PR #24793 完整报告

sgl-project/sglang

[DSV4] Cherry pick missing commits from deepseek_v4 branch and enhance tests

合并时间：2026-05-09 19:15

原文链接：http://prhub.com.cn/sgl-project/sglang/pull/24793

执行摘要

- 一句话：Cherry-pick DSV4 缺失提交并增强 per-commit 测试
- 推荐动作：值得精读此 PR，尤其是 _drop_file_cache_after_load 的设计模式和测试架构重构思路。对于 DSV4 部署和 RL 训练场景有直接帮助。

功能与动机

Cherry-pick 来自 deepseek_v4 分支的两个缺失 commit，包括自闭合 invoke 标签修复（#23915）和权重缓存释放功能。同时增强 per-commit v4 测试覆盖率，新增 B200 FP4、B200 FP4 MegaMoE 和 H200 FP8 测试，并添加斜杠命令等基础设施。

实现拆解

1. 自闭合 invoke 标签解析支持：修改 deepseekv32_detector.py 中的 invoke_regex 以匹配 < | DSML | invoke name="x"/> 语法，新增 _unpack_invoke_match 辅助方法，在 detect_and_parse 和 parse_streaming_increment 中使用。更新 test_function_call_parser.py 添加相应单元测试。
2. 权重缓存释放功能：在 weight_utils.py 中新增 _drop_file_cache_after_load 函数，利用 posix_fadvise 释放 page cache。在 safetensors_weights_iterator、multi_thread_safetensors_weights_iterator 和 buffered_multi_thread_safetensors_weights_iterator 中添加 drop_cache_after_load 参数，并在 yield 后调用。同时优化 multi_thread_safetensors_weights_iterator 的返回值以支持删除 state_dict 引用。修改 loader.py 和 server_args.py 添加 --weight-loader-drop-cache-after-load 参数。
3. 测试文件迁移与重构：将 test/registered/4-gpu-models/test_deepseek_v4_flash_fp4_b200.py 移动并重命名为 test/registered/dsv4/test_deepseek_v4_flash_fp4_b200.py，从 nightly 改为 per-commit（注册 stage-c-test-dsv4-4-gpu-b200），新增 LowLatency recipe 类，将 MaxThroughput 改为 MegaMoE 类。同样将 B200 FP4 测试文件改为 H200 FP8 测试文件，并修正配置和模型名。
4. CI 依赖脚本统一：新增 scripts/ci/cuda/ci_install_dsv4_dep.sh，整合 FlashMLA、DeepEP 等 dsv4 依赖的安装流程，删除旧的 ci_install_flash_mla.sh。新脚本支持架构检测和强制重建逻辑。

5. 协议支持 reasoning_effort=max：在 protocol.py 和 test_protocol.py 中添加对 reasoning_effort=max 的接受和测试。

关键文件：

- test/registered/dsv4/test_deepseek_v4_flash_fp4_b200.py（模块 测试；类别 test；类型 rename-or-move；符号 TestDSV4FlashFP4B200, setUpClass, tearDownClass, test_gsm8k）：核心测试文件迁移与重构，新增 LowLatency 和 MegaMoE 测试，从 nightly 改为 per-commit CI。
- python/sglang/srt/model_loader/weight_utils.py（模块 运行时；类别 source；类型 data-contract；符号 _drop_file_cache_after_load）：新增 _drop_file_cache_after_load 函数，支持权重加载后释放文件缓存，优化 RL 训练内存。
- python/sglang/srt/function_call/deepseekv32_detector.py（模块 函数调用；类别 source；类型 core-logic；符号 _unpack_invoke_match）：修改 invoke_regex 以支持自闭合标签，新增 _unpack_invoke_match 辅助方法。

关键符号：_drop_file_cache_after_load, _unpack_invoke_match, detect_and_parse, parse_streaming_increment, safetensors_weights_iterator, multi_thread_safetensors_weights_iterator, buffered_multi_thread_safetensors_weights_iterator

关键源码片段

test/registered/dsv4/test_deepseek_v4_flash_fp4_b200.py

核心测试文件迁移与重构，新增 LowLatency 和 MegaMoE 测试，从 nightly 改为 per-commit CI。

```
"""B200 per-commit CI: DeepSeek-V4-Flash FP4 (LowLatency recipe).

Launches TP=4 with flashinfer_mxfp4 MoE runner + EAGLE speculative decoding.
Runs 12 ServerSanity probes plus a GSM8K accuracy gate.

Registry: stage-c-test-dsv4-4-gpu-b200 (per-commit, 4x B200)
"""

register_cuda_ci(est_time=1800, suite="stage-c-test-dsv4-4-gpu-b200")

class TestDSV4FlashFP4B200(ServerSanityMixin, CustomTestCase):
    """LowLatency recipe: TP=4, FP4 (mxfp4), EAGLE spec decoding."""
    @classmethod
    def setUpClass(cls):
        cls.model = try_cached_model(MODEL)
        cls.base_url = DEFAULT_URL_FOR_TEST
        cls.process = popen_launch_server(
            cls.model, cls.base_url, timeout=SERVER_LAUNCH_TIMEOUT,
            other_args=[
                "--trust-remote-code",
                "--tp", "4",
```

```
                "--moe-runner-backend", "flashinfer_mxfp4",
                "--speculative-algorithm", "EAGLE",
                "--speculative-num-steps", "3",
                "--speculative-eagle-topk", "1",
                "--speculative-num-draft-tokens", "4",
                "--chunked-prefill-size", "4096",
                "--disable-flashinfer-autotune",
            ],
        )
    # ... tearDownClass, test_gsm8k 从略

class TestDSV4FlashFP4B200MegaMoE(ServerSanityMixin, CustomTestCase):
    """Balanced recipe: TP=4, DP=4, MegaMoE."""
    # ... 使用 _MEGAMOE_ENV 和 DeepEP 配置
```

python/sglang/srt/model_loader/weight_utils.py

新增 _drop_file_cache_after_load 函数，支持权重加载后释放文件缓存，优化 RL 训练内存。

```
import os

def _drop_file_cache_after_load(path: str) -> None:
    """释放权重文件页面缓存，避免 CPU OOM（用于 RL 训练时）。

    使用 posix_fadvise() 告知内核不再需要文件的页面，使其可被回收。
    """
    # 获取 posix_fadvise 和 DONTNEED 常量，仅在 Linux 上有效
    posix_fadvise = getattr(os, "posix_fadvise", None)
    dontneed = getattr(os, "POSIX_FADV_DONTNEED", None)
    if posix_fadvise is None or dontneed is None:
        return
    fd = None
    try:
        fd = os.open(path, os.O_RDONLY)
        # fd, offset=0, len=0 => 建议整个文件
        posix_fadvise(fd, 0, 0, dontneed)
    except OSError as e:
        logger.debug("Failed to drop file cache for %s: %s", path, e)
    finally:
        if fd is not None:
            os.close(fd)

# 在 safetensors_weights_iterator 中调用
def safetensors_weights_iterator(
    hf_weights_files: List[str],
    disable_mmap: bool = False,
    prefetch: bool = False,
    prefetch_num_threads: int = 4,
    drop_cache_after_load: bool = False, # 新增参数
) -> Generator[Tuple[str, torch.Tensor], None, None]:
```

```
    # ...
    for st_file in tqdm(sorted_files, ...):
        # 加载权重
        with safetensors.safe_open(st_file, framework="pt", device="cpu") as f:
            for name in f.keys():
                yield name, f.get_tensor(name)
        # 在所有 tensor 都 yield 后释放页面缓存
        if drop_cache_after_load:
            _drop_file_cache_after_load(st_file)
```

评论区精华

来自 gemini-code-assist[bot] 的 review 评论：

- 建议在 multi_thread_safetensors_weights_iterator 中使用 safetensors.torch.load_file 替代手动迭代，以提高效率和代码简洁性。
- 指出 TestDSV4FlashFP8H200 类中文档字符串、print 语句和模型变量不一致（类名宣称 FP8 但描述为 FP4，且使用 MODEL 而非 MODEL_FP8）。
- 建议在 ci_install_dsv4_dep.sh 中使用 pip install . 替代 python3 setup.py install。 这些评论未在 PR 中得到明确答复或解决。
- 使用 safetensors.torch.load_file 优化效率 (performance): 未得到作者回复或修改，当前实现保留了 safe_open 方式，但返回值改为了 (st_file, result)。
- 测试类名与实际配置不一致 (testing): 未修正，可能影响测试准确性。
- 使用 pip install 替代 setup.py install (infra): 未采纳，脚本中仍可能使用了 setup.py install（具体实现未验证）。

风险与影响

- 风险：
  1. 测试文件迁移可能导致旧 CI 套件（nightly-4-gpu-b200）失效，需确保注册正确。
  2. 自闭合标签解析可能影响现有 V3.2 函数调用解析的兼容性，需验证。
  3. 权重缓存释放功能依赖于 posix_fadvise，在不支持的系统上静默失效，用户可能误以为生效。
  4. multi_thread_safetensors_weights_iterator 返回值从 state_dict 改为 (st_file, state_dict)，所有调用者需相应更新（已在 loader.py 中兼容）。
  5. new --weight-loader-drop-cache-after-load 参数默认关闭，需要用户显式启用，可能未被广泛采用。
- 影响：对用户的影响：
  - DSV4 测试从 nightly 移至 per-commit，问题能在更早阶段暴露。
  - RL 训练用户可使用 --weight-loader-drop-cache-after-load 减少 CPU 内存占用。
  - 使用 DeepSeek V3.2/V4 函数调用的用户将获得自闭合标签的正确解析。
  - H200 FP8 测试配置更准确。

对系统的影响：

- CI 依赖安装统一化，减少维护成本。
- 权重加载路径增加可选 cache 清理，可能带来微小性能开销。
- 风险标记：测试文件迁移可能遗漏注册，自闭合标签解析兼容性风险，posix_fadvise 静默降级，返回值改动可能影响调用者

关联脉络

- PR #24802 slash command rerun UX: emoji semantics + result writeback: 为 DSV4 测试提供斜杠命令基础设施，本 PR 的测试中使用了这些命令触发 CI。
- PR #23915 Fix dsv4 self-closing invoke tags + accept reasoning_effort=max: 本 PR cherry-pick 的第一个 commit 来自此 PR，修复自闭合标签和支持 reasoning_effort=max。