PR #34492 完整报告

sgl-project/sglang

XPU: remove SGLANG_USE_SGL_XPU flag

合并时间：2026-08-27 17:38

原文链接：http://prhub.com.cn/sgl-project/sglang/pull/34492

执行摘要

- 一句话：移除 XPU 开关，默认启用 sgl-kernel MoE，可参数回退
- 推荐动作：值得精读。关注三个设计决策：一是「环境变量开关 → server args 显式后端」的配置收敛模式，可复用到其他平台的类似开关清理；二是双路径并存的分支写法 is_xpu() and not get_moe_runner_backend().is_triton()，在保持默认高性能的同时保留逃生通道；三是参数化测试对后端优先级矩阵的覆盖方式（server 参数优先于 layer 参数），可作同类后端选择测试的参考模板。

功能与动机

PR body 明确说明："Remove the SGLANG_USE_SGL_XPU flag so that the high-performance kernels (MoE) from sgl-kernel are used by default. If one wants to fall back to the triton kernel for MoE, use the server argument --moe-runner-backend triton." 评审者 alexnails 进一步推动方案升级："why are we keeping this and not just not using is_xpu full time + server args where triton and other applicable intel backends apply?" 作者据此将「flag 默认 true」改为「彻底删除 flag」，让后端选择职责完全收敛到平台探测与 server 参数。

实现拆解

变更以删除 use_intel_xpu_backend() 为入口，分五步完成：

1. 删除开关定义：python/sglang/srt/utils/common.py 中删除 use_intel_xpu_backend() 函数（原逻辑为 get_bool_env_var("SGLANG_USE_SGL_XPU") and is_xpu()），所有调用点改为「is_xpu() + get_moe_runner_backend().is_triton()」组合判断，把是否启用 sgl-kernel 的决定权从环境变量交给 --moe-runner-backend server 参数。
2. MoE 主路径切换：在 fused_moe.py、unquant.py 的 forward_xpu、fp8.py 的 apply 中，原先依赖 _use_sgl_xpu 或 use_intel_xpu_backend() 的分支全部改写为 is_xpu() and not get_moe_runner_backend().is_triton()，XPU 默认走 sgl_kernel.fused_experts，Triton 成为显式回退路径；同时 moe/utils.py 新增 MoeRunnerBackend.INTEL_XPU = "intel_xpu" 枚举成员与 is_intel_xpu() 方法，为显式选择 XPU 内核预置配置项。server_args.py 有 1 行新增（材料未展示具体内容，推测与 intel_xpu 后端枚举或帮助文本相关，属不确定性）。

3. 权重布局适配：unquant.py 的 _use_xpu_moe_ld_padding(use_triton_kernels) 在 is_xpu() 之外追加 not get_moe_runner_backend().is_triton()，确保只有走 sgl-kernel XPU 路径时才给 expert 权重做 L3 行步长填充（_empty_xpu_moe_expert_weight），Triton 路径（含 --moe-runner-backend triton 与 triton_kernels 构建）保持原布局，避免对非 XPU 权重做无意义填充。
4. 视觉注意力默认后端：vision.py 的 _determine_attention_backend 在 XPU 分支从「默认 triton_attn、仅 flag 开启时 xpu_attn」改为默认 xpu_attn，server 参数或 layer 参数仍可覆盖；新增 test_vision_backend_selection.py 参数化测试，覆盖 4 种优先级组合（server 参数优先于 layer 参数）。
5. 测试与 CI 配套：test_moe_ld_padding.py 适配新判断条件；三个 XPU LLM 模型测试（test_xpu_qwen3_30b_a3b.py、test_xpu_qwen3_5_35b_a3b.py、test_xpu_gemma_4_26b_a4b.py）删除 SGLANG_USE_SGL_XPU 环境变量设置。整个提交链经历了「flag 默认 true → 修 CI → alexnails 质疑后彻底删除 → 补视觉后端测试」的演进过程。

关键文件：

- python/sglang/srt/layers/quantization/unquant.py（模块 量化层；类别 source；类型 core-logic；符号 _use_xpu_moe_ld_padding, forward_xpu）：核心权重布局逻辑所在：_use_xpu_moe_ld_padding 由环境变量判断改为 is_xpu() + get_moe_runner_backend().is_triton() 组合判断，forward_xpu 默认切到 sgl-kernel 路径，是本次行为变更的关键源码文件。
- python/sglang/srt/layers/moe/moe_runner/triton_utils/fused_moe.py（模块 MoE 内核；类别 source；类型 dependency-wiring；符号 fused_moe）：Triton MoE 内核的入口文件，模块级 _use_sgl_xpu 被移除，fused_moe 的 XPU 分支改为运行时判断，是默认切换到 sgl-kernel 的主要调度点。
- python/sglang/srt/layers/moe/utils.py（模块 MoE 配置；类别 source；类型 configuration；符号 is_intel_xpu）：新增 MoeRunnerBackend.INTEL_XPU 枚举成员与 is_intel_xpu() 方法，为显式选择 Intel XPU MoE 后端预置配置项。
- python/sglang/srt/utils/common.py（模块 通用工具；类别 source；类型 configuration；符号 use_intel_xpu_backend）：删除 use_intel_xpu_backend() 函数，环境变量开关的源头被移除，所有调用点依赖此变更。
- python/sglang/srt/layers/quantization/fp8.py（模块 FP8 量化；类别 source；类型 core-logic）：FP8 MoE 路径同样由环境变量开关切换为 is_xpu() + moe runner 后端判断，保证量化模型走一致的默认内核。
- python/sglang/srt/layers/attention/vision.py（模块 视觉后端；类别 source；类型 core-logic；符号 _determine_attention_backend）：XPU 视觉注意力默认后端从 triton_attn 改为 xpu_attn，是评审讨论聚焦点，也是新增参数化测试覆盖的对象。
- python/sglang/srt/server_args.py（模块 服务配置；类别 source；类型 configuration）：服务参数解析层为后端选择提供配置入口，新增 1 行（具体内容未在材料中展示，推测与 moe runner 后端枚举或帮助文本相关）。
- test/registered/xpu/test_vision_backend_selection.py（模块 XPU 测试；类别 test；类型 test-coverage；符号 test_xpu_backend_selection_priority）：新增参数化测试，验证

XPU 视觉注意力后端选择优先级（server 参数优先于 layer 参数），直接回应该变更的评审疑问。

- test/registered/xpu/test_moe_ld_padding.py（模块 XPU 测试；类别 test；类型 test-coverage）：权重行步长填充测试适配新的后端判断条件，验证 Triton 与 sgl-kernel 路径下的填充差异。
- test/registered/xpu/llm_models/test_xpu_qwen3_30b_a3b.py（模块 XPU 测试；类别 test；类型 test-coverage）：XPU LLM 模型测试移除 SGLANG_USE_SGL_XPU 环境变量设置，说明默认路径已切换。

关键符号：_use_xpu_moe_ld_padding, forward_xpu, fused_moe, use_intel_xpu_backend, is_intel_xpu, _determine_attention_backend, test_xpu_backend_selection_priority

关键源码片段

python/sglang/srt/layers/moe/moe_runner/triton_utils/fused_moe.py

Triton MoE 内核的入口文件，模块级 _use_sgl_xpu 被移除，fused_moe 的 XPU 分支改为运行时判断，是默认切换到 sgl-kernel 的主要调度点。

```
# 平台相关内核导入：XPU 平台直接依赖 sgl_kernel 的 moe_sum_reduce 与
# silu_and_mul，这一依赖在改动前后都保持；变化在于 MoE 主路径的分支条件。
if _is_cuda:
    from sgl_kernel import moe_sum_reduce
    from sglang.kernels.ops.activation.activation import gelu_and_mul, silu_and_mul
elif _is_cpu and _is_cpu_amx_available:
    pass
elif _is_hip:
    from sgl_kernel import gelu_and_mul, silu_and_mul
    # ... aiter 相关逻辑 ...
elif _is_xpu:
    from sgl_kernel import moe_sum_reduce, silu_and_mul
elif _is_musa:
    from sgl_kernel import moe_sum_reduce
    _silu_and_mul_musa = torch.nn.SwishGLU()


def fused_moe(
    hidden_states,
    w1,
    w2,
    topk_weight,
    topk_ids,
    # ... 其余参数：b1、b2、各 scale、block_shape、gemm 参数等 ...
):
    """XPU 上默认走 sgl-kernel 的 fused_experts，除非显式选择 triton 后端。

    原先由模块级变量 _use_sgl_xpu（来自 SGLANG_USE_SGL_XPU 环境变量）决定，
    现在改为运行时判断：is_xpu() 且当前 moe runner 不是 triton 时，直接
    调用 sgl_kernel.fused_experts，并把 triton 路径作为显式回退，从而
```

```
    默认获得高性能内核，同时保留 --moe-runner-backend triton 逃生通道。
    """
    if _is_xpu and not get_moe_runner_backend().is_triton():
        topk_weight, topk_ids, _ = topk_output
        from sgl_kernel import fused_experts as sgl_fused_experts

        return sgl_fused_experts(
            hidden_states,
            w1,
            w2,
            topk_weight,
            topk_ids,
            b1=b1,
            b2=b2,
            use_fp8_w8a8=use_fp8_w8a8,
            w1_scale=w1_scale,
            w2_scale=w2_scale,
            # ... 其余参数透传 ...
        )
    # ... 其余平台的 triton / 其他内核路径 ...
```

test/registered/xpu/test_vision_backend_selection.py

新增参数化测试，验证 XPU 视觉注意力后端选择优先级（server 参数优先于 layer 参数），直接回应该变更的评审疑问。

```
"""参数化验证 XPU 视觉注意力后端的选择优先级。

覆盖四种组合：未指定任何后端时默认 xpu_attn；server 参数显式指定
xpu_attn 或 triton_attn 时以其为准；layer 传入的 passed_backend 不能
覆盖 server 配置（xpu_attn + triton_attn 组合仍选 xpu_attn）。
"""
import sys
from types import SimpleNamespace

import pytest

from sglang.srt.layers.attention import vision
from sglang.test.ci.ci_register import register_xpu_ci

register_xpu_ci(est_time=300, suite="stage-a-test-1-gpu-xpu")


@pytest.mark.parametrize(
    ("server_backend", "passed_backend", "expected"),
    [
        (None, None, "xpu_attn"), # server 未设置，默认 xpu_attn
        ("xpu_attn", None, "xpu_attn"),
        ("xpu_attn", "triton_attn", "xpu_attn"), # server 配置优先于 layer 参数
        ("triton_attn", None, "triton_attn"), # 显式 triton 可覆盖默认
```

```
    ],
)
def test_xpu_backend_selection_priority(
    monkeypatch,
    server_backend, # 来自 server 参数
    passed_backend, # 来自 layer 参数
    expected,
):
    # 用 SimpleNamespace 模拟全局 multimodal manager 配置
    monkeypatch.setattr(
        vision,
        "get_mm",
        lambda: SimpleNamespace(mm_attention_backend=server_backend),
    )

    backend = vision.VisionAttention._determine_attention_backend(None, passed_backend)

    assert backend == expected


if __name__ == "__main__":
    sys.exit(pytest.main([__file__]))
```

评论区精华

- 开关去留的关键转向（alexnails）：*"why are we keeping this and not just not using is_xpu full time + server args where triton and other applicable intel backends apply?"* 该评论直接把方案从「flag 默认开启」推向「彻底删除 flag」，作者随即提交 *"Remove SGLANG_USE_SGL_XPU"*，并回复 *"Good question. I have updated the PR to remove this flag entirely. It requires much more changes in the code."*
- vision.py 后端优先级确认（alexnails → Xia-Weiwen）：*"do we just assume triton_attn will be serverargs override? (read docstring, just double checking)"*；作者回应 *"I think triton_attn will override xpu_attn if one does not specify a backend or specify triton. I will try with DeepSeek-OCR-2 to confirm."*，随后确认 *"DeepSeek-OCR-2 does not use this VisionAttention. I have added a test case for this."*，即新增的参数化测试 test_xpu_backend_selection_priority。
- CI 失败定性（mingfeima → Xia-Weiwen）：mingfeima 先认为失败像是 flaky 并触发 rerun，作者判断 *"The failure seems real. I will fix it."*，随后修复真实失败。
- 是否保留 SGLANG_USE_SGL_XPU 开关 (design): 方案从「flag 默认 true」升级为「彻底删除 flag」，所有分支改为 is_xpu() + get_moe_runner_backend().is_triton() 判断。
- XPU 视觉注意力后端选择优先级 (question): 默认 xpu_attn，server 参数优先于 layer 参数；新增 test_xpu_backend_selection_priority 覆盖 4 种组合。
- CI 失败是否为 flaky (testing): 修复真实 CI 失败后合并。

风险与影响

- 风险：
  - 默认内核路径切换：XPU 上 MoE 默认从 Triton 切到 sgl_kernel.fused_experts，如果用户安装的 sgl-kernel 未包含 XPU 构建或版本不一致，默认路径可能直接 import 失败或产生数值差异；PR 的 Accuracy/Speed Tests 章节为空，缺少实测数据支撑。
- 判断依赖解析时机：_use_xpu_moe_ld_padding() 在权重创建阶段调用，现在依赖 get_moe_runner_backend()（server args）。若在 server args 解析完成前触发 create_weights，后端判断可能不准确，进而影响权重布局一致性。
- 视觉后端默认变更：vision.py 中 XPU 视觉注意力默认从 triton_attn 变为 xpu_attn，影响所有经 VisionAttention 的多模态模型（labels 含 Multi-modal）；虽然新增测试覆盖优先级矩阵，但 xpu_attn 的成熟度与精度未见 benchmark。
- 环境变量删除兼容性：用户脚本或部署配置中若仍设置 SGLANG_USE_SGL_XPU=0/1，该变量会静默失效（被忽略），没有告警或迁移提示，可能造成用户对实际生效后端产生误解。
- 影响：
  - 用户影响：XPU 部署用户默认获得 sgl-kernel MoE 高性能路径，Triton 回退改为参数化；多模态模型在 XPU 上默认视觉注意力后端切换为 xpu_attn。
- 系统影响：后端选择逻辑由「环境变量 + 平台探测」收敛为「平台探测 + server args」，职责更清晰，配置面更小；涉及量化（unquant/fp8）、MoE 调度（fused_moe/utils）、注意力（vision）、服务配置（server_args）多个子系统，但分支影响基本限定在 XPU 平台内。
- 团队影响：Intel/XPU 维护线（Xia-Weiwen、airMeng、mingfeima）后续需确保 sgl-kernel 的 XPU 构建随默认路径发布；CI 中 XPU suite 的默认路径覆盖增强，run-ci 标签被触发验证。
- 影响程度：共 19 个文件、+96/-46，属于中等规模的跨模块行为变更，主要影响 XPU 平台用户。
- 风险标记：默认内核路径切换，跨模块联动，环境变量删除兼容性，视觉后端默认变更，缺少精度性能实测

关联脉络

- PR #36360 [Intel XPU] Fix cross-encoder rerank hang on B580 runners: 同为 Intel XPU 维护线，涉及 XPU 运行时行为修复与 XPU 测试。
- PR #35222 [CPU] Enable ERNIE models on CPU: Intel 后端与 MoE topk 配置相关，与本 PR 的 MoE 内核选择逻辑同属 Intel 后端演进。
- PR #36309 [AMD][Bugfix] Skip invalid fused MoE reduction for direct top-1 output: 修改了相同的 fused_moe.py 文件，属于 MoE 内核分支的并行维护。
- PR #36430 [CPU] Fix truncated KV prefix in intel_amx spec verify: Intel 后端注意力内核修复，与视觉注意力默认后端变更同属 Intel 内核路径收敛。