PR #33335 完整报告

sgl-project/sglang

spec: build every draft worker from a draft ServerArgs copy

合并时间：2026-08-03 12:22

原文链接：http://prhub.com.cn/sgl-project/sglang/pull/33335

执行摘要

- 一句话：draft worker 全部改用独立 ServerArgs 副本，杜绝配置污染
- 推荐动作：建议精读。该 PR 是投机解码配置隔离的架构性修复，展示了三种可复用的设计手法：从已解析配置 + overrides log 重放来构造子工作负载的配置副本；用 preserve_config 窗口控制配置发布的起止边界；基于数据流时序论证删除无效写入。理解 draft_server_args_copy 与 maybe_init_draft_worker 的配合，对后续 config 迁移工作有直接参考价值。合并前应确认 CI 的 speculative 套件通过。

功能与动机

PR body 明确指出：EAGLEWorkerV2、StandaloneWorkerV2、MultiLayerEagleWorkerV2 和 FrozenKVMTPWorkerV2 会把 draft 的 context_length 写到与 target worker 共享的 ServerArgs 实例上，调度器又在构建前把 draft 的 load_format 写到同一对象；由于 worker 与 EagleDraftWorker 都不做复制，target 的配置在进程剩余生命周期内携带 draft 值。而 dflash 和 dspark 从未有此问题，因为 build_draft_tp_worker 先 deepcopy。因此本 PR 的目标是让四条 v2 worker 路径获得与 dflash/dspark 相同的配置隔离，同时保住 --speculative-draft-load-format 的传递。

实现拆解

1. 新增副本生成入口：在 python/sglang/srt/speculative/draft_worker_common.py 中新增 _draft_load_format_fields() 与 draft_server_args_copy()。副本以进程已解析配置（而不是 pristine seed）为起点：先遍历 get_context().overrides_log() 重放所有已审核的 override 字段，再 deepcopy(server_args)，最后通过 override("draft_worker.copy", ...) 写入 context_length（跟随 target_model_config.context_len）与 load_format（跟随 get_spec().speculative_draft_load_format）。这样 load-time 覆盖（chunked-prefix gate、SM100 GDN prefill 默认值）能进入 draft 层，而 target 实例本身不被触碰。
2. 改造调度器交接点：python/sglang/srt/managers/scheduler.py 的 maybe_init_draft_worker() 在调用 create_worker 之前只创建一次副本，并同时传给 worker 工厂与 DraftWorkerClass 构造；围绕构造用 get_context().preserve_config() + set_server_args(draft_server_args) 把副本发布为当前配置，构造结束后自动恢复 target 配置。原来把 load_format 写到共享 self.server_args 的逻辑被删除。
3. 清理四个 v2 worker：EAGLEWorkerV2、StandaloneWorkerV2、MultiLayerEagleWorkerV2、FrozenKVMTPWorkerV2 各自 __init__ 中的

spec_worker.match_target_context_length override 全部删除，职责上移到 draft_server_args_copy。

4. 删除无效的 hot-token-map 写入：EAGLEWorkerV2.init_token_map 中设置 json_model_override_args={"hot_vocab_size": N} 的代码被删除。原因是该写入从 alloc_memory_pool 触发，远晚于 EagleDraftWorker.__init__ 构建 ModelConfig；json_model_override_args 只在 ModelConfig 构造时到达 hf_config，因此写入对 draft 模型无效，只污染共享实例。hot_token_id 本身不变，draft checkpoint 自带的 hot_vocab_size 行为照旧。
5. 测试、ratchet 与文档配套：新增 test/registered/unit/spec/test_draft_server_args_copy.py（副本语义：context_length 跟随 target、target 实例不受影响、load_format 仅在配置时注入、draft 特定字段优先于已解析字段）和 test_spec_worker_draft_isolation.py（调度器交接契约：工厂与 worker 都收到副本、构建期间副本是发布配置、结束后 target 配置回槽）。test_server_args_writer_ratchet.py 基线从 31 降到 26；.claude/skills/sglang-runtime-context/SKILL.md 更新为“每个 draft 都在自己的配置发布下构建”的说明。

关键文件：

- python/sglang/srt/speculative/draft_worker_common.py（模块 草稿配置；类别 source；类型 core-logic；符号 draft_server_args_copy, _draft_load_format_fields, draft_server_args_overrides）：核心实现文件：新增 draft_server_args_copy 与 _draft_load_format_fields，是四个 v2 worker 与调度器交接的统一配置副本入口；同时把 load_format 注入合并进 draft_server_args_overrides，保住 dflash/dspark 的 --speculative-draft-load-format 传递。
- python/sglang/srt/managers/scheduler.py（模块 调度器；类别 source；类型 dependency-wiring；符号 Scheduler.maybe_init_draft_worker）：调度器交接点：maybe_init_draft_worker 改为先制作副本再传给工厂与 worker，并用 preserve_config 发布窗口包裹构造；删除了原来直接写共享实例的 load_format 覆盖。
- test/registered/unit/spec/test_spec_worker_draft_isolation.py（模块 隔离测试；类别 test；类型 test-coverage；符号 _StopConstruction, _scheduler, TestSchedulerDraftServerArgs, _captured_draft_args）：调度器级契约测试：验证副本在 create_worker 前只创建一次、工厂与 worker 收到同一副本、构建期间副本是发布配置而 target 配置随后回槽。
- test/registered/unit/spec/test_draft_server_args_copy.py（模块 副本测试；类别 test；类型 test-coverage；符号 TestDraftServerArgsCopy, test_the_draft_context_length_follows_the_target, test_the_target_instance_is_left_alone, test_the_draft_load_format_applies_only_when_configured）：副本语义单测：context_length 跟随 target、target 实例不受影响、load_format 仅在配置时注入、draft 特定字段优先于已解析字段、build overrides 携带 load_format。
- python/sglang/srt/speculative/eagle_worker_v2.py（模块 EAGLE 草稿；类别 source；类型 core-logic；符号 EAGLEWorkerV2.init, EAGLEWorkerV2.init_token_map）：删除 EAGLEWorkerV2.__init__中的 context_length 覆盖与 init_token_map 中无效的

hot_vocab_size 写入，是行为变化最大的 worker 文件。

- python/sglang/srt/speculative/standalone_worker_v2.py（模块 独立草稿；类别 source；类型 core-logic；符号 StandaloneWorkerV2.init）：删除 StandaloneWorkerV2.__init__中共享实例上的 context_length 覆盖。
- python/sglang/srt/speculative/multi_layer_eagle_worker_v2.py（模块 多层草稿；类别 source；类型 core-logic；符号 MultiLayerEagleWorkerV2.init）：删除 MultiLayerEagleWorkerV2.__init__中共享实例上的 context_length 覆盖。
- python/sglang/srt/speculative/frozen_kv_mtp_worker_v2.py（模块 冻结草稿；类别 source；类型 core-logic；符号 FrozenKVMTPWorkerV2.init）：删除 FrozenKVMTPWorkerV2.__init__中共享实例上的 context_length 覆盖。
- test/registered/unit/test_server_args_writer_ratchet.py（模块 写入基线；类别 test；类型 test-coverage；符号 TestServerArgsWriterRatchet, _BASELINE）：ServerArgs 写者基线从 31 降到 26，量化体现了共享实例写入的清除。
- .claude/skills/sglang-runtime-context/SKILL.md（模块 技能文档；类别 docs；类型 documentation）：同步更新运行期上下文技能文档：每个 draft 都在自己的配置发布下构建，发布在构建结束时结束。

关键符号：draft_server_args_copy, _draft_load_format_fields, draft_server_args_overrides, Scheduler.maybe_init_draft_worker, EAGLEWorkerV2.init, EAGLEWorkerV2.init_token_map, StandaloneWorkerV2.init, MultiLayerEagleWorkerV2.init, FrozenKVMTPWorkerV2.init

关键源码片段

python/sglang/srt/speculative/draft_worker_common.py

核心实现文件：新增 draft_server_args_copy 与 _draft_load_format_fields，是四个 v2 worker 与调度器交接的统一配置副本入口；同时把 load_format 注入合并进 draft_server_args_overrides，保住 dflash/dspark 的 --speculative-draft-load-format 传递。

```
def _draft_load_format_fields() -> dict:
    # 读取 bag 中 `--speculative-draft-load-format` 的解析结果；未配置时返回
    # 空字典，避免把 None 写进副本导致后续 `build_load_config` 误读。
    draft_load_format = get_spec().speculative_draft_load_format
    if draft_load_format is None:
        return {}
    return dict(load_format=draft_load_format)


def draft_server_args_overrides(target_model_config, draft_backend) -> dict:
    """dflash / dspark 路径的预发布字段调整（build_draft_tp_worker 使用）。"""
    return dict(
        skip_tokenizer_init=True,
        speculative_draft_attention_backend=draft_backend,
        prefill_attention_backend=None,
        decode_attention_backend=None,
        attention_backend=draft_backend,
```

```
        context_length=target_model_config.context_len,
        disable_chunked_prefix_cache=get_schedule().disable_chunked_prefix_cache,
        # 调度器不再向共享实例写 load_format，这里必须显式带上，
        # 否则 `--speculative-draft-load-format` 在 dflash/dspark 上静默失效。
        **_draft_load_format_fields(),
    )


def draft_server_args_copy(server_args: ServerArgs, target_model_config) -> ServerArgs:
    """为自建 draft（EAGLE / standalone / multi-layer / frozen-KV MTP）生成专属副本。

    起点是进程当前已解析的配置，而不是 pristine seed：副本会在 draft 构建期间
    被发布，因此此前发生的 load-time 覆盖（chunked-prefix gate、SM100 GDN
    prefill 默认值等）必须进入 draft 层。在此基础上 `context_length` 跟随
    target（draft 读取 target 的 KV），`load_format` 跟随 draft 专用参数；
    target 自身的实例保持原样，不再被 draft 值污染。
    """
    draft_load_format = get_spec().speculative_draft_load_format
    if draft_load_format is not None:
        logger.info(f"Using draft model load_format: '{draft_load_format}'")

    # 重放 overrides_log：只有这些经过审核的 mutation point 才能在进程解析后
    # 合法写入配置，把它们合并进副本等价于在"已解析配置"时刻拍快照。
    resolved = {}
    for _source, fields in get_context().overrides_log():
        resolved.update(fields)

    draft_server_args = deepcopy(server_args)
    # 副本写入口：post-resolution 的 ServerArgs 拒绝裸赋值，统一走 override。
    draft_server_args.override(
        "draft_worker.copy",
        **{
            **resolved,
            "context_length": target_model_config.context_len,
            **_draft_load_format_fields(),
        },
    )
    return draft_server_args
```

python/sglang/srt/managers/scheduler.py

调度器交接点：maybe_init_draft_worker 改为先制作副本再传给工厂与 worker，并用 preserve_config 发布窗口包裹构造；删除了原来直接写共享实例的 load_format 覆盖。

```
def maybe_init_draft_worker(self):
    if self.spec_algorithm.is_none():
        self.draft_worker = None
        self.external_corpus_manager = None
        return
```

```
from sglang.srt.speculative.draft_worker_common import draft_server_args_copy

# 在进入 worker 工厂之前只创建一次副本：注册的插件算法（通过
# SpeculativeAlgorithm.register）可能依据交给它的 config 选择 worker 类，
# 因此工厂与 worker 必须看到同一个 draft 配置对象。
draft_server_args = draft_server_args_copy(
    server_args=self.server_args,
    target_model_config=self.tp_worker.model_runner.model_config,
)
draft_worker_kwargs = dict(
    server_args=draft_server_args,
    gpu_id=self.ps.gpu_id,
    ps=self.ps,
    nccl_port=self.nccl_port,
    target_worker=self.tp_worker,
)

DraftWorkerClass = self.spec_algorithm.create_worker(draft_server_args)
# 模型级权重加载读取的是运行时 bags 而非传入的实例：draft 构建期间
# 必须把副本发布为当前配置；preserve_config 会在构造结束后自动恢复
# target 的配置，后续 alloc_memory_pool / cuda-graph capture 都回到
# target 的 bags，不再读到被污染的共享实例。
with get_context().preserve_config():
    get_context().set_server_args(draft_server_args)
    self.draft_worker = DraftWorkerClass(**draft_worker_kwargs)

#  （ngram 外部语料库分支从略）
if self.spec_algorithm.is_ngram():
    ...
```

评论区精华

本 PR 自身无新增 review 评论（唯一的评论是机器人通知，声明 Gemini Code Assist 已停止服务）；三轮评审讨论和逐条意见分流都发生在被取代的前身 #33239 上，PR body 说明「the review discussion and the triage of each round of comments is on #33239; the code here is identical to that PR's final revision」。核心讨论要点集中在三处：其一，副本必须从已解析配置而非 pristine seed 出发，因为 load-time 覆盖（chunked-prefix gate、SM100 GDN prefill 默认值）必须传达给 draft 层；其二，副本必须在 create_worker 之前只创建一次，注册插件算法可能依据传入 config 选择 worker 类，工厂与 worker 必须看到同一对象；其三，EAGLE hot-token-map 写入为何删除而非移动——它从 alloc_memory_pool 触发，远晚于 ModelConfig 构建，json_model_override_args 只在该构造点到达 hf_config，因此写入对 draft 模型无效，只污染共享实例。

- 评审讨论全部沉淀在前身 PR #33239 (other): 代码以 #33239 最终版为准直接落地；作者明确请求合并前关注 CI 的 speculative 套件。

风险与影响

- 风险：
  - 端到端验证缺口：作者明确说明开发机上没有投机解码 checkpoint，未做 EAGLE/MTP 端到端运行，CI 的 speculative 套件是合并前唯一 gate。这直接覆盖 maybe_init_draft_worker 的发布窗口与四个 worker 的构建路径，是本 PR 最大的风险点。
  - 配置发布窗口语义变更：draft 构建期间副本被发布，构建结束后的所有读取（alloc_memory_pool、init_attention_backends、cuda-graph capture）回到 target 的 bags。此前能读到"被污染的共享实例"的代码，现在读到的将是 target 配置——这是预期语义，但仍属于行为变化。
  - 依赖 overrides_log 完整性：draft_server_args_copy 的重放机制假设所有进程级写入都经过 overrides_log。writer ratchet（31 → 26）正是这一假设的护栏，但未来新增非审核写入入口会静默丢失字段。
  - hot-token-map 行为变化点：删除 json_model_override_args 写入属于行为变更。作者论证其无效，但若存在依赖该隐式 hot_vocab_size 的 checkpoint，需要端到端验证确认无损。
  - 影响：影响面覆盖所有 speculative decoding 路径：EAGLE、standalone、multi-layer EAGLE、frozen-KV MTP 以及间接涉及的 dflash/dspark。修复了 target ServerArgs 被 draft 值污染、进程后续构造继承 draft 配置的缺陷，使 draft 与 target 配置彻底隔离。对用户而言主要是正确性与可维护性提升，无 API 变化；对团队而言这是 ServerArgs 写者收缩（31 处 → 26 处）与 runtime_context bags 迁移工作的重要一环，为后续 config 所有权重构铺路。
  - 风险标记：核心路径变更，缺少端到端验证，配置发布窗口语义变更，依赖 overrides_log 完整性

关联脉络

- PR #33239 前身 PR（关闭未合并，评审讨论所在）：PR body 明确说明本 PR 取代 #33239，代码与其最终修订版一致；GitHub 将链式基线当作栈处理导致其无法重定向与合并。
- PR #33338 config: retire the last process-global config field reads: 同一 config 迁移线：清除进程级配置读取、改走命名空间访问器；本 PR 的 draft_server_args_copy 依赖 get_context 的 overrides_log 与发布窗口，且两者都改动了 .claude/skills/sglang-runtime-context/SKILL.md。
- PR #33336 config: keep runtime hicache and weight-version updates off ServerArgs: 同类配置所有权收归 ServerArgs 之外的改造，共同推进 writer ratchet 基线下调；本 PR 将 ratchet 从 31 收敛到 26。
- PR #33298 [Spec] Support sampling in the DSPARK graph-folded draft proposal: 同为 speculative decoding 功能线，且 DSPARK/dflash 的 build_draft_tp_worker 路径在本 PR 中需要同步获得 load_format 注入，属于直接交互的代码路径。