SiKernel算子接入框架验证流程

1 概述

目前sikernel repo中的算子，主要服务于大模型推理场景，需要接入推理引擎中以发挥作用。框架团队提供了sikernel算子接入的基本流程，对于新开发算子，可以按照相关流程进行集成与验证。

框架团队已提供了2种常用推理引擎（sglang，vllm）所需算子的适配接入流程，下面分别进行简要介绍。

2 sglang

sglang推理引擎所需算子的接入验证repo地址为：

https://gitlabsoft.siorigin.com/algo/framework/sgl-kernel-sipu

以DSA系列算子的新需求为例，进行接入与验证的流程介绍。

2.1 环境准备

可参考以下执行命令：

代码块

```
# 获取用于验证的repo
git clone git@gitlabsoft.siorigin.com:algo/framework/sgl-kernel-sipu.git

# 切换至适配sikernel算子的开发分支
cd sgl-kernel-sipu
git checkout ds-v32-dev

# submodule更新
git submodule init
git submodule update --init sikernel
cd sikernel
git submodule init
git submodule update

# 获取用于验证跑测算子的docker image
# 如无docker，需要先进行安装，可参考如下：
# sudo apt update
```

```
# sudo apt install -y docker.io
# docker安装成功后，可pull docker image
docker pull harbor.siorigin.com/sglang-sipu/release:v0.5.1-v0.1.3
```

2.2 代码开发

首先需要根据需求，在sikernel repo中开发sglang所需要的SIPU算子，自测验证通过后提交至主分支（dev）。然后在sgl-kernel-sipu repo中，将sikernel的submodule切换（更新）至已经开发完成指定算子的commit（一般切换至dev，并pull至最新即可）。

sikernel算子开发完成后，还需要在sgl-kernel-sipu repo中做集成，主要修改内容包括：

代码块

```
# 修改相关代码和接口实现，将sikernel中的API对接至相关的逻辑中
csrc/elementwise/*.cpp

# 将新增待验证的kernel path添加至list底部
sgl-sikernel-integration/release.list
```

2.3 验证测试

使用docker进行跑测验证：

代码块

```
# 启动docker
docker run -it --rm --net=host -v "$(pwd):/sgl-workspace/sgl-kernel-sipu" -v
"/share_data/sicx_sdk/release:/share_data/sicx_sdk/release"
harbor.siorigin.com/sglang-sipu/release:v0.5.1-v0.1.3 /bin/bash

# 进入开发repo并做环境初始化
cd /sgl-workspace/sgl-kernel-sipu
python -m pip install -U pytest-xdist pytest-timeout
source setup.sh

# 编译sikernel开发依赖
make install-kernels
make install-dev

# 测试开发适配的sikernel算子，以hadamard transform为例
```

```
python -m pytest tests/test_hadamard.py -v -n4 --timeout=240
```

其余单个算子的测试命令可参考：

https://gitlabsoft.siorigin.com/algo/framework/sgl-kernel-sipu/-/tree/ds-v32-dev/sgl-sikernel-integration?ref_type=heads#sikernel-%E7%AE%97%E5%AD%90-1

验证通过后，可直接将改动push到ds-v32-dev分支中，并同步给框架侧同事。

3 vllm

参考文档：vllm编译与算子接入流程

3.1 环境准备

可参考以下执行命令：

代码块

```
# 获取用于验证的repo
git clone git@gitlabsoft.siorigin.com:algo/framework/vllm-sipu.git

# 新建自己的分支，算子接入与代码测试在自己的分支上进行
cd vllm-sipu && git checkout -b <your_name/branch_name>

# docker安装部分可参考2.1
# 拉取镜像
docker pull harbor.siorigin.com/vllm-sipu/vllm-sipu-dev:latest
```

3.2 代码开发

首先需要根据需求，在sikernel repo中开发vllm所需要的SIPU算子，自测验证通过后提交至主分支（dev）。然后在vllm-sipu repo中，将sikernel的submodule切换（更新）至已经开发完成指定算子的commit（一般切换至dev，并pull至最新即可）。

sikernel算子开发完成后，还需要在vllm-sipu repo中做集成，主要修改内容包括：

代码块

```
# 修改相关代码和接口实现，将sikernel中的API对接至相关的逻辑中
csrc/sipu/sikernel/*.cpp

# 将新增待验证的kernel path添加至yaml底部
```

```
5   csrc/op_list.yaml
```

3.3 验证测试

使用docker进行跑测验证：

代码块

```
1   # 启动docker
2   docker run -it --name <container_name> \
3       --network host \
4       -w /workspace \
5       -e VLLM_SIPU_SIKERNEL_SRC_DIR=/workspace/sikernel \
6       -v $(pwd)/vllm-sipu:/workspace/vllm-sipu \
7       -v $(pwd)/sikernel:/workspace/sikernel \
8       -v /share_data:/share_data:rshared \
9       -v "$HOME/.ssh:/root/.ssh:ro" \
10      harbor.siorigin.com/vllm-sipu/vllm-sipu-dev:latest /bin/bash
11
12  # 只需要编译一次，后续sikernel或者vllm-sipu的代码改动都不需要重新执行编译
13  # vllm-sipu的构建后端会在运行时自动检测代码变动并自动完成编译更新
14  cd /workspace/vllm-sipu && \
15  pip install -e . --no-build-isolation -v --no-deps
16
17  # docker中执行
18  pip install -r requirements/test.txt -i https://mirrors.aliyun.com/pypi/simple
19
20  # 在vllm-sipu直接执行相应的case
21  python -m pytest xxxxxx[test case] -s -vvv --maxfail=1
```

其余单个算子的测试命令可参考：

https://gitlabsoft.siorigin.com/algo/framework/vllm-sipu#sikernel-operator-test-commands

3.4 MR流程

代码块

```
1   # 在docker外执行
2   cd /path/to/vllm-sipu && \
3   pip install -r requirements/lint.txt && \
4   pre-commit install
```

```
5
6   # pre-commit会自动执行代码风格检查，静态类型检查，commit信息检查
7   git add && git commit xxxx
```

在gitlab界面创建MR，评论 test ci 触发CI测试，验证通过后，可进行merge，并同步给框架侧同事。