PR #23273 完整报告

sgl-project/sglang

[NVIDIA] [GDN] Enable FlashInfer MTP verify on SM100+ (Blackwell)

合并时间：2026-06-02 09:56

原文链接：http://prhub.com.cn/sgl-project/sglang/pull/23273

报告：启用 FlashInfer GDN MTP 验证于 SM100+ (Blackwell)

执行摘要

本 PR 在 SM100+（Blackwell）GPU 上启用了 FlashInfer GDN 的 MTP 验证，之前该路径抛出 NotImplementedError。通过导入 bf16 状态 MTP kernel 并移除配置 guard，FlashInfer 现在可以成为 SM100+ 上的默认 MTP 后端。基准测试显示吞吐量和 TPOT 略优于 Triton，准确率（GSM8K、GPQA）达标。变更涉及 4 个文件，核心改动约 15 行，配套测试增加了 FlashInfer 专用评估类。

功能与动机

> "Enables FlashInfer GDN MTP (speculative decoding) verify on SM100+ (Blackwell) hardware, previously raising NotImplementedError. SM90 (Hopper) MTP was already supported; this PR completes SM100+ coverage."

之前 SM100+ 用户无法使用 FlashInfer 进行 MTP 验证，必须回退到 Triton。完成 SM100+ 覆盖后，用户可以在 Blackwell 硬件上获得统一且略有提升的推测解码性能。

实现拆解

1. 导入 bf16 MTP kernel（gdn_flashinfer.py）：在 _get_flashinfer_gdn_kernels 中新增从 flashinfer.gdn_kernels.gdn_decode_bf16_state 导入 gated_delta_rule_mtp 并重命名为 gated_delta_rule_mtp_bf16，在返回元组中暴露该函数。
2. 添加 bf16 适配器（gdn_flashinfer.py）：新增内部函数 _mtp_bf16_adapted，将 FlashInfer bf16 状态 MTP kernel 包装成与现有 verify 接口兼容的形式（处理 intermediate_states_buffer 的切片和 A_log 的 float 转换）。target_verify 根据 ssm_states.dtype 选择调用 fp32 路径或 bf16 适配路径。
3. 移除配置 guard（server_args.py）：在 _handle_linear_attn_backend 中删除 and self.speculative_algorithm is None 条件，使 SM100+ 在启用任意推测算法（包括 MTP）时都能自动默认 linear_attn_decode_backend='flashinfer'。
4. 更新后端路由注释（gdn_backend.py）：修正 verify_kernel 选择逻辑的注释，指出 SM100+ 现在可以使用 FlashInfer 进行 MTP 验证（原来错误地声称不支持）。
5. 测试配套（test_qwen35_fp4_mtp.py）：
   - 将重复的启动参数抽取为 MTP_BASE_ARGS 常量。
   - 提取 _run_mtp_gsm8k 工具函数，复用评估逻辑。

- 新增 TestQwen35FP4MTPFlashInfer 类，通过 --linear-attn-decode-backend flashinfer --enforce-disable-flashinfer-allreduce-fusion 启动服务器并运行 gsm8k 评估。
- 原有 TestQwen35FP4MTP 类保持不变，仍使用 Triton 后端。
- 测试注册时间从 340s 调整为 740s，以适应两个测试类。

python/sglang/srt/layers/attention/linear/kernels/gdn_flashinfer.py

核心变更文件：导入 bf16 状态 MTP kernel，新增 _mtp_bf16_adapted 适配器函数，统一 SM90 和 SM100+ 的 verify 路径。

```
def _get_flashinfer_gdn_kernels():
    """Lazy import for FlashInfer GDN prefill, decode and verify (MTP) kernels.

    Returns (available, prefill_fn, mtp_fn, decode_fn, mtp_bf16_fn).
    """
    global _flashinfer_gdn_available, _flashinfer_chunk_gated_delta_rule, _flashinfer_gated_delta_
    rule_mtp, _flashinfer_gated_delta_rule_decode, _flashinfer_gated_delta_rule_mtp_bf16
    if _flashinfer_gdn_available is None:
        try:
            os.environ.setdefault("FLASHINFER_DISABLE_VERSION_CHECK", "1")

            from flashinfer.gdn_decode import (
                gated_delta_rule_decode_pretranspose,
                gated_delta_rule_mtp,
            )
            from flashinfer.gdn_kernels.gdn_decode_bf16_state import (
                gated_delta_rule_mtp as gated_delta_rule_mtp_bf16, # 新增：导入 bf16 状态 MTP
                kernel
            )
            from flashinfer.gdn_prefill import chunk_gated_delta_rule

            _flashinfer_chunk_gated_delta_rule = chunk_gated_delta_rule
            _flashinfer_gated_delta_rule_mtp = gated_delta_rule_mtp
            _flashinfer_gated_delta_rule_mtp_bf16 = gated_delta_rule_mtp_bf16 # 新增：保存 bf16
            版本函数句柄
            _flashinfer_gated_delta_rule_decode = gated_delta_rule_decode_pretranspose
            _flashinfer_gdn_available = (
                torch.cuda.is_available() and torch.cuda.get_device_capability()[0] >= 9
            )
            if _flashinfer_gdn_available:
                logger.info("FlashInfer GDN kernels loaded successfully")
        except (ImportError, RuntimeError) as e:
            logger.warning(f"FlashInfer GDN kernels not available: {e}")
            _flashinfer_gdn_available = False
            _flashinfer_gated_delta_rule_decode = None
    return (
        _flashinfer_gdn_available,
        _flashinfer_chunk_gated_delta_rule,
```

```
        _flashinfer_gated_delta_rule_mtp,
        _flashinfer_gated_delta_rule_decode,
        _flashinfer_gated_delta_rule_mtp_bf16, # 新增：在返回元组中提供 bf16 版本
    )
```

test/registered/models_e2e/test_qwen35_fp4_mtp.py

测试覆盖：新增 TestQwen35FP4MTPFlashInfer 类验证 FlashInfer 后端下 MTP 的 gsm8k 准确率，同时抽取公共参数和工具函数降低重复。

```
def _run_mtp_gsm8k(test_case):
    """工具函数：启动 GSM8K 评估并验证准确率与推测接受长度。"""
    args = SimpleNamespace(
        model=test_case.model,
        eval_name="gsm8k",
        num_shots=5,
        num_examples=200,
        max_tokens=16000,
        num_threads=128,
        repeat=1,
        temperature=0.6,
        top_p=0.95,
        top_k=20,
        base_url=test_case.base_url,
        host="http://127.0.0.1",
        port=int(test_case.base_url.split(":")[-1]),
    )
    metrics = run_eval(args)
    print(f"{metrics=}")
    test_case.assertGreaterEqual(
        metrics["score"], ACC_THRESHOLDS[test_case.model]["gsm8k"]
    )

    server_info = requests.get(test_case.base_url + "/server_info")
    avg_spec_accept_length = server_info.json()["internal_states"][0][
        "avg_spec_accept_length"
    ]
    print(f"{avg_spec_accept_length=}")
    test_case.assertGreater(avg_spec_accept_length, 3.3)


class TestQwen35FP4MTPFlashInfer(ReasoningTokenUsageMixin, CustomTestCase):
    """验证 FlashInfer 后端下的 MTP 推理准确率（GSM8K）。"""
    reasoning_parser_name = "qwen3"

    @classmethod
    def setUpClass(cls):
        cls.model = QWEN35_FP4_MODEL
        cls.base_url = DEFAULT_URL_FOR_TEST
        cls.init_reasoning_token_verifier()
```

```
        envs.SGLANG_ENABLE_SPEC_V2.set(True)
        cls.process = popen_launch_server(
            cls.model,
            cls.base_url,
            timeout=DEFAULT_TIMEOUT_FOR_SERVER_LAUNCH,
            other_args=MTP_BASE_ARGS
            + [
                "--linear-attn-decode-backend",
                "flashinfer", # 指定 FlashInfer 后端
                "--enforce-disable-flashinfer-allreduce-fusion", # 避免融合引入干扰
            ],
        )

    @classmethod
    def tearDownClass(cls):
        envs.SGLANG_ENABLE_SPEC_V2.set(False)
        kill_process_tree(cls.process.pid)

    def test_gsm8k(self):
        _run_mtp_gsm8k(self)
```

评论区精华

> Fridge003: “Can we add a test for this usage. Maybe under test/registered/4-gpu-models/test_qwen35_fp4_flashinfer.py” YAMY1234: “Added under test/registered/4-gpu-models/test_qwen35_models.py, thanks!”（实际添加在 test_qwen35_fp4_mtp.py 中）

审查者关心新功能的测试覆盖，作者快速响应并添加了专门的 FlashInfer MTP 测试类。

风险与影响

- 风险：需要 FlashInfer >= 0.6.7；bf16 适配路径依赖上游 bug 修复（flashinfer#3147）；测试仅覆盖 gsm8k 单配置，不覆盖 topk>1 等场景。
- 影响：SM100+ 用户无需手动干预即可获得 FlashInfer MTP 加速；团队需维持两条 MTP 后端，但代码复用度高；性能提升约 1-5%，无显著退化。

关联脉络

该 PR 完成了 GDN MTP 在 Blackwell 上的最后一环，与以下内容关联：

- 上游 FlashInfer PR #2810（padding index guard）和 #3147（OOB crash fix）是功能正确性的基础。
- 关联 Issue #2679 和 #2810 跟踪了 bf16 状态 MTP kernel 的设计与 padding 修复。
- 同仓库近期 PR #26866（Support spec v2 tree drafting）和 #26424（topk=1 fastpath）都属于 speculative decoding 的持续优化链条。