `_seq_lens_cpu`、`_num_computed_tokens_cpu` 两个惰性字段与 `seq_lens_cpu`、`num_computed_tokens_cpu` 两个 `@deprecated` property, 同时移除 `typing_extensions.deprecated` 导入; `unpadded()` 不再切片这两个私有字段, 改为

透传 `seq_lens_cpu_upper_bound`。保留 device 侧 `compute_num_computed_tokens()` (`seq_lens - query_lens`) 与 `token_to_req_indices()` 缓存。影响：所有需要 CPU 长度的调用点要么显式 `seq_lens.cpu()`，要么接受 `seq_lens_cpu_upper_bound` 的乐观语义，否则必须自行加守卫。

2. 生产者与消费者同步迁移：`vllm/v1/worker/gpu_model_runner.py` 不再构造 `_seq_lens_cpu / _num_computed_tokens_cpu`（并删掉 `async spec decode` 下把二者置 `None` 的特殊分支），只保留 `seq_lens_cpu_upper_bound = self.optimistic_seq_lens_cpu[:num_reqs_padded]`；`vllm/v1/worker/ubatch_utils.py::_make_metadata_with_slice` 去掉两个字段的切片与 `tokens_skipped` 修正逻辑。其余读取方——`vllm/v1/attention/backends/utils.py`、`vllm/v1/spec_decode/llm_base_proposer.py`、`vllm/v1/attention/backends/mha/indexer.py`、`vllm/v1/attention/backends/mha/rocm_aiter_mha_sparse.py`、`vllm/model_executor/models/whisper_causal.py`、`vllm/model_executor/layers/attention/cross_attention.py`、`vllm/distributed/kv_transfer/kv_connector/v1/offloading/scheduler.py`、`benchmarks/attention_benchmarks/mha_runner.py`——同步删除对旧字段 / 旧属性的引用。
3. 为“需要精确 CPU 长度”的两条路径加显式守卫：`vllm/models/deepseek_v4/compressor.py::_get_c128_boundary` 从读 `metadata._num_computed_tokens_cpu` 改为 `seq_lens_cpu_upper_bound - query_lens` 反推每个请求的 `start`；`CompressorMetadataBuilder.build` 新增 `async_spec_decode = async_scheduling and speculative_config is not None`，在该组合下直接把 `c128_boundary` 置 `None`，因为 `upper bound` 在 `async spec decode` 行上只是“假设 draft 全被接受”的乐观值。`vllm/v1/attention/backends/mamba_attn.py::_compute_common_metadata` 中 `ReplaySSM` 分支改为读取 `seq_lens_cpu_upper_bound` 并自算 `query_lens_cpu` 得到 `num_computed_d`，若 `decode_base_cpu / seq_lens_cpu` 缺失或处于 `async spec decode`，则抛 `ValueError`（文案改为“requires exact CPU sequence lengths and decode-base counts”），把“用错长度”的静默风险改成显式失败。
4. 配置与环境变量清理（对用户的破坏面）：`vllm/envs.py` 删除 `VLLM_MM_HASHER_ALGORITHM` 与 `VLLM_PREFIX_CACHE_RETENTION_INTERVAL` 的声明、`env_with_choices` 注册和读取 `lambda`；`vllm/config/multimodal.py` 删除 `_get_mm_hasher_algorithm()`，`mm_hasher_algorithm` 直接默认 `"blake3"`；`vllm/config/cache.py` 删除 `_get_prefix_cache_retention_interval()`，`prefix_cache_retention_interval` 改为 `Field(default=0, ge=0)`；`vllm/config/attention.py` 删除 `use_fp4_indexer_cache` 别名、`__post_init__` 中的告警与 `True -> "mxfp4"` 折叠逻辑，以及 `compute_hash()` 里的 `ignored_factors={"use_fp4_indexer_cache"}`（连带删除 `init_logger/logger`）；`vllm/platforms/rocm.py` 移除 `CUDA_VISIBLE_DEVICES` → `HIP_VISIBLE_DEVICES` 的自动拷贝。三处默认值 (`blake3 / 0 / indexer_kv_dtype="auto"`) 与旧行为一致，只有“通过环境变量或旧字段覆盖”的用法失效。
5. 测试配套：`tests/v1/e2e/spec_decode/draft_model/test_async.py` 删除约 60 行手写 `sync_tracker` fixture（`multiprocessing.Value` 计数 + 直接替换 `CommonAttentionMetadata.seq_lens_cpu` property + 打印调用栈），改为

`monkeypatch.setenv("VLLM_GPU_SYNC_CHECK", "error")`，把“禁止隐式同步”的断言下沉到 worker 内置的 GPU sync check，同时删掉 `torch._dynamo.reset()` 与 subprocess 输出依赖；`tests/engine/test_arg_utils.py` 删除已失效的 `test_prefix_cache_retention_interval_from_deprecated_env`。整体上源码与测试同步收敛，没有新增正向功能测试。

关键文件：

- `vllm/v1/attention/backend.py`（模块 注意力元数据；类别 source；类型 core-logic；符号 `CommonAttentionMetadata.seq_lens_cpu`, `CommonAttentionMetadata.num_computed_tokens_cpu`, `CommonAttentionMetadata.compute_num_computed_tokens`, `CommonAttentionMetadata.unpadded`）：本次契约收缩的中心：删除会触发隐式 D2H 同步的 `seq_lens_cpu / num_computed_tokens_cpu` 属性与私有字段，并让 `unpadded()` 改传 `seq_lens_cpu_upper_bound`，决定了所有下游 backend 如何获取 CPU 侧长度。
- `vllm/models/deepseek_v4/compressor.py`（模块 压缩器后端；类别 source；类型 data-contract；符号 `_get_c128_boundary`, `CompressorMetadataBuilder.build`）：
C128 边界判定原本依赖被删除的 `_num_computed_tokens_cpu`，本次改为从 `seq_lens_cpu_upper_bound` 反推，并新增 `async_spec_decode` 守卫，是“乐观上界不可用于精确判断”的范式实现。
- `vllm/v1/attention/backends/mamba_attn.py`（模块 线性注意力；类别 source；类型 core-logic；符号 `_compute_common_metadata`）：Mamba ReplaySSM 改为读取 `seq_lens_cpu_upper_bound` 并自算 query 长度，且在 `async spec decode` 或长度缺失时改为显式 `ValueError`，是本次唯一把“静默用错数据”改成“显式失败”的路径。
- `vllm/v1/worker/gpu_model_runner.py`（模块 模型执行器；类别 source；类型 data-contract）：attention metadata 的生产者：删除对 `_seq_lens_cpu / _num_computed_tokens_cpu` 的填充与 `async spec decode` 下的置 None 分支，只保留 `seq_lens_cpu_upper_bound`，决定了新契约下 CPU 侧信息的唯一来源。
- `vllm/config/cache.py`（模块 缓存配置；类别 source；类型 configuration；符号 `CacheConfig.prefix_cache_retention_interval`, `_get_prefix_cache_retention_interval`）：删除 `VLLM_PREFIX_CACHE_RETENTION_INTERVAL` 的默认值回退函数，把默认值内联为 `Field(default=0, ge=0)`，是本次面向用户的配置破坏面之一。
- `vllm/config/multimodal.py`（模块 多模态配置；类别 source；类型 configuration；符号 `MultiModalConfig.mm_hasher_algorithm`, `_get_mm_hasher_algorithm`）：删除 `VLLM_MM_HASHER_ALGORITHM` 的兼容读取函数，`mm_hasher_algorithm` 默认值直接写为 `"blake3"`，FIPS 场景需改用配置项而非环境变量。
- `vllm/config/attention.py`（模块 注意力配置；类别 source；类型 configuration；符号 `AttentionConfig.use_fp4_indexer_cache`, `AttentionConfig.post_init`, `AttentionConfig.compute_hash`）：移除 `use_fp4_indexer_cache` 弃用别名及其 `__post_init__` 折叠逻辑，并同步清理 `compute_hash()` 的 `ignored_factors`，改变旧字段到 `indexer_kv_dtype` 的隐式映射。
- `vllm/envs.py`（模块 环境变量；类别 source；类型 configuration；符号 `VLLM_MM_HASHER_ALGORITHM`, `VLLM_PREFIX_CACHE_RETENTION_INTERVAL`）：两个环境变量的声明与解析逻辑被整体删除，是判定“用户旧启动脚本会静默失效”的直接依

据。

- `vllm/platforms/rocm.py` (模块 平台适配; 类别 `source`; 类型 `configuration`) : 移除 `CUDA_VISIBLE_DEVICES` → `HIP_VISIBLE_DEVICES` 的自动拷贝, 改变 ROCm 用户的设备可见性语义, 是容易被忽略的一项破坏性变更。
- `tests/v1/e2e/spec_decode/draft_model/test_async.py` (模块 同步检测测试; 类别 `test`; 类型 `test-coverage`; 符号 `test_no_sync_with_spec_decode`, `sync_tracker`, `SyncTracker`, `tracking_seq_lens_cpu`) : 用 `VLLM_GPU_SYNC_CHECK=error` 取代约 60 行手写的同步探测 fixture, 是本次唯一有 review 争议的文件, 也决定了该用例未来的回归检出能力。
- `vllm/v1/worker/ubatch_utils.py` (模块 批切分工具; 类别 `source`; 类型 `core-logic`; 符号 `_make_metadata_with_slice`) : `ubatch` 切片路径原本会同步裁剪两个已删除字段并做 `tokens_skipped` 修正, 本次一并删除, 体现“隐式同步字段被移除后切片逻辑也要跟着收敛”。
- `tests/engine/test_arg_utils.py` (模块 参数测试; 类别 `test`; 类型 `test-coverage`; 符号 `test_prefix_cache_retention_interval_from_deprecated_env`) : 删除针对已移除环境变量的用例 `test_prefix_cache_retention_interval_from_deprecated_env`, 标志该兼容路径正式退场。

关键符号: `CommonAttentionMetadata.seq_lens_cpu`,
`CommonAttentionMetadata.num_computed_tokens_cpu`,
`CommonAttentionMetadata.compute_num_computed_tokens`,
`CommonAttentionMetadata.unpadded`, `_get_c128_boundary`,
`CompressorMetadataBuilder.build`, `_compute_common_metadata`,
`_make_metadata_with_slice`, `_get_mm_hasher_algorithm`,
`_get_prefix_cache_retention_interval`, `AttentionConfig.post_init`,
`AttentionConfig.compute_hash`, `test_no_sync_with_spec_decode`,
`test_prefix_cache_retention_interval_from_deprecated_env`

关键源码片段

`vllm/v1/attention/backend.py`

本次契约收缩的中心: 删除会触发隐式 D2H 同步的 `seq_lens_cpu / num_computed_tokens_cpu` 属性与私有字段, 并让 `unpadded()` 改传 `seq_lens_cpu_upper_bound`, 决定了所有下游 backend 如何获取 CPU 侧长度。

```
# vllm/v1/attention/backend.py —— 迁移后的最终形态 (节选)
@dataclass
class CommonAttentionMetadata:
    # 旧的 _seq_lens_cpu / _num_computed_tokens_cpu 惰性字段已被删除,
    # 连带删除两个 @deprecated property: 它们首次访问时会执行
    # seq_lens.to("cpu"), 即一次隐式 D2H 同步, 直接破坏 full async scheduling.
    # 现在想拿 CPU 侧序列长度只能显式调用 seq_lens.cpu(),
    # 或者使用下面这个语义为“上界”的 CPU 字段。
    seq_lens_cpu_upper_bound: torch.Tensor | None = None
    """(batch_size,) CPU upper bound on seq_lens. Precise for prefill rows
    and for all rows outside async spec decode; optimistic for async-spec
```

decode rows (assumes every draft was accepted). Not safe for kernels that need exact per-row context lengths on decode rows."""

```
_num_computed_tokens_cache: torch.Tensor | None = None
_token_to_req_indices_cache: torch.Tensor | None = None
```

保留下来的 device 侧实现：全程不碰 CPU，供注意力内核直接使用。

```
def compute_num_computed_tokens(self) -> torch.Tensor:
    """Compute num_computed_tokens on device (seq_lens - query_lens)."""
    if self._num_computed_tokens_cache is None:
        query_lens = self.query_start_loc[1:] - self.query_start_loc[:-1]
        self._num_computed_tokens_cache = self.seq_lens - query_lens
    return self._num_computed_tokens_cache
```

TODO(lucas): remove once we have FULL-CG spec-decode support

```
def unpadded(
    self, num_actual_tokens: int, num_actual_reqs: int
) -> "CommonAttentionMetadata":
    # unpadded() 不再搬运已删除的私有字段，而是把 upper bound 一并切片带走，
    # 这样下游 backend 仍能拿到 CPU 侧近似长度，且不会触发任何同步。
    maybe_slice_reqs = lambda x: x[:num_actual_reqs] if x is not None else None
    return CommonAttentionMetadata(
        query_start_loc=self.query_start_loc[: num_actual_reqs + 1],
        query_start_loc_cpu=self.query_start_loc_cpu[: num_actual_reqs + 1],
        seq_lens=self.seq_lens[:num_actual_reqs],
        num_reqs=num_actual_reqs,
        num_actual_tokens=num_actual_tokens,
        max_query_len=self.max_query_len,
        max_seq_len=self.max_seq_len,
        block_table_tensor=self.block_table_tensor[:num_actual_reqs],
        slot_mapping=self.slot_mapping[:num_actual_tokens],
        seq_lens_cpu_upper_bound=maybe_slice_reqs(self.seq_lens_cpu_upper_bound),
        is_prefilling=maybe_slice_reqs(self.is_prefilling),
        rswa_prefix_lens=maybe_slice_reqs(self.rswa_prefix_lens),
        replayssm_decode_base_cpu=maybe_slice_reqs(self.replayssm_decode_base_cpu),
    )
```

vllm/models/deepseek_v4/compressor.py

C128 边界判定原本依赖被删除的 `_num_computed_tokens_cpu`，本次改为从 `seq_lens_cpu_upper_bound` 反推，并新增 `async_spec_decode` 守卫，是“乐观上界不可用于精确判断”的范式实现。

vllm/models/deepseek_v4/compressor.py —— C128 边界判定与 `async spec decode` 守卫

```
def _get_c128_boundary(metadata: CommonAttentionMetadata) -> bool | None:
    # 过去直接读 metadata._num_computed_tokens_cpu（已被删除的隐式同步字段）；
    # 现在从 CPU 上界减去本步 query 长度来反推每个请求的 start：
    # start = seq_lens - query_lens，与 upper bound 的推导口径保持一致。
    seq_lens_cpu = metadata.seq_lens_cpu_upper_bound
    if seq_lens_cpu is None:
```

```

return None

query_lens = metadata.query_start_loc_cpu[1:] - metadata.query_start_loc_cpu[:-1]
starts = seq_lens_cpu - query_lens
starts_list = starts.tolist()
query_start_loc = metadata.query_start_loc_cpu.tolist()
return any(
    start % 128 + query_start_loc[i + 1] - query_start_loc[i] >= 128
    for i, start in enumerate(starts_list)
)

```

```

class CompressorMetadataBuilder(AttentionMetadataBuilder):
    def build(
        self,
        common_prefix_len: int,
        common_attn_metadata: CommonAttentionMetadata,
        fast_build: bool = False,
    ) -> CompressorMetadata:
        token_to_req_indices = common_attn_metadata.token_to_req_indices(
            self.token_to_req_indices
        )
        num_decode_tokens = None
        if _prefer_two_stage_compressor():
            _, _, num_decode_tokens, _ = split_decodes_and_prefills(
                common_attn_metadata, decode_threshold=1
            )
            # async spec decode 下 seq_lens_cpu_upper_bound 是“假设所有 draft 都被接受”
            # 的乐观值，用它判断是否跨 128 边界会算错，因此这种情况下直接放弃该优化。
            async_spec_decode = (
                self.vllm_config.scheduler_config.async_scheduling
                and self.vllm_config.speculative_config is not None
            )
        return CompressorMetadata(
            block_table=common_attn_metadata.block_table_tensor.clamp_(min=0),
            slot_mapping=common_attn_metadata.slot_mapping,
            block_size=self.block_size,
            token_to_req_indices=token_to_req_indices,
            num_decode_tokens=num_decode_tokens,
            c128_boundary=(
                _get_c128_boundary(common_attn_metadata)
                if self.block_size == 8 and not async_spec_decode
                else None
            ),
        )

```

[vllm/v1/attention/backends/mamba_attn.py](#)

Mamba ReplaySSM 改为读取 `seq_lens_cpu_upper_bound` 并自算 `query` 长度，且在 `async spec decode` 或长度缺失时改为显式 `ValueError`，是本次唯一把“静默用错数据”改成“显式失败”的路径。

```
# vllm/v1/attention/backends/mamba_attn.py —— ReplaySSM 的精确长度要求
if self.use_replayssm and not self.use_flashinfer_replayssm and num_decodes > 0:
    decode_base_cpu = common_attn_metadata.replayssm_decode_base_cpu
    # 旧实现读 common_attn_metadata._num_computed_tokens_cpu (已删除) ;
    # 现在改为“CPU 上界 - 本步 query 长度”，并显式拒绝无法保证精确性的组合。
    seq_lens_cpu = common_attn_metadata.seq_lens_cpu_upper_bound
    async_spec_decode = (
        self.vllm_config.scheduler_config.async_scheduling
        and self.vllm_config.speculative_config is not None
    )
    if decode_base_cpu is None or seq_lens_cpu is None or async_spec_decode:
        # 从“静默算错 write position”改成显式报错：
        # async spec decode 下 upper bound 是乐观值，不能用来推导 decode 写入位置。
        raise ValueError(
            "--use-replayssm requires exact CPU sequence lengths and "
            "decode-base counts to derive decode write positions"
        )
    query_lens_cpu = (
        common_attn_metadata.query_start_loc_cpu[1 : num_decodes + 1]
        - common_attn_metadata.query_start_loc_cpu[:num_decodes]
    )
    num_computed_d = seq_lens_cpu[:num_decodes] - query_lens_cpu
    decode_base_d = decode_base_cpu[:num_decodes]
    align_mode = self.vllm_config.cache_config.mamba_cache_mode == "align"
    block_size = self.kv_cache_spec.block_size
    # 注意：query_lens_cpu 现在在更早的位置计算，后续
    # decode_steps_cpu / valid_decode_rows 等都复用同一个 CPU 张量。
```

评论区精华

唯一实质性 review 交锋集中在测试改造上 ([tests/v1/e2e/spec_decode/draft_model/test_async.py:46](#)) :

coderabbitai[bot] (标记 Major / Functional Correctness) : “Update the test setup around the GPU sync check to patch `gpu_sync_debug._SYNC_CHECK_MODE` directly instead of only setting `VLLM_GPU_SYNC_CHECK` via `monkeypatch.setenv`. Apply the patch before exercising the single-GPU `UniProcExecutor`, preserving the "error" mode...”

mgoin 回复: “@yewentao256 I believe in CI we actually enable sync error by default, so this might not be needed”

讨论的实质是：新测试把断言责任交给 worker 侧读取的环境变量，如果 `vllm` 已在 `fixture` 阶段被导入、或 `gpu_sync_debug` 在导入时就把模式固化成 `_SYNC_CHECK_MODE`，那么 `monkeypatch.setenv` 可能不会生效，测试会退化成“只验证能跑通”。mgoin 认为 CI 已经默认

开启 sync error，因此这行 setenv 可能多余——但这同时意味着该测试在本地开发环境下的同步断言强度更弱。最终 PR 未做修改即合并（sfeng33 直接 APPROVED，mgoin 的 COMMENTED 也落在同一行），属于“维护者接受现状、未形成明确结论”的状态。

- 新测试用 monkeypatch.setenv 设置 VLLM_GPU_SYNC_CHECK 是否可靠 (testing): PR 未采纳修改即合并（sfeng33 APPROVED，mgoin 的 COMMENTED 落在同一行），维护者接受现状；但“setenv 生效时机”与“是否依赖 CI 默认值”没有形成书面结论。
- 移除隐式同步属性后，测试断言责任向 worker 内置检查转移 (design): 测试与实现同步收敛，维护者接受以环境变量方式兜底；代价是本地（非 CI）执行时同步断言强度不确定。

风险与影响

- 风险：

1. 破坏性升级路径缺显式报错：vllm/envs.py 删除 VLLM_MM_HASHER_ALGORITHM / VLLM_PREFIX_CACHE_RETENTION_INTERVAL 后，用户继续导出这两个变量不会有任何提示，行为静默退回默认值 (blake3 / 0)；如果外部代码仍引用 envs.VLLM_MM_HASHER_ALGORITHM，则会直接 AttributeError。同理 use_fp4_indexer_cache 字段被删除后，旧 JSON/CLI 传参是报错还是被 @config 忽略，材料中未体现，需要 release note 明确。
2. seq_lens_cpu_upper_bound 的语义陷阱：该字段在 async spec decode 行上是乐观上界（“assumes every draft was accepted”），只适合不需要逐行精确上下文长度的场景。本 PR 只在 DeepSeek-V4 compressor 与 Mamba ReplaySSM 两处加了 async_spec_decode 守卫，vllm/v1/worker/ubatch_utils.py 与各 attention backend 中其余仍在读取该字段的路径没有在材料里显示同等判断，存在后续误用该上界做精确判断的隐患。
3. Mamba ReplaySSM 的失败模式前移不到位：_compute_common_metadata 在每步构建 metadata 时抛 ValueError，意味着 --use-replayssm + async scheduling + spec decode 的组合会在运行期（而非启动校验期）才失败；虽然从“静默算错 write position”变成“显式报错”是改善，但错误出现时机对生产更不友好。
4. ROCm 设备可见性行为变更：vllm/platforms/rocm.py 不再把 CUDA_VISIBLE_DEVICES 自动拷贝到 HIP_VISIBLE_DEVICES，只设置前者的 ROCm 用户会看到可选设备集合变化，可能表现为静默失效或直接报错（取决于剩余校验分支），也影响容器编排脚本。
5. 测试断言强度与可诊断性下降：如讨论所述，VLLM_GPU_SYNC_CHECK 的生效时机未经证实；同时删除 sync_tracker 后失去附带的调用栈打印，一旦回归，定位隐式同步来源的成本变高。
6. 正向影响：删除惰性 .to("cpu") 属性后，误触发 D2H 同步的入口被彻底封死，对 full async scheduling 下的 spec decode 吞吐是结构性收益，“性能”标签正来自此。- 影响：
：对用户：这是一次需要迁移的破坏性变更——使用 VLLM_MM_HASHER_ALGORITHM (FIPS 场景常用 sha256/sha512)、VLLM_PREFIX_CACHE_RETENTION_INTERVAL、--use-fp4-indexer-cache 或 ROCm 上只设 CUDA_VISIBLE_DEVICES 的部署必须调整启动参数，否则会静默回退默认值；但默认路径 (blake3、interval=0、indexer_kv_dtype=auto) 行为不变，因此多数用户升级无感。对系统：

CommonAttentionMetadata 的数据契约被收缩，async spec decode + 混合 KV 缓存组、DeepSeek-V4 C128A compressor、Mamba ReplaySSM 这些依赖 CPU 侧长度的路径改由 upper bound + 显式守卫驱动，正确性责任从“隐式同步”转移到调用点判断。对团队：改动横跨 v1 worker、attention backend、config、platforms 与 ROCm 路径（28 个文件），但大多是机械替换，真正的设计决策集中在两处守卫；同时确立了“弃用到期即删除、不再保留静默兼容层”的维护节奏。- 风险标记：破坏性变更：删除环境变量与配置别名，隐式同步属性移除，元数据契约收缩，乐观上界可能被误用于精确判断，Mamba ReplaySSM 运行期报错而非启动校验，ROCM 设备可见性行为变更，测试同步断言可靠性存疑

关联脉络

- PR #56138 [Bugfix] Pin EPLB and MLA host-to-device transfer buffers: 同一主题脉络：都在消除 host/device 间的隐式拷贝与同步。本 PR 直接删除会触发 D2H 同步的 seq_lens_cpu 属性，是该方向的延续性收口。
- PR #55450 [Bugfix][Core] Retire Mamba states across null gaps: 同属 Mamba/ 混合缓存的状态管理线。本 PR 修改 vllm/v1/attention/backends/mamba_attn.py 的 ReplaySSM 写入位置推导并增加显式报错，与该 PR 对 mamba 状态语义的修正互为上下文。
- PR #56214 [Model] Support DeepSeek-V4.1-Flash: 同属 DeepSeek-V4 路径：该 PR 引入了 vllm/models/deepseek_v4_1、kv_cache_utils 与 MLA indexer 相关改动，本 PR 随后收缩了同族 vllm/models/deepseek_v4/compressor.py 所依赖的 CPU 侧元数据契约，两者需要一起理解才能判断 C128 边界逻辑是否仍正确。
- PR #56433 [ROCM][Bugfix] Fix AITER preshuffled FP8 block-scale kernel: 本 PR 同时改了 vllm/v1/attention/backends/mla/rocm_aiter_mla_sparse.py 与 vllm/platforms/rocm.py，与该 PR 的 ROCm MLA 路径相邻，合并顺序会影响 ROCm 侧注意力后端的元数据读取。
- PR #53675 [Multimodal] Use GPU NVDEC for EPD encoder-only instance video media IO: 该 PR 修改了 vllm/multimodal/hasher.py 与多模态配置默认值，与本 PR 删除 mm_hasher_algorithm 的环境变量回退（default_factory → "blake3"）落在同一配置面，升级时需要一并核对多模态哈希相关行为。